(12) Unexamined Patent Application Publication (A)

(19) Japan Patent Office (JP)

(11) Patent Application Publication No. H7-250328

(43) Publication Date: 1995.9.26

(51) Int. Cl.[6]	Ident. Code	Internal Ref. No.	FI	Technical Indication
H04N 7/32				
			H04N 7/137	Z

Examination Request: Not Made No. of Claims: 28 FD (103 Pages Total)

(21) Filing No. H6-321525

(22) Filing Date 1994.11.29

(31) Priority No. H6-5432

(32) Priority Date 1994.1.21

(33) Priority Country Japan (JP)

(71) Applicant 000006013
Mitsubishi Electric Corp.
2-2-3 Marunouchi, Chiyoda-ku, Tokyo

(72) Inventor Kazuya Ishihara
c/o Mitsubishi Electric Corp. System LSI Development Research Laboratory
4-1 Mizuhara, Itami-shi, Hyōgo-ken

(72) Inventor Shin’ichi Uramoto
c/o Mitsubishi Electric Corp. System LSI Development Research Laboratory
4-1 Mizuhara, Itami-shi, Hyōgo-ken

(74) Agent Hisao Fukami (and Three Others)

Continued on Final Page

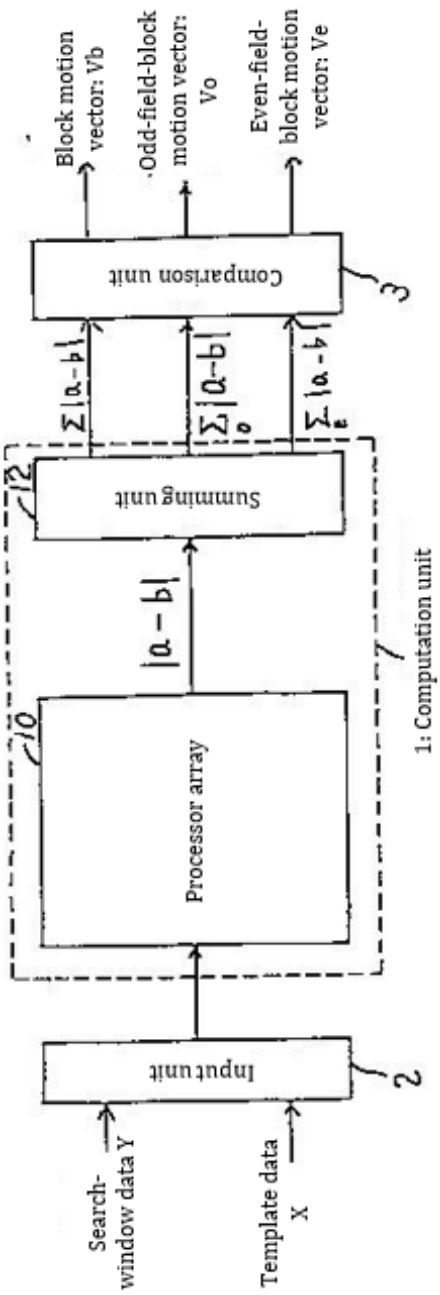
(54) [Title] Moving-Vector Detection Device

GOOGLE EXHIBIT 1008

(57) [Abstract]

[Object] To provide a motion-vector detection device that can quickly detect motion vectors used in moving-image predictive compensation according to a plurality of prediction modes at a small hardware volume.

[Configuration] A processor array 10 includes element processors that are arranged in a matrix corresponding to each pixel of a template block that is a current-image pixel block. Each stores search-window-block pixel data that is a corresponding reference-image pixel block and seeks evaluation-function-value components with template-block pixel data. A summing unit 12 respectively classifies the evaluation-function-value components imparted from the element processors according to a plurality of prediction modes, sums these according to each classification, and generates evaluation-function values for each prediction mode. A comparison unit 3 determines, according to the evaluation-function values imparted from the summing unit, displacement vectors corresponding to evaluation-function values imparting the most favorable similarity as motion vectors for each prediction mode. This enables motion vectors according to a plurality of prediction modes to be simultaneously detected.



[Claims]
[Claim 1] A motion-vector detection device for seeking motion vectors used in motion-compensating predictive coding by block-matching processing between a current screen image and a reference screen image, comprising:
an evaluation means for indicating degrees of similarity between a current-image block of a predetermined size that is a target of motion-vector detection in the current screen and each of a plurality of reference-image blocks in a region associated with the current-image block in the reference screen and seeking in parallel evaluation-function values determined according to each of a plurality of preestablished prediction modes for each reference block; and a motion-vector determination means of receiving the evaluation-function values of each reference block and each of the plurality of prediction modes from the evaluation means and determining in parallel motion vectors of each prediction mode for the current-image block.
[Claim 2] The motion-vector detection device of claim 1, wherein the evaluation means includes
a plurality of first computation means that is disposed corresponding to pixels included in the current-image block, includes a storage means of temporarily storing corresponding pixel data in the reference-image blocks and corresponding pixel data in the current-image block, and subjects the pixel data stored in the storage means to computation making up a portion of the evaluation function; and
a second computation means of classifying outputs of the plurality of first computation means in correspondence with each of the plurality of prediction modes and subjecting the outputs of the first computation means to the remainder of the evaluation function in parallel according to each classification.
[Claim 3] The motion-vector detection device of claim 2, wherein the plurality of first computation means is provided with a plurality of processors that is disposed so each transfers the pixel data only along one direction and is disposed in a matrix of rows and columns corresponding to the current-image block.
[Claim 4] The motion-vector detection device of any one of claims 1 to 3, wherein the current screen and the reference screen are configured by a frame including an even field and an odd field, and
the plurality of prediction modes includes a motion prediction mode using pixels included in the odd field in the current-image block, a motion prediction mode using pixels included in the even field, and a motion prediction mode using pixels included in the frame.
[Claim 5] The motion-vector detection device of any one of claims 1 to 3, wherein the current screen and the reference screen are each configured by one field, and
the plurality of prediction modes includes a motion prediction mode that divides the current-image block equally in two along a screen vertical direction and uses pixels in an upper-half of the block divided equally in two, a motion prediction mode that uses pixels in a lower half of the current-image block divided equally in two, and a motion prediction mode that uses all pixels of the current-image block.
[Claim 6] The motion-vector detection device of any one of claims 1 to 5, wherein a second computation means includes
a switching means of realizing any one of three classifications that are a frame prediction mode that divides the current-image block into four blocks—upper and lower blocks divided equally in two in a vertical direction in relation to a frame screen and an even-field

pixel block and an odd-field pixel block—receives in parallel for each divided block outputs of corresponding first computation means, and switches the configuration of the four divided blocks according to a mode designation signal to use pixels in the frame; an odd-field prediction mode that uses the pixels in an odd field and an even-field motion-prediction mode that uses pixels in an even field or a motion prediction mode that uses pixels in a field; and an upper-half motion prediction mode that uses pixels in the upper half of the current-image block and a lower-half motion prediction mode that uses the pixels in the lower half of the current-image block.

[Claim 7] The motion-vector detection device of any one of claims 1 to 6, further comprising: an input conversion means of setting, in response to a format designation signal that designates a format of the screen, a screen forming format of input image data to the format designated by the format designation signal and imparting this to the evaluation means.

[Claim 8] The motion-vector detection device of any one of claims 1 to 7, wherein a plurality of first computation means is disposed so as to essentially configure a one-dimensional array and is provided with a plurality of processors that transmits both input current-image pixel data and reference-image-block pixel data only along one direction, the motion-vector detection device further comprising:

a buffer means that is disposed for every predetermined number of the plurality of processors, stores the reference-image pixel data, sequentially outputs the stored pixel data in order of storage, and has a variable effective stored-pixel count.

[Claim 9] The motion-vector detection device of any one of claims 1 to 8, further comprising: a third computation means of seeking in parallel evaluation-function values according to the plurality of prediction modes for reference-image blocks satisfying predetermined conditions in the associated region; and

a region designation means of designating for each of the plurality of prediction modes a region of reference-image blocks the evaluation means should compute according to an output of the third computation means; wherein the evaluation means calculates the evaluation values in the plurality of prediction modes for the designated region.

[Claim 10] The motion-vector detection device of claim 9, wherein the region designation means designates a reference-image region to be computed for each of the plurality of prediction modes.

[Claim 11] The motion-vector detection device of claim 9, wherein the region designation means designates one reference-image region in common for the plurality of prediction modes.

[Claim 12] The motion-vector detection device of claim 9, wherein the region designation means designates one reference-image region in common for the plurality of prediction modes and includes a means of enabling only computation according to one prediction mode for the evaluation means.

[Claim 13] A motion-vector detection device for seeking motion vectors used in motion-compensating predictive coding processing by block-matching processing between a reference frame image and a current frame image, the frames including an even field and an odd field, comprising:

a computation means that includes a plurality of processors disposed corresponding to each pixel of a current-frame-image block that is a target of motion-vector detection, inputs pixel data of the current-frame-image block and pixel data of a reference-image block in a

region associated with the current-frame-image block in the reference frame image, and performs predetermined computation of the pixel data corresponding to both blocks;
a summing means of classifying outputs of the computation means into computation results for pixels in the even field, computation results for pixels in the odd field, and computation results for pixels in the frame and adding the outputs of the computation means to seek a sum for each classification; and
a motion-vector determination means of determining in a parallel aspect according to outputs of the summing means a motion vector for an odd-field-pixel block of the current-image block, a motion vector for an even-field-pixel block, and a motion vector for a frame-pixel block.

[Claim 14] A motion-vector detection device for seeking motion vectors used in motion-compensating predictive coding processing by block-matching processing between a reference field image and a current field image, comprising:
a computation means that includes a plurality of processors disposed corresponding to each pixel of a current-field-image block that is a target of motion-vector detection in the current field image, inputs pixel data of the current-field-image block and pixel data of a reference-field-image block in a region associated with the current-field-image block in the reference field image, and performs predetermined computation of the pixel data corresponding to both input blocks;
a summing means of dividing outputs of the computation means in two along a vertical direction in the field to divide the current-field-image block into an upper-half block and a lower-half block, classifying the outputs of the computation means into computation results for pixels in the upper-half block and computation results for pixels in the lower-half block and computation results for all pixels in the current-image block, and adding the outputs of the computation means to seek a sum for each classification; and
a motion-vector determination means of determining in a parallel aspect according to outputs of the summing means a motion vector for the upper-half block, a motion vector for the lower-half block and a motion vector for the current-image block.

[Claim 15] A motion-vector detection device for seeking motion vectors used in motion-compensating predictive coding by block-matching processing between a reference screen image and a current screen image, comprising:
a computation means that includes a plurality of processors disposed corresponding to each pixel of a current-image block that is a target of motion-vector detection of the current screen image, inputs pixel data of the current-image block and pixel data of a reference-image block in a region associated with the current-image block in the reference screen image, and performs predetermined computation of the pixel data corresponding to both input blocks, the plurality of processors including—when the screen is configured by a frame—a first processor group that corresponds to pixels in a block in an upper half in a screen vertical direction of the current-image block and is disposed corresponding to pixels in an odd field, a second processor group that corresponds to the pixels in the block in the upper half and is disposed corresponding to pixels in an even field, a third processor group that corresponds to pixels in a block in a lower half in the screen vertical direction of the current-image block and is disposed corresponding to the pixels in the odd field, and a fourth processor group that corresponds to the pixels in the block in the lower half and is disposed corresponding to the pixels in the even field;

a classification means that includes a first combining means of receiving in parallel outputs of the first to fourth processor groups and combining the outputs of the first and second processor groups and combining the outputs of the third and fourth processor groups to generate two sets of output signals and a second combining means of combining the outputs of the first and third processor groups and the outputs of the second and fourth processor groups to generate two sets of output signals and activates one among the first and second combining means in response to a mode designation signal to transmit the two sets of outputs from the activated combining means;
an evaluation-value generation means of adding and summing the two sets of outputs of the classification means according to each set and generating in a parallel aspect evaluation values for each set and an evaluation value for an entirety of the current-image block; and
a motion-vector determination means of determining in a parallel aspect a motion vector of the current-image block for each set and the current-image block according to an output of the evaluation-value generation means.
[Claim 16] A motion-vector detection device for detecting motion vectors used in motion-compensating predictive coding by block-matching processing between a current screen image and a reference screen image, comprising:
a buffer means that includes a means of varying an input order and an output order of image data and temporarily stores input reference-screen-image data; and
a means of calculating a motion vector of a current-image block that is a target of motion-vector detection from the image data read from the buffer means and the current-screen-image data.
[Claim 17] A motion-vector detection device for detecting motion vectors used in motion-compensating predictive coding by block-matching processing between a current screen image and a reference screen image, comprising:
a plurality of processors that is disposed corresponding to pixels of a current-image block that is a target of motion-vector detection, each including a storage means of storing corresponding current-image-block pixel data and pixel data corresponding to a reference-image block in the reference screen image corresponding to the current-image block and a computation means of performing predetermined computation processing on the pixel data stored in the storage means and transmitting the data stored in the storage means only along one direction;
a buffer means that is disposed for every predetermined number of the plurality of processors, temporarily stores the reference pixel data—which is transmitted from a previous processor or an input unit—sequentially transmits this to a subsequent processor, and has a variable effective stored-pixel-data count; and
a motion-vector determination means of determining a motion vector of the current-image block according to computation-result outputs of the plurality of processor means.
[Claim 18] A motion-vector detection device for detecting motion vectors used in motion-compensating predictive coding by block-matching processing between a current screen image and a reference screen image, comprising:
a first calculation means of executing, according to a plurality of preestablished prediction modes, block-matching processing between a current-image block that is a target of motion-vector detection and reference-image blocks corresponding to representative evaluation points among all evaluation points in a search region in the reference screen

associated with the current-image block and seeks an optimal representative evaluation point indicating a most favorable correlation for the plurality of prediction modes;
a plurality of second calculation means that is provided corresponding to each of the plurality of prediction modes and performs—according to the sought optimal representative evaluation point—block-matching processing between the current-image block and the reference-image blocks for all evaluation points in a region of a predetermined size centered around the corresponding optimal representative evaluation point according to the plurality of prediction modes, each seeking an optimal evaluation value and vector indicating a most favorable correlation for each of the plurality of prediction modes; and
a motion-vector determination means of seeking an optimal prediction mode among the plurality of prediction modes according to outputs of the plurality of second calculation means and determining the optimal vector determined according to the sought optimal prediction mode as a motion vector for the current-image block.
[Claim 19] A motion-vector detection device for detecting motion vectors used in motion-compensating predictive coding by block-matching processing between a current screen image and a reference screen image, comprising:
a first calculation means of executing, according to a plurality of prediction modes, block-matching processing between a current-image block that is a target of motion-vector detection and reference-image blocks expressed by representative evaluation points among all evaluation points in a search region in a reference screen associated with the current-image block and seeking an optimal representative evaluation point indicating a most favorable correlation for each of the plurality of prediction modes;
a plurality of second calculation means that is provided corresponding to each of the plurality of prediction modes, executes—according to the optimal representative evaluation point of the corresponding mode from the first calculation means—block-matching processing between the current-image block and the reference-image blocks for all evaluation points in a region of a predetermined size centered around the optimal representative evaluation point according to the plurality of prediction modes, and calculates an optimal vector and an optimal evaluation value representing the evaluation point indicating the most favorable correlation; and
a motion-vector determination means of determining an optimal prediction mode among the plurality of prediction modes according to outputs of the plurality of second calculation means and determining the optimal vector determined according to the determined optimal prediction mode as a motion vector for the current-image block.
[Claim 20] A motion-vector detection device for detecting motion vectors used in motion-compensating predictive coding by block-matching processing between a current screen image and a reference screen image, comprising:
a first calculation means of executing, according to a plurality of prediction modes, block-matching processing between a current-image block that is a target of motion-vector detection and reference-image blocks expressed by representative evaluation points among all evaluation points in a search region in a reference image associated with the current-image block and seeking an optimal representative evaluation point corresponding to the prediction mode indicating a most favorable correlation for each of the plurality of prediction modes;

a plurality of second calculation means of executing—according to the optimal representative evaluation point from the first calculation means—block-matching processing between the current-image block and the reference-image blocks for all evaluation points in a region of a predetermined size centered around the optimal representative evaluation point according to the plurality of prediction modes and calculating an optimal vector and an optimal evaluation value for each prediction mode; and

a motion-vector determination means of determining an optimal prediction mode among the plurality of prediction modes according to outputs of the second calculation means and determining the optimal vector corresponding thereto as a motion vector for the current-image block.

[Claim 21] A motion-vector detection device for detecting motion vectors used in motion-compensating predictive coding by block-matching processing between a current screen image and a reference screen image, comprising:

a first calculation means of performing, according to a plurality of prediction modes, block-matching processing between a current-image block that is a target of motion-vector detection and each reference-image block corresponding to a representative evaluation point among all evaluation points in a search region in a reference screen associated with the current-image block and seeking an optimal prediction mode indicating a most favorable correlation among the plurality of prediction modes and an optimal representative evaluation point corresponding thereto;

a second calculation means of executing—according to information indicating the optimal prediction mode and the optimal-representative-evaluation-point information from the first calculation means—block-matching processing between the current-image block and the reference-image blocks according to the optimal prediction mode for all evaluation points in a region of a predetermined size in the search region including the optimal evaluation point and calculating evaluation-function values for each reference-image block; and

a motion-vector determination means of determining a motion vector for the current-image block according to outputs of the second calculation means.

[Claim 22] A motion-vector detection device for seeking motion vectors used in motion-predicting coding processing by block-matching processing between a current screen image and a reference screen image, comprising:

a processor array having processors disposed corresponding to each pixel of a current-screen block made of pixels of Q rows by P columns of the current screen image and data storage means that are disposed corresponding to each of the P columns, each storing R reference-screen pixel data, the processors and the data storage means being mutually connected so as to transfer the reference-screen-image pixel data one dimensionally along one direction and each of the processors including a first register means of storing corresponding current-screen-image pixel data, a second register means of storing the imparted reference-screen-image pixel data, and a computation means of subjecting the pixel data stored by the first and second register means to predetermined computational processing to output evaluation-function-value components;

a data application means of imparting the reference-screen-image pixel data to the processor array at a speed of one pixel data per one cycle;

an evaluation-value generation means of adding the evaluation-function-value components output from the processor array to generate one evaluation-function value per one cycle;
a motion-vector determination means of detecting a motion vector of the current-screen block according to the evaluation values imparted over a predetermined number of cycles from the evaluation-value generation means; and
a means of setting the predetermined number of cycles to A·(P·(Q+R)) cycles, A being any natural number.

[Claim 23] A motion-vector detection device for seeking motion vectors used in motion-predicting coding processing by block-matching processing between a current screen image and a reference screen image, comprising:
A processor arrays, each having processors disposed corresponding to each pixel of a current-screen-image block made of Q rows by P columns and data storage means that are disposed corresponding to each of the P columns, each storing R reference-screen pixel data, A being a natural number, each of the A processor arrays having the processors and the data storage means mutually connected so as to transfer the reference-screen-image pixel data one dimensionally along one direction, and each of the processors including a first register means of storing corresponding current-screen-image pixel data, a second register means of storing the imparted reference-screen-image pixel data, and a computation means of subjecting the pixel data stored in the first and second register means to predetermined computational processing to output evaluation-function-value components;
a means of imparting the same reference-screen-image pixel data to the A processor arrays at a ratio of one pixel data per one cycle;
A evaluation-value generation means that are provided corresponding to the A processor arrays and generate evaluation-function values indicating a degree of similarity between the current-screen block and a reference-screen block from the evaluation-function-value components output from the corresponding processor arrays;
A motion-vector determination means that are provided corresponding to each of the A evaluation-value generation means and determine a motion vector for each processor array according to the evaluation values imparted over A·P(Q+R) cycles from the corresponding evaluation-value generation means; and
a current-screen-block updating means of sequentially updating an entirety of current-screen-image-block pixel data of the A processor arrays with the A·P(Q+R) cycles as unit cycles.

[Claim 24] A motion-vector detection device for seeking, by block-matching processing between a current screen image and a reference screen image, motion vectors used in motion-predicting coding processing, comprising:
a processor array provided with processors disposed corresponding to pixels of Q rows by P columns of the current screen image and data storage means that are disposed corresponding to each of the P columns, each storing R reference-screen-image pixel data, each of the processors including a first register means of storing corresponding current-screen pixel data, a second register means of storing imparted reference-screen-image pixel data, and a computation means of subjecting the pixel data stored by the first and second register means to predetermined computational processing to output evaluation-function-value components,

each processor array being divided into a plurality of sub-blocks disposed corresponding to pixels of (P/b) columns by (Q/c) rows, the sub-blocks being disposed in a matrix;
a plurality of data storage means that is provided corresponding to each column of processors, each storing R reference-screen-image pixel data, the data storage means being shared by sub-blocks disposed in the same column—in each sub-block column, the data storage means being mutually connected with the processors of one sub-block so as to one-dimensionally transfer the reference-screen-image pixel data in one direction and being connected only to processors disposed in the same column for the remaining sub-blocks—and the processors in the remaining sub-blocks being mutually connected so as to transfer the imparted reference-screen-image pixel data only along one direction only in the same column;
a means of determining a motion vector of an associated current-screen-image block for each sub-block according to the evaluation-function-value components imparted at each cycle over (P/b)·[(Q/c)+R]·b·c cycles from the processors of each of the plurality of sub-blocks; and
a means of storing pixel data of mutually different current-screen-image pixel blocks in each of the plurality of sub-blocks and sequentially updating the current-screen-image pixel data stored in the plurality of sub-blocks every (P/b)·[(Q/c)+R] cycles.
[Claim 25] A motion-vector detection device for seeking motion vectors used in motion-predicting coding by block-matching processing between a current screen image and a reference screen image, comprising:
a processor array having processors disposed in a matrix corresponding to each pixel of a current-screen block made of pixels of Q rows by P columns of the current screen image and data storage means that are disposed corresponding to each pixel column, each storing R reference-screen pixel data, the plurality of processors and data storage means being mutually connected so as to transfer the reference-screen-image pixel data one dimensionally along one direction and each of the plurality of processors including a first register means of storing pixel data of a first current-screen block, a second register means of storing pixel data corresponding to a second current-screen block, a third register means of storing the imparted reference-screen-image pixel data, a selection means of selecting the pixel data of one among the first and second register means, and a computation means of subjecting the pixel data selected by the selection means and the pixel data stored in the third register means to predetermined computational processing to generate evaluation-function-value components;
an evaluation-value generation means of summing the evaluation-function-value components output from the processor array to generate one evaluation-function value per one cycle;
a motion-vector determination means of determining a motion vector of the current-screen block according to the evaluation-function values output by the evaluation-value generation means;
an updating means of updating, during a motion-vector detection cycle for one among the first and second current-screen blocks, a content of the register means corresponding to the other current-screen block; and
a switching control means of switching the register selected by the selection means every detection cycle.

[Claim 26] A motion-vector detection device for seeking motion vectors used in motion-compensating predictive coding by block-matching processing between a current screen image and a reference screen image, comprising:
an evaluation-value generation means of generating evaluation-function values indicating degrees of similarity between a current-image block of a predetermined size that is a target of motion-vector detection in the current screen and each of a plurality of reference-image blocks in a region in the reference screen associated with the current-image block; and
a determination means of determining a motion vector of the current-image block according to the evaluation-function values imparted from the evaluation-value generation means; wherein
the determination means, when a position in a horizontal and vertical direction for the original-image blocks that are the reference-image blocks is represented as a vector (H,V), when the evaluation-function values of a plurality of reference-image blocks are the same, is provided with
(i) a first selection means of selecting as a motion-vector candidate a vector of a reference-image block where |H|+|V| is smaller,
(ii) a second selection means of selecting as the motion-vector candidate a vector of a reference-image block where |H|+(-V+α) is smaller,
(iii) a third selection means of selecting as the motion-vector candidate a vector of a reference-image block where |H|+(V+α) is smaller, and
a means of placing any one among the first to third selection means in an active state in response to a mode designation signal, α being a constant determined according to a search range in a Y direction of the motion vector.
[Claim 27] A motion-vector detection device for seeking motion vectors used in motion-compensating predictive coding by block-matching processing between a current screen image and a reference screen image, comprising:
an integer-precision vector determination means of subjecting a current-screen block of a predetermined size of the current screen image and a plurality of reference-screen blocks of a region in the reference screen associated with the current-screen block to block-matching processing according to a plurality of prediction modes and determining a motion vector at integer precision for each of the plurality of prediction modes;
a pixel-data storage means of respectively receiving current-screen-block pixel data and a plurality of reference-screen-block pixel data indicated by motion vectors of each of the plurality of prediction modes determined by the integer-precision vector determination means from the integer vector determination means and storing these; and
a fractional-precision vector determination means of receiving the current-screen-block pixel data and the pixel data of each of the plurality reference-screen blocks stored in the pixel-data storage means, performing block-matching processing at fractional precision, and determining a motion vector at fractional precision of each of the plurality of prediction modes.
[Claim 28] A motion-vector detection device for detecting motion vectors used in motion-compensating predictive coding by block-matching processing between a current screen image and a reference screen image, comprising:
an evaluation-value calculation means of seeking evaluation-function values indicating degrees of similarity between a current-screen block of a predetermined size of the current

screen and a plurality of reference blocks associated with the reference screen image in a search region of a predetermined size;
a setting means of setting an effective region of the search region for the current-screen block; and
a means of determining a motion vector of the current-screen block according to evaluation values of the evaluation-value calculation means for reference blocks included in the effective region set by the setting means.
[Description]
[0001]
[Field] This invention relates to a device for detecting motion vectors used for motion compensation of a moving image in predictive-coding processing.
[0002]
[Background] To transmit or store an image signal having a large data volume, data compression art of reducing the data volume is essential. Image data is provided with considerable redundancy due to correlation between proximal pixels, characteristics of human perception, and the like. Data compression art of suppressing this data redundancy to reduce a transmission data volume is called high-efficiency coding. One such high-efficiency coding method is an inter-frame predictive coding method. This inter-frame predictive coding method executes the following processing.
[0003] Prediction errors are calculated, which are differences between pixel data of a current screen (frame or field) to be presently coded and pixel data of the same positions in a previous screen that is referred to. The calculated prediction errors are used for subsequent coding. In this method, an image with little motion can be coded with high efficiency because of significant correlation between the screens. However, an image with significant motion has little correlation between the screens. This increases the errors and creates a disadvantage of actually increasing the transmission data volume.
[0004] As a method of solving the above problem, there is a motion-compensating inter-frame (inter-field) predictive coding method. This method executes the following processing. First, before calculating the prediction errors, motion vectors are calculated in advance using the pixel data of the current screen (frame or field) and the previous screen. A prediction image of the previous screen is moved according to the calculated motion vectors. Pixel data of the previous screen of positions shifted according to the motion vector are made to be reference pixels, and the reference pixels are used as prediction values. Next, prediction errors between the pixels of this moved previous screen and the current screen are calculated and the prediction errors and the motion vectors are transmitted.
[0005] FIG. 151 is a block diagram schematically illustrating an overall configuration of a coder that codes image data according to a conventional motion-compensating predictive coding method. In FIG. 151, the coder includes a preprocessing circuit 910 that executes predetermined preprocessing of an input image signal, a source coding circuit 912 that executes redundancy removal for the signal preprocessed by this preprocessing circuit 910 and quantization of the input signal, and a video multiplex coding circuit 914 that codes the signal from the source coding circuit 912 according to a predetermined format and multiplexes this signal into a code string of a preestablished data structure.

[0006] The preprocessing circuit 910 uses temporal and spatial filters to convert the input image signal into Common Intermediate Format (CIF) and executes filter processing for noise removal.
[0007] The source coding circuit 912 performs orthogonal transform processing such as a discrete cosine transform (DCT) on the imparted signal, performs motion compensation for the input signal, and quantizes the orthogonally transformed image data.
[0008] The video multiplex coding circuit 914 performs two-dimensional variable-length coding of the imparted image signal; also performs variable-length coding of various attributes (such as motion vectors) of blocks, the unit of data processing; and afterward multiplexes this into the code string of the preestablished data structure.
[0009] The coder further includes a transmission buffer 916 that subjects the image data from the video multiplex coding circuit 914 to buffer processing and a transmission coding circuit 918 for making the image data from the transmission buffer 916 compatible with a transmission channel.
[0010] The transmission buffer 916 smooths an information generation speed to a constant speed. The transmission coding circuit 918 executes adding error-correction bits, adding audio-signal data, and the like.
[0011] FIG. 152 is a diagram illustrating one example of a specific configuration of the source coding circuit illustrated in FIG. 151. In FIG. 152, a source coding circuit 912 includes a motion-compensation predictor 920 that detects motion vectors of an input image signal imparted from a preprocessing circuit and generates a reference image that is motion-compensated according to the motion vectors, a loop filter 922 that subjects reference-image pixel data from the motion-compensation predictor 920 to filter processing, a subtractor 924 that seeks differences between an output of the loop filter 922 and the input image signal, an orthogonal transformer 926 that orthogonally transforms an output of the subtractor 924, and a quantizer 928 that quantizes the data orthogonally transformed by the orthogonal transformer 926.
[0012] A configuration of the motion-compensation predictor 920 is described in detail below, but the motion-compensation predictor includes a frame memory, which stores pixel data of one frame previous, and performs detection of the motion vectors and generation of the motion-compensated reference-image pixel data according to the input image-signal data and the pixel data in this frame memory. The loop filter 922 is provided to improve image quality.
[0013] The orthogonal transformer 926 performs an orthogonal transform such as a DCT transform of the data from the subtractor 924 with a block of a predetermined size (normally 8×8 pixels) as one unit. The quantizer 928 quantizes this orthogonally transformed pixel data.
[0014] The motion-compensation predictor 920 and the subtractor 924 execute motion-compensating inter-frame prediction, removing temporal redundancy in the moving image. Moreover, the orthogonal transform by the orthogonal transformer 926 removes spatial redundancy in the moving-image signal.
[0015] The source coding circuit 912 further includes an inverse quantizer 930 for converting the data quantized by the quantizer 928 into a signal state prior to quantization, an inverse orthogonal transformer 932 that performs an inverse orthogonal transform of an output of this inverse quantizer 930, and an adder 934 that adds the output of the loop filter 922 and an output of the inverse orthogonal transformer 932. The inverse quantizer

930 and the inverse orthogonal transformer 932 generate the image used for inter-frame prediction for the next frame. This generated image data is written in the frame memory included in the motion-compensation predictor 920. Because the adder 934 adds the image signal (inter-frame difference data) to the output of the loop filter 922, the image data of the current frame is reproduced. Normally, this inverse quantization processing, inverse orthogonal-transform processing, and adding processing are generally called a local decoding process. Next, calculating the motion vectors is specifically described. To calculate the motion vectors, a block-matching method is generally used.

[0016] Now, as illustrated in (A) in FIG. 153, a state is supposed where an image A in an (m-1)th frame is transitioned to A′ in an mth frame. In the block-matching method, a screen (in this situation, one frame) is divided into blocks of P×Q pixels (generally, P = Q). Blocks that best approximate blocks focused on in the current frame are sought from the previous frame. Shifts from these focused-on blocks to the best approximating blocks in the previous frame are referred to as motion vectors. This is described in more detail below.

[0017] As illustrated in (B) in FIG. 153, it is now supposed that the mth frame is a coding target frame. The frame is divided into blocks of N×N pixels (P = Q = N). It is now supposed that a value of pixel data in an uppermost and leftmost pixel position (Nk,N1) in a block of N×N pixels in the mth frame is Xm(Nk,N1). An absolute-value sum of differences in the data of corresponding pixels between a block in the previous frame whose pixel positions are shifted according to a vector (i,j) and a block in the current frame is sought. Next, this shift vector (i,j) is substituted with various values and respective difference absolute-value sums are sought. These difference absolute-value sums are generally called evaluation-function values. The position (i,j) that imparts the smallest difference absolute-value sum is defined as the motion vector.

[0018] One motion vector needs to be transmitted per one block of pixels. A small block size increases transmission information, preventing effective data compression. Meanwhile, a large block size makes effective motion detection difficult. Therefore, the block size is generally made to be 16×16 pixels and a motion-vector search range (maximum change width of i,j) is generally made to be -15 to +15. Motion-vector calculation by the block-matching method is specifically described below.

[0019] FIG. 154 is a diagram illustrating a specific method of calculating motion vectors by the block matching method. Now, as illustrated in FIG. 154, an image 950 made of 352 dots (pixels) × 288 lines is supposed. The image 950 is divided into a plurality of blocks, one block being a group of 16×16 pixels. Detection of the motion vectors is executed in these block units. With a block 954 in a previous frame in the same position as a block 952 that is a target of the motion-vector detection processing (“template block”) as reference, a block 956 that is plus and minus sixteen pixels greater in a horizontal direction and a vertical direction on the screen is made to be a search block (“search area”). A motion-vector search for the template block 952 is executed in this search area 956. The search method of the motion vectors according to the block-matching method is provided with the following processing steps.

[0020] A block having a displacement corresponding to a motion-vector candidate (illustrated as vector (i,j) in FIG. 154) is sought. An evaluation-function value such as a difference absolute-value sum (or a difference squared sum) of the pixels of this sought block and the pixels of the template block in corresponding positions is sought.

[0021] The above operations are executed for all displacements of the vector (i,j) from (-16,-16) to (+16,+16). After seeking the evaluation-function values (evaluation values) for all prediction-image blocks (all image blocks in the search area 956), the prediction-image block with the smallest evaluation-function value is detected. The vector heading from the block in the same position as ("directly behind") the template block (in FIG. 154, the block 954 illustrated by the vector (0,0)) to the prediction-image block with the smallest evaluation-function value is determined to be the motion vector for this template block 952.
[0022] FIG. 155 is a diagram illustrating an overall configuration of a motion-vector detection device realized by conventional hardware and is illustrated by, for example, A. Altieri et al. on p. 2453 to 2456 of the Proceedings of ICASSP-89, IEEE, 1989. In FIG. 155, the motion-vector detection device includes a search-area input register 962 for inputting one column of pixel data of a search area, a processor array 966 including a plurality of processors disposed in a matrix of rows and columns of the same size as evaluation points of a template block (motion-vector candidates), search-area registers 964a and 964b that store data of the same column in the search area for the processor array, and a motion-vector detection unit 968 that detects a motion vector according to computation results of the processor array 966.
[0023] In the processor array 966, the processors are disposed corresponding to each evaluation point—that is, each displacement vector (shift vector (i,j)). That is, a processor Pij disposed in row i, column j calculates a displacement vector D(i,j).
[0024] FIG. 156 is a diagram illustrating a configuration of a processor included in the processor array illustrated in FIG. 155. In FIG. 156, a processor 970 includes a three-input register 972 that receives search-area pixel data transmitted from processors in three directions in horizontal and vertical directions in the array 966 and allows one of these inputs to pass through in response to a selection signal SEL, a distortion calculation unit 974 that calculates a distortion (difference absolute-value sum) based on the search-area pixel data Y from the three-input register 972 and template-block pixel data X imparted from the outside, and a two-input register 976 that receives a distortion D from the distortion calculation unit 974 and a distortion from a processor adjacent in the horizontal direction in the array 966 and selects one according to a selection signal To to be passed through.
[0025] The processor 970 illustrated in FIG. 156 is disposed two dimensionally in the processor array 966 illustrated in FIG. 156 in correspondence with the evaluation points in the search area—that is, all displacement vectors serving as the motion-vector candidates. Each processor of the processor array 966 (see FIG. 155) is imparted in common with the same template pixel data X. At this time, the processor 970 is also imparted with corresponding image data in the search-area block. For example, in a situation where the template-block pixel data X is X(m,n), search-area-block pixel data Y(i+m,j+n) is imparted to the processor Pij. The search-window data is transferred via the search-area side registers 964a and 964b illustrated in FIG. 155 and the processor 970s in the processor array 966. To accurately impart the search-area-block pixel data Y(i+m,j+n) to each processor for the template-block pixel data X(m,n) imparted from the outside, the template block and the search-area block must be scanned with a certain regularity.
[0026] FIG. 157 is a diagram illustrating a scanning aspect of the data of the template block in the above motion-vector detection device. In FIG. 157, as illustrated by the arrows in FIG. 157, a template block 999 is first scanned along the same column, heading downward from

above. Next, the pixel data of one column adjacent is scanned, heading upward from below. This generates the template-block pixel data, which is sequentially imparted to the motion-vector detection device. This scanning method is called a “snaking scan.” The search-area-block pixel data imparted to the processor array 966 is similarly scanned according this “snaking scan” of the template-block pixel data. The processors 970 need to transfer the search-area pixel data in an upward/downward direction or a leftward direction in FIG. 156 according to disposition positions thereof in the array. The three-input register 972 is provided to realize this three-direction transfer. The two-input register 976 (see FIG. 156) is provided to transmit the distortion calculated by the processor 970 to the motion-vector detection unit 968 to seek the displacement vector imparting the smallest distortion (evaluation-function value) in the motion-vector detection unit 968 (see FIG. 155) after the evaluation-function values of each displacement vector are calculated. The motion-vector detection unit 968 detects the smallest distortion among the distortions from the processors in the processor array 966, seeks the position of the processor imparting this smallest distortion, and determines the position of this processor as the motion vector. Next, operations of the motion-vector detection device illustrated in this FIG. 155 are briefly described.

[0027] The processor Pij disposed in row i, column j, in the processor array 966 calculates the distortion D(i,j), which is represented as follows:

$$D(i,j) = \Sigma|X(m,n) - Y(m+i, n+j)|.$$

Here, the sum Σ is performed in relation to m and n. Variation ranges of m and n are determined by the size of the template block.

[0028] Now, as illustrated in FIG. 158, pixels disposed in m rows by n columns as a template block 980 are supposed. In a first cycle, search-area-block pixel data indicated by reference sign 982 is stored in each processor in the processor array 966. From the outside, a pixel X(1,1) of the first row and the first column in the template block 980 is imparted in common to all processors in the processor array 966. Each processor 970 in the array 966 seeks the difference absolute values between the search-area-block (search-window) pixel data Y stored therein and the imparted template-block pixel data X and totals these.

[0029] In the next cycle, the search-area block is shifted downward by one row in FIG. 158 in the processor array 966. Search-area-block (search-window) pixel data indicated by reference sign 983 is stored in the processor array 966. In this state, next, the next pixel data X(2,1) of the template block 982 is imparted. The search-window pixel data secured in the processor Pij is Y(m+i,n+j+1). The difference absolute values are again taken and totaled using these pixel data. This operation is repeated M times.

[0030] When the above operation is repeated M times, search-area pixel data of one column of the search area is written from the outside via the search-area input register 962 illustrated in FIG. 158. Image data of one column of the search area that is become unnecessary is released. By this, new search-area pixel data is stored in the search-area side registers 964a and 964b and the processor array 966. This operation is repeatedly executed for each column.

[0031] That is, as illustrated in FIG. 159, first, calculation of the difference absolute-value sums is executed using the search window (block including all rows in the search area). After completing M cycles, similar calculation is again executed using pixel data of the next search window (block rightwardly adjacent by one column in the search area). Subsequently, similar operations are executed for search windows 994, …. Finally, when

calculation for all pixel data in a search area 996 is executed, the processor Pij seeks and holds the distortion D(i,j). The distortion D(i,j) sought by this processor Pij is transmitted to the motion-vector detection unit 968, and here, the displacement vector imparting the smallest distortion is detected as the motion vector.

[0032]

[Technical Problem] Several methods are proposed as a prediction-image detection method in a motion-compensating inter-frame (or inter-field) predictive coding method. To obtain a more favorable coding efficiency, motion-detection processing needs to be performed according to a plurality of prediction-image detection methods, an optimal prediction-image detection method needs to be afterward selected, and motion vectors need to be detected according to this optimal prediction-image detection method. A screen may be configured in field units or frame units. One frame is configured by two fields (an even field and an odd field). For each situation, there are the following prediction-image detection methods as examples.

[0033] (A) Situation of coding pixel data in field units:

(a) A field image is divided into a plurality of blocks in units of P×Q pixels, and one motion vector is detected for each block (one prediction image is generated).

[0034] (b) The divided blocks are further divided in two, up and down, in a vertical direction of the screen, and one motion vector is detected for each of the blocks divided in two. Therefore, for a block of P×Q pixels, a motion vector for the block in the upper half and a motion vector for the block in the lower half are detected (two prediction images are generated).

[0035] (B) Situation of coding pixels in frame units:

(a) A frame image is divided into a plurality of blocks in units of a block of P×Q pixels, and one motion vector is detected for each block (one prediction image is generated).

[0036] (b) The block of P×Q pixels is divided into two pixel groups of pixels present in the same field—that is, pixels belonging to the even field and pixels belonging to the odd field—and one motion vector is detected for each pixel group. Therefore, for a block of P×Q pixels, a motion vector for the pixel group of the even field and a motion vector for the pixel group belonging to the odd field are detected (two prediction images are generated).

[0037] In a situation where the motion-vector detection device is configured as above, each processor in the processor array is disposed corresponding to a displacement vector (evaluation point) that is a motion-vector candidate. Moreover, each processor receives the same template-block pixel data. Therefore, the processors can only seek the distortion (evaluation-function value) all according the same prediction-pixel method, and a plurality of motion vectors cannot be detected in a parallel aspect according to a plurality of prediction-image detection methods. As is clear from the formula for the distortion D(i,j) calculated by the processor Pij above, this is because once the imparted template-block pixel data X(m,n) is determined, the search-window pixel data Y(m+i,n+j) imparted accordingly is also uniquely determined. Therefore, to execute a plurality of prediction-image detection methods to improve coding efficiency, a motion-vector detection device must be provided for at least each of the plurality of prediction-image detection methods and these motion-vector detection devices must be operated in parallel, creating a problem of an increased device scale and increased power consumption.

[0038] Furthermore, in the above motion-vector detection device, the same template-block pixel data is imparted in common to all processors in the processor array. As such, a circuit

for writing pixel data of the template block requires a large driving power, creating a problem of increased current consumption in this template-block pixel-data writing circuit and increased power consumption for the overall device.

[0039] Furthermore, with the above motion-vector detection device, each processor is disposed corresponding to a displacement vector (evaluation point) that serves as a motion-vector candidate. If the search area is +16 to -16 in the vertical direction and -16 to +16 in the horizontal direction, the number of displacement vectors serving as motion-vector candidates is 33×33 = 1,089, creating a problem where the number of processors is very large and the device occupies a large area.

[0040] Furthermore, in each cycle of computation, data transferring in the processor array is performed via the processors. At this time, three-input registers are used to determine the data transfer directions between the processors, and any one of these three-input registers needs to be driven. This creates a problem of increased power consumption at the time of data transfer.

[0041] Furthermore, as a motion-vector search method, there is a full search method and a hierarchical search method. The full search method is a method of seeking evaluation-function values (such as difference absolute-value sums or difference squared sums) for all displacement vectors (evaluation points) in the search area and seeking motion vectors from the evaluation-function values of all evaluation points. One example of the hierarchical search method is given in JP H3-68597 B2, and this method includes the following search steps. Among the evaluation points in the search area, representative evaluation points are determined (thinning of evaluation points), and the evaluation-function values are sought for the representative evaluation points. From the evaluation-function values of the representative evaluation points, a representative evaluation point imparting the smallest evaluation-function value is sought, and this is made to be an optimal representative evaluation point. Next, evaluation-function values are sought for all evaluation points in a region of a predetermined size centered around this optimal representative evaluation point, an optimal evaluation point is sought from these evaluation-function values, and the motion vector is determined.

[0042] With a hierarchical search method, an evaluation-function value does not need to be sought for all evaluation point in a search area such that a device scale can be decreased and a motion-vector detection time can be shortened. However, because motion-vector detection is performed by thinning the evaluation points, there is a disadvantage of less precision compared to the full search method. Conversely, although the full search method has an advantage of being more precise than a hierarchical search method, it also has a disadvantage of having a large device scale. That is, in a situation where it is supposed that a number of computations per unit time is constant and a device is configured using the same amount of hardware, applying a hierarchical search method enables a prediction image to be sought over a wide search area but creates a disadvantage of less precision. Meanwhile, applying the full search method narrows the search area but provides an advantage of excellent motion-vector precision.

[0043] In particular, in moving-image coding methods, which are currently being standardized by the ISO (International Organization for Standardization), prediction for a current screen image is performed using screens that precede and follow it temporally. As such, motion detection needs to be performed using screens in positions temporally separated by several screens (frames or fields). As an example of this, there is motion-

vector detection of a B picture in storage-media coding art. In storage-media coding art, because image data is stored in storage media, there are no temporal-axis restrictions. Therefore, for image data called a B picture, there is sequential-direction prediction of detecting a prediction image using a past image, reverse-direction prediction of detecting a prediction image using a future image, and interpolative prediction of performing prediction using both a previous screen and a subsequent screen. In a situation of performing prediction using such previous and subsequent screens, to follow fast motion, a search area needs to be widened (in a narrow search area, the image sticks out from the search area, preventing accurate motion-vector detection). It is deemed favorable to perform the search with a range of ±64 to ±128 pixels in both the horizontal and vertical directions around the template block as the search area. To realize such a wide search area without increasing a device scale requires use of the hierarchical search method above.
[0044] However, motion prediction methods premised on a conventional hierarchical search method can only detect a prediction image according to a motion detection method following a single prediction mode, and what kind of hierarchical search method to adopt in a motion detection method provided with a plurality of prediction modes such as above is not yet proposed.
[0045] Therefore, an object of this invention is to provide a motion-vector detection device that can detect a prediction image in correspondence with a plurality of prediction modes without increasing a device scale.
[0046] Another object of this invention is to provide a motion-vector detection device that can detect a prediction image (motion vectors) at high speed according to a plurality of prediction modes without increasing current consumption and increasing device complexity.
[0047] Yet another object of this invention is to provide a motion-vector detection device that can detect motion vectors according to a hierarchical search method with high precision without increasing a device scale.
[0048] Yet another object of this invention is to provide a motion-vector detection device that can detect by a high-speed and simple circuit configuration motion vectors for each of a plurality of prediction modes (prediction-image detection methods) according to a hierarchical search method.
[0049]
[Solution to Problem] The motion-vector detection device of claim 1 is provided with an evaluation means for indicating degrees of similarity between a current-image block of a predetermined size that is a target of motion-vector detection in the current screen and each of a plurality of reference-image blocks in a region associated with the current-image block in the reference screen and seeking in parallel evaluation-function values determined according to each of a plurality of prediction modes for each reference block and a motion-vector determination means of receiving in parallel the evaluation-function values of each reference block and each of the plurality of prediction modes from this evaluation means and determining in parallel motion vectors for each prediction mode for the current-image block.
[0050] In the motion-vector detection device of claim 2, the evaluation means of claim 1 includes a plurality of first computation means that is disposed corresponding to pixels included in the current-image block, includes a storage means of temporarily storing corresponding pixel data in the reference-image blocks and corresponding pixel data in the

current-image block, and subjects the pixel data stored in the storage means to computation making up a portion of the evaluation function and a second computation means of classifying outputs of this plurality of first computation means in correspondence with each of the plurality of prediction modes and subjecting the outputs of the first computation means to the remainder of the evaluation function in a parallel aspect according to each classification.

[0051] In the motion-vector detection device of claim 3, the plurality of first computation means of claim 1 is provided with a plurality of processors that is disposed so each transfers the pixel data stored in the storage means only along one direction and is disposed in a matrix of rows and columns corresponding to the current-image block.

[0052] In the motion-vector detection device of claim 4, the current screen and the reference screen are configured by a frame including an even field and an odd field and the plurality of prediction modes includes a motion prediction mode using pixels included in the odd field (prediction-image detection method), a motion prediction mode using pixels included in the even field, and a motion prediction mode using pixels included in the frame.

[0053] In the motion-vector detection device of claim 5, the current screen and the reference screen are each configured by one field and the plurality of prediction modes includes a motion prediction mode that uses pixels included in an upper-half block obtained by dividing the current-image block equally in two in a vertical direction, a motion prediction mode that uses pixels included in a lower-half block, and a motion prediction mode that uses the pixels included in both the upper-half and lower-half blocks.

[0054] In the motion-vector detection device of claim 6, a second computation means in the motion-vector detection device of any one of claims 1 to 5 includes a switching means of realizing three classifications by dividing the current-image block into two blocks—upper and lower in a vertical direction—at a time of configuring a frame screen, dividing these into four blocks—even-field blocks including pixels of an even field and odd-field blocks including pixels included in an odd field—receiving for these four divided blocks outputs of corresponding first computation means in parallel, and changing a configuration of the four divided blocks according to a prediction-mode designation signal. The three classifications realized by this switching means are a frame prediction mode that uses pixels included in the frame, an odd-field prediction mode that uses the pixels included in the odd field, and even-field prediction that uses the pixels included in the even field or, where the screen is configured in field units, a field-prediction mode that performs motion prediction using pixels in this field, an upper-half prediction mode that uses pixels included in the upper half in the current-image block, and a lower-half prediction mode that uses the pixels included in the lower-half block of the current-image block.

[0055] The motion-vector detection device of claim 7 further includes an input setting means of setting, in response to a format designation signal that designates a format of the screen, a screen format of input image data to the format designated by this forming designation signal and imparting this to the evaluation means.

[0056] In the motion-vector detection device of claim 8, a plurality of first computation means in the motion-vector detection device of any one of claims 1 to 7 is disposed so as to essentially configure a one-dimensional array and provided are a plurality of processors that transmits input current-image data along one direction and a buffer means that is disposed for every predetermined number of this plurality of processors, stores reference-

pixel-block image data imparted from an input unit or a previous processor, outputs this stored pixel data in order of storage, and has a variable effective stored-pixel count.
[0057] In the motion-vector detection device of claim 9, the motion-vector detection device of any one of claims 1 to 8 is further provided with a third computation means of seeking in parallel evaluation-function values for each of the plurality of prediction modes according to the plurality of prediction modes for reference-image blocks corresponding to representative evaluation points satisfying predetermined conditions in a region associated with the current-image-pixel block and a region designation means of designating for each prediction mode a region of reference-image blocks a first computation means should compute based on an output of this third computation means. The first computation means is disposed corresponding to the region designated by this region designation means and seeks in parallel evaluation-function values according the plurality of prediction modes in the region-designated reference-image-block region.
[0058] The motion-vector detection device of claim 10 is the motion-vector detection device in claim 9, wherein the region designation means separately designates a reference-image-block region for each of the plurality of prediction modes.
[0059] In the motion-vector detection device of claim 11, the region designation means of claim 9 designates one reference-image region for the plurality of prediction modes.
[0060] In the motion-vector detection device of claim 12, the region designation means in claim 9 sets one reference-image block in common for the plurality of prediction modes and enables only a computation operation of calculating the evaluation-function values according to only one prediction mode for a first computation means.
[0061] The motion-vector detection device of claim 13 is provided with a computation means that includes a plurality of processors disposed corresponding to each pixel of a current-frame-image block that is a target of motion-vector detection, inputs pixel data of the current-frame-image block and pixel data of a corresponding reference-image block, and performs predetermined computation of the input pixel data corresponding to both blocks; a summing means of classifying outputs of this computation means into computation results for pixels in an even field and computation results for pixels in an odd field and adding the outputs of the computation means to seek a sum for each classification; and a motion-vector determination means of setting in a parallel aspect a motion vector for each classification according to outputs of this summing means. By this, a motion vector according to an odd-field prediction mode using the odd-field pixels and motion vectors respectively according to an even-field prediction mode using the even-field pixels and a frame prediction mode using frame pixels are determined in parallel.
[0062] The motion-vector detection device of claim 14 is provided with a computation means that includes a plurality of processors disposed corresponding to each pixel of a current-field-image block that is a target of motion-vector detection, inputs pixel data of this current-field-image block and pixel data in a corresponding reference-image block, and performs predetermined computation of the input pixel data corresponding to both blocks; a summing means of classifying outputs of this computation means into computation results for pixels in a block in a vertical-direction upper half of the current-image block, computation results for pixels in a block in a vertical-direction lower half of the current-image block, and computation results for pixels in the current-image block and adding the outputs of the computation means to seek a sum for each classification; and a motion-vector determination means of determining in a parallel aspect according to outputs of this

summing means a motion vector for each classification. By this, motion vectors respectively according to an upper-half-block prediction mode using the pixels of the upper-half block of the current-field-image block, a lower-half-block prediction mode using the pixels of the lower-half block of the current-image block, and a field prediction mode using the pixels of the current-image block are determined in parallel.

[0063] The motion-vector detection device of claim 15 includes a computation means that includes a plurality of processors disposed corresponding to each pixel of a current-image block that is a target of motion-vector detection of a current screen image, inputs each pixel data of this current-image block and a reference-image block in a region associated with this current-image block in a reference screen image, and performs predetermined computation of the pixel data corresponding to both input blocks. A plurality of processors includes—supposing a screen to be configured by a frame—a first processor group that corresponds to pixels in an upper-half block in a screen vertical direction of this current-image block and corresponds to pixels in an odd field, a second processor group that corresponds to the pixels in the block in this upper half and is disposed corresponding to pixels in an even field, a third processor group that corresponds to pixels in a lower half in the screen vertical direction of the current-image block and is disposed corresponding to pixels in an odd field, and a fourth processor group that corresponds to the pixels in the lower half and is disposed corresponding to the pixels in the even field.

[0064] The motion-vector detection device of claim 15 further includes a classification means of classifying outputs of this computation means into three types in response to a mode designation signal. This classification means includes a first combining means of combining the first and second processor groups and combining outputs of the third and fourth processors groups to generate two types, a second combining means of combining the first and third processor group and outputs of the second and fourth processor groups, and a means of selecting the outputs of one among the first and second combining means in response to the mode designation signal.

[0065] The motion-vector detection device of claim 15 is further provided with an evaluation-value generation means of adding and summing the outputs of this classification means according to each type and generating in a parallel aspect evaluation-function values for each classification and an evaluation-function value for an entirety of the current-image block and a motion-vector determination means of determining in a parallel aspect a motion vector of the current-image block for each classification and the current-image block according to an output of this evaluation-value generation means.

[0066] The motion-vector detection device of claim 16 includes a means of varying an input order and an output order of reference-image data and is provided with a buffer means of temporarily storing input reference-screen-image data and a means of calculating a motion vector of a current-image block that is a target of motion-vector detection from the reference-image data read from this buffer means and current-screen-image data.

[0067] The motion-vector detection device of claim 17 is provided with a plurality of processors that is disposed corresponding to pixels of a current-image block that is a target of motion-vector detection, each including a storage means of storing pixel data of a corresponding current-image block and pixel data corresponding to a reference-image block in a reference screen image associated with this current-image block, and a computation means of transmitting the data stored in this storage means only along one direction and performing predetermined computation processing on the pixel data stored

in the storage means; a buffer means that is disposed for every predetermined number of this plurality of processors, temporarily stores the reference-image-block pixel data—which is transmitted from an input unit or a previous processor—sequentially transmits the stored pixel data to a subsequent processor in order of storage, and has a variable effective stored-pixel count; and a motion-vector determination means of determining a motion vector of the current-image block from computation-result outputs of the plurality of processor means.

[0068] The motion-vector detection device of claim 18 includes a first calculation means of performing, according to a plurality of prediction modes, block-matching processing between a current-image block that is a target of motion-vector detection and reference-image blocks corresponding to representative evaluation points among all evaluation points in a search region in a reference screen associated with this current-image block and seeking in parallel an optimal representative evaluation point indicating a most favorable correlation for each of the plurality of prediction modes; a second calculation means of performing—according to the optimal representative evaluation points for each of the plurality of prediction modes from this first calculation means—block-matching processing between the current-image block and the reference-image blocks for all evaluation points included in a region in the reference screen of a predetermined size centered around the evaluation points indicated by the optimal representative evaluation points and calculating motion vectors for each of the plurality of prediction modes; and a means of determining an optimal prediction mode among the plurality of prediction modes according to outputs of this second calculation means and determining the optimal vector corresponding thereto as the motion vector.

[0069] The motion-vector detection device of claim 19 includes a first calculation means of executing in a parallel aspect, according to a plurality of prediction modes, block-matching processing between a current-image block that is a target of motion-vector detection and reference-image blocks corresponding to representative evaluation points among all evaluation points in a search region having a predetermined size in a reference screen associated with this current-image block and seeking an optimal representative evaluation point indicating a most favorable degree of similarity for each of the plurality of prediction modes; second calculation means that are provided corresponding to each of the plurality of prediction modes, performs—according to the optimal representative evaluation points for each of the corresponding prediction modes from the first calculation means—block-matching between the current-image block and the reference-image blocks for all evaluation points included in a region of a predetermined size in a search region including the corresponding optimal representative evaluation points according to the corresponding of prediction modes, and calculates optimal vector and an optimal evaluation-function value for the corresponding prediction modes; and a motion-vector determination means of determining a motion vector for the current-image block from among the optimal vectors according to outputs of this second calculation means.

[0070] The motion-vector detection device of claim 20 includes a first calculation means of executing in a parallel aspect, according to a plurality of prediction modes, block-matching processing between a current-image block that is a target of motion-vector detection and reference-image blocks corresponding to representative evaluation points among all evaluation points in a search region of a predetermined size in a reference screen associated with this current-image block and seeking an optimal representative evaluation

point indicating a most favorable correlation for each of the plurality of prediction modes; a first optimal-evaluation-point determination means of determining a first optimal evaluation point indicating a highest degree of similarity among the optimal representative evaluation points for each of the plurality of prediction modes according to outputs of this first calculation means; a second calculation means of seeking a displacement vector according to each of the plurality of prediction modes for all evaluation points in a region of a predetermined size in a search region including the evaluation point indicated by this optimal vector according to the optimal vector from this first optimal-evaluation-point determination means and seeking an optimal displacement vector from this sought displacement vector for each prediction mode; and a motion-vector determination means of receiving the optimal displacement vectors from this second calculation means and determining an optimal displacement vector indicating the highest correlation to be a motion vector of the current-image block.

[0071] The motion-vector detection device of claim 21 includes a first calculation means of executing in a parallel aspect, according to a plurality of prediction modes, block-matching processing between a current-image block that is a target of motion-vector detection and reference-image blocks corresponding to representative evaluation points among all evaluation points of a search region of a predetermined size in a reference screen associated with this current-image block and seeking an optimal representative evaluation point indicating a most favorable correlation for each of the plurality of prediction modes; a first determination means of determining an optimal prediction mode indicating a highest correlation and the corresponding optimal representative evaluation point according to outputs of this first calculation means; a second calculation means of performing—according to the optimal prediction mode and the optimal representative evaluation point from this first determination means—block-matching processing according to the optimal prediction mode for all evaluation points in a predetermined region in a search region including this optimal evaluation point and seeking evaluation-function values for each reference-image block; and a motion-vector determination means of determining a motion vector for the current-image block according to the evaluation-function values from this second calculation means.

[0072] The motion-vector detection device of claim 22 includes a processor array including processors disposed in a matrix corresponding to each pixel of a current-screen block made of pixels of Q rows by P columns of a current screen image and data storage means that are disposed corresponding to each of the P columns, each storing R reference-screen-image pixel data. The processors and the data storage means are mutually connected so as to transfer the reference-screen-image pixel data one dimensionally along one direction. Each of the processors includes a first register that stores corresponding current-screen-image pixel data, a second register means of storing the imparted reference-screen-image pixel data, and a computation means of subjecting the pixel data stored by the first and second register means to predetermined computation processing to output components of evaluation-function values indicating a degree of similarity between a reference-screen block and the current-screen block stored in each processor.

[0073] The vector detection device of claim 22 is further provided with a data application means of imparting the reference-screen-image pixel data to the processor array at a speed of one pixel data per one cycle; an evaluation-value generation means of summing the evaluation-function-value components output from the processor array to generate one

evaluation-function value per one cycle; a motion-vector determination means of detecting a motion vector of the current-screen block according to the evaluation values imparted over a predetermined number of cycles from this evaluation-value generation means; and a means of setting the predetermined number of cycles to A·(P·(Q+R)) cycles. A is any natural number. The vector detection device of claim 23 includes A processor arrays, each including pixels of Q rows by P columns and disposed corresponding to different current-screen-image blocks. Each of the A processor arrays includes processors disposed in a matrix corresponding to each pixel of the corresponding current-screen block and data storage means that are disposed corresponding to each column of processors, each storing R reference-screen-image pixel data. The processors and the data storage means are mutually connected so as to transfer the reference-screen-image pixel data one dimensionally along one direction. Each of the processors includes a first register means of storing pixel data of the corresponding current-screen block, a second register means of storing the imparted reference-screen-image pixel data, and a computation means of subjecting the pixel data stored in the first and second register means to predetermined computation processing and outputting components of evaluation-function values indicating a degree of similarity between the corresponding current-screen block and the imparted reference-screen-image block.

[0074] The vector detection device of claim 23 is further provided with A evaluation-value generation means that are provided corresponding to each of the A processor arrays, each generating the evaluation-function values from the evaluation-function-value components output by the corresponding processors; A motion-vector determination means that are provided corresponding to the A evaluation-value generation means, each determining a motion vector according to the evaluation-function values imparted over A·P(Q+R) cycles of the corresponding evaluation-value generation means; and a means of sequentially updating an entirety of the current-screen-block pixel data of the A processor arrays every A·P(Q+R) cycles. The vector detection device of claim 24 includes a plurality of processors disposed in a matrix corresponding to each pixel of a current-screen block included in pixels of Q rows by P columns. Each of the plurality of processors includes a first register means of storing corresponding current-screen-image pixel data, a second register means of storing imparted reference-screen-block-image pixel data, and a computation means of subjecting the pixel data stored by the first and second register means to predetermined computational processing to output evaluation-value components indicating a degree of similarity between the reference-screen block and the current-screen block.

[0075] The vector detection device of claim 24 further includes data storage means that are disposed corresponding to each column of processors, each storing R reference-screen-image pixel data. Each processor array is divided into a plurality of sub-blocks corresponding to (P/b)·(Q/c) pixels, and these blocks are disposed in a matrix. The pixel data of the same reference-screen block is imparted to sub-blocks disposed in the same column. Moreover, in each column, one sub-block has the processors and the data buffers mutually connected so as to one dimensionally transfer the reference-screen-block pixel data and the remaining sub-blocks in the same column has the processors connected so each column transfers the reference-screen-block pixel data along only one direction. Sub-blocks disposed in the same column share a data buffer, and each sub-block inputs the reference-screen-image pixel data via the data storage means.

[0076] The vector detection device of claim 24 is further provided with a means of adding the evaluation-function-value components imparted over $(P/b)\cdot[(Q/c)+R]\cdot b\cdot c$ cycles from each sub-block each cycle to respectively generate evaluation-function values and determining a motion vector for each corresponding sub-block according to the generated evaluation-function value; and a means of storing pixel data of mutually different current-screen blocks in each of the plurality of sub-blocks and sequentially updating the current-screen block every $(P/b)\cdot[(Q/c)+R]\cdot b\cdot c$ cycles. The vector detection device of claim 25 includes a plurality of processors disposed in a matrix corresponding to pixels of a current-screen block; and data storage means disposed corresponding to each column of processors that store reference-screen-image pixel data. Each of these processors includes a first register that stores pixel data of a first current-screen block, a second register means of storing corresponding pixel data of a second current-screen block, a third register means of storing pixel data of an imparted reference-screen block, a selection means of selecting one among outputs of the first and second register means, and a computation means of subjecting the pixel data output by the selection means and the pixel data stored in the third register means to predetermined computational processing to generate evaluation-function-value components.

[0077] The vector detection device of claim 25 is further provided with an evaluation-value generation means of generating evaluation-function values according to the evaluation-function-value components output from this processor array, a motion-vector determination means of determining a motion vector according to the evaluation-function value imparted from this evaluation-value generation means, a means of updating, during a motion-vector detection cycle for one among the first and second current-screen blocks, the register storing the other current-screen-block pixel data with pixel data of another current-screen block, and a means of switching the register selected by the selection means every motion-vector detection cycle. The vector detection device of claim 26 includes an evaluation-value generation means of generating evaluation-function values indicating degrees of similarity between a current-image block of a predetermined size that is a target of motion-vector detection in a current screen and each of a plurality of reference-image blocks in a region associated with the current-image block in a smallest screen; and a determination means of determining a motion vector of the current-image block according to the evaluation-function values imparted from this evaluation-value generation means. This determination means, when the evaluation-function values are the same, according to a vector (H,V) for the current-screen block of the reference-screen block, is provided with (i) a first selection means of selecting as a motion-vector candidate a vector of a reference-image block where $|H|+|V|$ is smaller, (ii) a second selection means of selecting as the motion-vector candidate a vector of a reference-image block where $|H|+(-V+\alpha)$ is smaller, (iii) a third selection means of selecting as the motion-vector candidate a vector of a reference-image block where $|H|+(V+\alpha)$ is smaller, and a means of placing any one among the first to third selection means in an active state in response to an operation-mode designation signal. Here, α is a constant determined according to a range of a region associated with the current-image block in the reference screen in a Y direction.

[0078] The vector detection device of claim 27 is provided with an integer-precision vector determination means of subjecting a current-screen block of a predetermined size of the current screen image and a plurality of reference-screen blocks included in a predetermined region in a reference screen associated with this current-screen block to

block-matching processing according to a plurality of prediction modes and determining a motion vector at integer precision for each of the plurality of prediction modes; a storage means of storing current-screen-block pixel data from this integer-precision vector determination means and pixel data of each of the plurality of reference-screen blocks indicated by the motion vectors of each of the plurality of prediction modes determined by this integer-precision vector determination means, and a fractional-regulation vector determination means of performing block-matching processing at fractional precision for the current-screen-block pixel data stored in this storage means and each of the plurality of reference-screen blocks to determine motion vectors at fractional precision of each of the plurality of prediction modes.The vector detection device of claim 28 is provided with an evaluation-value calculation means of seeking evaluation-function values indicating degrees of similarity between a current-screen block of a predetermined size of a current screen and a plurality of reference-screen blocks corresponding to a plurality of reference screen images in a search region of a predetermined size; a setting means of setting an effective region of the search region; and a means of determining a motion vector of the current-screen block according to evaluation-function values of the evaluation-value calculation means for reference-screen blocks included in the effective region set by this setting means.

[0079]

[Actions] In the invention of claim 1, the evaluation means calculates in parallel the evaluation-function values corresponding to each of the plurality of prediction modes (prediction-image detection method) and the motion-vector determination means determines the motion vectors for each prediction mode according to the outputs of this evaluation means such that motion vectors can be detected corresponding to a plurality of prediction modes at high speed and without increasing a device scale.

[0080] In the invention of claim 2, the evaluation means has the processors disposed corresponding to the pixels of the current-image block and these processors perform the computation between the pixels of the corresponding current-image block and the corresponding pixels of the corresponding reference-image blocks to output the evaluation-function-value components. As such, a common computation can be executed for a plurality of prediction modes using the same circuit configuration and a device scale can be reduced.

[0081] In the invention of claim 3, it is only required that the first computation means transfer the pixel data only along one direction; this enables reduced current consumption at a time of pixel-data transferring.

[0082] In the invention of claim 4, the plurality of prediction modes includes the odd-field prediction mode, the even-field prediction mode, and the frame prediction mode such that motion vectors can be detected at high speed corresponding to each of this plurality of prediction modes.

[0083] In the invention of claim 5, the plurality of prediction modes includes the upper-half-block prediction mode, the lower-half-block prediction mode, and the field prediction mode such that in a situation where a screen format is a field screen, motion vectors can be detected at high speed in correspondence with this plurality of prediction modes.

[0084] In the invention of claim 6, the outputs of the first computation means are classified by the classification means and the configuration of this classification is changed by the

switching means. As such, easy and flexible correspondence is possible with any prediction mode.
[0085] In the invention of claim 7, the screen format of the input image data is set and imparted to the evaluation means according to the screen-format designation signal. As such, motion vectors can be detected according to a desired prediction mode for image data of any screen format.
[0086] In the invention of claim 8, the effective stored-pixel count of the buffer means disposed between the processors is variable. As such, a size of a search area serving as a motion-vector search range can be easily changed.
[0087] In the invention of claim 9, the third computation means calculates the evaluation-function values by thinning the evaluation points of the search area to seek the optimal representative evaluation point for each of the plurality of prediction modes, the motion-vector search range of the first computation means is determined, and the first computation means seeks the evaluation-function values using all evaluation points in the determined search area. As such, there is no need to seek evaluation-function values for all evaluation points and motion vectors can be detected according to a plurality of prediction modes at high speed without increasing a device scale.
[0088] In the invention of claim 10, the computation region of the evaluation-function value of the first computation means is determined by the region setting means in correspondence with each of the plurality of predetermined modes and the first computation means calculates the evaluation-function values according to the plurality of prediction modes in the determined computation region such that an optimal motion vector can be determined according to a plurality of prediction modes at high speed without increasing a device scale.
[0089] In the invention of claim 11, the first computation means calculates the evaluation-function values according to the plurality of prediction modes in one computation region by the region setting means. As such, a device scale can be further reduced and optimal motion vectors can be detected according to a plurality of prediction modes.
[0090] In the invention of claim 12, the region setting means determines the optimal prediction mode among the plurality of prediction modes and the computation region, the first computation means calculates the evaluation-function values for the evaluation points in the computation region according to this optimal prediction mode, and the motion vector is detected according to this calculation result. As such, a device scale can be further reduced and a motion vector can be determined for a search area of a wide range at high speed and with high precision.
[0091] In the invention of claim 13, the processors are disposed corresponding to the pixels of the current-image block and the processors outputs can be easily classified according to the plurality of prediction modes. Therefore, calculation of evaluation-function values according to a plurality of prediction modes (an even-field prediction mode, an odd-field prediction mode, and a frame prediction mode) can be executed in a parallel aspect and motion vectors can be detected at high speed without increasing a device scale.
[0092] In the invention of claim 14, the processors are disposed corresponding to the pixels of the current-image block and the outputs of the processors can be easily classified according to the plurality of prediction modes. Therefore, evaluation-function values can be calculated in parallel according to a plurality of prediction modes (an upper-half-block

prediction mode, a lower-half-block prediction mode, and a field prediction mode) and motion vectors can be easily detected at high speed without increasing a device scale.
[0093] In the invention of claim 15, the processors are disposed corresponding to the pixels of the current-image block and the outputs of the processors can be easily classified according to the plurality of prediction modes. Moreover, the classification means classifies the outputs of the processors according to four attributes and the configuration of these four classifications are switched to be changed into three classifications. As such, correspondence is possible even with different prediction modes without changing a device configuration in any way and a highly versatile motion-vector detection device can be realized.
[0094] In the invention of claim 16, the buffer means changes the input/output order of the input image data in imparting to the evaluation means. As such, motion vectors can be detected according to a predetermined plurality of prediction modes for input image data of various screen configurations without changing a device configuration in any way.
[0095] In the invention of claim 17, the effective stored-pixel count of the buffer means disposed between the processors is variable. As such, a size of a search area of motion vectors can be easily changed.
[0096] In the invention of claim 18, the motion vectors for each prediction mode are detected by the first and second calculation means according to the hierarchical search method and according to the plurality of prediction modes and next the optimal motion vector is determined from the motion vectors corresponding to this plurality of prediction modes by the vector determination means. As such, motion vectors can be detected efficiently without increasing a device scale.
[0097] In the invention of claim 19, the optimal evaluation-point candidate is determined by thinning the evaluation points by the first calculation means, the optimal vector is next calculated for each of the plurality of prediction modes by the second calculation means, and finally the motion vector is determined. As such, motion vectors can be detected at high speed without increasing a device scale in any way.
[0098] In the invention of claim 20, the candidate of the optimal evaluation point is calculated by executing the computation of the evaluation-function values by thinning the evaluation points in the motion-vector search area by the first calculation means, the evaluation-function values are calculated for the evaluation points included in the peripheral region thereof according to the plurality of prediction modes by the second calculation means according to this optimal evaluation-point candidate, and the motion vector is detected according to this calculation result. As such, an evaluation-point search region of a second calculation means can used in common with a plurality of prediction modes and motion vectors can be detected at high speed without increasing a device scale.
[0099] In the invention of claim 21, the evaluation points are thinned by the first calculation means and the evaluation-function values are sought according to the plurality of prediction modes, the optimal prediction mode and the corresponding optimal evaluation point are sought according to the outputs of this first calculation means, the evaluation-function values are sought using all evaluation points in the optimal-evaluation-point proximal region according to the optimal prediction mode and the optimal evaluation point, and the motion vector is finally determined from these evaluation-function values. As such, motion vectors can be detected at high speed according to a plurality of prediction modes without increasing a scale of a second calculation means in any way. In the motion-

vector detection device of claim 22, by changing the coefficient A of the predetermined cycle A·(P·(Q+R)), a motion-vector search range can be changed and a motion-vector detection device is obtained that can easily correspond to a plurality of search ranges.
[0100] In the motion-vector detection device of claim 23, the pixel data of different current-screen blocks are stored in the A processor arrays, the pixel data of the current-screen blocks sequentially stored by the processor arrays every processing unit cycle that is A·P(Q+R) cycles are updated overall, and the same reference-screen-block pixel data is imparted to each processor array such that motion vectors can be determined in A processor arrays in a pipeline and a device is obtained that detects motion vectors at high speed and efficiently. In the motion-vector detection device of claim 24, each of the plurality of sub-blocks is disposed corresponding to different current-screen blocks, each sub-block detects the motion vector of the current-screen block, the pixel data of the same reference-screen block is imparted to the sub-blocks in the same column, and the pixel data of the current-screen blocks of the sub-blocks are sequentially updated every processing unit cycle. As such, a large current-screen block can be divided into a plurality of sub-blocks to efficiently detect motion vectors in a pipeline for each sub-block.
[0101] In the motion-vector detection device of claim 25, the pixel data of the first and second register means of the processors are updated every processing unit cycle and the first and second register means are switched by the selection means such that motion vectors for each block of a current-screen image can be sequentially detected in a pipeline. In the motion-vector detection device of claim 26, any one among the vectors near the central portions, the central-portion lower-end portions, and the central-portion upper ends of the respective search ranges of the first to third selection means is selected as the motion-vector candidate according to the operation mode such that this motion-vector detection device can select an optimal motion-vector candidate according to a configuration used alone or in a plurality and a motion-vector detection device that can be realized that can easily correspond to various system configurations.
[0102] In the motion-vector detection device of claim 27, the motion vectors at the fractional precision are further detected for each prediction mode according to the motion vectors of each of the plurality of prediction modes by the integer-precision vector-determination means such that high-precision motion vectors can be detected for each of a plurality of prediction modes. In the motion-vector detection device of claim 28, the effective region of the search region is set and the motion vector is detected according to the evaluation-function values for the reference-scene blocks in this effective region such that motion vectors can be detected using only evaluation-function values for reference-screen blocks necessary for a current-screen block positioned at, for example, a current-screen end and an accurate motion vector can be detected. Moreover, a search range can be restricted for each current-screen block according to a processing usage and a highly-versatile motion-vector detection device can be realized that can set a search range according to a processing usage for each current-screen block.
[0103]
[Examples]
[Motion-Vector Detection Device of Full Search Method] FIG. 1 is a block diagram schematically illustrating an overall configuration of a motion-vector detection device that is one example of this invention. FIG. 1 illustrates a configuration in a situation where pixels are coded in frame units.

[0104] In FIG. 1, the motion-vector detection device includes an input unit 2 that receives search-window (a search window is described below) pixel data Y included in a predetermined region in a search area and template-block pixel data X and respectively outputs the received data X and Y at predetermined timings; a computation unit 1 that calculates three evaluation-function values (difference absolute-value sums) for displacement vectors for one template block based on the data X and Y imparted from the input unit 2; and a comparison unit 3 that receives the three evaluation-function values sought by the computation unit 1—Σ|a-b|, Σo|a-b|, and Σe|a-b|—seeks a smallest evaluation-function value for each evaluation-function value, and determines the displacement vectors corresponding to each smallest evaluation-function value as motion vectors. Here, Σ|a-b| corresponds to an evaluation-function value of a frame-prediction mode of seeking an evaluation-function value using all pixels of a current-image block (template block). a and b represent pixel data corresponding to the template-block pixel data and a reference-image block (search-window block). Σo|a-b| corresponds to an evaluation-function value according to an odd-field prediction mode of detecting a prediction image using pixels included in an odd field. Σe|a-b| represents an evaluation-function value calculated according to an even-field prediction mode of detecting a prediction image using pixels included in an even field.
[0105] The motion vectors determined according to the respective prediction modes—that is, a block motion vector Vb determined according to the frame prediction mode, an odd-field-block motion vector Vo determined according to the odd-field prediction mode, and an even-field-block motion vector Ve determined according to the even-field prediction mode—are output in parallel from the comparison unit 3.
[0106] A configuration of the computation unit 1 is described in detail below, but the computation unit includes a processor array 10 that includes a plurality of element processors disposed in a two-dimensional array corresponding to each pixel of the template block and a summing unit 12 that classifies computation-result values output by each element processor of the processor array 10 (in the present example, difference absolute values) according to prediction mode and adds these for each classification to seek a sum. The evaluation-function values Σ|a-b|, Σo|a-b|, and Σe|a-b| corresponding to each prediction mode are output in parallel from the summing unit 12.
[0107] The element processors included in the processor array 10 store mutually different template-block pixel data, and pixel data of the reference-image block in the search area associated with this template block is sequentially transmitted along one direction via the element processors. The element processors in the processor array execute computation between the pixel data of the template block and the pixel data of one reference-image block (“search-window block”). Outputs of the element processors are output in parallel. Therefore, each element processor calculates a component of the evaluation-function value indicating a correlation between one template block and one search-window block. The template-block pixel data is normally stored in the processor array 10 during a cycle of seeking the motion vectors for this template block. The search-window-block pixel data is shifted by one pixel between the element processors in this processor array 10 every one computation cycle (evaluation-function calculation cycle).
[0108] The element processors are disposed corresponding to each pixel of the template block. In the even-field prediction mode and the odd-field prediction mode, the evaluation-function values are respectively calculated using the pixels belonging to the odd field and

the pixels belonging to the even field of the template block (and the search-window block). A frame includes an even field and an odd field, and odd-field pixels and even-field pixels are alternately disposed every one horizontal scan line on the screen. Therefore, with the outputs of the element processors as well, the outputs of the element processors disposed corresponding to the even-field pixels represent components of the evaluation-function value according to the even-field prediction mode and the output of the element processors disposed corresponding to the odd-field pixels represent components of the evaluation-function value according to the odd-field prediction mode. Therefore, by classifying the outputs of these element processors according to their disposition positions in the processor array, evaluation-function values corresponding to each of the plurality of prediction modes can be calculated. In a situation of the frame prediction mode, these can be obtained by simply adding the evaluation-function values of the even field and the odd field. Next, a specific configuration of the computation unit illustrated in FIG. 1 is described.

[0109] FIG. 2 is a diagram illustrating sizes of the template block and the search area used in this example. A template block 20 includes pixels disposed in Q rows by P columns. A search area 22 has a search range in a horizontal direction of +t1 to -t2 and a search range in a vertical direction of +r1 to -r2. That is, the search area 22 includes $(t2+P+t1)\times(r2+Q+r1)$ pixels.

[0110] FIG. 3 is a diagram schematically illustrating a configuration of an element processor included in the processor array illustrated in FIG. 1. In FIG. 3, an element processor PE includes M cascaded data registers 25-1 to 25-M for storing the template-block pixel data X. The data registers 25-1 to 25-M store mutually different template-block pixel data. The element processor PE also includes N cascaded data registers 26-1 to 26-N for storing the search-window pixel data. N is an integer multiple of (n times) M. Moreover, Q, the number of rows in the template block 20 illustrated in FIG. 2, is an integer multiple of (m times) the number M of the data registers 25-1 to 25-M.

[0111] In the element processor PE, predetermined computation (for example, difference absolute-value computation) is executed using the template-block pixel data stored in the M data registers 25-1 to 25-M. With this computation, the data registers 26-1 to 2-6N for storing the search-window pixel data correspond one-to-one with the data registers 25-1 to 25-M for storing the template-block pixel data (N = M), and the predetermined computation may be executed using the stored data of each of the corresponding registers. Another combination may be used.

[0112] The element processor PE executes computation relating to the M template-block pixel data (“template data”). A computation means in the element processor PE (not illustrated in FIG. 3) is used in a multiplex aspect for these M data registers 25-1 to 25-M. Therefore, in this situation, only one computation means is provided for one element processor PE. By providing a plurality of data registers for template-data storage in the element processor PE, the number of element processors is reduced. Moreover, by selecting and imparting to the computation means the outputs of a plurality of template-data storage registers in a multiplex aspect to execute computation, computation is performed a plurality of times in a transfer period of the search-window-block pixel data (“search-window data”). By this, a plurality of components of the evaluation-function value is output from one element processor PE in a multiplex aspect. As such, the summing unit that adds the outputs of the element processors of the processor array 10 can adopt a small configuration (because a number of input signal lines to the summing unit can be small). As

illustrated by the arrows in FIG. 3, the search-window data Y and the template data X are transmitted only in one direction.

[0113] FIG. 4 is a diagram illustrating the configuration of the processor array 10 illustrated in FIG. 1. In FIG. 4, the processor array 10 includes linear processor arrays LA1 to LAP disposed in P columns. The linear processor arrays LA1 to LAP are provided with the same configuration; each includes m element processors PE1 to PEm disposed in a cascade and a data buffer DL that stores R (= r1+r2) search-window data and also functions as a delay means.

[0114] The element processors PE1 to PEm transmit the search-window data and the template data along one direction (vertical direction in FIG. 4) in the linear processor array LA (generically referring to LA1 to LAP) whereto they belong. When transferring the data to an adjacent linear processor array, the most upstream element processor PE1 imparts the search-window data to the data buffer DL included in an upstream linear processor array (left side in FIG. 4), and the template data is transmitted to the most downstream element processor PEm of the upstream linear processor array. That is, the search-window data Y is sequentially transmitted along one direction in the processor array via the element processors PE and the data buffer DL and the template data is transmitted along the one direction via only the element processors PE. In the configuration illustrated in FIG. 4, the element processors PE are arranged in a matrix of rows and columns; the linear processor arrays receive the search-window data and the template data output by the most upstream element processor of the previous linear processor array. As such, the linear processor arrays LA1 to LAP are connected in a cascade, and the processor arrays configure a systolic array and are essentially equivalent to a one-dimensional processor array.

[0115] As above, the data buffer DL is provided with the delay function or a buffer function and is provided with a function of outputting the imparted search-window data in a first-in, first-out (FIFO) aspect. As the data buffer DL, R data latches with a shift function (R shift registers) may be used or a register file storing R data may be used.

[0116] Now, as illustrated in (A) in FIG. 5, a situation is supposed of dividing an image 36 of one frame into sixty-four macro blocks, 8×8, and detecting a motion vector for each macro block. The hatched macro block in the image 36 of one frame is made to be a template block TB1. For this template block TB1, motion vectors are respectively detected according to the three prediction modes.

[0117] As illustrated in (B) in FIG. 5, it is now supposed that the template block TB1 includes template data a(0,0) to a(15,15) disposed in sixteen rows by sixteen columns. In the template block TB1, the pixels belonging to the even field and the pixels belonging to the odd field are alternately disposed in each row. A method of detecting the motion vector by calculating the evaluation-function values using all template-block pixels a(0,0) to a(15,15) illustrated in (B) in FIG. 5 corresponds to the frame prediction method, and this determines the block motion vector Vb.

[0118] As illustrated in (A) in FIG. 6, a prediction mode of calculating the evaluation-function values using the pixels a(2k,0) to a(2k,15) included in the even field disposed in even rows of a zeroth row, a second row, … , and a fourteenth row in the template block TB1 imparts the even-field-block motion vector Ve. However, k = 0 to 7.

[0119] Furthermore, as illustrated in (B) in FIG. 6, a prediction mode of calculating the evaluation-function values using the pixels a(2k+1,0) to a(2k+1,15) disposed in a first

column, …, and a fifteenth column in the template block TB1 is the odd-field prediction mode and imparts the odd-field-block motion vector Vo.

[0120] For a reference-image search to detect the prediction image, as illustrated in FIG. 7, three macro blocks MB1, MB2, and MB3, illustrated as being hatched, in a previous frame image 35 are stored in the processor array 10. The data of the pixels included in the template block TB1 illustrated in (A) of FIG. 5 is stored in each data register 25-1 to 25-M of the element processor PE. One element processor PE stores Q/m template-block pixel data disposed in the vertical direction. Meanwhile, for the search-window-block image data, one element processor PE stores Q·n/m pixel data adjacent in the vertical direction. Therefore, as illustrated in FIG. 8, an element processor of a processor array stores a total of P·Q search-window pixel data—Q in the vertical direction and P in the horizontal direction. The P·Q pixel data are referred to as search-window-block pixel data in the following description. The remaining R (= r1+r2) · P search-window pixel data are stored in the data buffer DL. A block configured by the pixel data stored in this data buffer DL is referred to as a side-window block 44 in the following description.

[0121] The search-window block 42 and the side-window block 44 illustrated in this FIG. 8 are collectively referred to as a search window 40 in the following description.

[0122] Generally, as illustrated in FIG. 9, one element processor PE stores M template-block pixel data and N search-window pixel data. The element processors PE1 to PEm of one linear processor array store Q search-window pixel data and template pixel data adjacent in the vertical direction. Next, specific operations are described.

[0123] To simplify the following description, the following conditions are supposed.

Template-block size: P = Q = 16

Search range of motion vector: r1 = r2 = 16, t2 = 0, t1 = 15, m = 16, M = N = 1

That is, in the above supposition, as illustrated in FIG. 10, the motion-vector search range 22 includes (16+16+16)×(16+15) = 1,488 pixels in the frame image. The size of the template block 20 is 16×16 = 256 pixels. Moreover, from the condition of M = N = 1, the element processor PE stores one search-window-block pixel data and one template-block pixel data.

[0124] FIG. 11 illustrates a configuration of the search area for the pixels of the odd field or the even field under the above supposition. As illustrated in FIG. 11, the template block of the frame configuration includes the pixels of the even field and the pixels of the odd field. Therefore, a sub-template block 20a configured by the pixels of the even field or the odd field is provided with a configuration of sixteen pixels × eight pixels. Likewise, because the search area also includes the pixels of the even field and the pixels of the odd field, for the pixels of the even field or the odd field, a sub-search area 22a is (8+8+8)×(16+15) = 744 pixels. That is, the displacement vectors differ between a prediction mode in frame units and a prediction mode in field units (because a number of rows of macro blocks in frame units is twice a number of rows of macro blocks in field units).

[0125] FIG. 12 illustrates data storage conditions in a processor array in a first operation cycle in a motion-vector detection operation. In FIG. 12, the pixel data of a template block 43 made of pixel data of sixteen rows × sixteen columns is stored in each corresponding element processor PE in the processor array. In correspondence therewith, the pixel data of a search-window block 42 of sixteen rows × sixteen columns is stored in each element processor PE in the processor array. In the template block 22, this state corresponds to a frame displacement vector (0,-16). Here, a frame displacement vector refers to a

displacement vector used when performing prediction operations according to the frame prediction mode. For the odd-sub-template block 20a configured by the pixels of the odd field, a displacement vector for the pixels of the odd field in this search-window block 42 is used. As such, this corresponds to an odd-field displacement vector (0,-8). Likewise, with the even-sub-template block 20a including only the pixels included in the even field, this corresponds to a displacement vector for the even field—that is, an even-field displacement vector (0,-8).

[0126] In this state, difference absolute values, which are the components of the evaluation functions between the template-block pixel data and the corresponding search-window-block pixel data stored in each element processor PE, are sought. The difference absolute values sought in each element processor PE are transmitted to the summing unit 12 illustrated in FIG. 1. The summing unit 12 calculates in a separate and parallel manner a sum of the difference absolute values transmitted from the element processors corresponding to the odd-sub-template block (odd-sub-block evaluation-function value) and a sum of the difference absolute values transmitted from the element processors disposed corresponding to the even-sub-template block (even-sub-block evaluation-function value). After calculating this even-sub-block evaluation-function value and odd-sub-block evaluation-function value, these are added. This adding computation corresponds to seeking a sum of the difference absolute values transmitted from all element processors of the processor array, and from this, an evaluation-function value according to the frame prediction mode corresponding to the template block (block evaluation-function value) is sought. By the above operations, the evaluation-function values for each displacement vector according to the three prediction modes are sought. Each prediction mode merely uses different pixels; as such, by classifying outputs of the element processors PE, the evaluation-function values for the plurality of prediction modes can be calculated in parallel.

[0127] Next, in a state where the template-block pixel data is held in each element processor PE in the processor array, one pixel's worth of only the search-window-block pixel data is transferred.

[0128] In this state, as illustrated in (A) in FIG. 13, the data in the top row in the search-window block 42 is transferred to the data buffer DL of the adjacent column, and accordingly, the first data of this search-window block 42 is shifted out. In parallel with this shifting out, the search-window-block pixel data is newly input. The shifted-out search-window-block pixel data and the newly shifted in search-window pixel data are illustrated by the crosshatched regions in (A) in FIG. 13.

[0129] In this state, as illustrated in (B) in FIG. 13, the pixel data of a search-window block 42a shifted one row down in the search window 40 is stored in the element processor PE in the processor array 10. That is, as illustrated in (B) in FIG. 13, in each of the linear processor arrays LA1 to LA16, one pixel's worth of the search-window pixel data is shifted upward. In the linear processor arrays LA1 to LA16, the newly shifted in search-window pixel data is stored in the data buffer DL. The data newly shifted into the processor group (PE group) is imparted from a corresponding data register. That is, in the element processor group (PE group), by this one-pixel shifting operation, a search-window block shifted downward by one pixel row of the search window is stored.

[0130] In this state, as illustrated in (C) in FIG. 13, a block displacement vector between the template block 43 and the search-window block 42a is (0,-15).

[0131] In this shifting operation, with the sub-window block 42a, one row of the pixels of the even field is shifted in and shifted out. That is, for the even field, the displacement vector thereof moves by (0,-1). For the odd field, this is included in the search-window block 42a, and the displacement vector for this odd field does not change. Therefore, as illustrated in (A) in FIG. 14, an even-sub-template block 43a (including processors in the hatched region) becomes a displacement vector (0,-8) for an odd field OF.
[0132] Meanwhile, as illustrated in (B) in FIG. 14, an odd-sub-template block 43b (hatched region) becomes the displacement vector (0,-7) for an even field EF. In this state, the same summing computation of the difference absolute values as above is again performed, and the evaluation-function value of the template block, the evaluation-function value of the even-sub-template block, and the evaluation-function value of the odd-sub-template block are output. As the evaluation-function value of the even-sub-template block 43a, therefore, the evaluation-function value for the even field and the evaluation-function value for the odd field are alternately output. Similarly, with the odd-sub-template block 43b as well, the evaluation-function value for the odd field and the evaluation-function value for the even field are alternately output. In the even-field prediction mode, the pixels of this even-sub-template block are used, the motion vectors are sought from both the odd field and the even field. Likewise, the motion vectors are detected from both the even field and the odd field for the odd-sub-template block as well.
[0133] The above operations are repeated, and when the displacement vector for the template block 43 becomes (0,0), for the even-sub-template block 43a, it is the displacement vector (0,0) for the even field and for the odd-sub-template block 43b, it is the displacement vector (0,0) for the odd field. In this state, as illustrated in (A) in FIG. 15, the search-window block 42b in the search window 40 corresponds to directly behind the template block 43. In this state, as illustrated in (B) in FIG. 15, the data of the search window 40 stored in the processor array is in a position shifted to the right by one column in the diagram in a region of sixteen pixels × sixteen pixels thereabove. The pixel data of the search-window block 42b and the template block 43 are stored in the element processor group, and the pixel data of the remaining side-window block 44 is stored in the data buffer group. In this manner, by shifting out one pixel of the data that is unnecessary for the operation of calculating the evaluation-function values and newly shifting in one pixel's worth of the search-window pixel data in parallel with this shifting-out operation, the search-window pixel data of the next column can be newly stored in parallel with the evaluation-function calculation operation.
[0134] Furthermore, the operations for calculating the evaluation-function values are repeatedly executed, and when the displacement vector for the template block 43 enters a state of (0,+16), for the even-sub-template block 43a, it is the displacement vector (0,+8) for the even field and for the odd-sub-template block 43b, it is the displacement vector (0,+8) for the odd field. In this state, as illustrated in (A) in FIG. 16, a search-window block 42c is disposed in a lowermost region of the search window 40. After the evaluation-function values for the displacement vectors of each of the three prediction modes are calculated in this state, one pixel's worth of search-window data PY1 that is become unnecessary is shifted out from the processor array and search-window pixel data PY2 is newly shifted in.
[0135] In this state, as illustrated in (B) in FIG. 16, the pixel data of a hatched region 50 is stored in the element processors in the processor array. At the next step, the evaluation-

function value for the search window shifted to the right in the diagram by one column in the search area is calculated. To prepare the pixel data of the search window necessary for this next step in the processor array, after shifting in the search-window pixel data PY2, the shifting operation is repeated fifteen more times. By this, in the hatched region 50 illustrated in (B) in FIG. 16, the fifteen pixel data disposed in the same column as the pixel data PY1 are sequentially shifted out, and in parallel therewith, the fifteen pixel data following the pixel data PY2 are shifted in. In the data buffer group, the pixel data of the thirty-two rows × sixteen columns illustrated in (B) in FIG. 16 are stored.

[0136] As illustrated in (A) in FIG. 17, in the region 50, pixel-data groups A1 to G1 are stored in the element processors, and the remaining pixel-data groups A2 to G2 are stored in the data buffer. The linear processor arrays store pixel-data groups indicated by the same letter—such as pixel data A1 and A2, B1 and B2. The data buffer inputs and outputs the data in a first-in, first-out aspect. Therefore, when the sixteen hatched pixels in (A) in FIG. 17 are shifted in, the pixel data A1 is entirely shifted out. In this state, as illustrated in (B) in FIG. 17, the pixel data A2 to G2 stored in the data registers are divided in two—pixel data A2a to G2a belonging to an upper-half block being stored in the element-processor group (PE group) and the remaining pixel data A2b to G2b and the pixel data B1 to G1 being stored in the data-register group (DL group). In the linear processor array LA16, the shifted-in pixel data (hatched) is stored.

[0137] As above, by disposing the element processors and the data registers so linear arrays are substantially configured in the processor array, by simply continuously shifting in the search-window pixel data, as illustrated in FIG. 18, the pixel data of the search window 40a shifted to the right in the diagram by one column can be stored in the processor array and the pixel data of the search-window block 42d can be stored in the element-processor group. In this state, search-window blocks 42d and onward corresponding to the block displacement vector (1,-16) for the template block 43 (for the odd-sub-template block, corresponding to the displacement vector (1,-8) for the odd field and for the even-sub-template block, corresponding to the displacement vector (1,-8) for the even field) are stored in the element processors.

[0138] In this state, the above operations—that is, calculating the difference absolute values and calculating the sum—are again executed to derive the evaluation-function values for each prediction mode.

[0139] When the above operations are executed (16+16+16)×15+(16+16) = 752 times, as illustrated in FIG. 19, the evaluation-function value using a search-window block 42e corresponding to the displacement vector (15,16) for the template block 43 is calculated. This search-window block 42e corresponds to the displacement vector (15,+8) for the even field for the even-sub-template block and corresponds to the displacement vector (15,+8) for the odd field for the odd-sub-template block. That is, for the template block 43, the evaluation-function value for the search-window block 42e in the bottom-right corner of a search area 45 is calculated and calculation of all evaluation-function values for this template block 43 is completed.

[0140] After executing calculation of the evaluation-function value for this search-window block 42e, for all calculated evaluation-function values, in the comparison unit (see FIG. 1), the smallest evaluation-function values for each of the three prediction modes are sought, and the displacement vectors corresponding thereto are determined as the motion vector

for this template block 43, the motion vector of the odd-sub-template block, and the motion vector of the even-sub-template block.

[0141] [Specific Circuit Configurations] FIG. 20 is a diagram illustrating a specific configuration of the element processor. In FIG. 20, the element processor PE includes a data register 25 for storing the template-block pixel data, a data register 26 for storing the search-window pixel data, and a difference-absolute-value circuit 27 that seeks a difference absolute value between the pixel data a and b stored in the data register 25 and the data register 26. The data register 25 transmits the data stored therein along one direction in response to a clock signal φ2, and the data register 26 transfers the pixel data from the adjacent element processor or the data buffer according to a clock signal φ1. As configurations of the data registers 25 and 26, shift registers that transfer and latch data in response to a clock signal can be used. Alternatively, memories with different write paths and read paths can be used as the data registers 25 and 26. As a configuration of the difference-absolute-value circuit 27, an appropriate configuration is used according to expression formats of the pixel data a and b (displaying the two's complement, displaying the absolute value, or the like). A circuit configuration of any format can be used as long as a sign of the difference value is inverted when the difference between the pixel data a and b is sought and this difference is negative.

[0142] FIG. 21 is a waveform diagram illustrating operations of the element processor illustrated in FIG. 20. As illustrated in FIG. 21, in a motion-vector detection cycle of the template block, the clock signal φ2 is not generated and an inactive state (the "L" level in FIG. 21) is maintained. The clock signal φ1 determines a transfer period of search-window pixel data PY. The search-window pixel data PY is stored in the data register 26 and transferred according to this clock signal φ1. No template-block pixel data PX in the data register 25 is transferred, and the data register 25 stores the template-block pixel data a corresponding to this element processor PE. The pixel data b stored in the data register 26 changes sequentially to b1, b2, … every one cycle of the clock signal φ1, and difference absolute values |a-b1|, |a-b2|, … are output from the difference-absolute-value circuit 27 at each cycle. By this configuration, the components of the evaluation-function values are calculated in each element processor during the search-window transfer period.

[0143] Note that in the configuration illustrated in FIG. 21, one data register each is provided to store the template-block pixel data and the search-window pixel data. A configuration may be used where a plurality of data registers is provided and one element processor PE is disposed corresponding to a plurality of template-block pixel data.

[0144] FIG. 22 is a diagram illustrating a configuration of the summing unit 12 illustrated in FIG. 1. In FIG. 22, the summing unit 12 includes a summing circuit 12a that receives all outputs of the element processors PE included in the processor array and seeks a sum thereof; a summing circuit 12b that receives the outputs of element processors PEo disposed corresponding to the pixels of the odd field among the element processors PE included in the processor array (element processors corresponding to the pixels configuring the odd-sub-template block) and seeks a sum thereof; and a summing circuit 12c that receives the outputs of element processors PEe disposed corresponding to the pixels of the even field among the element processors PE included in the processor array (element processors disposed corresponding to the pixels configuring the even-sub-template block) and seeks a sum thereof. The summing circuits 12a, 12b, and 12c operate in parallel and respectively output the evaluation function values Σ|a-b|, Σo|a-b|, and Σe|a-

b|. By providing the summing circuits 12a to 12c in correspondence with each prediction mode and operating these in parallel, the evaluation-function values corresponding to each prediction mode can be output in parallel. Classification of the element processors PE into the even-field element processors PEe and the odd-field element processors PEo is realized simply by wiring connections.

[0145] FIG. 23 is a diagram illustrating another configuration of the summing unit 12. In the summing unit 12 illustrated in FIG. 23, instead of the summing circuit 12a that seeks the sum of the outputs of all element processors PE, provided is an adding circuit 12d that adds the output of the summing circuit 12b that derives the evaluation-function value of the odd-sub-template block and the output of the summing circuit 12c that derives the evaluation-function value of the even-sub-template block. The evaluation-function value Σ|a-b| of the template block is output from the adding circuit 12d. Combining the odd-sub-template block and the even-sub-template block yields the template block. Therefore, adding the odd-sub-template-block evaluation-function value Σo|a-b| and the even-sub-template-block evaluation-function value Σe|a-b| yields the evaluation-function value Σ|a-b| for the template block. The adding circuit 12d is a two-input adding circuit. Therefore, a device scale is significantly smaller than the summing circuit 12a (see FIG. 22) that adds all outputs of the element processors in the processor array to seek the sum thereof. This enables the evaluation-function values of a plurality of prediction modes (template block, odd-sub-template block, even-sub-template block) to be calculated at high speed and simultaneously without increasing the device scale of the summing unit 12.

[0146] [Variation] FIG. 24 is a diagram illustrating a variation of the summing unit illustrated in FIG. 1. In FIG. 24, the summing unit 12 includes a summing circuit 12ba that receives the outputs of the element processor PEo corresponding to the pixels of the odd-sub-template block and performs adding and summing in response to a clock signal φ3, a summing circuit 12ca that receives the outputs of the element processors PEe disposed in correspondence with the pixels of the odd-sub-template block and performs summing computation in response to the clock signal φ3, a summing circuit 12e that receives the output of the summing circuit 12a and the outputs of the element processors PEo disposed corresponding to the pixels of the odd-sub-template block according to a clock signal /φ3 and performs summing, a summing circuit 12f that receives the outputs of the element processors PEe of the even-sub-template block and an output of the summing circuit 12ca according to the clock signal /φ3 and performs summing computation, and an adding circuit 12d that adds outputs of the summing circuits 12e and 12f. An evaluation-function value Σoo|a-b| is output from the summing circuit 12ba. This evaluation-function value is a sum of the difference absolute values of the pixels of the odd field included in the even-sub-template block and the search-window block. The summing circuit 12ca outputs as the evaluation-function value a difference absolute-value sum Σee|a-b| between the pixels of the even field included in the even-sub-template block and the search-window block. The evaluation-function value Σ|a-b| of the template block is output from the adding circuit 12d. Next, the operations of the summing unit illustrated in this FIG. 24 are described with reference to FIG. 25, which is an operation waveform diagram thereof.

[0147] The clock signal φ3 is provided with a period twice as long as the clock signal φ1 that regulates the search-window-pixel-data transfer cycle. The summing circuits 12ba and 12ca are activated and perform the summing computations when this clock signal φ3 is “H” and hold and output the summing-operation results thereof when the clock signal φ3 is “L”

and in an inactive state. The summing circuits 12e and 12f perform the summing computations for the imparted data when an inverted signal (phase is shifted 180°) of the clock signal φ3—that is, the clock signal /φ3—is "H" and continue to hold and output this summation result when the clock signal /φ3 is "L." Next, the operations are described with reference to FIG. 25.

[0148] Now, it is supposed that the outputs of all element processors PE of the processor array are PE1. At this time, the clock signal φ3 becomes "H," the summing circuits 12ba and 12ca incorporate this imparted difference absolute value P1 (PEo and PEe) and perform summing computation, and summing-computation results Σo(1) and Σe(1) are respectively output. At this time, the summing circuits 12e and 12f do not perform summing computations and hold and output the summing-computation results of the previous cycle.

[0149] When the clock signal φ3 falls to "L," at this time, a difference absolute value P2 is imparted from the element processor of the processor array. In this state, the summing circuits 12ba and 12ca do not perform computation and hold the previous summing-computation results. Meanwhile, the summing circuits 12e and 12f execute computations, performing a summing computation of the outputs of the summing circuits 12ba and 12ca and the output P2 of the element processor imparted at this time. Therefore, an evaluation-function value Σo(1)+Σe(1)+P2 is output from the adding circuit 12d.

[0150] Next, when the clock signal φ1 rises to "H," a difference absolute value P3 is output from the element processor. The summing circuits 12ba and 12ca perform the summing computations and respectively output summation values Σo(2) (= Σo(1)+P3o) and Σe(2) (= Σe(1)+P3e). Here, P3o and P3e indicate the outputs of the element processors provided corresponding to the pixels of the odd-sub-template blocks and P3e indicates the output of the element processor disposed corresponding to the pixels of the even-sub-template block. Next, when a difference absolute value P4 is imparted from the element processor, the summing circuits 12e and 12f perform the summing computation and Σo(2)+Σe(2)+P4 is output from the adding circuit 12d. These operations are subsequently repeated.

[0151] That is, the summing circuits 12ba and 12ca perform the summing computation by thinning out one pixel at a time, and the summing circuits 12e and 12f perform the summing computation by compensating for this thinned-out pixel data. Therefore, as illustrated in FIG. 26, the sub-template block (block configured by the hatched region in the template block 43) comes to derive the difference absolute value from the pixel data included in the same field in the search-window block 42. That is, with the odd-sub-template block, the difference absolute-value sum is sought using only the pixels belonging to the odd field in the search-window block, and with the even-sub-template block, the difference absolute-value sum is computed using only the pixels of the even field in the search-window block. In this manner, (a) motion-vector detection of using all pixel data of the template block and all pixel data in the search area in a situation of frame units, (b) a mode of determining the motion vector using, for example, only the pixels belonging to the odd-sub-template block and the odd field in the search area, and (c) a mode of detecting the motion vector using only the pixels belonging to the even-sub-template block and the even field in the search area can be realized.

[0152] [Comparison Unit] FIG. 27 is a diagram schematically illustrating a configuration of the comparison unit illustrated in FIG. 1. In FIG. 27, the comparison unit 3 includes a comparison circuit 3a for seeking the motion vector for the template block—that is, the block motion vector Vb—a comparison circuit 3b for seeking the motion vector for the odd-

sub-template block—that is, the odd-field-block motion vector Vo—and a comparison circuit 3c for seeking the motion vector for the even-sub-template block—that is, the even-field-block motion vector Ve. The comparison circuits 3a to 3c are provided with the same configuration, and FIG. 27 specifically illustrates only the configuration of the comparison circuit for seeking the motion vector for the template block—that is, the block motion vector Vb.

[0153] The comparison circuit 3a includes a comparator 132 that compares the evaluation-function value Σ|a-b| for the template block imparted from the summing unit and the evaluation-function value stored in a register latch 130 and a registers latch 136 that stores a count value imparted from a counter 138 according to an output of the comparator 132. The comparator 132 generates a data-update instruction signal to register latches 130 and 136 in a situation where the evaluation-function value Σ|a-b| newly imparted from the summing unit is less than the evaluation-function value stored in the register latch 130. The counter 138 counts the clock signal φ1 and generates a count value representing a displacement vector. Therefore, in a situation where the evaluation-function value Σ|a-b| newly imparted from the summing unit is less than the evaluation-function value stored earlier in the register latch 130, a storage content of the register latch 130 is updated to the newly imparted evaluation-function value and the register latch 136 stores the count value imparted from the counter 138 at this time. The count value of the counter 138 is imparted in common to the comparison circuits 3b and 3c. This configuration enables the displacement vector that imparts the smallest evaluation-function value in the search area to be detected and the smallest evaluation-function value to be held. The displacement vector value stored in this register latch 136 may be decoded by a decoder 137 and converted into a value representing a specific motion vector Vb. The held data of the register latch 136 may be directly output as the motion vector. Moreover, a configuration may be used of outputting the data stored in the register latch 130 as an evaluation-function value Eb for this motion vector.

[0154] The comparison circuits 3b and 3c only differ in the values of the evaluation-function values respectively imparted from the summing unit, and the operations thereof are the same as those of the comparison circuit 3a.

[0155] [Example 2] FIG. 28 is a diagram schematically illustrating an overall configuration of a motion-vector detection device that is a second example of this invention. In the configuration illustrated in this FIG. 28, an evaluation-function value Σ|a-b| for a template block, an evaluation-function value Σn|a-b| for an upper-half block of this template block, and an evaluation-function value Σl|a-b| for a lower-half block of the template block are output from a summing unit 12. In relation to the configuration illustrated in FIG. 1, the same configuration is had for the summing unit 12 and the only difference is a connection between element processors included in a processor array 10 and summing circuits included in the summing unit 12.

[0156] In the configuration of the motion-vector detection device illustrated in this FIG. 28, as illustrated in FIG. 29, a screen is configured by one field. A template block 43 is divided in two in a vertical direction, being divided into an upper-sub-template block 43u configured by pixels of the upper-half block and a lower-sub-template block 43l configured by the pixels included in the lower-half block.

[0157] Likewise, a search-window block 42 is also divided into an upper-half block 42u and a lower-half block 42l. With the upper-sub-template block 43u, an evaluation-function

value is calculated according to a displacement vector fu with the upper-half block 42u in the search-window block 42u. With the lower-sub-template block 43l, an evaluation-function value for a displacement vector fl for the lower-half block 42l of the search-window block 42 is calculated. Moreover, an evaluation-function value of a displacement vector f for the template block 43 and the search-window block 42 is calculated.

[0158] In the processor array, the element processors respectively seek difference absolute values between corresponding pixel data in the template block 43 and corresponding pixel data in the search-window block 42. Therefore, in one computation cycle, the displacement vectors of the upper-sub-template block 43u, the lower-sub-template block 43l, and the template block 43 are the same.

[0159] By classifying the outputs of these element processors into the upper-sub-template block 43u and the lower-sub-template block 43l, in one cycle, the evaluation-function value for the template block 43, the evaluation-function value for the upper-sub-template block 43u, and the evaluation-function value for the lower-sub-template block 43l can be calculated simultaneously. The configuration of this processor array 10 and a transfer sequence of the pixel data are the same as the first example illustrated in FIG. 1. What is different is a classification aspect of the element processors in the summing unit that is described below.

[0160] FIG. 30 is a diagram illustrating the configuration of the summing unit 12 in this second example. In FIG. 30, the summing unit 12 includes a summing circuit 12g that, among element processors PE included in the processor array 10, adds an output PEu of the element processors disposed corresponding to the upper-sub-template block 43u; a summing circuit 12h that, among the element processors included in the processor array, seeks a sum of an output PEl of the element processors disposed corresponding to the lower-sub-template block 43l; and an adding circuit 12d that adds outputs of the summing circuits 12g and 12h. The evaluation-function value $\Sigma u|a-b|$ for the upper-sub-template block 43u is output from the summing circuit 12g, and the evaluation-function value $\Sigma l|a-b|$ for the lower-sub-template block 43l is output from the summing circuit 12h. The evaluation-function value $\Sigma|a-b|$ for the template block 43 is output from the adding circuit 12d. Classification of the element processor PE into the upper-half block and the lower-half block is realized by wiring connections.

[0161] A configuration of the comparison unit 3 is similar to that of the first example, and the upper-block motion vector Vu, the lower-block motion vector Vl, and the block motion vector Vb are determined according to the evaluation-function values $\Sigma u|a-b|$, $\Sigma l|a-b|$, and $\Sigma|a-b|$.

[0162] The above configuration enables motion-vector detection for the template block, motion-vector detection for the upper-sub-template block, and motion-vector detection for the lower-sub-template block to be executed simultaneously in a coding system of coding in field units and optimal motion vectors to be detected according to a plurality of prediction modes.

[0163] [Example 3] FIG. 31 is a diagram illustrating a correspondence relationship between an element-processor group and pixels of a template block in a motion-vector detection device that is a third example of this invention. FIG. 31 illustrates as one example a situation where a template block 43 is configured by pixels disposed in sixteen rows × sixteen columns. The template block 43 is divided in two along a vertical direction on a screen, first being divided into an upper-sub-template block 43u configured by pixel data of

an upper-half block and a lower-sub-template block 43l including pixels of a lower-half block. In correspondence with the sub-template blocks 43u and 43l, the element-processor group is likewise divided into groups.
[0164] Next, the template block 43 is further divided into pixels of even rows and pixels of odd rows. In FIG. 31, the even rows refer to a zeroth row, a second row, … and a fourteenth row and the pixels of the odd rows refer to pixels of a first row, a third row, … and a fifteenth row. In accordance therewith, element processors are also divided into element processors corresponding to the even-row pixels and element processors corresponding to the odd-row pixels.
[0165] Therefore, the element processors PE are divided into four groups: an element-processor group PE#0U corresponding to the pixels that belong to the upper-sub-template block 43u and are disposed in the even rows, an element-processor group PE#1U corresponding to the pixels that belong to the upper-sub-template block 43u and are arranged in the odd rows, an element-processor group PE#0L corresponding to the pixels that belong to the lower-sub-template block 43l and are disposed in the even rows, an element-processor group PE#1L corresponding to the pixels that belong to the lower-sub-template block 43l and are arranged in the odd rows.
[0166] In a situation where the screen is configured in frame units, an evaluation-function value of an even-field-sub-template block is calculated by the element-processor groups PE#0U and PE#0L and an evaluation-function value of an odd-field-sub-template block is calculated by the element-processor groups PE#1U and PE#1L. In a situation where the screen is configured in field units, an evaluation-function value of the upper-sub-template block 43u is calculated by the element-processor groups PE#0U and PE#1U and calculation of an evaluation-function value of the lower-sub-template block 43l is executed by the element-processor groups PE#0L and PE#1L. Therefore, as illustrated in FIG. 31, classifying the element-processor groups into the four groups of PE#0U, PE#1U, PE#0L, and PE#1L enables motion-vector calculation according to prediction modes defined for each screen configuration by changing a combination form of these element processors according to the screen configuration.
[0167] FIG. 32 is a diagram illustrating an overall configuration of the motion-vector detection device that is the third example of this invention. In FIG. 32, in a processor array 10 included in a computation unit 1, the element processors are divided into the four groups of PE#0U, PE#1U, PE#0L, and PE#1L. Therefore, from the processor array 10, at a time of operation, difference absolute values |a-b|ue, |a-b|uo, |a-b|le, and |a-b|lo are output in correspondence with each group.
[0168] A summing unit 12 includes a classification changing unit 122 that changes the combinations of the group of four difference absolute values imparted from this processor array 10 according to a mode designation signal S and groups these into difference absolute values of template-block units and sub-template-block units and a summing computation unit 120 that performs a summing computation for each type of the difference absolute values imparted from the classification changing unit 122. From the summing computation unit 120, according to the classification of the difference absolute values imparted from this classification changing unit 122, each corresponding evaluation-function value is derived and imparted to a comparison unit 3. The comparison unit 3 outputs a motion vector Vb of the template block and motion vectors Ve and Vo or Vu and

Vl corresponding to the sub-template blocks according to the evaluation-function values imparted from this summing computation unit 120.
[0169] In a situation where the mode designation signal S designates coding in frame units, the classification changing unit 122 generates from these four groups of difference absolute values a difference absolute-value group |a-b| for the template block, a difference absolute value |a-b|o for the odd-sub-template block, and a difference absolute-value group |a-b|e for the even field. When the mode designation signal S indicates coding in field units, the classification changing unit S generates the difference absolute-value group |a-b| in template-block units, a difference absolute-value group |a-b|u from the upper-sub-template block using the upper-half block, and a difference absolute-value group |a-b|l from the lower-sub-template block made of the lower-half pixel group.
[0170] As illustrated in FIG. 32, in the classification changing unit 122, changing the combinations of the four difference absolute-value groups output from this processor array 10 by the mode designation signal S enables motion vectors respectively used in prediction image detection in frame units and prediction image detection in field units to be easily created. One motion-vector detection device can correspond to coding of different screen formats, and a highly versatile motion-vector detection device can be realized.
[0171] [Variation] FIG. 33 is a diagram illustrating a variation of the summing unit illustrated in FIG. 32. In FIG. 33, the summing unit 12 includes summing circuits 120a, 120b, 120c, and 120d that are provided corresponding to each group PE#1U, PE#1L, PE#0U, and PE#0L of element processors of the processor array and seek sums of the outputs |a-b|u0, |a-b|l0, |a-b|ue, and |a-b|le of each corresponding group and the classification changing unit 122 that switches transmission paths of outputs of two of the four summing circuits—that is, the summing circuits 120b and 120c. The classification changing unit 122 includes selectors 122a and 122b that switch the transmission paths of the summing circuits 120b and 120c. The selector 122a selects the output of the summing circuit 120b when the mode designation signal S indicates coding in frame units, and the selector 122b selects the output of the summing circuit 120c when the mode designation signal S indicates coding in frame units.
[0172] The summing unit 12 further includes an adding circuit 120e that adds the output of the summing circuit 120a and an output of the selector 122a, an adding circuit 120f that adds the output of the summing circuit 120d and an output of the selector 122b, and an adding circuit 120g that adds outputs of the adding circuits 120e and 120f. An evaluation-function value Σo|a-b| of the odd-sub-template block or an evaluation-function value Σu|a-b| for the upper-sub-template block is output from the adding circuit 120e. An evaluation-function value Σ|a-b| for the template block is output from the adding circuit 120g. An evaluation-function value Σe|a-b| for the odd-sub-template block or an evaluation-function value Σl|a-b| of the lower-sub-template block is output from the adding circuit 120f.
[0173] A configuration of providing an input unit in the summing unit 12 with the summing circuits 120a to 120d in correspondence with each group of element processors and switching the transmission paths of the outputs of two summing circuits by the mode designation signal imparts the following advantages. In a situation of simply realizing the configuration of the summing unit illustrated in FIG. 32 and switching the transmission paths of the outputs of the element processors according to the mode designation signal, a need arises of providing a selector for switching this transmission path to each element processor. As a result, a number of selectors increases, increasing the device scale.

Meanwhile, with each of the summing circuits 120a to 120d, the number of element processors assigned thereto is half of what is assigned to the summing circuits 12b and 12c illustrated in FIG. 23, and the device scale can be decreased. Therefore, essentially, in the configuration illustrated in this FIG. 33, although the number of summing circuits is increased compared to the configuration illustrated in FIG. 23, an occupied area does not increase that much. If selectors for switching the transmission paths of the summing circuits 120b and 120c are provided, the number of selectors that is necessary is two; an element count can be significantly reduced and the device scale can be reduced. The adding circuits 120e, 120f, and 120g are two-input adding circuits; a scale thereof is small compared to the summing circuits, and the area occupied by the device does not increase that much even if these adding circuits 120e and 120f are provided. Therefore, as illustrated in FIG. 33, by adopting the configuration of providing summing circuits corresponding to each group of element processors in the input unit and switching the transmission paths of the outputs of two summing circuits among the four summing circuits, the area occupied by the device can be significantly reduced.

[0174] Note that instead of the configuration illustrated in FIG. 33, a configuration may be used where the selectors 122a and 122b are provided to receive the outputs of the element-processor groups PE#1L and PE#0U and provided are a summing circuit that sums the output of one selector and the output of the element-processor group PE#1U and a summing circuit that sums the output of the other selector and the output of the remaining element-processor group PE#0L. This configuration essentially corresponds to the configuration illustrated in FIG. 23.

[0175] [Example 4] As described above, when search-window data is shifted $(16+16+16)\cdot 15+(16+16) = 752$ times in a processor array (repeating shifting-in and shifting-out), as illustrated in (A) in FIG. 34, all displacement vectors (evaluation-function values) for a search area (search range) for one template block (block N) are sought and a motion vector for this template block TB (block N) can be determined. At this time, as illustrated in (A) in FIG. 34, a region 503a is present in the processor array and search-window pixel data of a search area 502 adjacent thereto is also stored in processors and a data buffer in the processor array. Therefore, when sixteen pixels included in this region 503a are shifted out and sixteen subsequent reference pixels (search-window data) of the region 502 are simultaneously shifted in, in the processor array, as illustrated in (B) in FIG. 34, pixel data of a search-window block 504 of the displacement vector (16,-16) for this template block TB is stored. Therefore, by making $752+16 = 768$ cycles one processing unit, motion vectors can be searched in a range of sixteen displacements in a horizontal direction × thirty-three displacements in a vertical direction. As such, by repeating this processing unit a plurality of times, a horizontal-direction search range can be expanded and motion vectors can be searched in a search area that is widened in the horizontal direction.

[0176] Generally, the processors are respectively stored in a one-to-one correspondence with search-window pixels of Q rows by P columns; in a situation where the data buffer stores R pixel data, a motion-vector search is executed in one unit of the search area in $P\cdot(Q+R)$ cycles. At this time, with the searched-for displacement vectors, a displacement vector per shift-out of one column of the pixel data of the search window is $(R+1)$. As such, the displacement vector of $P\cdot(R+1)$ can be evaluated. In this situation, the search range in the horizontal direction becomes P. Moreover, as illustrated in FIG. 35, in a situation where

a size of a search-window block 506 is made to be P columns by Q rows, a size of a search window 505a is made to be (Q+R)·P, and the search area is made to be regions 505a and 505b, to evaluate all displacement vectors in these search areas 505a and 505b, the search-window pixel data needs to be shifted (P-1)·(Q+R)+R times and reference-image pixel data (search-window pixel data) needs to be shifted. Afterward, by shifting-in the reference-image pixel data R more times, the reference-image pixel data included in a region 505c is stored in the processor array. Therefore, to store the pixel data of the search window of the region 505c in the processor array, a total of (P-1)·(Q+R)+R = P·(Q+R) cycles are necessary. Because (R+1) displacement vectors are evaluated in each column of the search area, the number of evaluated displacement vectors becomes P·(R+1).

[0177] FIG. 36 illustrates a specific example of the search range in the horizontal direction. In FIG. 36, in case (I), the horizontal-direction search range is -8 to 7. In a situation of case (II), the horizontal-direction search range is -16 to +15. In this situation, in one processing unit cycle, a motion-vector search in a range of -16 to -1 in the horizontal direction is performed and in another processing unit cycle, a motion-vector search is performed for a range of 0 to +15 in the horizontal direction. In case (III), the horizontal-direction search range is made to be -32 to +31. In this situation, four processing unit cycles are necessary, and a motion-vector search is performed for horizontal-direction ranges of -32 to -17, -16 to -1, 0 to +15, and 16 to +31 in each processing unit cycle.

[0178] FIG. 37 is a diagram illustrating a motion-vector-search operation sequence in various horizontal search ranges. FIG. 37 illustrates as one example -8 to +7, -16 to +15, and -32 to +31 as the horizontal search ranges. As illustrated in (A) in FIG. 37, in a situation where the horizontal search range is -8 to +8, the template block is updated every processing unit time (768 cycles) T. That is, a motion vector of the template block is determined every processing unit time T. (A) in FIG. 37 illustrates a state where motion vectors are sequentially determined for template blocks N, N+1, …. In a situation where the horizontal search range is -16 to +15, as illustrated in (B) in FIG. 37, two processing unit times, 2T, are necessary. In 2T, a motion-vector search is performed in a range of -16 to -1 in the horizontal direction in the first processing unit time and a motion-vector search is performed from 0 to +15 in the horizontal direction in the remaining T. The motion vectors for one template block are determined every 2T.

[0179] As illustrated in (C) in FIG. 37, in a situation where the horizontal search range is -32 to +31, a motion vector for a horizontal range of 16 is detected in one processing unit time T. As such, a processing time of 4T is necessary for one template block. As above, in a situation where a vertical search range is -16 to +16, by making the processing unit time 768 cycles, motion vectors can be searched in a range of sixteen horizontal displacements by thirty-three vertical displacements; grasping this processing unit time enables the horizontal search range to be easily expanded. Note that FIG. 37 does not illustrate an update cycle count of the template block. Only a processing time necessary for motion-vector determination for simply one template block is considered.

[0180] FIG. 38 is a diagram illustrating an overall configuration of a motion-vector detection device used in this fourth example. As described with the previous example, a motion-vector detection device 600 includes an input unit 2 that inputs the search-window pixel data and the template-block pixel data, a computation unit 1 that subjects the search-window pixel data and the template-block pixel data imparted from the input unit 2 to predetermined computation processing and generates an evaluation-function value for the

displacement vector, and a comparison unit 3 for determining a motion vector for this template block according to the evaluation-function value from the computation unit 1. In FIG. 38, the evaluation-function values and the motion vectors are respectively generated according to three prediction modes, but illustrated is only one motion vector being generated. This motion-vector detection device 600 further includes a reference-image memory 602 that stores reference-screen-image pixel data; a current-image memory 604 that stores current-screen pixel data; a read control circuit 606 that controls pixel-data read operations of the reference-image memory 602 and the current-image memory 604; a comparison control circuit 608 that controls a comparison operation of the comparison unit 3—that is, a motion-vector detection operation—and a search-range setting unit 607 that sets the search range in the horizontal direction of this motion vector and imparts the set horizontal-direction search-range information to the read control circuits 606 and 608.
[0181] The comparison control circuit 608 sets the motion-vector detection operation cycle in the comparison unit 3 according to the horizontal-direction search-range setting information from this search-range setting unit 607. The read control circuit 606 sets a read region of the pixel data from the reference-image memory 602 also according to the horizontal-direction search-range setting information from this search-range setting unit 607 and adjusts a current-image-pixel-data read timing from the current-image memory 604. FIG. 39 is a diagram illustrating one example of a configuration of the comparison unit 3 illustrated in FIG. 38. In FIG. 39, the comparison unit 3 includes a register 610 that stores an evaluation-function value of a motion-vector candidate, a comparator 611 that compares the candidate evaluation-function value stored in the register 610 and the evaluation-function value imparted from the computation unit 1, a counter 613 that generates values indicating the displacement vectors of each evaluation-function value, and a register 612 that stores a counts value of the counter 613 as the motion-vector candidate according to an output signal of the comparator 611. When the comparator 611 indicates that the evaluation-function value imparted from the computation unit 1 is less than the evaluation-function value stored in the register 610, contents of the registers 610 and 612 are updated. That is, the register 610 stores the evaluation-function value imparted from the computation unit 1 at this time and the register 612 stores the count value imparted from the counter 613.
[0182] The counter 613 counts a clock signal φ imparted from the comparison control circuit 608 and generates displacement-vector information, and a count value thereof is reset according to a reset signal Re generated when the processing cycle ends. The comparison control circuit changes a generation timing of the reset signal Re according to the horizontal-direction search-range setting information from this search-range setting unit 607. FIG. 40 is a diagram illustrating operations of this comparison control circuit 608. As illustrated in FIG. 40, from the comparison control circuit 608, the clock signal φ is imparted to the counter 613 for thirty-two cycles and the counter 613 performs a counting operation. In the next sixteen cycles, the clock signal φ is not generated, and an inactive state is entered into. That is, this is for relatively moving the template block that is relatively moved to a lowest portion of the search window back to an upper (-16 in the vertical direction) position. The shifting operation is repeated 752 times with these forty-eight cycles as one subunit. By these 752 cycles, a motion-vector search is performed for sixteen displacements in the horizontal direction. In the next sixteen cycles, generation of the clock signal φ is again prohibited (the inactive state is entered into). These 752 cycles +

sixteen cycles correspond to one processing unit cycle T. The comparison control circuit 608 sets the motion-vector search range according to the horizontal-direction search-range information set by the search-range setting unit 607. In this set cycle, an operation performed in units of operation cycles of one column—that is, forty-eight cycles—is repeated.

[0183] FIG. 41 is a diagram illustrating a reset timing of the count value of the counter 613. As illustrated in FIG. 41, in the situation where the horizontal search range is -8 to +7, the reset signal Re is generated (the active state is entered into) every 768 cycles. In the situation where the horizontal search range is -16 to +15, the reset signal Re is generated every 768·2 cycles, and in the situation where the horizontal search range is -32 to +31, the reset signal Re is generated every 768·4 cycles. By varying the generation timing of this reset signal Re to match processing times, the motion-vector search range can be easily changed. As a configuration of the counter 613, a counter representing a position H in the horizontal direction and a counter indicating a position V in the vertical direction may be separately provided or a configuration of simply generating a count value including both H information and V information by a binary counter may be used.

[0184] Note that when comparison operations are prohibited, update operations of the registers 610 and 612 (see FIG. 39) are also obviously prohibited. This may use a configuration of forcibly setting the output of the comparator 611 to a register non-updating state during this comparison-operation prohibition period. The configuration may be such that the comparison operations of the comparator 611 (see FIG. 39) are again prohibited at this time. FIG. 42 is a diagram illustrating one example of a configuration of the comparison control circuit illustrated in FIG. 38. In FIG. 42, the comparison control circuit 608 includes counters 615 and 616 that count a clock signal CLK generated when the pixel data is applied from the reference image memory to the processor array (included in the computation unit 1), a comparator 617 that compares horizontal-direction search-range setting information A from the search-range setting unit 607 and a count-up signal φCUP from the counter 615, and a clock generator 618 that generates a clock signal φ according to a thirty-two-count instruction signal φ32 and a forty-eight-count instruction signal φ48 from the counter 616. The counter 615 generates the count-up signal φCUP when a clock-cycle count included in one processing unit cycle thereof is counted. The horizontal-direction search-range setting information A designates how many times greater the search range in the horizontal direction is than the unit search range. The comparator 617 compares the count-up signal φCUP from this counter 615 and the horizontal-direction search-range setting information A and generates the reset signal Re (enters into the active state) when these match.

[0185] The counter 616 generates the count-up signal φ32 when the clock signal CLK is counted thirty-two times and generates the count-up signal φ48 when the clock signal is counted forty-eight times. The counter 616 has a maximum count value thereof set to 48 and generates the count-up signal φ48 when the pixel data of one column of the search window is processed. When the count-up signal φ32 is imparted from the counter 616, the clock generator 618 sets the clock signal φ to the inactive state until the count-up signal φ48 is imparted next. Until the count-up signal φ32 is generated (the active state is entered into), the clock generator 618 generates the clock signal φ according to (in synchronization with) the clock signal CLK. The clock signal φ from this clock generator 618 controls the counting operation of the counter 613 (see FIG. 39) included in the comparison unit 3.

[0186] By using the configuration of the comparison control circuit 608 illustrated in FIG. 42, the range in the horizontal direction of the motion-vector search range in the comparison unit 3 can be easily controlled. FIG. 43 is a diagram illustrating one example of a configuration of the read control circuit 606 illustrated in FIG. 38. In FIG. 43, the read control circuit 606 includes an address counter 620 that generates a read address of the reference-image memory 602, an address control circuit 621 that sets an initial count value and count range of this address counter 620 according to the horizontal-direction search-range setting information A, an address counter 622 that generates a read address of the current image memory 604, and an address control circuit 623 that sets an operation timing and an initial address of the address counter 622 according to the horizontal-direction search-range setting information A. The address control circuit 621 sets a range of addresses relating to the pixel data in the horizontal direction among the addresses generated from the address counter 620 according to the horizontal-direction search-range information A. The address control circuit 623 generates a pointer indicating a position of the template block and imparts this to the address counter 622. The address counter 622 generates the read address according to the pointer information indicating the position of the template block, and the template-block pixel data is read from the current image memory 604. With the address control circuit 623, a generation timing of the pointer indicating the position of the template block to this address counter 622 is set to A·T according to this horizontal-direction search-range setting information A. Meanwhile, with the address counter 620, a column address range in the reference-image memory 602 is set to be A times greater under control of the address control circuit 621. By this, the reference-image pixel data and the template-block pixel data can be read at respectively desired timings according to the designated horizontal-direction search range.

[0187] Note that as is clear from the operations of the comparison unit illustrated in this FIG. 40, although the horizontal-direction search range can be set as an integer multiple of the processing unit time T, the horizontal-direction search range can also be increased one column at a time. It is also possible to increase or decrease the horizontal-direction search range one column at a time because a motion-vector evaluation operation with thirty-two cycles of operations in each column and a reference-image-pixel-data input operation for vector evaluation of the next column of sixteen cycles are performed and the motion-vector detection operation is performed in units of operations of this one column. In this situation, by making the processing unit one column and making the horizontal-direction search-range information A a value indicating the horizontal-direction search range, the above configuration can be used as-is. As above, according to the fourth example of this invention, the horizontal-direction search range of the motion vector can be changed, an optimal horizontal-direction search range can be set according to processing usage, and a highly versatile image coding system can be easily constructed.

[0188] [Example 5] FIG. 44 is a diagram schematically illustrating an overall configuration of a motion-vector detection device that is a fifth example of this invention. In FIG. 44, the motion-vector detection device includes an A-side computation unit 1a and a B-side computation unit 1b provided in parallel. Both the A-side computation unit 1a and the B-side computation unit 1b include the processor array 10 and the summing unit 12 described in the previous example (see FIG. 1). The motion-vector detection device further includes a comparison unit 3a that receives evaluation-function values eva, evb, and evc generated by the A-side computation unit 1a according to each of three prediction modes

and determines an optimal motion vector for each; a comparison unit 3b that receives evaluation-function values eva, evb, and evc generated by the B-side computation unit 1b according to each of a plurality of prediction modes and determines an optimal motion vector for each; a selector 632 that alternately selects the motion vectors output by the comparison units 3a and 3b and outputs respective motion vectors MVA, MVB, and MVC of the A-side computation unit 1a and the B-side computation unit 1b; and an input unit 630 that alternately imparts template-block pixel data PX imparted from the outside to the A-side computation unit 1a and the B-side computation unit 1b every processing unit cycle and imparts search-window pixel data PY imparted from the outside in common to the A-side computation unit 1a and the B-side computation unit 1b. The A-side computation unit 1a and the B-side computation unit 1b have horizontal-direction search ranges thereof set to -16 to +15. Therefore, the A-side computation unit 1a and the B-side computation unit 1b each require two processing unit times, 2T, to process one template block. Therefore, the input unit 630 alternately updates the template-block pixel data every one processing unit time and the selector 632 switches and outputs the motion vectors output by the comparison units 3a and 3b every one processing unit time T.

[0189] As illustrated in FIG. 45, a search-window block 635 has the displacement vector (0,-16) in relation to a template block N and has the displacement vector (-16,-16) for a template block N+1. The template block N is stored in, for example, the processor array of the A-side computation unit 1a, and pixel data of the template block N+1 is stored in the B-side computation unit 1b. By applying the search-window pixel data PY in common to the A-side computation unit 1a and the B-side computation unit 1b, motion-vector search operations can be performed in parallel for two template blocks. What differs is the motion-vector detection operation starting timing in the A-side computation unit 1a and the B-side computation unit 1b. FIG. 46 illustrates processing sequences of the A-side computation unit 1a and the B-side computation unit 1b. As illustrated in FIG. 46, when a motion-vector search operation for -16 to -1 in the horizontal direction of the template block N is being performed by the A-side computation unit 1a, a motion-vector search operation in a range of 0 to +15 in the horizontal direction is being performed for a template block N-1 by the B-side computation unit 1b. When the motion-vector search operation in 0 to +15 in the horizontal direction for the template block N-1 is completed, a motion vector for this template block N-1 is determined. Meanwhile, the A-side computation unit 1a performs the motion-vector detection operation in 0 to +15 in the horizontal direction for the block N during the next processing-unit time T. The B-side computation unit 1b stores the pixel data of the template block N+1 via the input unit 630 and performs the detection operation of the motion vector for -16 to -1 in the horizontal direction of this template block N+1. Subsequently, the template-block pixel data of the A-side computation unit 1a and the B-side computation unit 1b are alternately updated every processing unit time T. The comparison units 3a and 3b therefore alternately determine and output motion vectors according to each prediction mode every processing unit time T. The selector 632 switches a selection aspect every processing unit time and selects and outputs the motion vectors output from the comparison units 3a and 3b.

[0190] As above, by providing two computation units and two comparison units, storing pixel data of different template blocks in each computation unit, and imparting the same search-window pixel data in common to the computation units, motion-vector detection operations of the template blocks can be performed in a pipeline and high-speed motion-

vector detection can be performed. That is, this is because a motion vector appears to be determined every T even though 2T is actually necessary for motion-vector detection. FIG. 47 is a diagram illustrating operations in a situation where four sets of computation units and comparison units are provided in parallel. In this FIG. 47, the horizontal-direction search range is set to -32 to +31 and a 4T processing time is necessary for motion-vector detection for one template block. Each computation unit stores pixel data of different template blocks, and each computation unit is imparted in common with the same search-window pixel data. In this situation, the processing of the four template-block pixel data is performed in a pipeline and motion vectors are sequentially determined and output from the computation units every processing time T. Therefore, in this situation as well, motion vectors can be output every unit time T even in a situation where the processing time is 4T.

[0191] That is, in a situation where the processing unit time is made to be T and the horizontal-direction search range thereof is increased to be A times as large, by providing A sets of the computation unit and the comparison unit and storing different template-block pixel data in each computation unit, the motion-vector detection operations for the A template blocks can be performed in a pipeline. Note that in an operation of this pipeline aspect, an update time of the template-block pixel data is ignored. By configuring this so waiting is performed for the template-block pixel-data update time every processing unit time and starting/restarting the motion-vector detection operation in all computation units at the same timing after the template-block pixel data is stored in the computation units, the motion-vector-detection operation sequence illustrated in FIG. 46 and FIG. 47 can be easily realized.

[0192] As configurations of the input unit 630 and the selector 632 illustrated in FIG. 44, it is favorable for the input unit 630 to be provided with a configuration where transmission paths of the template-block pixel data are simply sequentially rewritten according to the number of template blocks to be processed and for the selector 632 to use a configuration of sequentially selecting the outputs of the comparison units every processing unit time T; these configurations can be easily realized. Moreover, in this situation, a configuration of reading the template-block pixel data from the current-image memory is not different at all from the situation of updating the template-block pixel data at the processing-unit time T; in a situation of simply outputting the search-window pixel data from the reference-image memory, only the range of reading the search-window pixel data is changed according to this processing time, and this can use the configuration of the fourth example.

[0193] As above, according to the configuration of the fifth example of this invention, because a plurality of sets of computation units and comparison units is provided in parallel, different template-block pixel data are imparted to each set of computation units and comparison units, the search-window pixel data is imparted in common, and the template block is updated every processing unit time for each set, motion vectors can be detected according to a plurality of prediction modes at a high speed. Moreover, because the vector search range in the horizontal direction can be changed even in a situation where a configuration is used of two sets of computation units and comparison units (it is favorable to update the template-block pixels every 2T processing times), the horizontal-direction search range of the motion vector can be easily changed. Moreover, in a situation where the computation unit and the comparison unit are configured by one chip, in a situation where the processing unit time T is set in this one chip, providing A chips in parallel enables easy application to image coding systems with different horizontal-

direction search ranges. This is because it is sufficient to provide the configuration of example 4 in each chip.
[0194] [Example 6] FIG. 48 is a diagram schematically illustrating an overall configuration of a processor array including element processors disposed in sixteen rows × sixteen columns. In FIG. 48, element processors PE are disposed in a one-to-one correspondence with pixels of a template block having pixels disposed in sixteen rows × sixteen columns. That is, the element processors PE include one register that stores search-window pixel data and one register that stores template-block pixel data (in FIG. 3, N = M = 1). In FIG. 48, linear processors arrays LA0 to LA15 disposed corresponding to each column of the template block include sixteen element processors PE and a data buffer DL (DL0 to DL15), each storing thirty-two pixels of data. In FIG. 48, for convenience in description below, each of the linear processor arrays LA0 to LA15 includes an element-processor group PGiU including eight element processors provided on an upstream side (upper portion in FIG. 48) and an element-processor group PGiL including eight element processors provided on a downstream side (however, i = 0 to 15). In the linear processor arrays LA0 to LA15, the element processors are mutually connected to transmit the template-block pixel data one-dimensionally in one direction and the linear processor arrays LA0 to LA15 are mutually connected to transmit search-window pixel data PY one-dimensionally along one direction. That is, the element processors and the data buffers are disposed in a systolic array.
[0195] In a situation of the configuration illustrated in this FIG. 48, operations similar to example 4 described above can be performed. That is, a motion-vector search can be performed with a vertical-direction search range as -16 to +16 for a template block disposed in sixteen rows by sixteen columns. At this time, a search range in a horizontal direction is changeable, as in example 4 above. FIG. 49 is a diagram schematically illustrating an overall configuration of the processor arrays (computation units) when the processor arrays illustrated in FIG. 48 are redisposed in correspondence with a template block made of pixels disposed in eight rows by eight columns. In FIG. 49, the processor arrays (computation units) include four processor blocks PBA to PBD, each including the element processors disposed in eight rows by eight columns. The processor block PBA includes eight columns of processor groups PG8U to PG15U, and the processor block PBB includes processor groups PG0U to PG7U. The processor block PBC includes processor groups PG8L to PG15L, and the processor block PBD includes processor groups PG0L to PG7L.
[0196] The processor arrays include processor blocks disposed in two rows by two columns, and the data buffers DL0 to DL15 are divided in correspondence with each column of processor blocks. That is, the data buffers DL0 to DL7 are provided in correspondence with the processor blocks PBB and PBD and the data buffers DL8 to DL15 are disposed in correspondence with the processor blocks PBA and PBC. Each of the data buffers DL0 to DL15 stores data of thirty-two pixels. All processor groups disposed in the same column input the search-window pixel data from the corresponding data buffer. For example, the processor groups PG0U and PG0L input the search-window pixel data from the data buffer DL0. Each of the processor groups PG0U to PG15U of the processor blocks PBA and PBB provided on the upstream side sequentially transfers the search-window pixel data imparted from the corresponding data buffers DL0 to DL15 via the element processors therein and shifts this out from each most upstream element processor. That is, for the search-window pixel data, each of the processor groups PG0U to PG15U simply

performs an operation of transferring the search-window pixel data in series, and no systolic array is configured. Meanwhile, for the processor blocks PBC and PBD on the downstream side, mutual connections are made to configure a systolic array so the data buffers DL0 to DL7 corresponding to the processor groups PG0L to PG7L transfer the search-window pixel data one-dimensionally only in one direction. That is, the search-window pixel data shifted out from the most upstream element processor in the processor group is imparted to the data buffer of the adjacent column on the upstream side. Likewise, with the processor block PBC as well, the processor groups PG8L to PG15L and the data buffers DL8 to DL15 configure a systolic array that performs a transferring operation one-dimensionally only along one direction for the search-window pixel data.

[0197] For the template-block pixel data, each of the processor blocks PBA to PBD is mutually connected to element processors in a systolic array to perform image-data transferring only along a one-dimensional direction. Each of the processor blocks PBA to PBD is imparted with and stores pixel data of different template blocks. In FIG. 49, to illustrate the different template blocks, template-block pixel data PXA to PXD are illustrated as being respectively imparted to the processor blocks PBA to PBD. Next, the operations are described. A size of the template block is eight rows by eight columns. Moreover, to store the data of the thirty-two pixels, as illustrated in FIG. 50, each of the data buffers DL0 to DL15 adopts eight displacements in the horizontal direction and thirty-three displacements in the vertical direction as the processing unit ($P\cdot(Q+R)$ is the processing unit time).

[0198] Now, as illustrated in FIG. 51, a state is supposed where respective pixel data of template blocks MA, MB, MC, and MD are respectively stored in the processor blocks PBA to PBD and a search-window block 650 is stored. Displacements of the search-window block 650 are a displacement (8,-8) for the template block MA, a displacement (0,-8) for the template block MB, a displacement (-8,-8) for the template block MC, and a displacement (-16,-8) for the template block MD. Therefore, four processing unit times are made to be a processing time for the processor blocks PBA to PBD to each perform motion-vector detection operations in parallel in a pipeline. That is, as illustrated in FIG. 52, motion-vector detection is performed in a search area (search region) of thirty-three displacements in the horizontal direction and thirty-three displacements in the vertical direction.

[0199] Now, the motion-vector detection operation of one search window in one processor block is supposed. Now, as illustrated in (A) in FIG. 53, the search window is provided with pixels disposed in forty rows by eight columns. A state is supposed where the displacement of the uppermost search-window block is, for example, (-16,-16), the shifting operation is performed thirty-two times, and the displacement (-16,+16) is reached. In this situation, in the processor array, only the eight hatched pixels in (B) in FIG. 53 remain as search-window pixel data that receives the motion-vector detection operation. By shifting out the eight pixels indicated by this hatched region, an evaluation value for a motion vector in the next column—that is, (-15,-16)—can be calculated. In each processing unit time of the four processing unit times, the motion-vector detection operations are performed for the horizontal-direction displacements -16 to -9, -8 to -1, 0 to +7, and +8 to +15. The motion-vector detection operations at the different phases of each processing unit time in these four processing unit times are respectively executed in the processor blocks PBA to PBD.

[0200] FIG. 54 illustrates one example of the processing-operation sequence in the processor blocks. The processing unit time T is $(32+8)\times 8 = 320$ cycles. In a period t1,

motion-vector detection for a horizontal displacement of -16 to -9 of a template block N is performed in the processor block PBA. In the processor block PBD, motion-vector detection is performed in a range of a horizontal-direction displacement of +8 to +15 for a template block N-3. In the processor block PBC, motion-vector detection is performed in a range of a horizontal-direction displacement of 0 to +7 for a template block N-2. In the processor block PBD, motion-vector detection is performed in a horizontal-direction displacement of -8 to -1 for a template block N-1. Here, blocks N-3, N-2, N-1 and N, N+1, … are sequentially disposed along the horizontal direction. In the processor block PBA, a motion-vector detection operation is performed in the horizontal-direction displacement of +8 to +15 for the block N at a time t4 after four processing unit times are elapsed and a motion vector for this template block N is determined. In the processor block PBB, a motion-vector detection operation is performed in the horizontal-direction range of +8 to +15 of the template block N-3 at a time t1, and as a result, a motion vector for the template block N-3 is determined. From the next period t2, a motion-vector detection operation is performed for the new template block N+1.

[0201] In the processor block PBC, a motion-vector search operation is completed in the horizontal-direction displacement of +8 to +15 of the template block N-2 at a time t2, and a motion vector is determined for this template block N-2. From the next period t3, a motion-vector search is performed for the new template block N+2. For the processor block PBD, a motion-vector search operation is completed for the template block N-1 at the time t3, and a motion vector is determined for this template block N-1. From the next period t4, a motion-vector detection operation is performed for the new template block N+3. That is, in the processor blocks PBA to PBD, a motion-vector detection operation for a new template block is started by shifting one processing unit time. At the completion of each processing unit time, a motion vector for one template block is determined the processing results of any among the processor blocks PBA to PBD.

[0202] As above, by dividing the processor array in four and making the processing time four processing unit times, motion-vector detection operations can be performed in a pipeline for four template blocks. Note that in the above example, the element processors disposed in correspondence with the template-block pixels disposed in sixteen rows by sixteen columns are divided into four processor blocks, each having eight rows by eight columns of element processors. Generally, in a situation where an element processor corresponds to one pixel, element processors disposed in P columns by Q rows are divided into processor blocks having (P/b) columns by (Q/c) rows of element processors, and in a situation where the processor blocks are disposed in b columns and c rows, in a situation where the data buffers store R pixel data, using $(P/b)\cdot[(Q/c)+R]$ as the processing unit time enables each of the processor blocks to evaluate $b\cdot c\cdot(P/b)\cdot(R+1)$ displacement vectors to detect the motion vectors. At this time, the processing time is $b\cdot c\cdot(P/b)\cdot[(Q/c)+R]$ cycles.

[0203] [Specific Configuration] FIG. 55 is a diagram illustrating a configuration of a principal portion of a motion-vector detection device used in the sixth example of this invention. FIG. 55 illustrates a configuration of a computation unit 642 in the motion-vector detection device. In FIG. 55, the computation unit 642 includes the four processor blocks PBA to PBD and the data-buffer groups DLA and DLB. The processor blocks PBA to PBD are respectively provided with the same configurations as those illustrated in FIG. 49. The data-buffer group DLA includes the data buffers DL8 to DL15 illustrated in FIG. 48 and FIG. 49, and the data-buffer group DLB includes the data buffers DL0 to DL7. To change an

internal configuration of this computation unit 642 according to the size of the template block, selectors Sla to SLg are provided that switch transmission paths of the template-block pixel data and the search-window pixel data according to a control signal output by a connection control circuit 644. Internal configurations of the selectors SLa to SLg are described in detail below. The connection control circuit 644 establishes connections of the selectors Sla to SLg according to a mode designation signal (template-block size designation signal) φM imparted from outside the computation unit for this motion-vector detection device. This computation unit 642 is also provided with a multiplexer 646 that selectively distributes the search-window pixel data PX imparted from the input unit 640 to the processor blocks PBA to PBD under control of the connection control circuit 644. The connection control circuit 644 switches an output path of this multiplexer 646 at each processing unit time (T) when this designation signal φM indicates a template-block size of eight rows by eight columns. By this, the template-block pixel data PX of eight rows by eight columns imparted from the input unit 640 every processing unit time T is sequentially imparted to the processor blocks PBA to PBD every processing unit time. When the mode designation signal φM designates a template-block size of sixteen rows by sixteen columns, the connection control circuit 644 places only one predetermined output of this multiplexer 646 in a selected state and imparts this to the processor block PBC. Imparted to the input unit 640 from the reference-image memory and the current-image memory that are not illustrated are the search-window pixel data PY and the template-block pixel data PX. When the template block has the size of sixteen rows by sixteen columns, from the outside, the template-block pixel data is imparted every four processing unit times (4T) to update the template-block pixel data of the processor blocks PBA to PBD. In a situation where the template block has the size of eight rows by eight columns, the template-block pixel data of eight rows by eight columns is imparted every processing unit time T, and the processor blocks PBA to PBD each sequentially receive an update to the template-block pixel data thereof every processing unit time.

[0204] FIG. 56 is a diagram schematically illustrating a configuration of the selector SLb illustrated in FIG. 55. Although FIG. 56 illustrates switching elements configured by mechanical switches, these switching elements are configured by transistor elements such as transfer gates or transmission gates. Moreover, FIG. 56 only illustrates a configuration associated with a path of transmitting the template-block pixel data. A configuration of the selector unit for transferring the search-window pixel data is illustrated below. The selector block SLb includes a switching element SWau that connects inputs and outputs of two processor groups adjacent in the processor block PBA when a signal /φLA is activated, a switching element SWXa that imparts the template-block pixel data PXA to an input of the processor group 15U (the most downstream element processor of this group 15U) when the signal /φLA is activated, a switching element SWal that connects the inputs and outputs of two processor groups adjacent in the processor block PBC when the signal /φLA is activated, a switching element Wbb that connects the outputs and the inputs of processor groups disposed in the same column of the processor blocks PBA and PBC when a signal φLA is activated, and a switching element SWba that connects the output of the processor group of the processor block PBA to the input of the processor group of one column above of the processor block PBC.

[0205] The signal φLA is placed in an active state when the template-block size of sixteen rows by sixteen columns is designated, and the signal /φLA is placed in the active state

when the template-block size of eight rows by eight columns is designated. The processor group PG8U is further provided with a switching element SWd that is placed in a conductive state when the signal /φLA is activated. An output of the processor group PG8U is also connected to an input of the processor group PG7L of the processor block PBC via the switching element SWbb. The processor group PG7U of the processor block PBB and the processor group PG7L of the processor block PBC are also provided with switching elements SWXc and SWXd that respectively transmit the template-block pixel data PXB and PXD in response to the signal /φLA. The switching elements SWXc and SWXd are placed in a conductive state when the signal /φLA is activated.

[0206] An output unit of the processor group PG8U is further provided with the switching element SWd that becomes conductive in response to the signal φLA. A switching element configured similarly to this switching element SWd is also provided to an output unit of the processor group PG8L (illustrated in FIG. 56). This switching element SWd is provided to directly impart this template-block pixel data to the next circuit in a situation of further processing the template-block pixel data at the next circuit. As illustrated in this FIG. 56, by providing a switching element to each input unit and output unit of the processor groups, the dispositions of the element processors can be easily changed according to the size of the template block. That is, according to the size of the template block, the processor groups can be connected in a systolic array in each processor blocks PBA to PBD or one systolic array can be configured by the processor blocks PBA to PBD overall.

[0207] FIG. 57 is a diagram schematically illustrating a configuration of a portion associated with transferring search-window pixel data of the selector illustrated in FIG. 55. Although FIG. 57 also illustrates the switching elements configured by mechanical switches, these are also configured by transistor elements. The selector SLa is respectively provided to the output units of the processor groups PG8U to PG15U included in the processor block PBA and includes a switching element STa that transmits the pixel data output by a corresponding processor group to an output node OTN in response to the signal /φLA and a switching element STb that transmits the pixel data output by the processor groups PG8U to PG15U to the input units of the data buffers DL7 to DL14 one column upstream in response to the signal φLA. Like the signal /φLA described in FIG. 56 above, the signal /φLA is placed in the active state when the template-block size of eight rows by eight columns is designated and places the switching element STa in a conductive state. The signal φLA is placed in the active state when the template-block size of sixteen rows by sixteen columns is designated and places the switching element STb in the conductive state.

[0208] The selector SLb includes a switching element STc that connects the outputs and inputs of processor groups disposed in the same column in the processor blocks PBA and PBC when the signal φLA is activated and a switching element STd that connects the outputs of the processor groups PG8L to PG15L to the inputs of the data buffers DL7 to DL14 provided one column upstream. The selector SLc is provided to the respective output units of the data buffers DL8 to DL15 and includes a switching element STe that is placed in a conductive state when the signal /φLA is activated and transmits the pixel data output from corresponding data buffers DL8 to DL15 to the inputs of the processor groups PG8U to PG15U in the processor block PBA disposed in the same column (the most downstream element processor in the group).

[0209] Here, the output unit of the processor group PG8L is not provided with the switching element STd. This is because the processor group PG8L and the data buffer DL7

are included in processor blocks corresponding to different template blocks such that there is no need to transfer the search-window pixel data from this processor group PG8L to the data buffer DL7. The output unit of this processor group PG8L is provided with a switching element similar to the switching element STa that is placed in the conductive state when the signal /φLA is activated (not illustrated in FIG. 57). In the processor blocks PBB and PBD as well, a configuration similar to the selectors SLa, SLb, and SLc in the processor blocks PBA and PBC is provided. At this time, as the selector SLg, the data buffer DL7 is provided with a switching element that becomes conductive when the signal /φLA is activated and transmits the search-window pixel data PY to the input of this data buffer DL7.

[0210] FIG. 57 illustrates as one example conductive/nonconductive states of the switching elements when the template-block size of eight rows by eight columns is designated. By establishing the conductive/nonconductive states of these switching elements by the signals φLA and /φLA, a processor disposition corresponding to the template-block size can be obtained. Note that in the configuration illustrated in FIG. 57, the switching element STb is illustrated as being included in the selector SLa. However, this switching element STb may be provided in the selector SLb. FIG. 58 is a diagram illustrating a configuration of a motion-vector determination unit used in this sixth example. FIG. 58 illustrates only a configuration of a portion associated with one prediction mode. Fundamentally, the configuration illustrated in FIG. 58 is provided in correspondence with each prediction mode (changes can be made such as sharing several circuit portions).

[0211] In FIG. 58, the motion-vector determination unit is provided in correspondence with each processor block PBA to PBD and includes summing units 650a to 650d that derive evaluation-function values by respectively seeking a sum of evaluation-function-value components imparted from the corresponding processor blocks and one-input, two-output selectors 652a to 652d provided corresponding to each summing unit 650a to 650d. Each of selector 652a to 652d transmits the evaluation-function value imparted from the corresponding summing unit to a first output thereof when the signal φLA is in the inactive state and the template-block size of eight rows by right columns is designated and transmits the imparted evaluation-function value to a second output when the signal φLA is in the active state and the template-block size of sixteen rows by sixteen columns is designated. The motion-vector detection unit further includes comparison units 654a to 654d disposed corresponding to the respective first outputs of the selectors 652a to 652d, an adding unit 655 that adds the evaluation-function values imparted from the respective second outputs of the selectors 652a to 652d, a comparison unit 656 that determines a motion vector of the template block of sixteen rows by sixteen columns by comparing the output data of the adding unit 655, and a selector 658 that selects the output data of the comparison units 654a to 654d and 656 according to a signal φLAT. The comparison units 654a to 654d and 656 each select the smallest value of the imparted evaluation-function value and output data mvA to mvD and mvL indicating motion vectors for the corresponding template blocks.

[0212] A configuration may be used where the comparison units 654a to 654d enter the inactive state when the signal φLA is activated—that is, when the template-block size of sixteen rows by sixteen columns is designated—and only the comparison unit 656 is placed in the active state. Moreover, the configuration may be such that when the signal φLA is in the inactive state and the template-block size of eight rows by eight columns is designated

operations of the adding unit 655 and the comparison unit 656 are prohibited. The selector 658 receives the motion-vector data output by the comparison units 654a to 654d and 656. The signal φLAT imparted to the selector 658 is a multi-bit signal. By this signal φLAT, the selector 658 selects and outputs the motion-vector data output by the comparison unit 656 by selecting mvL when the template block of sixteen rows × sixteen columns is designated. When the template-block size of eight rows by eight columns is designated, the selector 658 sequentially selects and outputs every processing unit time the motion-vector data mvA to mvD output by the comparison units 654a to 654d. The multi-bit signal φLAT imparted to this selector 658 is imparted from an external control device. This external control device generates the mode designation signal φM illustrated in FIG. 55, reads and imparts to the computation unit the search-window pixel data and the template-block pixel data at respectively predetermined timings from the reference-image memory and the current-image memory, and controls selection operations in this selector 658. An overall configuration of the motion-vector detection device provided with this external control device is a configuration similar to the configuration described above with reference to FIG. 38 and is not illustrated here.

[0213] [Variation] Now, as illustrated in (A) in FIG. 59, a situation is supposed where the pixels are subsampled in both the horizontal and vertical directions in the template block having the size of sixteen rows by sixteen columns. That is, in (A) in FIG. 59, component derivation of the evaluation-function values is performed for pixels indicated by the symbol ○ and no evaluation-function-value components are derived for pixels indicated by the symbol ×. In this situation, by extracting only pixels that derive the evaluation-function-value components, as illustrated in (B) in FIG. 59, the template block made of pixels disposed in eight rows by eight columns is obtained. With the search-window data as well, subsampling is performed along both the horizontal and vertical directions. A search window 660 is provided with pixels disposed in forty rows by eight columns. The pixels included in this search window 660 are all pixels wherefrom evaluation-function-value components are to be derived. Now, when a displacement vector of a search-window block 661 is (H,V), in the next cycle, as illustrated in (D) in FIG. 59, the displacement vector of a search-window block 662 becomes (H,V+2). That is, this is because in (A) in FIG. 59 the pixels represented by the symbol × are omitted. A change amount of this displacement vector becomes two along the column direction as well. Therefore, in a situation of detecting a motion vector using a subsampling image such as is illustrated in (B) in FIG. 59, a motion-vector search range in the horizontal and vertical directions thereof becomes fifteen displacements in the horizontal direction and sixty-five displacements in the vertical direction such that the motion-vector search range is expanded equivalently. By this, even in a situation of using a subsampling image, the motion-vector search range can be expanded to perform more accurate motion-vector detection.

[0214] Note that this 2:1 subsampling image is, needless to say, expandable to a general subsampling image. Moreover, in a situation of expanding the configurations illustrated in FIG. 56 and FIG. 57 into a general configuration, it is sufficient for the configuration of the processor block PBA to simply be expanded as-is in an upward direction in the diagram. Moreover, in the above configuration, the processor array can be redisposed by a control signal according to the size of the template block, but in a situation where the application usage is fixedly set in advance, the following configuration can be used. That is, only the processor groups and the data buffers are disposed and, according to the usage thereof,

paths for transmitting the search-window pixel data and the template-block pixel data are set by mask wiring. The same chip can be used to easily correspond to a plurality of template-block sizes.

[0215] Furthermore, this is simply dividing the template block of sixteen rows by sixteen columns into template blocks of eight rows by eight columns; by making only the configuration of the motion-vector detection unit the configuration illustrated in FIG. 58, the motion-vector detection operation can be performed simultaneously for four template blocks of eight rows by eight columns (in this situation, the displacement vectors for all the template blocks become the same value). As above, according to the configuration of the sixth example of this invention, because the configuration is such that the disposition of the element processors included in the computation unit can be changed according to the size of the template block, an image coding system with a high degree of freedom can be configured.

[Example 7] FIG. 60 is a diagram illustrating a configuration of a principal portion of a motion-vector detection device that is a seventh example of this invention. FIG. 60 illustrates a configuration of one element processor PE. In FIG. 60, the element processor PE includes a register (A) 660a and a register (B) 660b that each store pixel data of different template blocks, a selector 662 that selects one pixel data output by the register (A) 660a and the register (B) 660b according to a control signal φA and transmits this to an adjacent element processor or data buffer, a multiplexer 661 that imparts template-block pixel data imparted from an adjacent element processor or an input unit in response to the signal φA to one among the register (A) 660a and the register (B) 660b, a selector 663 that selects and outputs the pixel data stored and output by the register (A) 660a and the register (B) 660b in response to the signal φA, a register 664 that stores search-window pixel data imparted from an adjacent element processor or data buffer and transmits this to an adjacent element processor or data buffer in response to a clock signal that is not illustrated, a subtractor 665 that subtracts pixel data a output by the selector 663 and pixel data b output (stored) by the register 664, and an absolute-value computer 666 that takes an absolute value of the output data of the subtractor 665.

[0216] In a situation where the signal φA designates the register (A) 660a, template-block pixel data PX imparted from an adjacent element processor or the input unit is imparted to the register (A) 660a via the multiplexer 661 and the data stored by this register (A) 660a or the output pixel data is selected by the selector 662 and imparted to an adjacent element processor or an output unit. At this time, the selector 663 is in a state of selecting the pixel data stored (or the pixel data output) by the register (B) 660b. A difference absolute value |a-b| is output as an evaluation-function-value component from the absolute-value computer 666, and this is summed in a summing unit that is not illustrated to generate an evaluation-function value. The register (A) 660a, the register (B) 660b, and the register 664 may be provided with a configuration of a shift register provided with a transfer gate therein and may be provided with a configuration of simply latching the data, and a data transfer unit may be configured to be disposed outside the element processor.

[0217] In the configuration illustrated in this FIG. 60, computation is performed using the template-block pixel data stored in one among the register (A) 660a and the register (B) 660b. The pixel data of another template block is imparted to and stored by the other register. That is, during the motion-vector detection operation for one template block, a loading operation of the pixel data of the other template block is performed for the other

register. FIG. 61 is a diagram illustrating one example of a processing operation sequence in the motion-vector detection device including the element processor illustrated in this FIG. 60. Processing operations of the seventh example of this invention are described below with reference to FIG. 61. At a time T1, a computation operation (difference absolute-value computation) is performed using pixel data of a template block N stored in the register (A), and by sequentially shifting search-window pixel data PI stored in the register 664 illustrated in FIG. 60, a motion-vector detection operation for this template block N is performed. That is, the selector 663 illustrated in FIG. 60 is set to a state of selecting the pixel data stored in the register (A) by a signal /φA. The multiplexer 661 and the selector 662 are set to a state of selecting the register (B) by the signal φA. By this, pixel data of a template block (N+1) to be instructed next is stored in the register (B) 660b by sequential shift operations.

[0218] When the period T1 is completed and the motion-vector detection operation of the template block N is completed, at this time, loading of the pixel data of the template block N+1 to be processed next is completed. The multiplexer 661 and the selector 662 are set to a selection state of the register (A) 660a according to the signal φA. The selector 663 is set to a state of selecting the register (B) 660b. In this state, the motion-vector detection operation is performed using the pixel data of the template block N+1 stored in the register (B) 660b. At the time of the motion-vector detection operation, access to a reference-image memory is performed (because the search-window pixel data needs to be shifted into a computation unit). However, a current-image memory storing the template-block pixel data is not accessed. Therefore, during this time, the pixel data of the template block to be processed next can be read from the current-image memory and imparted to the computation unit. That is, in a period T2, the pixel data of the template block N+2 is loaded and this imparted pixel data is sequentially stored in the register (A) 660a.

[0219] Subsequently, in periods T3, T4, T5, and T6, a motion-vector search for a template block N+2 for the register (A) 660a, loading pixel data of a template block N+4, a motion-vector search of the template block N+4, and loading pixel data of a template block N+6 are performed. For the register (B) 660b, in each period T3 to T6, loading pixel data of a template block N+3, a motion-vector search of the template block N+3, loading pixel data of a template block N+5, and motion-vector detection of the template block N+5 are performed. As above, two registers for storing the template-block pixel data are provided in each element processor. While performing motion-vector detection operations using the template-block pixel data stored in one register, the template-block pixel data to be processed next is stored in the other register. This enables input of the pixel data of the template blocks and motion-vector detection processing of the template blocks to be executed in parallel to significantly reduce a processing time.

[0220] FIG. 62 is a diagram schematically illustrating an overall configuration of the motion-vector detection device of the seventh example of this invention. A motion-vector detection device 670 includes an input unit 671 that inputs the search-window pixel data and the template-block pixel data, a computation unit 672 that subjects the template-block pixel data and the search-window-block pixel data imparted from this input unit 671 to predetermined computation to generate an evaluation-function value emv, and a comparison unit 673 that determines a motion vector MV according to the evaluation-function value emv from the computation unit 672. FIG. 62 only illustrates the evaluation-function value emv and the motion vector MV determined according to one prediction

mode among a plurality of prediction modes. The computation unit 672 and the comparison unit 673 respectively detect the evaluation-function value and the motion vector according to the plurality of prediction modes. A control device 674 is provided to control storage of the template-block pixel data in this computation unit 672. The control device 674 alternately places the signals φA and /φA in the active state every unit processing time and switches the register whereto the template-block pixel data is input in the computation unit 672. This control device 674 also controls a data reading operation for a reference-image memory 675 that stores the search-window pixel data and a current-image memory 676 that stores the template-block pixel data. While pixel data in a search area is imparted to the input unit 671 and input to the computation unit 672 from the reference-image memory 675 for one template block under control of this control device 674, the pixel data of the template block to be processed next is read from the current-image memory 676 and imparted to the computation unit 672 via the input unit 671.

[0221] [Variation 1] FIG. 63 is a diagram illustrating a configuration of a first variation of the seventh example of this invention. FIG. 63 illustrates a configuration of a first element processor that receives the template-block pixel data from the input unit 671. However, configurations of the other element processors are identical to the configuration of the element processor illustrated in this FIG. 63. In the element processor PE illustrated in FIG. 63, no multiplexer and selector for distributing the template-block pixel data are provided. Other configurations are identical to the configuration illustrated in FIG. 60, and corresponding portions are labeled with the same reference signs. The multiplexer 675 that transmits the template-block pixel data to one among the register (A) 660a and the register (B) 660b according to the signal φA is provided between the input unit 671 and the first element processor PE. The template-block pixel data output by the register (A) 660a is imparted to the register (A) of an adjacent element processor at the time of the shift operation. Likewise, the template-block pixel data stored in the register (B) 660b is imparted to the register (B) 660b of an adjacent element processor at the time of the shift operation. That is, the multiplexer 675 provided next of the input unit 671 switches the transmission path of the template-block pixel data stored in the register (A) 660a and the path of the template-block pixel data stored in the register (B) 660b. Between the element processors, the register (A) is connected to the register (A) and the register (B) is likewise connected to the register (B). In a situation of the configuration illustrated in this FIG. 63, there is no need to provide a multiplexer and a selector for transferring the template-block pixel data between adjacent element processors in the element processor PE and a scale of the element processor can be reduced.

[0222] Note that in the configuration illustrated in FIG. 63, the register (A) 660a and the register (B) 660b are illustrated as being provided only with a function of simply storing data, but these may be provided with a data shift function like a shift register. In a situation of a shift-register configuration, a clock signal is imparted so the transfer operation is performed only in one register when transferring data, and no clock signal for pixel-data transferring is imparted to the other register.

[Variation 2] FIG. 64 is a diagram illustrating a configuration of a second variation of the seventh example of this invention. FIG. 64 illustrates configurations of the registers 660a and 660b for storing the template-block pixel data of one element processor. Other configurations are identical to those of the element processors illustrated in FIG. 60 and FIG. 63.

[0223] In (A) in FIG. 64, the register (A) 660a includes a transfer gate 676a that transmits the template-block pixel data imparted from a downstream adjacent element processor or the input unit in response to a clock signal φ1A, a latch 677a that latches the template-block pixel data imparted via the transfer gate 676a, and a transfer gate 678a that transmits the pixel data stored by this latch circuit 677a to an upstream adjacent element processor or the output unit in response to a clock signal /φ1A. The register (B) 660b includes a transfer gate 676b that transmits the template-block pixel data imparted from a downstream adjacent element processor or the input unit in response to a clock signal φ1B, a latch circuit 677b that latches the pixel data imparted via this transfer gate 676b, and a transfer gate 678b that transmits the pixel data latched by the latch circuit 677b to an upstream adjacent element processor or the output unit in response to a clock signal /φ1B. The selector 663 that imparts the template-block pixel data to the computation unit (subtractor and absolute-value circuit) is imparted with the pixel data respectively latched by the latch circuits 677a and 677b. When the signal φA designates the register (A) 660a, the selector 663 selects the pixel data latched by the latch 677a. When the signal φA designates the register (B) 660b, the selector 663 selects the pixel data latched by the latch 677b.

[0224] The configurations of the register (A) 660a and the register (B) 660b illustrated in FIG. 64 are essentially shift registers. The input units of these registers 660a and 660b are mutually connected, and the output units thereof are also mutually connected. The clock signals φ1A and /φ1A (or φ1B and / φ1B) are imparted only to registers whereto the template-block pixel data is to be loaded. In a situation where the latched pixel data is subjected to computation processing, both the clock signals φ1B and /φ1B (or φ1A and /φ1A) are made to be at a low level of an inactive state and latches therein are separated from a register exterior. This can prevent the operation of transferring the template-block pixel data of one register from affecting the pixel data stored in the other register. Next, the operations of the element processor illustrated in (A) in this FIG. 64 are described with reference to (B) in FIG. 64, which is an operation waveform diagram thereof.

[0225] When loading the template-block pixel data to the register (A), the signals φ1A and /φ1A are sequentially made into a high level. The clock signals φA and /φA are clock signals of two phases that do not overlap each other. The clock signals φ1B and /φ1B for the register (B) 660b are both set to the low level, and the transfer gates 676b and 678b are both placed in an off state. The selector 663 selects the pixel data latched by the latch 677b and imparts this to the computation unit according to the signal φA. By this, the motion-vector detection operation is performed using the template-block pixel data stored in the latch 677b—that is, the register (B). During this motion-vector detection operation, the clock signals φ1A and /φ1A are sequentially placed in the active state and shift-in/shift-out operations of another template-block pixel data are performed. When the clock signal φ1A is set to the high level, the transfer gate 676a is placed in an on state. Meanwhile, at this time, the signal /φ1A is at the low level and the transfer gate 678a is in an off state. Therefore, the template-block pixel data imparted from the downstream adjacent element processor or the input unit is latched by the latch 677a. Next, the clock signal φ1A is set to the low level, the transfer gate 676a is placed in the off state, the clock signal /φ1A is set to the high level, the transfer gate 678a is placed in the on state, and the pixel data stored in the latch 677a is transmitted to the upstream adjacent element processor or the output unit. This enables the template-block pixel data to be processed next to be sequentially

transferred via the register (A) 660a and the template-block pixel data to be stored in the corresponding register (A) 660a.
[0226] When the motion-vector detection operation is completed using the pixel data stored in the register (B) 600b, at this time, the template-block pixel data to be processed next is already stored in the register (A) 660a. In this state, the clock signals φ1A and /φ1A are both set to the low level and the transfer gates 676a and 678a are both placed in the off state. Simultaneously, the selector 663 is set to a state of selecting the register (A) 660a by the signal φA. By this, a motion-vector detection operation using the template-block pixel data stored in the register (A) 660a is performed. At this time, the clock signals φ1B and /φ1B are now sequentially set to the high level next and the template-block pixel data to be processed next is stored in the register (B) 660b.
[0227] In a situation of the configuration illustrated in this FIG. 64, a register provided with a shift-register configuration is merely being used, and there is no need to provide a multiplexer and a selector for switching the transmission path of the template-block pixel data. Therefore, the scale of the element processor can be further reduced. As above, according to the seventh example of this invention, because two registers that store the pixel data of different template blocks are provided in the element processor and the configuration is such that template-block pixel data to be processed next is stored in (loaded to) the other register in parallel with a motion-vector detection operation using the pixel data stored in one of these registers, template-block pixel data input and the motion-vector detection operation can be performed in parallel, a time required to update the template-block pixel data can appear to be eliminated, and a processing time of motion-vector detection can be significantly reduced.
[0228] [Example 8] FIG. 65 is a diagram illustrating a configuration of a principal portion of a motion-vector detection device of an eighth example of this invention. FIG. 65 illustrates a configuration of a comparison unit that detects a motion vector of the motion-vector detection device. In FIG. 65, the comparison unit includes a register 680 that stores a smallest evaluation-function value among evaluation-function values ev calculated by a computation unit, a comparator 682 that compares the evaluation-function value stored in the register 680 and an imparted evaluation-function value, a counter 684 that generates information representing a displacement vector having the imparted evaluation-function value, a register 686 that stores a displacement vector corresponding to the smallest evaluation-function value as a motion-vector candidate, a priority-order determination circuit 690 that is activated in response to an equivalence indicating signal φeq from the comparator 682 and determines a priority order by receiving a count value output by the counter 684 and the displacement-vector value stored in the register 686, and an update control circuit 688 that updates contents of the registers 680 and 686 in response to a signal φsml indicating that the evaluation-function value imparted from the comparator 682 is small and an update instruction signal φrw from the priority-order determination circuit 690. The update control circuit 688 updates the contents of the registers 680 and 686 when one of the signals φsml and φrw is in an active state.
[0229] The counter 684 may be a normal binary counter or a counter that separately generates count values representing a horizontal-direction vector H and a vertical-direction vector V. In a situation of the counter separately indicating the vector values in the horizontal direction and the vertical direction, the count values are set according to a size of a vertical-direction search range; the count value of the horizontal-direction vector

counter increases by 1 when this vertical-direction vector counter counts up to a predetermined value. Either configuration may be used. The priority-order determination circuit 690 is configured to be able to realize three priority-order determination standards; one of these three priority-range order determination standards is selected according to a usage that is used, and priority-order determination is performed according to the selected priority-order standard.

[0230] FIG. 66 and FIG. 67 are diagrams illustrating a first priority-order determination standard used in the priority-order determination circuit 690 illustrated in FIG. 65 as lists. FIG. 66 and FIG. 67 illustrate a priority-order determination standard in a situation where a horizontal-direction search range is -16 to +15 and the vertical-direction search range is -16 to +16. FIG. 66 illustrates the priority-range determination standard where the H vector in the horizontal direction is 0 or in a negative region, and FIG. 67 illustrates the priority-order determination standard of when the horizontal-direction displacement vector H is a positive value. In the situation of the first priority-order determination standard illustrated in FIG. 66 and FIG. 67, the priority order of a displacement vectors is set according to P(H,V) = |H|+|V|. When evaluation-function values are the same value, the priority order is deemed high when this priority-order evaluation value P(H,V) is small, and the displacement vector having the higher priority order is selected as the motion-vector candidate.

[0231] In the situation of the priority-order determination standard illustrated in FIG. 66 and FIG. 67, in the priority-order determination standard, a region that is directly behind (vector 0,0) is given the highest priority order, the priority order decreasing in moving away from this central portion. FIG. 68 and FIG. 69 are diagrams illustrating a second priority-order determination standard of the priority-order determination circuit illustrated in FIG. 65. In FIG. 68 and FIG. 69 as well, priority-order evaluation values in a situation where -16 to +15 in the horizontal direction and -16 to +16 in the vertical direction are made to be the motion-vector search range are illustrated as lists. In the situation of the priority-order determination standard illustrated in FIG. 68 and FIG. 69, the priority-order evaluation value P(H,V) of the displacement vector (H,V) is represented as P(H,V) = |H|+(-V+16). The smaller the priority-order-evaluation-standard value P(H,V), the higher of a priority order is imparted.

[0232] In the situation of the priority-order determination standard illustrated in FIG. 68 and FIG. 69, the displacement vector (0,16) is imparted the highest priority order, the priority order decreasing in moving away therefrom. That is, a central, lowermost end portion in a search area is given the highest priority order. FIG. 70 and FIG. 71 are diagrams illustrating a third priority-order determination standard of the priority-order determination circuit illustrated in FIG. 65. In FIG. 70, the horizontal direction in this third priority-order determination standard indicates priority-order-determination-standard values in the negative region, and FIG. 71 illustrates the priority-order-evaluation-standard values in the positive region in the horizontal direction of the search range. In this third priority-order determination standard, the priority-order evaluation value P(H,V) is represented as follows:

$$P(H, V) = |H| + (V + 16).$$

Displacement vectors with a smaller value for the priority-order evaluation value P(H,V) are imparted a higher priority order. According to this third priority-order determination standard, the highest priority order is imparted to the uppermost end portion in the

vertical direction in a horizontal-direction central portion (displacement vector where the value of the vertical-direction displacement vector: V vector is the smallest). By providing three determination standards as above as the priority-order determination standards, as below, the search range can be easily expanded in the vertical direction for motion-vector detection.

[0233] (A) in FIG. 72 illustrates a state where the motion-vector detection device is configured by one chip CH1. In this state, the priority orders of the displacement vectors are determined according to the first priority-order determination standard. In this situation, as illustrated in (B) in FIG. 72, in a motion-vector search range SAR, because the priority order is determined according to |H|+|V|, a displacement vector in a central portion thereof (vector (0,0)) is imparted the highest priority order. Therefore, a search-window block directly behind a template block is imparted with the highest priority order. In a situation of expanding the motion-vector search range to be twice as large in the vertical direction, as illustrated in (A) in FIG. 73, two motion-vector-detection-device chips CH2 and CH3 are used. Search-window pixel data PYU of an upper half of the search range is imparted to the motion-vector-detection-device chip CH2, and motion-vector detection for the upper half of this search range is performed in this motion-vector-detection-device chip CH2. Search-window pixel data PYL of a lower half of the search range is imparted to the motion-vector-detection-device chip CH3, and motion-vector detection is performed for the region in the lower half of the search range. The same template-block pixel data PX is imparted in common to the motion-vector-detection-device chips CH2 and CH3. The motion-vector-detection-device chip CH2 performs priority-order determination of the displacement vectors according to the second priority-order determination standard, and the motion-vector-detection-device chip CH3 performs priority-order determination of the displacement vectors according to the third motion-vector determination standard.

[0234] In this situation, as illustrated in (B) in FIG. 73, the motion-vector-detection-device chip CH2 imparts the highest priority order to the displacement vector in the horizontal-direction central portion and the lowermost end portion in the vertical direction in the search range thereof. Meanwhile, the motion-vector-detection-device chip CH3 imparts the highest priority order to the displacement vector in the horizontal-direction central portion and the uppermost portion in the vertical direction in the search range. Therefore, by further determining an optimal vector by a comparison unit CMP illustrated in (A) in FIG. 73 from the motion vectors detected by the motion-vector detection device CH2 and the motion-vector detection device CH3 according to the evaluation-function values thereof, motion-vector detection can be performed where priority-order determination that imparts the highest priority order to the displacement vector in the position directly behind even in an expanded motion-vector search range. Therefore, motion-vector detection can be performed according to a priority order continually centered around the displacement vector (0,0) even when the motion-vector search range is expanded in the vertical direction, and motion-vector selection that designates search-window blocks closer to the template block as prediction image blocks can be performed. Moreover, by using two motion-vector-detection-device chips, even in the situation where the search range in the vertical direction is expanded to be twice as large, by operating the two motion-vector-detection-device chips in parallel, a time required for this motion-vector detection is identical to a processing time necessary prior to expansion, and this enables high-speed processing.

[0235] Here, the motion-vector search range in the vertical direction is not limited to being twice as large and can be easily expanded to three or four times as large. In a situation of expanding the vertical-direction search range of this motion vector, in a situation where the search range in the vertical direction serving as the unit is -α to + α, the second priority-order determination standard is expressed as P(H,V) = |H|+(-V+ α) and the third priority-order determination standard is expressed as |H|+(V+ α). Although it is possible to separately form priority-order determination circuits respectively provided with these three priority-order determination standards according to chip usage, as described below, it is also possible to simply use one of these three priority-order determination standards according to a control signal. FIG. 74 is a diagram illustrating one example of a configuration of the priority-order determination circuit illustrated in FIG. 65.
[0236] In FIG. 74, the priority-order determination circuit 690 includes a selector 701 that selects in a sequentially time-divided manner according to a timing signal φtim the counter 684 that outputs data mv indicating a displacement vector of a search-window block being processed and a vector MVa from the register 686 that stores the motion-vector candidate; an H absolute-value circuit 702 that extracts a horizontal-direction vector component from the vector imparted from the selector 701 and generates an absolute value thereof; a V absolute-value circuit 703 that generates data indicating an absolute value of a vertical-direction component V from the vector data impaired from the selector 701; a V extraction circuit 704 that extracts the vertical-direction vector component V of the vector imparted from the selector 701; and a -V extraction circuit 705 that generates -V, a value where the sign of the vertical-direction component of the vector imparted from the selector 701 is inverted. In a situation where the vector information output by the counter 684 is simply a binary count value, these circuits 702 to 705 may each be provided therein with decoder circuits that generates the H components and V components. Moreover, in a situation where the counter 684 is of a configuration of separately generating the H component and the V component, these circuits 702 to 705 simply receive the corresponding components, perform desired computation processing, and generate necessary data. Moreover, the circuits 703 and 705 may be configured to receive the output of the V extraction circuit 704. In this situation, the V absolute-value circuit 703 is substituted with an absolute-value circuit that receives the data output by the V extraction circuit 704 and the -V extraction circuit 705 is substituted with a sign-inverting circuit that inverts the sign of the data output by the V extraction circuit 704.
[0237] The priority-order determination circuit 690 further includes a selector 706 that selects according to a mode designation signal φmoda one among the vertical-component data V output by the V extraction circuit 704 and the vertical-direction component -V whose sign is inverted that is output by the -V extraction circuit 705, an adder 707 that adds the data output by the selector 706 and a constant value 16 (= α), a selector 708 that selects according to a mode designation signal φmodb one among the data |V| output by the V absolute-value circuit 703 and the data output by the adder 707, and an adder 709 that adds the data output by the selector 708 and the data output by the H absolute-value circuit 702. The selector 706 selects the data -V output by the -V extraction circuit 705 when the mode designation signal φmoda designates the second priority-order determination standard and selects the data V output by the V extraction circuit 704 when the signal φmoda designates the third priority-order determination standard. The selector 708 selects the data |V| output by the V absolute-value circuit 703 when the mode

designation signal φmodb designates the first priority-order determination standard and selects the data output by the adder 707 when the second or third priority-order determination standard is designated.

[0238] The priority-order determination circuit 690 further includes a multiplexer 710 that transmits the output of the adder 709 to one or the other of two outputs thereof according to the timing signal φtim, registers 711 and 712 that are provided respectively corresponding to the one and the other output of the multiplexer 710, and a comparator 713 that compares sizes of the data stored in the registers 711 and 712. The comparator 713 generates the update control signal φrw. This multiplexer 710 stores the output data of this adder 709 in the registers 711 and 712 in a time-divided manner according to the timing signal φtim. That is, when the timing signal φtim is set to a state of selecting the count value of the counter 684, the selector 701 selects the count value of the counter 684. Meanwhile, the multiplexer 710 transmits the output of the adder 709 to the register 712. Meanwhile, when this timing signal φtim is set to a state of selecting the data MVa output by the register 686, the multiplexer 710 stores the data output by the adder 709 in the register 711 according to this timing signal φtim. By this, the register 711 stores the priority-order-determination-standard value of the motion-vector candidate stored by the register 686 and the register 712 stores the priority-order-evaluation-standard-value data of the displacement vector of the search-window block currently being processed output by the counter 684. After the priority-order-evaluation-standard-value data are respectively stored in the registers 711 and 712, the comparator 713 compares the sizes thereof. In a situation where the priority-order-evaluation-standard value of the displacement vector of the search-window block being processed is smaller than the priority-order-evaluation-standard value of the displacement vector of the motion-vector candidate stored in the register 686, the update control signal φrw in the active state is output from the comparator 713.

[0239] FIG. 75 is a diagram illustrating one example of the operation sequence of the priority-order determination circuit illustrated in FIG. 74. FIG. 75 illustrates only one priority-order determination operation cycle. The mode designation signals φmoda and φmodb are respectively set to predetermined states, and the corresponding priority-order determination standards to be followed by the motion-vector detection device are set. When it is indicated that the evaluation-function value of the motion-vector candidate stored by the register 686 and the evaluation-function value of the displacement vector output by the counter 684 are equal, this priority-order determination circuit is placed in the active state. The timing signal φtim is placed in the active state when this active state is entered into. When the timing signal φtim rises to a high level at a time Ta, the selector 701 selects the motion-vector data provided with the smallest evaluation-function value imparted up to this point stored in the register 686. A predetermined priority-order-evaluation-standard-value calculation operation is performed according to the data from this register 686, and the stored data of the register 711 is placed in a confirmed state at the time Ta.

[0240] When the timing signal φtim becomes a low level at a time Tb, the selector 701 selects the data output by the counter 684. The predetermined priority-order-evaluation-standard-value calculation determination operation is performed according to the data output by this counter 684, and this calculated priority-order-evaluation-standard value is stored in the register 712 at the time Tb. When the content of the register 712 is confirmed,

the comparison operation of the comparator 713 is performed at a time Tc, and the state of the update control signal φrw output from this comparator 713 is determined at the time Tc. By generating the priority-order-evaluation-standard value by time-division multiplexing, the device scale can be reduced. FIG. 76 is a diagram illustrating the configuration of a timing-signal generation unit. As illustrated in FIG. 76, a timing-signal generation circuit 715 outputs the timing signal φtim that is placed in the active state / inactive state for a predetermined period in response to the equivalence indicating signal φeq from the comparator 682 illustrated in FIG. 65. This timing-signal generation circuit 715 is configured by, for example, an oscillation circuit (for example, a ring oscillator) that is placed in an operable state in response to this equivalence indicating signal φeq. FIG. 76 also illustrates an active control circuit 716 that generates the active control signal φact that places each component of the priority-order determination circuit in the active state in response to this equivalence indicating signal φeq. By this active control circuit 716, the priority-order determination circuit 690 is placed in an operational state only when the priority-order determination operation is necessary, and this reduces a consumption current.

[0241] Note that in the configuration illustrated in FIG. 74, a configuration may be used that directly imparts the timing signal φtim to the registers 711 and 712 without using the multiplexer 710 provided to the input units of the registers 711 and 712 that store the priority-order-evaluation-standard values (see the configuration of [A] in FIG. 64). As above, according to the configuration of the eighth example of this invention, because the configuration enables the priority orders of the displacement vectors to be changed according to the motion-vector search range, even in a situation where the motion-vector search range is expanded in the vertical direction, the motion-vector detection operation can be performed by making the priority order of the displacement vector (0,0) the highest, and an optimal motion vector can be selected even in an expanded motion-vector search range.

[0242] [Example 9] FIG. 77 is a diagram schematically illustrating an overall configuration of a motion-vector detection device that is a ninth example of this invention. The configuration illustrated in this FIG. 77 illustrates a configuration in a situation of coding pixels in frame units. In FIG. 77, the motion-vector detection device includes an input unit 2 that receives search-window pixel data PY and template-block pixel data PX and outputs the imparted data PY and PX at respectively predetermined timings, an integer-precision computation unit 1 that calculates a plurality of evaluation-function values (difference absolute values) according to a plurality of prediction modes (in the present example, three) based on the pixel data imparted from the input unit 2, and a comparison unit 3 that determines an optimal motion vector for each of the plurality of prediction modes according to the evaluation-function values imparted from this computation unit 1. The integer-precision computation unit 1 is provided with the same configuration as the computation units described above—for example, in FIG. 1. This integer-precision computation unit 1 calculates evaluation-function values for displacement vectors of each search-window block according to a block motion-vector prediction mode, an odd-field-block prediction mode, and an even-field-block prediction mode. The comparison unit 3 determines a displacement vector corresponding to the smallest value of each evaluation-function value for each prediction mode imparted from this integer-precision computation unit 1 as the motion vector of each prediction mode.

[0243] The motion-vector detection device further includes a fractional-precision prediction-image memory 722 that detects a region (prediction-image region) in a search area expressed by the displacement vector imparting the evaluation function of this smallest value in response to the minimum-value detection by the comparison unit 3 and stores a prediction-image region for fractional-precision (half-pixel-precision) evaluation where this prediction-image region is respectively expanded by one pixel in both the horizontal and vertical directions and a template-block memory 724 that stores the template-block pixel data imparted from the integer-precision computation unit 1. The prediction-image memory 722 stores prediction-image-region pixel data for each of the plurality of prediction modes. That is, when sizes of a template block and the search-window blocks are pixels of sixteen rows by sixteen columns, provided in the prediction-image memory 722 are a region that stores pixel data of eighteen rows by eighteen columns including the search-window block designated by the motion vector determined according to the block prediction mode and regions that store pixel data of prediction-image regions of eighteen columns by ten rows surrounding sixteen columns by eight rows respectively corresponding to odd-field pixels and even-field pixels of regions designated by the motion vectors respectively determined according to the odd-field-block prediction mode and the even-field-block prediction mode.

[0244] The motion-vector detection device further includes a half-pixel-precision computation unit 726 that receives the template-block pixel data stored in the template-block memory 724 and the pixel data included in the prediction-image block determined by the block prediction mode stored in the prediction-image memory 722 and performs predetermined computational processing at fractional precision (half-pixel precision) to generate evaluation-function values for each displacement vector at fractional precision (half-pixel precision); a half-pixel-precision computation unit 738 that receives the template-block pixel data stored in the template-block memory 724, the pixel data of the prediction-image region corresponding to the odd-field-block prediction mode stored in the prediction-image memory 722, and the pixel data included in the prediction-image region determined by the even-field-block prediction mode and performs predetermined computation on each at half-pixel precision to generate evaluation-function values; a comparison unit 730 that determines a motion vector at fractional precision according to the block motion-vector prediction mode by seeking the smallest values of the evaluation-function values output from the half-pixel-precision computation unit 726; and a comparison unit 732 that detects the smallest values according to the evaluation-function values from the half-pixel-precision computation unit 738 and respectively detects motion vectors for the odd-field-block prediction mode and the even-field-block prediction mode. The half-pixel-precision computation unit 738 executes in a time-divided manner a motion-vector detection operation at moving fractional precision by the odd-field-block prediction mode and a motion-vector detection operation according to the even-field-block prediction mode. These operations are described in detail below. The motion-vector detection device further receives the motion vectors respectively determined according to the plurality of prediction modes output from the comparison unit 3 and the motion vectors respectively corresponding to the plurality of prediction modes at fractional precision imparted from the comparison units 730 and 732 and outputs a block motion vector VB according to the block prediction mode, a motion vector Vo determined according to the odd-field-block prediction mode, and a motion vector Ve determined according to the even-field-block

prediction mode. A selector 734 is provided so the motion vector Vo according to the odd-field-block prediction mode and the motion vector of fractional precision according to the even-field-block prediction mode are imparted in a time-divided manner from the comparison unit 732.

[0245] As illustrated in FIG. 78, when detecting the motion vectors at half-pixel precision, the integer-precision computation unit 1 calculates evaluation-function values of eight proximal points centered around the motion vectors determined for each of the plurality of prediction modes. The vector imparting the smallest evaluation-function value among the nine proximal evaluation-function values including the evaluation-function values of the motion-vector positions at this integer precision is determined to be the motion vector. That is, the motion-vector determination operation is performed using the nine evaluation points of (-1/2,-1/2), (0,-1/2), (+1/2,-1/2), (-1/2,0), (0,0), (1/2,0), (-1/2,1/2), (0,1/2), and (1/2,1/2). FIG. 79 is a diagram illustrating a distribution of the pixels in the prediction-image region used when performing the motion-vector detection operation at this half-pixel precision. As illustrated in FIG. 79, pixel data included in a region 740 where a prediction-image region 742 designated by the motion vector is respectively expanded by one pixel in positive and negative horizontal directions and positive and negative vertical directions is used. The pixel data for use at the half-pixel computational precision (indicated by × in FIG. 79) is generated from the pixel data stored in this region 740 (indicated by ○ in FIG. 79). When calculating the evaluation-function values of the displacement vectors at half-pixel precision, only the pixel data represented by × in FIG. 79 is used. The evaluation-function values for each of the displacement vectors illustrated in FIG. 78 are generated by the pixel data indicated by × and the template-block pixel data.

[0246] Here, in the situation where the size of the template block includes pixels of sixteen rows by sixteen columns, the region 740 includes pixels of eighteen rows by eighteen columns. During the even-field-block prediction mode and the odd-field-prediction mode, the template block is provided with the size of eight rows by sixteen columns. Therefore, in this situation, the region 740 includes pixels of ten rows by sixteen columns. FIG. 80 is a diagram illustrating a configuration of a memory storing the pixel data of one prediction-image region of the prediction-image memory 722 illustrated in FIG. 77. The description below describes a configuration and operation of a storage device storing the pixel data of the prediction-image region corresponding to the motion vector determined according to the block prediction mode. This can be easily applied to configurations of storage devices storing the pixel data of the prediction-image regions of the even-field prediction mode and the odd-field prediction mode.

[0247] In FIG. 80, the prediction-image memory 722 includes a delay circuit 721 that delays the search-window pixel data PY imparted from the integer-precision computation unit 1 by a predetermined time, a memory 772 that stores the pixel data output from this delay circuit 721, and a write/read control circuit 774 that controls data writing and reading of the memory 772. An output node DO of the memory 772 is connected to the half-pixel-precision computation unit 726 illustrated in FIG. 77. The memory 772 may be a static random-access memory or a dynamic random-access memory. In a situation of a dynamic random-access memory, a high-speed access mode such as a page mode or a static-column mode is used. The write/read control circuit 774 includes a write-address generation circuit 781 that generates a write address to the memory 772, a read-address generation circuit 783 that generates a read address of the memory 772, a control circuit

786 that generates a signal designating a read mode and a write mode of this memory 772, and a selector 784 that selects one among the write address and the read address and imparts this to an address input node A of the memory 772 under control of this control circuit 786.

[0248] The write-address generation circuit 781 generates the write address in response to a latch instruction signal Rφ output by the comparison unit 3 illustrated in FIG. 77. This latch instruction signal Rφ is placed in an active state when, in the comparison unit 3, the evaluation-function value of the motion-vector candidate stored therein and the vector are updated (corresponding to, for example, the update control signal of the update control circuit 688 included in the comparison unit in FIG. 65). When this latch instruction signal Rφ is placed in the active state, the write address of the write-address generation circuit 781 is reset to an initial value. The write-address generation circuit 781 sequentially generates the write address from, for example, a zeroth position in response to a clock signal φck. This clock φck is a signal used to perform a shifting operation of the search-window pixel data in the integer-precision computation unit 1. The read-address generation circuit 783 also sequentially generates the read address from a zeroth position. In a situation where the control circuit 786 is instructing data writing, the selector 784 selects the write address from the write-address generation circuit 781 and imparts this to the memory 772. In a situation where the control circuit 786 is instructing the read mode, the selector 784 selects the read address from the read-address generation circuit 783 and imparts this to the memory 772.

[0249] The control circuit 786 generates a signal that determines a timing of data writing to the memory 772 and a read timing of data in response to the search-window pixel-data transfer clock signal φck. The control circuit 786 also designates the write operation mode and the read operation mode of the memory 772 and generates a chip enable signal (chip select signal) that places the memory 772 in a selected state in response to an operation-mode instruction signal φRW. The operation-mode instruction signal φRW imparted to the control circuit 786 may be imparted from an externally provided controller. Moreover, a count-up signal may be generated when, for example, a counter for displacement-vector-information generation included in the comparison unit 3 performs counting up, and this count-up signal may be used as the operation-mode designation signal φRW. This is because derivation of an evaluation-function value for one template block is being continuously executed until a count value of the counter for displacement-vector-information generation included in the comparison unit 3 reaches a predetermined count value. During a period of the motion-vector detection operation, the write operation to the memory 772 is performed. When the motion vector for one template block is determined, new template-block pixel data is stored in the integer-precision computation unit. As such, in parallel therewith, the pixel data of the search window for the next template block is stored (in a situation where only an evaluation-function value for one template block is generated). Therefore, at this time, a configuration of reading the pixel data from the memory 772 in parallel enables the data of the prediction-image region to be read from this prediction-image memory 722 when loading desired data to the integer-precision computation unit. Next, the operations are described.

[0250] Now, as illustrated in FIG. 81, a situation is supposed where the search window is provided with a size of forty-eight pixel rows by sixteen column pixels and a macro block (the template block and the search-window block) is provided with a size of sixteen pixel

rows by sixteen pixel columns. It is supposed that the evaluation-function-value calculation operation is being performed for the search-window block 742. The region necessary to seek the motion vector of fractional precision for this search-window block 742 is a region 748 of eighteen pixels by eighteen pixels including this region 742. Here, the motion-vector detection operation according to the block prediction mode is described. The region 748 includes search-window pixel data P0 to P325. As illustrated in FIG. 82, the search-window-data transfer clock signal φck is generated at each transfer of the search-window pixel data. When one clock signal φck is generated, one pixel of the search-window pixel data is shifted out. At the time of the evaluation-function-value calculation operation for the search-window block 742, the output data of the delay circuit 721 illustrated in FIG. 80 is data corresponding to the pixel P0. Here, the delay circuit 721 imparts a delay time equal to a time necessary to transfer one column of pixels of a search window 740a. In a situation where the evaluation-function value of the search-window block 742 is the smallest among the evaluation-function values hitherto obtained, the latch instruction signal Rφ (evaluation-function-value update instruction signal) is generated (placed in the active state) from the comparison unit. In response to this latch instruction signal Rφ, the write address of the write-address generation circuit illustrated in FIG. 80 is reset to the initial value 0. The pixel data P0 is written in the address of this reset initial value 0. Subsequently, as long as the signal Rφ is not placed in the active state, data of eighteen pixels—that is, P1, …, and P17—are continuously stored in positions of addresses 1 to 17 of the memory 772. Next, to prohibit writing of unnecessary data, the write-address generation circuit 781 enters a rest state for thirty clock periods—that is, a period of 30 φck—and the memory 772 enters a write prohibition state. The rest period of this writing prohibition uses a configuration of placing the memory 772 in a non-selected state for a 30 φck period after the control circuit 786 counts the clock signal φck eighteen times from the latch instruction signal Rφ in the active state being imparted. However, during this time, the latch instruction signal Rφ is not placed in the active state. In a situation where the latch instruction signal Rφ is placed in the active state in any of these periods, as described above, the write address of the write-address generation circuit 781 is rest to the initial value 0.

[0251] When thirty clock periods (a thirty φck cycle) are elapsed, the write-address generation circuit 781 again generates the write address. The write address at this time is an address 18, and the pixel data P18 is stored in the position of this address 18. In a situation where this pixel data P18 is imparted from the delay circuit 721 (see FIG. 80), the search window 740a is moved to a search window 740b of the next column. When the template block changes, the search area shifts in the positive horizontal direction by one macro-block size (sixteen columns). Therefore, even in a situation where the macro block imparting the motion vector borders the search area, all pixel data necessary for fractional-precision motion-vector detection can be obtained. In this situation, as described below, a configuration may be used that ignores pixel data outside the search area. Moreover, when the template block is changed, likewise, the search-window pixel data is shifted out. At this time, by prohibiting the write operation to the memory 772, after completing the first displacement-vector operation detection of the next template block, top pixel data of one column previous of the search area is output from the delay circuit 721. Therefore, in the delay circuit 721 in particular, the content thereof does not need to be reset when the

template block is updated, and the search-window pixel data output from the integer-precision computation unit can be used as-is.
[0252] By the series of operations above, only pixel data of the search-window block corresponding to the displacement vector serving as the motion-vector candidate is continually stored in the memory 772. In a situation of storing the odd-field pixel data and the even-field pixel data, a similar configuration is used; as illustrated in FIG. 83, clock signals φcka and φckb are generated every one cycle of the clock-signal signal φck. It is sufficient for each to be used as a clock signal for writing the odd-field pixel data and the even-field pixel data. The even-field write-address generation circuit and the write-address generation circuit for the odd-field pixel data are respectively imparted with the reset signal Rφ from the comparison unit performing the motion-vector detection operation according to the odd-field prediction mode and the reset signal Rφ from the comparison unit performing motion-vector detection according to the even-field prediction mode. This configuration enables desired pixel data necessary for motion-vector detection at fractional precision according to each of the plurality of prediction modes to be stored.
[0253] Template-block pixel data sequentially output from the integer-precision computation unit 1 when updating the template-block pixel data thereof is sequentially stored in the template-block memory 724. FIG. 84 is a diagram illustrating one example of a configuration of the half-pixel-precision computation units 726 and 738 illustrated in FIG. 77. Because the half-pixel-precision computation units 726 and 738 are provided with the same configuration, FIG. 84 illustrates only the configuration of the half-pixel-precision computation unit 726. In FIG. 84, the half-pixel-precision computation unit 726 includes a fractional-precision prediction-image generation circuit 802 that receives the search-window pixel data imparted from the prediction image memory 722 and generates a prediction image necessary to calculate an evaluation value of fractional precision, a difference-absolute-value sum circuit 804 that seeks a difference absolute-value sum between the pixel data of the prediction image generated by the fractional-precision prediction-image generation circuit 802 and the template-block pixel data PX imparted from the template-block memory 724, and a comparison unit 806 that detects the displacement vector imparting the smallest difference absolute-value sum among the output values of the difference absolute-value sum circuit 804.
[0254] The fractional-precision prediction-image generation circuit 802 generates a plurality of prediction-image data in parallel. This configuration is described in detail below. The difference absolute-value sum circuit 804 also generates evaluation values for displacement vectors serving as motion-vector candidates in a parallel aspect. The comparison unit 806 determines as the motion vector a displacement vector corresponding to the smallest value among the plurality of difference absolute-value sums imparted from the difference absolute-value sum circuit 804 and the evaluation-function values of motion vectors at integer precision in the comparison unit 3 illustrated in FIG. 77. Next, specific configurations of each circuit illustrated in this FIG. 84 are described. For motion-vector detection at half-pixel precision, as illustrated in FIG. 85, for a focus pixel P, eight proximal pixel data, Q1 to Q4 and Q6 to Q9, need to be sought by interpolation. The search-window pixel data is made to be P1 to P9, a transfer period between these horizontal-direction adjacent column pixels is made to be Th, and a delay time between adjacent rows is made to be Hv (18 Th: in a situation where the search-window-block size at fractional precision is eighteen pixels ×eighteen pixels). The data corresponding to the pixels P1 to P9

illustrated in FIG. 85 are imparted to the fractional half-pixel-precision computation units 726 and 738, and interpolation data Q1 to Q9 are generated therein.

[0255] FIG. 86 illustrates a specific configuration of the half-pixel-precision computation unit illustrated in FIG. 77. The half-pixel-precision computation unit 726 (or 738) includes a delay circuit 835a that delays the imparted search-window pixel data by 1 Hv period and a delay circuit 835b that further delays the data output by this delay circuit 835a by 1 Hv period. These two cascaded delay circuits 835a and 835b form a path that generates the data corresponding to each row illustrated in FIG. 85. The half-pixel-precision computation unit 726 (or 738) further includes a delay circuit 836a that delays input search-window pixel data P by 1 Th period, a delay circuit 836d that further delays the output data of this delay circuit 836a by 1 Th period, a delay circuit 836b that delays the output data of the 1 Hv delay circuit 835a by 1 Th period, a delay circuit 836e that delays the output data of the delay circuit 836 by 1 Th period, a delay circuit 836c that delays the output data of the 1 Hv delay circuit 835b by 1 Th period, and a delay circuit 836f that further delays the output data of the delay circuit 836c by 1 Th period. These 1 Th delay circuits 36a to 836f generate the search-window pixel data necessary for interpolation.

[0256] The half-pixel-precision computation unit 726 (or 738) further includes an adding shift circuit 830a that adds the input search-window pixel data P and the output data of the 1 Th delay circuit 836a and multiplies this by a coefficient (1/2). The adding shift circuit 830a realizes multiplying the coefficient (1/2) by a shift operation of the pixel-data bits in a one-bit lower direction. This half-pixel-precision computation unit 726 (or 738) further includes an adding shift circuit 830b that executes an adding shift operation on the output data of the 1 Th delay circuit 836a and the output data of the 1 Th delay circuit 836e, an adding shift circuit 836c that executes an adding shift operation on the output data of the 1 Th delay circuit 836b and the output data of the 1 Th delay circuit 836e, an adding shift circuit 830d that executes an adding shift operation on the output data of the 1 Th delay circuit 835a and the output data of the 1 Th delay circuit 836b, an adding shift circuit 830e that executes an adding shift operation on the output data of the 1 Th delay circuit 835b and the output data of the 1 Th delay circuit 836c, and an adding shift circuit 830f that executes an adding shift operation on the output data of the 1 Th delay circuit 835c and the output data of the 1 Th delay circuit 836f. These adding shift circuits 830a to 830f generate the data for generating interpolation data between four pixels.

[0257] The half-pixel-precision computation unit 726 (or 738) further includes an adding shift circuit 830g that executes the adding shift operation on the output data of the adding shift circuit 830a and the output data of the adding shift circuit 830d, an adding shift circuit 830h that executes the adding shift operation on the output data of the 1 Th delay circuit 836a and the output data of the 1 Th delay circuit 836b, an adding shift circuit 830i that executes the adding shift operation on the output data of the adding shift circuit 830b and the output data of the adding shift circuit 830c, an adding shift circuit 830j that executes the adding shift operation on the output data of the 1 Th delay circuit 836b and the output data of the 1 Th delay circuit 835a, an adding shift circuit 830k that executes the adding shift operation on the output data of the 1 Th delay circuit 836e and the output data of the 1 Th delay circuit 836b, an adding shift circuit 830l that executes the adding shift operation on the output data of the adding shift circuit 830d and the output data of the adding shift circuit 830e, an adding shift circuit 830m that executes the adding shift operation on the output data of the 1 Th delay circuit 836b and the output data of the 1 Th delay circuit 836c,

and an adding shift circuit 830n that executes the adding shift operation on the output data of the adding shift circuit 830c and the output data of the adding shift circuit 830f. These adding shift circuits 830g to 830n generate the pixel data in the positions of the interpolation pixel data Q9 to Q6 and Q4 to Q1 illustrated in FIG. 85.

[0258] The difference absolute-value sum circuit 804 includes difference absolute value-sum circuits 804a to 804h that receive the outputs Q9 to Q6 and Q4 to Q1 of the adding shift circuits 830g to 830l and the template-block pixel data A and seek a difference absolute-value sum of the imparted signals. The search-window block pixel data P and the template-block pixel data A are in a relationship of being in a state directly behind each other. The difference absolute-value sum circuits 804a to 804h each correspond to a displacement vector (at fractional precision) and are each labeled with a unique code. The evaluation-function value evm corresponding to the motion vector again detected at integer precision is imparted to the comparison unit 806. The comparison unit 806 receives the outputs of the difference absolute-value sum circuits 804a to 804h and the evaluation-function value evm, detects the difference absolute-value sum imparting the smallest value, and determines the displacement vector corresponding to this smallest difference absolute-value sum to be the motion vector at fractional precision.

[0259] In the configuration illustrated in FIG. 86, a configuration may be used where only one difference absolute-value sum circuit is provided or four difference absolute-value sum circuits are provided, this or these are activated in a time-divided manner, and the output data of the adding shift circuits 830g to 830n are sequentially added and totaled. Moreover, in the above configuration, the motion vector is detected at half-pixel precision. A configuration may also be used of detecting motion vectors at a finer fractional precision such as quarter-pixel precision. Moreover, although there are nine evaluation points, a configuration may be used that uses more evaluation points. FIG. 87 is a diagram illustrating an operation sequence of the motion-vector detection device in the ninth example of this invention, and time is indicated on the horizontal axis. Processing for an Nth block (template block) is performed in the integer-precision computation unit 1 and the comparison unit 3, and a motion-vector detection operation is performed. According to the motion vector detected at integer precision of this template block N, motion-vector detection is performed at half-pixel precision. A motion-vector detection operation is performed at fractional precision of a frame for the template block N, and in parallel therewith, motion-vector detection is performed at fractional precision for the odd field of the template block N (sub-template block). When motion-vector detection at fractional precision of the odd field is completed, next, the motion vector of the even field of this block N is detected at fractional precision.

[0260] Meanwhile, in the integer-precision computation unit 1, a motion-vector detection operation is executed at integer precision for the next template block N+1. When the motion-vector detection operation for this block N+1 is completed, the motion-vector detection operation is executed again at fractional precision for the frame and the odd field for the block N+1. When the motion-vector detection at fractional precision of the odd field is completed, a motion-vector detection operation is executed at fractional precision for the even field. By pipelining the operations of the integer-precision computation unit 1 and the operations of the half-pixel-precision computation units 726 and 738 as above, the operations of the integer-precision computation unit 1 and the computation operations of the half-pixel-precision computation units 726 and 738 can be executed by being

temporally separated from each other, enabling motion-vector detection to be executed at fractional precision with a margin in a timing requirement for the computation operations. Moreover, in this situation as well, the integer-precision computation unit 1 and the half-pixel-precision computation units 726 and 738 operate in parallel with each other, enabling fractional-precision motion-vector detection to be performed at high speed.

[0261] Note that during the motion-vector detection operation at integer precision in the integer-precision computation unit 1, the search-window pixel data is output therefrom. In this situation, it is supposed that the search-window pixel data of the prediction-image memory 722 is rewritten by the search-window pixel data imparted from the integer-precision computation unit 1 when the half-pixel-precision computation unit 738 is performing the evaluation-function-value calculation operation. In this situation, providing a buffer memory between the integer-precision computation unit 1 and the prediction-image memory 722 can prevent the stored pixel data of this prediction-image memory from being rewritten by the search-window pixel data imparted from the integer-precision computation unit 1 when the stored data of the prediction-image memory 722 is being used in the half-pixel-precision computation unit 738 and enables a fractional-precision computation operation to be reliably performed. Moreover, providing such a buffer can mitigate timing conditions between computation at integer precision and computation at fractional precision, enabling computation at integer precision and computation at fractional precision to be executed in a parallel aspect with a margin.

[0262] Note that in the above example, the block prediction mode, the even-field prediction mode, and the odd-field prediction mode are supposed as the prediction modes. However, in a situation where coding of the pixels is performed in field units, the motion vector can be detected according to the block prediction mode, an upper-half prediction mode, and a lower-half prediction mode, and the configuration of this ninth example can be used as-is with such prediction modes as well. Only a period of writing data to the prediction-image memory of the pixel data for the upper-half-block prediction mode and the lower-half-block prediction mode is changed. As above, according to the configuration of the ninth example of this invention, because the configuration is such that motion-vector detection is performed at fractional precision using the template-block pixel data and the search-window pixel data imparted from the integer-precision computation unit, a motion vector can be performed at fractional precision for each of the plurality of prediction modes, a more optimal motion vector can be selected, and high-precision motion-vector detection can be performed.

[0263] [Example 10] FIG. 88 is a diagram illustrating positions of template blocks when a current screen is divided into twenty-five 5×5 macro blocks (template blocks) MB (0,0) to MB (4,4). The positions of the template blocks are indicated under the reference sign MB as vectors along positive directions in horizontal and vertical directions with the upper-left template block as the origin. The template blocks MB (0,0) to MB (4,4) are each provided with a configuration of sixteen-pixel rows by sixteen-pixel columns. During a motion-vector detection operation, motion-vector detection is performed for all template blocks of a current-screen image. A motion-vector search range is set to be the same for each template block. In this situation, as illustrated in FIG. 89 for example, with the template block MB (0,0), no search-window pixels are present in a region 10 enclosed in the dashed line. Therefore, a motion-vector detection operation of the pixels included in this region 810

needs to be prohibited. A method and configuration of limiting the motion-vector search range that are a tenth example of this invention are described below.
[0264] FIG. 90 is a diagram illustrating a configuration of a search range used as the processing unit. The unit search range is set to sixteen displacements in the horizontal direction and thirty-three displacements in the vertical direction. Cycles necessary to calculate motion-vector evaluation-function values in this unit search range are 1 processing time T (33·16 = 768 cycles). In this unit search range, when a total prohibition signal Ia1 is imparted, evaluation of displacement vectors in this search range is prohibited (in both the horizontal direction and the vertical direction). As illustrated in (A) in FIG. 91, when an upper-half prohibition signal Iuph is placed in an active state, displacement-vector evaluation is prohibited in a region in an upper half of a unit search range 812—that is, a region where vectors in the vertical direction (V vectors) have a negative value (-1 to -16). Moreover, when a lower-half prohibition signal Ilwh is placed in an active state, evaluation is prohibited in a region 812l in a lower half in the unit search range 812—that is, a region where vertical-direction displacement is in a positive region of 1 to +16.
[0265] As illustrated in (B) in FIG. 91, when evaluating at half-pixel precision, when the upper-half prohibition signal Iuph is imparted, displacement vectors having the vector -1/2 in the vertical direction are excluded from the search range. Likewise, when the lower-half prohibition signal Inwh is placed in the active state, at the time of half-pixel-precision motion-vector evaluation, a region where the vertical-direction displacement vector is +1/2 is excluded from evaluation. As illustrated in FIG. 92, when a left-half prohibition signal Ilfh is imparted, at a time of motion-vector evaluation at integer precision, all displacement vectors in this unit search range are evaluated. However, when calculating motion vectors at half-pixel precision, in a situation where a motion vector at integer precision is present in a leftmost region 812l of this unit search range 812, evaluation of displacement vectors having a vector component (H vector) of -1/2 in the horizontal direction is excluded.
[0266] This is because pixels for generating a vertical component of -1/2 are present outside this search-range region 812 and desired interpolation pixel data cannot be generated. Likewise, as illustrated in FIG. 93, when a right-half prohibition signal Irth is placed in an active state, motion-vector detection is performed with only vectors of a region 812r on a left end in the horizontal direction in the search range 812 as the evaluation target. In a situation where a vector of the region 812r is selected as an optimal vector—that is, the motion vector—in a fractional-precision (half-pixel-precision) computation unit, fractional-precision vectors (half-pixel-precision vectors) in the horizontal positive direction are excluded from evaluation. By using the five types of prohibition signals Ial, Iuph, Ilwh, Ilfh, and Irth, a motion-vector search range for one template block can be set to a region wherein only accurate pixel data is present. Next, specific operations of limiting the motion-vector search range are described.
[0267] First, a range of sixteen displacements in the horizontal direction and thirty-three displacements in the vertical direction is made to be the unit search range. A configuration of an integer-precision computation unit is made similar to the situation of the fourth example. That is, the motion-vector search operation is performed at integer precision using element processors disposed in sixteen rows by sixteen columns and data-buffer columns that each store data of thirty-two pixels. In this situation, the processing-unit time T is 16×(16+32) = 768 cycles. Now, as above, the horizontal-direction search range is made

to be -16 to +15.5 and the vertical-direction search range is made to be -16 to +16. In this situation, the integer-precision displacement vectors include -16 to +15 as the horizontal-direction component H, and two processing unit times, 2T, are necessary to evaluate these integer-precision displacement vectors. Moreover, at this time, the fractional-precision computation unit performs the motion-vector detection operation at half-pixel precision.
[0268] Now, a situation is supposed of performing motion-vector detection of the template block MB (0,0) in the upper-left end of the current screen illustrated in FIG. 88. In this situation, no pixels are present on left-side and upper-side regions of the template block MB (0,0). Therefore, motion-vector evaluation for these regions is excluded as below. As illustrated in FIG. 94, in a first processing unit time T, a motion-vector search is performed in a range of -16 to -1 in the horizontal direction. In this processing time, as illustrated in (A) in FIG. 95, because no pixel data is present in a region 815, the motion-vector search operation in this region 815 is prohibited. That is, a total prohibition signal is generated and displacement-vector evaluation in the region 815 is prohibited. In the next processing unit time T, the displacement vectors in a region of 0 to +15 displacements in the horizontal direction are evaluated. In this situation, as illustrated in (B) in FIG. 95, in a region 816, no pixels are present in a region 817a where displacements in the vertical direction are -1 to -16. Therefore, in this situation, first, displacement vectors in the region 817a are excluded from evaluation. Moreover, in a situation of performing motion-vector detection at half-pixel precision, there is a need to only use pixels in a region 817b. Therefore, when creating a prediction image at half-pixel precision, because no pixel data is present for displacement vectors with a horizontal-direction displacement of -1/2, displacement-vector evaluation at fractional precision for this region is prohibited. Therefore, at this time, the left prohibition signal is placed in the active state. This enables motion-vector detection to be performed using only the pixel data present in the current screen in the full search range.
[0269] FIG. 96 is a flowchart illustrating operations at the time of the motion-vector search at fractional precision (half-pixel precision) for the template block MB (0,0). First, a motion vector at integer precision (H,V) is detected (step S1). Next, at steps S2 and S3, it is determined whether the horizontal component H and V of this motion vector (H,V) are 0. When it is determined at step S2 that the horizontal component H of the integer-precision motion vector is 0, because the left prohibition signal is imparted, at step S4, displacement vectors with a negative component in the horizontal direction (horizontal direction -1/2) are excluded from evaluation when predicting at fractional precision. Meanwhile, when it is determined at step S3 that the vertical component V is 0, because a fractional-precision prediction image is generated centered around the macro block in an uppermost end portion of the current screen, displacement vectors of a vertical-direction negative component (-1/2) including pixels not included in the current screen at this time are excluded from evaluation (step S5). In a situation where neither the horizontal component H nor the vertical component V is 0, all displacement vectors of fractional precision centered around an integer-precision motion vector are evaluated (step S6).
[0270] For the template blocks MB (1,0), MB (2,0), and MB (3,0) that are blocks in the upper end of the current screen, evaluation of displacement vectors whose vertical component V is negative is prohibited. Therefore, as illustrated in FIG. 97, in a full-evaluation period, only the upper prohibition signal is placed in the active state. For the template block MB (4,0) in the upper-right end of the current screen, as illustrated in (A) in

FIG. 98, at the first processing unit time, the upper prohibition signal is placed in the active state. In this state, as illustrated in (B) in FIG. 98, in the first search region 815, the motion-vector search operation is prohibited in a region 817c in an upper half (region where the vertical-direction displacement is negative). At the next processing unit time, both the upper prohibition signal and the right prohibition signal are placed in the active state. In this state, as illustrated in (C) in FIG. 98, in the next search range 816, displacement-vector evaluation in the upper-half region 817a is prohibited. In the lower-side region (region whose vertical component V is no less than 0) 817b, only displacement vectors present in a region on a left end of this region 817b are evaluated. That is, only evaluation operations for displacement vectors (H,V) where H = 0 and V ≥ 0 are performed. This determines the motion vector using only the screen -image pixel data at integer precision for the template block MB (4,0). Next, according to this determined motion vector of integer precision, a motion vector at fractional precision is determined. The motion-vector detection operation at this fractional precision includes the processing steps illustrated in FIG. 99.

[0271] As illustrated in steps S20 to S22 in FIG. 99, at the first processing unit time, a motion-vector search is performed by excluding displacement vectors in the vertical direction. At the next processing unit time, displacement vectors whose horizontal component H is 0 and whose vertical component V is no less than 0 are evaluated. By these steps S20 and S22, the motion vector (H,V) at integer precision is determined (S22). Next, motion-vector detection at fractional precision is performed. At steps S23 and S24, it is determined whether the horizontal component H and the vertical component V of the motion vector (H,V) of this integer precision are equal to 0. In a situation where at step S23 it is determined that the horizontal component H is 0, the search-window block positioned on the right end in the current screen needs to be used as the prediction image. In this situation, displacement vectors with a horizontal component of a positive fractional precision—that is, a horizontal component of +1/2—need to be excluded in performing a motion-vector search at fractional precision (step S25). Meanwhile, in a situation where the vertical component V is determined to be 0, because pixels corresponding to a motion vector where V = -1 are not included in the current screen, displacement vectors with a negative vertical component at fraction precision (-1/2) are excluded in performing a motion-vector search at fractional precision. In a situation where both the horizontal component H and the vertical component V are 0, the operations of steps S25 and S26 are both performed. In a situation where both the horizontal component H and the vertical component V of the integer-precision motion vector (H,V) are not 0, a motion-vector search is performed in the full search range at fractional precision around the motion vector (H,V) of integer precision (step S27).

[0272] For the template blocks MB (0,1), MB (0,2), and MB (0,3) on the left end of the current screen, at the first processing unit time, the total prohibition signal is placed in the active state and displacement vectors in a region of horizontal-direction displacements of -16 to -1 are excluded from evaluation. At the next processing unit time, the left prohibition signal is placed in the active state. At a time of a motion-vector search at integer precision, all displacement vectors in horizontal-direction displacements 0 to +15 are evaluated. During operations at a time of motion-vector detection at fractional precision, steps S2, S4, and S6 illustrated in FIG. 96 are performed. For template blocks MB (i,j [i ≠ 0, j ≠ 0]) in a central portion of the current screen, as illustrated in FIG. 101, during two processing unit times, all displacement vectors in the search range are evaluated.

[0273] For the template blocks MB (4,1), MB (4,2), and MB (4,3) on the right end of the current screen, as illustrated in FIG. 102, at the first processing unit time T, evaluation-function values are generated for all displacement vectors of the search range. At the next processing unit time, only the right prohibition signal is placed in the active state. That is, a motion-vector search is performed with only displacement vectors of a horizontal-direction displacement of 0 as the evaluation target. In a motion-vector search operation at fractional precision for these template blocks of the right end, the processing of steps S23, S25, and S27 illustrated in FIG. 99 is executed. For the template block MB (0,4) of the lower-left end of the current screen, as illustrated in FIG. 103, at the first processing unit time, the total prohibition signal is placed in the active state. In this state, as illustrated in (B) in FIG. 103, all displacement vectors in the first search region 815 are excluded from evaluation.

[0274] At the next processing unit time, both the lower prohibition signal and the left prohibition signal are placed in the active state. In this state, as illustrated in (C) in FIG. 103, the displacement vectors in a region 817d in a lower half in the search region 816 (B = 1 to 16) are excluded from evaluation. That is, the motion-vector search of the template block MB (0,4) is performed using only the displacement vectors in a region 817e. FIG. 104 is a flowchart illustrating motion-vector search operations at fractional precision of this template block MB (0,4). That is, as illustrated in step S30 of FIG. 104, a motion-vector search at integer precision is performed in a region of a range of a horizontal component of no less than 0 and a vertical component of no greater than 0 and a motion vector at integer precision is determined according to search results thereof (step S31). According to the motion vector (H,V) determined at this integer precision, it is determined whether the horizontal component H and the vertical component V are 0 (steps S32 and S33). When it is determined at step S32 that the horizontal component H is 0, all negative vectors at fractional precision in the horizontal direction are excluded in evaluating the displacement vectors (step S34). Meanwhile, when it is determined at step S33 that the vertical component V is 0, the displacement vectors that are positive in the vertical direction at fractional precision are excluded in evaluating the displacement vectors at fractional precision (step S35). In a situation where it is determined that neither the horizontal component H nor the vertical component V of the motion vector (H,V) of integer precision is 0, all displacement vectors in the search range at this fractional precision are evaluated (step S36).

[0275] For the template blocks MB (1,4), MB (2,4), and MB (3,4) of a current-screen lower end, as illustrated in FIG. 105, the lower prohibition signal is placed in the active state over an entirety of the processing time. At a time of an integer-precision zero motion-vector search when this lower prohibition signal is active, a motion-vector search is performed in a range where the vertical component V is no greater than 0. At the time of a motion-vector search at fractional precision, the processing operations illustrated in steps S33, S35, and S36 illustrated in FIG. 104 are executed. For the template block MB (4,4) in the lower-right end of the current screen, as illustrated in (A) in FIG. 106, at the first processing unit time T, the lower prohibition signal is placed in the active state. In this state, as illustrated in (B) in FIG. 106, the displacement vectors in a region 817f in a lower-half region (B = 1 to 16) in a minimum search region 815 are excluded from evaluation. At the next processing unit time, as illustrated in (A) in FIG. 106, both the lower prohibition signal and the right prohibition signal are placed in the active state. In this state, as illustrated in (C) in FIG. 106, only

displacement vectors included in a region 817g where the horizontal component H is 0 in the search region 816 of the latter half is evaluated. The operations at the time of motion-vector detection at fractional precision when the lower prohibition signal is active and the right prohibition signal is active are the same as the processing operations at the previous NO teps S33, S35, and S36 and the processing operations of steps S23, S25, and S27.

[0276] By selectively placing prohibition signals in the active state according to the position of the template block to be processed as above, motion vectors can be determined using only pixel data included in the current screen image, motion vectors with high precision can be detected, and an accurate prediction image can be generated. Note that although not illustrated clearly in the above description, at a time of motion-vector detection at integer precision, motion vectors are determined according to a plurality of prediction modes. Therefore, at a time of motion-vector detection at fractional precision as well, a motion-vector search range at fractional precision is determined for each of a plurality of prediction vectors according to a combination of the values of the horizontal component and the vertical component of the motion vectors determined according to each of the plurality of prediction modes and the prohibition signals placed in the active state at that time.

[0277] FIG. 107 is a diagram schematically illustrating an overall configuration of a motion-vector detection device according to the tenth example of this invention. In FIG. 107, the motion-vector detection device includes a reference-image memory 820 that stores reference-image pixel data; a current-image memory 821 that stores current-screen-image pixel data; an integer-precision motion-vector detection unit 830 that receives search-window pixel data PY from the reference-image memory 820 and template-block pixel data PX from the current-image memory 821 to detect a motion vector at integer precision; and a fractional-precision motion-vector detection unit 832 that receives the search-window pixel data PY, the template pixel data PX, integer-precision motion-vector data MVI, and an evaluation-function value evm of this integer-precision motion-vector MVI imparted from the integer-precision motion-vector detection unit 830 to detect a motion vector at fractional precision. To simplify the drawing, FIG. 107 representatively illustrates only motion vectors determined according to one prediction mode. Motion vectors at integer precision and fractional precision are detected according to the plurality of prediction modes. Fractional-precision vectors from the fractional-precision motion-vector detection unit 832 are representatively illustrated by a vector MVF.

[0278] The motion-vector detection device further includes a read control circuit 838 that generates read addresses of the reference-image memory 820 and the current-image memory 821 and controls pixel-data read operations in these memories 820 and 821, a search-range setting unit 834 that receives the read address for the current-image memory 821 included in this read control circuit 838—that is, template-block pixel position information—to set a motion-vector search range, and a search-operation control unit 836 that controls motion-vector search operations in the integer-precision motion-vector detection unit 830 and the fractional-precision motion-vector detection unit 832 according to the motion-vector search-range information from the search-range setting unit 834—that is, the displacement-vector evaluation operations. This search-operation control unit 836 controls the displacement-vector evaluation operations in the integer-precision motion-vector detection unit 830 according to the search-range information from the search-range setting unit 834. This search-operation control unit 836 also controls

displacement-vector evaluation operations in the fractional-precision motion-vector detection unit 832 according to the integer-precision motion-vector information MVI from the integer-precision motion-vector detection unit 830 and the search-range setting information from the search-range setting unit 834—that is, the prohibition signals.
[0279] FIG. 108 is a diagram illustrating a configuration of an evaluation-operation control unit of an integer-precision motion-vector selection unit included in the search-operation control unit 836 illustrated in FIG. 107. FIG. 108 also illustrates a configuration of a comparison unit 3 included in the integer-precision motion-vector detection unit 830. The comparison unit 3 included in the integer-precision motion-vector detection unit 830 includes a register 680 that stores an evaluation-function value, a comparator 682 that compares an evaluation-function value evmi imparted from the computation unit and the evaluation-function value stored by the register 680, a counter 684 that generates displacement-vector information associated with the evaluation-function value imparted from the computation unit, and a register 686 that stores displacement-vector information imparting the smallest evaluation-function value. Configurations of the register 680, the comparator 682, the counter 684, and the register 686 are substantially similar to the configuration of the comparison unit illustrated in FIG. 65. However, to simplify the drawing, FIG. 108 does not illustrate the priority-order determination circuit 690 illustrated in FIG. 65. When imparted with an update instruction signal from the comparator 682, the update control circuit 839 updates the contents of the registers 680 and 686. This update control circuit 839 also prohibits update operations of the registers 680 and 686 regardless of the output signal of the comparator 682 when a prohibition signal Ia from the integer-precision search-operation control unit 840 is active.
[0280] The integer-precision search-operation control unit 840 includes an HV extraction circuit 841 that receives displacement-vector information depv from the counter 684 and extracts a horizontal component H and a vertical component V, an upper-half detector 842 that is placed in an actuating state when the upper-half prohibition signal Iuph is active and outputs a signal in an active state when the vertical component V imparted from the HV extraction circuit 841 is negative, a lower-half detector 843 that is placed in an actuating state when the lower-half prohibition signal Ilwh is active and outputs a signal in an active state when the vertical component V from the HV extraction circuit 841 is in a positive region, a latch circuit 844 that latches horizontal-component H data that is first output from the HV extraction circuit 841 when the right-half prohibition signal Irht is active, and a left-end detector 845 that is placed in an actuating state when the right-half prohibition signal Irht is active and outputs a signal in an active state when the horizontal component H output by the HV extraction circuit 841 and the horizontal component latched by the latch circuit 844 match. The total prohibition signal Ial, the output of the upper-half detector 842, the output signal of the lower-half detector 843, and the output signal of the left-end detector 845 are, for example, placed in a wire OR connection to generate a prohibition signal Ih that is imparted to the update control circuit 839. Instead of the wired OR connection, a configuration may be used that outputs the prohibition signal Ih by an OR gate that receives the signal Ial, the output signal of the right-half detector 842, the output signal of the lower-half detector 843, and the output signal of the left-end detector 845.
[0281] When the total prohibition signal Ial is in the active state, the update control circuit 839 places the update instruction signal in a continually inactive state and prohibits content updating of the registers 680 and 686. The upper-half detector 842 outputs the

signal in the active state when the displacement-vector information depv output from the counter 684 indicates a displacement vector with a negative vertical direction and the upper-half prohibition signal Iuph is in the active state. Therefore, when the upper-half prohibition signal Iuph is in the active state, the evaluation-function values of displacement vectors having a negative vertical component are not stored in the register 680 regardless of the output signal of this comparator 682 and the displacement vector of the register 866 is not updated. When the lower-half prohibition signal Ilwh is in the active state, the lower-half detector 843 outputs the signal in the active state when the displacement-vector information depv output from the counter 684 has a positive vertical component. This places the prohibition signal Ih in the active state and prohibits updating of the registers 680 and 686. When the right-half prohibition signal Irht is in the active state, the latch circuit 844 latches the horizontal component of the displacement-vector information output by the counter 684 generated at the beginning of the processing unit time thereof. This latches a region on a left end in a negative vector search range at the processing unit time where this right-half prohibition signal Irht is in the active state—that is, the first horizontal-component information in the horizontal direction at this processing unit time. When active, the left-end detector 845 outputs the signal in the active state when the horizontal component of the displacement-vector information depv output by the counter 684 differs from the horizontal component stored in the latch circuit 844. By this, the evaluation operation is performed only for displacement vectors positioned in a column on the left end in the search range at the processing unit time.

[0282] The above configuration enables the motion-vector search range to be determined according to the prohibition signals at a time of motion-vector detection at integer precision. The left-half prohibition signal Ilft is not imparted to this integer-precision search-operation control unit 840 because evaluation operations of the displacement vectors are performed over the entire range in the horizontal direction when this signal is imparted. Moreover, in the configuration illustrated in FIG. 108, a configuration may be used where the prohibition signal Ih is imparted to the comparator 680 to prohibit the comparison operation of the comparator. The comparator 682 whose comparison operation is prohibited maintains the update instruction signal in the inactive state. FIG. 109 is a diagram schematically illustrating a configuration of the fractional-precision search operation control unit included in the search-operation control unit 836 illustrated in FIG. 107. In FIG. 109, the fractional-precision search-operation control unit 850 includes an HV extractor 851 that extracts a horizontal component H and a vertical component V from motion-vector information MVI of integer precision output from the comparison unit 3 illustrated in FIG. 108, a V0 detector 852 that outputs a signal in an active state when the vertical component V from the HV extractor 851 is 0, a gate circuit 853 that places a vertical negative-component prohibition signal IVNF that prohibits evaluation of displacement vectors of a vertical-direction negative component at fraction precision in an active state when both the upper-half prohibition signal Iuph and the output signal of the V0 detector 852 are in an active state, a gate circuit 854 that places a vertical positive-component prohibition signal IVPF that prohibits evaluation of displacement vectors that are positive in the vertical direction at fractional precision in an active state when both the lower-half prohibition signal Ilwh and the output signal of the V0 detector 852 are in an active state, a latch circuit 855 that latches a horizontal component depv (H) of the displacement vector imparted from the counter 684 (see FIG. 108) in the beginning of the

processing unit time thereof when the left-half prohibition signal Ilft is active, a coincidence circuit 856 that is activated when the left-half prohibition signal Ilft is activated and places a horizontal negative-component prohibition signal IHNF that prohibits evaluation of displacement vectors that are negative in the horizontal direction at fractional precision in an active state when the latch horizontal component of the latch circuit 855 and the horizontal component H of the motion vector output from the HV extractor 851 are equal, a latch circuit 857 that latches the horizontal component depv (H) of the displacement vector imparted at the beginning of this processing unit time when the right-half prohibition signal Irht is activated, and a coincidence circuit 858 that is activated when the right-half prohibition signal Irht is activated and places the horizontal positive-component prohibition signal IHPF that prohibits evaluation of positive displacement vectors at fractional precision in an active state when the horizontal-component data latched by the latch circuit 857 and the horizontal component of the integer-precision motion vector MVI from the HV extractor 851 coincide.

[0283] The gate circuits 853 and 854 can each be configured by, for example, an AND gate that generates an output signal in an active state when a signal imparted to both inputs is in an active state (when the active state is high-level). The coincidence circuits 856 and 858 can be configured using, for example, an EXNOR gate and an AND gate (an AND gate is used to obtain a logical product of the output signals of each EXNOR gate because the horizontal-component data is a multi-bit signal. Various configurations are possible for the HV extractor 851 according to the configuration of this motion-vector information MVI. In a situation where the integer-precision motion-vector information MVI is a normal binary count value, it is sufficient to use a circuit that decodes this count value. Moreover, in a situation where this integer-precision motion-vector information MVI is provided with a horizontal component and a vertical component, it is sufficient to use the horizontal component and the vertical component as-is.

[0284] Furthermore, the total prohibition signal Ial is used as a signal IALF that prohibits evaluation operations for all displacement vectors of the fractional-precision motion-vector detection unit. The prohibition signals IALF, IVNF, and the like illustrated in this FIG. 109 are imparted to the comparison unit included in the fractional-precision motion-vector detection unit 832 illustrated in FIG. 107. An application aspect of the prohibition signals receives various modifications according to the configuration of this comparison unit. In a situation where displacement-vector evaluation at half-pixel precision is sequentially executed for each displacement vector, the motion-vector detection range at fractional precision can be controlled by a configuration similar to the configuration illustrated in FIG. 108. FIG. 110 is a diagram illustrating one example of a configuration of a principal portion of the fractional-precision motion-vector detection unit 832. The fractional-precision motion-vector detection device illustrated in this FIG. 110 corresponds to the configuration of the water-division-precision (about half-pixel) motion-vector detection device illustrated in FIG. 86. The water-division-prevision motion-vector detection device illustrated in this FIG. 110 includes difference absolute-value sum circuits 804a to 804g that generate evaluation-function values of eight proximal evaluation points and a comparison unit 806 that detects a smallest evaluation-function value from these difference absolute-value sum circuits 804a to 804g and the evaluation-function value evm of the integer-precision motion vector and outputs information indicating the displacement vector corresponding to this smallest evaluation-function value as a fractional-precision

motion vector MVF. Each difference absolute-value sum circuit 804a to 804g is uniquely associated with eight proximal displacement vectors (evaluation points) of fractional precision. The difference-absolute-value sum circuit 804a corresponds to the displacement vector (-1/2,-1/2), the difference-absolute-value sum circuit 804b corresponds to the displacement vector (0,-1/2), and the difference-absolute-value sum circuit 804c corresponds to the displacement vector (1/2,-1/2).

[0285] The difference-absolute-value sum circuit 804d corresponds to the displacement vector (-1/2,0), the difference-absolute-value sum circuit 804e corresponds to the displacement vector (1/2,0), the difference-absolute-value sum circuit 804f corresponds to the displacement vector (-1/2,1/2), and the difference-absolute-value sum circuit 804g corresponds to the displacement vector (0,1/2). The evaluation-function value evm corresponds to the displacement ver (0,0). The difference-absolute-value sum circuit 804a is imparted with the prohibition signals IHNF, IVNF, and IALF. The difference-absolute-value sum circuit 804b is imparted with the prohibition signals IVNF and IALF. The difference-absolute-value sum circuit 804c is imparted with the prohibition signals IHPF, IVNF, and IALF. The difference-absolute-value sum circuit 804d is imparted with the prohibition signals IHNF and IALF. The difference-absolute-value sum circuit 804e is imparted with the prohibition signals IHPF and IALF. The difference-absolute-value sum circuit 804f is imparted with the prohibition signals IHNF, IVPF, and IALF. The difference-absolute-value sum circuit 804g is imparted with the prohibition signals IHNF and IALF. These difference-absolute-value sum circuits 804a to 804g set an output value thereof to a maximum value when an imparted prohibition signal is placed in the active state. That is, when, for example, a bit count of the output data of each difference-absolute-value sum circuit 804a to 804g is N bits, the difference-absolute-value sum circuits 804a to 804g set all outputs of N bits to “1” when at least one corresponding prohibition signal is placed in the active state. By this, displacement vectors included in a region where the prohibition signal is placed in the active state in the comparison unit 806 are placed in a non-selected state and excluded from motion-vector evaluation. For the configuration of setting the output data of the difference-absolute-value sum circuits 804 (804a to 804g) to the maximum value when a prohibition signal is placed in the active state, it is sufficient to, for example, provide a gate that performs an OR computation of the prohibition signal and the output bits of the difference-absolute-value sum circuit 804 to each output bit. However, when in the active state, the prohibition signal is a high-level signal, and the bit “1” is made to correspond to a high-level signal.

[0286] In the above configuration, a displacement-vector evaluation target region at fractional precision can be easily set. The configurations illustrated in FIG. 108 to FIG. 110 are disposed corresponding to each of the plurality of prediction modes. FIG. 111 is a diagram schematically illustrating configurations of the search-range setting circuit 834 and the read control circuit 838 illustrated in FIG. 107. In FIG. 111, the read control circuit 838 includes a counter 862 that outputs information indicating a position of a template block in the current screen; a processing-unit setting circuit 863 that sets the processing unit time as the units of a motion-vector search time; a read-address generator 860 that generates the read address of the current-image memory 821; and a read controller 861 that controls updating of the counter 862, address updating of the read-address generator 860, and data read operations of the current-image memory 821 according to the processing-time information set by the processing-unit setting circuit 863.

[0287] The read-address generator 860 converts the count value of the counter 862 into top-address information and sequentially generates the read address of the current-image memory 821 from this top address. Normally, the current-image memory 821 includes a memory cell arranged in a matrix of rows and columns. In a situation of a configuration where the current-image memory 821 stores the pixel data so each row corresponds to pixels of one row in the horizontal direction on the screen, the read-address generator 860, under control of the read controller 861, reads data of sixteen pixels in one row and afterward reads the pixel data disposed in the same column of the next row. That is, after reading data of sixteen pixels from the row designated by the top address generated according to the template-block information of the counter 862, in the next row, data of sixteen pixels is read starting from the same column as the top address. Furthermore, the read-address generator 860 is provided with this address conversion function. It is sufficient for the configuration of this address generator to use a configuration of, for example, reading data of sixteen pixels in a top row in a top address (including row and column addresses) and afterward again reading data of sixteen pixels from a column designated by the top address. The read controller 861 updates the count value of the counter 862 according to the processing-time information set by the processing-unit setting circuit 863 and sets the current-image memory 821 to a data read state at a time of the motion-vector detection operations for a new template block. Although not illustrated, the processing-time information is imparted from the outside or, according to a usage, the stored processing-time information of this processing-unit setting circuit 863 is fixedly set.
[0288] The search-range setting circuit 834 includes a template-block position detector 871 that detects a position of a template block that receives the count value from the counter 862 and is processed, a cycle detection circuit 872 that detects which cycle the cycle being executed is with the processing unit time as the units according to the processing time set by the processing-unit setting circuit 863 and a search-window-pixel-data transfer clock signal φ, and a prohibition-signal generator 873 that generates the prohibition signals such as the total prohibitions signal Ial and the upper-half prohibition signal Iuph according to the template-block position information detected by the template-block position detector 871 and the cycle information detected by the cycle detection circuit 872. This prohibition-signal generator 873 places each prohibition signal in a generating state according to the sequence illustrated above in FIG. 92 to FIG. 106. For example, when the template-block position detector 871 detects the template block MB (0,0) in the upper-left end of the screen, the prohibition-signal generator 873 places the total prohibition signal Ial in the active state at the first cycle and places the upper-half prohibition signal Iuph and the left-half prohibition signal Ilwh in the active state at the next processing cycle.
[0289] The search-range setting circuit 834 further includes a template-block designation circuit 874 that generates information designating the template block to receive a restriction of the motion-vector search range, a coincidence detector 876 that determines whether the template block designated by the template-block designation circuit 874 and the template-block position information designated by the template-block position detector 871 coincide, a prohibited-area setting circuit 875 that sets a region where the displacement-vector evaluation operations are to be prohibited, a prohibition-signal generator 877 that generates a prohibition signal according to the coincidence detection signal from the coincidence detector 876 and the prohibited-area information from the

prohibited-area setting circuit 875, and a selector 878 that selects one among the outputs of the prohibition-signal generators 873 and 873 according to a mode designation signal φSML. A prohibition signal is imparted from this selector 878 to the search-operation control unit 836 (see FIG. 107). The template-block designation circuit 874 and the prohibited-area setting circuit 875 are respectively set with predetermined information from an external controller that is not illustrated. This enables the motion-vector search range to be restricted only for a desired template block. The prohibition-signal generator 877 places the prohibition signal in the active state so as to prohibit the motion-vector search operations (displacement-vector evaluation operations) in the prohibited area set by this prohibited-area setting circuit 875 when the coincidence signal from the coincidence detector 876 is in the active state. As a default mode, the prohibition information from the prohibition-signal generator 873 is selected by the selector 878. In a situation of setting a search range different from the motion-vector search range determined according to the position of the template block on this screen, the selector 878 is set to a state of selecting the prohibition signal output by the prohibition-signal generator 877 by the mode designation signal φSML. Providing the template-block designation circuit 874, the prohibited-area setting circuit 875, the coincidence detector 876, and the prohibition-signal generator 877 enables the motion-vector search range to be set according to the processing mode and enables a motion-vector search range according to the configuration of the image to be processed to be set.

[0290] For example, as illustrated in (A) in FIG. 112, for a template block 881 positioned in a central portion of a screen 880, as illustrated in (B) in FIG. 112, a motion-vector search in the four regions 882a, 882b, 882c, and 882d in the search range 882 can be prohibited. For example, in a situation where the screen is moving in one direction, a probability is high of a motion vector being detected only in an opposite direction of this moving direction. In this situation, by performing the motion-vector search only in this other direction, a motion vector with higher correlation can be detected. Moreover, when, unlike an image where inter-frame prediction is performed such as an I picture, an image where intra-frame prediction is performed such as a P picture is inserted, in a situation of detecting a motion vector by inter-frame prediction, a precision of the motion vector using a screen region formed by an intra-frame prediction mode in this P picture degrades compared to a situation of inter-frame prediction. As such, prohibiting motion-vector detection operations by inter-frame prediction using such a region enables a motion vector with high precision to be detected.

[0291] As above, according to the tenth example of this invention, because the configuration is such that the motion-vector search range can be restricted for a template block, motion-vector detection with higher precision can be performed. In particular, because the configuration is such that the vector search range of even motion-vector detection operations at fractional precision is restricted, motion vectors can be detected using only data of pixels actually present, motion vectors with higher precision can be detected, and a high-precision image coding system can be constructed.

[Configuration of Input Unit] FIG. 113 is a diagram schematically illustrating a configuration of an input unit of the motion-vector detection device. In FIG. 113, a motion-vector detection device 200 detects motion vectors by performing predetermined computation according to template-block data imparted from a current-image memory 204 that stores current-screen-image data and search-area pixel data read from a reference-

image memory 202 that stores reference-screen-image data. From the current-image memory 204, the template-block pixel data is read every one cycle of motion-vector detection and imparted to the computation unit 1. In this situation, because the element processors included in the computation unit 1 are essentially arranged linearly, by reading a predetermined number of pixel data that are in a column adjacent to a template block processed in the previous cycle and are arranged in the same row, pixel data of the next template block can be disposed in the processor array of the computation unit 1.
[0292] The reference-image memory 202 sequentially reads pixel data in a search area and outputs this to an input unit 2. With the data stored in this reference-image memory 202, a storage content thereof is changed by an output of an adding circuit 934 illustrated in FIG. 152.
[0293] The input unit 2 includes a search-window buffer memory 205 for storing the search-area pixel data from the reference-image memory 202 and a screen-format conversion circuit 207 for converting a screen format of the image data stored in this search-window buffer memory 205 into a designated screen format. The search-window buffer memory 205 stores, for example, pixel data of a search window one column adjacent to a search window stored in the processor array of the computation unit 1 in a search-area region. The screen format of the image data read from the reference-image memory 202 is either a frame format or a field format. In this situation, in a situation where a prediction mode of the motion-vector detection device 200 is configured to correspond to the frame format, the screen format of the pixel data to be processed is changed according to this internal prediction mode. An external processing device does not need to convert the screen format by converting pixel-data columns according to the prediction mode of this motion-vector device, which mitigates a load of the external processing device, and a circuit necessary for screen-format conversion does not need to be externally provided, which can reduce an external hardware volume. Moreover, regardless of whether the screen format of the image data to be processed is the field format or the frame format, motion detection can be executed in frame units, which enables a highly versatile motion-vector detection device to be realized.
[0294] FIG. 114 is a diagram schematically illustrating a configuration of the screen-format conversion circuit illustrated in FIG. 113. In FIG. 114, the screen-format conversion circuit 207 includes a frame/frame address generator 213 that generates write and read addresses in a situation where the screen format of the input image signal is the frame format, a field/frame address generator 215 that generates write and read addresses in a situation where the screen format of the input image data is the field format, and a selector 211 that selects one among the outputs of the address generators 213 and 215 according to a screen-format designation signal φPT and imparts this to the search-window buffer memory 205.
[0295] FIG. 115 is a diagram illustrating one example of a configuration of the frame/frame address generator illustrated in FIG. 114. FIG. 115 illustrates as one example a configuration for address generation in a situation where the search-window buffer memory is provided with a configuration of forty-eight rows × fourteen columns. The configuration illustrated in this FIG. 115 is modified as appropriate according to a storage capacity of the search-window buffer memory 205.
[0296] In FIG. 115, the frame/frame address generator 213 includes a write-address generator 220 that generates a write address when writing image data and a read-address

generator 222 for generating a read address for reading the data from this buffer memory 205 and imparting this to the computation unit 1. The write-address generator 220 includes a sixteen-ary counter 220a that counts a write clock signal φCW that imparts a timing of image-data writing generated at a time of writing and generates a count value thereof as a column-address pointer AYPW and a forty-eight-ary counter 220b that counts a count-up signal φUPY of the sixteen-ary counter 220a and generates a count value thereof as a row-address pointer AXPW. The forty-eight-ary counter 220b can be configured by a six-bit binary counter; when a maximum count value thereof is set to 48 and the count value reaches 48, this count value is initialized to 1 at the next clock cycle.
[0297] The read-address generator 222 is also provided with a similar configuration and includes a forty-eight-ary counter 222b that counts a read clock signal φCR generated at a time of data reading and generates a count value thereof as a row-address pointer AXPR at the time of data reading and a sixteen-ary counter 222a that counts a count-up signal φUPX of the forty-eight-ary counter 222b and generates a count value thereof as a column-address pointer AYPR at the time of data reading.
[0298] The frame/frame address generator 213 further includes a multiplexer (MUX) 224 that causes one among the column-address pointer AYPW generated at the time of data writing and the column-address pointer AYPR generated at the time of data reading to pass through as a column-address pointer AYP according to a read/write signal R/WZ indicating data writing/reading and a multiplexer 224b that causes one among the row-address pointer AXPW generated at the time of writing and the row-address pointer AXPR generated at the time of data reading to pass through as a row-address pointer AXP according to the read/write instruction signal R/WZ. The address pointers AYP and AXP from the multiplexers 224a and 224b are imparted to one input of the selector 211 illustrated in FIG. 35. Next, operations are described.
[0299] First, operations at the time of image-data writing are described with reference to (A) in FIG. 116. In this situation, the write-address generator 220 illustrated in FIG. 115 operates and the read-address generator 220 does not operate. In the write-address generator 220, the sixteen-ary counters 220a and 220b are reset to an initial value. First, the first row-address pointer AXPW and column-address pointer AYPW both become 0 and an X address and Y address (X,Y) = (0,0) of the buffer memory 205 is designated. By this, the data transmitted from the reference-image memory is written in the address (0,0). Next, the write clock signal φCW is generated and the count value of the sixteen-ary counter 220a increases by 1. By this, the column-address pointer AYPW increases by 1. The row-address pointer AXPW does not change. Therefore, the address (0,1) is designated and pixel data imparted second is written. Subsequently, the pixel data is sequentially stored in the zeroth row in the buffer memory 205 until the count value of the sixteen-ary counter 220a reaches 15. After sixteen pixel data are stored in this buffer memory 205, the count-up signal φUPY is generated from the sixteen-ary counter 220a and the count value of the forty-eight-ary counter 220b increases by 1. By this, the row-address pointer AXPW becomes 1 and the count value of the sixteen-ary counter 220a becomes 0. Therefore, seventeenth pixel data that is next imparted to the address (1,0) is written. By subsequently repeating this operation, the count value of the forty-eight-ary counter 220b—that is, the row-address pointer AXPW—increases by 1 every time sixteen pixel data are written and the data is written sequentially in each row. Normally, image data imparted to a reference-image memory is pixel data scanned in a raster-scan scanning

method, and pixel data disposed in one row in a horizontal direction are sequentially imparted. Therefore, at the time of data writing, the data is sequentially stored according to this raster-scan scanning method (because the data is written according to the raster-scan scanning method in the reference-image memory as well and the data is likewise read according to the raster-scan method).

[0300] Next, operations in a situation of sequentially reading the pixel data from the search-window buffer memory 205 illustrated in FIG. 115 and imparting these to the computation unit 1 are described with reference to (B) in FIG. 116. At the time of pixel-data reading, the read-address generator 222 operates and the write-address generator 220 does not operate. Both the sixteen-ary counter 222a and the forty-eight-ary counter 222b have the initial value set to 0, and the address read first is (0,0). Next, when the read clock signal φCR is generated, the count value of the forty-eight-ary counter 222b increases by 1 and the value of the row-address pointer AXPR increases by 1. Because the count-up signal φUPX is not generated, the count value of the sixteen-ary counter 222a is held at 1. Therefore, the pixel data of the address (1,0) is read second. Subsequently, forty-eight data are read according to the read clock signal φCR. When the count value of the forty-eight-ary counter 222b reaches 47, at the next clock cycle, the count-up signal φUPX is generated and the count value of the sixteen-ary counter 222a increases by 1, becoming 1. The forty-eight-ary counter 222b has the count value thereof reset to the initial value in synchronization with the generation of this count-up signal. Therefore, forty-ninth pixel data is stored in the address (0,1). Subsequently, by repeating this operation, the pixel data is sequentially read along a direction wherein the row address increases. By a configuration of sequentially increasing and reading this row address, the pixels in the search area can be scanned along a row-address increasing direction, and in the computation unit 1, all data necessary for computation is imparted by simply sequentially shifting the imparted data.

[0301] [Variation of Frame/Frame Address Generator] FIG. 117 is a diagram illustrating a variation of the frame/frame address generator. In FIG. 117, the frame/frame address generator 213 includes a sixteen-ary counter 221 that counts a clock signal imparted to a clock input CLK and, when a count value thereof reaches 15 and is reset to an initial value, outputs a count-up signal from a count-up output terminal up thereof and a forty-eight-ary counter 223 that counts the signal imparted to a clock input terminal CLK and, when a count value thereof reaches 47, outputs a count-up signal from an output terminal up at a time transitioning to the next clock cycle. To the clock input terminal CLK of the sixteen-ary counter 221, the write clock signal φCW or the count-up signal of the forty-eight-ary counter 223 is imparted via a multiplexer 227. To the clock input terminal CLK of the forty-eight-ary counter 223, one among the read clock signal φCR and the count-up signal of the sixteen-ary counter 221 is imparted via a multiplexer 225. Selection of the multiplexers 225 and 227 is set by the read/write instruction signal R/WZ. When the read/write designation signal R/WZ indicates a data write mode, the multiplexer 225 imparts the count-up signal of the sixteen-ary counter 221 to the clock input terminal CLK of the forty-eight-ary counter 223. The multiplexer 227 imparts the write clock signal φCW to the clock input terminal CLK of the sixteen-ary counter 221. In a situation of the configuration illustrated in FIG. 117, only one sixteen-ary counter and one forty-eight-ary counter are necessary, which can reduce a device scale.

[0302] FIG. 118 is a diagram illustrating a specific configuration of the field/frame address generator illustrated in FIG. 114. In FIG. 118, the field/frame address generator 215

includes a write-address generator 230 for generating the row- and column-address pointers APXW and APYW at the time of pixel-data writing and a read-address generator 235 that generates the row- and column-address pointers APXR and APYR at the time of pixel-data reading.

[0303] The write-address generator 230 includes a write-address-pointer generator 234 that counts the write clock signal φCW generated at the time of pixel-data writing, counts the count-up signal φUPY from the sixteen-ary counter 232 that generates a count value thereof as the write-column-address pointer APW, and generates a count value thereof as the write-address pointer APXW. The write-address-pointer generator 234 includes a twenty-four-ary counter 233 that counts the count-up signal φUPY from the sixteen-ary counter 232 and a T flip-flop 231 whose output logic changes according to a count-up signal φUa of this twenty-four-ary counter 233. An output of the T flip-flop 231 is used as a least significant bit of the write-row-address pointer APXW, and an output of the twenty-four-ary counter 233 is used as a remaining high-order bit of the write-row-address pointer APXW.

[0304] The read-address generator 235 includes a forty-eight-ary counter 238 that counts the read clock signal φCR generated at the time of pixel-data writing and generates a count value thereof as the read-row-address APXR and a sixteen-ary counter 236 that counts the count-up signal φUPX from the forty-eight-ary counter 238 and outputs a count value thereof as the read-column-address APYR. The address pointers APYW and APXW from the write-address generator 230 and the address pointers APYR and APXR from the read-address generator 235 are imparted to a multiplexer 240. The multiplexer 240 selects one among the write-address pointers APYW and APXW and the read-address pointers APYR and APXR according to the read/write designation signal R/WZ and transmits this to the selector 211 (see FIG. 114) as address pointers APY and APR. Next, operations are described.

[0305] First, operations of the write-address generator 230 are described with reference to FIG. 119. The sixteen-ary counter 232 and the twenty-four-ary counter 233 are respectively initialized to a maximum count value 15 and to 0. The T flip-flop 231 is initialized to a 0-outputting state. When the write clock signal φCW is imparted, the output of this sixteen-ary counter 232 becomes 0. At this time, the count-up signal φUPY is not generated (it is generated in the previous cycle and used to initialize the twenty-four-ary counter to 0, or generation of this count-up signal φUPY is prohibited at the time of setting the initial value). Therefore, in a situation where the write clock signal φCW is imparted the first time, the row- and column-address pointers APXW and APYW both become 0 and the address (0,0) is designated. Next, when the write clock signal φCW is imparted, the count value of the sixteen-ary counter 232 increases by 1. The read-address pointer APXW from the read-address generator 234 does not change and stays as 0. Therefore, for the zeroth row, the pixels are subjected to sequential address designation.

[0306] The write clock signal φCW is counted sixteen times, and when the count value of the sixteen-ary counter 232—that is, the write-column-address pointer APYW—becomes 15, the count-up signal φUPY is generated at a predetermined timing. The address (0,15) is designated, and after the pixel data is written, the count value of the twenty-four-ary counter 233 increases by 1 according to this count-up signal φUPY. Meanwhile, when the write clock signal φCW is imparted, the count value of the sixteen-ary counter 232 returns to 0. Here, a change point of the output of the twenty-four-ary counter 233 may be

configured to be synchronized with the write clock signal φCW. By this, the address (2,0) is designated, and when the write clock signal φCW is imparted next, address designation for sequentially storing the pixel data of the second row is performed.
[0307] The above operations are repeated, and when the count value of the twenty-four-ary counter 233 becomes a maximum value (23), the count-up signal φUa is generated and the output state of the T flip-flop 231 changes. That is, the output of the T flip-flop 231 becomes "1." The output of the twenty-four-ary counter 233 returns to 0 according to the count-up signal from the sixteen-ary counter 232. Because the output of the twenty-four-ary counter 233 is 0 and the output of the T flip-flop 231 is 1, this read-row-address pointer APXW designates 1. Therefore, when the write clock signal φCW is next imparted, the address (1,0) is designated; subsequently, after address designation is sequentially performed for the pixel-data storage positions of this first row, when the count value of the twenty-four-ary counter 233 changes according to the count-up signal φUPY of the sixteen-ary counter 232, the read-row-address pointer APXW becomes 3 and address designation is sequentially performed for the pixel-data storage positions of the third row.
[0308] Therefore, as illustrated in (A) in FIG. 120, the pixel data included in a first field (even field or odd field) is sequentially stored in the even rows of the search-window buffer memory 205 and the pixel data belonging to a second field (odd field or even field) is sequentially stored in the odd rows as illustrated in (B) in FIG. 120. A correspondence relationship between this first field and second field and the even field and odd field is arbitrary, but in terms of correspondence with the previous description, the first field corresponds to the even field and the second field corresponds to the odd field. Therefore, by storing the pixel data imparted in field units (for example, image data according to a general NTSC method) in units of two fields in the buffer memory, the pixel data included in an image of one frame is stored in the search-window buffer memory 205 in the frame format.
[0309] Operations of the read-address generator 235 are identical to the operations of the read-address generator 222 described above with reference to FIG. 115; the pixel data of the zeroth column is sequentially read from the zeroth row to the forty-seventh row according to the read clock signal φCR, the pixel data of the first column is read next, and this operation is subsequently repeated. A read sequence of the pixel data at this read-operation time is identical to that illustrated in (B) in FIG. 116.
[0310] As above, regardless of whether image data of a frame format or a field format is input, all can be converted into image data of the frame format and transmitted to the computation unit 1. Therefore, the computation unit 1 can execute motion-vector detection in frame units regardless of the screen format of the input image data.
[0311] [Second Configuration of Input Unit] FIG. 121 is a diagram schematically illustrating a second configuration of the input unit. In the configuration illustrated in this FIG. 121, image pixel-data columns of a frame format or a field format are converted into and output as pixel-data columns of a field-format image.
[0312] The screen-format conversion unit 207 includes a field/field address generator 254 that generates an address for the search-window buffer memory 205 when an image pixel-data column of the field format is imparted, a frame/field address generator 252 that generates an address that is converted into an image pixel-data column of the field format when an image pixel-data column of the frame format is imparted, and a selector 211 that selects one of the address pointers output by the field/field address generator 254 and the

frame/field address generator 252 according to the screen-format designation signal φPT and imparts this to the search-window buffer memory 205. A configuration of the field/field address generator 254 is identical to the configuration of the frame/frame address generator 213 illustrated in FIG. 115. The only difference is the screen format of the pixel-data column to be processed being the frame format or the field format.

[0313] (A) in FIG. 122 illustrates a correspondence relationship between a content of an address pointer generated by the field/field address generator 254 and the pixel-data column stored in the search-window buffer memory 205. As illustrated in (A) in FIG. 122, when the pixel-data column of the field-format image is imparted, the field/field address generator 254 first designates a zeroth row and sequentially designates a zeroth column to a fifteenth column in this zeroth row. After the pixel data is written in the fifteenth column in the zeroth row, next, a first row is designated and sixteen pixel data are again written in the first row. Subsequently, this operation is repeatedly executed through a forty-seventh row. However, here, the size of the search window used in the computation unit 1 is forty-eight pixels (rows) × sixteen pixels (columns).

[0314] When reading the pixel data, as illustrated in (B) in FIG. 122, the field/field address generator 254 first places the zeroth column in a selected state and sequentially designates the zeroth row to the forty-seventh row. By this, the pixel data of the zeroth column are sequentially read. When the pixel data of the zeroth column (forty-eight pixels) are read, next, the forty-eight pixels in a first column are sequentially read. This operation is repeatedly executed. By this, the pixel-data columns are read from the search-window buffer memory 205 in order of the search-window-block pixel data stored in the processor array in the computation unit 1.

[0315] FIG. 123 illustrates a generation sequence of the address pointers generated by the frame/field address generator 252. The frame/field address generator 252 generates an address pointer indicating a pixel-data storage position for the search-window buffer memory 205 when an image pixel-data column of the frame format is imparted. When writing the pixel data, as illustrated in (A) in FIG. 123, the frame/field address generator 252, like when inputting a field-format image, generates the address pointers so as to sequentially store the pixel data one row at a time. At this pixel-data writing time, the frame/field address generator 252 stores pixel data of ninety-five pixels (rows) × sixteen pixels (columns) as the unit. This is because a frame-format image includes an even field and an odd field (first field and second field) and to create an image of the field format the pixel data of one field image needs to be extracted from this frame image. When reading the pixel data, as illustrated in (B) in FIG. 123, the frame/field address generator 252 generates address pointers so as to read only the pixel data of the even rows in each column. By this, only the pixel data of the first field is extracted and the pixel data of a search window of forty-eight pixel rows × sixteen pixel columns is obtained.

[0316] FIG. 124 is a diagram illustrating a specific configuration of the frame/field address generator illustrated in FIG. 121. In FIG. 124, the frame/field address generator 252 includes a write-address generator 260 that generates an address pointer when writing pixel data and a read-address generator 265 that generates an address when reading pixel data. The write-address generator 260 includes a sixteen-ary counter 262 that counts the write clock signal φCW generated when writing pixel data and outputs a count value thereof as the column-address pointer APYW and a ninety-five-ary counter 264 that counts the count-up signal φUPY of the sixteen-ary counter 262 and outputs a count value thereof

as the row-address pointer APXW. This write-address generator 260 performs operations similar to the write-address generator described above with reference to FIG. 115 and designates an address of the next row when the pixel-data storage positions of one row are all designated by the sixteen-ary counter 262. The ninety-five-ary counter 264 sequentially designates the addresses of ninety-five rows.

[0317] The read-address generator 265 includes a forty-eight-ary counter 268 that counts the read clock signal φCR generated when reading pixel data, a sixteen-ary counter 266 that counts the count-up signal φUPX of the forty-eight-ary counter 268 and outputs a count value thereof as the column-address pointer APYR, and a one-bit address generator 267 that generates a fixed value "0." The output of the forty-eight-ary counter 268 and the output of the one-bit address generator 267 are used as the row-address pointer APXR. The fixed bit value "0" from the one-bit address generator 267 is used as the least significant bit of the row-address pointer APXR used when reading pixel data. Therefore, the row-address pointer APXR designates only even rows. By the configuration of this read-address generator 265, when reading pixel data, the pixel data can be read by sequentially designating the zeroth row, the second row, the fourth row, …, and the ninety-fourth row and, after reading the pixel data of one column, the count value of the sixteen-ary counter 266 can be increased by 1 to sequentially read the pixel data of the next column.

[0318] By using a configuration such as is illustrated in this FIG. 121, image pixel-data columns of the field format can be generated regardless of whether the screen format of the input image pixel data is the field format or the frame format and motion vectors can be detected in field units regardless of the screen format of the input image. Moreover, there is no need to externally provide a circuit that converts image pixel-data columns of the frame format into image pixel-data columns of the field format, and a highly versatile motion-vector detection device can be realized that can correspond with any screen format without increasing a device scale.

[0319] [Third Configuration of Input Unit] FIG. 125 is a diagram illustrating a third configuration of the input unit of the motion-vector detection device used in this invention. In FIG. 125, the screen-format conversion circuit 207 includes a field/field address generator 272 that generates a write-address pointer when an image pixel-data column of the field format is imparted, a frame/field address generator 274 that generates a write-address pointer when an image pixel-data column of the frame format is imparted, a selector 276 that selects the output address of one among the address generators 272 and 274 according to the screen-format designation signal φPT, a write prohibition circuit 277 that prohibits writing data to the search-window buffer memory 205 according to a value of a least significant bit of the address pointer generated from the frame/field address generator 274, and a read-address generator 279 that generates a read address when reading pixel data from the buffer memory 205. Only one among the field/field address generator 272 and the frame/field address generator 274 is placed in the active state according to the screen-format designation signal φPT. The configuration where one among the address generators 272 and 274 is activated according to this screen-format designation signal φPT is similar in the configurations of the first and second screen-format conversion circuits above.

[0320] The search-window buffer memory 205 includes a memory-cell array 280 having a plurality of memory cells (for example, SRAM cells) arranged in a matrix of rows and columns, a write control circuit 282 that selects a memory cell in the memory-cell array

280 according to the write-address pointer imparted via the selector 276 of the screen-format conversion circuit 207 when writing data and writes the input image-pixel data to the selected memory cell, and a read control circuit 284 that selects a corresponding memory cell from the memory-cell array 280 according to the address pointer imparted from the read-address generator 279 when reading data and reads the stored data thereof. The write control circuit 282 and the read control circuit 284 respectively operate according to the clock signals φCW and φRW. When the write prohibition circuit 278 indicates writing prohibition, the write control circuit 282 does not execute writing of data even if pixel data is imparted from the outside.

[0321] The field/field address generator 272 is provided with the same configuration as the write-address generator 220 illustrated in FIG. 115. It is supposed that the memory-cell array 280 is provided with memory cells disposed in forty-eight rows by sixteen columns to store the data of the search window. In a situation where an image pixel-data column of the field format is imparted, the field/field address generator 272 is activated and, under control of the write control circuit 282, the pixel-data column of the field-format image is sequentially written in the memory-cell array 280 similarly to what is described above. In a situation where a pixel-data column of an image of the frame format is imparted, writing of the pixel data is executed according to the address pointer generated by the frame/field address generator 274. At this time, in a situation where writing of pixel data for one row is performed according to the address generated by the frame/field address generator 274, writing for the next row is the pixel data of another field in the frame, and writing of data is prohibited. That is, the pixel data of one field (even field or odd field; first field or second field) among the frame-format image is extracted according to the address generated by the frame/field address generator 274, data writing to the memory-cell array 280 is performed by the write control circuit 282 only for this one field, and writing of the pixel data is prohibited under control of the write prohibition circuit 278 for another field. Therefore, in the memory-cell array 280, pixel data of an image of the field format is stored regardless of whether the format of the input image is the frame format or the field format.

[0322] The read-address generator 279 is provided with a configuration similar to the read-address generator 235 illustrated in FIG. 118 and generates the address so the memory cells are sequentially selected by column in the memory-cell array 280. The read control circuit 284 selects a memory cell in the memory-cell array 280 according to the address from the read-address generator 279 according to the read clock signal φRW and reads the pixel data had by this selected memory cell. Therefore, pixel-data columns of an image of the field format are sequentially output from the read control circuit 284.

[0323] FIG. 126 is a diagram illustrating a specific configuration of the frame/field address generator 274 illustrated in FIG. 125. In FIG. 126, the frame/field address generator 274 includes a sixteen-ary counter 292 that counts the write clock signal φ CW and generates a count value thereof as the column-address pointer APYW and a seven-bit counter (ninety-five-ary counter) 294 that counts the count-up signal φUPY of the sixteen-ary counter 292. The seven-bit counter 294 includes a one-bit counter 296 that includes only the least significant bit (LSB) and a six-bit counter 298 that outputs a remaining high-order six-bit count value. The count value of the one-bit counter 296 is imparted to the write prohibition circuit 278 as a write prohibition instruction signal. When the count value (LSB) of this one-bit counter 296 is “1,” the write prohibition circuit 278 deems that writing prohibition is designated and prohibits the data write operation by the write control circuit 282. The

count value of the six-bit counter 298 is output as the row-address pointer APXW. Next, operations of the frame/field address generator 274 illustrated in this FIG. 47 are described with reference to FIG. 127, which is an operation timing chart thereof.
[0324] At initialization, the count values of the sixteen-ary counter 292 and the seven-bit counter 294 are both set to 0. The sixteen-ary counter 292 counts the write clock signal φCW and generates the column-address pointer APYW. When the count value of this sixteen-ary counter 292 reaches 15, writing of data of sixteen pixels of one row—that is, the zeroth row—is completed. This written pixel data belongs to the first field.
[0325] Next, the count value of the sixteen-ary counter 292 returns to 0. In response to the count value of this sixteen-ary counter 292 returning to the initial value, the count value of the seven-bit counter 294 increases by 1. At this time, the count value of the one-bit counter 296 that outputs the least significant bit LSB becomes 1 and the count value of the six-bit counter 298 does not change. Therefore, the row-address pointer APXW is maintained as 0. At this time, because the count value (LSB) of the one-bit counter 296 is 1, the write prohibition circuit 278 prohibits the write operation of the write control circuit 282. By this, writing of the pixel data is prohibited in a period until the count value of the sixteen-ary counter 292 reaches 15, and writing of the pixel data belonging to the second field is prohibited.
[0326] Next, when the count value of the sixteen-ary counter 292 again returns to 0, the count value of the seven-bit counter 294 increases by 1 according to the count-up signal φUPY. By this, the count value (LSB) of the one-bit counter 296 becomes 0 and the count value of the six-bit counter 298 becomes 1. Because the count value (LSB) of this one-bit counter 296 is 0, the write prohibition circuit 278 cancels writing prohibition. By this, the write control circuit 282 sequentially writes the pixel-data columns of the image of the frame format. The pixel data written at this time belongs to the first field and is written in the first row.
[0327] Next, when the count value of the sixteen-ary counter 292 again reaches 15 and writing of the pixel data of one row is completed, the count-up signal φUPY is generated. By this, the count value of the seven-big counter 294 increases by 1, the count value of the one-bit counter 296 becomes 1, and the count value of the six-bit counter 298 is maintained as 1. Therefore, in this situation, writing of the pixel data is prohibited according to the count value 1 of the one-bit counter 296. Subsequently, by repeating this operation, writing for the pixel data of the second field among the pixel-data columns of the image of the frame format is prohibited and only the pixel data of the first field is written in the memory-cell array 280.
[0328] That is, as illustrated in FIG. 128, by selecting only the pixel data of the first field deemed necessary from a frame-format image 290 and writing this in the memory-cell array 280, a search window can be formed with only the first-field pixel data and pixel data of a field-format image can be obtained from pixel-data columns of an image of the frame format.
[0329] Reading of the data is identical to the data read operation of the field/field image illustrated above (see [B] in FIG. 122).
[0330] As above, by extracting only image data of a field deemed necessary from a frame-format image and storing this in a memory cell, motion vectors can continually be detected in field units regardless of an image format of input image data.

[0331] [Configuration of Data Buffer] FIG. 129 is a diagram illustrating the configuration of a data register for storing side window block data included in the processor array. In FIG. 129, a data register (buffer) DL includes a plurality of cascade connected registers provided with a data transfer function. The plurality of registers is divided into two groups. In FIG. 129, the two register groups G#1 and G#n are representatively shown. The register groups G#1, …, G#n respectively include cascade connected registers RG1 and RG2. These register groups G#1 to G#n are provided with a function for transmitting search window pixel data along one direction in response to a clock signal not illustrated.
[0332] The data buffer DL further includes a selector 300 that selects an output of any of the register groups G#1 to G#n according to a selection signal φSEL. This selection signal φSEL is imparted from the exterior. The selector 300 makes it possible to change the number of steps of the registers of the data buffer DL. For example, if the register group G#1 is selected, the number of steps of the registers of the data buffer DL is two, and two pixels of the search window pixel data can be stored. If the register group G#n is selected, the data buffer DL can only store search window pixel data equal to the number of steps of the registers included therein.
[0333] The data buffer DL stores pixel data of the side window block (see FIG. 8). Therefore, it is possible to change the size of the side window block by changing the number of steps of the registers included in the data buffer, and it is thereby possible to change the size of the search window, that is, the range of the search area. This makes easy correspondence possible even if the search area has been changed based on specifications. Note that there can be any number of registers included in a register group and any number of register groups.
[0334] [Variation 1 of Data Buffer] FIG. 130 is a diagram illustrating a variation of the data buffer. In FIG. 130, the data buffer DL includes a plurality of cascade connected registers RGa to RGn. Data transfer is possible via these registers RGa to RGn. Conducting selection gates SELa to SELn are respectively provided to the output unit of the registers RGa to RGn in response to activation signals φa to φn. One among the selection signals φa to φn imparted from the exterior is made active. A conducting selection gate SELi transmits output of a corresponding register RGi to a data wire 302. This data wire 302 is connected to an input unit of an element processor PEm furthest downstream in the same linear processor array in the element processor array.
[0335] According to the configuration illustrated in FIG. 130, the register step number of the data buffer can be set for each register unit according to the selection signals φa to φn, and can more flexibly handle specification changes in the search area.
[0336] [Another Configuration Example of Data Buffer] FIG. 131 is a diagram illustrating another specific configuration of the data buffer. In FIG. 131, the data buffer DL includes an input circuit 310 for inputting search window pixel data, a register file 312 provided with a plurality of files, which stores search window pixel data imparted from the input circuit 310 to each file, a write address generating circuit 316 which generates a write address pointer when writing data to the register file 312, a read address generating circuit 314 which generates a read address pointer used when reading data from the register file 312, an output circuit 318 which reads the content of a file from the register file 312 which has been designated by the read address generating circuit 314, and an initial address setting circuit 319 which sets an initial address of the read address generating circuit 314. The register file 312 stores one set of pixel data to one file, configured using, for example, an

SRAM cell. The write address generating circuit 316 and the read address generating circuit 314 respectively increase or decrease the address sequentially one at a time according to a write clock signal φW and a read clock signal φR. In the configuration of this data buffer DL, the writing of data is executed after the reading of data has been performed. This is to prevent data to be read from being changed due to newly imparted data (however, this is when the read address and write address are set to designate the same address). Next, the operation of the data buffer illustrated in FIG. 131 will be described referring to FIG. 152.

[0337] In FIG. 152, a configuration is given as an example wherein the number of files in the register file 312 is eight, and addresses 0 to 7 are respectively assigned to the eight files. Furthermore, in FIG. 152, the initial address setting circuit 319 designates address 4 as the initial address of the read address generator 314.

[0338] The write address generating circuit 316 designates a writing address sequentially from address 0 according to the write clock signal φW. The input circuit 310 writes the imparted search window pixel data to the corresponding file in the register file 312 according to the write address pointer from the write address generating circuit 316. Therefore, search window pixel data SY1, SY2, … input for the addresses 0, 1, … are written sequentially in the register file 312. The read address generating circuit 314 designates an address sequentially from an initial address 4 set by the initial address setting circuit 319 according to the read clock signal φR. When the read address pointer generated by the read address generating circuit 314 becomes zero, the pixel data SY1 stored in address 0 of the register file 312 is read from the output circuit 318. The subsequent pixel data SY2, SY3, … stored in the register file 312 are read sequentially according to the read clock signal φR. Therefore, in this configuration, the input search window pixel data is output five clock cycles late. Therefore, by changing the initial address of the read address from the initial address setting circuit 319, it is possible to change the delay time of this data buffer, it is possible to change the number of effective search window pixel data stored in the data buffer, and it is possible to change the size of the search area (search window) in the vertical direction. The size of the search area changing in the vertical direction due to changing the delay time of the data buffer will be described in detail below.

[0339] As illustrated in FIG. 133 (A), an example is considered wherein the read address pointer RP shows address 0 when the write address pointer WP shows address 4. In this state, search window pixel data SY1 to SY5 are stored in the register file 312.

[0340] As illustrated in FIG. 133 (B), when the read address pointer RP shows address 3,the write address pointer WP shows address 7. In this state, the pixel data SY1 to SY3 stored to the files of addresses 0, 1, and 2 have already been read. Therefore, before reading the pixel data SY4, the pixel data of pixel data SY4 to SY8 is stored as effective pixel data (pixel data to be read) in the register file 312.

[0341] As illustrated in FIG. 133(C), when the clock cycle proceeds further and the write address pointer WP shows address 3, the read address pointer RP shows address 7. In this state, the pixel data SY5 to SY7 stored to the addresses 4 to 6 are pixel data that have already been read and deleted. Even in this state, pixel data SY8 to SY12 stored in the register file 312 become the effective pixel data.

[0342] In the next cycle, as illustrated in FIG. 133 (D), the write address pointer WP designates address 4, and the read address pointer RP designates address 0. In this state, pixel data SY9 to SY13 become the effective pixel data.

[0343] That is, the input search window pixel data is stored in all of the files in the register file 312 (eight files in the example illustrated), but the number of effective search window pixel data is equal to or less than the number of all of the register files (five in the example illustrated). The number of effective search window pixel data stored in the register file 312 is equivalent to a value adding one to the initial address designated by the initial address setting circuit 319. Therefore, the register file 312 is configured to be equivalent to a shift register having five step registers RG1 to RG5 cascade connected thereto as illustrated in FIG. 133(E). The pixel data of the side window block illustrated in FIG. 8 is stored in the data buffer. The number of steps of the register illustrated in FIG. 133(E) is equal to the number of pixels of the side window block in the vertical direction. Therefore, by using the configuration illustrated in FIG. 131, it is possible to change the length of the side window block in the vertical direction, and it is thereby possible to adjust the length of the search window in the vertical direction.
[0344] According to the configuration described above, any size of the search area can be handled in a flexible manner.
[0345] [Hierarchical Motion-Vector Detection Method] FIG. 134 is a diagram illustrating the configuration of a search area and template block used in the detection method for a hierarchical motion vector according to this invention. In FIG. 134, the template block 350 is configured by 16 pixels (rows) by 16 pixels (columns) similar to that used in the full search motion-vector detection device described previously. The motion vector search range is ± 64 pixels for both the horizontal and vertical directions. Therefore, the search area 352 is configured by (64 + 16 + 64) x (64 + 16 + 64) = 142 x 142 pixels.
[0346] Searching for a motion vector is performed in two steps in the hierarchical motion vector detection method. In the first step, the evaluation function value is calculated by thinning the evaluation point (corresponding to displacement vector, points where evaluation-function-value is sought: each point between (-64, -64) and (64, 64)) in the search area 352, and performing block-matching processing. The optimal evaluation function value indicating the most favorable correlation and the corresponding displacement vector are sought from the calculated evaluation function value. The detection of this optimal evaluation function value and the corresponding displacement vector is executed in parallel in a plurality of respective prediction modes. The block-matching processing in the first step will be described in detail next.
[0347] As illustrated in FIG. 135(A), first, the evaluation function value for the template block 350 of the search window block 354a at the evaluation point (-64, -64) is calculated. The calculation operation of this evaluation function value is similar to that described in the previous full search method of the motion vector detection device in that the calculation of the difference absolute values between corresponding pixels and sum of the sought difference absolute value are sought. Another evaluation function value such as a difference squared sum may be used instead of the difference absolute value as is evaluation function value.
[0348] Next, as illustrated in FIG. 135(B), seven evaluation points from (-64, -63) to (-64, -57) are skipped (thinned), and the evaluation function value of the search window block 354b corresponding to the evaluation point (-64, -56) is calculated. When the evaluation function value calculation for skipping seven evaluation points is repeated and the calculation of the evaluation function value of the evaluation point (64, 64) on the lower portion of the search area 352 is completed, an evaluation function value calculation from

the upper side of the diagram of the search area 352 can once again be performed. When converting evaluation point sequences, as illustrated in FIG. 135(C), seven pixel portion evaluation points are skipped in the horizontal direction, and then the evaluation portion where the evaluation function value is calculated becomes (-56, -64). When evaluation points are skipped seven pixel portions at a time in the horizontal direction thereafter, the evaluation function value is calculated, and the evaluation point is reached on the lower portion of the search area 352, the evaluation point for seven pixel portions is skipped in the horizontal direction and evaluation function values are calculated sequentially from the upper portion of the search area 352 once again.

[0349] When the series of operations illustrated in FIG. 135(A) to (C) are repeated, as illustrated in FIG. 136, the evaluation function value of all representative evaluation points E (the evaluation points where the evaluation even number value is calculated, the evaluation points disposed every eight evaluation points in the horizontal and vertical direction) in the search area 352 are calculated. The evaluation function value indicating the highest correlation is detected from the evaluation function values of these representative evaluation points E, and the corresponding evaluation point Est is determined. This optimal representative evaluation point Est is sought in a plurality of respective prediction modes. The series of operations above are a schematic search operation sequence of the first step, and the search operations of the second step are performed next.

[0350] In the search operation of the second step, as illustrated in FIG. 137, a peripheral region (full search region) 356 of the optimal representative evaluation point Est is made to be the search area, and all of the evaluation points in this peripheral region 356 are searched. The peripheral region 356 contains at least a skipped evaluation point. In the configuration illustrated in FIG. 137, the search range of the peripheral region 356 is ± 16 horizontally and vertically using the optimal representative evaluation point as a reference. Therefore, the peripheral region 356 includes a 32 x 32 evaluation point. The evaluation function value is calculated for all evaluation points of 32 x 32 = 931 in this peripheral region 356, the optimal evaluation point is sought according to the calculated evaluation function values, and a displacement vector corresponding to this optimal evaluation point is determined as a motion vector. This full search operation is executed in parallel in a plurality of respective prediction modes.

[0351] The number of evaluation points in the first step is (8+1+8) x (8+1+8) = 17 x 17 = 289, and the number of evaluation points in the second step is 32 x 32 = 1024, making a total of 1313. Therefore, it is possible to detect motion vectors having a wider image region than the full search method as the search area, and because a motion vector is determined in parallel in a plurality of respective prediction modes, it is possible to efficiently determine a motion vector without increasing the device scale. A specific configuration will be described in detail next.

[0352] [First Example of Hierarchical Search Method Motion Vector Detection Device] FIG. 138 is a diagram schematically illustrating the configuration of a first example of a hierarchical search method motion vector detection device. In FIG. 138, the hierarchical search method motion vector detection device includes a rough search computation unit 402 which simultaneously calculates evaluation function values for representative evaluation points in the search area respectively according to a field prediction mode, an upper subtemplate block prediction mode, and a lower subtemplate block prediction mode,

and these calculated evaluation function values ef, eu, and el are imparted to a comparison selection unit 404 along with corresponding displacement vectors (indicating evaluation point locations) vf, vu, and vl. The specific structure of this rough search computation unit 402 will be described hereafter, and it executes the search operation of the first step described referring to FIGS. 135(A) to (C) and FIG. 136. Here, the motion vector detection device detects motion vectors in units of fields. The evaluation function value and displacement vector calculated in accordance with the field prediction mode are indicated using ef and vf, the calculated evaluation function value and corresponding displacement vector according to the upper subtemplate block prediction mode are indicated using eu and vu, and the evaluation function value and corresponding displacement vector calculated according to the lower subtemplate block prediction mode are indicated using el and vl.

[0353] The comparison selection unit 404 receives the evaluation function values ef, eu, and el, and the displacement vectors vf, vu and vl from the rough search computation unit 402, determines an optimal representative evaluation point for the respective field prediction mode, upper subtemplate block prediction mode, and lower subtemplate block prediction mode, and the displacement vectors vfst, vust, and vlst indicating the optimal representative evaluation points for each prediction mode are output. The optimal displacement vectors vfst, vust, and vlst respectively indicate the optimal displacement vector by the field prediction mode, the optimal displacement vector by the upper subtemplate block prediction mode, and the optimal displacement vector by the lower subtemplate block prediction mode.

[0354] The motion vector detection device further includes a fine search computation unit 406 that extracts the optimal representative evaluation point peripheral regions for each optimal displacement vector according to the optimal displacement vectors vfst, vust, and vlst imparted from the comparison selection unit 404, and performs a full search in the extracted peripheral region. The fine search computation unit 406 includes first to third full search computation units 410, 412, and 414 provided to the respective optimal displacement vectors vfst, vust, and vlst. The first full search computation unit 410 extracts an optimal representative evaluation point peripheral region according to the optimal displacement vector vfst, calculates an optimal evaluation point respectively according to the field prediction mode, the upper subtemplate block prediction mode, and the lower subtemplate block prediction mode, and evaluation function values Ef1 Eu1, and El1 corresponding to displacement vectors Vf1, Vu1, and Vl1 indicating optimal evaluation points are imparted to a comparison selection unit 408.

[0355] The second full search computation unit 412 extracts an optimal representative evaluation point peripheral region according to the optimal displacement vector vust sought according to the upper subtemplate block prediction mode, finds an evaluation function value for this optimal representative evaluation point peripheral region according to the three prediction modes, and imparts it to the comparison selection unit 408 with the corresponding displacement vector.

[0356] The third full search computation unit 414 extracts an optimal representative evaluation point peripheral region according to the optimal displacement vector vlst determined according to the lower subtemplate block prediction mode, calculates an evaluation function value for this optimal representative evaluation point peripheral region using the full search method, and imparts the evaluation function values Ef3, Eu3, and El3

for the respective prediction modes to the comparison selection unit 408 along with the corresponding displacement vectors Vf3, Vu3, and Vl3.
[0357] The comparison selection unit 408 finds the evaluation function value having the highest correlation (the lowest evaluation function value) from among the total of nine types of evaluation function values Ef1, Eu1, El1, Ef2, Eu2, El2, Ef3, Eu3, and El3, and outputs the corresponding prediction mode, displacement vector MV, and evaluation function value EV.
[0358] FIG. 139 is a diagram schematically illustrating the configuration of the rough search computation unit 402 and the comparison selection unit 404 illustrated in FIG. 138. In FIG. 139, the rough search computation unit 402 includes a memory cell array 412 for storing pixel data of a search window block size from reference image data stored in an external plain memory, a write address generator 414 for generating a write address pointer when writing pixel data to the memory cell array, a read address generator 416 which generates a read address pointer when reading pixel data from the memory cell array 412, an output circuit 418 for latching and transmitting one row of data read from the memory cell array 412, an element processor array 419 including an element processor arranged in a matrix of rows and columns corresponding to pixels of a template block, and a summing unit 420 that respectively sums the difference absolute values output from the element processor of the element processor array 419 according to the prediction modes, and outputs the evaluation function values Σ |a-b|, Σu |a-b|, and Σl |a-b| corresponding to each prediction mode. Template block pixel data is transmitted from a template block data memory 425 to a template block pixel data storage unit of the element processor of the element processor array 419.
[0359] The comparison selection unit 404 includes a comparison unit 422 which receives evaluation function values from the summing unit 420 in parallel and outputs displacement vectors vfst, vust, and vlst indicating the optimal evaluation points to each prediction mode. The configuration of this comparison selection unit 404 is similar to the configuration of the comparison unit described previously referring to FIG. 30 and FIG. 27, so descriptions of the operation will not be repeated.
[0360] FIG. 140 is a diagram illustrating a specific configuration of the element processor array 419. In FIG. 140, the element processor array 419 includes an element processor PE arranged in a matrix of rows and columns. Sixteen element processors PE0 to PE15 arranged in the horizontal direction of FIG. 140 configure a linear processor array LA. Sixteen linear processor arrays LA0 to LA15 are arranged. The element processors PE (PE0 to PE15) can transmit search window block pixel data and template block pixel data to the adjacent (right side in FIG. 140) element processor in the linear processor arrays LA (LA0 to LA15) thereof.
[0361] The memory cell array 412 includes memory cells arranged in sixteen rows by sixteen columns. FIG. 140 illustrates sixteen word lines WL0 to WL15 disposed corresponding to the sixteen rows of memory cells in the memory cell array 412. When reading data from the memory cell array 412, data from one row of memory cells is selected at the same time and imparted to the output circuit 418. Because pixel data is received from outside the device when writing search window pixel data from an external plain memory, as previously illustrated in FIG. 114 to FIG. 127, one word line is selected and pixel data is written sequentially to this selected word line. This memory cell array 412 is preferably configured by an SRAM cell array (to make it high speed access possible).

[0362] FIG. 141 is a diagram illustrating a configuration of the element processor PE15 included in the element processor array 419. In FIG. 141, the element processor PE15 includes a data register 431 for storing and transmitting search window pixel data from the output circuit 418, a data register 432 for storing template block pixel data from the template block data memory 425, and a difference absolute value circuit 433 for seeking a difference absolute value of stored data of the data registers 431 and 432. The configuration of this element processor PE15 illustrated in FIG. 141 is the same as the configuration of the element processor illustrated in FIG. 20, only differs in that pixel data is received from the output circuit 418 instead of simply receiving search window pixel data from the data buffer, and the operation is similar in all other aspects. Because there is no data buffer provided, the search window pixel data is transmitted only in the linear processor arrays LA (LA0 to LA15). Next, the transfer operation of data from the memory cell array illustrated in FIG. 140 to the element processor will be described collectively referring to FIG. 142.

[0363] As illustrated in FIG. 142(A), it is assumed that search window block 354a of the displacement vector (-64, -64) is stored in the element processor array 419. At this time, the write address generator 414 illustrated in FIG. 139 changes row address 0 to row address 7 sequentially, and stores sixteen pieces of pixel data from row -48 to row -41 in the search area 352 in each row of row address 0 to 7. In this state, pixel data of the displacement vector (-56, -64) is stored in the memory cell array 412.

[0364] When the calculation of the evaluation function value for the displacement vector (-64, -64) is complete, the read address pointer RP from the read address generator 416 is generated from 0 to 7 sequentially. The pixel data of row -48 is stored in the row address 0. By reading one row of pixel data and transferring to the element processor array 419 via the output circuit 418, the pixel data of row -48 to row -41 is stored in order in the element processor array 419. At this time, a transfer of search window block pixel data between adjacent element processors is executed in the element processor array 419. Therefore, when the pixel data of row -41 in row address 7 is stored in the element processor array 419, all pixel data of the displacement vector (-56, -64) is stored in the element processor array 419. This is because pixel data shifts in the element processor array one row at a time simultaneously with storing one row of pixel data to the element processor array 419.

[0365] The write address pointer WP from the write address generator 414 changes from row address 8 to 15 in parallel with the reading of pixel data from this memory cell array 412, and pixel data of row -40 to row -33 is stored. When the calculation of evaluation function value of the search window block 354b of this displacement vector (-56, -64) is complete, pixel data of row address 8 to 15 are read sequentially by row units from the memory cell array 412 according to the read address pointer RP and imparted to the element processor array 419. Pixel data corresponding the displacement vector (-48, -64) is thereby stored in the element processor array 419. Pixel data of row -32 to row -25 is stored on the position of row address 0 to 7 according to the write address pointer WP in parallel with this data reading operation. Hereafter, the calculation of evaluation function values for representative evaluation points in the search area 352 is executed by repeating the operations above.

[0366] By executing the reading of pixel data row units and shifting the element processor array 419, it is possible to store the search window block for the next representative

evaluation point in the element processor array 419 simply by reading half of the pixel data of the search window block.

[0367] FIG. 143 is a diagram illustrating a configuration for transferring peripheral region pixel data of the fine search computation unit 406 according to the displacement vectors vfst, vust, and vlst corresponding to optimal representative evaluation points from the comparison selection unit 404. In FIG. 143, the data transfer control system includes an input/output control circuit 454 that activates a reading control circuit 452 when it detects completion of the optimal representative evaluation point detection in the rough search computation unit 402 (the number of representative evaluation points in the search area is predetermined, can be easily identified by counting the clock signal prescribing the rough search operation cycle), and generates a data write timing and an address of a first full search computation unit memory 455a, a second full search computation unit memory 455b, and a third full search computation unit memory 455c.

[0368] When activated, the reading control circuit 452 decodes the displacement vectors vfst, vust, and vlst corresponding to optimal representative evaluation points sequentially, generates an address including peripheral regions of these displacement vectors, and reads the peripheral region pixel data of the respective displacement vectors vfst, vust, and vlst sequentially from a reference image memory 450. The peripheral region pixel data read from this reference image memory 450 is stored sequentially to the memories 455a, 455b, and 455c under control of the input/output control circuit 454. Reading of stored data in the memories 455a to 455c is executed in parallel with the pixel data storing operation to the memories 455a to 455c, and peripheral region pixel data is transferred to the respectively corresponding first full search computation unit, second full search computation unit, and third full search computation unit. The memories 455a to 455c store all pixel data in the peripheral regions (32 x 32 evaluation point pixel data, not that large of a storage capacity is required). Data stored to the full search computation unit memories 455a to 455c is transmitted to the input unit of the corresponding full search computation unit after the stored pixel data is read in synchronization with the search operation in the fine search computation unit 406. The configuration of the first full search computation unit 410, the second full search computation unit 412, and the third full search computation unit 414 is the same as the motion vector detection device described previously referred to in FIG. 28. Only the size of the search area is different, and the evaluation function value calculation is executed according to a plurality of prediction modes (field prediction mode, an upper subtemplate block prediction mode, and a lower subtemplate block prediction mode) by performing similar operations. Displacement vectors corresponding to the respective plurality of prediction modes are thereby output.

[0369] The comparison selection unit 408 seeks a minimum value of the evaluation function value corresponding to the nine types of optimal displacement vectors imparted from the fine search computation unit 406, and outputs this minimum evaluation function value and the corresponding displacement vector as the motion vector MV and the evaluation function value EV. This configuration can be easily realized simply by seeking the minimum value from among nine inputs.

[0370] [Second Example of Hierarchical Search Method Motion Vector Detection Device] FIG. 144 is a diagram illustrating the configuration of a second example of a hierarchical search method motion vector detection device. The motion vector detection device illustrated in FIG. 144 differs from the configuration illustrated in FIG. 138 in that the fine

search computation unit 406 only outputs the evaluation function values and optimal displacement vectors only for the respective three prediction modes, that is, three types of evaluation function values and optimal displacement vectors.
[0371] The fine search computation unit 406 includes first, second, and third full search computation units 460, 462, and 464 provided in parallel. The first full search computation unit 460 sets the displacement vector vfst corresponding to the optimal representative evaluation point determined according to the field prediction mode imparted from the comparison selection unit 404 as the central evaluation point, performs block-matching processing in according to field prediction mode for all evaluation points included in the peripheral region thereof, and seeks the displacement vector vf and the evaluation function value Ef thereof corresponding to the optimal evaluation point.
[0372] The second full search computation unit 462 receives the displacement vector vust corresponding to the optimal representative evaluation point determined according to the upper subtemplate block prediction mode imparted from the comparison selection unit 404, seeks an optimal evaluation point according to the upper subtemplate block prediction mode for all evaluation points included in a peripheral region having this optimal representative evaluation point as the center, and outputs the displacement vector Vu and the evaluation function value Eu corresponding to the sought optimal evaluation point.
[0373] The third full search computation unit 464 seeks an optimal evaluation point according to the lower subtemplate block prediction mode for all evaluation points included in a peripheral region having an evaluation point indicated by the displacement vector vlst as the center according to the displacement vector vlst corresponding to the optimal representative evaluation point determined in accordance with the lower subtemplate block prediction mode from the comparison selection unit 404, and outputs the corresponding displacement vector Vs and the evaluation function value Es.
[0374] The comparison selection unit 408 receives the optimal displacement vectors Vf, Vu, and Vs, and evaluation function values Ef, Eu, and Es output from the fine search computation unit 406, seeks a (smallest) evaluation function value showing the highest degree of similarity from among these three evaluation function values Ef, Eu, and Es, and outputs this minimum evaluation function value EV and the corresponding displacement vector as the motion vector MV.
[0375] FIG. 145 is a diagram for describing the configuration of the first full search computation unit 460. As illustrated in FIG. 145(A), when determining the evaluation function value according to the field prediction mode, the size of a template block 472 is 16 pixels x 16 pixels, and the size of the search window block 470 is similarly 16 pixels x 16 pixels. A sum of the difference absolute value of the corresponding pixels of the search window block 470 and the template block 472 is sought.
[0376] As illustrated in FIG. 145(B), the search range is 32 pixels x 32 pixels (when the search range is ± 8 in the horizontal and vertical direction). The size of the search window is 32 pixels x 16 pixels. The first full search computation unit 460 includes a processor array similar to the configuration illustrated in FIG. 4. The element processor group is arranged in 16 rows and 16 columns, and the data buffer group is arranged to store pixel data in 8 rows and 16 columns. The calculation of evaluation function values is performed sequentially by shifting the search window pixel data in a search area 474.

[0377] FIG. 146 is a diagram schematically illustrating the configuration of the second full search computation unit. The template block of the second full search computation unit 462 is provided in a size of 8 pixels (rows) x 16 pixels (columns). This is because only the pixel data included in the block on the upper half of the template block of 16 pixels x 16 pixels. In this case, the search window block similarly has a size of 8 pixels (rows) x 16 pixels (columns). When the search range is 32 pixels x 32 pixels, the pixel data of the lower 8 rows in the 32 pixel x 32 pixel region is not used in the upper subtemplate block prediction mode (this is because the calculation of evaluation function values is performed in units of blocks of 16 pixels x 16 pixels). That is, when the search window block 477 reaches the bottommost portion in the search window 476, the pixel data of a remaining region 478 in the search range of 32 pixels by 32 pixels is not used. Therefore, a search area 479 in this upper subtemplate block prediction mode is provided in a size of 24 pixel rows x 32 pixel columns. The second full search computation unit 462 is provided with a configuration similar to the configuration illustrated in FIG. 4, but the element processor group is arranged in 8 rows and 16 columns, and the data buffer group is disposed to store pixel data in 16 rows and 16 columns. The pixel data in this search area 479 is shifted sequentially, and the second full search computation unit 462 performs the calculation of evaluation function values and the detection of an optimal displacement vector.

[0378] FIG. 147 is a diagram for describing the configuration of the third full search computation unit. In the lower subtemplate block prediction mode, the pixel data included in the block in the lower half of the 16 pixel x 16 pixel template block is used. Therefore, the size of a search window block 482 is 8 pixel rows x 16 pixel columns. From among pixel data in the peripheral region of 32 pixel rows x 32 pixel columns, the pixel data of the upper 8 rows is not used. For example, when evaluation function values are calculated in the positions of blocks 482 and 483 in FIG. 147, the pixel data of block 483 is not used. Only pixel data included in block 482 is used. Therefore, a search area 484 in this lower subtemplate block prediction mode is configured by the pixel data of the lower 24 rows x 32 pixels from among 32 pixel rows x 32 pixel columns. The third full search computation unit 464 is therefore provided with a configuration similar to the configuration illustrated in FIG. 4, but includes an element processor group arranged in 8 rows and 16 columns, and a data buffer group which stores pixel data arranged in 16 rows and 16 columns. The calculation of evaluation function values is performed by shifting pixel data in the search area 484 sequentially one pixel at a time.

[0379] In the configuration illustrated in the second example, three computation units are provided in the fine search computation unit 406, but optimal displacement vectors are calculated for these three computation units according to respectively different prediction modes, and it is possible to reduce the scale of the device, and because an optimal displacement vector is detected according to a plurality of prediction modes in parallel, and a motion vector is determined according to these detected optimal displacement vectors, it is possible to detect optimal motion vectors at high speed and it is possible to obtain an optimal prediction image.

[0380] Note that a configuration similar to the configuration illustrated in FIG. 64 can be used for transmitting the pixel data in the respective search areas to the full search computation units 460, 462, and 464.

[0381] [Third Example of Hierarchical Search Method Motion Vector Detection Device] FIG. 148 is a diagram schematically illustrating a third example of a hierarchical search method

motion vector detection device. In FIG. 148, a comparison selection unit 492 receives evaluation function values ef, eu, and el and corresponding displacement vectors vf, vu, and vl imparted from the rough search computation unit 402, seeks an optimal representative evaluation point, detects an optimal prediction mode, and outputs a displacement vector vst and evaluation function value est corresponding to the optimal representative evaluation point corresponding to the detected optimal prediction mode.

[0382] The fine search computation unit 406 includes one full search computation unit 494. The full search computation unit 494 determines optimal displacement vectors Vf, Vu, and Vl and corresponding optimal function values Ef, Eu, and El according to a field prediction mode, an upper subtemplate block prediction mode, and a lower subtemplate block prediction mode, for all pixel data (or evaluation points) included in the peripheral region thereof according to the displacement vector vst corresponding to the optimal representative evaluation point imparted from the comparison selection unit 492. The configuration of the motion vector detection device illustrated in FIG. 1 can be used as the configuration of the full search computation unit 494.

[0383] The comparison selection unit 408 determines an optimal prediction mode from the evaluation function values Ef, Eu, and El imparted from the full search computation unit 494, and outputs a displacement vector and evaluation function value corresponding to this optimal prediction mode as the motion vector MV and the evaluation function value EV.

[0384] FIG. 149 is a diagram illustrating one example of the configuration of the comparison selection unit 492 illustrated in FIG. 148. In FIG. 149, the comparison selection unit 492 includes a first comparison unit 502 which receives an evaluation function value ef and a corresponding displacement vector vf calculated according to the field prediction mode and outputs a displacement vector vfst and evaluation function value efst corresponding to an optimal representative evaluation point, a second comparison unit 504 which receives an evaluation function value eu and a corresponding displacement vector vu calculated according to the upper subtemplate block prediction mode and outputs an evaluation function value eust and a corresponding displacement vector vust corresponding to an optimal representative evaluation point, and a third comparison unit 506 which receives an evaluation function value el and a corresponding displacement vector vl calculated according to the lower subtemplate block prediction mode and outputs a displacement vector vlst and a corresponding evaluation function value elst corresponding to an optimal representative evaluation point. The configuration of the first, second, and third comparison units 502, 504, and 506 is similar to the configuration illustrated in FIG. 27, and a minimum evaluation function value is determined as the evaluation function value corresponding to the optimal representative evaluation point.

[0385] The comparison selection unit 492 further includes an adder circuit 508 for adding the optimal evaluation function values eust and elst output from the second and third comparison units 504 and 506, a comparator 510 for comparing the output of the adder circuit 508 and the optimal evaluation function value efst from the first comparison unit 502, a gate circuit 514 for transmitting the optimal evaluation function value efst and displacement vector vfst from the first comparison unit 502 when the output of the comparator 510 indicates that the optimal evaluation function value efst is smaller than the output of the adder circuit 508, a comparator 512 activated when the output of the comparator 510 indicates that the output of the adder circuit 508 is larger, for comparing the evaluation function values eust and elst from the second and third comparison units

504 and 506, and a selector 516 for allowing the evaluation function value eust and displacement vector vust from the second comparison unit 504 to pass through one of the evaluation function value elst and the displacement vector vlst from the third comparison unit 506 according to output from the comparator 512.
[0386] The comparison selection unit 492 illustrated in FIG. 149 performs the following determination operations:
if efst < eust+elst
est = efst, vst = vfst
e1se if eust < elst
est = eust, vst = vust
else est = elst, vst = vlst
The comparison selection unit 408 illustrated in FIG. 148 is also provided with a configuration similar to the comparison selection unit 492 illustrated in FIG. 149. That is, the comparison selection unit 408 illustrated in FIG. 148 is provided with a configuration wherein the comparison units 502, 504, and 506 in the input step are removed from the configuration illustrated in FIG. 149.
[0387] With the motion vector detection device in the third example, there is one full search computation unit used in the fine search computation unit, making it possible to reduce the scale of the device.
[0388] [Fourth Example of Hierarchical Search Method Motion Vector Detection Device] FIG. 150 is a diagram illustrating the configuration of a fourth example of a hierarchical search method motion vector detection device. In the motion vector detection device illustrated in FIG. 150, a comparison selection unit 522 receives the evaluation function values ef, eu, and el and corresponding displacement vectors vf, vu, and vl imparted from the rough search computation unit 402, and outputs optimal prediction mode indication data pst indicating a prediction mode imparting the displacement vector vst corresponding to the optimal representative evaluation point and the optimal representative evaluation point thereof.
[0389] The fine search computation unit 404 includes one full search computation unit 520. The full search computation unit 520 detects a motion vector by performing a full search operation having a peripheral region with the displacement vector vst corresponding to the optimal representative evaluation point given from the comparison selection unit 522 in the center as a search area. At this time, the full search computation unit 520 determines a motion vector by executing block-matching processing only according to a designated prediction mode indicated by the optimal prediction mode indication data pst imparted from the comparison selection unit 522. The configuration of the comparison selection unit 492 illustrated in FIG. 150 can be used as the configuration of the comparison selection unit 522. That is, the first comparison unit 502, second comparison unit 504, and third comparison unit 506 illustrated in FIG. 150 may have a configuration added thereto wherein identification bits indicating these are output as prediction mode indication data. These prediction mode identification bits are imparted to the gate circuit 514 and the selector 516, and are selected and output along with the corresponding evaluation function values and motion vectors. This makes it possible to easily generate optimal prediction mode indication data.
[0390] In the full search computation unit 520, the prediction mode to be followed is determined according to the optimal prediction mode indication data pst. The

configuration of the motion vector detection device illustrated in FIG. 1 can be used as is as the configuration of the full search computation unit 520. That is, a configuration should be used wherein one of the motion vectors and evaluation function values following the three created prediction modes is selected according to the optimal prediction mode indication data pst. Instead of this, it is also possible to use the configuration illustrated in FIG. 138 as the configuration of the full search computation unit 520. When using the configuration of the fine search computation unit illustrated in FIG. 138, one of the first to third full search computation units is put in an operation state according to the optimal prediction mode indication data pst. Prediction mode indication data Ps is also output from the full search computation unit 520. In this case, the optimal prediction mode indication data pst is used to generate the prediction mode indication data Ps.

[0391] Note that the diagrams of the hierarchical search method motion vector detection devices in the first to third examples do not illustrate that the prediction mode indication data Ps is output. However, they may be configured so that the prediction mode indication data Ps is output in each example.

[0392] Furthermore, a discrimination step different from the comparison selection unit illustrated in FIG. 149 may be used as the discrimination method used when the comparison selection unit 522 determines the optimal prediction mode and detects the optimal displacement vector determined according to the optimal prediction mode thereof.

[0393] Additionally, the detection of motion vectors is executed with integer accuracy in the fine search computation unit 404. A configuration may be used for detecting motion vectors with fractional accuracy such as half pel accuracy. For example, the configuration in JP H5-105432 which the group of inventors of the present application have already applied for, or the configuration illustrated in FIG. 86 may be used for the configuration for detecting motion vectors with fractional accuracy. Additionally, in the first to fourth hierarchical search method motion vector detection devices, motion vector detection is carried out with coding using field units, that is, when detecting prediction images. In this case, a configuration may be used for performing coding using frame units. In this case, a configuration can be used in the full search method motion vector detection device previously described in FIG. 1 and onward for detecting motion vectors according to a frame prediction mode, an odd number field prediction mode, and an even number field prediction mode.

[0394] Additionally, the position of representative evaluation points searched for by the rough search computation unit in the hierarchical search method motion vector detection device may not be for each eight pixels, and the representative evaluation points may be determined by thinning the evaluation points according to other conditions. Furthermore, the search range of the fine search computation unit may be thinned and to minimize the evaluation points, and the size thereof is arbitrary.

[0395] As described above, because the present invention was configured to detect motion vectors by performing hierarchical searching in a plurality of prediction modes, motion vectors can be detected by searching with high precision with a small amount of hardware or number of computations having a wide range as the regional search area, it is possible to accurately create a prediction image, and it is possible to realize high precision coding.

[0396]

[Advantageous Effects of Invention] The invention of claim 1 is configured so an evaluation means calculates in parallel evaluation-function values corresponding to each of a plurality

of prediction modes and a motion-vector determination means simultaneously determines motion vectors for each prediction mode according to outputs of this evaluation means. As such, motion vectors can be detected at high speed according to a plurality of prediction modes without increasing a hardware volume.
[0397] The invention of claim 2 is configured so an evaluation means has processors disposed corresponding to each pixel of a current-image block (template block) and these processors perform predetermined computation (seek a component value of an evaluation function) between pixels of a current-image block and pixels of a corresponding reference-image block (search-window block) to output an evaluation-function value. As such, computation in common for a plurality of prediction modes can be executed using one circuit configuration and a device scale can be reduced.
[0398] The invention of claim 3 is configured so a plurality of element processors transfers stored pixel data only along one direction. As such, a search-window block can be rewritten by one transfer operation of search-window pixel data, an evaluation-function value can be efficiently calculated, and current consumption at a time of pixel-data transfer can be reduced.
[0399] The invention of claim 4 is configured so motion vectors are detected according to an odd-field prediction mode, an even-field prediction mode, and a frame prediction mode. As such, motion vectors can be efficiently determined for prediction-image generation in frame units.
[0400] The invention of claim 5 is configured so motion vectors are detected according to an upper-half-block prediction mode (upper-sub-template-block prediction mode), a lower-half-block prediction mode (lower-sub-template-block prediction mode), and a field prediction mode. As such, motion vectors can be detected at high speed and efficiently at a time of coding in field units.
[0401] The invention of claim 6 is configured so processors storing current-image-block (template-block) pixel data are divided into four groups of a processor group corresponding to pixels belonging to an upper-half block and belonging to an even field, a processor group corresponding to pixels belonging to the upper-half block and belonging to an odd field, a processor group corresponding to pixels belonging to a lower-half block and belonging to the even field, and a processor group belonging to the lower-half block and the odd field and a group configuration of these processors groups is changed according to a mode designation signal to correspond with a plurality of prediction modes. As such, a highly versatile motion-vector detection device can be realized that can flexibly correspond to any among coding in frame units and coding in field units.
[0402] The invention of claim 7 is configured so input image data is imparted, according to a screen-format designation signal, to an evaluation means of generating an evaluation-function value after a screen format is set to a designated screen format. As such, even if image data of any screen format is imparted, motion vectors can be easily detected according to a desired plurality of prediction modes.
[0403] The invention of claim 8 is configured so an effective stored-pixel count of a buffer means for storing side-window-block pixel data disposed in a processor array is variable. As such, a size of a search area can be easily changed.
[0404] The invention of claim 9 is configured so an optimal representative evaluation point is sought by thinning evaluation points in a search area and performing block-matching processing according to a plurality of prediction modes for representative evaluation

points and next block-matching processing is performed for all evaluation points included in a peripheral region of this sought optimal representative evaluation point to detect a motion vector. As such, a motion vector for generating an optimal prediction image can be determined at high speed with a wide range as a search area.

[0405] The invention of claim 10 is configured so a search area of a first computation means that performs a full search is determined for each of a plurality of prediction modes and an optimal motion vector is detected by calculating an evaluation-function value according to each of the plurality of prediction modes in each determined search area. As such, optimal motion vectors can be determined according to a plurality of prediction modes.

[0406] The invention of claim 11 is configured so one optimal representative evaluation point is established in common for a plurality of prediction modes and a motion vector is detected by performing a full-search operation according to the plurality of prediction modes according to this optimal representative evaluation point. As such, a device scale can be further reduced and an optimal motion vector can be easily detected according to a plurality of prediction modes with a wide range as a search area.

[0407] The invention of claim 12 performs computation of evaluation-function values and detection of motion vectors for all evaluation points in a peripheral region including an optimal representative evaluation point according to a prediction mode imparting this optimal evaluation point. As such, a device scale can be further reduced and an optimal motion vector can be determined by searching a wide range.

[0408] The invention of claim 13 is configured so evaluation-function-value components are generated using processors arranged in a matrix corresponding to each pixel of a current-image block; outputs of each processor of a processor array are classified into processor groups corresponding to pixels in an even field, an odd field, and a frame; a sum of the outputs of the processors is sought for each classification; and respective motion vectors for the odd field, the even field, and the frame are determined in parallel according to this summing result. As such, motion vectors can be detected at high speed and efficiently with a small hardware volume.

[0409] The invention of claim 14 generates in parallel evaluation-function-value components using processors arranged in a matrix corresponding to each pixel of a current-image block (template block), classifies outputs of these processors into outputs of processors corresponding to pixels in a field and respective pixels in an upper-half block (upper-sub-template block) and a lower-half block (lower-sub-template block) in the current-image block, and detects motion vectors by seeking a sum of outputs of the processors for each classification. As such, respective motion vectors for a field prediction mode, an upper-half-block prediction mode (upper-sub-template-block prediction mode), and a lower-half-block prediction mode (lower-sub-template-block prediction mode) can be detected with a small hardware volume, at high speed, and efficiently.

[0410] The invention of claim 15 is configured so evaluation-function-value components of a reference-image block (search-window block) and a current-image block (template block) are generated in parallel using processors arranged in a matrix corresponding to each pixel of the current-image block; the processor matrix is divided in four into a processor group corresponding to pixels belonging to an upper-half block and an even field, a processor group corresponding to pixels belonging to the upper-half block and the even field, a processor group corresponding to pixels of a lower-half block and the even field,

and a processor group corresponding to pixels of the lower-half block and an odd field; and combinations of these processor groups are changed according to a mode designation signal to generate evaluation-function values corresponding to three prediction modes. As such, a motion-vector detection device can be obtained that can easily correspond without changing a device configuration even for different prediction modes and is highly versatile.
[0411] The invention of claim 16 is configured with a buffer means of an input unit to vary an input order and an output order of input reference-image data. As such, motion vectors can be detected by performing block-matching processing by changing to a screen format according to a desired prediction mode even if input reference-image data of various screen formats are imparted and a highly versatile motion-vector detection device can be obtained.
[0412] In the invention of claim 17, an effective stored-pixel count of a buffer means disposed for every predetermined number of processors in a processor array disposed corresponding to each pixel of a current-image block (template block) and a reference-image block (search-window block) is variable. As such, a search area of motion vectors can be easily changed.
[0413] The invention of claim 18 is configured so first, optimal representative evaluation points for each of a plurality of prediction modes are determined for representative evaluation points in a search area of a wide range; next, all evaluation points in an optimal-representative-evaluation-point peripheral region for each of the prediction modes are searched to determine motion-vector candidates (optimal displacement vectors) for each of the plurality of prediction modes; and motion vectors are determined from these determined optimal displacement vectors. As such, motion vectors can be efficiently detected according to a plurality of prediction modes and according to a hierarchical search method without increasing a hardware volume.
[0414] The invention of claim 19 is configured so optimal representative evaluation points are determined according to a plurality of prediction modes using only representative evaluation points in a search area of a wide range; next, optimal displacement vectors are calculated by performing a search operation for all evaluation points of a peripheral region centered around the optimal evaluation point for each prediction mode; and finally, motion vectors are determined from the optimal displacement vectors. As such, motion vectors can be detected at high speed according to a plurality of prediction modes without increasing a device scale.
[0415] The invention of claim 20 determines optimal representative evaluation points according to a plurality of prediction modes using only representative evaluation points in a search area of a wide range, performs block-matching processing for each prediction mode by performing a full search for all evaluation points in this optimal-representative-evaluation-point peripheral region according to the plurality of prediction modes to determine optimal displacement vectors, and determines motion vectors from these optimal displacement vectors. As such, optimal motion vectors can be detected at high speed without increasing a hardware volume.
[0416] The invention of claim 21 performs block-matching processing according to a plurality of prediction modes using only representative evaluation points in a wide range to determine an optimal representative evaluation point and an optimal prediction mode imparting this optimal representative evaluation point and performs block-matching processing according to the optimal prediction mode as a search operation for each

evaluation point included in a peripheral region including this optimal representative evaluation point. As such, optimal motion vectors can be determined at high speed according to a plurality of prediction modes without increasing a device scale. The invention of claim 22 is configured so element processors disposed corresponding to pixels of Q rows by P columns and data storage means disposed corresponding to each element-processor column, each storing R pixel data, are provided; these element processors and data storage means are disposed in a systolic array so as to transfer reference-screen image data one dimensionally along one direction; and motion vectors are detected every A·(P·(Q+R)) cycles, A being a variable constant. As such, a motion-vector search range in a horizontal direction can be changed without changing a device configuration and a motion-vector detection device that can easily correspond with a plurality of search ranges can be realized.

[0417] The invention of claim 23 is configured so A processor arrays are disposed that are provided with a configuration where element processors arranged in a matrix corresponding to pixels of P columns by Q rows and data buffers where each can store R pixel data are disposed corresponding to each column of processor arrays, pixel data of different current-screen blocks are stored in each of these A processor arrays, current-screen-block pixel data stored by the processor arrays are configured to be sequentially updated every A·(Q+R)·P cycles, and the same reference-screen pixel data is imparted to the A processor arrays. As such, a motion vector can be detected for a current-screen block in a pipeline and a motion-vector detection device that can detect motion vectors at high speed can be realized.

[0418] The invention of claim 24 is configured so element processors are divided into a plurality of sub-blocks in a processor array made of element processors disposed in a matrix corresponding to pixels of P columns by Q rows and data buffers disposed corresponding to each column of the element-processor array, each storing R pixel data; a configuration is such that pixel data of different current-screen blocks are stored for each sub-block and the same reference-screen pixel data is imparted to these sub-blocks; and the pixel data of the current-screen blocks of the sub-blocks are sequentially updated every processing unit cycle. As such, even in a situation where a size of a current-screen pixel block is small, easily changing a device scale enables a corresponding motion vector to be detected. Moreover, motion vectors of a plurality of current-screen blocks can be detected in a pipeline and a motion-vector detection device that can determine motion vectors of each current-screen block at high speed can be obtained.

[0419] The invention of claim 25 is configured so a first register for storing first-current-screen-image pixel data and a second register for storing second current-screen-image pixel data are provided in each element processor disposed in a matrix corresponding to current-screen pixels of P columns by Q rows and one register is used to store other full-screen-image pixel data to the other register at a time of evaluation-function-value generation. As such, motion-vector detection of a current-screen block can be executed in a pipeline and a motion-vector detection device that can detect motion vectors at high speed without increasing a device scale can be realized. The invention of claim 26 is configured so in a comparison unit that determines an optimal displacement vector as a motion vector according to a value of an evaluation-function value, as a priority order for displacement vectors imparting the same evaluation-function value, three standards are prepared of a priority-order determination standard that imparts a highest priority order to a search-

range central region, a priority-order determination standard imparting a highest priority order to a displacement vector of a lowermost end portion in a vertical direction in a central portion in a horizontal direction of the search range, and a priority-order determination standard imparting a highest priority order to a displacement vector in an uppermost end portion in the vertical direction in the central portion of in the horizontal direction in the search range and one of these priority-order determination standards is selected according to a usage to be used. As such, regardless of a usage to be used, a displacement vector close to a displacement vector that is directly behind can continually be determined as a motion vector and a motion-vector detection device that can detect a motion vector with higher precision can be realized.

[0420] The invention of claim 27 is configured so an integer-precision vector determination means of detecting a motion vector at integer precision for each of a plurality of prediction modes; a storage means of receiving reference-screen-image pixel data and current-screen-image pixel data imparted from this integer-precision vector determination means, the determined integer-precision motion vector storing the reference-screen-block pixel data for each of the plurality of prediction modes; and the reference-screen-block pixel data stored in this storage means and the current-screen-block pixel data imparted from the integer-precision vector determination means are received to determine motion vectors at fractional precision for each of the plurality of prediction modes. As such, it is unnecessary to newly access a reference-image memory and a current-image memory when determining a motion vector at fractional precision, motion vectors can be determined at fractional precision at high speed according to each of a plurality of prediction modes, and a motion vector with high precision can be determined at high speed. Moreover, motion-vector determination at integer precision and motion-vector determination at fractional precision can be executed in a parallel aspect and motion vectors at fractional precision and integer precision can be determined at high speed.

[0421] The invention of claim 28 is configured so an effective region of a motion-vector search range is set for each block of a predetermined size in a current-screen image and a motion-vector search is performed only in this effective region. As such, motion vectors can be determined by performing evaluation only for displacement vectors in a necessary region, a more optimal motion vector can be determined, and a high-performance image coding system can be constructed.

[Brief Description of Drawings]

[FIG. 1] A diagram schematically illustrating an overall configuration of a full-search motion-vector detection device that is a first example of this invention.

[FIG. 2] A diagram illustrating sizes of a template block and a search area used in the motion-vector detection device illustrated in FIG. 1.

[FIG. 3] A diagram schematically illustrating a configuration of an element processor included in a processor array of the motion-vector detection device illustrated in FIG. 1.

[FIG. 4] A diagram illustrating a configuration of the processor array illustrated in FIG. 1.

[FIG. 5] (A) illustrates a division configuration of a screen and a configuration of a macro block serving as a block-matching-processing unit, and (B) is a diagram illustrating one example of a configuration of the template block.

[FIG. 6] (A) illustrates a configuration of an even-field-sub-template block, and (B) is a diagram illustrating a configuration of an odd-field-sub-template block.

[FIG. 7] A diagram illustrating a division configuration of a reference image screen and a presence region of macro-block pixel data stored in the processor array.
[FIG. 8] A diagram illustrating an example of a storage state of reference-image pixel data stored in the processor array illustrated in FIG. 1.
[FIG. 9] A diagram illustrating an example of a storage state of reference-block-image pixel data and current-image-block pixel data stored in a linear processor array in the processor array illustrated in FIG. 1.
[FIG. 10] A diagram illustrating specific examples of a motion-vector search range and a current-image block.
[FIG. 11] A diagram illustrating specific examples of the current-image block and the search area when using the odd-field-sub-template block or the even-field-sub-template block.
[FIG. 12] A diagram illustrating storage conditions of the reference-image-block pixel data and template-block pixel data stored in the processor array illustrated in FIG. 1.
[FIG. 13] (A) illustrates a transfer state in the processor array of search-window pixel data when the search window moves one column in a horizontal direction, (B) illustrates as an example moving conditions of the search-window pixel data in a processor array, and (C) is a diagram illustrating a distribution state in the search area of the search-window pixel data stored in the processor array after search-window-pixel-data input.
[FIG. 14] (A) is a diagram illustrating as an example a displacement vector for an odd field of the even-field-sub-template block, and (B) is a diagram illustrating as an example a displacement vector of an even field of the odd-field-sub-template block.
[FIG. 15] A diagram illustrating the stored data in the processor array during an evaluation-function-value calculation operation and a position in the search area.
[FIG. 16] (A) illustrates a state where a search-window block reaches a lowermost portion of the search window, and (B) is a diagram illustrating a position in the search area of the search-window pixel data stored in the processor array in the state illustrated in (A).
[FIG. 17] (A) is a diagram illustrating a search-window-pixel-data storage state in the processor array after evaluation-function-value calculation for one search window, and (B) is a diagram illustrating the search-window-pixel-data storage state in the processor array after a sixteen-pixel shift-in operation.
[FIG. 18] A diagram illustrating a positional relationship between the template block and the search-window block in the state illustrated in (B) in FIG. 17.
[FIG. 19] A diagram illustrating a state when the search-window block reaches a final position in the search area.
[FIG. 20] A diagram illustrating one example of the configuration of the element processor included in the processor array illustrated in FIG. 1.
[FIG. 21] A timing chart illustrating operations of the element processor illustrated in FIG. 20.
[FIG. 22] A diagram illustrating a specific configuration of the summing unit illustrated in FIG. 1.
[FIG. 23] A diagram illustrating another configuration of the summing unit illustrated in FIG. 1.
[FIG. 24] A diagram illustrating yet another configuration of the summing unit illustrated in FIG. 1.
[FIG. 25] A timing chart illustrating operations of the summing unit illustrated in FIG. 24.

[FIG. 26] A diagram illustrating as an example operations of the summing unit illustrated in FIG. 24.
[FIG. 27] A diagram illustrating a configuration of the comparison unit illustrated in FIG. 1.
[FIG. 28] A diagram illustrating a configuration of a second example of a full-search motion-vector detection device of this invention.
[FIG. 29] A diagram illustrating a configuration of a template block and a search-window block used in the motion-vector detection device illustrated in FIG. 28.
[FIG. 30] A diagram illustrating a configuration of the summing unit illustrated in FIG. 28.
[FIG. 31] A diagram illustrating as an example a division configuration of a template block used in a full-search motion-vector detection device that is a third example of this invention.
[FIG. 32] A diagram illustrating a configuration of the third example of the full-search motion-vector detection device of this invention.
[FIG. 33] A diagram schematically illustrating a configuration of the summing unit illustrated in FIG. 32.
[FIG. 34] A diagram for describing operations of a fourth example of this invention.
[FIG. 35] A diagram for describing a horizontal-direction search-range expansion operation in the fourth example of this invention.
[FIG. 36] A diagram illustrating a specific example of the horizontal-direction search-range expansion in the fourth example of this invention.
[FIG. 37] A diagram illustrating a specific operation sequence of the fourth example of this invention.
[FIG. 38] A diagram schematically illustrating an overall configuration of a motion-vector detection device of the fourth example of this invention.
[FIG. 39] A diagram illustrating one example of a configuration of the comparison unit illustrated in FIG. 38.
[FIG. 40] A diagram for describing operations of the comparison unit illustrated in FIG. 39.
[FIG. 41] A diagram for describing the operations of the comparison unit illustrated in FIG. 39.
[FIG. 42] A diagram illustrating one example of a configuration of the comparison control circuit illustrated in FIG. 38.
[FIG. 43] A diagram schematically illustrating a configuration of the read control circuit illustrated in FIG. 38.
[FIG. 44] A diagram schematically illustrating an overall configuration of a motion-vector detection device of a fifth example of this invention.
[FIG. 45] A diagram for describing operations of the motion-vector detection device of the fifth example of this invention.
[FIG. 46] A diagram illustrating a specific operation sequence of the motion-vector detection device of the fifth example of this invention.
[FIG. 47] A diagram illustrating a specific sequence of a motion-vector detection operation in a situation where a processing time is made to be four processing unit times in the fifth example of this invention.
[FIG. 48] A diagram illustrating a configuration of a processor array of a motion-vector detection device in a sixth example of this invention.
[FIG. 49] A diagram schematically illustrating a configuration of the processor array when the processor array illustrated in FIG. 48 is redisposed according to a template-block size.

[FIG. 50] A diagram illustrating a unit search range in the processor array illustrated in FIG. 49.
[FIG. 51] A diagram for describing operations of the motion-vector detection device illustrated in FIG. 49.
[FIG. 52] A diagram illustrating a configuration of a motion-vector search range used in the sixth example of this invention.
[FIG. 53] A diagram illustrating a transfer aspect of reference-screen pixel data in the processor array in a processing unit time in the sixth example of this invention.
[FIG. 54] A diagram illustrating a specific sequence of the motion-vector detection operation of the motion-vector detection device illustrated in FIG. 49.
[FIG. 55] A diagram illustrating a disposition of a processor-array unit in the motion-vector detection device in the sixth example of this invention.
[FIG. 56] A diagram schematically illustrating a configuration of a portion associated with template-block pixel data of the selector illustrated in FIG. 55.
[FIG. 57] A diagram schematically illustrating a configuration of a portion associated with search-window pixel data of the selector illustrated in FIG. 55.
[FIG. 58] A diagram schematically illustrating configurations of a computation unit and a comparison unit of the motion-vector detection device in the sixth example of this invention.
[FIG. 59] A diagram illustrating a configuration of a sub-sampling image used in the sixth example of this invention.
[FIG. 60] A diagram schematically illustrating a configuration of an element processor of a seventh example of this invention.
[FIG. 61] A diagram for describing operations of the element processor illustrated in FIG. 60.
[FIG. 62] A diagram illustrating a schematic configuration of the motion-vector detection device of the seventh example of this invention.
[FIG. 63] A diagram illustrating a first variation of the element processor of the seventh example of this invention.
[FIG. 64] A diagram illustrating a second variation of the element processor of the seventh example of this invention and an operation waveform thereof.
[FIG. 65] A diagram schematically illustrating a configuration of a comparison unit in an eighth example of this invention.
[FIG. 66] A diagram illustrating first priority-order determination standards of the priority-order determination circuit illustrated in FIG. 65.
[FIG. 67] A diagram illustrating the first priority-order determination standards of the priority-order determination circuit illustrated in FIG. 65.
[FIG. 68] A diagram illustrating second determination standards of the priority-order determination circuit illustrated in FIG. 65.
[FIG. 69] A diagram illustrating the second determination standards of the priority-order determination circuit illustrated in FIG. 65.
[FIG. 70] A diagram illustrating third priority-order determination standards of the priority-order determination circuit illustrated in FIG. 65.
[FIG. 71] A diagram illustrating the third priority-order determination standards of the priority-order determination circuit illustrated in FIG. 65.

[FIG. 72] A diagram illustrating a configuration of a motion-vector detection device when using the first priority-order determination standards.
[FIG. 73] A diagram illustrating configurations of the motion-vector detection device when using the second and third priority-order determination standards and the corresponding priority-order determination standards.
[FIG. 74] A diagram schematically illustrating one example of a configuration of the priority-order determination circuit illustrated in FIG. 65.
[FIG. 75] A diagram for describing operations of the priority-order determination circuit illustrated in FIG. 74.
[FIG. 76] A diagram schematically illustrating the selector illustrated in FIG. 74 and a configuration for generating a multiplexer switching signal.
[FIG. 77] A diagram schematically illustrating a configuration of a motion-vector detection device of a ninth example of this invention.
[FIG. 78] A diagram for describing operations of a fractional-precision computation unit of the motion-vector detection device illustrated in FIG. 77.
[FIG. 79] A diagram schematically illustrating a configuration of a prediction image generated at a time of fractional-precision motion-vector detection.
[FIG. 80] A diagram illustrating one example of a configuration of the prediction-image memory and the template-block memory illustrated in FIG. 77.
[FIG. 81] A diagram for describing operations of the prediction-image memory illustrated in FIG. 80.
[FIG. 82] A signal waveform diagram illustrating the operations of the prediction-image memory illustrated in FIG. 80.
[FIG. 83] A diagram illustrating generation aspects of clock signals imparted to a memory unit respectively used in a block-size prediction mode, an odd-field prediction mode, and an even-field prediction mode in the prediction-image memory illustrated in FIG. 80.
[FIG. 84] A diagram schematically illustrating a configuration of the half-pixel-precision computation unit illustrated in FIG. 77.
[FIG. 85] A diagram for describing operations of the fractional-precision prediction-pixel generation circuit illustrated in FIG. 84.
[FIG. 86] A diagram illustrating one example of a specific configuration of the half-pixel-precision computation unit illustrated in FIG. 84.
[FIG. 87] A diagram illustrating one example of a sequence of operations of the ninth example of this invention.
[FIG. 88] A diagram illustrating a position of a template block on a current screen.
[FIG. 89] A diagram for describing operational principles of a tenth example of this invention.
[FIG. 90] A diagram illustrating a configuration of a unit search range used in the tenth example of this invention.
[FIG. 91] A diagram for describing effects of an upper-half prohibition signal and a lower-half prohibition signal used in the tenth example of this invention.
[FIG. 92] A diagram for describing effects of a left-half prohibition signal in the tenth example of this invention.
[FIG. 93] A diagram for describing effects of a right-half prohibition signal in the tenth example of this invention.

[FIG. 94] A diagram for describing search-range restricting operations in a template block of a current-screen upper-left end.
[FIG. 95] A diagram for describing the search-range restricting operations in the template block of the current-screen upper-left end.
[FIG. 96] A flowchart for describing operations at a time of motion-vector detection at fractional precision in the template block of the current-screen upper-left end.
[FIG. 97] A diagram illustrating a generation aspect of a prohibition signal in a current-screen upper-end template block.
[FIG. 98] A diagram for describing a prohibition-signal generation aspect in a template block of a current-screen upper-right end and operations of motion-vector search-range restriction.
[FIG. 99] A flowchart illustrating motion-vector detection operations for the template block of the current-screen upper-right end.
[FIG. 100] A diagram illustrating a prohibition-signal generation aspect in a current-screen-left-end-template block.
[FIG. 101] A diagram illustrating a prohibition-signal generation aspect in a template block in a current-screen central portion.
[FIG. 102] A diagram illustrating a prohibition-signal generation aspect for a template block in a current-screen right end.
[FIG. 103] A diagram for describing a prohibition-signal generation aspect and a search-range restriction aspect for a template block of a current-screen lower-left end.
[FIG. 104] A flowchart describing a time of motion-vector detection operations at fractional precision of the template block of the current-screen lower-left end.
[FIG. 105] A diagram illustrating a prohibition-signal generation aspect for a template block of a current-screen lower end.
[FIG. 106] A diagram illustrating a prohibition-signal generation aspect and a motion-vector search range restriction aspect for a template block of a current-screen lower-right end.
[FIG. 107] A diagram schematically illustrating an overall configuration of a motion-vector detection device of the tenth example of this invention.
[FIG. 108] A diagram illustrating one example of the search operation control and a configuration of a comparison unit included in the integer-precision motion-vector detection unit illustrated in FIG. 107.
[FIG. 109] A diagram schematically illustrating a configuration of a search-range control unit for fractional precision of the search-operation control unit illustrated in FIG. 107.
[FIG. 110] A diagram schematically illustrating configurations of a computation unit and a comparison unit included in the fractional-precision motion-vector detection unit illustrated in FIG. 107 and prohibition signals imparted thereto.
[FIG. 111] A diagram schematically illustrating configurations of the search-range setting unit and the read control circuit illustrated in FIG. 107.
[FIG. 112] A diagram for describing effects of the template-block designation circuit and the prohibition-range setting circuit illustrated in FIG. 111.
[FIG. 113] A diagram schematically illustrating a configuration of a reference-image-pixel-data input unit of the motion-vector detection device according to this invention.
[FIG. 114] A diagram schematically illustrating a configuration of the screen-format conversion circuit illustrated in FIG. 113.

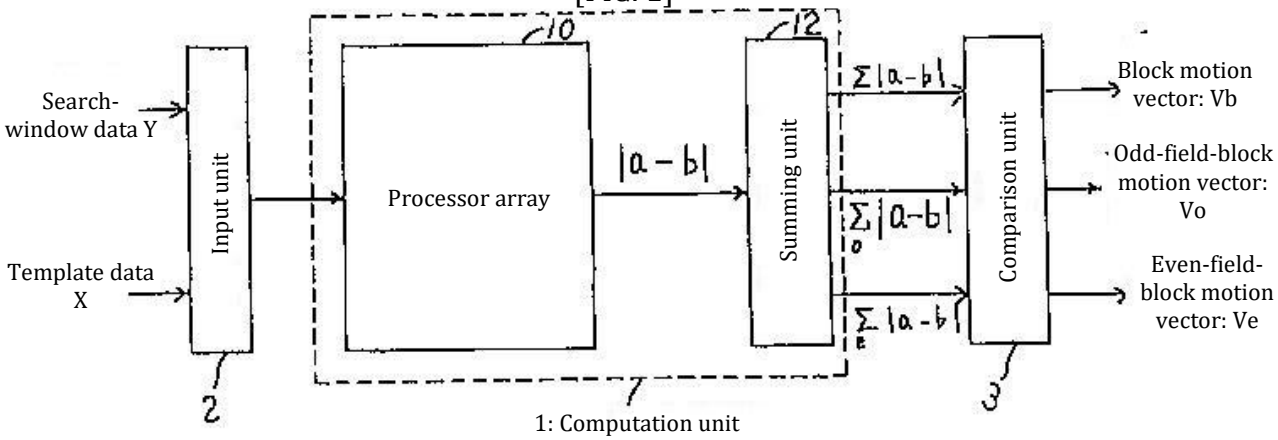
[FIG. 115] A diagram illustrating a specific configuration of the frame/frame address generator illustrated in FIG. 114.
[FIG. 116] (A) illustrates an address generation order when writing data of the address generator illustrated in FIG. 115, and (B) is a diagram illustrating an address generation sequence of the address generator illustrated in FIG. 115 when reading pixel data.
[FIG. 117] A diagram illustrating a variation of the address generator illustrated in FIG. 115.
[FIG. 118] A diagram illustrating one example of a configuration of the field/frame address generator illustrated in FIG. 115.
[FIG. 119] A timing chart illustrating operations of the address generator illustrated in FIG. 118.
[FIG. 120] A diagram illustrating address generation sequences of the address generator illustrated in FIG. 118, (A) illustrating the address generation sequence when writing pixel data of a first field and (B) illustrating the address generation sequence when writing pixel data of a second field.
[FIG. 121] A diagram illustrating another configuration of the screen-format conversion circuit illustrated in FIG. 113.
[FIG. 122] A diagram illustrating address generation sequences of the field/field address generator illustrated in FIG. 121, (A) illustrating the address generation sequence when writing field-image pixel data and (B) being a diagram illustrating the address generation sequence when reading the field-image pixel data.
[FIG. 123] A diagram illustrating address generation sequences of the frame/field address generator illustrated in FIG. 121, (A) illustrating the address generation sequence when writing frame-image pixel data and (B) being a diagram illustrating the address generation sequence when reading the pixel data.
[FIG. 124] A diagram illustrating one example of a configuration of the frame/field address generator illustrated in FIG. 121.
[FIG. 125] A diagram illustrating yet another configuration of the screen-format conversion circuit and the search-window buffer memory illustrated in FIG. 113.
[FIG. 126] A diagram illustrating one example of a configuration of the frame/field address generator illustrated in FIG. 125.
[FIG. 127] A timing chart illustrating operations of the address generator illustrated in FIG. 126.
[FIG. 128] A diagram illustrating as an example a method of generating a field image from a frame image by the address generator illustrated in FIG. 127.
[FIG. 129] A diagram illustrating a variation of the data buffer illustrated in FIG. 119.
[FIG. 130] A diagram illustrating a variation of the data buffer illustrated in FIG. 129.
[FIG. 131] A diagram illustrating yet another configuration of the data buffer illustrated in FIG. 4.
[FIG. 132] A diagram for describing operations of the data buffer illustrated in FIG. 131.
[FIG. 133] A diagram for specifically describing operations of the data buffer illustrated in FIG. 131.
[FIG. 134] A diagram illustrating one example of configurations of the template block and a search area used in hierarchical-search motion-vector detection operations.
[FIG. 135] A diagram for describing operations of a first stage of hierarchical-search motion-vector detection.

[FIG. 136] A diagram illustrating as an example a distribution of representative evaluation points after the first-stage operations are completed in the hierarchical-search motion-vector detection operations.
[FIG. 137] A diagram for describing operations of a second stage in hierarchical-search motion-vector detection.
[FIG. 138] A diagram schematically illustrating a configuration of a first example of a hierarchical-search motion-vector detection device.
[FIG. 139] A diagram schematically illustrating a configuration of the rough-search computation unit illustrated in FIG. 138.
[FIG. 140] A diagram schematically illustrating configurations of the element-processor array and the memory-cell array illustrated in FIG. 139.
[FIG. 141] A diagram schematically illustrating a configuration of the most-downstream element processor illustrated in FIG. 140.
[FIG. 142] A diagram illustrating as an example data-transfer operations from the memory-cell array to the element-processor array illustrated in FIG. 140.
[FIG. 143] A diagram schematically illustrating a configuration of a reference-image pixel-data transfer control system to the fine-search computation unit illustrated in FIG. 138.
[FIG. 144] A diagram schematically illustrating a configuration of a second example of the hierarchical-search motion-vector detection device of this invention.
[FIG. 145] A diagram schematically illustrating a configuration of the full-search computation unit illustrated in FIG. 144.
[FIG. 146] A diagram schematically illustrating a configuration of the second full-search computation unit illustrated in FIG. 144.
[FIG. 147] A diagram schematically illustrating a configuration of the third full-search computation unit illustrated in FIG. 145.
[FIG. 148] A diagram schematically illustrating a configuration of a third example of a full-search motion-vector detection device.
[FIG. 149] A diagram schematically illustrating a configuration of the comparison selection unit that seeks an optimal representative evaluation point illustrated in FIG. 148.
[FIG. 150] A diagram illustrating a configuration of a fourth example of the hierarchical-search motion-vector detection device according to this invention.
[FIG. 151] A diagram illustrating an overall configuration of a conventional image-signal coding circuit.
[FIG. 152] A diagram illustrating a configuration of the source coding circuit illustrated in FIG. 151.
[FIG. 153] A diagram for describing operations of motion compensation of an image.
[FIG. 154] A diagram illustrating a relationship between a search area, a disposition example of a template block, and a motion vector when performing motion compensation according to a block-matching method.
[FIG. 155] A diagram schematically illustrating an overall configuration of a conventional motion-vector detection device.
[FIG. 156] A diagram illustrating a configuration of an element processor included in the processor array illustrated in FIG. 155.
[FIG. 157] A diagram illustrating a scan of the template block and a scanning method of a search window in the motion-vector detection device illustrated in FIG. 155.

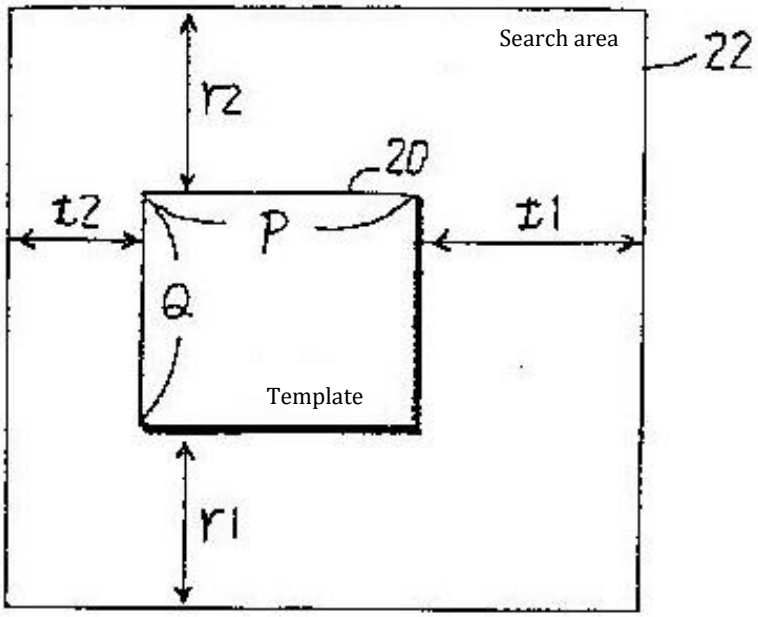
[FIG. 158] A diagram illustrating as an example operations of the motion-vector detection device illustrated in FIG. 155.
[FIG. 159] A diagram for describing operations of the motion-vector detection device illustrated in FIG. 155.
[Reference Signs List]
1, 1a, 1b computation unit; 2 input unit; 3, 3a, 3b comparison unit; 10 processor array; 12 summing unit; 20 template block; 22 search area; PE element processor; 25-1 to 25-M data register; 26-1 to 26-N data register; LA1 to LAP linear processor array; 27 difference-absolute-value circuit; 12a, 12b, 12c summing circuit; 12d adding circuit; 12ba, 12ca, 12e, 12f summing circuit; 132 comparator; 138 counter; 12g, 12h summing circuit; 122 classification changing unit; 120 summing computation unit; 120a, 120b, 120c, 120d summing circuit; 120e, 120f, 120g adding circuit; 122a, 122b selector; 205 search-window buffer memory; 207 screen-format conversion circuit; 211 selector; 213 frame/frame address generator; 215 field/frame address generator; 200 motion-vector detection device; 252 frame/field address generator; 254 field/field address generator; 272 field/field address generator; 274 frame/field address generator; 278 write prohibition circuit; 289 read-address generator; 282 write control circuit; 284 read control circuit; DL data buffer; RG1, RG2 register; RGa to RGn register; 312 register file; 314 read-address generator; 316 external address generator; 319 initial-address setting circuit; 402 rough-search computation unit; 404 comparison selection unit; 406 fine-search computation unit; 408 comparison selection unit; 410 first full-search computation unit; 412 second full-search computation unit; 414 third full-search computation unit; 412 memory-cell array; 419 element-processor array; 420 summing unit; 422 comparison unit; 425 template-block-data memory; 460 first full-search computation unit; 462 second full-search computation unit; 464 third full-search computation unit; 492 comparison selection unit; 494 full-search computation unit; 520 full-search computation unit; 522 comparison selection unit; 600 motion-vector detection device; 602 reference-image memory; 604 current-image memory; 606 read control circuit; 608 comparison control circuit; 607 search-range setting unit; 610 register; 611 comparator; 612 register; 613 counter; 630 input unit; 632 selector; PG0U to PG15U, PG0L to PG15L processor group; PBA, PBB, PBC, PBD processor block; SLa, SLb, SLc, SLd, SLe, SLf, SLg selector; DLA, DLB data-buffer group; 644 connection control circuit; SWau, SWXa, SWXb, SWXc, SWal, SWbl, SWbb switching element; STa, STb, STd, STc, STe switching element; 650a to 650d summing unit; 652a to 652d selector; 654a to 654d comparison unit; 655 adding unit; 656 comparison unit; 660a register (A); 660b register (B); 661 multiplexer; 662, 663 selector; 664 register; 665 subtractor; 666 absolute-value circuit; 672 computation unit; 673 comparison unit; 674 control device; 675 reference-image memory; 676 current-image memory; 679 multiplexer; 677a, 677b latch; 680 register; 682 comparator; 684 counter; 686 register; 688 update control circuit; 690 priority-order determination circuit; CH1, CH2, CH3 motion-vector-detection-device chip; 701 selector; 702 H absolute-value circuit; 703 V absolute-value circuit; 704 V extraction circuit; 705 -V extraction circuit; 706 selector; 707 adder; 708 selector; 709 adder; 710 multiplexer; 711, 712 register; 713 comparator; 715 timing-signal generation circuit; 716 active control circuit; 722 prediction-image memory; 724 template-block memory; 726, 738 half-pixel-precision computation unit; 730, 732 comparison unit; 734 selector; 802 fractional-precision prediction-image generation circuit; 820 reference-image memory; 821 current-image memory; 821 current-image

memory; 830 integer-precision motion-vector detection unit; 832 fractional-precision motion-vector detection unit; 834 search-range setting unit; 836 search operation control unit; 838 read control circuit.

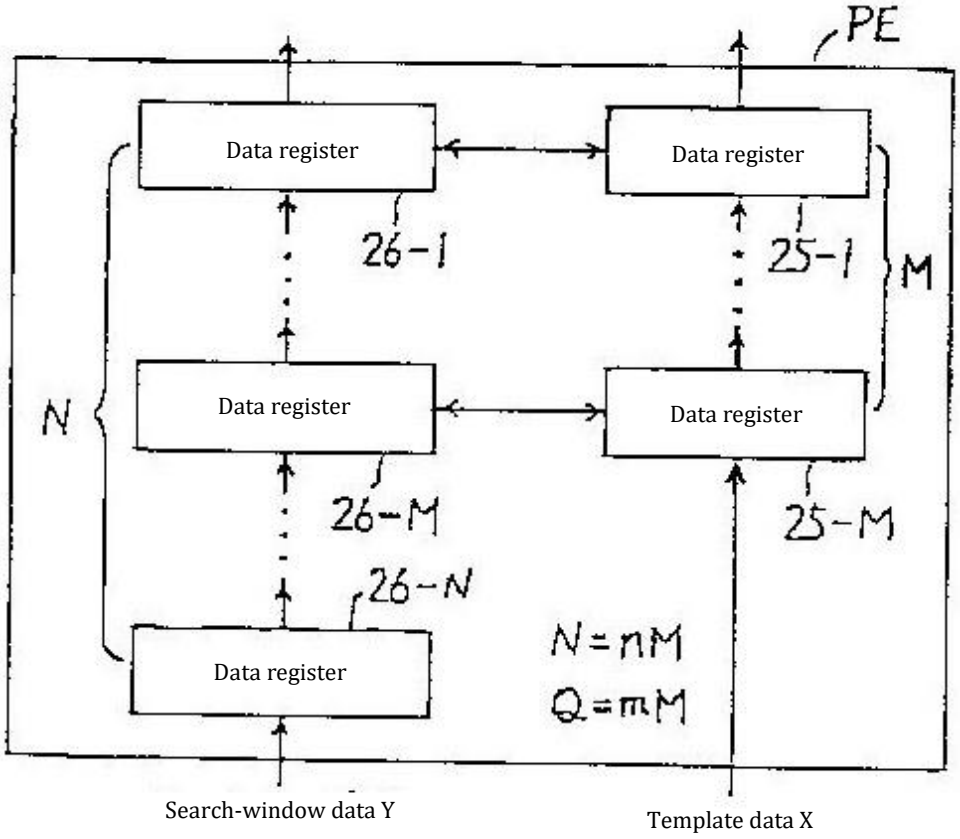
[FIG. 1]


Search-window data Y
Template data X
Input unit
Processor array
|a − b|
Summing unit
Comparison unit
Block motion vector: Vb
Odd-field-block motion vector: Vo
Even-field-block motion vector: Ve
10
12
2
3
1: Computation unit

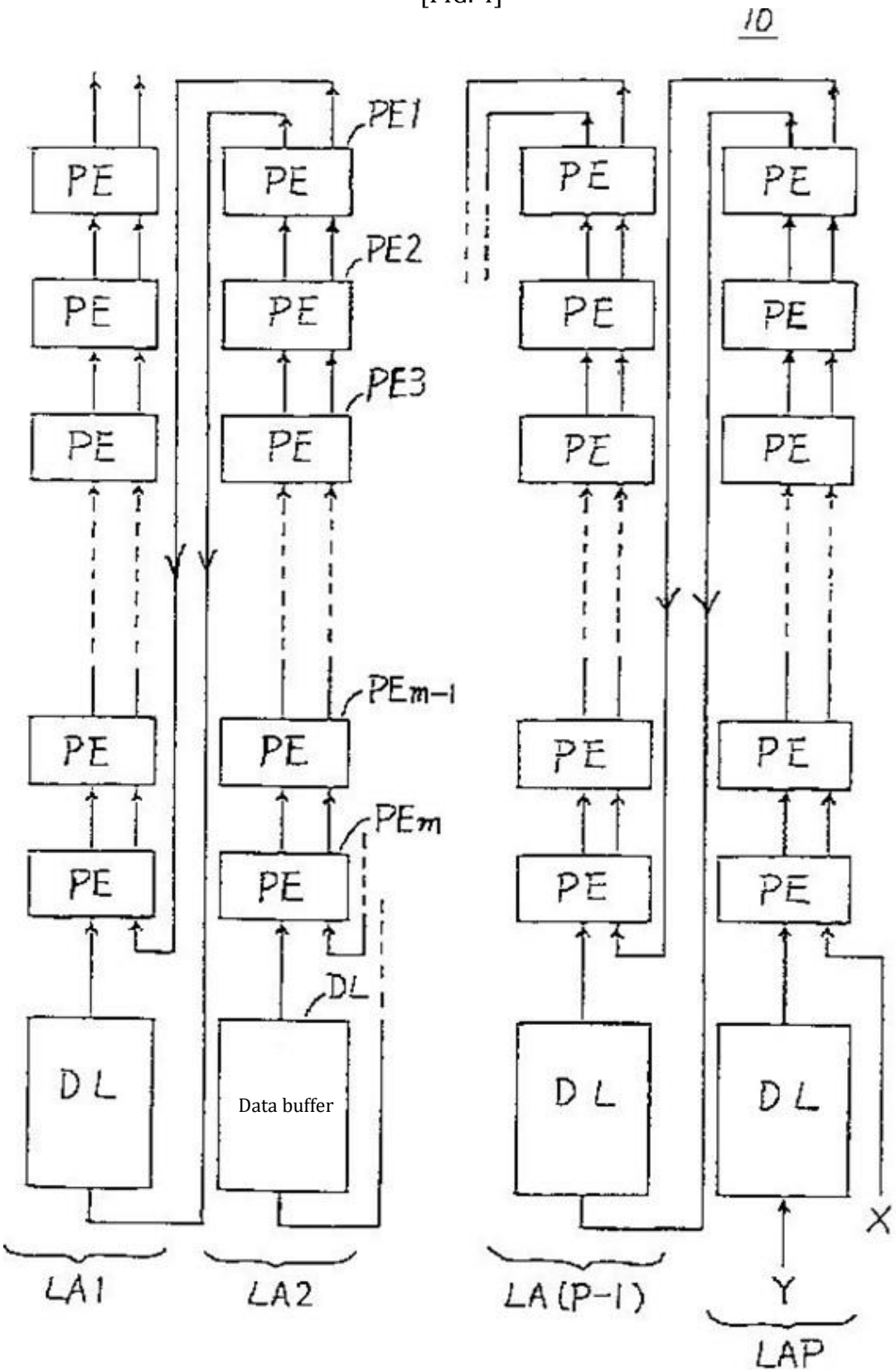
[FIG. 2]


Search area
22
r2
20
t2
P
t1
Q
Template
r1

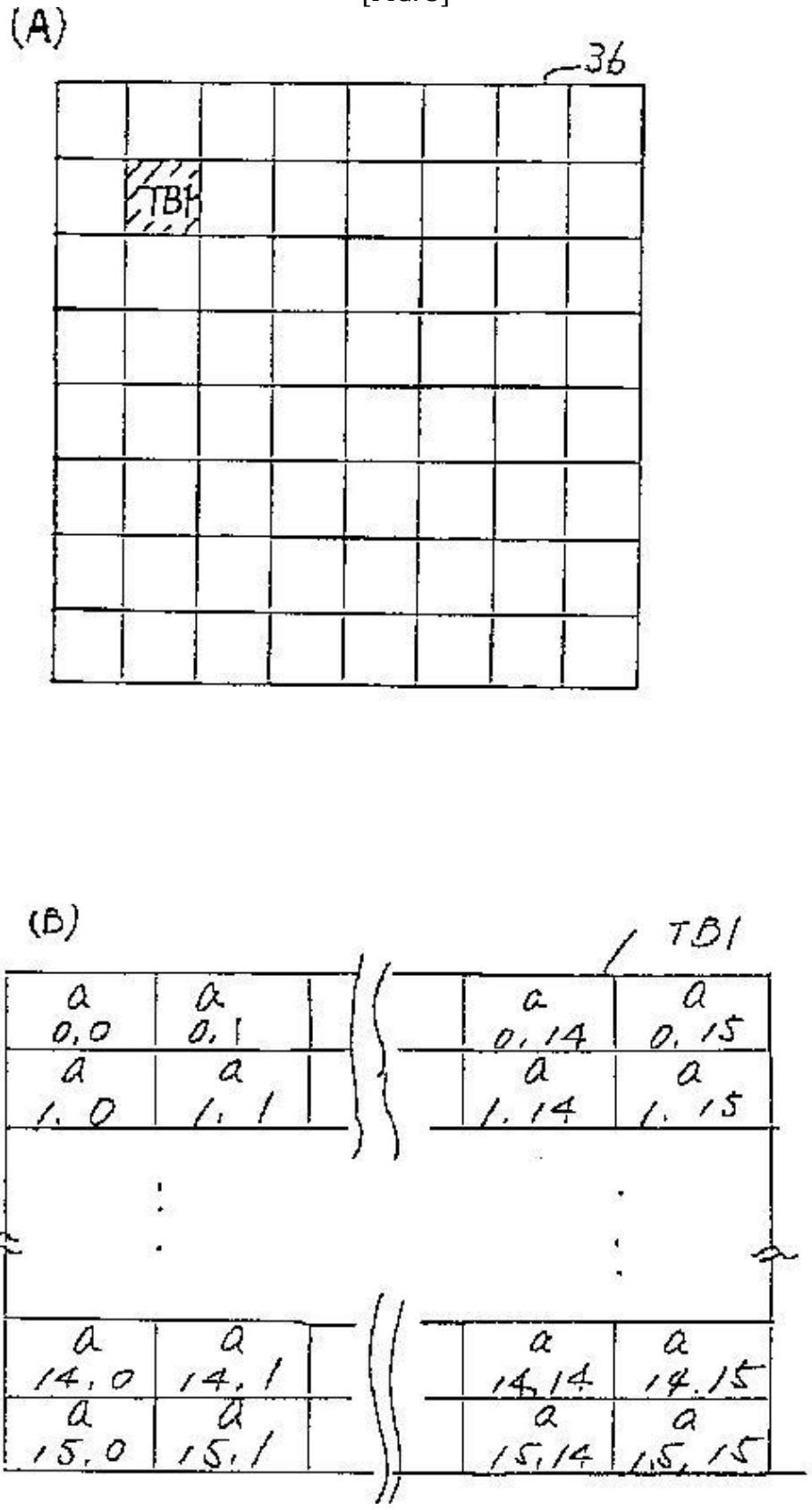
[FIG. 3]


PE
Data register
Data register
26-1
25-1
M
N
Data register
Data register
26-M
25-M
26-N
Data register
N=nM
Q=mM
Search-window data Y
Template data X

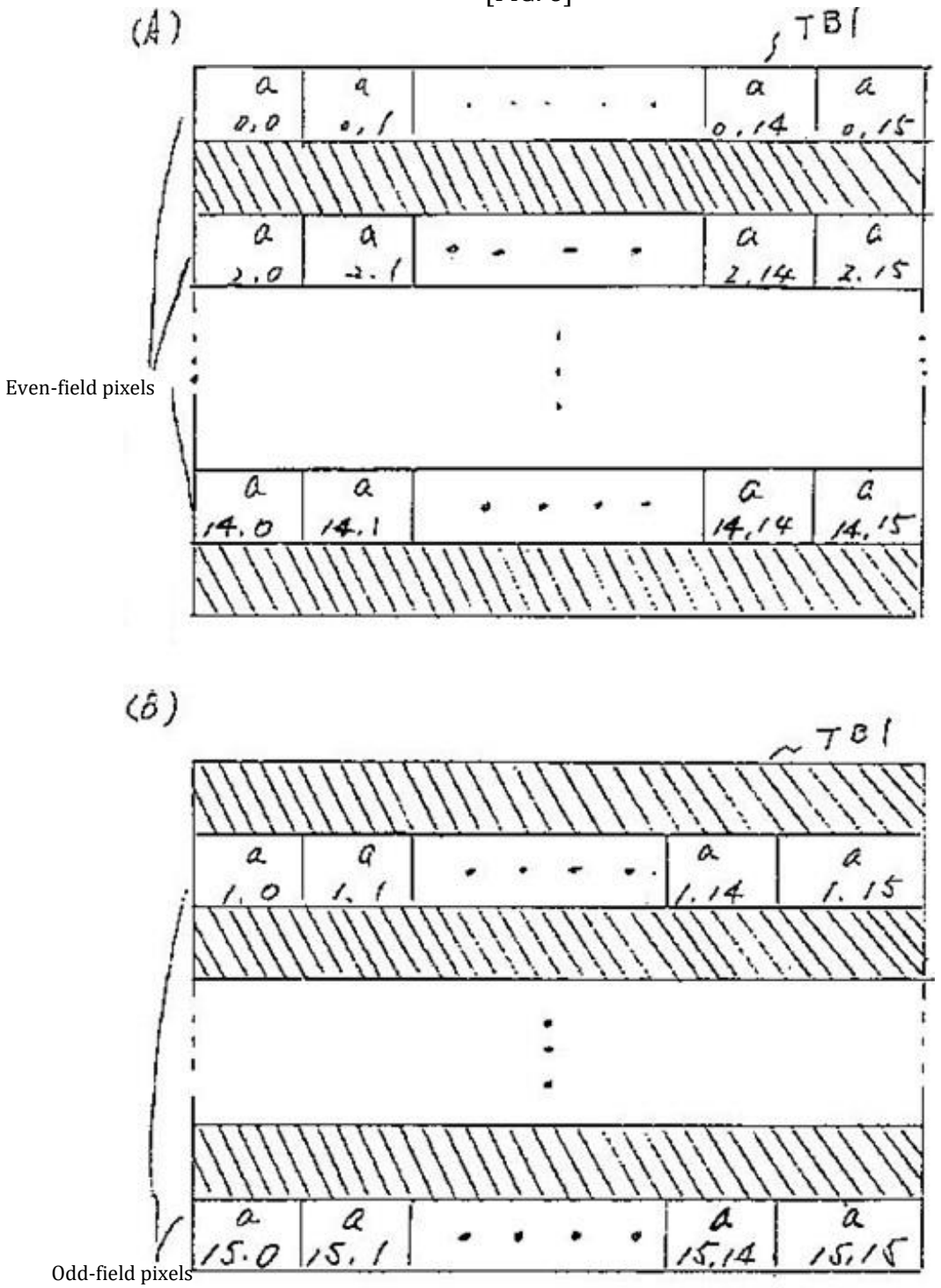
[FIG. 4]



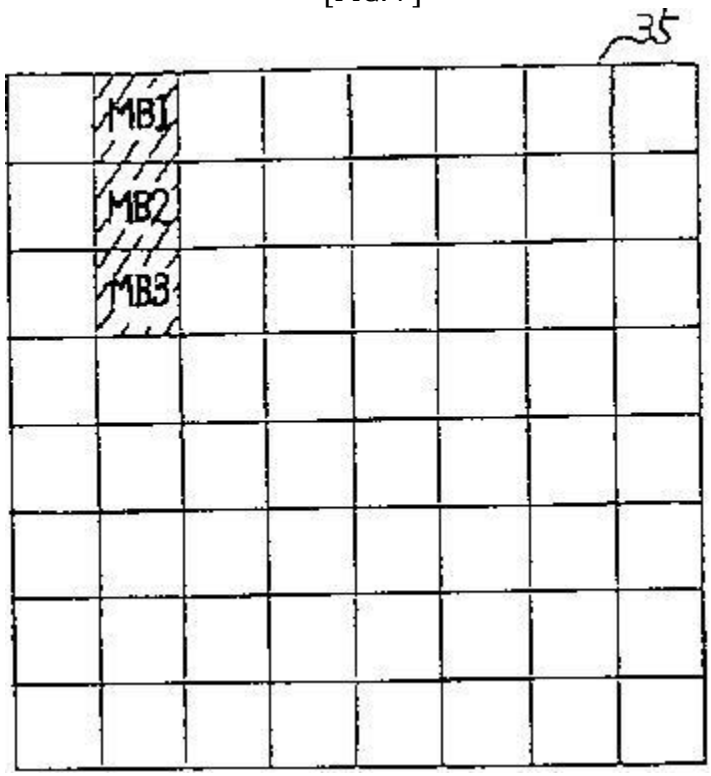
[FIG. 5]


(A)
36
TBI
(B)
TBI
a 0,0
a 0,1
a 0,14
a 0,15
a 1,0
a 1,1
a 1,14
a 1,15
a 14,0
a 14,1
a 14,14
a 14,15
a 15,0
a 15,1
a 15,14
a 15,15

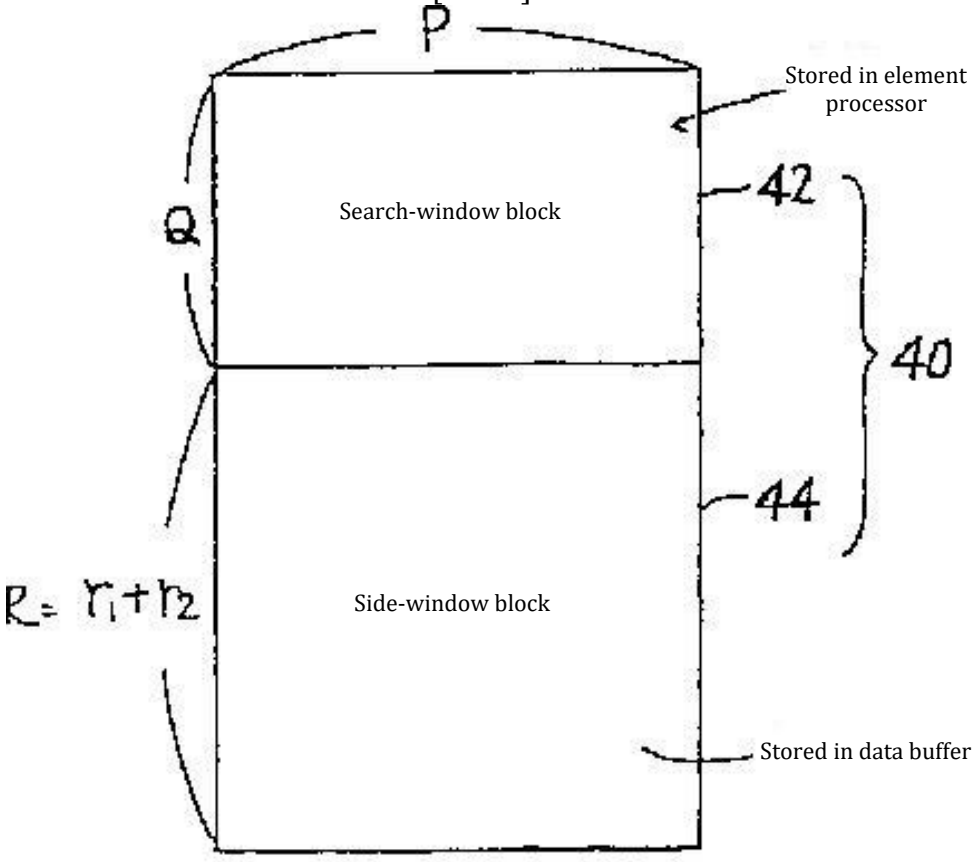
[FIG. 6]


(A)
TB1
a 0,0
a 0,1
a 0,14
a 0,15
a 2,0
a 2,1
a 2,14
a 2,15
Even-field pixels
a 14,0
a 14,1
a 14,14
a 14,15
(B)
TB1
a 1,0
a 1,1
a 1,14
a 1,15
a 15,0
a 15,1
a 15,14
a 15,15
Odd-field pixels

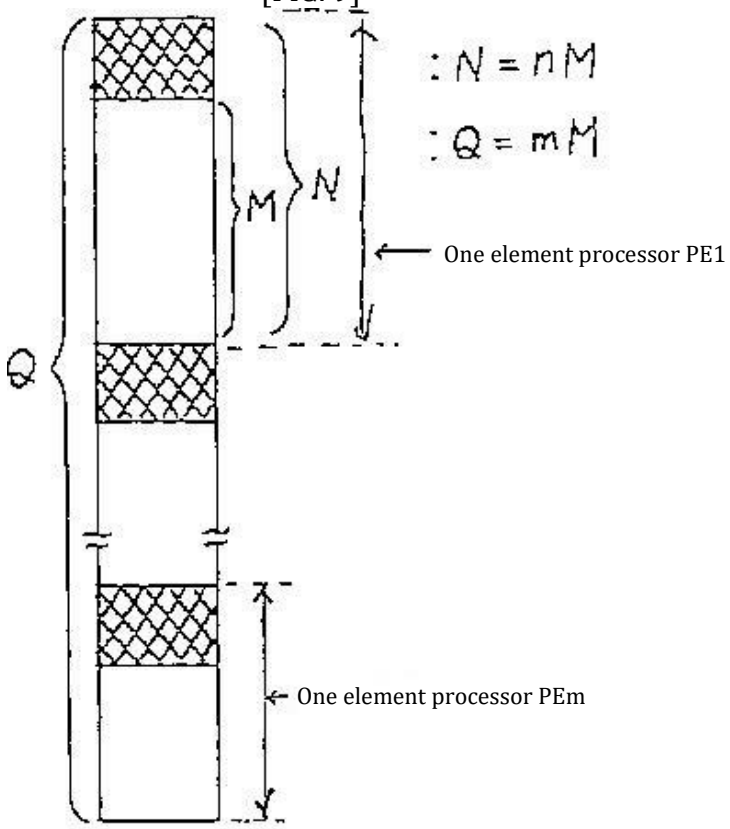
[FIG. 7]


35
MB1
MB2
MB3

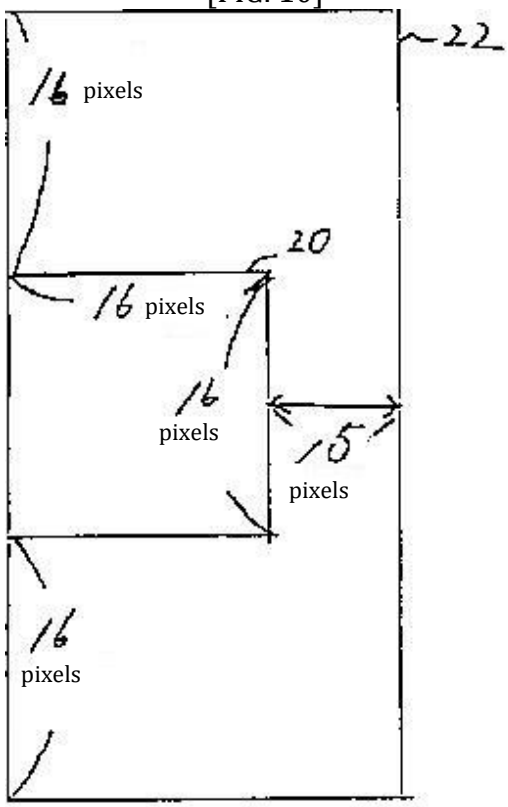
[FIG. 8]


P
Stored in element processor
Q
Search-window block
42
40
44
R= r1+r2
Side-window block
Stored in data buffer

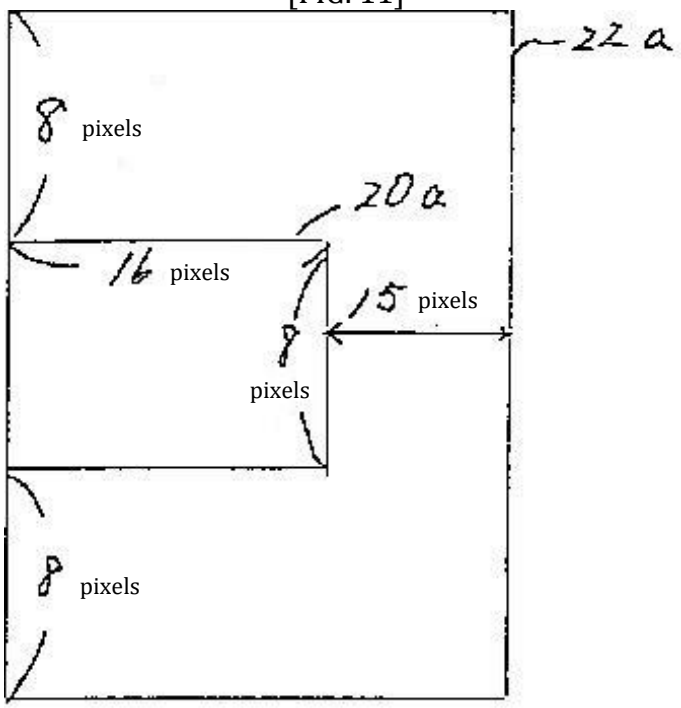
[FIG. 9]


: N = nM
: Q = mM
M
N
Q
One element processor PE1
One element processor PEm

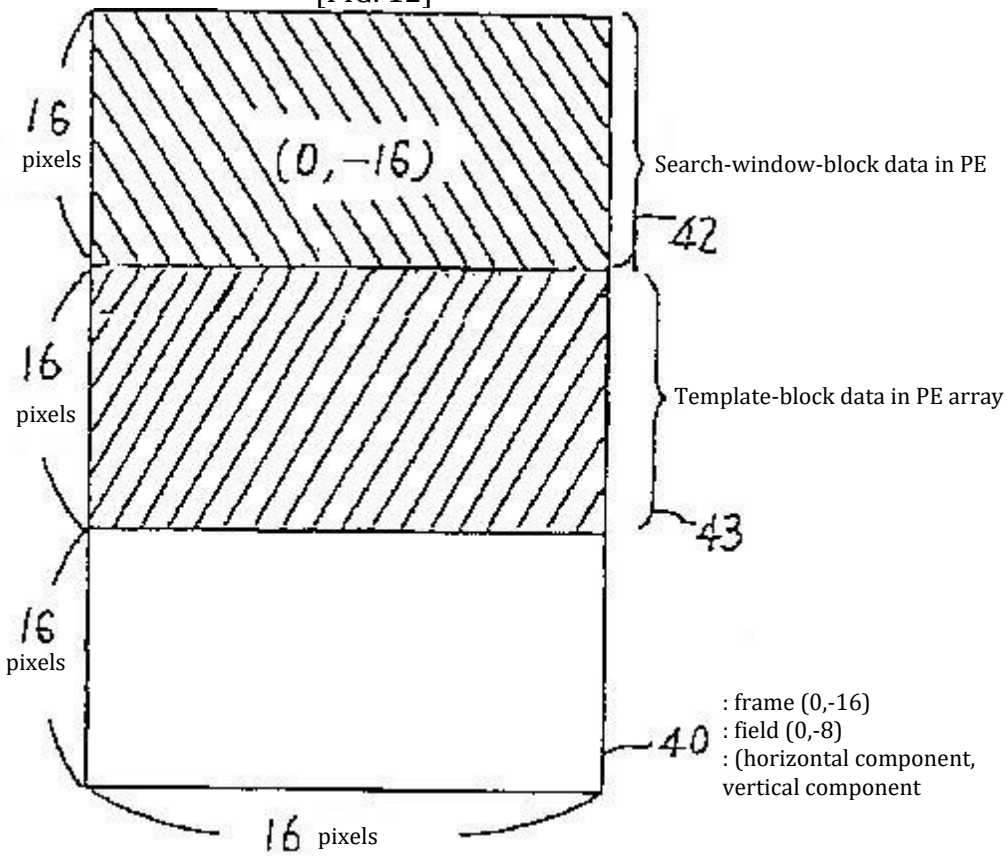
[FIG. 10]


22
16 pixels
20
16 pixels
16
pixels
15
pixels
16
pixels

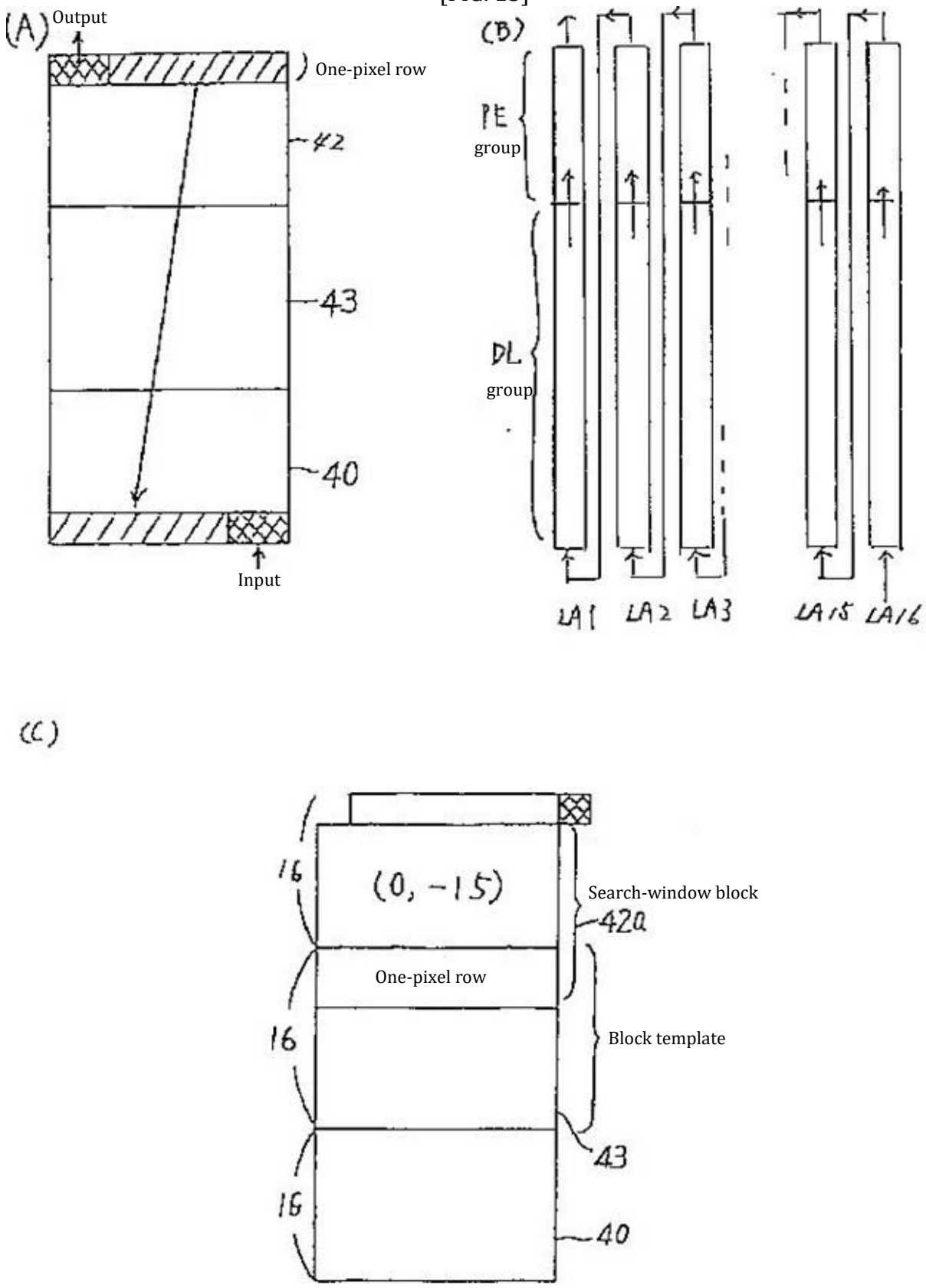
[FIG. 11]


22a
8 pixels
20a
16 pixels
15 pixels
8 pixels
8 pixels

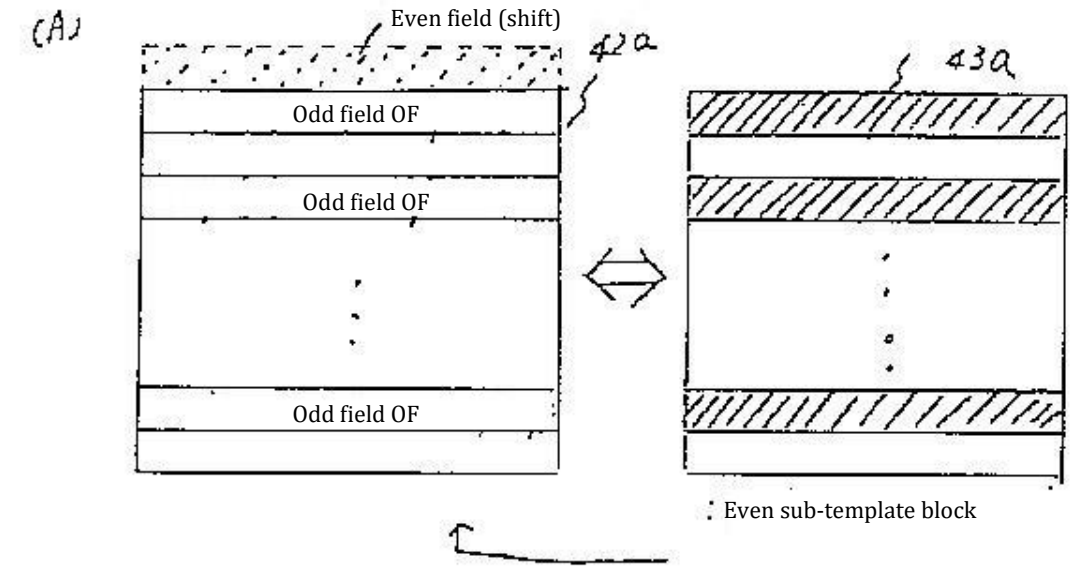
[FIG. 12]

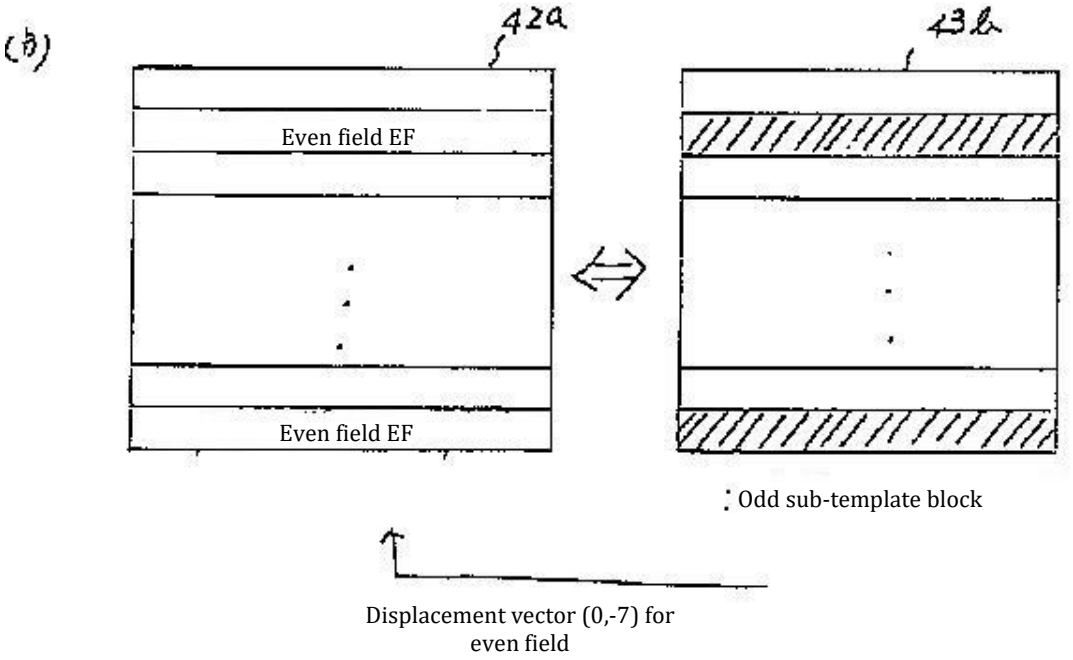

16 pixels
(0,−16)
Search-window-block data in PE
42
16 pixels
Template-block data in PE array
43
16 pixels
40
: frame (0,-16)
: field (0,-8)
: (horizontal component, vertical component
16 pixels

[FIG. 13]

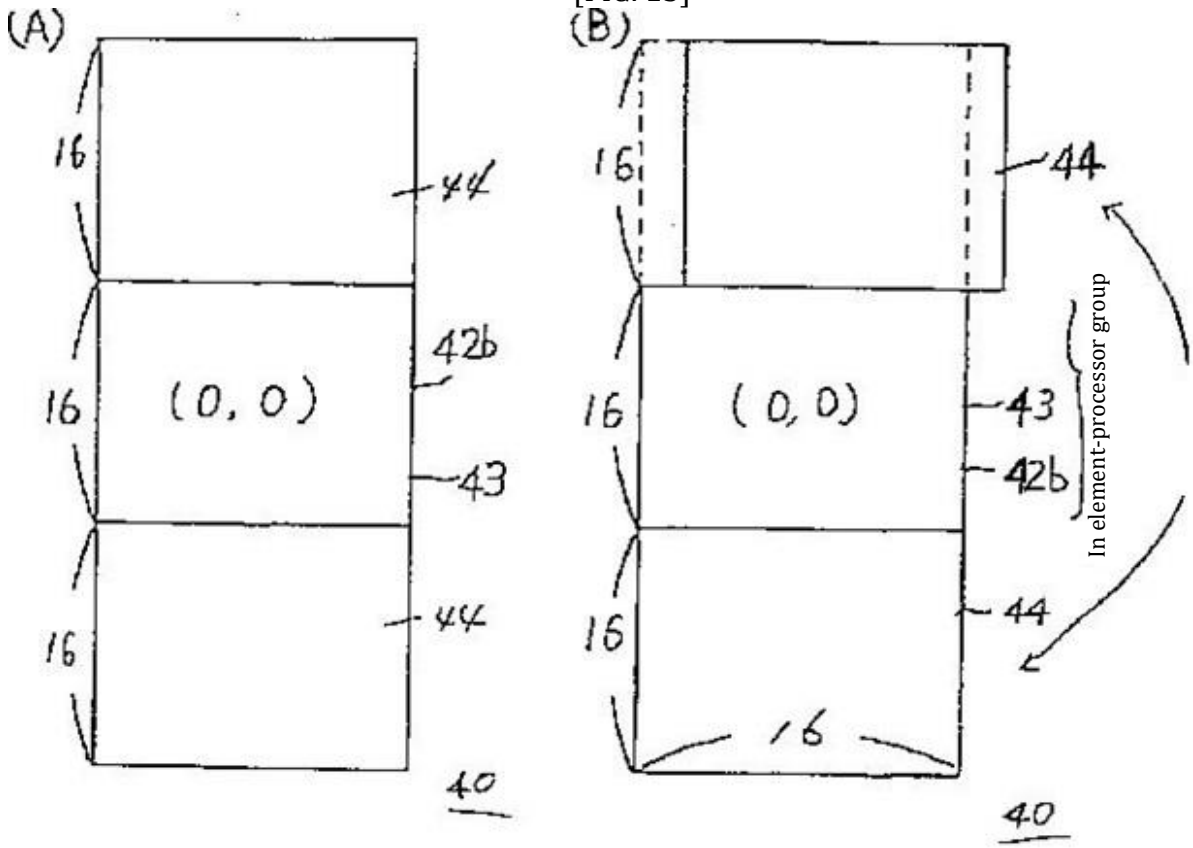

(A)
Output
One-pixel row
42
43
40
Input
(B)
PE group
DL group
LA1
LA2
LA3
LA15
LA16
(C)
16
(0, -15)
Search-window block
42a
One-pixel row
16
Block template
43
16
40

[FIG. 14]

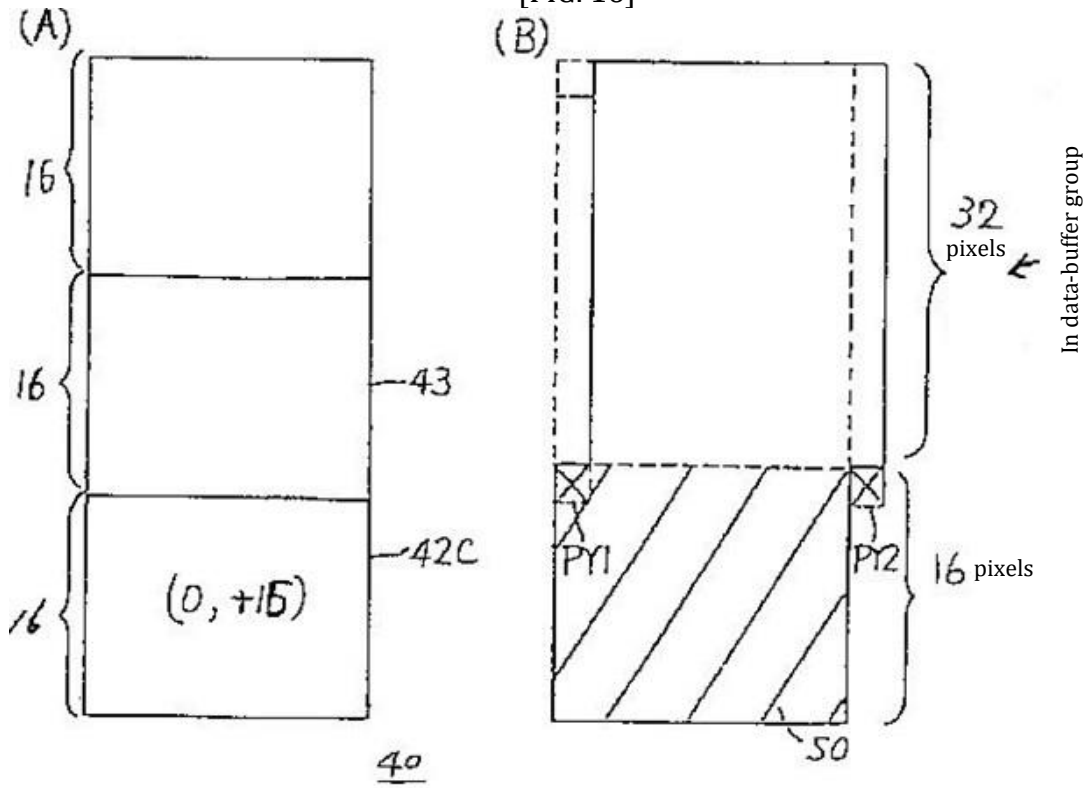

(A)
Even field (shift)
42a
43a
Odd field OF
Odd field OF
Odd field OF
: Even sub-template block
Displacement vector (0,-8)
for odd field


(b)
42a
43b
Even field EF
Even field EF
: Odd sub-template block
Displacement vector (0,-7) for
even field

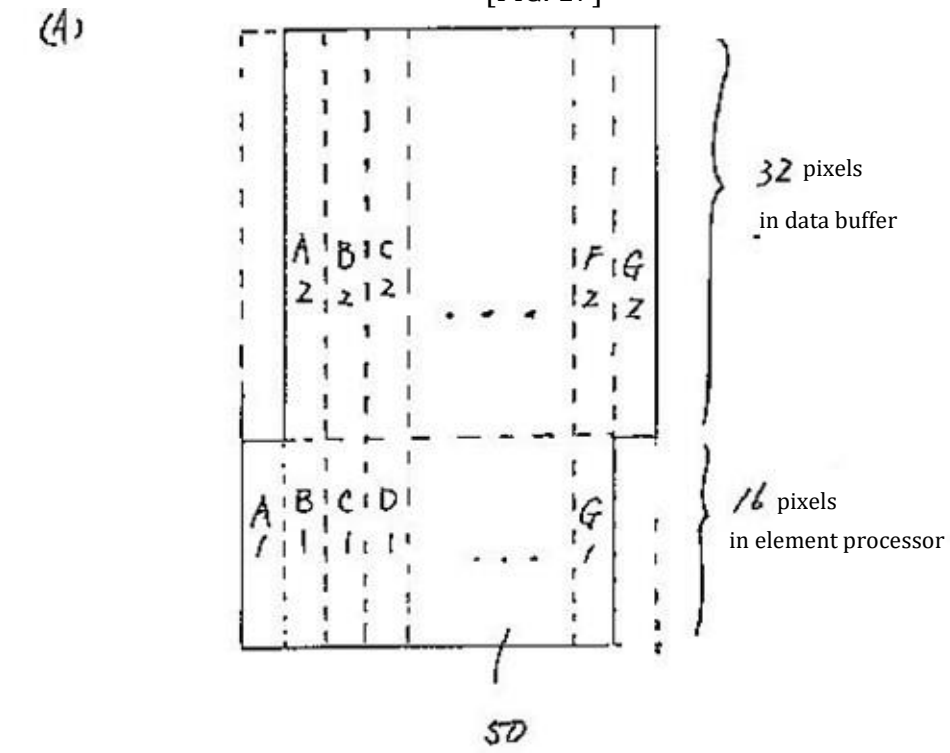
[FIG. 15]

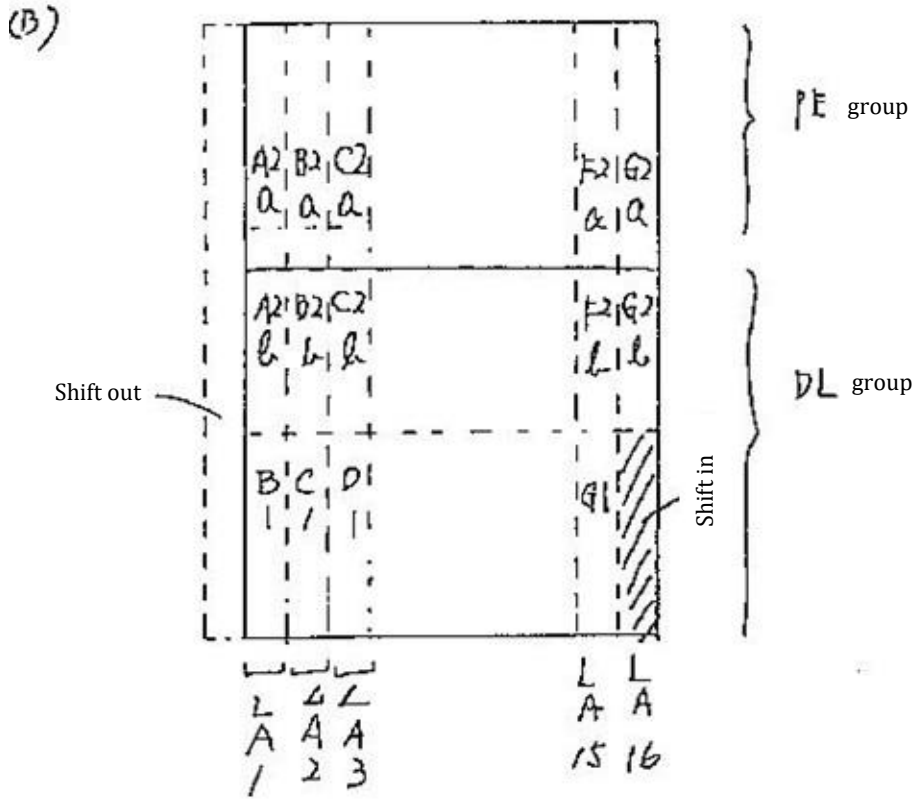

(A)
16
44
42b
(0, 0)
43
16
44
16
40
(B)
16
44
16
(0, 0)
43
42b
In element-processor group
16
44
16
40

[FIG. 16]

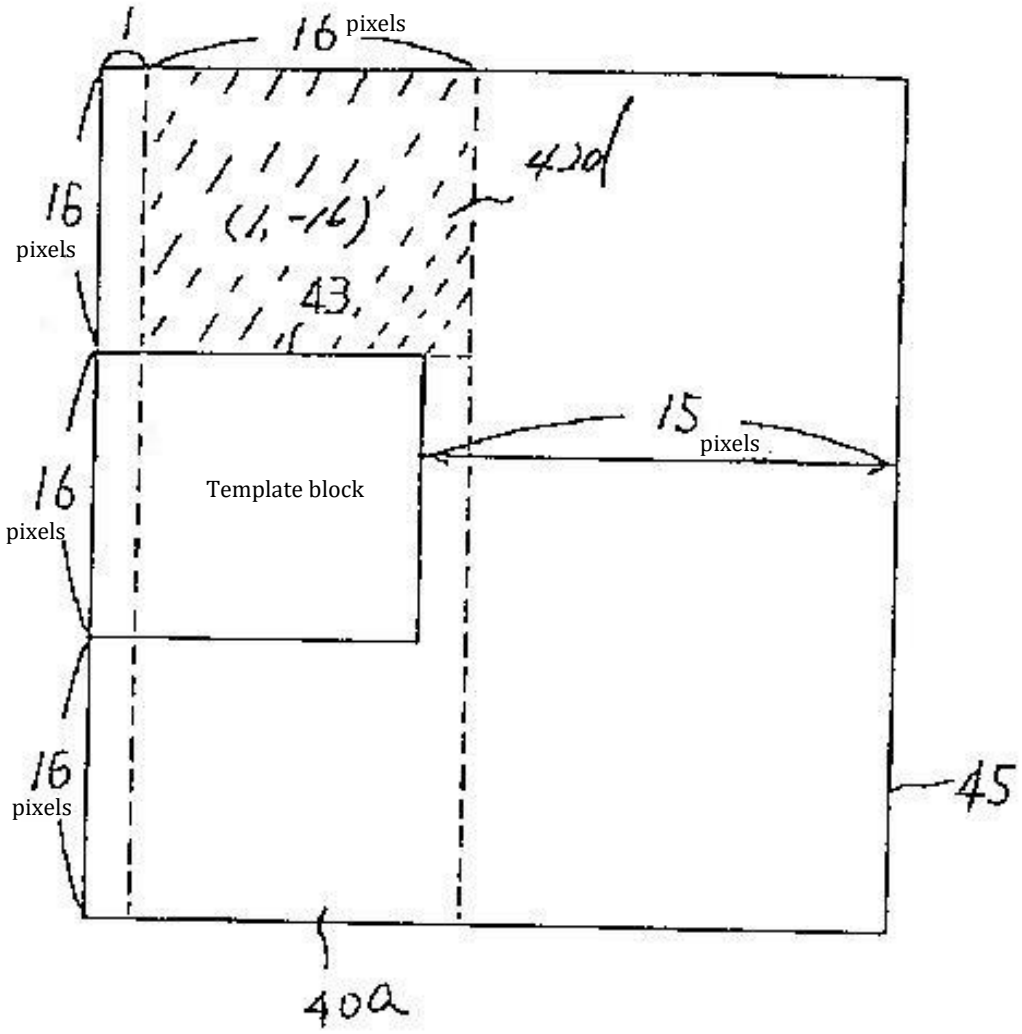

(A)
16
16
43
42c
(0, +16)
16
40
(B)
32
pixels
In data-buffer group
PY1
PY2
16 pixels
50

[FIG. 17]


(A)
32 pixels
in data buffer
16 pixels
in element processor
50


(B)
PE group
DL group
Shift out
Shift in

[FIG. 18]



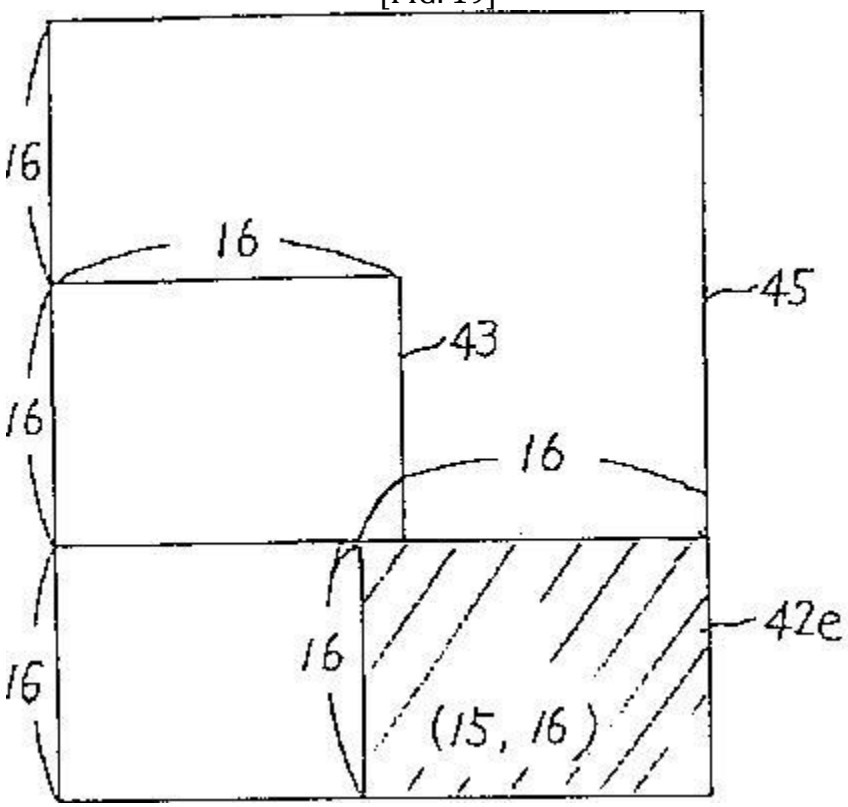
: Odd-sub-template block

: (1,-8) for odd field

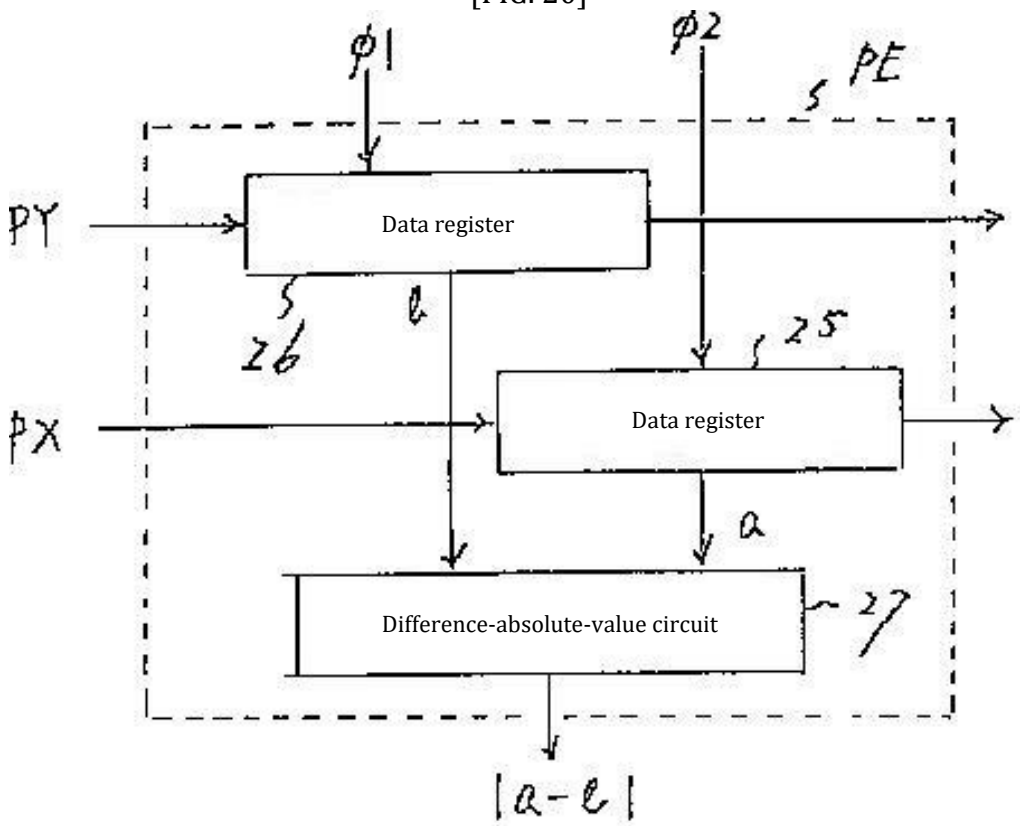
: Even-sub-template block

(1,-8) for even field

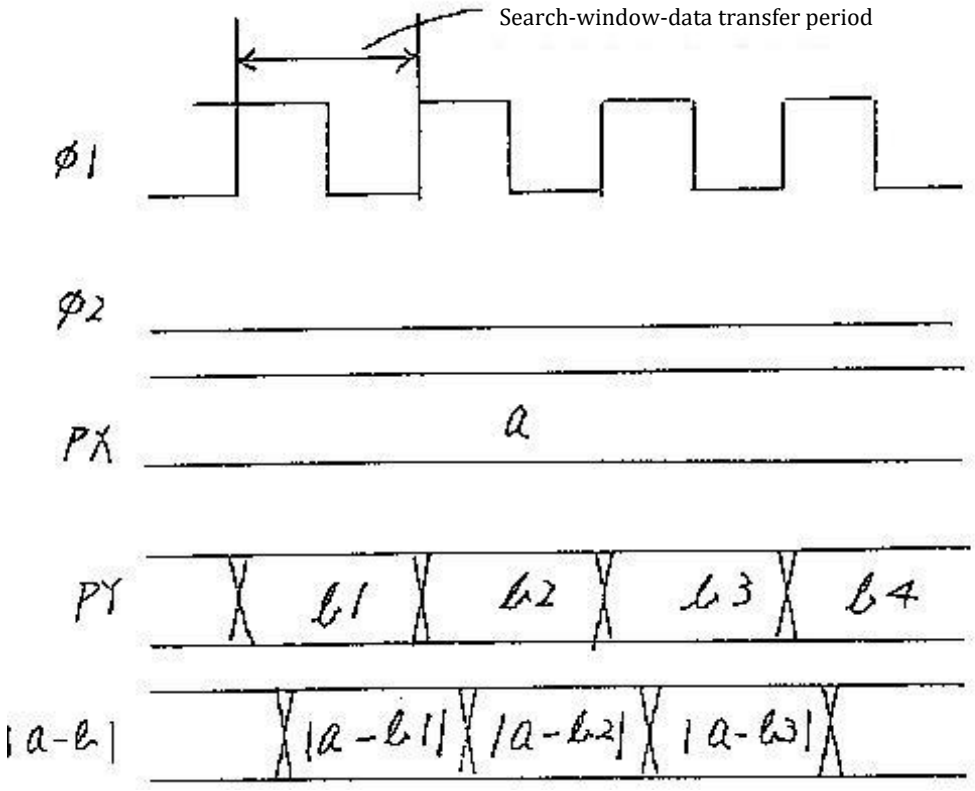
[FIG. 19]


16
16
45
43
16
16
16
42e
16
(15, 16)

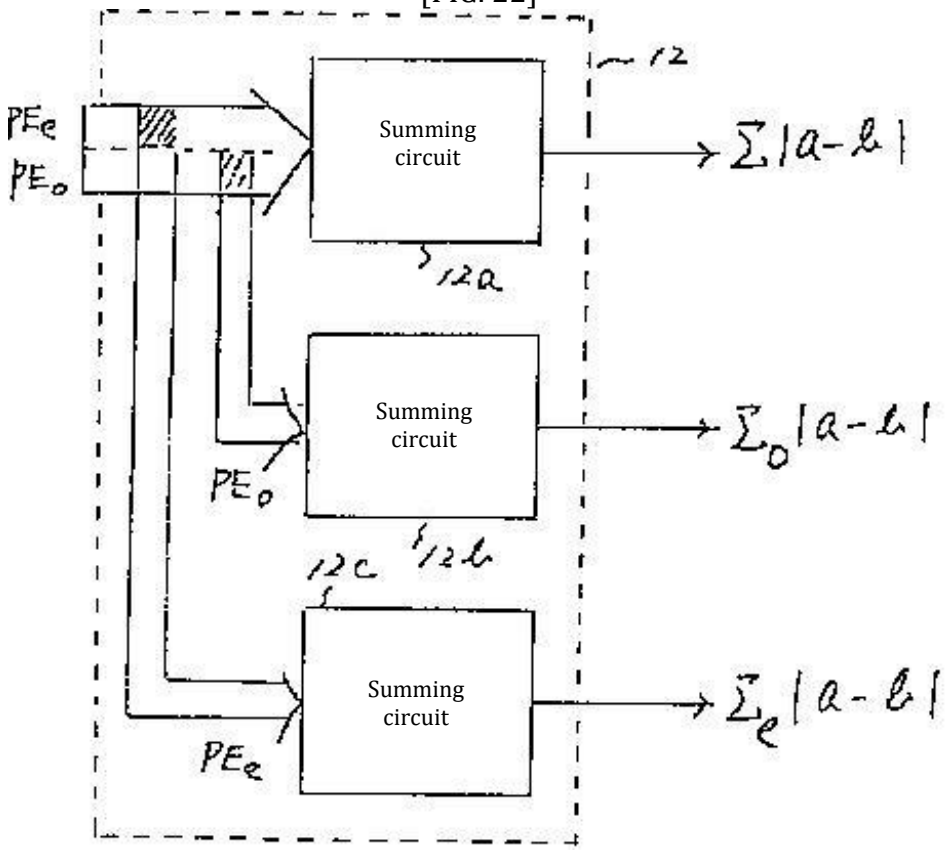
[FIG. 20]


φ1
φ2
PE
5
PY
Data register
5
26
b
25
PX
Data register
a
Difference-absolute-value circuit
27
|a-b|

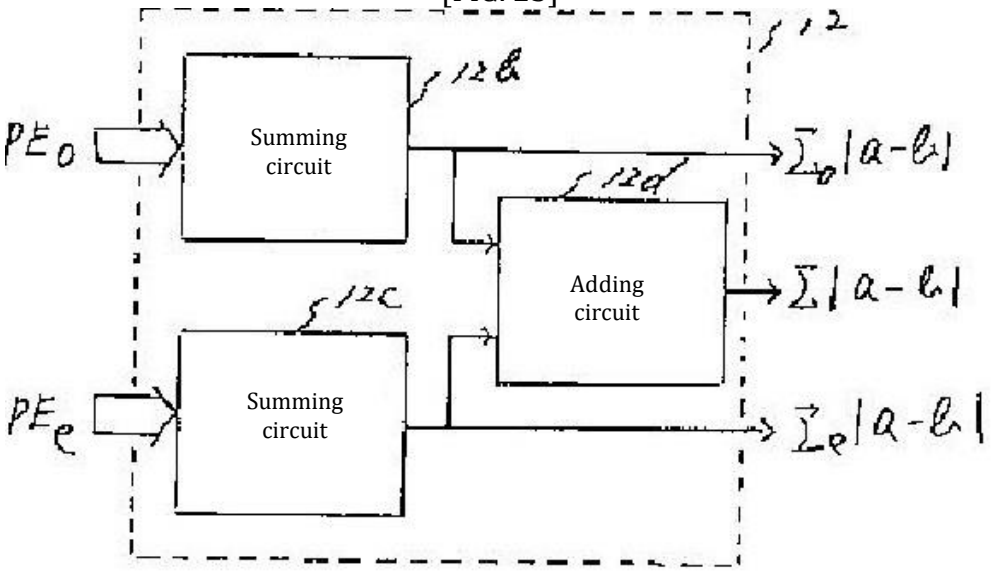
[FIG. 21]


Search-window-data transfer period
φ1
φ2
PX
a
PY
b1
b2
b3
b4
|a-b|
|a-b1|
|a-b2|
|a-b3|

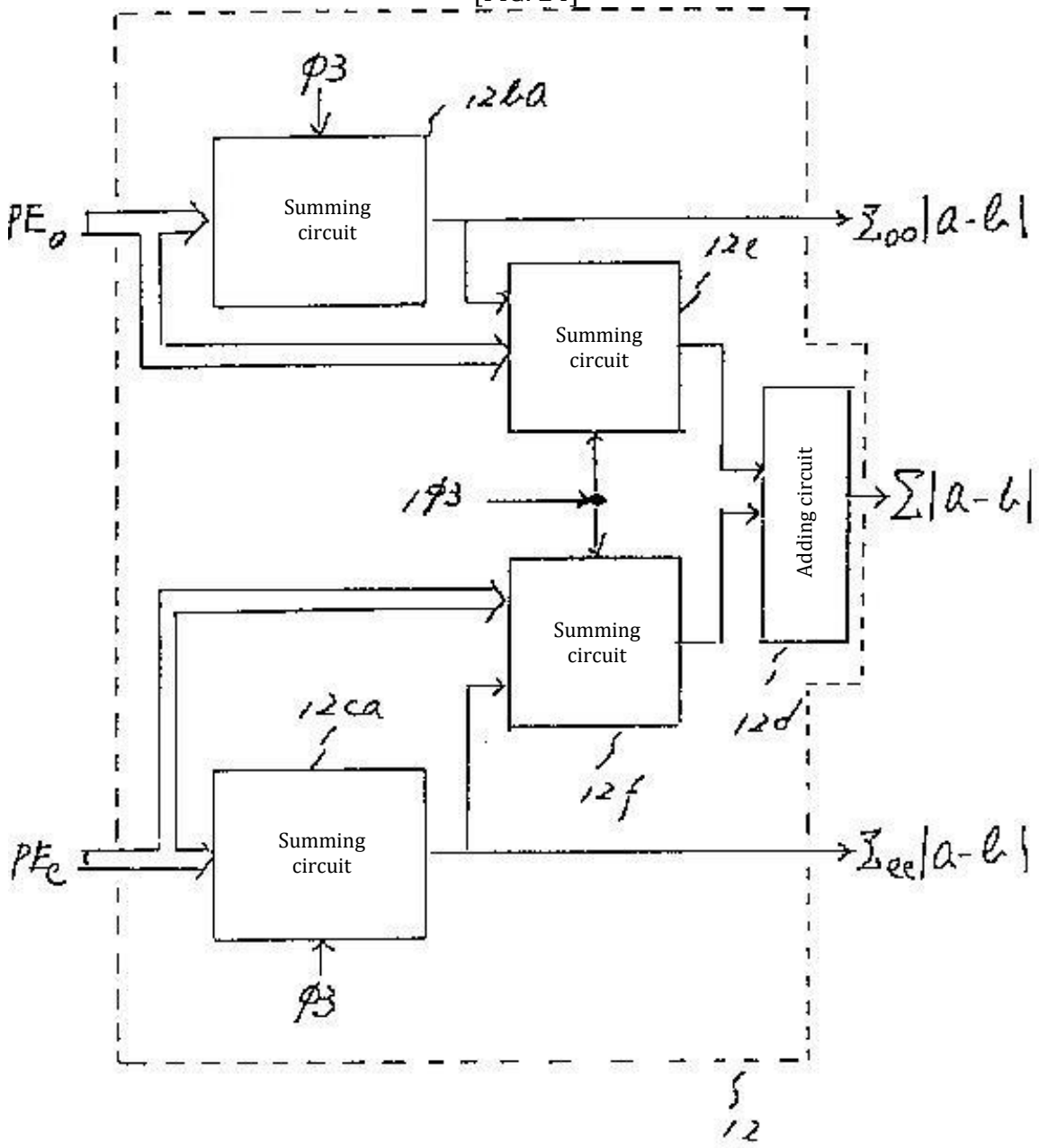
[FIG. 22]


12
PEe
PEo
Summing circuit
Σ|a-b|
12a
Summing circuit
Σo|a-b|
PEo
12b
12c
Summing circuit
Σe|a-b|
PEe

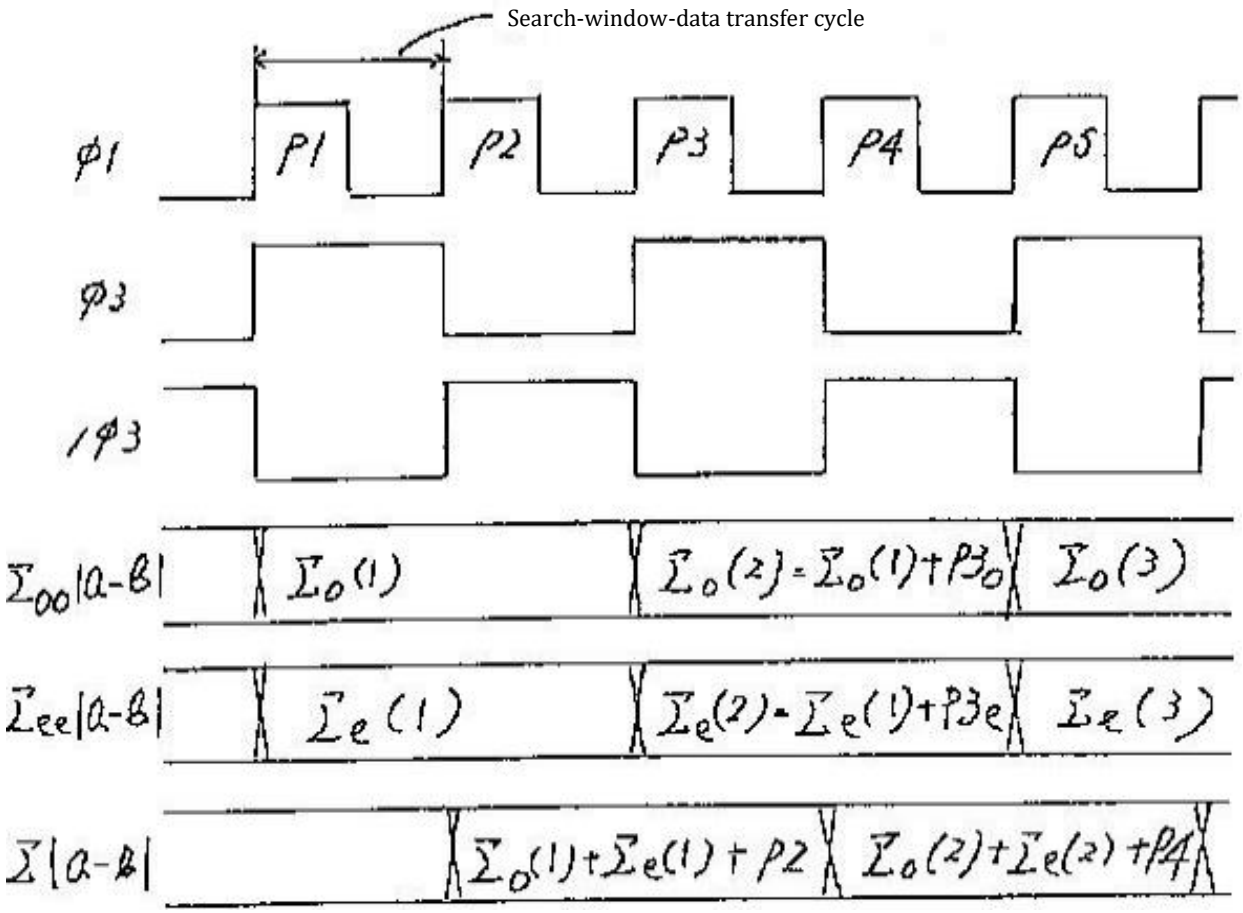
[FIG. 23]


12
12b
12d
12c
Summing circuit
Adding circuit
Summing circuit
PE0
PEe
Σo|a-b|
Σ|a-b|
Σe|a-b|

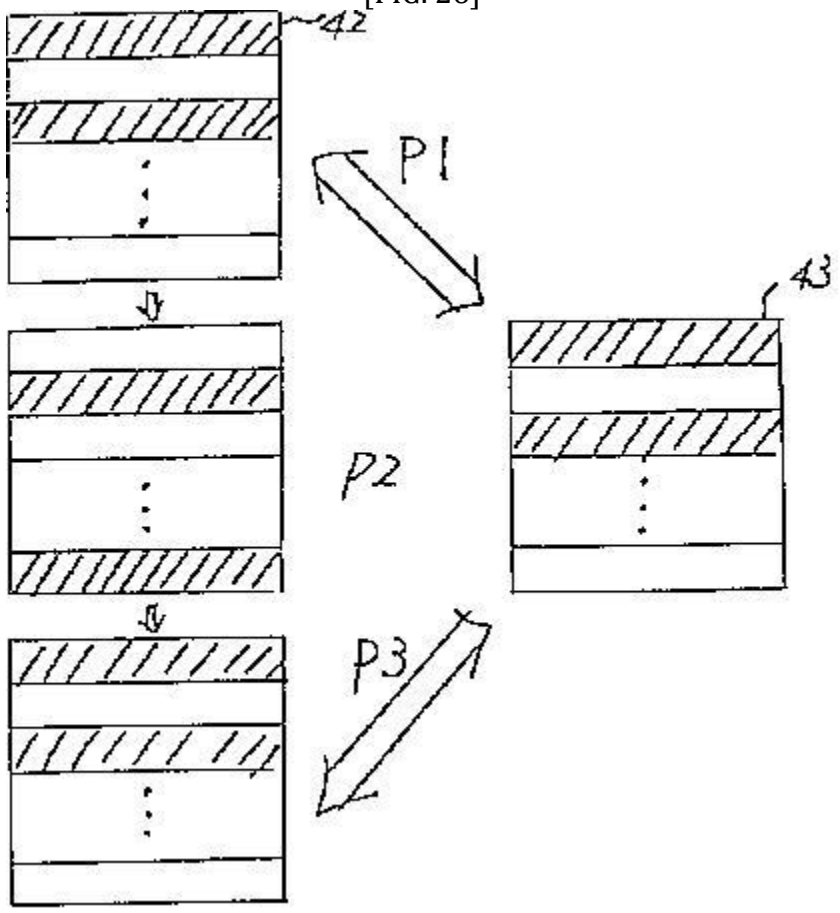
[FIG. 24]


φ3
12ba
Summing circuit
PE_o
Σ_oo|a-b|
12e
Summing circuit
Adding circuit
1φ3
Σ|a-b|
Summing circuit
12ca
12d
12f
Summing circuit
PE_e
Σ_ee|a-b|
φ3
12

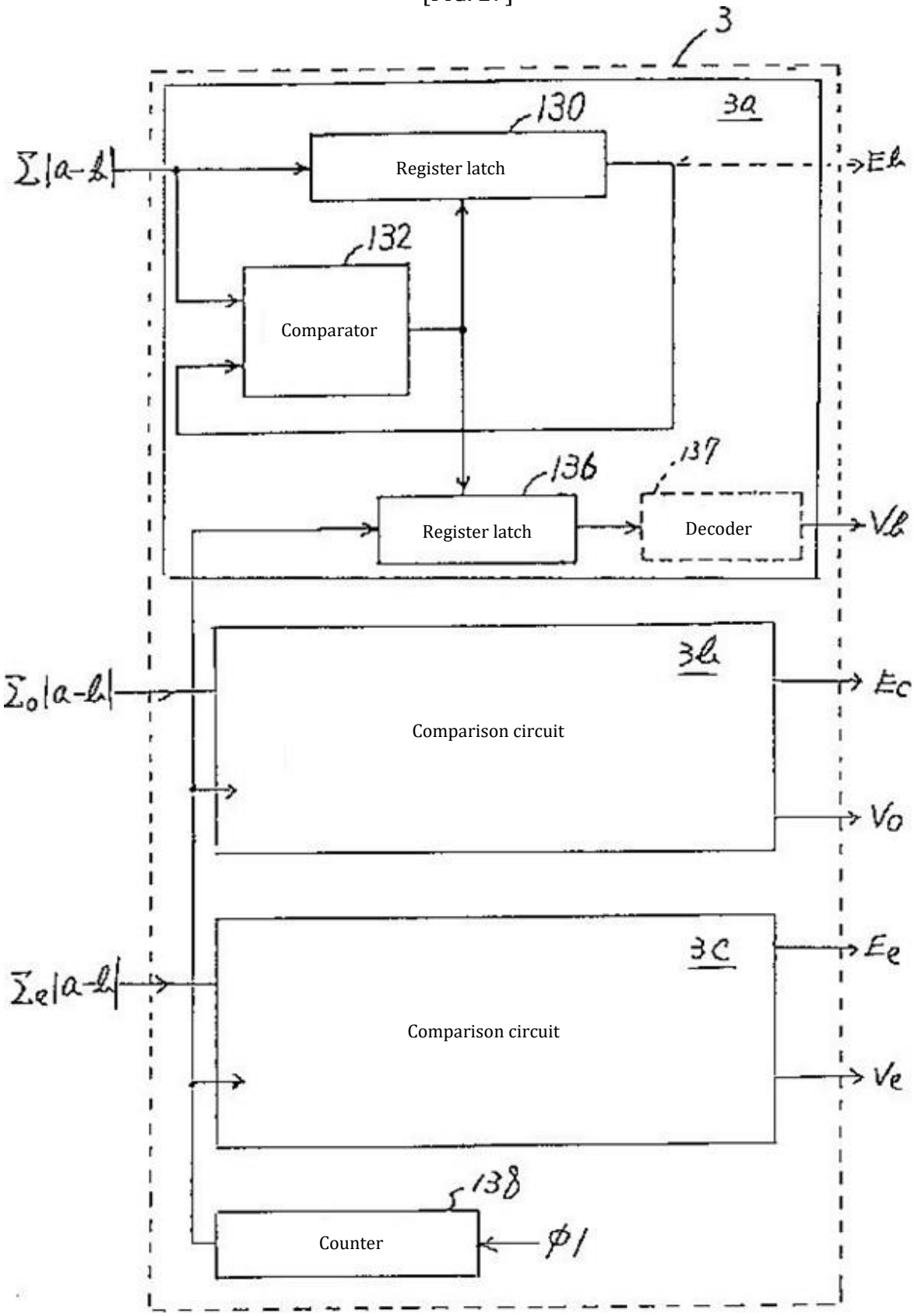
[FIG. 25]


Search-window-data transfer cycle
φ1
P1
P2
P3
P4
P5
φ3
/φ3
Σ00|a-b|
Σo(1)
Σo(2)=Σo(1)+P3o
Σo(3)
Σee|a-b|
Σe(1)
Σe(2)=Σe(1)+P3e
Σe(3)
Σ|a-b|
Σo(1)+Σe(1)+P2
Σo(2)+Σe(2)+P4

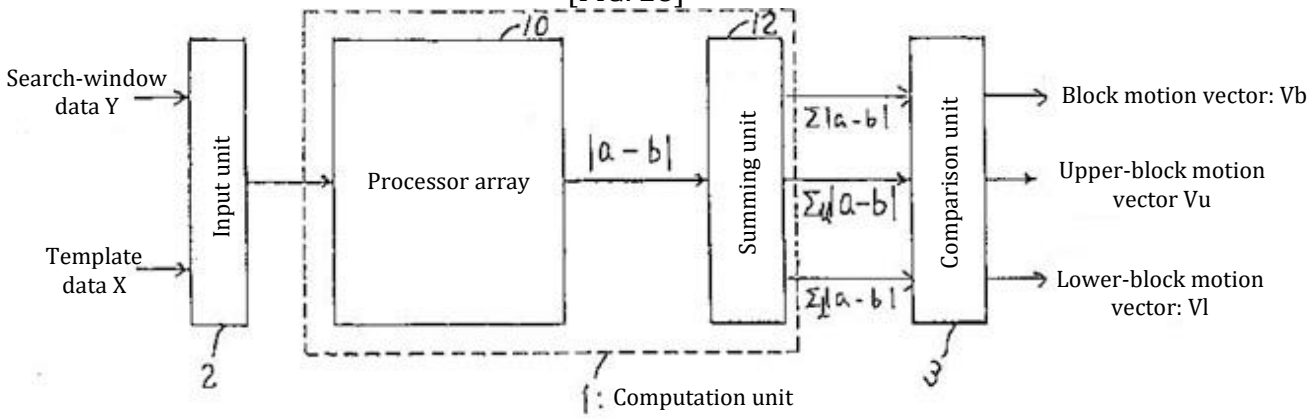
[FIG. 26]



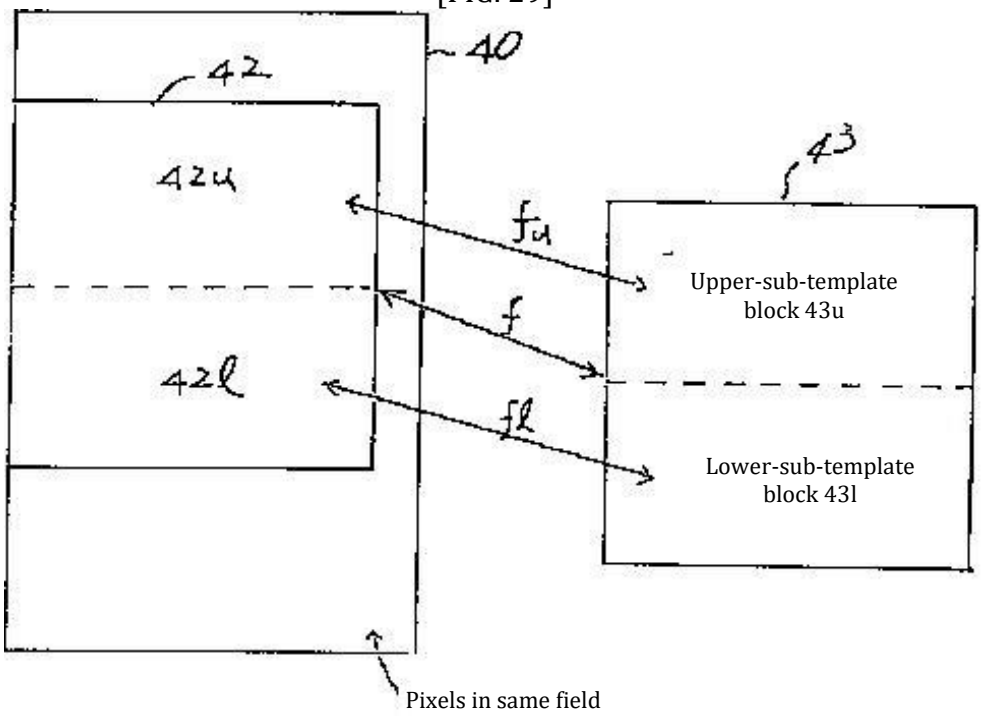
[FIG. 27]


3
130
3a
Register latch
Eb
132
Comparator
136
137
Register latch
Decoder
Vb
3b
Comparison circuit
Ec
Vo
3c
Comparison circuit
Ee
Ve
138
Counter
φ1

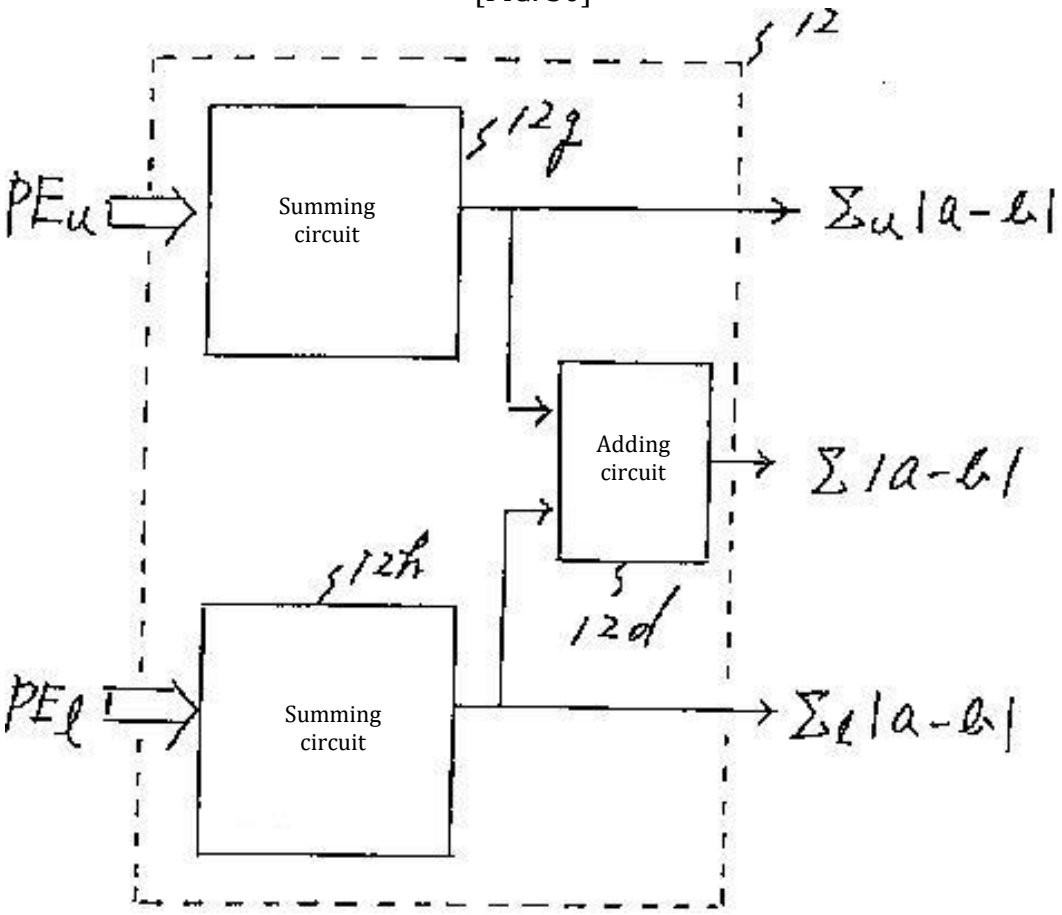
[FIG. 28]


Search-window data Y
Template data X
Input unit
Processor array
|a - b|
Summing unit
Σ|a-b|
Σu|a-b|
Σl|a-b|
Comparison unit
Block motion vector: Vb
Upper-block motion vector Vu
Lower-block motion vector: Vl
10
12
2
3
1: Computation unit

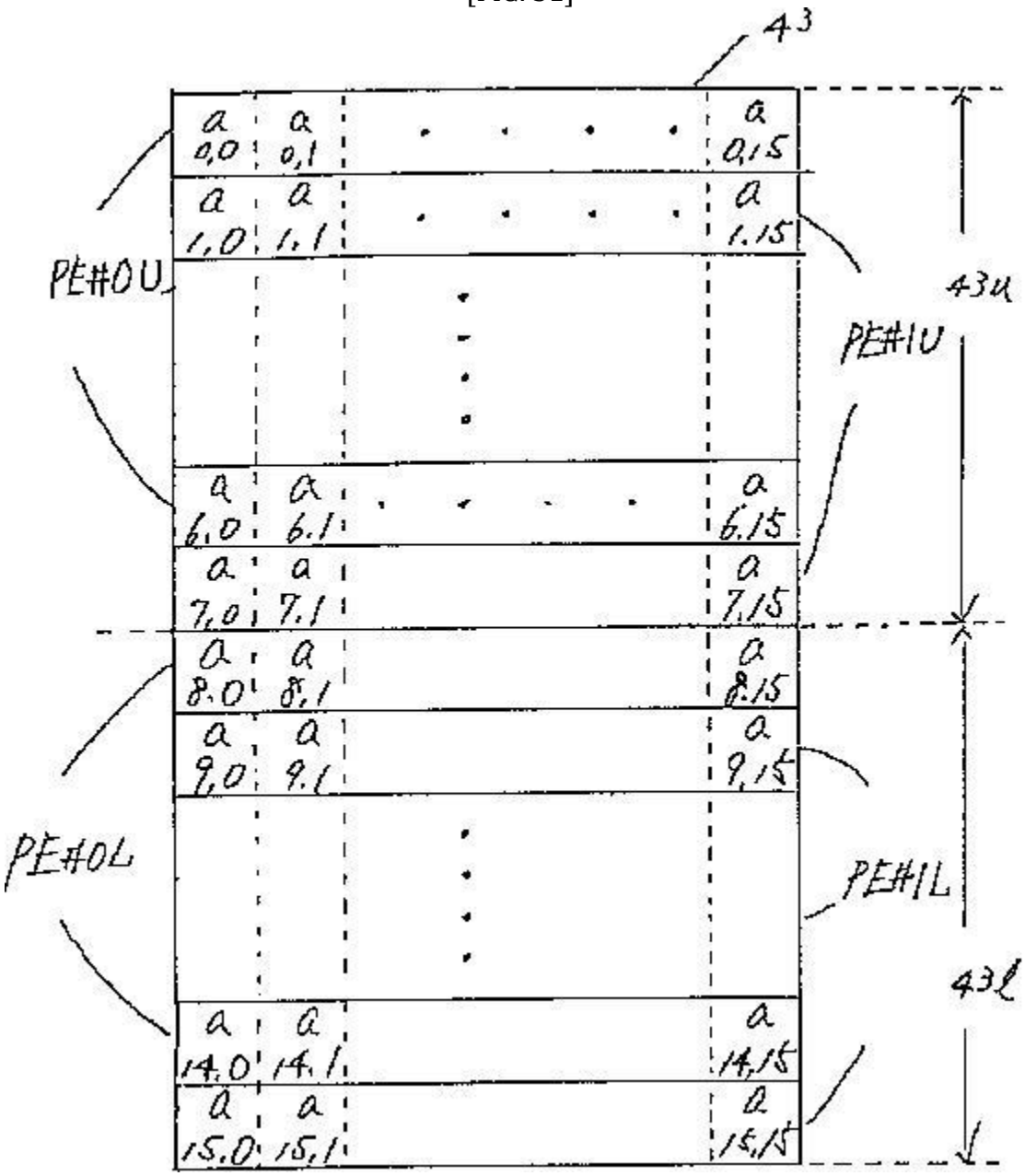
[FIG. 29]


40
42
42u
42l
43
fu
f
fl
Upper-sub-template block 43u
Lower-sub-template block 43l
Pixels in same field

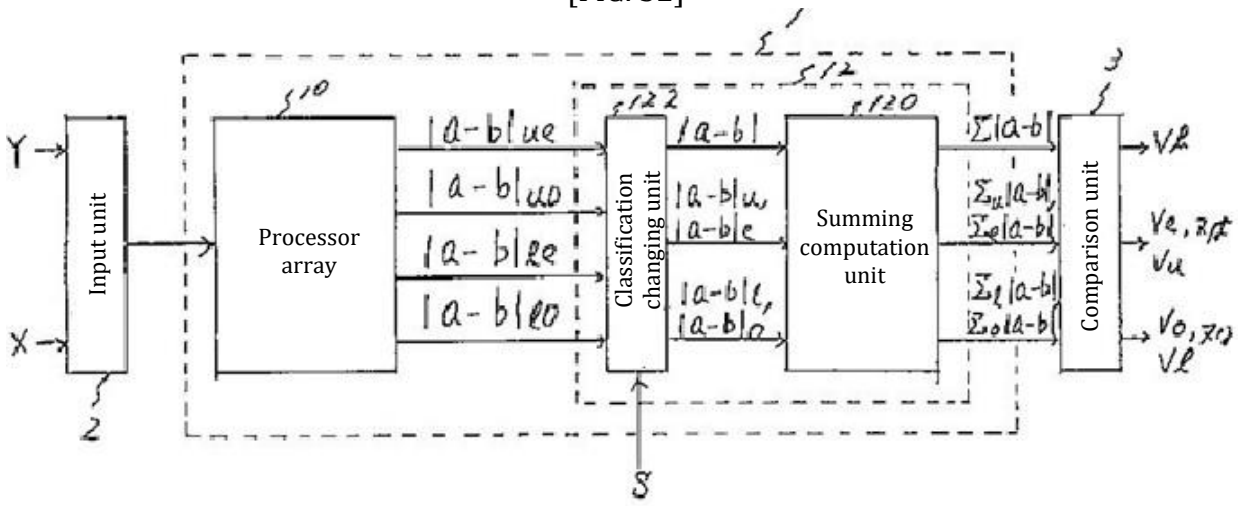
[FIG. 30]


12
12g
PEu
Summing circuit
Σu|a-b|
Adding circuit
Σ|a-b|
12h
12d
PEl
Summing circuit
Σl|a-b|

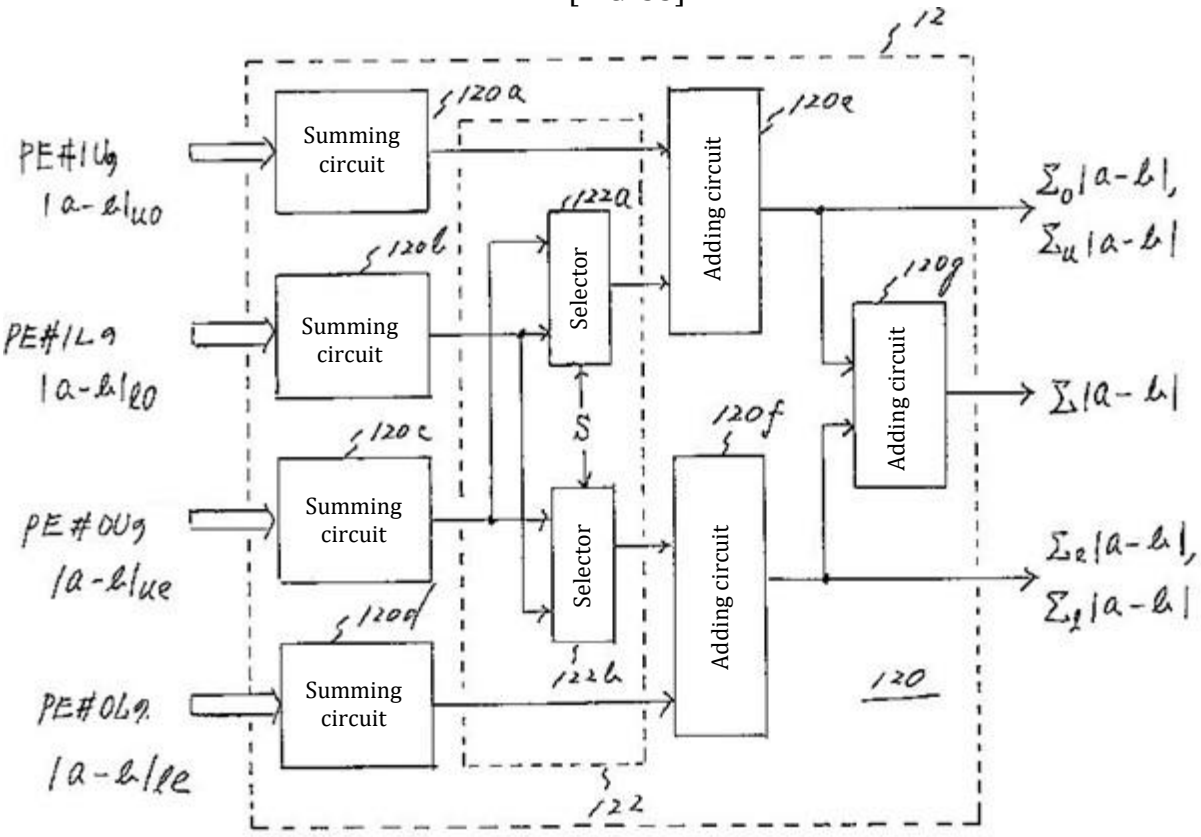
[FIG. 31]


43
a 0,0
a 0,1
a 0,15
a 1,0
a 1,1
a 1,15
PE#0U
PE#1U
43u
a 6,0
a 6,1
a 6,15
a 7,0
a 7,1
a 7,15
a 8,0
a 8,1
a 8,15
a 9,0
a 9,1
a 9,15
PE#0L
PE#1L
43l
a 14,0
a 14,1
a 14,15
a 15,0
a 15,1
a 15,15

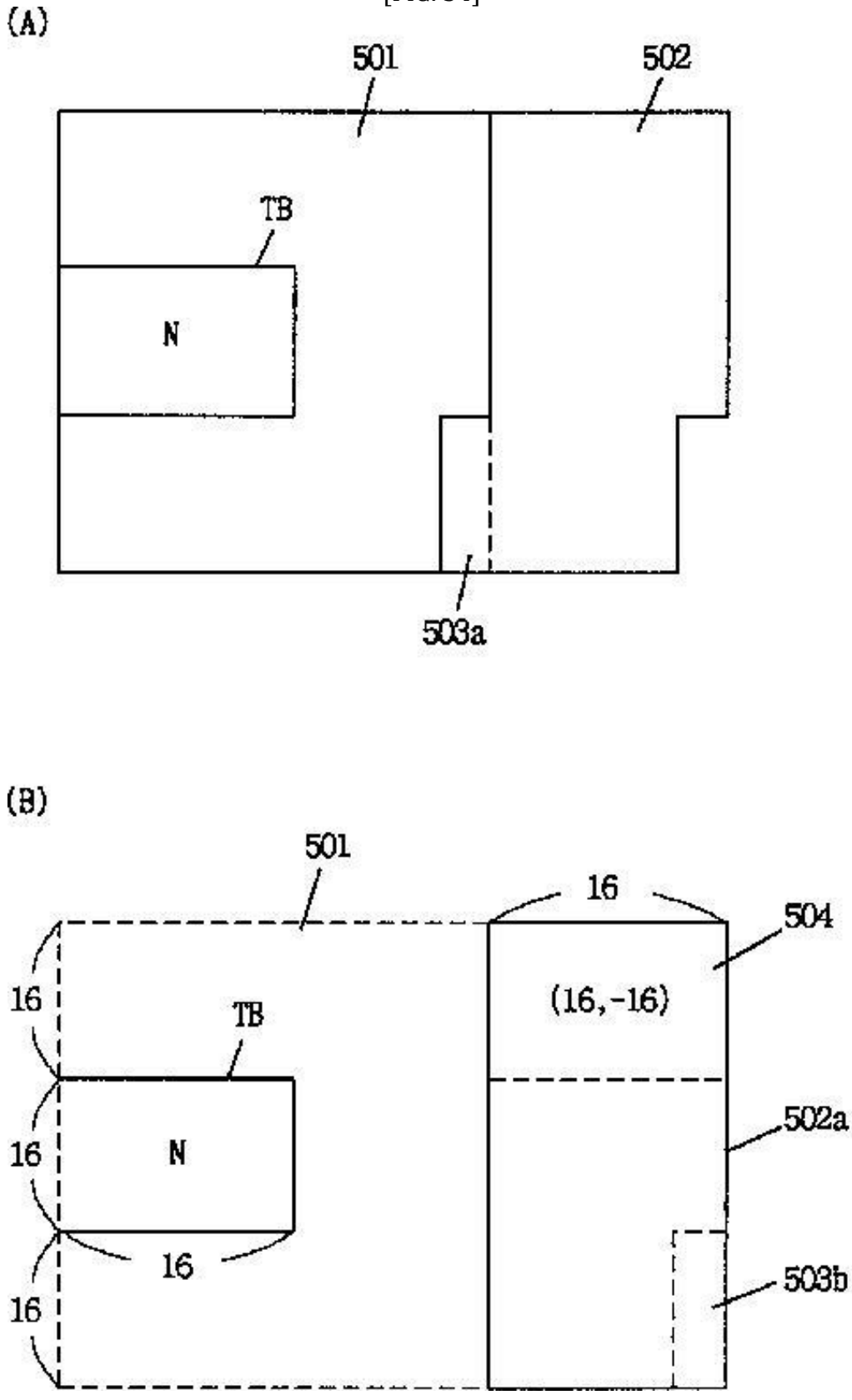
[FIG. 32]


Y
X
Input unit
2
Processor array
10
|a-b|ue
|a-b|uo
|a-b|le
|a-b|lo
Classification changing unit
122
|a-b|
|a-b|u,
|a-b|e
|a-b|l,
|a-b|o
S
Summing computation unit
120
Σ|a-b|
Σu|a-b|,
Σe|a-b|
Σl|a-b|
Σo|a-b|
Comparison unit
3
Ve
Vu
Vl
Vo
12

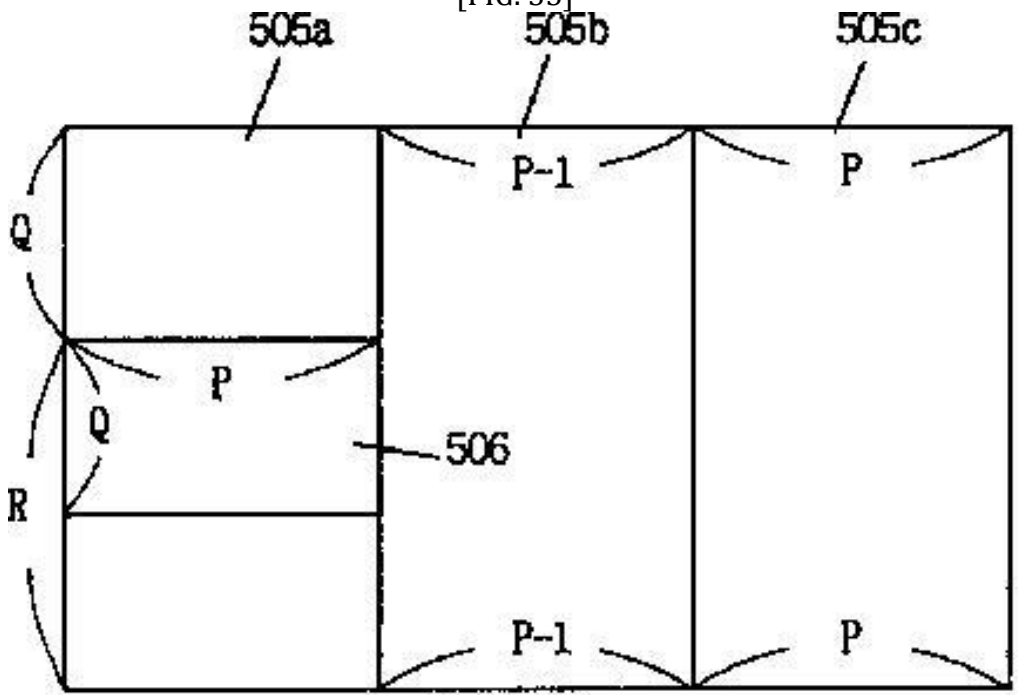
[FIG. 33]


12
PE#1Ug
|a-b|uo
PE#1Lg
|a-b|lo
PE#0Ug
|a-b|ue
PE#0Lg
|a-b|le
Summing circuit
120a
Summing circuit
120b
Summing circuit
120c
Summing circuit
120d
Selector
122a
Selector
122b
S
Adding circuit
120e
Adding circuit
120f
Adding circuit
120g
Σo|a-b|,
Σu|a-b|
Σ|a-b|
Σe|a-b|,
Σl|a-b|
120
122

[FIG. 34]

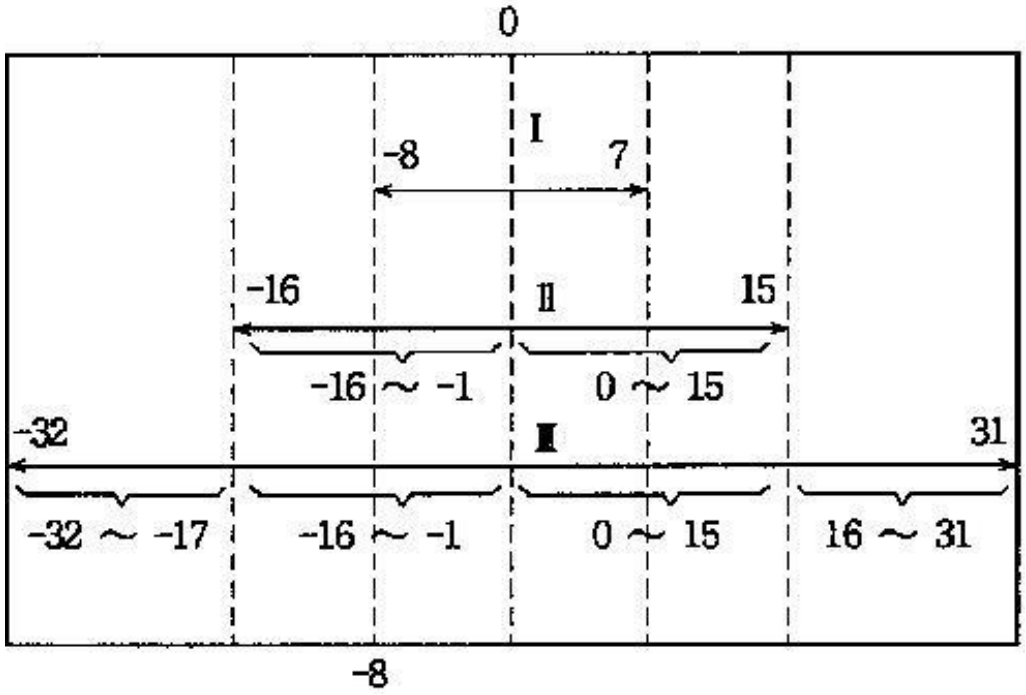


[FIG. 35]

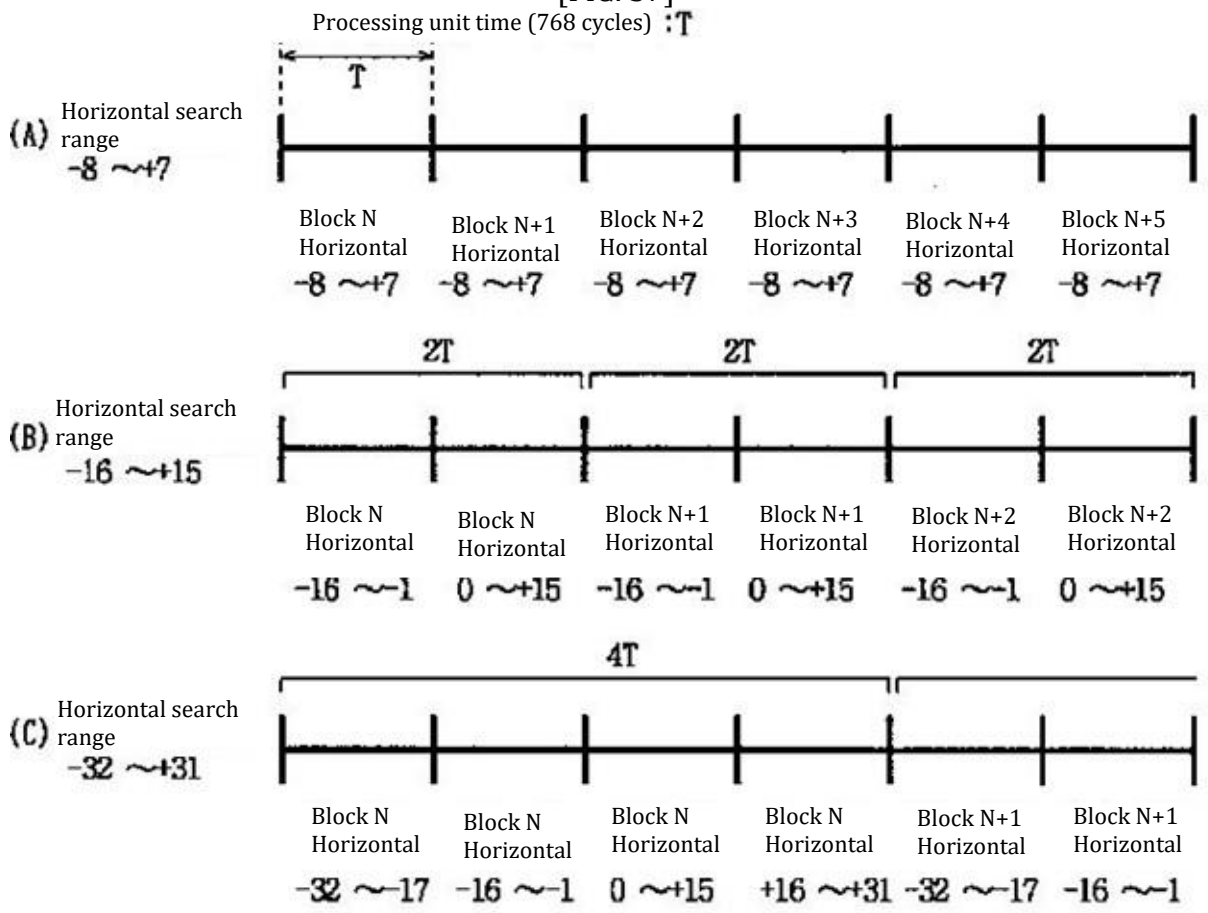


[FIG. 36]

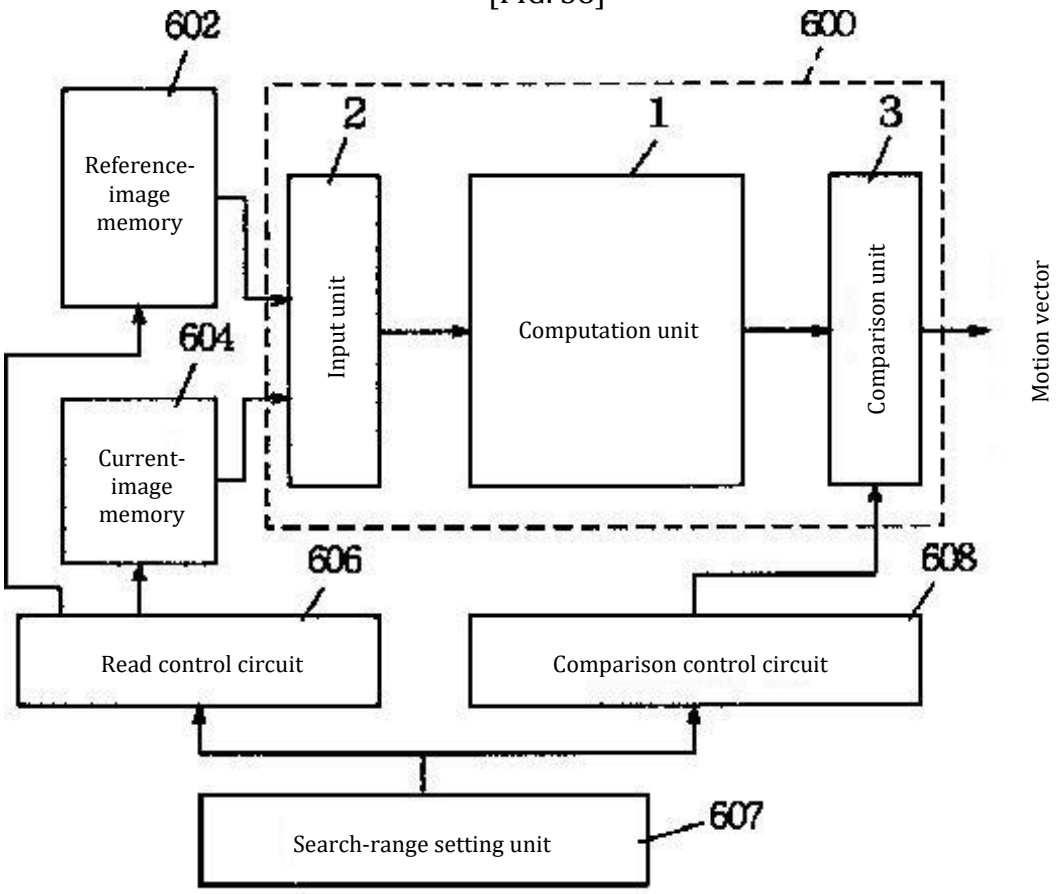
Processing unit cycle: 768 cycles



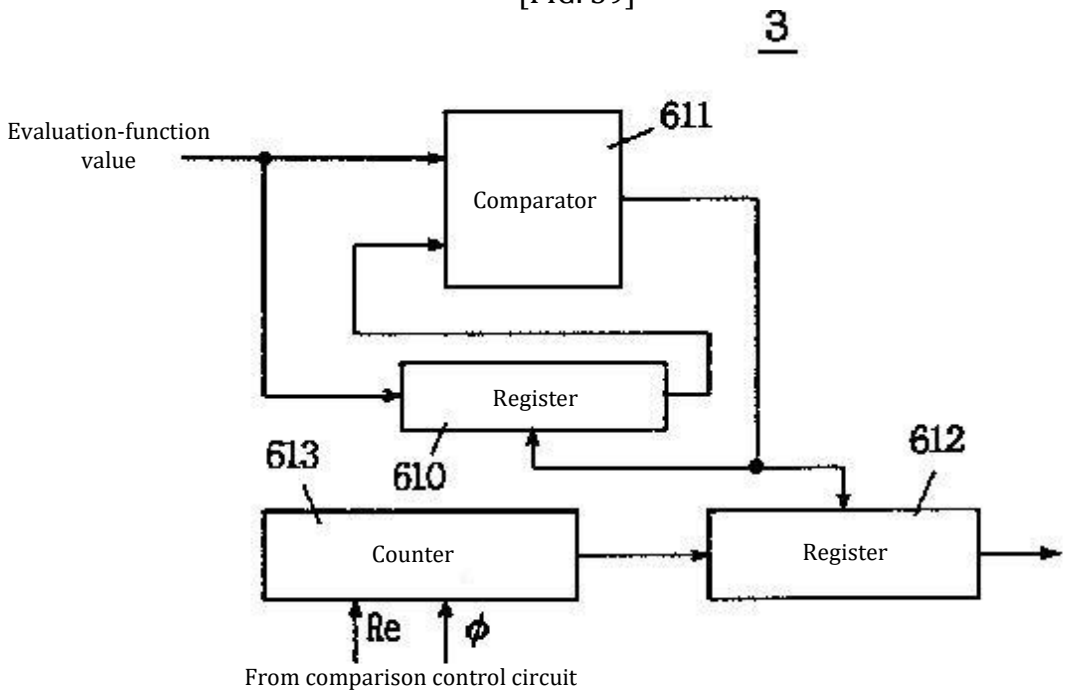
[FIG. 37]


Processing unit time (768 cycles) :T
T
(A) Horizontal search range
-8 ~+7
Block N Horizontal
-8 ~+7
Block N+1 Horizontal
-8 ~+7
Block N+2 Horizontal
-8 ~+7
Block N+3 Horizontal
-8 ~+7
Block N+4 Horizontal
-8 ~+7
Block N+5 Horizontal
-8 ~+7
2T
2T
2T
(B) Horizontal search range
-16 ~+15
Block N Horizontal
-16 ~-1
Block N Horizontal
0 ~+15
Block N+1 Horizontal
-16 ~-1
Block N+1 Horizontal
0 ~+15
Block N+2 Horizontal
-16 ~-1
Block N+2 Horizontal
0 ~+15
4T
(C) Horizontal search range
-32 ~+31
Block N Horizontal
-32 ~-17
Block N Horizontal
-16 ~-1
Block N Horizontal
0 ~+15
Block N Horizontal
+16 ~+31
Block N+1 Horizontal
-32 ~-17
Block N+1 Horizontal
-16 ~-1

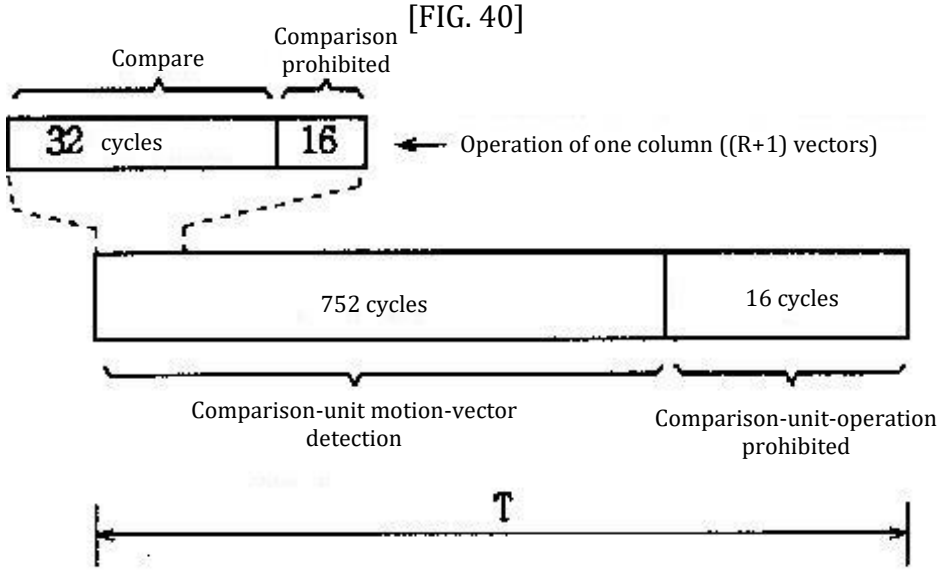
[FIG. 38]


602
600
2
1
3
Reference-image memory
Input unit
Computation unit
Comparison unit
Motion vector
604
Current-image memory
606
608
Read control circuit
Comparison control circuit
Search-range setting unit
607

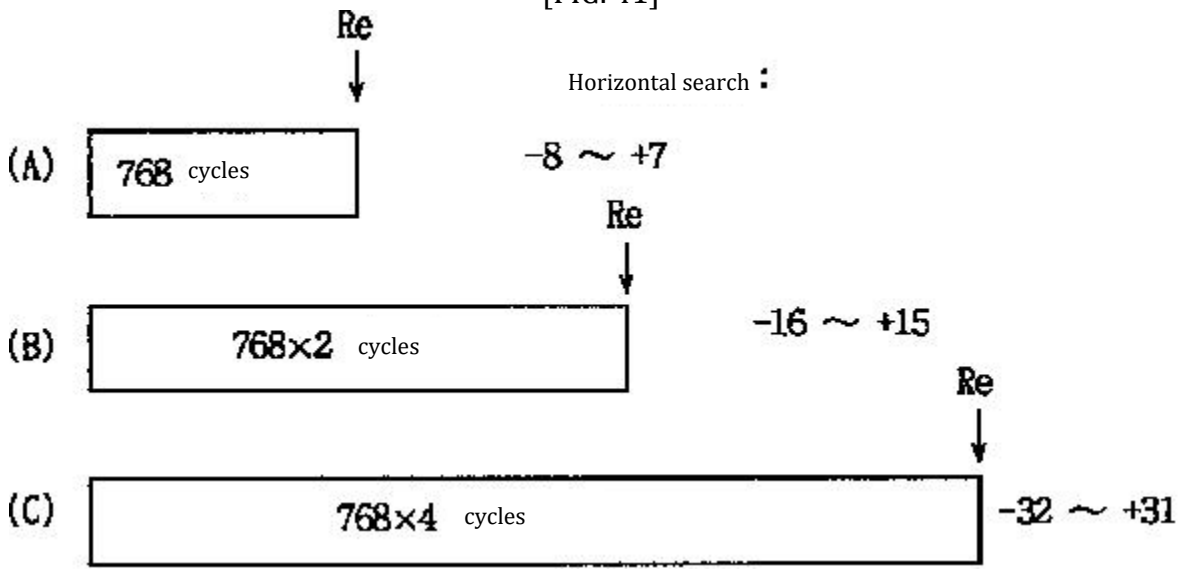
[FIG. 39]


3
Evaluation-function value
611
Comparator
Register
613
610
612
Counter
Register
Re
φ
From comparison control circuit

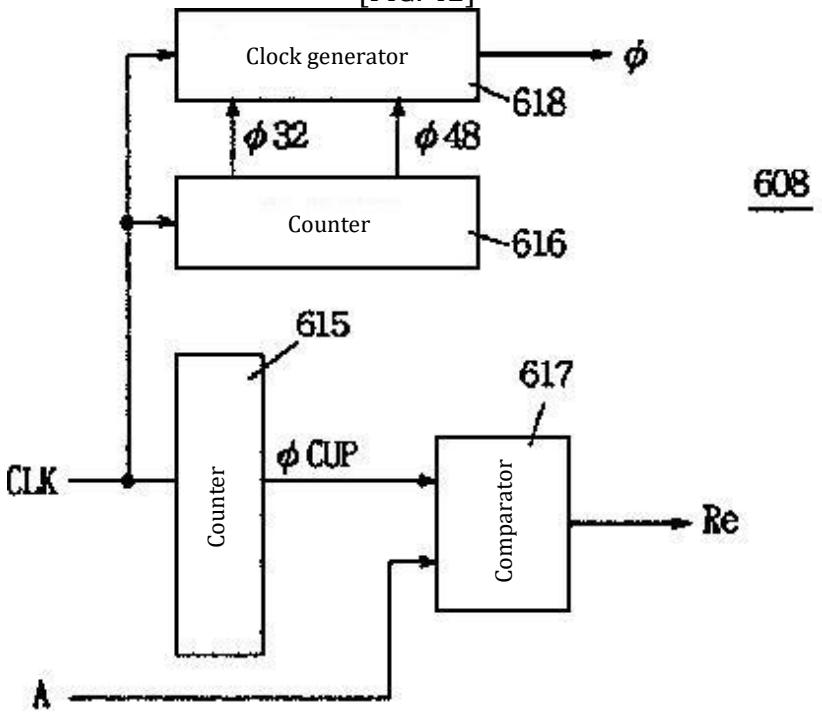
[FIG. 40]


Compare
Comparison prohibited
32 cycles
16
Operation of one column ((R+1) vectors)
752 cycles
16 cycles
Comparison-unit motion-vector detection
Comparison-unit-operation prohibited
T

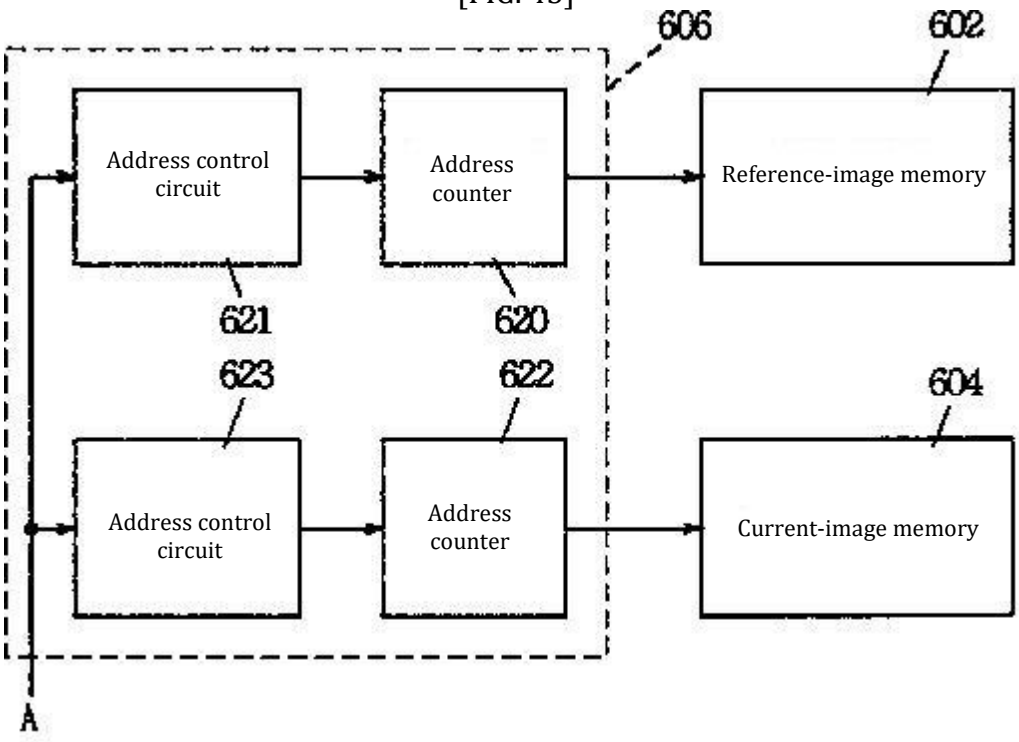
[FIG. 41]


Re
Horizontal search :
(A)
768 cycles
−8 ～ +7
Re
(B)
768×2 cycles
−16 ～ +15
Re
(C)
768×4 cycles
−32 ～ +31

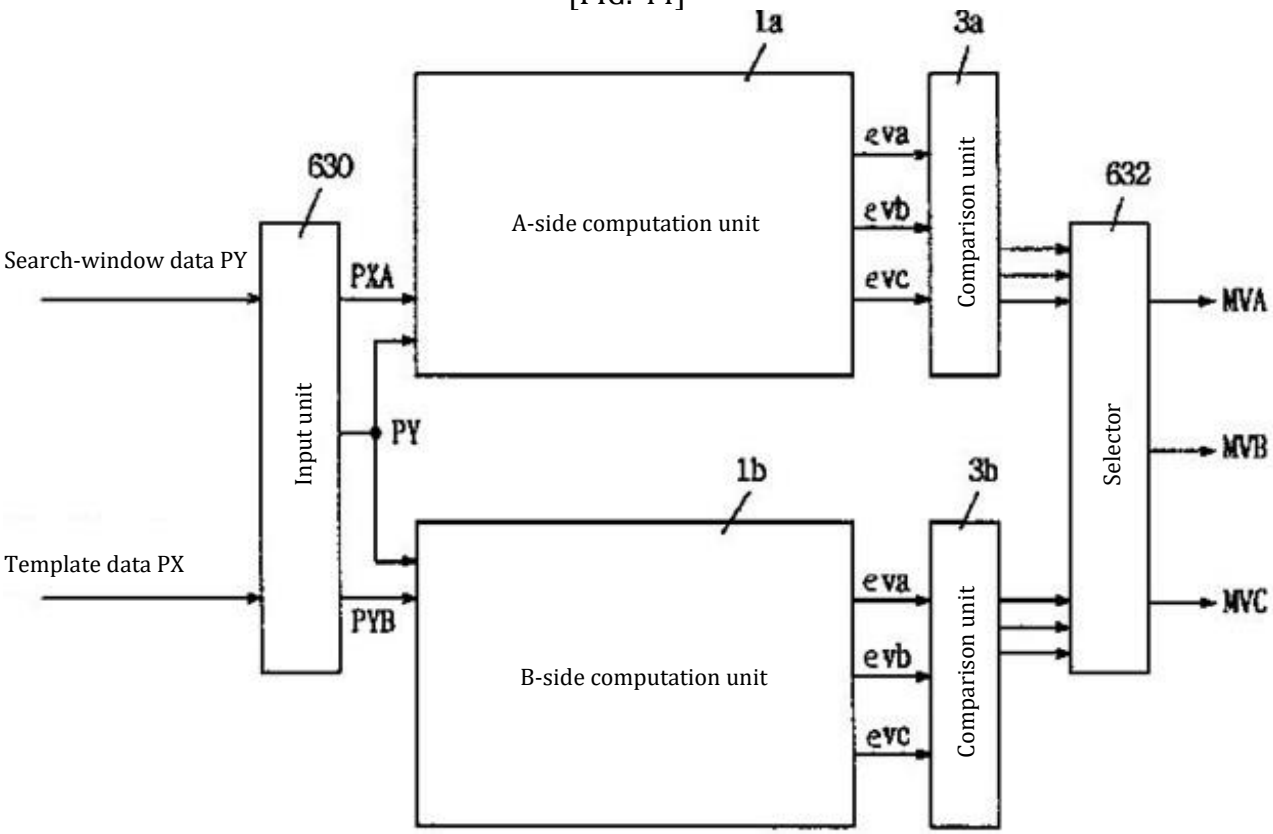
[FIG. 42]


Clock generator
φ
618
φ32
φ48
608
Counter
616
615
617
CLK
Counter
φCUP
Comparator
Re
A

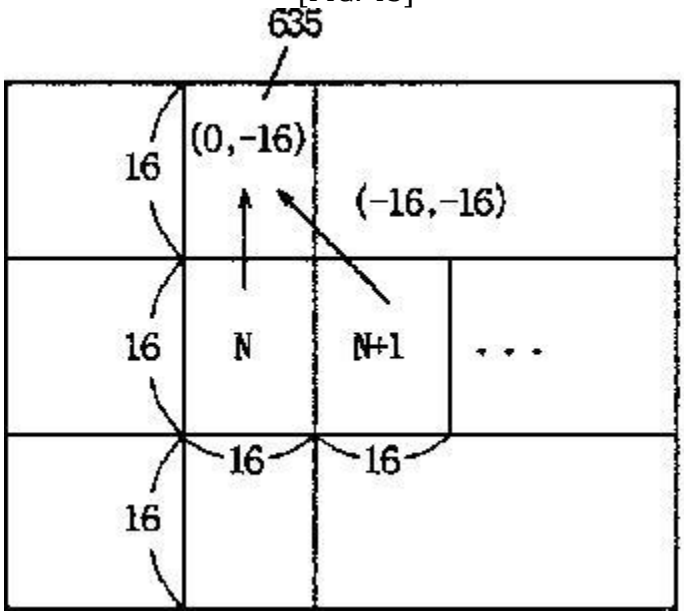
[FIG. 43]


606
602
Address control circuit
Address counter
Reference-image memory
621
620
623
622
604
Address control circuit
Address counter
Current-image memory
A

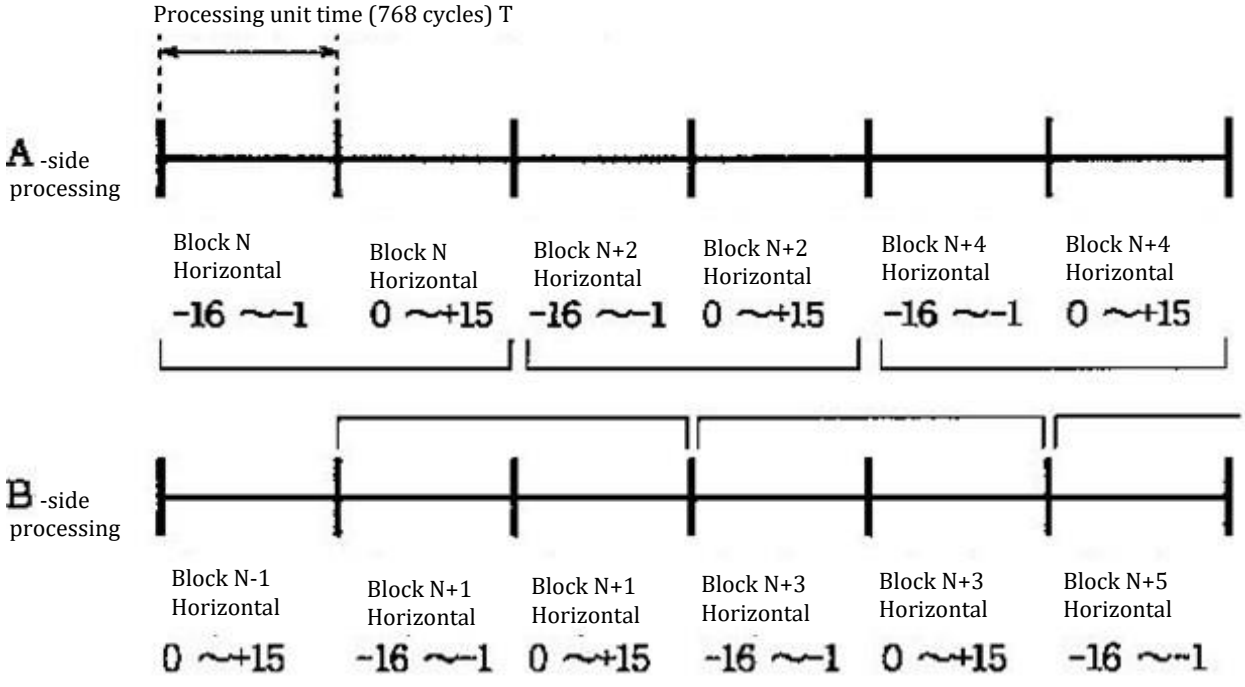
[FIG. 44]


1a
3a
630
632
Search-window data PY
PXA
A-side computation unit
eva
evb
evc
Comparison unit
MVA
Input unit
PY
Selector
MVB
1b
3b
Template data PX
PYB
B-side computation unit
eva
evb
evc
Comparison unit
MVC

[FIG. 45]

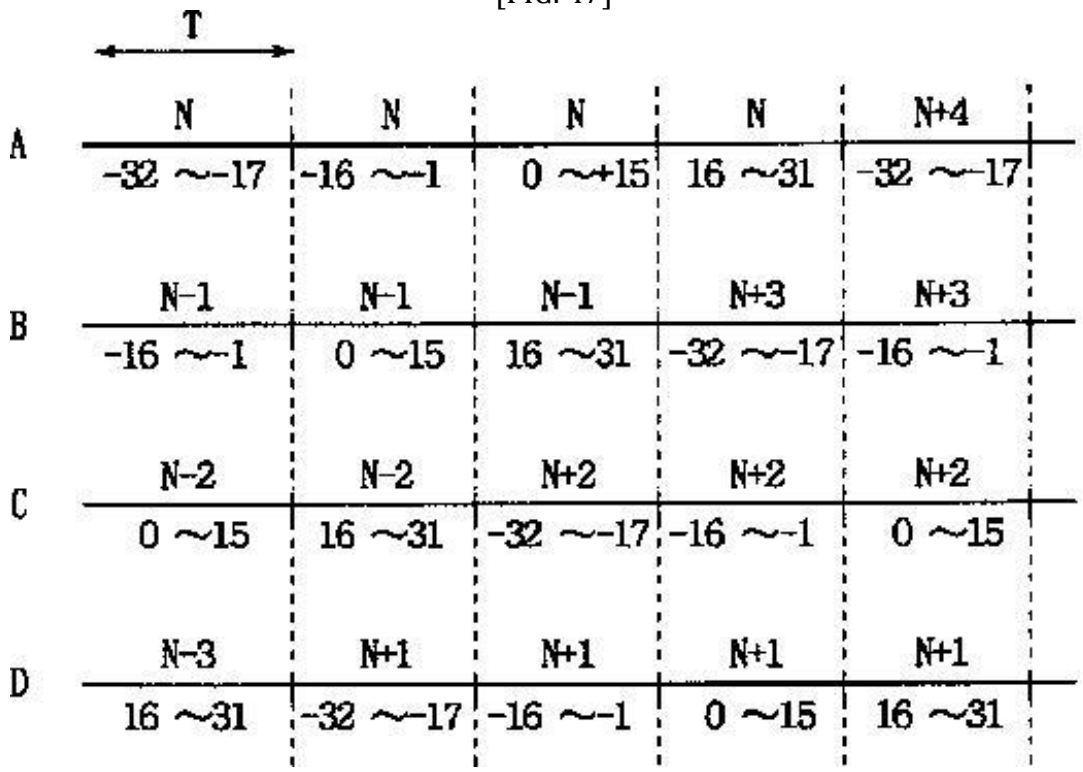


[FIG. 46]

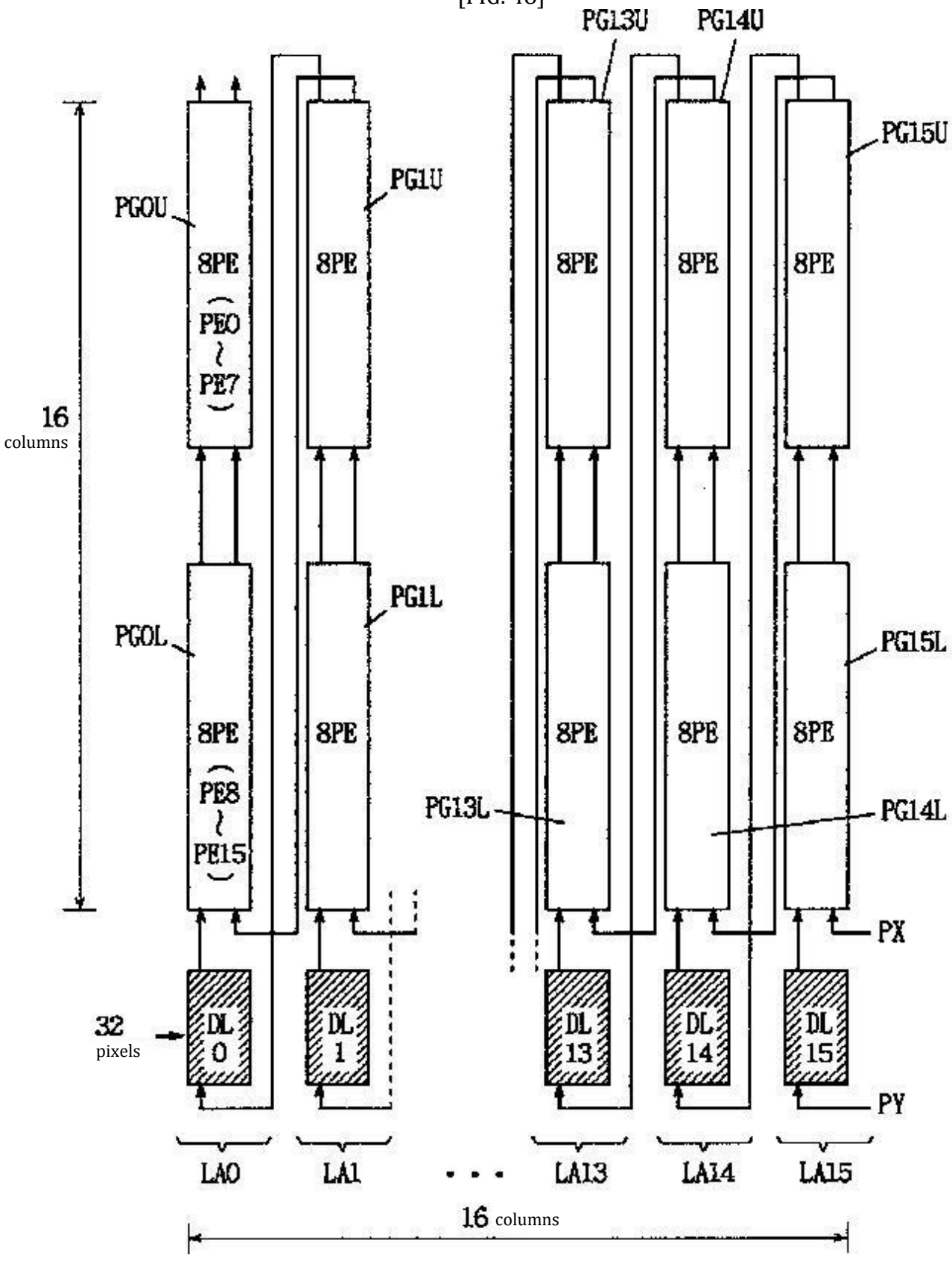


: Search-window pixel data in common

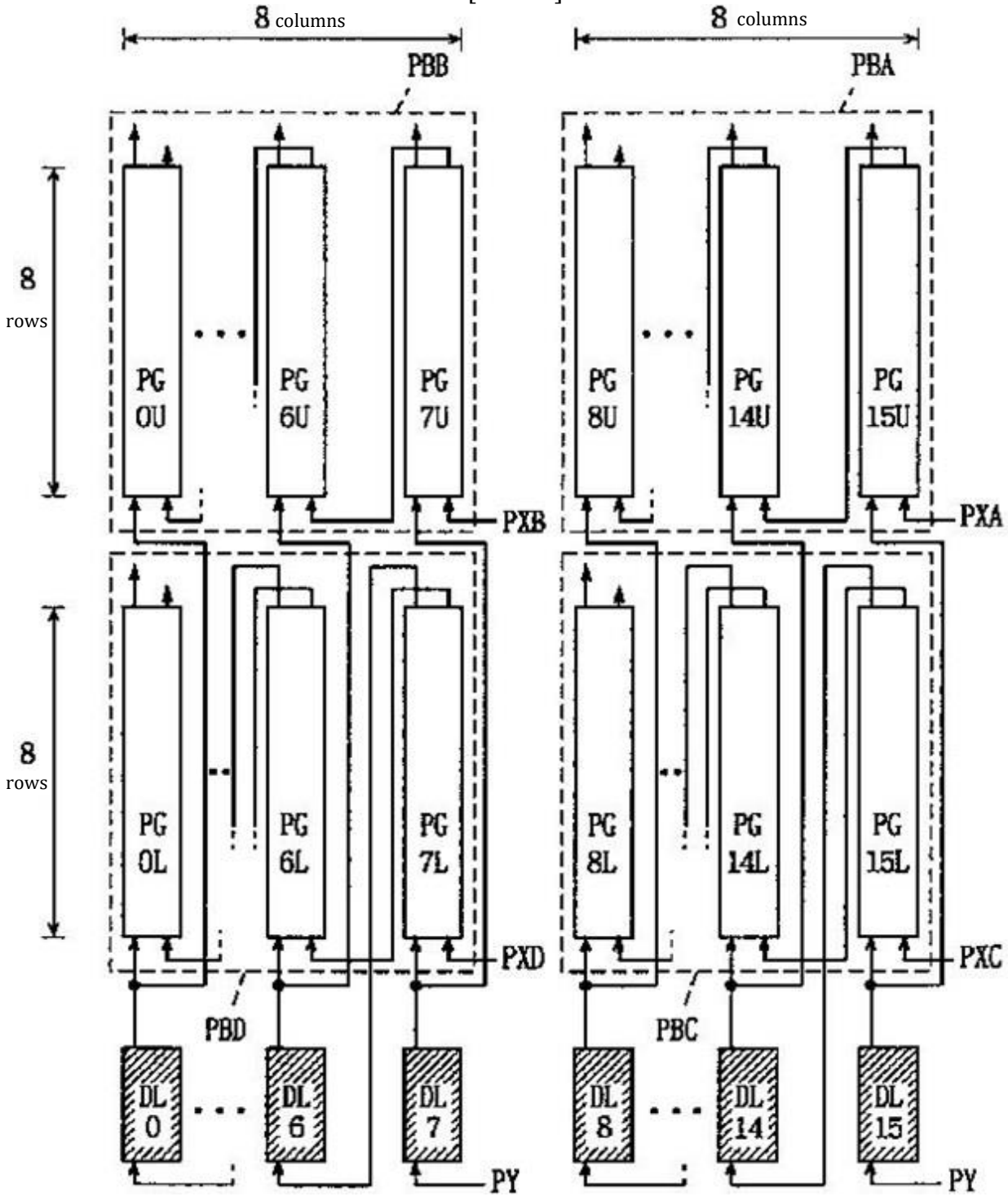
[FIG. 47]


T
A
N
N
N
N
N+4
-32 ~-17
-16 ~-1
0 ~+15
16 ~31
-32 ~-17
B
N-1
N-1
N-1
N+3
N+3
-16 ~-1
0 ~15
16 ~31
-32 ~-17
-16 ~-1
C
N-2
N-2
N+2
N+2
N+2
0 ~15
16 ~31
-32 ~-17
-16 ~-1
0 ~15
D
N-3
N+1
N+1
N+1
N+1
16 ~31
-32 ~-17
-16 ~-1
0 ~15
16 ~31

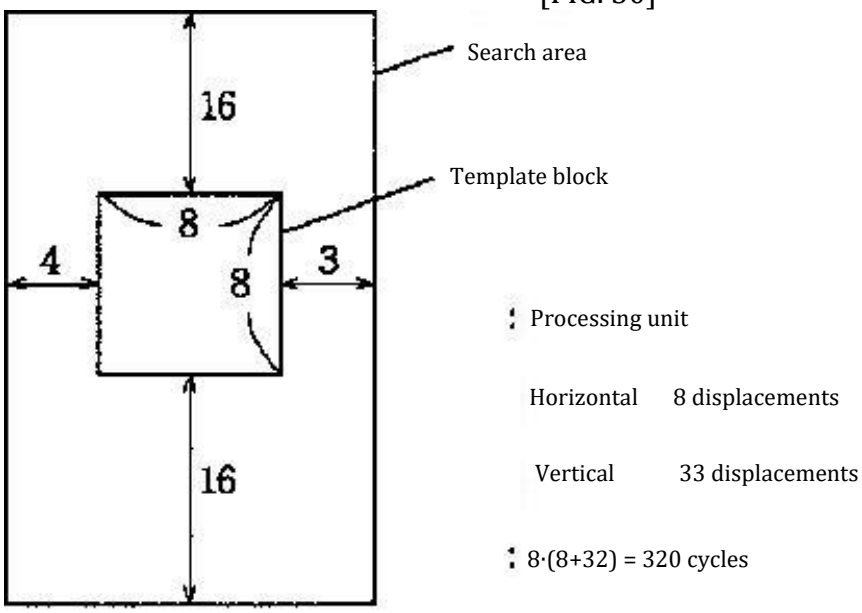
[FIG. 48]


PG13U
PG14U
PG15U
PG0U
PG1U
8PE
PE0
~
PE7
16
columns
PG1L
PG0L
PG15L
PE8
PE15
PG13L
PG14L
PX
32
pixels
DL
0
DL
1
DL
13
DL
14
DL
15
PY
LA0
LA1
LA13
LA14
LA15
16 columns

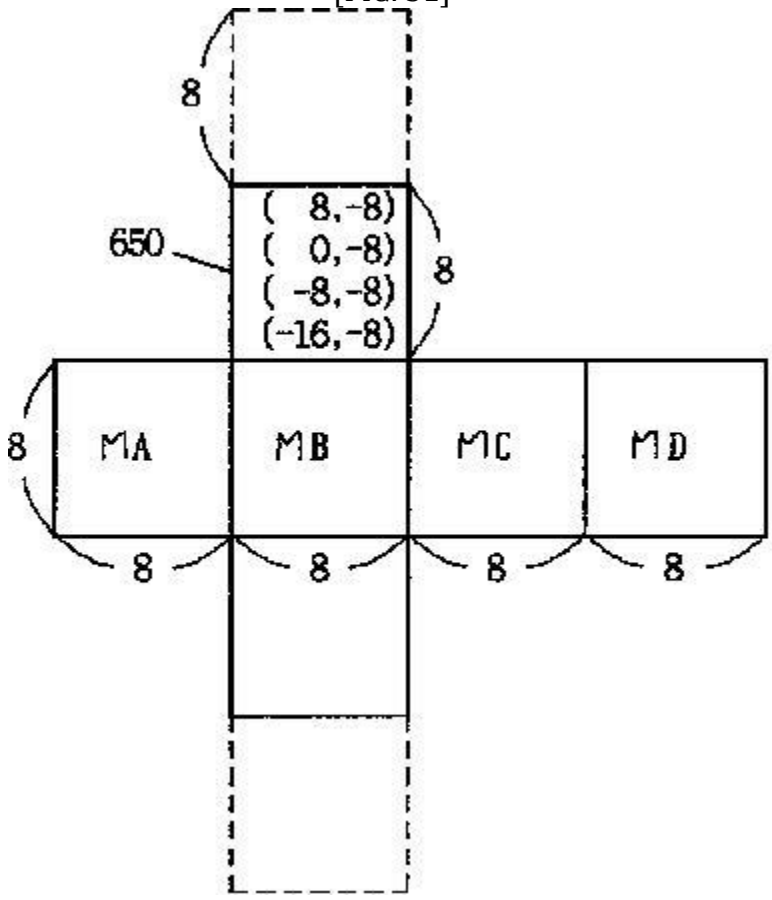
[FIG. 49]


8 columns
8 columns
PBB
PBA
8 rows
8 rows
PG 0U
PG 6U
PG 7U
PG 8U
PG 14U
PG 15U
PXB
PXA
PG 0L
PG 6L
PG 7L
PG 8L
PG 14L
PG 15L
PXD
PXC
PBD
PBC
DL 0
DL 6
DL 7
DL 8
DL 14
DL 15
PY
PY

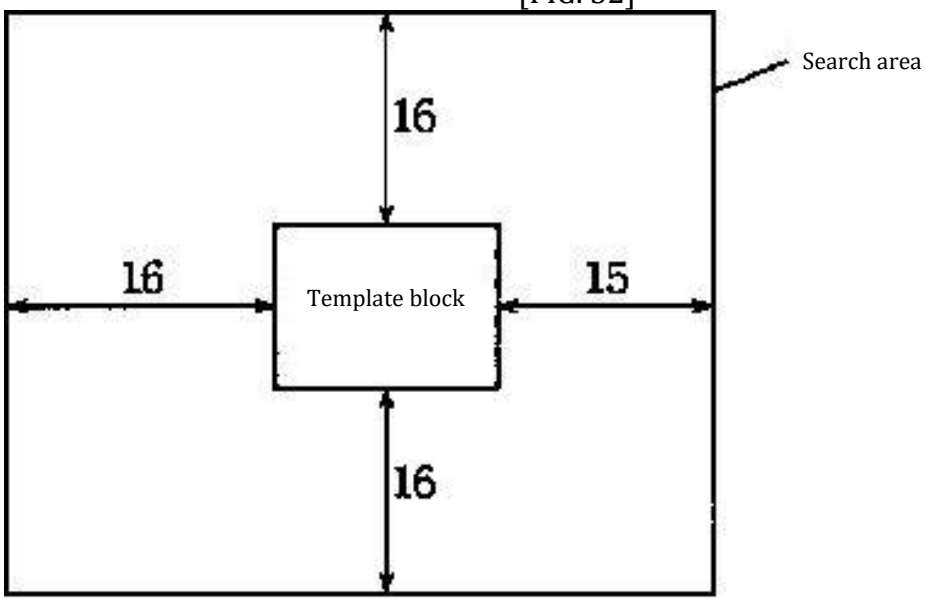
[FIG. 50]


Search area
16
Template block
8
4
8
3
16
: Processing unit
Horizontal 8 displacements
Vertical 33 displacements
: 8·(8+32) = 320 cycles

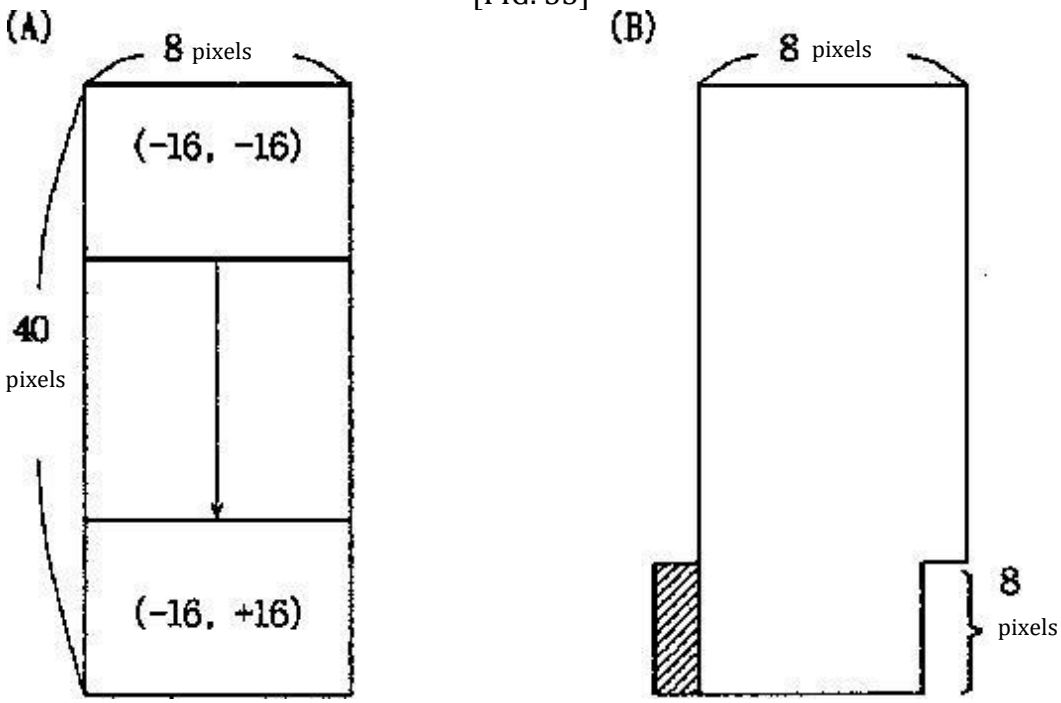
[FIG. 51]


8
650
(8,-8)
(0,-8)
(-8,-8)
(-16,-8)
8
8
MA
MB
MC
MD
8
8
8
8

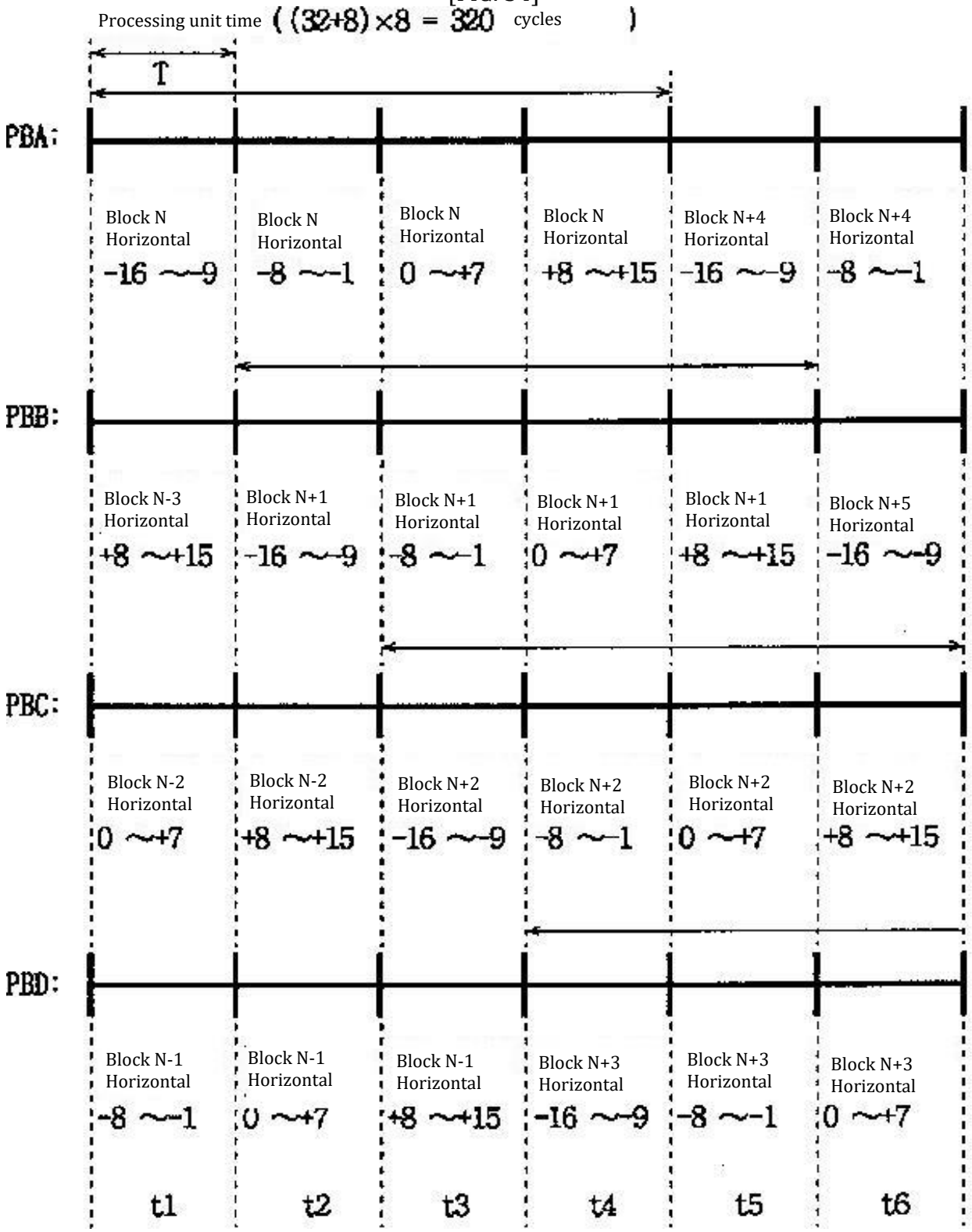
[FIG. 52]


Search area
16
16
Template block
15
16

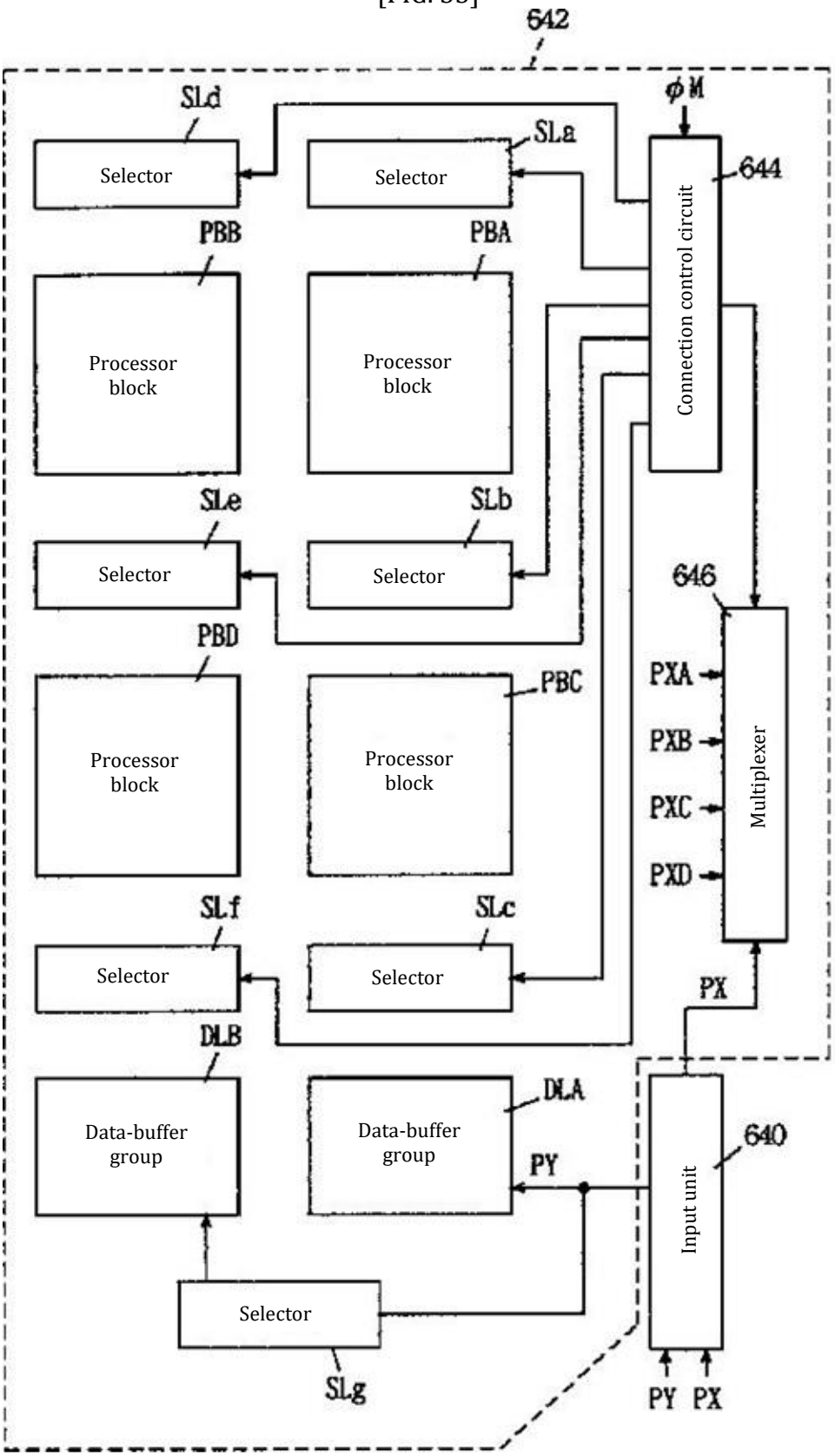
[FIG. 53]


(A)
8 pixels
(-16, -16)
40
pixels
(-16, +16)
(B)
8 pixels
8
pixels

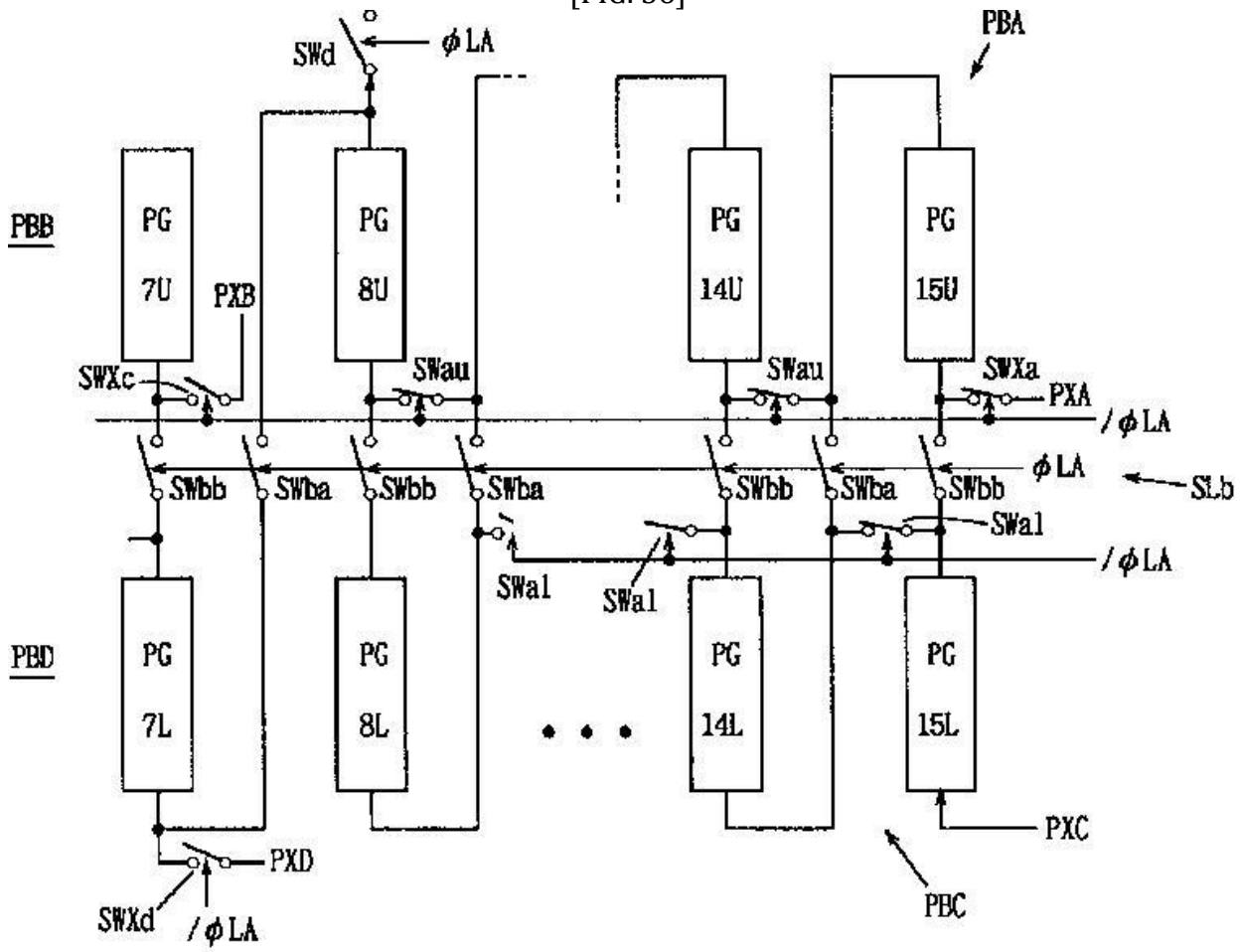
[FIG. 54]


Processing unit time ((32+8) ×8 = 320 cycles)
T
PBA:
Block N Horizontal -16 ~-9
Block N Horizontal -8 ~-1
Block N Horizontal 0 ~+7
Block N Horizontal +8 ~+15
Block N+4 Horizontal -16 ~-9
Block N+4 Horizontal -8 ~-1
PBB:
Block N-3 Horizontal +8 ~+15
Block N+1 Horizontal -16 ~-9
Block N+1 Horizontal -8 ~-1
Block N+1 Horizontal 0 ~+7
Block N+1 Horizontal +8 ~+15
Block N+5 Horizontal -16 ~-9
PBC:
Block N-2 Horizontal 0 ~+7
Block N-2 Horizontal +8 ~+15
Block N+2 Horizontal -16 ~-9
Block N+2 Horizontal -8 ~-1
Block N+2 Horizontal 0 ~+7
Block N+2 Horizontal +8 ~+15
PBD:
Block N-1 Horizontal -8 ~-1
Block N-1 Horizontal 0 ~+7
Block N-1 Horizontal +8 ~+15
Block N+3 Horizontal -16 ~-9
Block N+3 Horizontal -8 ~-1
Block N+3 Horizontal 0 ~+7
t1
t2
t3
t4
t5
t6

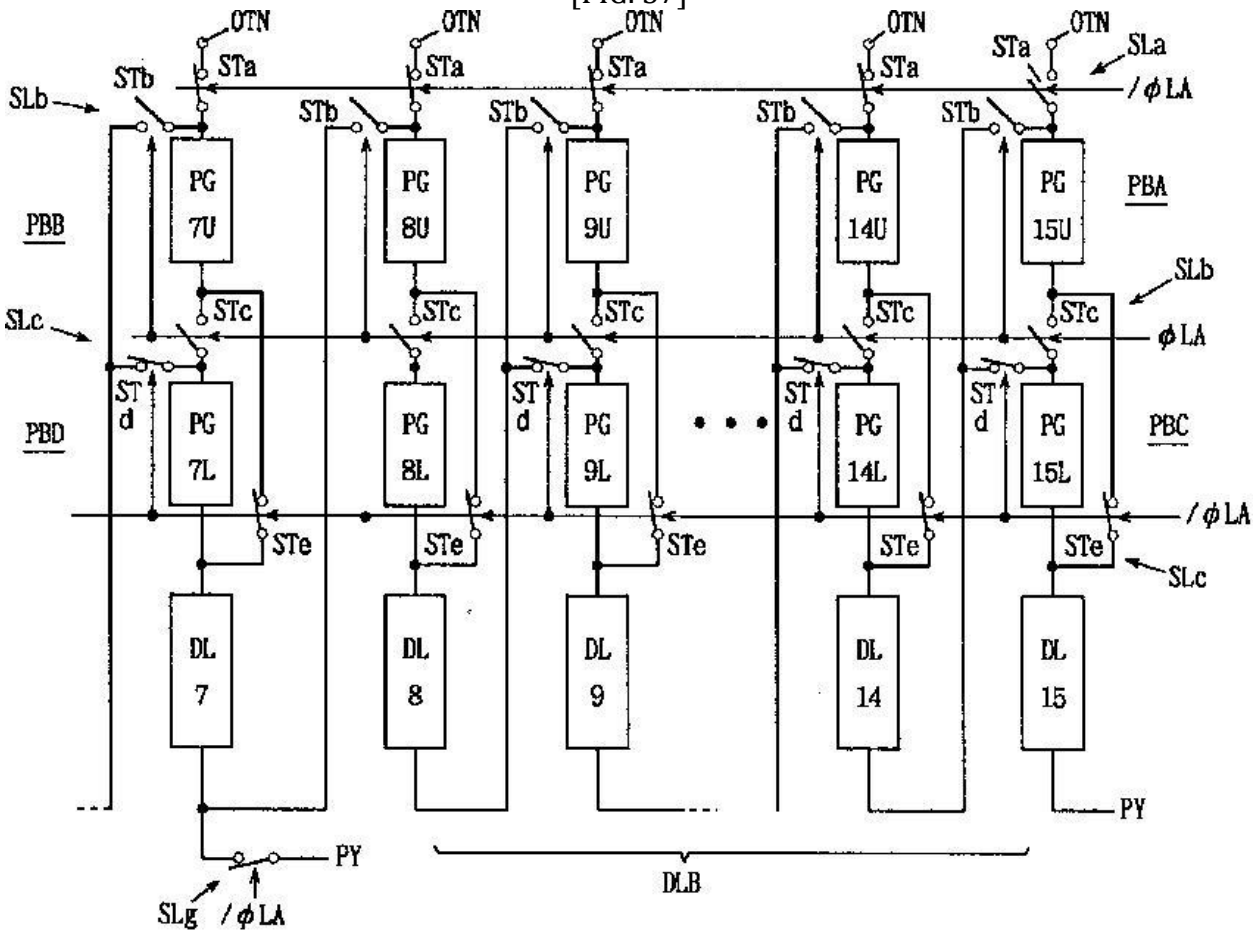
[FIG. 55]


642
SLd
Selector
SLa
Selector
φM
644
Connection control circuit
PBB
Processor block
PBA
Processor block
SLe
Selector
SLb
Selector
646
PBD
Processor block
PBC
Processor block
PXA
PXB
PXC
PXD
Multiplexer
SLf
Selector
SLc
Selector
PX
DLB
Data-buffer group
DLA
Data-buffer group
PY
640
Input unit
Selector
SLg
PY
PX

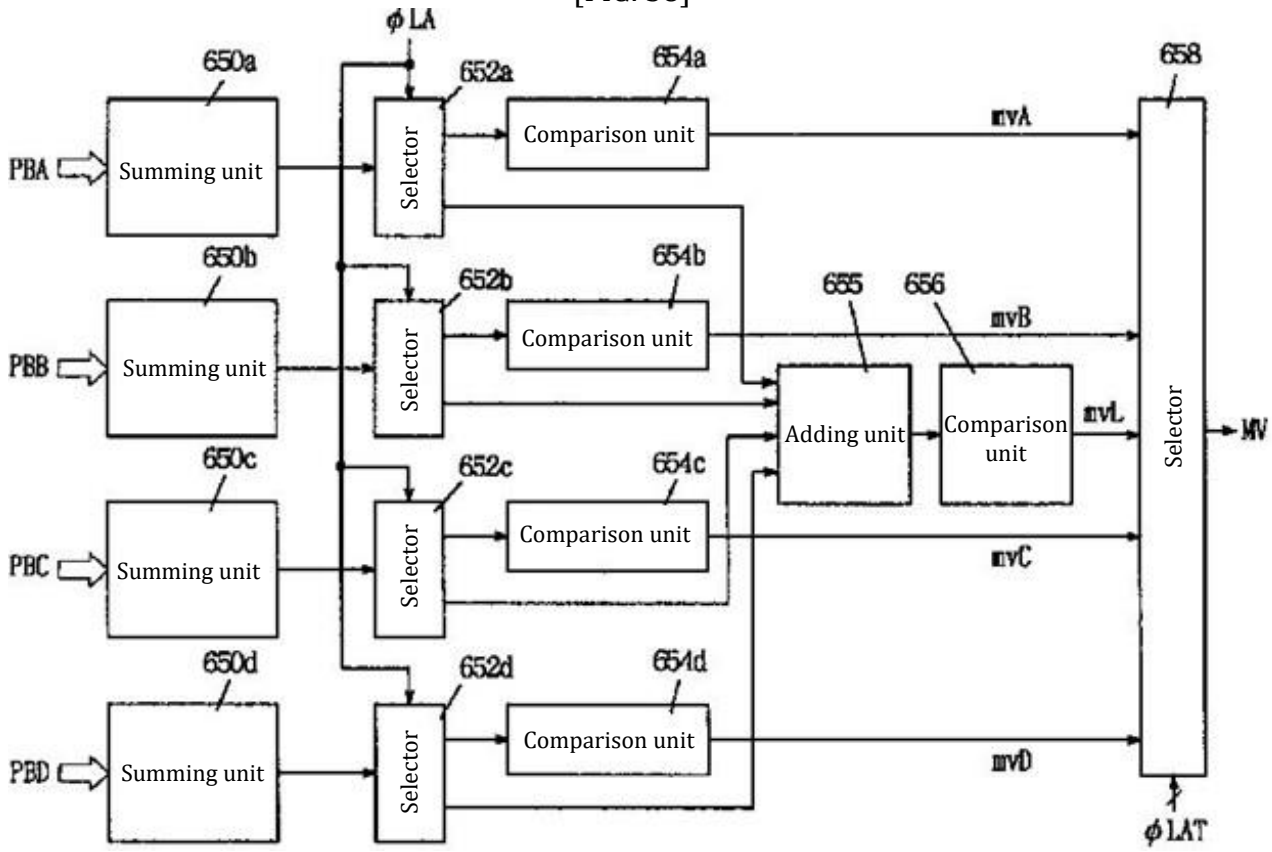
[FIG. 56]


SWd
φLA
PBA
PBB
PG 7U
PG 8U
PG 14U
PG 15U
PXB
SWXc
SWau
SWXa
PXA
/φLA
φLA
SLb
SWbb
SWba
SWal
PBD
PG 7L
PG 8L
PG 14L
PG 15L
PXC
PXD
SWXd
PBC

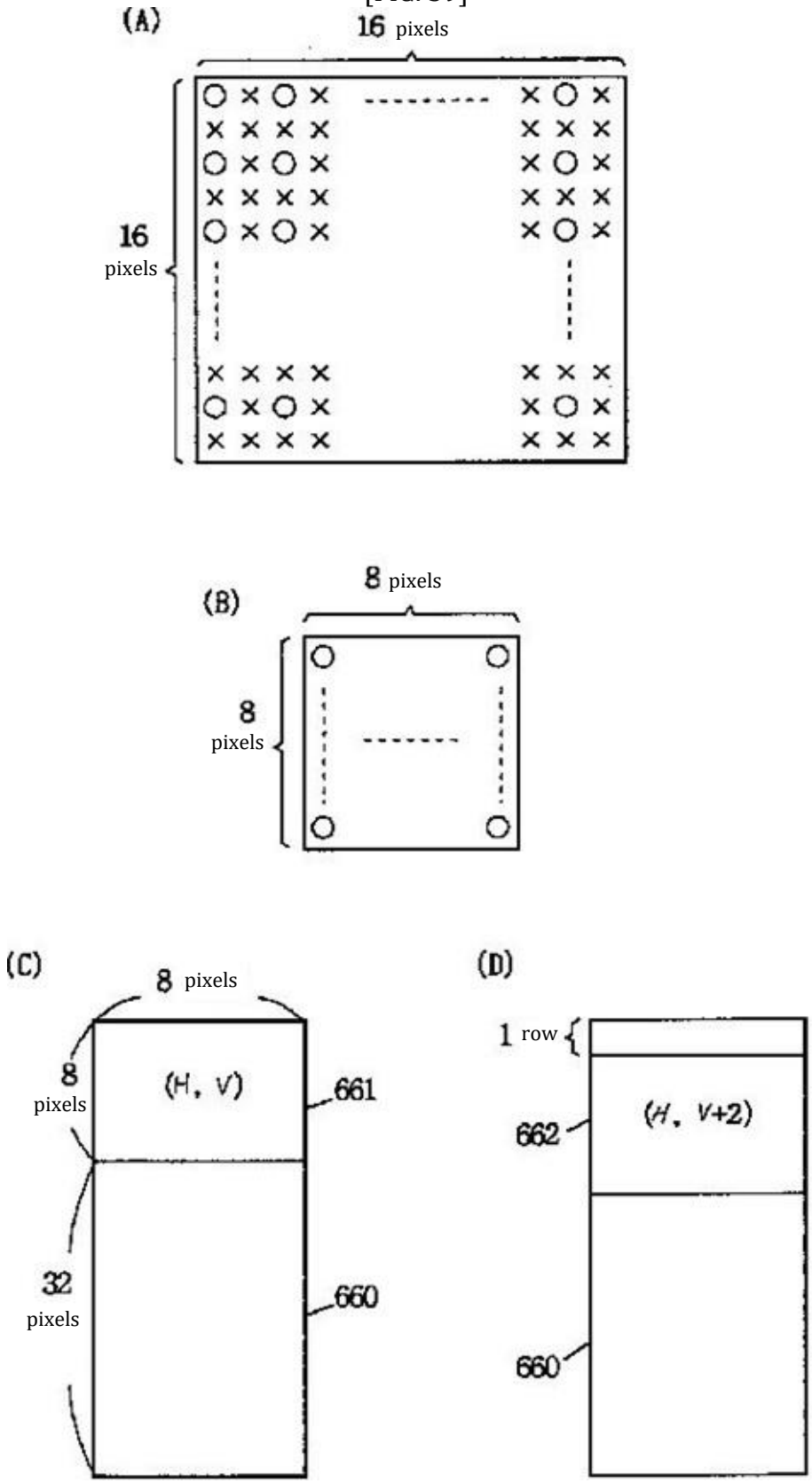
[FIG. 57]


OTN
STa
STb
STc
STd
STe
SLa
SLb
SLc
/φLA
φLA
PBA
PBB
PBC
PBD
PG 7U
PG 8U
PG 9U
PG 14U
PG 15U
PG 7L
PG 8L
PG 9L
PG 14L
PG 15L
DL 7
DL 8
DL 9
DL 14
DL 15
PY
DLB
SLg

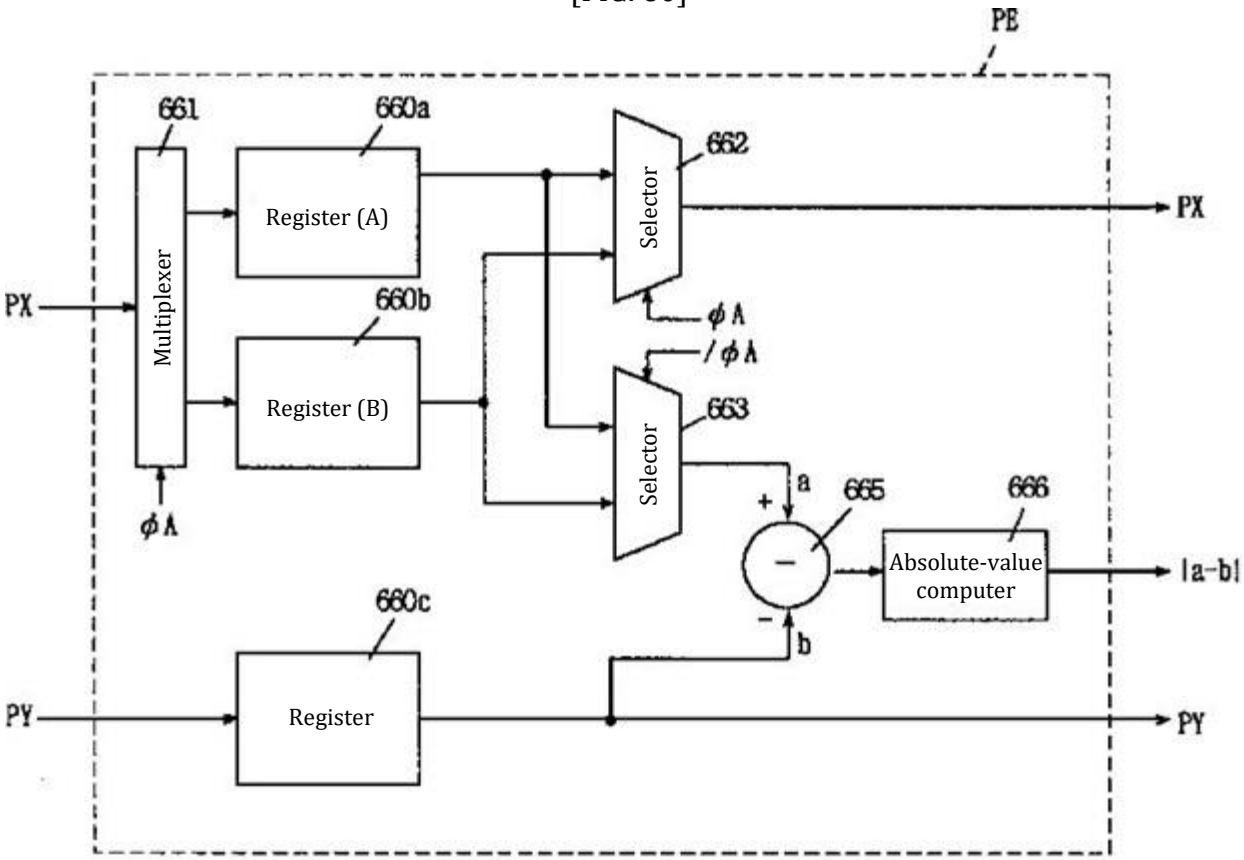
[FIG. 58]


φLA
650a
652a
654a
658
PBA
Summing unit
Selector
Comparison unit
mvA
650b
652b
654b
655
656
PBB
Summing unit
Selector
Comparison unit
mvB
Adding unit
Comparison unit
mvL
Selector
MV
650c
652c
654c
PBC
Summing unit
Selector
Comparison unit
mvC
650d
652d
654d
PBD
Summing unit
Selector
Comparison unit
mvD
φLAT

[FIG. 59]


(A)
16 pixels
16 pixels
(B)
8 pixels
8 pixels
(C)
8 pixels
8 pixels
(H, V)
661
32 pixels
660
(D)
1 row
(H, V+2)
662
660

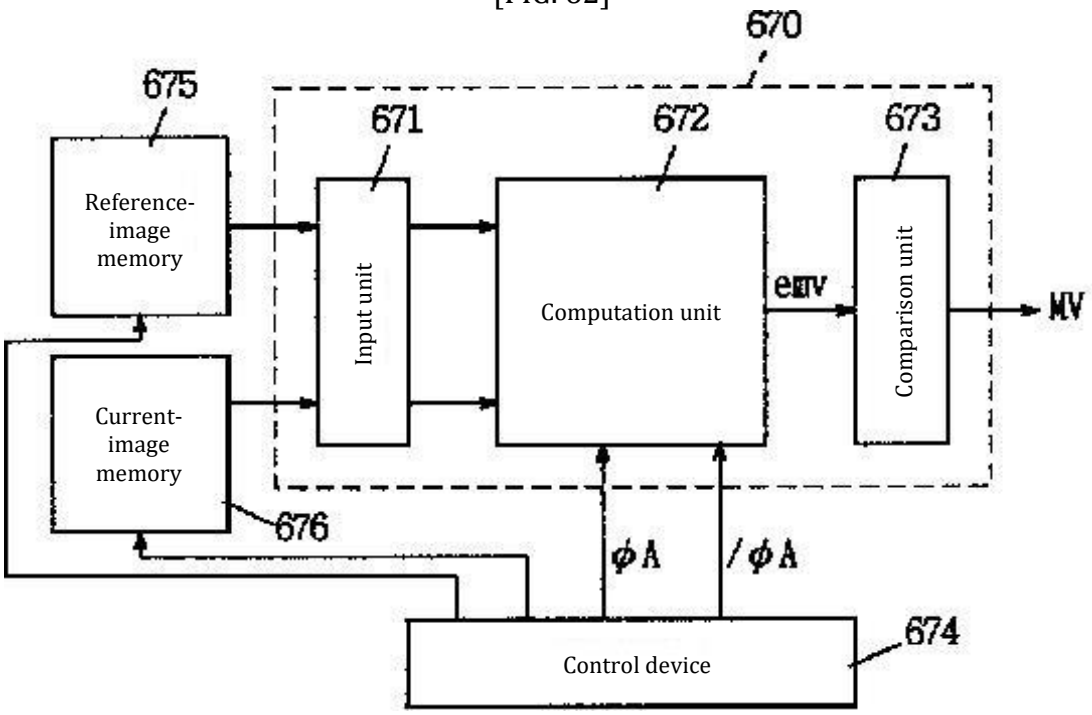
[FIG. 60]



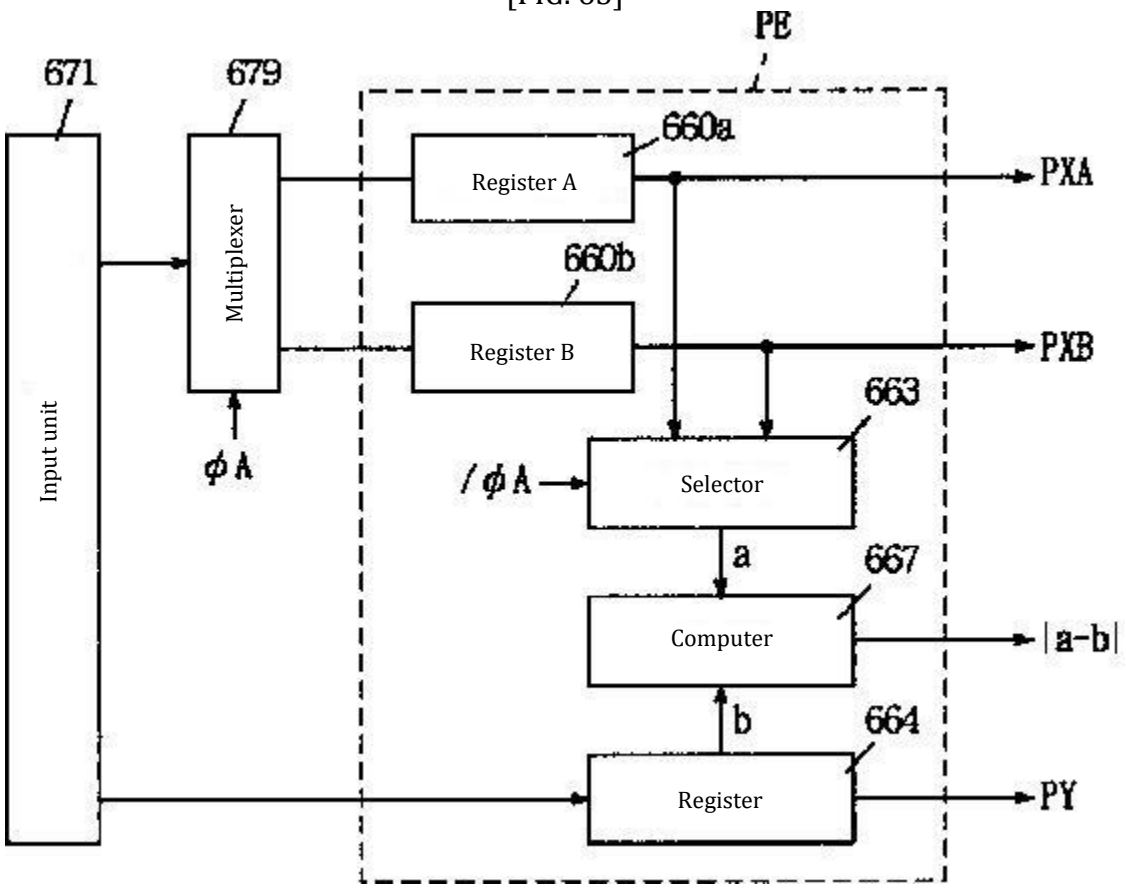
[FIG. 61]

	T1	T2	T3	T4	T5	T6
Register (A)	Search template block N	Load template block N+2	Search template block N+2	Load template block N+4	Search template block N+4	Load template block N+6
Register (B)	Load template block N+1	Search template block N+1	Load template block N+3	Search template block N+3	Load template block N+5	Search template block N+5

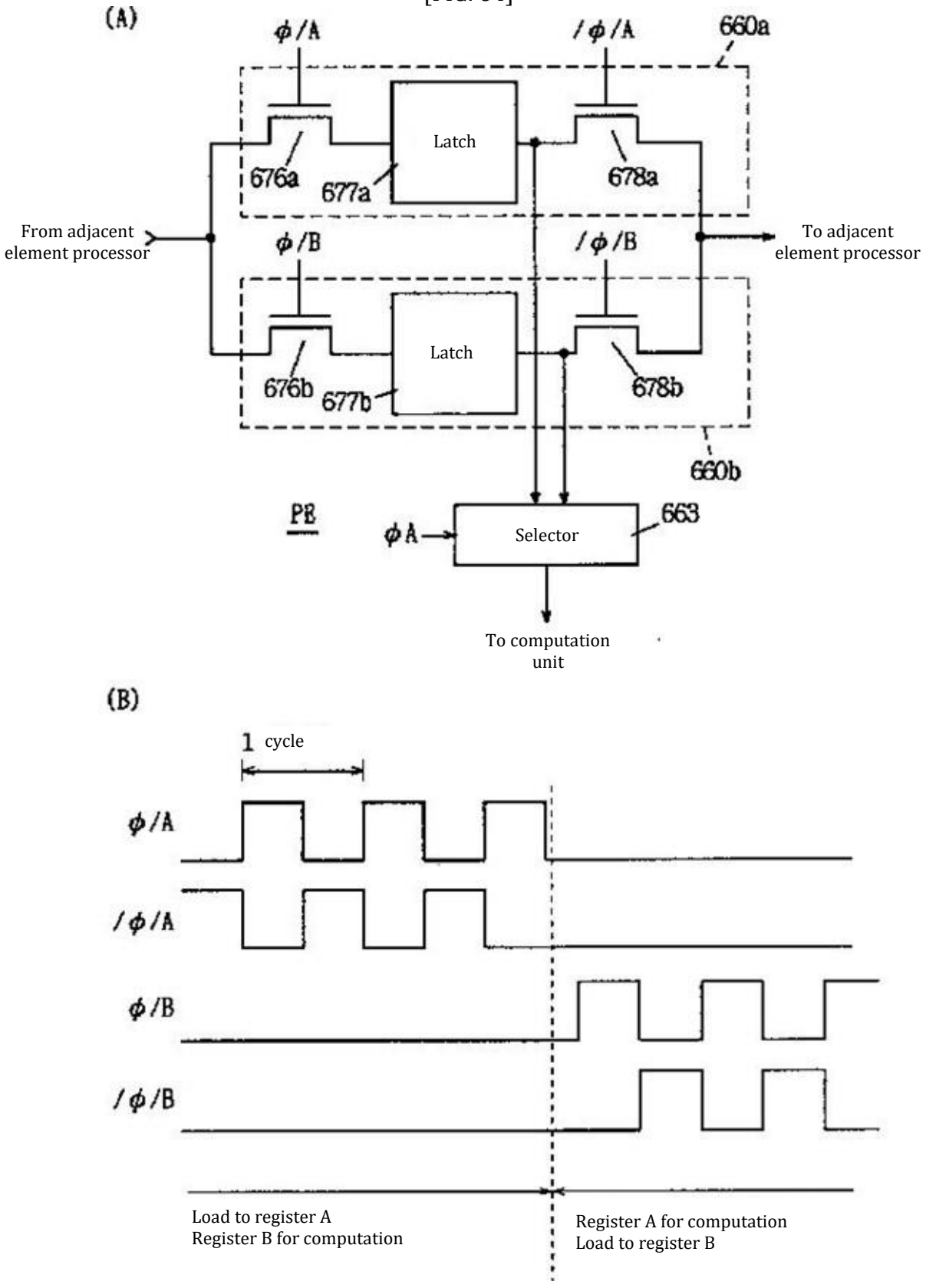
[FIG. 62]


670
675
671
672
673
Reference-image memory
Input unit
Computation unit
env
Comparison unit
MV
Current-image memory
676
φA
/φA
674
Control device

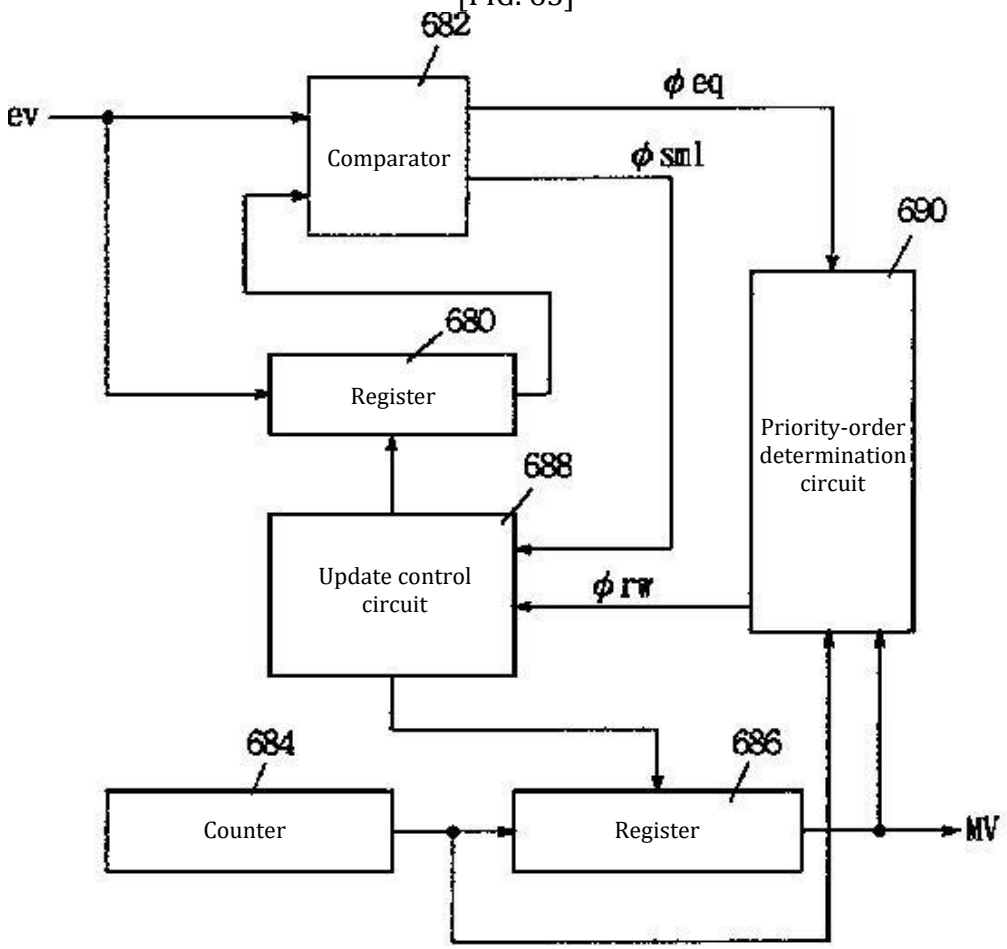
[FIG. 63]


PE
671
679
660a
Register A
PXA
660b
Register B
PXB
Input unit
Multiplexer
φA
663
/φA
Selector
a
667
Computer
|a-b|
b
664
Register
PY

[FIG. 64]


(A)
φ/A
/φ/A
660a
Latch
676a
677a
678a
From adjacent element processor
φ/B
/φ/B
To adjacent element processor
Latch
676b
677b
678b
660b
PE
φA
Selector
663
To computation unit
(B)
1 cycle
φ/A
/φ/A
φ/B
/φ/B
Load to register A
Register B for computation
Register A for computation
Load to register B

[FIG. 65]


682
ev
Comparator
φ eq
φ sml
690
680
Register
Priority-order determination circuit
688
Update control circuit
φ rw
684
686
Counter
Register
MV

[FIG. 66]

$P(H,V) = |H|+|V|$

H vector

V vector	-16	-15	-14	-13	-12	-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1	0
-16	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
-15	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15
-14	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14
-13	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13
-12	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12
-11	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11
-10	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10
-9	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9
-8	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8
-7	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7
-6	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6
-5	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5
-4	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4
-3	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3
-2	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2
-1	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
0	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
2	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2
3	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3
4	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4
5	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5
6	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6
7	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7
8	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8
9	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9
10	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10
11	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11
12	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12
13	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13
14	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14
15	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15
16	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16

[FIG. 67]

$$P(H,V) = |H|+|V|$$

V \	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
-16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	32
-15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	31
-14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	30
-13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	29
-12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	28
-11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	27
-10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	26
-9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	25
-8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	24
-7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	23
-6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	22
-5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	21
-4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	20
-3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	19
-2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	18
-1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	17
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	16
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	17
2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	18
3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	19
4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	20
5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	21
6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	22
7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	23
8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	24
9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	25
10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	26
11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	27
12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	28
13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	29
14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	30
15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	31
16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	32

[FIG. 68]

$$P(H,V) = |Hi+(-V+16)$$

H vector

V vector	-16	-15	-14	-13	-12	-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1	0
-16	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
-15	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32	31
-14	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32	31	30
-13	45	44	43	42	41	40	39	38	37	36	35	34	33	32	31	30	29
-12	44	43	42	41	40	39	38	37	36	35	34	33	32	31	30	29	28
-11	43	42	41	40	39	38	37	36	35	34	33	32	31	30	29	28	27
-10	42	41	40	39	38	37	36	35	34	33	32	31	30	29	28	27	26
-9	41	40	39	38	37	36	35	34	33	32	31	30	29	28	27	26	25
-8	40	39	38	37	36	35	34	33	32	31	30	29	28	27	26	25	24
-7	39	38	37	36	35	34	33	32	31	30	29	28	27	26	25	24	26
-6	38	37	36	35	34	33	32	31	30	29	28	27	26	25	24	26	22
-5	37	36	35	34	33	32	31	30	29	28	27	26	25	24	26	22	21
-4	36	35	34	33	32	31	30	29	28	27	26	25	24	26	22	21	20
-3	35	34	33	32	31	30	29	28	27	26	25	24	26	22	21	20	19
-2	34	33	32	31	30	29	28	27	26	25	24	26	22	21	20	19	18
-1	33	32	31	30	29	28	27	26	25	24	26	22	21	20	19	18	17
0	32	31	30	29	28	27	26	25	24	26	22	21	20	19	18	17	16
1	31	30	29	28	27	26	25	24	26	22	21	20	19	18	17	16	15
2	30	29	28	27	26	25	24	26	22	21	20	19	18	17	16	15	14
3	29	28	27	26	25	24	26	22	21	20	19	18	17	16	15	14	13
4	28	27	26	25	24	26	22	21	20	19	18	17	16	15	14	13	12
5	27	26	25	24	26	22	21	20	19	18	17	16	15	14	13	12	11
6	26	25	24	26	22	21	20	19	18	17	16	15	14	13	12	11	10
7	25	24	26	22	21	20	19	18	17	16	15	14	13	12	11	10	9
8	24	26	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8
9	26	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7
10	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6
11	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5
12	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4
13	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3
14	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2
15	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
16	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

[FIG. 69]

$$P(H,V) = |H|+(-V+16)$$

V \	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
-16	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47
-15	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46
-14	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45
-13	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44
-12	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43
-11	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42
-10	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41
-9	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40
-8	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39
-7	26	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38
-6	22	26	24	25	26	27	28	29	30	31	32	33	34	35	36	37
-5	21	22	26	24	25	26	27	28	29	30	31	32	33	34	35	36
-4	20	21	22	26	24	25	26	27	28	29	30	31	32	33	34	35
-3	19	20	21	22	26	24	25	26	27	28	29	30	31	32	33	34
-2	18	19	20	21	22	26	24	25	26	27	28	29	30	31	32	33
-1	17	18	19	20	21	22	26	24	25	26	27	28	29	30	31	32
0	16	17	18	19	20	21	22	26	24	25	26	27	28	29	30	31
1	15	16	17	18	19	20	21	22	26	24	25	26	27	28	29	30
2	14	15	16	17	18	19	20	21	22	26	24	25	26	27	28	29
3	13	14	15	16	17	18	19	20	21	22	26	24	25	26	27	28
4	12	13	14	15	16	17	18	19	20	21	22	26	24	25	26	27
5	11	12	13	14	15	16	17	18	19	20	21	22	26	24	25	26
6	10	11	12	13	14	15	16	17	18	19	20	21	22	26	24	25
7	9	10	11	12	13	14	15	16	17	18	19	20	21	22	26	24
8	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	26
9	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
10	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
11	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
12	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
13	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
14	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
15	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
16	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

[FIG. 70]

H vector

V vector	-16	-15	-14	-13	-12	-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1	0
-16	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-15	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
-14	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2
-13	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3
-12	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4
-11	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5
-10	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6
-9	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7
-8	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8
-7	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9
-6	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10
-5	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11
-4	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12
-3	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13
-2	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14
-1	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15
0	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
1	33	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17
2	34	33	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18
3	35	34	33	32	31	30	29	28	27	26	25	24	23	22	21	20	19
4	36	35	34	33	32	31	30	29	28	27	26	25	24	23	22	21	20
5	37	36	35	34	33	32	31	30	29	28	27	26	25	24	23	22	21
6	38	37	36	35	34	33	32	31	30	29	28	27	26	25	24	23	22
7	39	38	37	36	35	34	33	32	31	30	29	28	27	26	25	24	23
8	40	39	38	37	36	35	34	33	32	31	30	29	28	27	26	25	24
9	41	40	39	38	37	36	35	34	33	32	31	30	29	28	27	26	25
10	42	41	40	39	38	37	36	35	34	33	32	31	30	29	28	27	26
11	43	42	41	40	39	38	37	36	35	34	33	32	31	30	29	28	27
12	44	43	42	41	40	39	38	37	36	35	34	33	32	31	30	29	28
13	45	44	43	42	41	40	39	38	37	36	35	34	33	32	31	30	29
14	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32	31	30
15	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32	31
16	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32

$$P(H,V) = |H| + (+V+16)$$

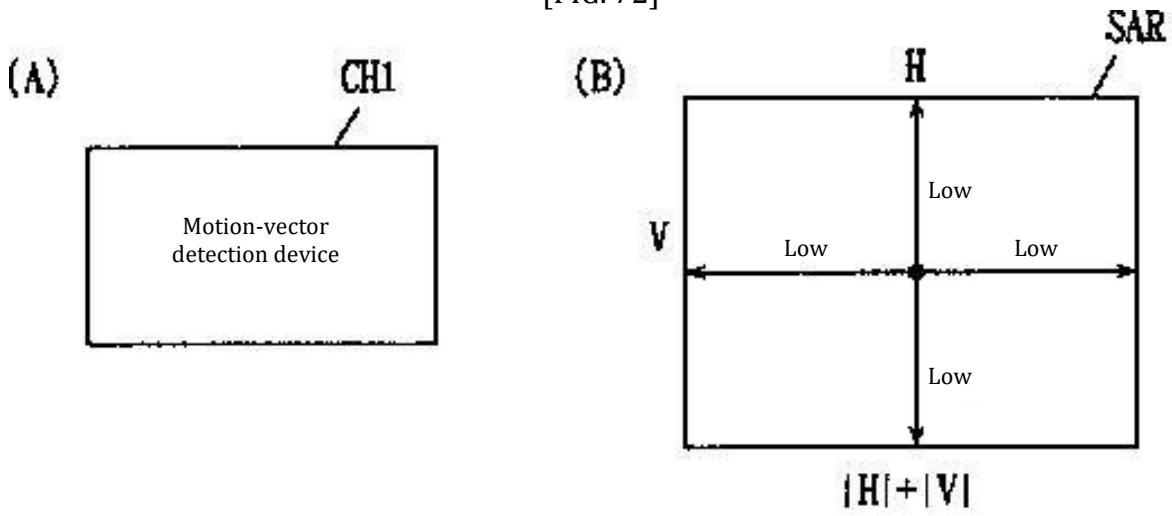
[FIG. 71]

H vector

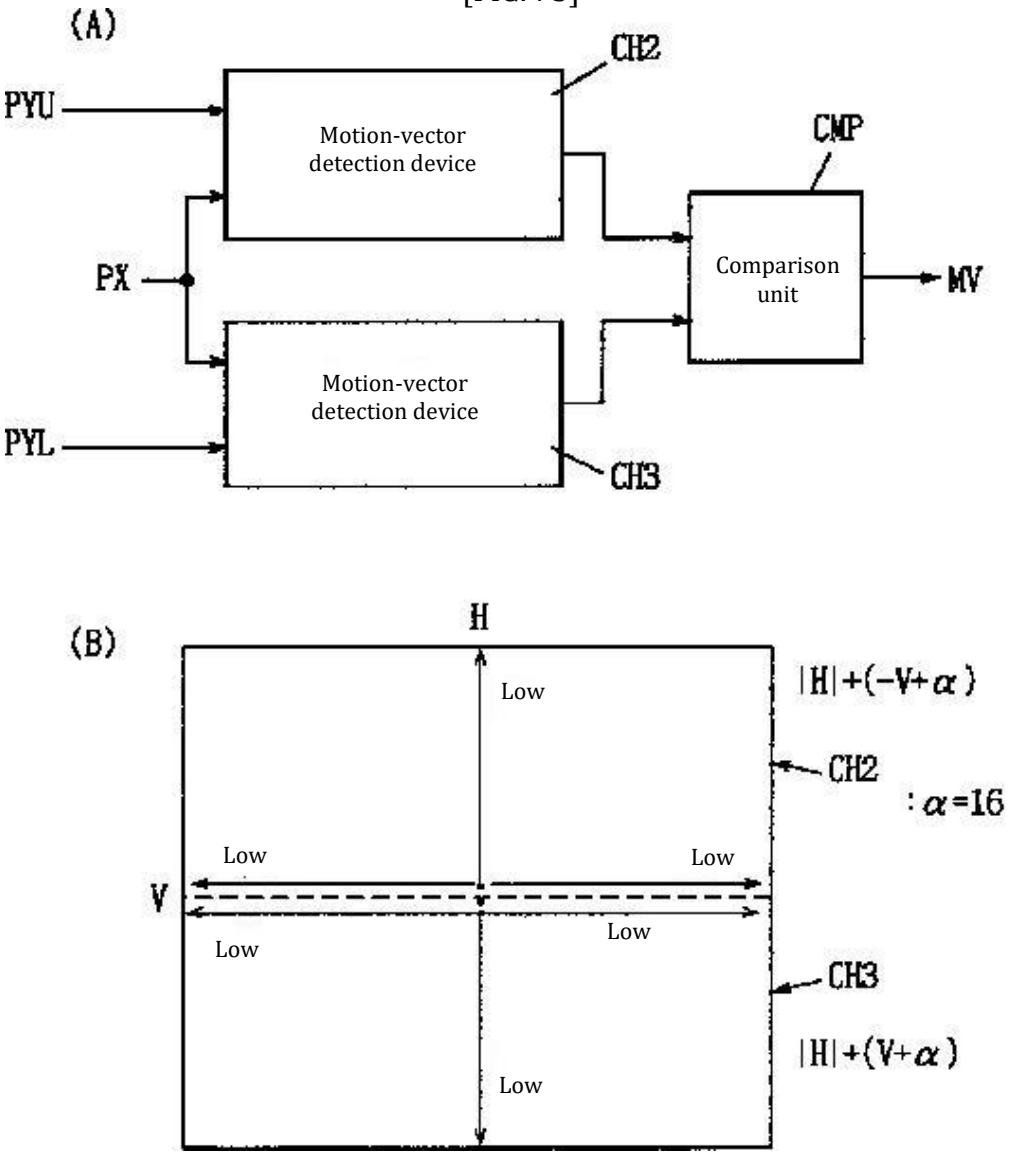
V \	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
-16	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
-15	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
-14	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
-13	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
-12	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
-11	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
-10	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
-9	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
-8	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
-7	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
-6	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
-5	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26
-4	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27
-3	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28
-2	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29
-1	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
0	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
1	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
2	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33
3	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34
4	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35
5	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36
6	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37
7	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38
8	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39
9	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40
10	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41
11	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42
12	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43
13	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44
14	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45
15	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46
16	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47

$$P(H,V) = |H| + (+V+16)$$

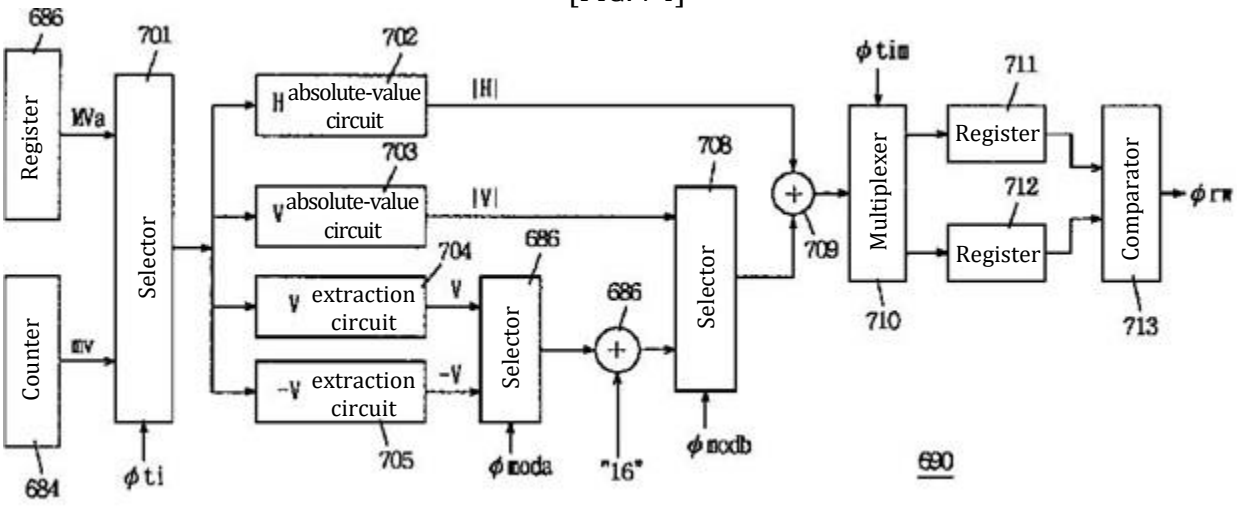
[FIG. 72]


(A)
CH1
Motion-vector
detection device
(B)
H
SAR
Low
V
Low
Low
Low
|H|+|V|

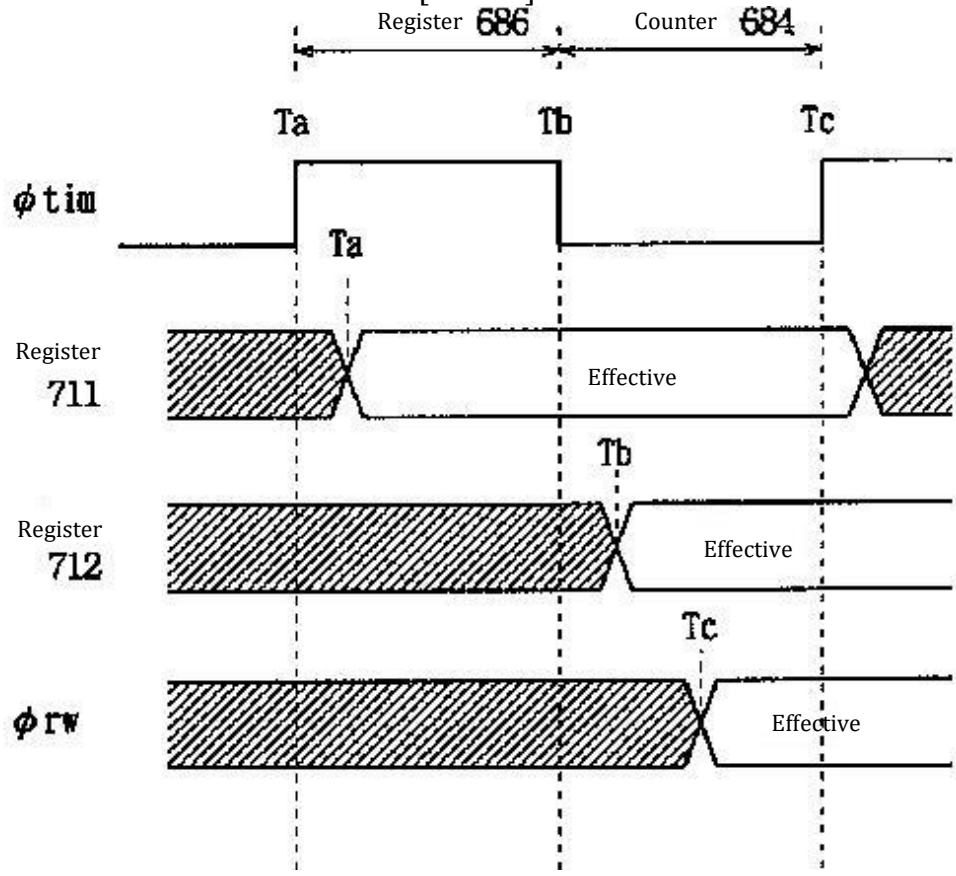
[FIG. 73]


(A)
PYU
PX
PYL
Motion-vector detection device
Motion-vector detection device
CH2
CH3
CMP
Comparison unit
MV
(B)
H
V
Low
Low
Low
Low
Low
|H|+(-V+α)
CH2
:α=16
CH3
|H|+(V+α)

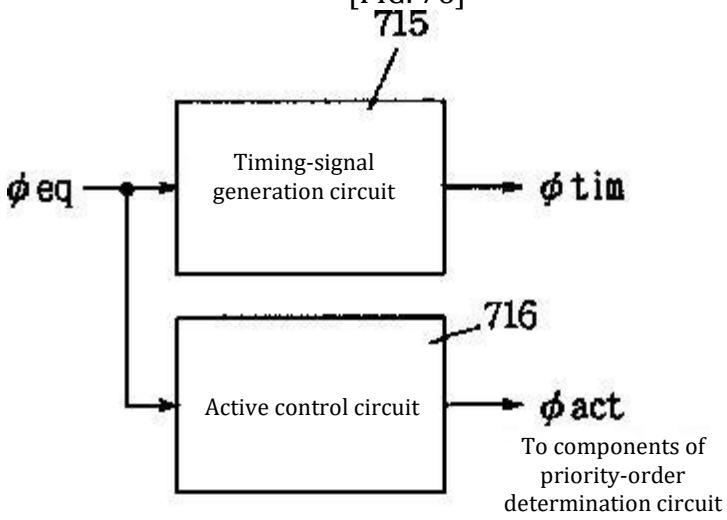
[FIG. 74]


686
Register
Wa
701
Selector
Counter
mv
684
φti
H absolute-value circuit
V absolute-value circuit
V extraction circuit
-V extraction circuit
702
703
704
705
|H|
|V|
V
-V
686
Selector
φmoda
"16"
686
708
Selector
φmodb
709
φtim
Multiplexer
710
711
Register
712
Register
Comparator
713
φrw
690

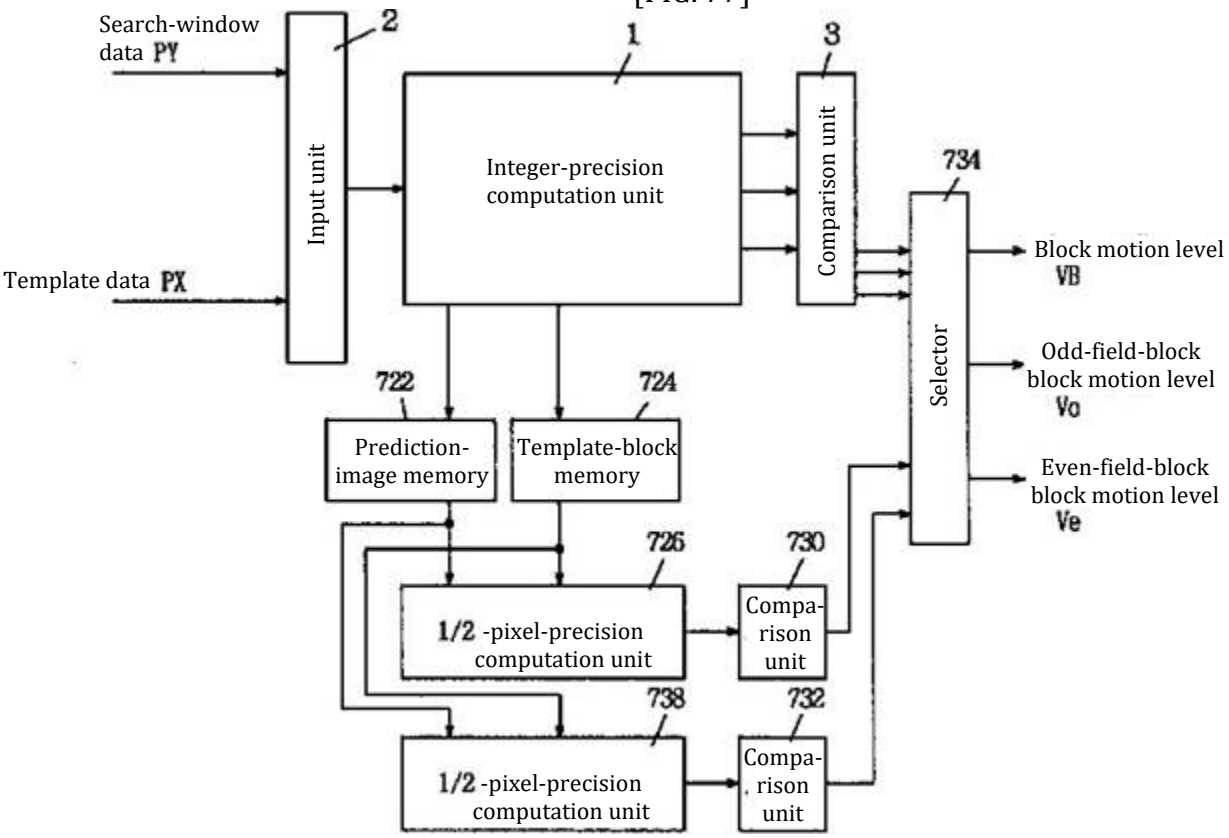
[FIG. 75]


Register 686
Counter 684
Ta
Tb
Tc
φtim
Ta
Register
711
Effective
Tb
Register
712
Effective
Tc
φrw
Effective

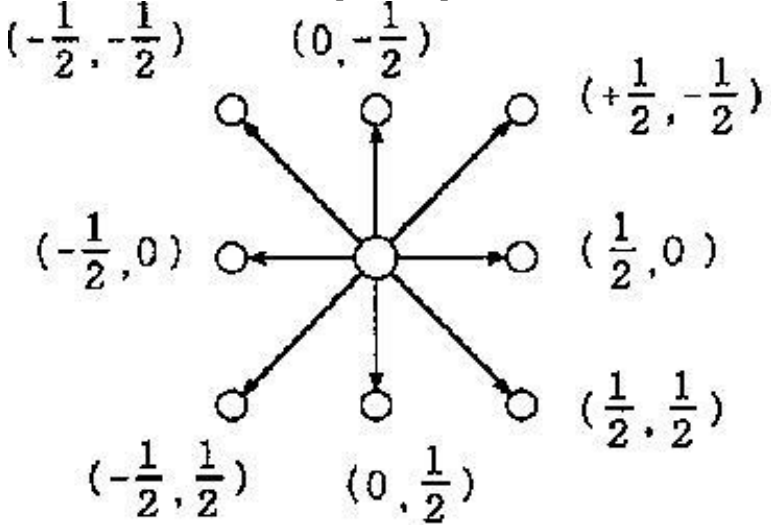
[FIG. 76]


715
Timing-signal generation circuit
φeq
φtim
716
Active control circuit
φact
To components of priority-order determination circuit

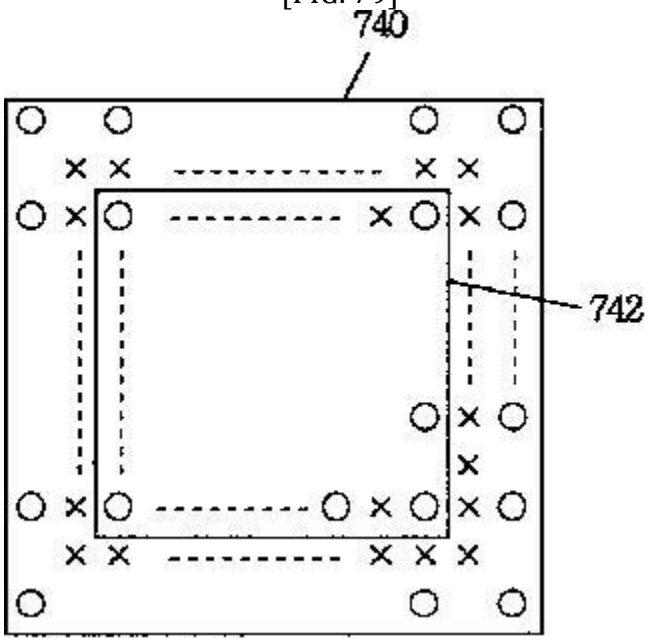
[FIG. 77]


Search-window data PY
Template data PX
2
Input unit
1
Integer-precision computation unit
3
Comparison unit
734
Selector
Block motion level VB
Odd-field-block block motion level Vo
Even-field-block block motion level Ve
722
Prediction-image memory
724
Template-block memory
726
1/2 -pixel-precision computation unit
730
Comparison unit
738
1/2 -pixel-precision computation unit
732
Comparison unit

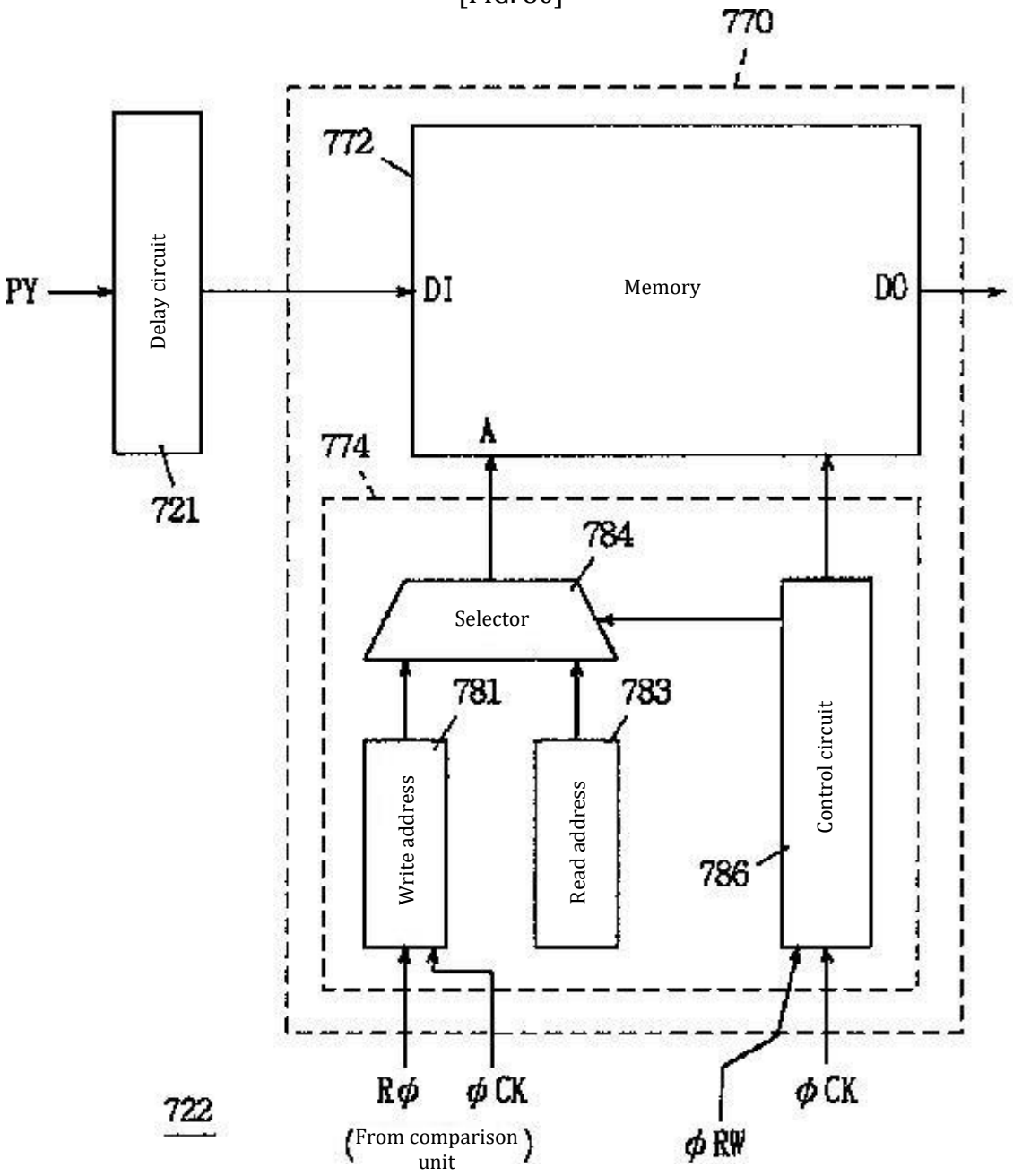
[FIG. 78]


$(-\frac{1}{2},-\frac{1}{2})$
$(0,-\frac{1}{2})$
$(+\frac{1}{2},-\frac{1}{2})$
$(-\frac{1}{2},0)$
$(\frac{1}{2},0)$
$(\frac{1}{2},\frac{1}{2})$
$(-\frac{1}{2},\frac{1}{2})$
$(0,\frac{1}{2})$

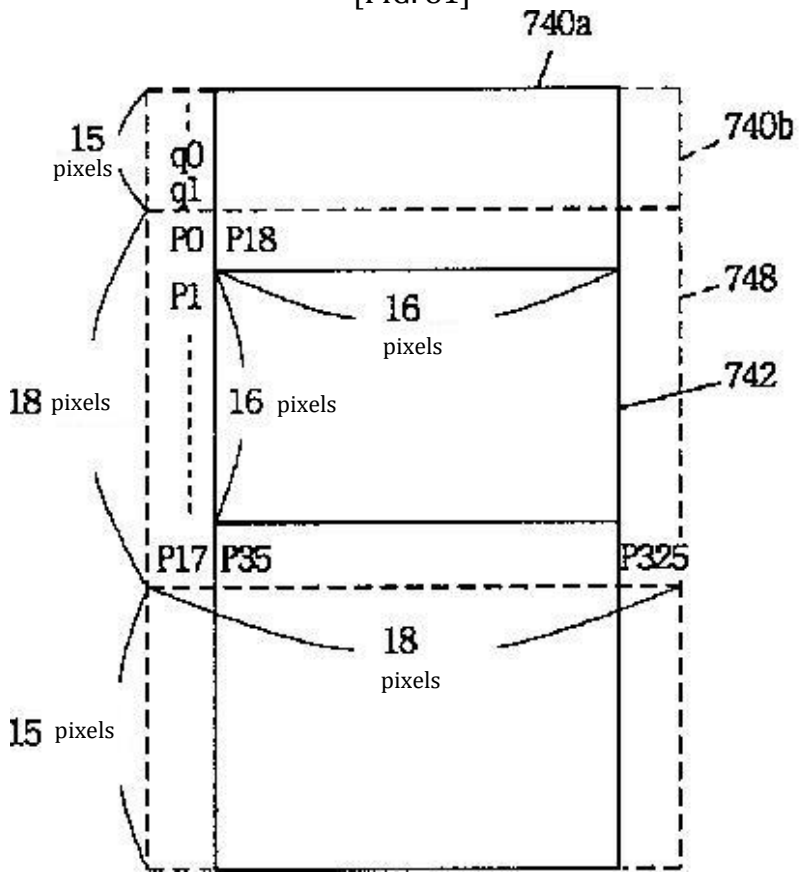
[FIG. 79]


740
742

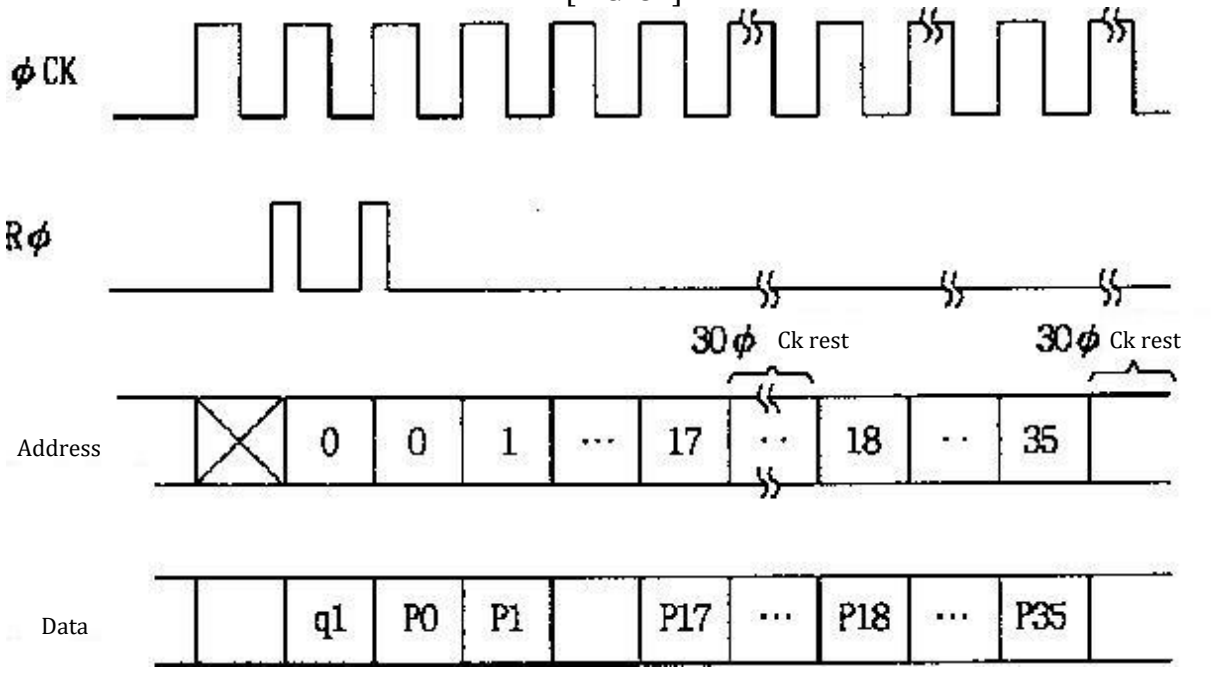
[FIG. 80]


770
772
PY
Delay circuit
DI
Memory
DO
774
A
721
784
Selector
781
783
Write address
Read address
Control circuit
786
722
Rφ
φCK
(From comparison unit)
φRW
φCK

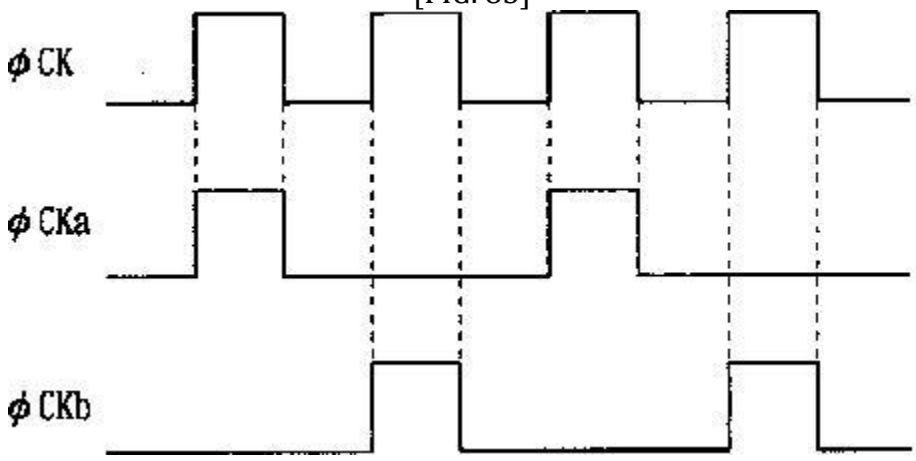
[FIG. 81]


740a
740b
748
742
15 pixels
q0
q1
P0
P18
P1
16 pixels
18 pixels
16 pixels
P17
P35
P325
18 pixels
15 pixels

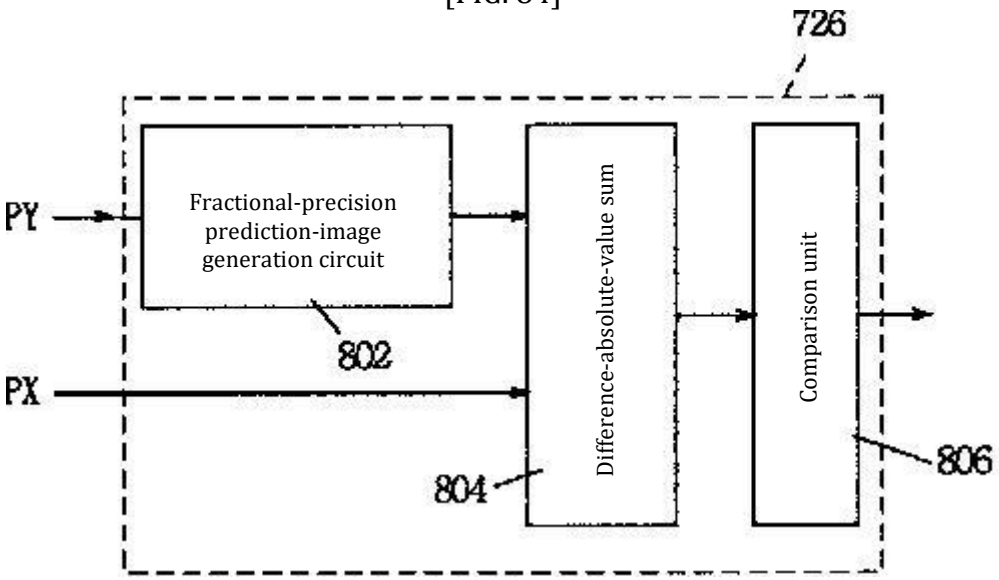
[FIG. 82]


φCK
Rφ
30φ Ck rest
30φ Ck rest
Address
0
0
1
17
18
35
Data
q1
P0
P1
P17
P18
P35

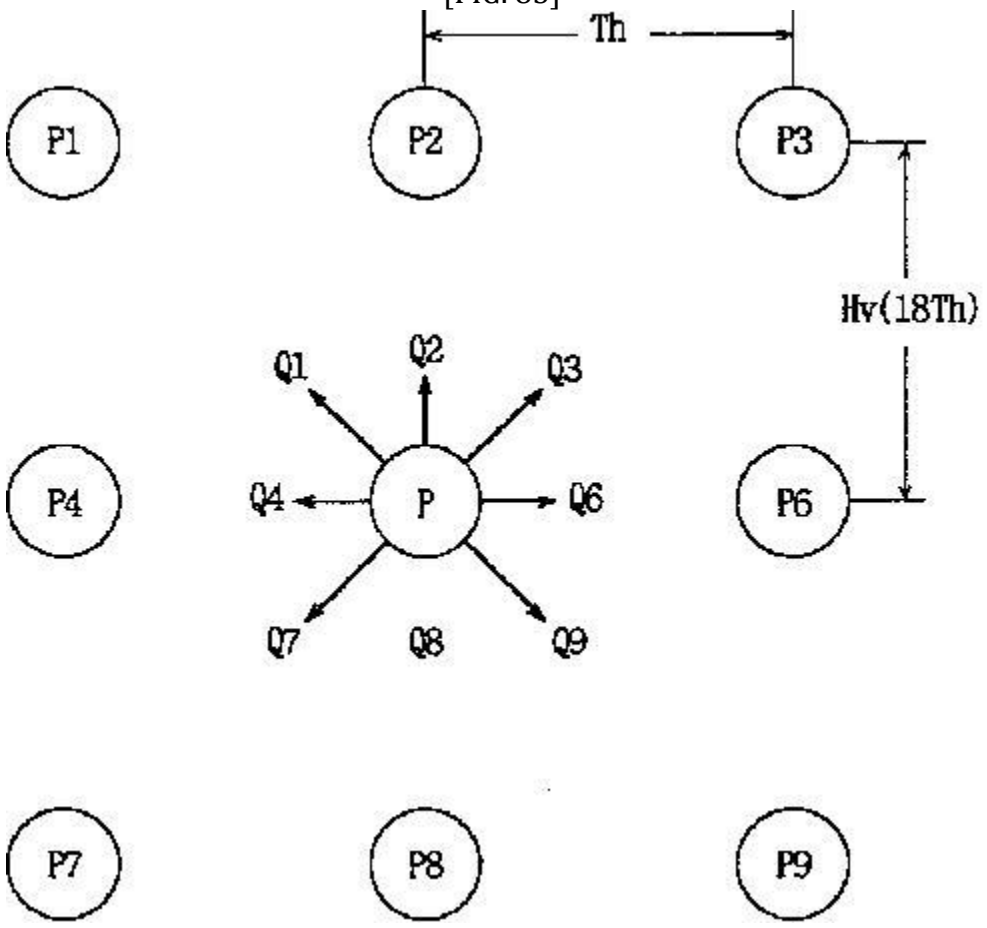
[FIG. 83]


φ CK
φ CKa
φ CKb

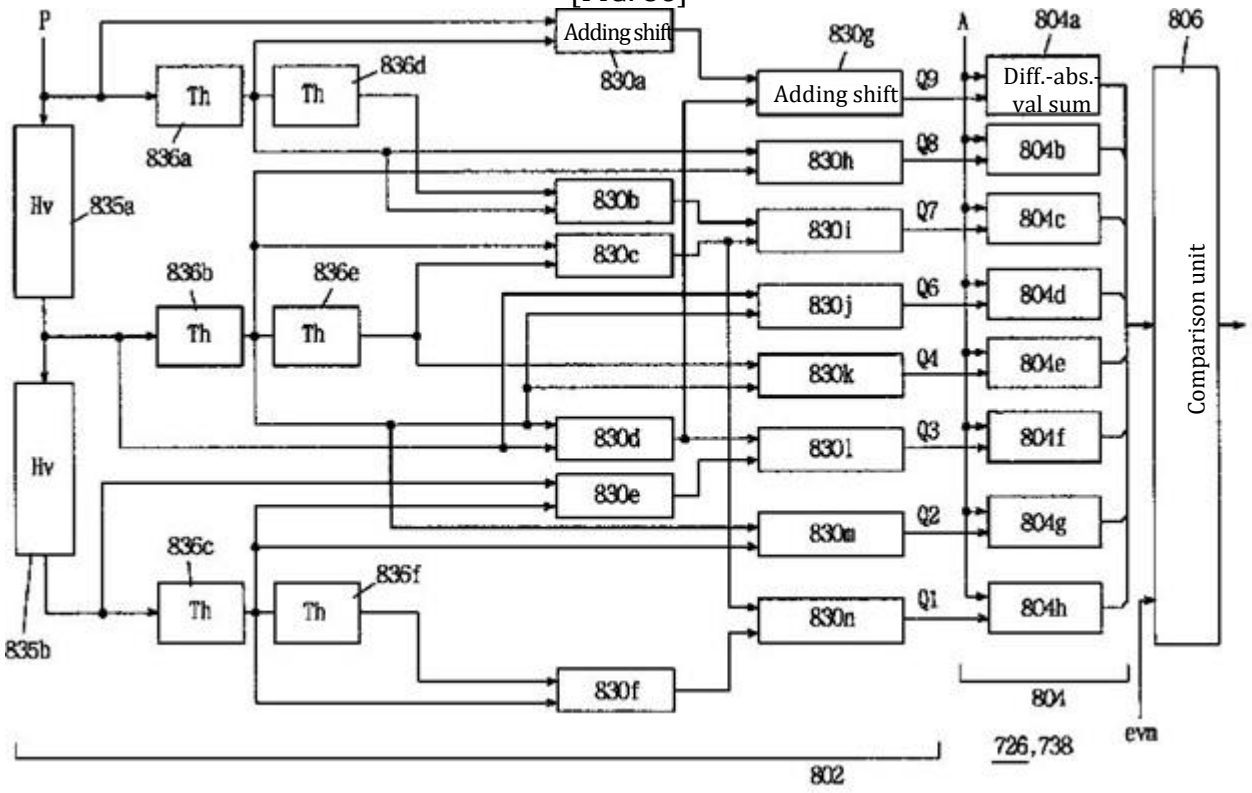
[FIG. 84]


726
PY
PX
Fractional-precision prediction-image generation circuit
802
804
Difference-absolute-value sum
Comparison unit
806

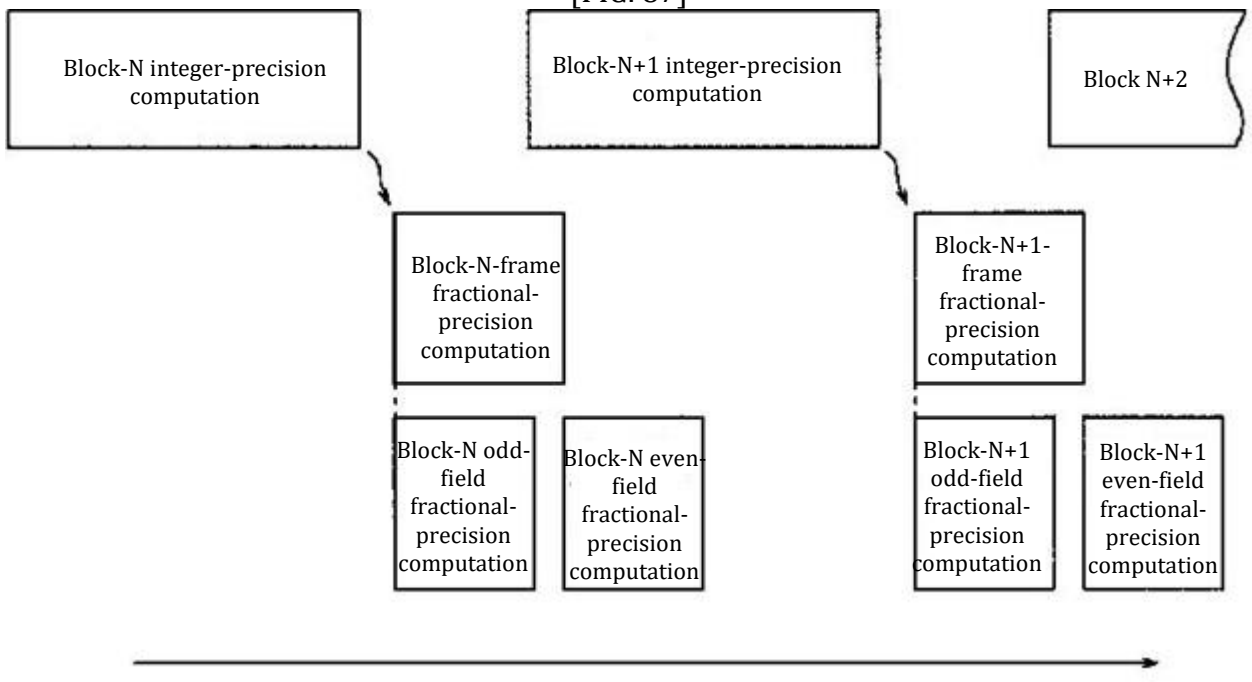
[FIG. 85]



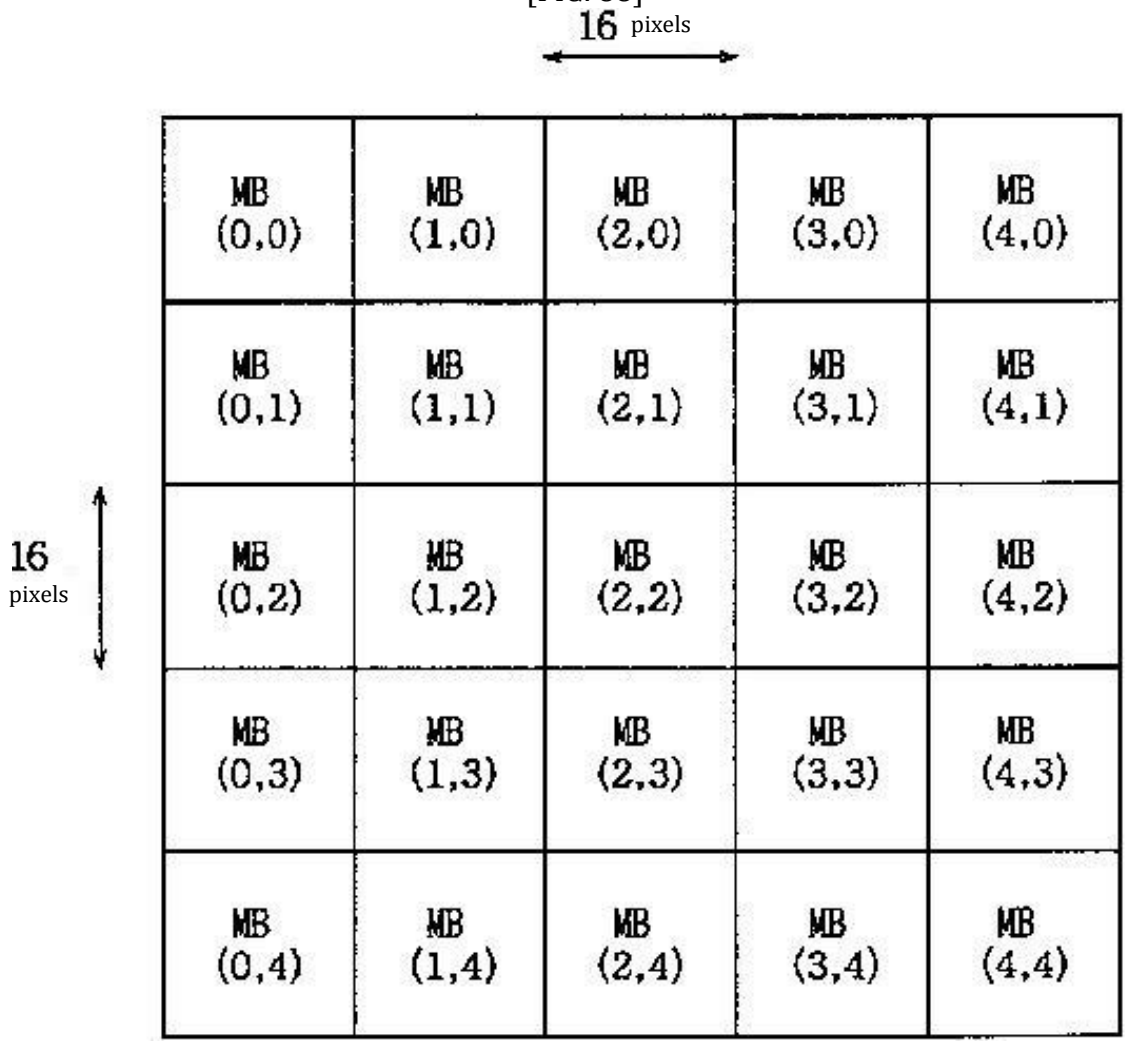
[FIG. 86]


P
Adding shift
830a
830g
A
804a
806
Th
836d
Th
836a
Adding shift
Q9
Diff.-abs.-val sum
Hv
835a
830h
Q8
804b
830b
830i
Q7
804c
830c
836b
836e
830j
Q6
804d
Th
Th
830k
Q4
804e
Hv
830d
830l
Q3
804f
830e
836c
830m
Q2
804g
836f
Th
Th
830n
Q1
804h
835b
830f
Comparison unit
804
726,738
evn
802

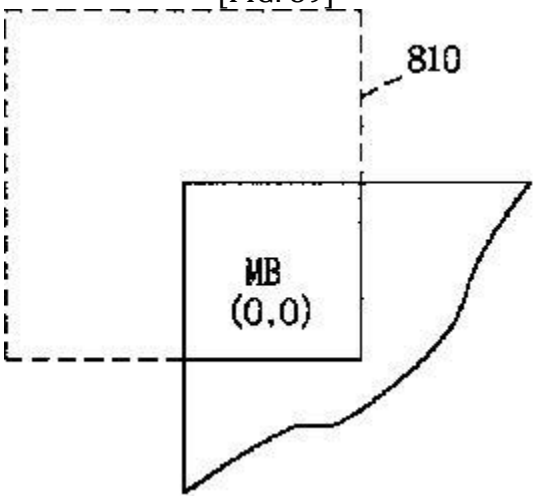
[FIG. 87]


Block-N integer-precision computation
Block-N+1 integer-precision computation
Block N+2
Block-N-frame fractional-precision computation
Block-N+1-frame fractional-precision computation
Block-N odd-field fractional-precision computation
Block-N even-field fractional-precision computation
Block-N+1 odd-field fractional-precision computation
Block-N+1 even-field fractional-precision computation

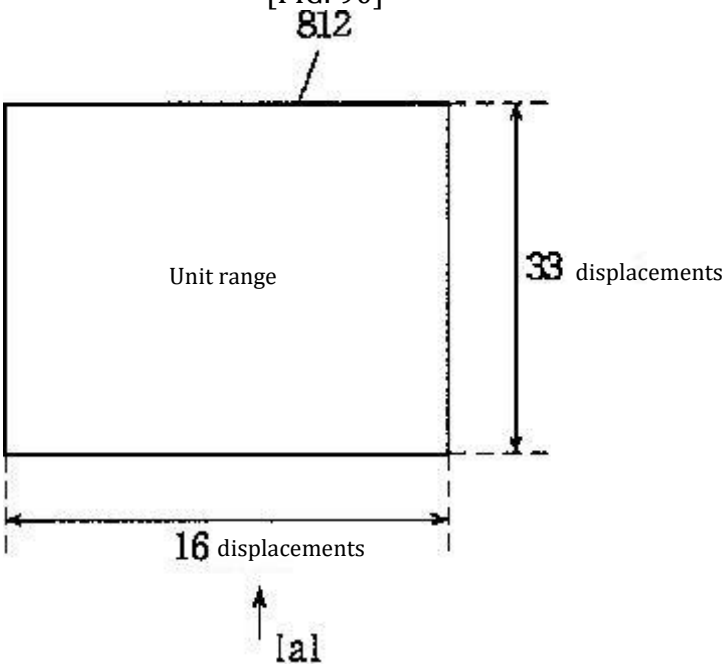
[FIG. 88]


16 pixels
16 pixels
MB (0,0)
MB (1,0)
MB (2,0)
MB (3,0)
MB (4,0)
MB (0,1)
MB (1,1)
MB (2,1)
MB (3,1)
MB (4,1)
MB (0,2)
MB (1,2)
MB (2,2)
MB (3,2)
MB (4,2)
MB (0,3)
MB (1,3)
MB (2,3)
MB (3,3)
MB (4,3)
MB (0,4)
MB (1,4)
MB (2,4)
MB (3,4)
MB (4,4)

[FIG. 89]

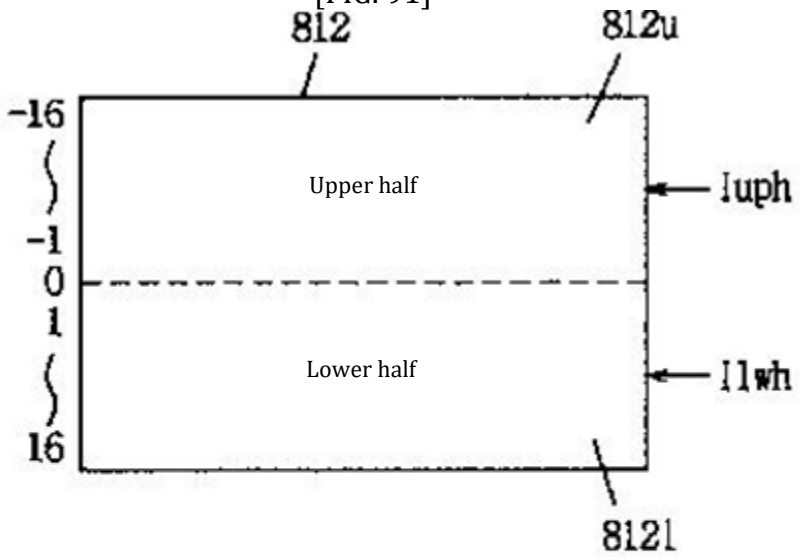

810
MB (0,0)

[FIG. 90]

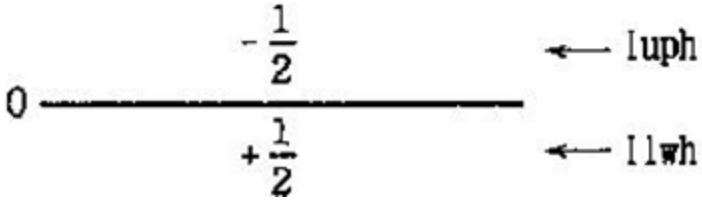


[FIG. 91]

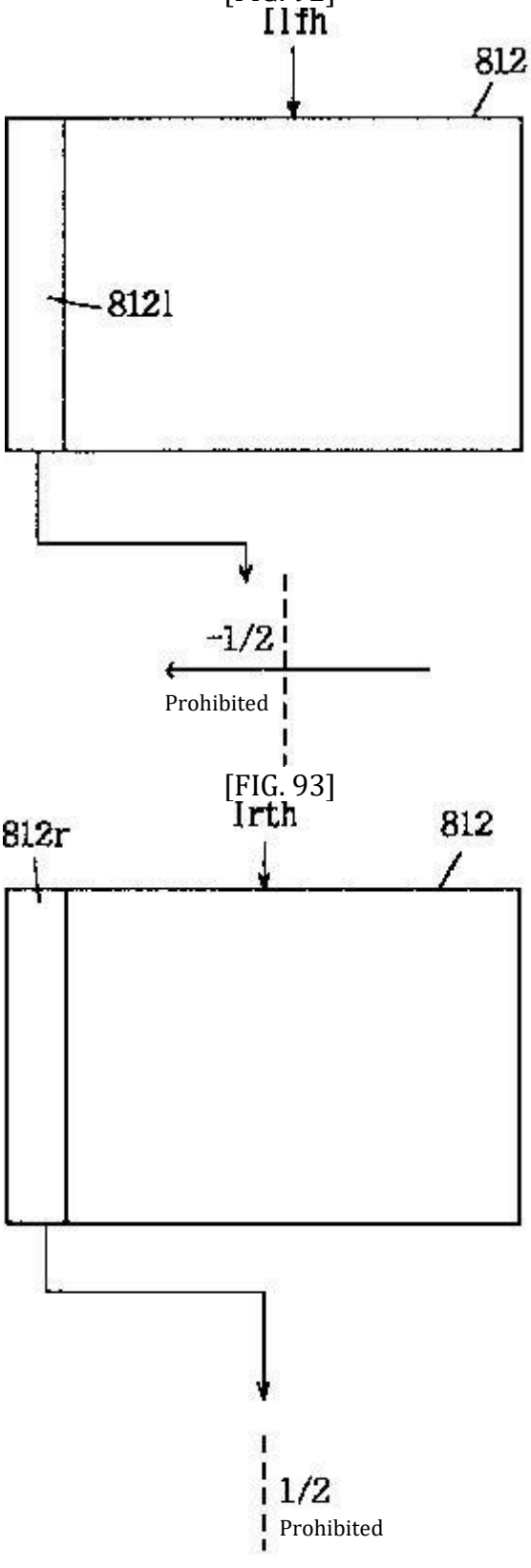
(A)



(B)

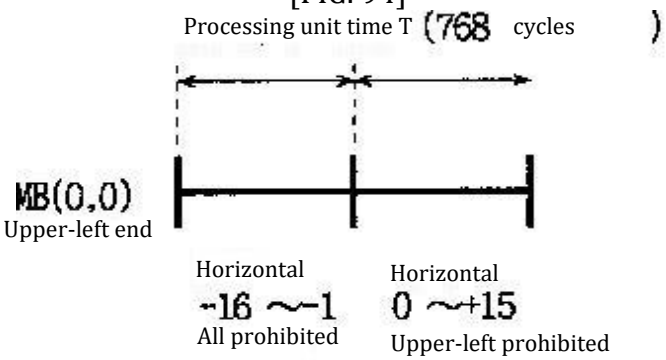


[FIG. 92]

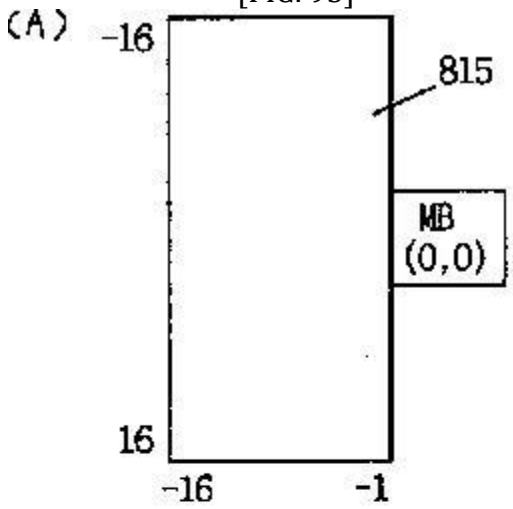


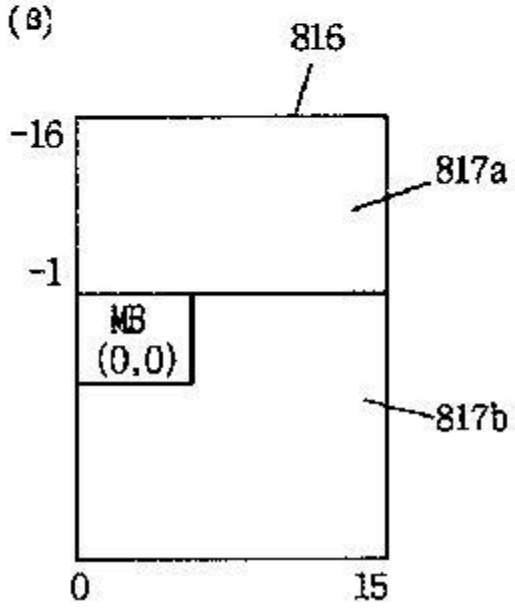
[FIG. 93]

[FIG. 94]

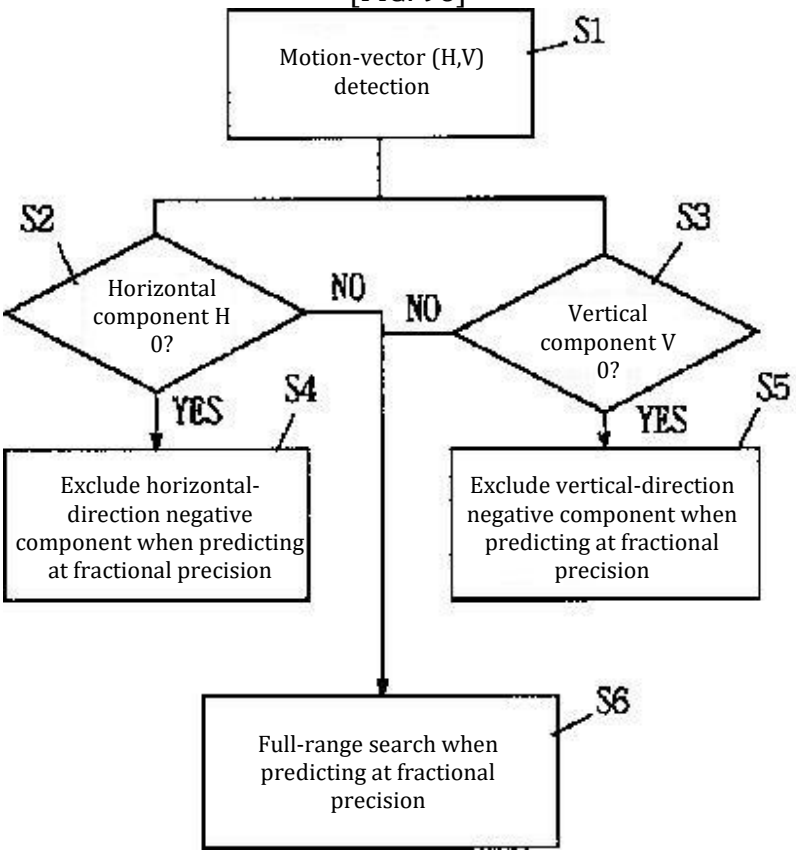

Processing unit time T (768 cycles)
MB(0,0)
Upper-left end
Horizontal
-16 ~-1
All prohibited
Horizontal
0 ~+15
Upper-left prohibited

[FIG. 95]

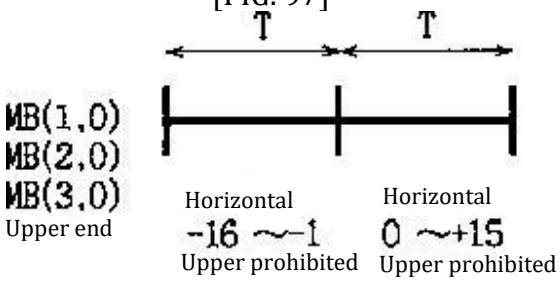

(A)
-16
815
MB
(0,0)
16
-16
-1


(B)
816
-16
817a
-1
MB
(0,0)
817b
0
15

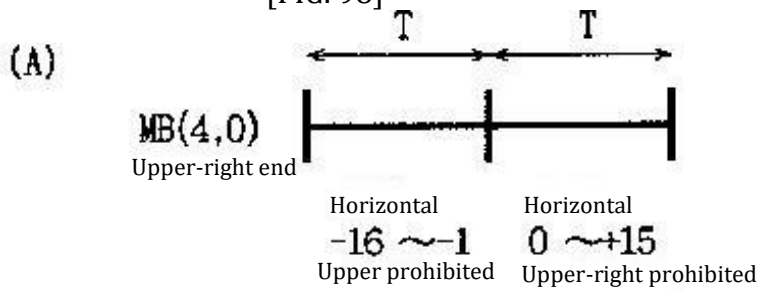
[FIG. 96]

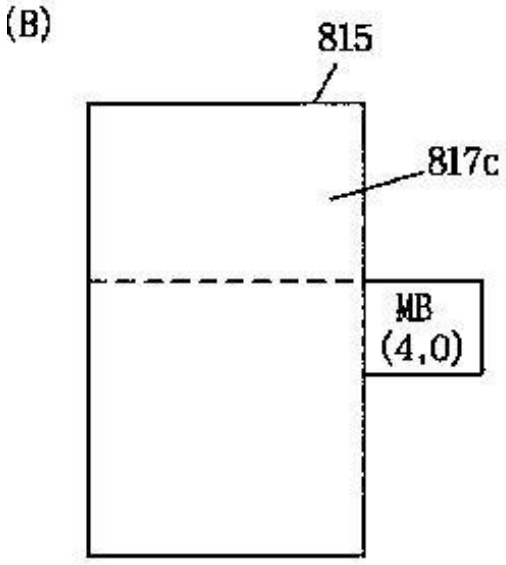

Motion-vector (H,V) detection
S1
S2
Horizontal component H 0?
NO
S3
Vertical component V 0?
NO
YES
S4
Exclude horizontal-direction negative component when predicting at fractional precision
YES
S5
Exclude vertical-direction negative component when predicting at fractional precision
S6
Full-range search when predicting at fractional precision

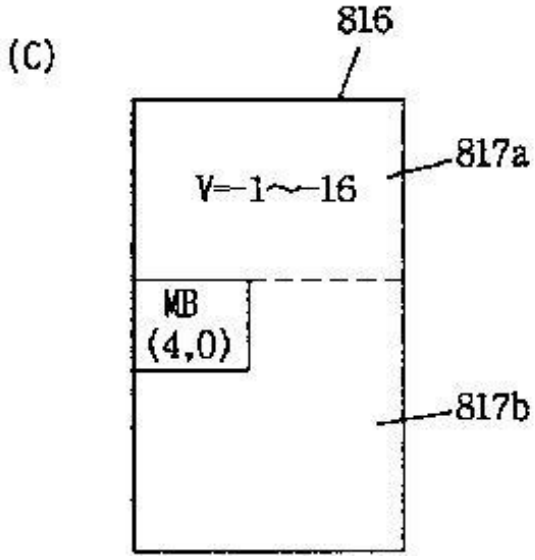
[FIG. 97]


T
T
MB(1,0)
MB(2,0)
MB(3,0)
Upper end
Horizontal
-16 ~-1
Upper prohibited
Horizontal
0 ~+15
Upper prohibited

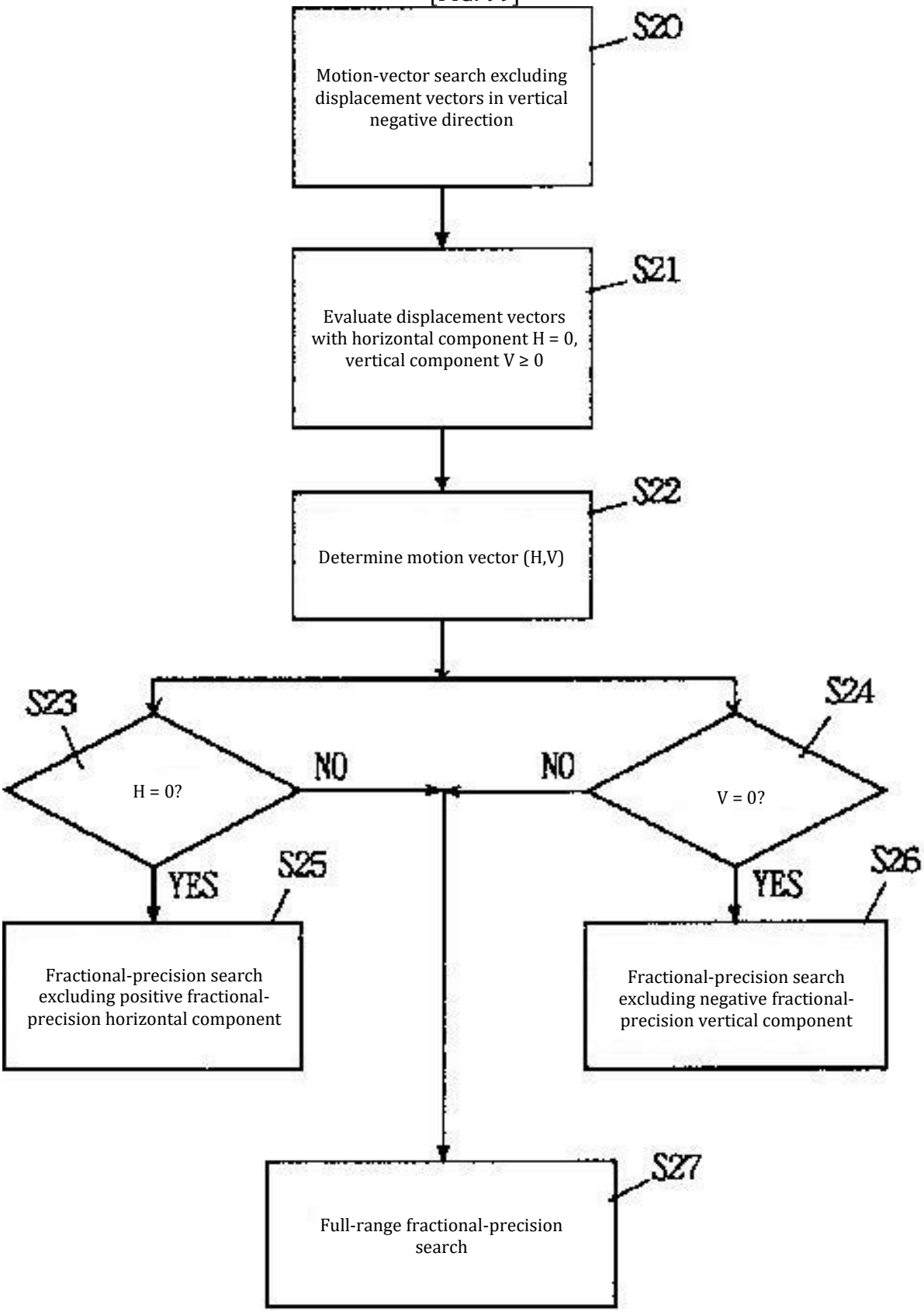
[FIG. 98]


(A)
T
T
MB(4,0)
Upper-right end
Horizontal
-16 ~-1
Upper prohibited
Horizontal
0 ~+15
Upper-right prohibited

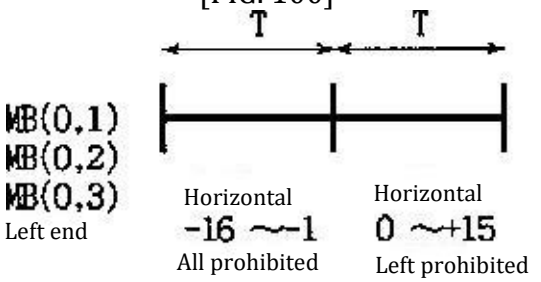

(B)
815
817c
MB
(4,0)


(C)
816
V=-1~-16
817a
MB
(4,0)
817b

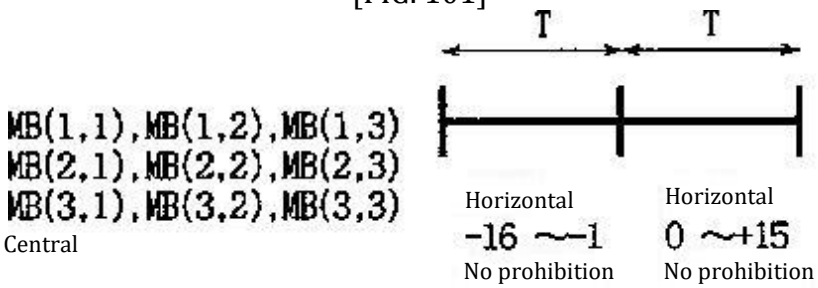
[FIG. 99]


S20
Motion-vector search excluding displacement vectors in vertical negative direction
S21
Evaluate displacement vectors with horizontal component H = 0, vertical component V ≥ 0
S22
Determine motion vector (H,V)
S23
H = 0?
NO
S24
V = 0?
NO
YES
S25
Fractional-precision search excluding positive fractional-precision horizontal component
YES
S26
Fractional-precision search excluding negative fractional-precision vertical component
S27
Full-range fractional-precision search

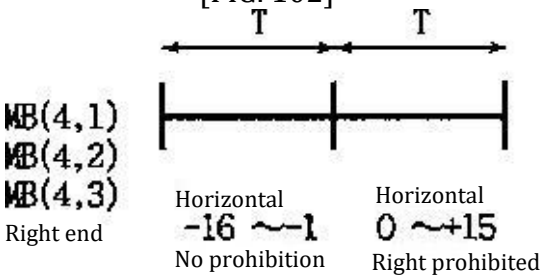
[FIG. 100]



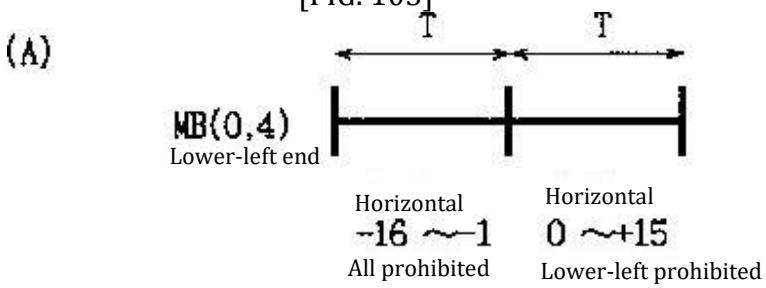
[FIG. 101]

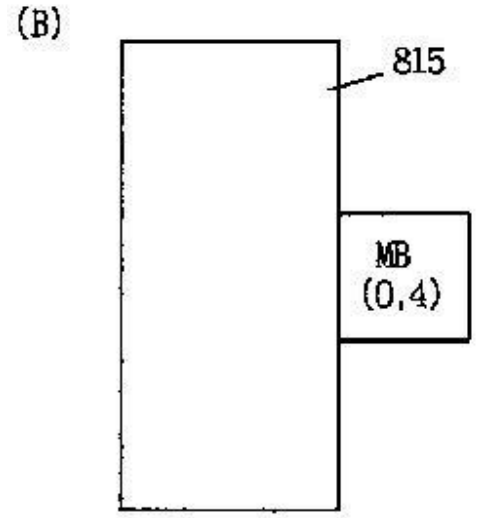


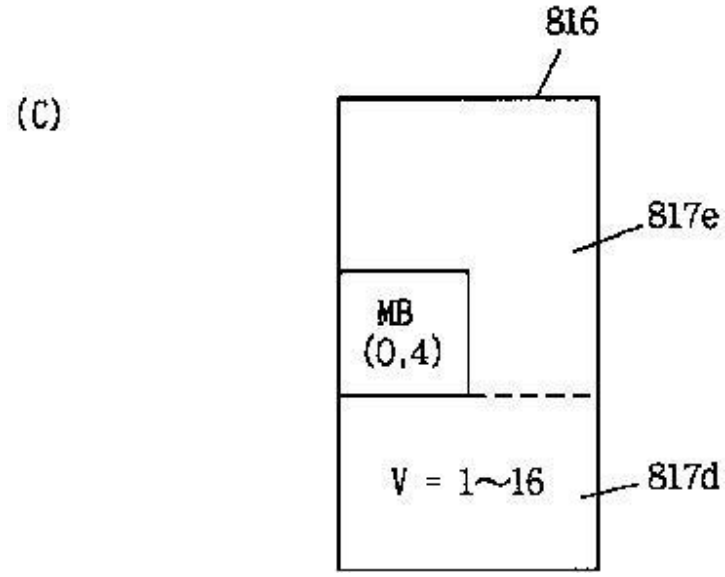
[FIG. 102]



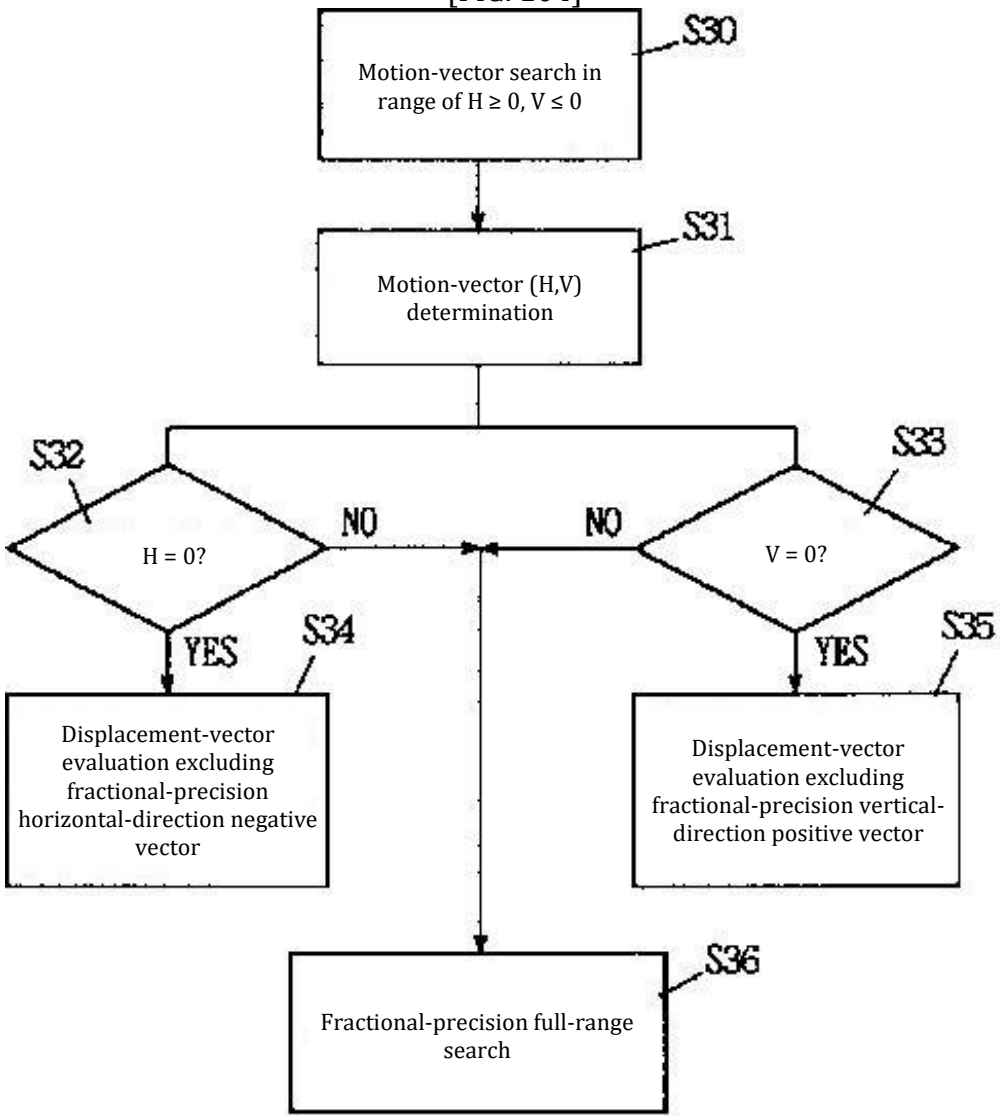
[FIG. 103]


(A)
T
T
MB(0,4)
Lower-left end
Horizontal
-16 ~-1
All prohibited
Horizontal
0 ~+15
Lower-left prohibited

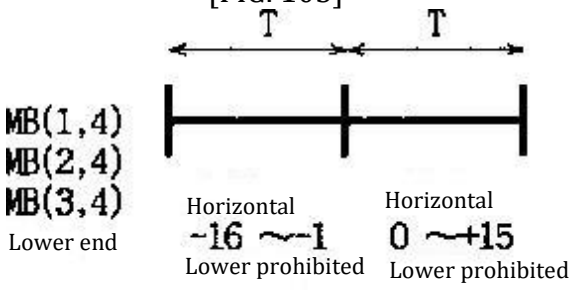

(B)
815
MB
(0,4)


(C)
816
817e
MB
(0,4)
V = 1~16
817d

[FIG. 104]

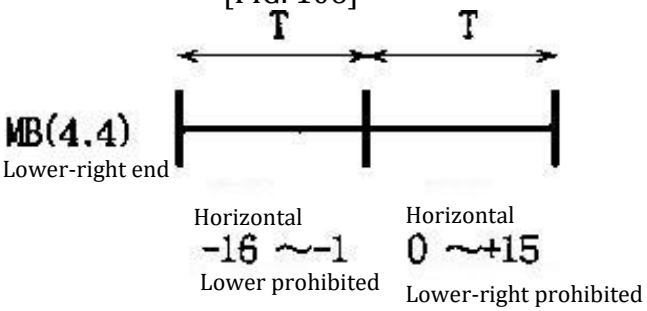


[FIG. 105]

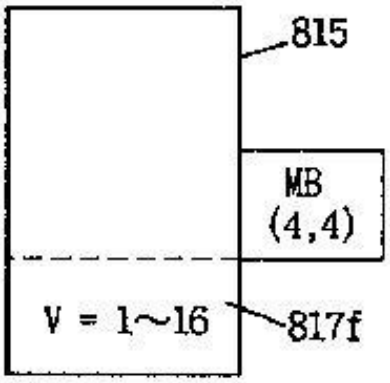


[FIG. 106]

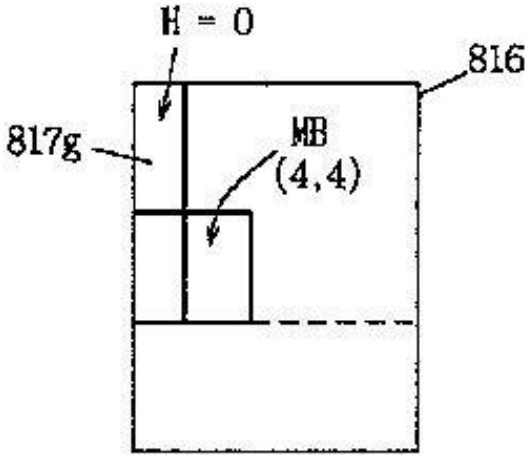
(A)


T
T
MB(4.4)
Lower-right end
Horizontal
-16 ~-1
Lower prohibited
Horizontal
0 ~+15
Lower-right prohibited

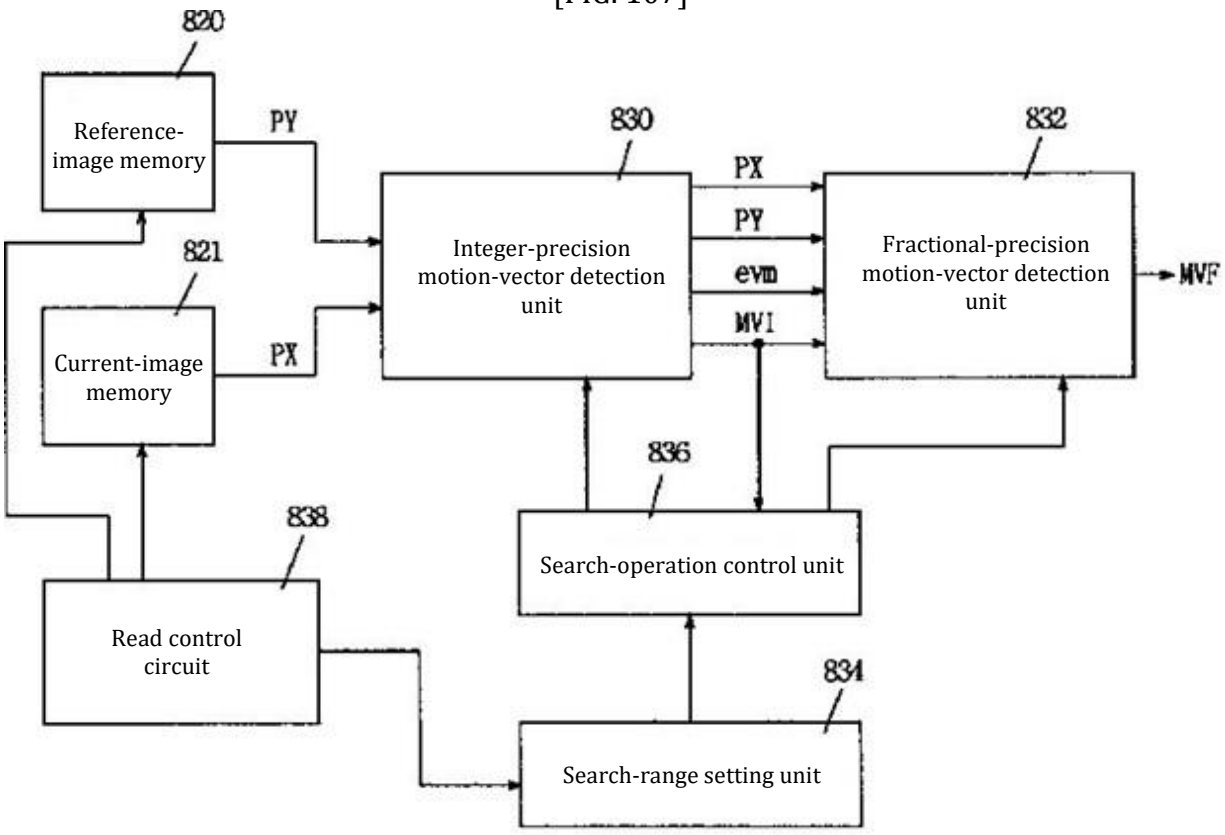
(B)


815
MB
(4,4)
V = 1~16
817f

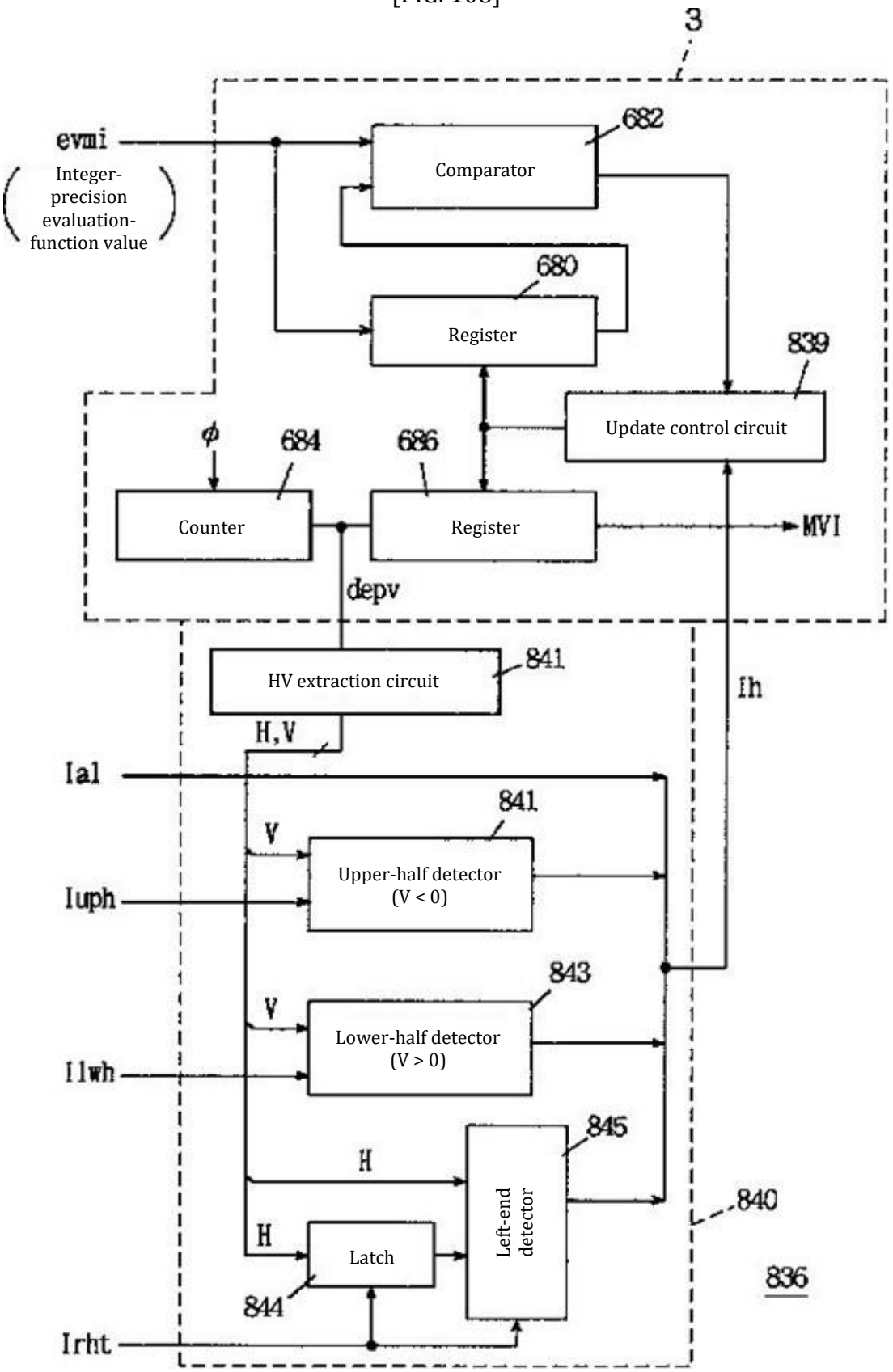
(C)


H = 0
816
817g
MB
(4,4)

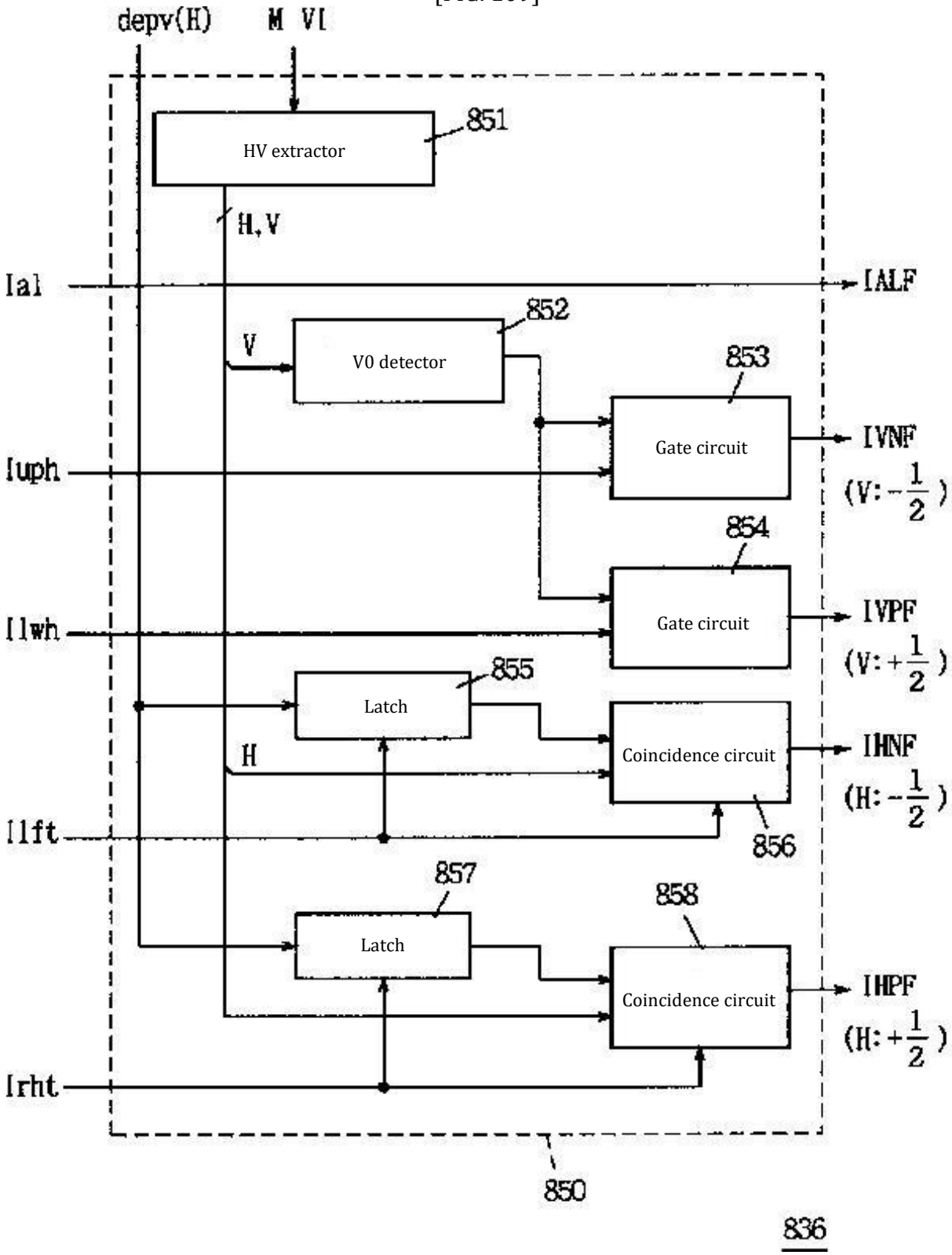
[FIG. 107]


820
Reference-image memory
PY
830
PX
PY
evm
MVI
832
Integer-precision motion-vector detection unit
Fractional-precision motion-vector detection unit
MVF
821
Current-image memory
PX
836
838
Search-operation control unit
Read control circuit
834
Search-range setting unit

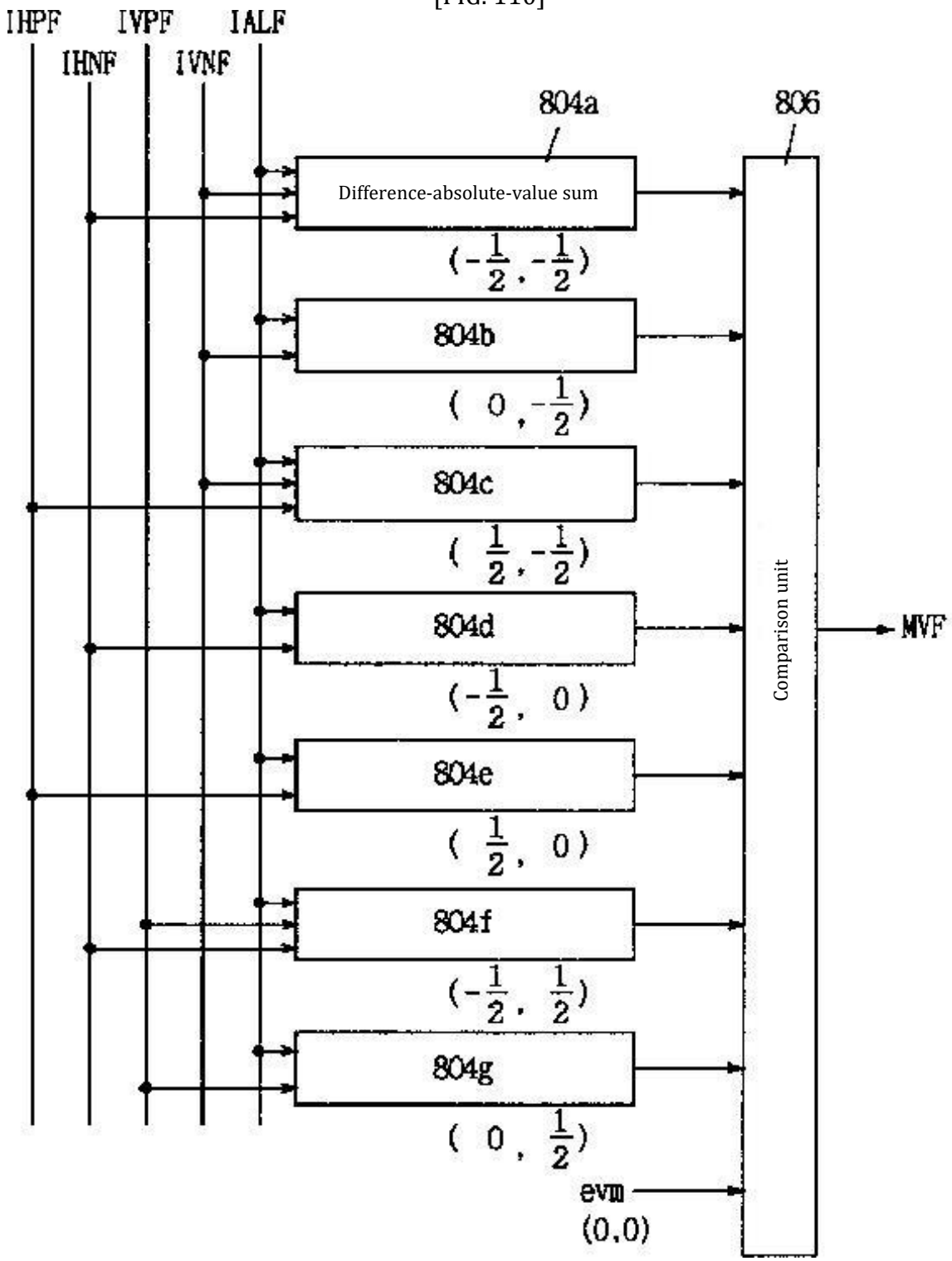
[FIG. 108]


3
evmi
Integer-precision evaluation-function value
682
Comparator
680
Register
839
Update control circuit
φ
684
686
Counter
Register
MVI
depv
841
HV extraction circuit
Ih
H,V
Ial
841
V
Upper-half detector (V < 0)
Iuph
843
V
Lower-half detector (V > 0)
Ilwh
845
H
H
Latch
Left-end detector
840
844
836
Irht

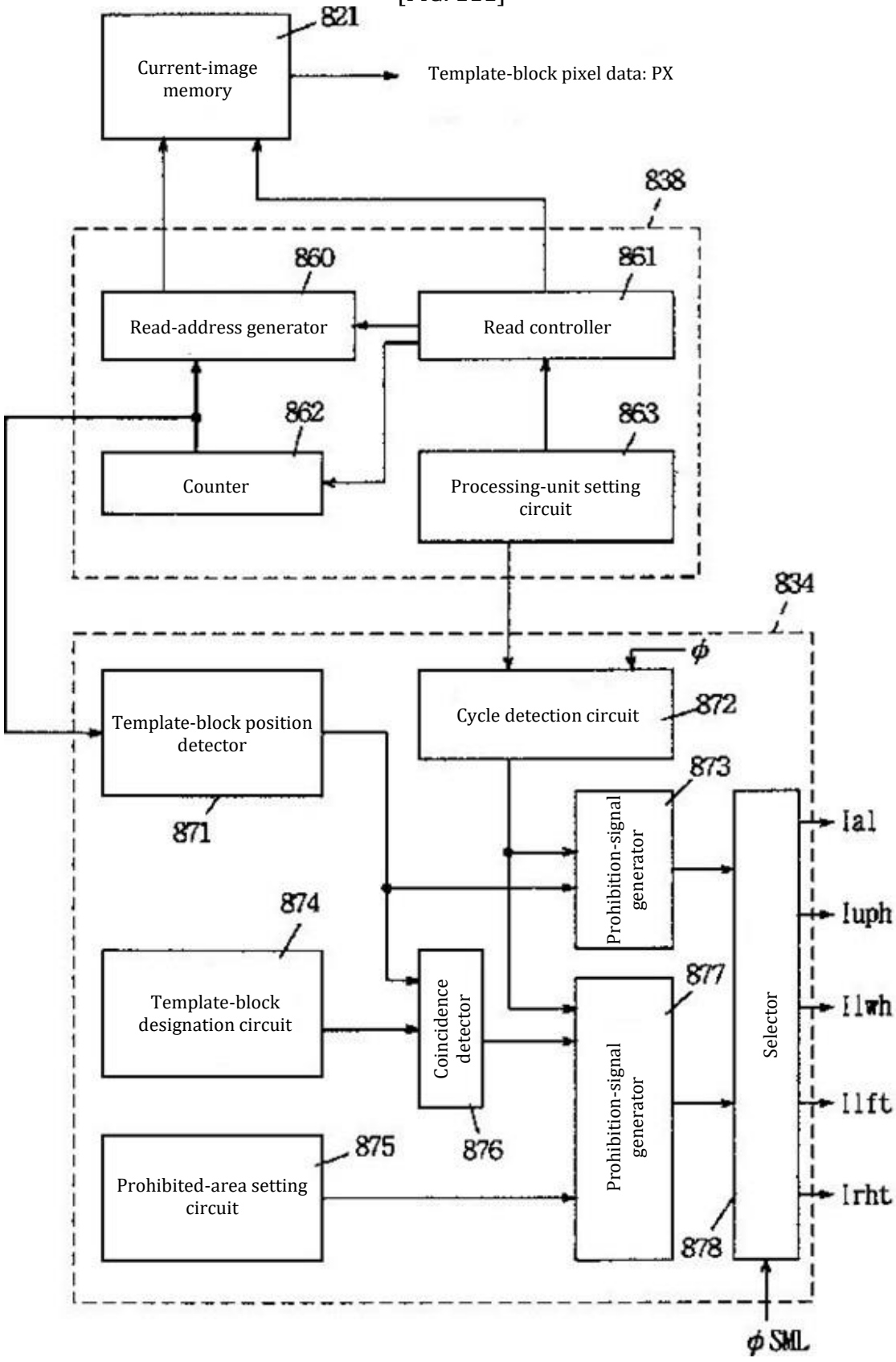
[FIG. 109]


depv(H)
M VI
HV extractor
851
H,V
Ial
IALF
852
V
V0 detector
853
Gate circuit
IVNF
(V: -1/2)
Iuph
854
Gate circuit
IVPF
(V: +1/2)
Ilwh
855
Latch
Coincidence circuit
IHNF
(H: -1/2)
H
856
Ilft
857
858
Latch
Coincidence circuit
IHPF
(H: +1/2)
Irht
850
836

[FIG. 110]

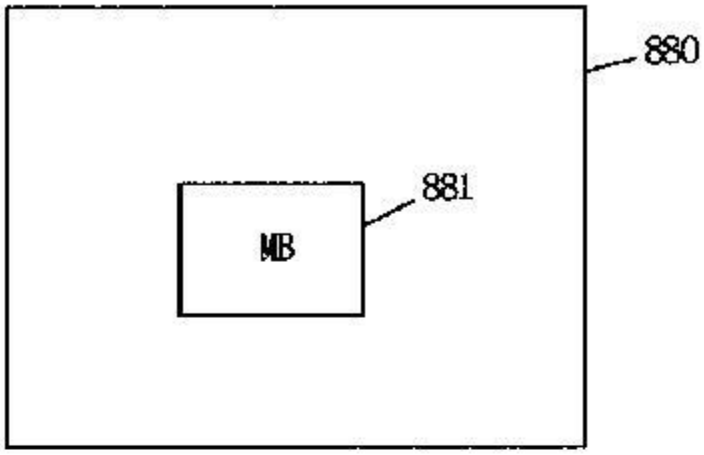

IHPF
IVPF
IALF
IHNF
IVNF
804a
806
Difference-absolute-value sum
$(-\frac{1}{2}, -\frac{1}{2})$
804b
$(0, -\frac{1}{2})$
804c
$(\frac{1}{2}, -\frac{1}{2})$
804d
$(-\frac{1}{2}, 0)$
804e
$(\frac{1}{2}, 0)$
804f
$(-\frac{1}{2}, \frac{1}{2})$
804g
$(0, \frac{1}{2})$
evm
(0,0)
Comparison unit
MVF

[FIG. 111]

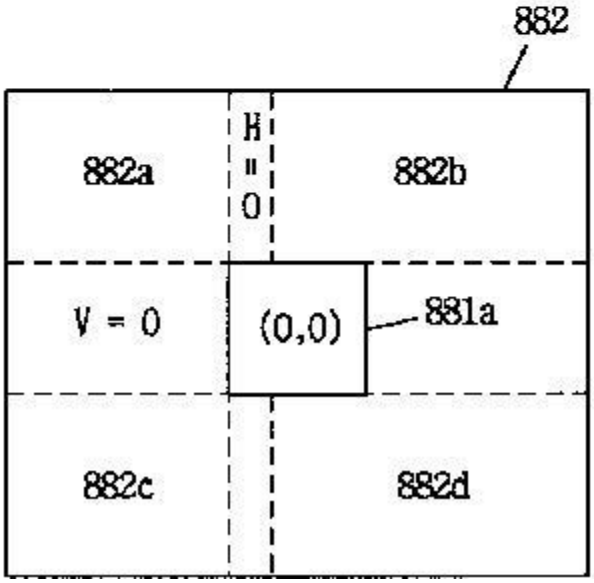

821
Current-image memory
Template-block pixel data: PX
838
860
Read-address generator
861
Read controller
862
Counter
863
Processing-unit setting circuit
834
φ
Template-block position detector
871
Cycle detection circuit
872
873
Prohibition-signal generator
Ial
Iuph
874
Template-block designation circuit
Coincidence detector
876
877
Prohibition-signal generator
Selector
Ilwh
Ilft
Irht
875
Prohibited-area setting circuit
878
φSML

[FIG. 112]

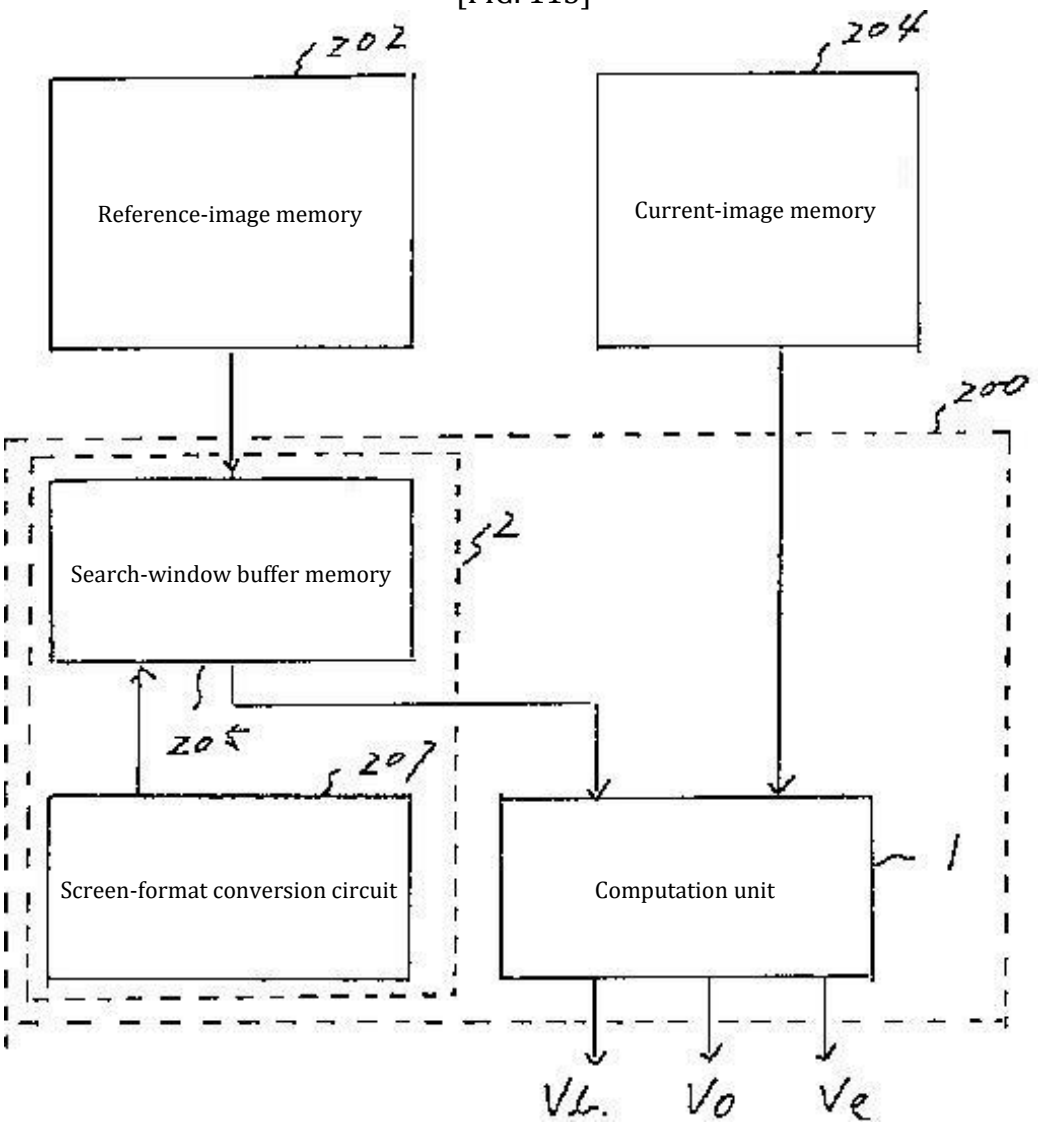
(A)



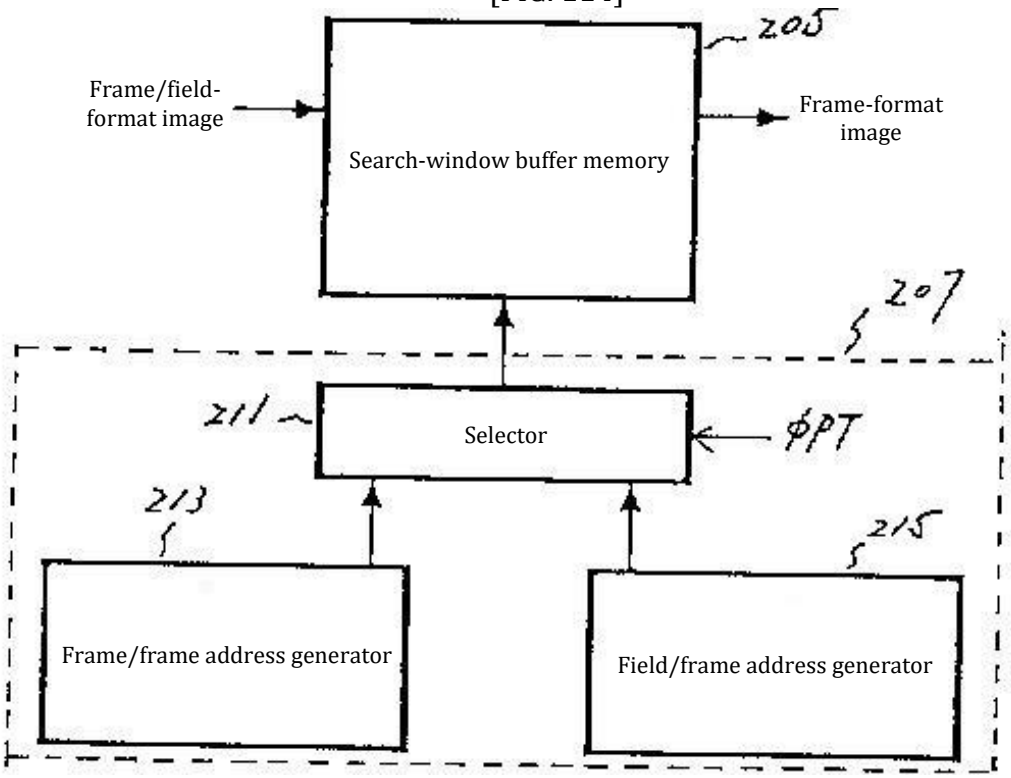
(B)



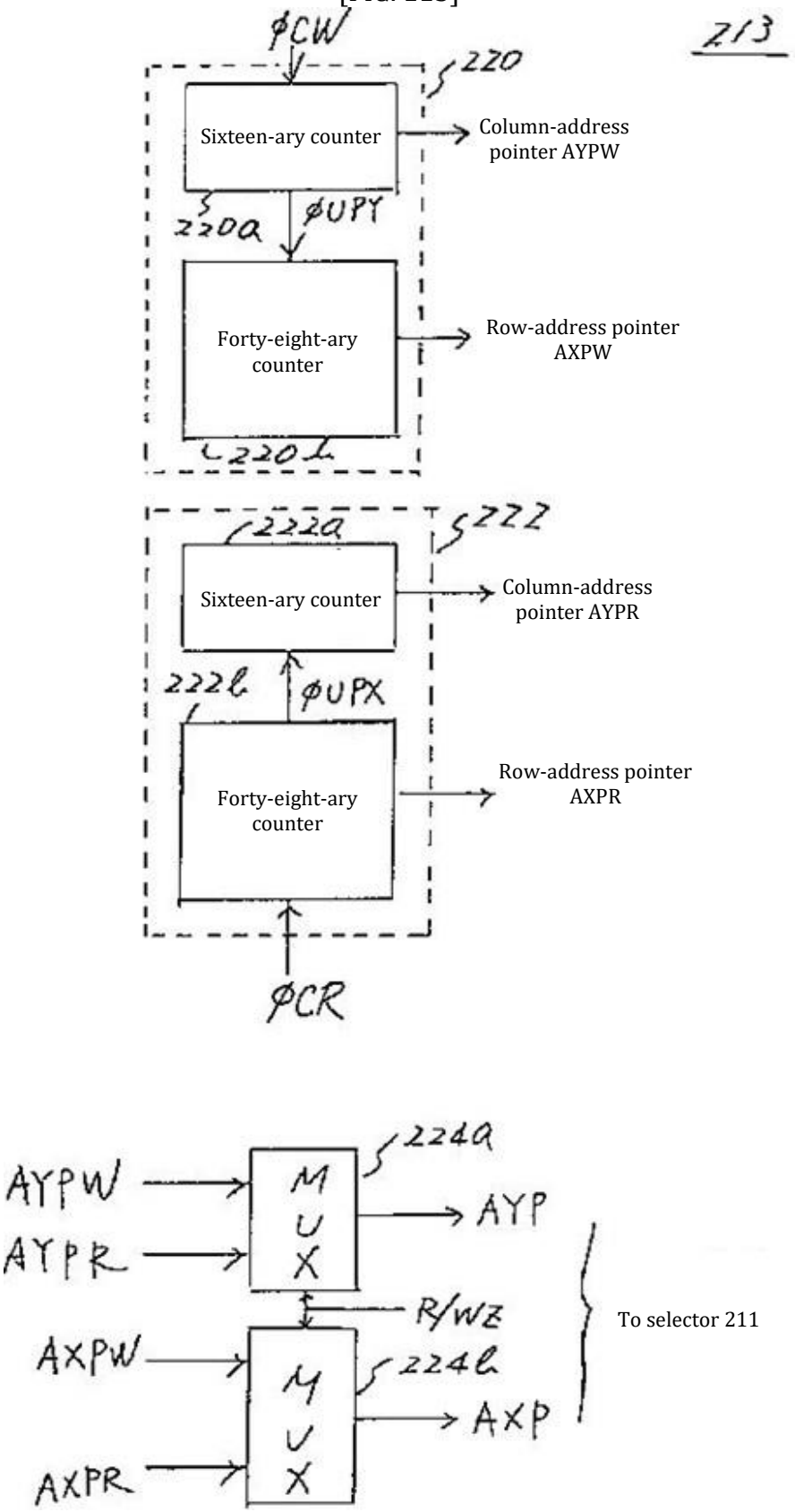
[FIG. 113]


202
204
Reference-image memory
Current-image memory
200
Search-window buffer memory
52
205
207
Screen-format conversion circuit
Computation unit
1
VL
Vo
Ve

[FIG. 114]

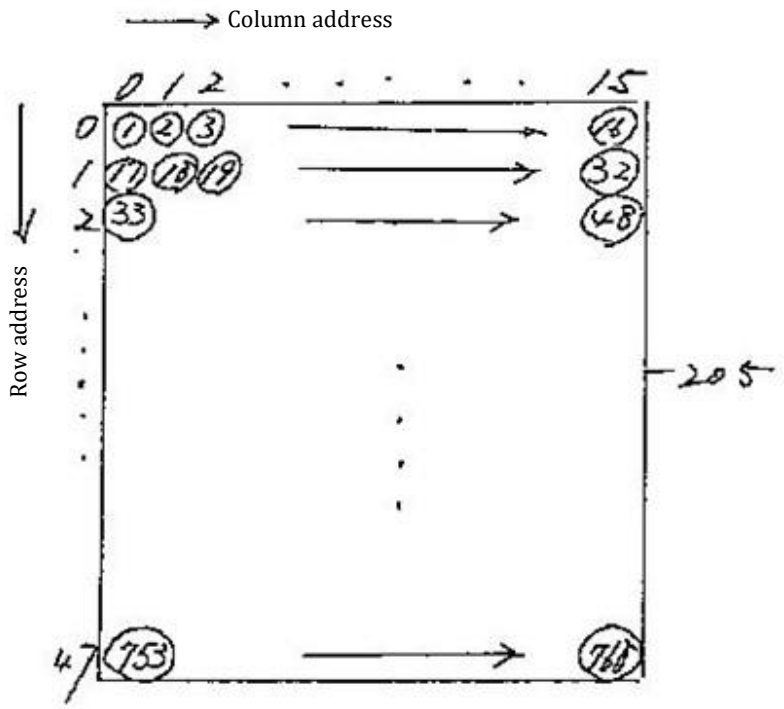

205
Frame/field-format image
Search-window buffer memory
Frame-format image
207
211
Selector
ϕPT
213
215
Frame/frame address generator
Field/frame address generator

[FIG. 115]

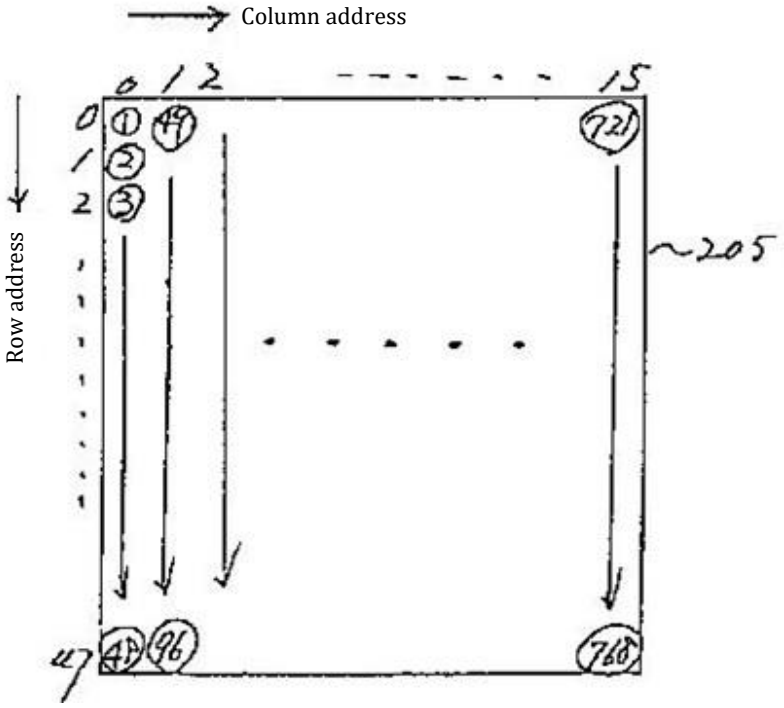

213
ϕCW
220
Sixteen-ary counter
Column-address pointer AYPW
220a
ϕUPY
Forty-eight-ary counter
Row-address pointer AXPW
220b
222
222a
Sixteen-ary counter
Column-address pointer AYPR
222b
ϕUPX
Forty-eight-ary counter
Row-address pointer AXPR
ϕCR
224a
AYPW
AYPR
M U X
AYP
R/WZ
AXPW
AXPR
224b
AXP
To selector 211

[FIG. 116]

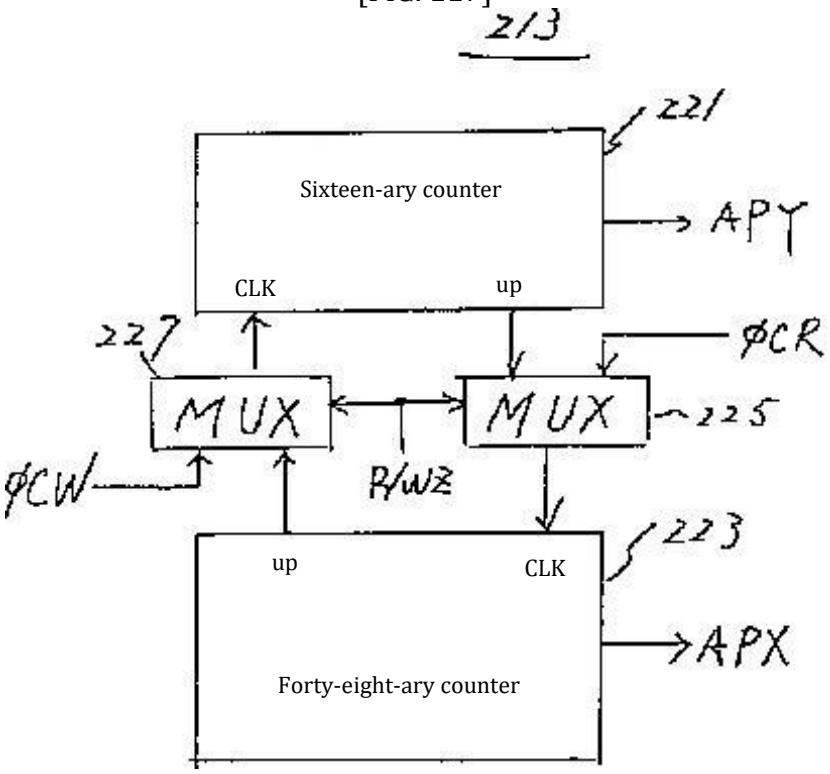
(A) Write time



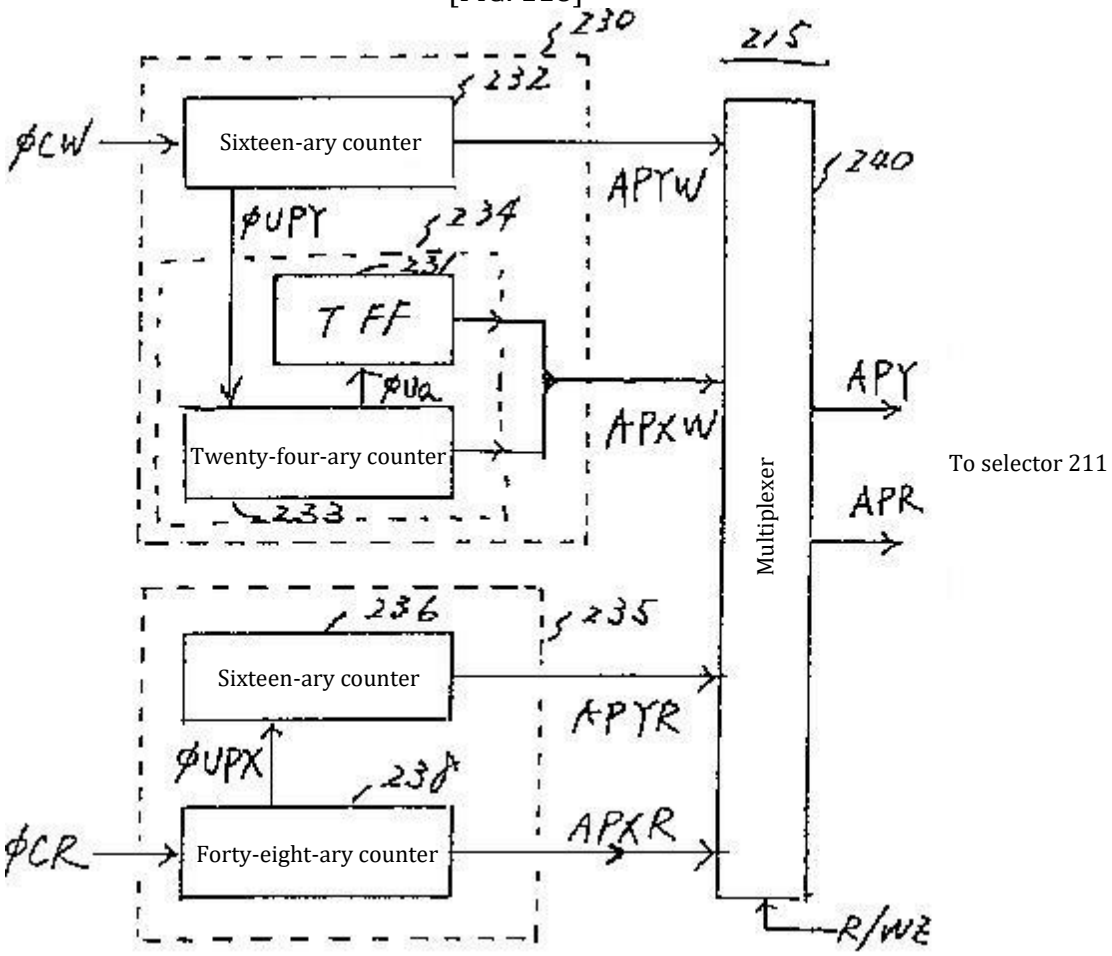
(B) Read time



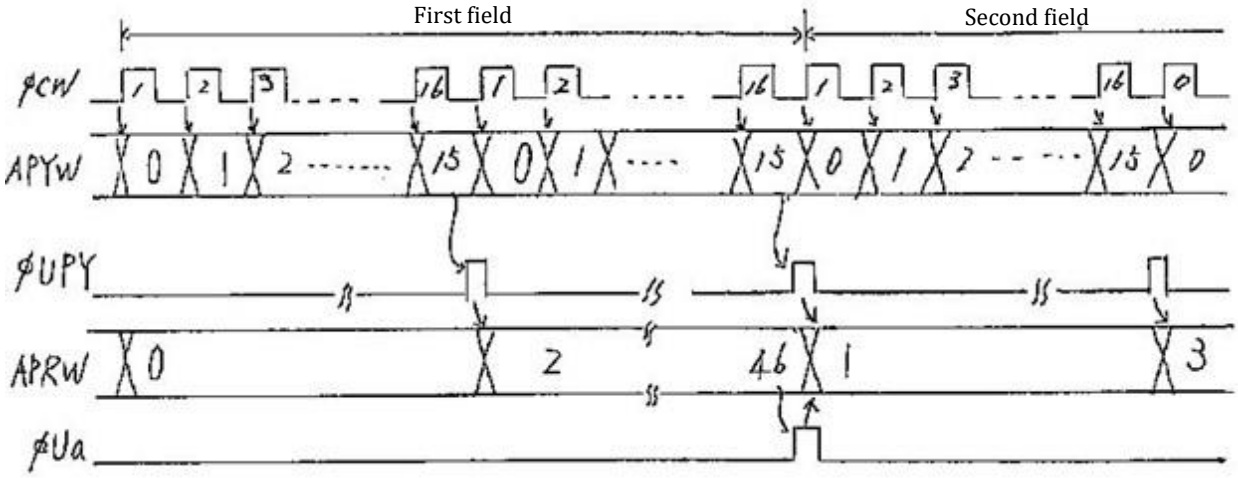
[FIG. 117]


213
221
Sixteen-ary counter
APY
CLK
up
227
ØCR
MUX
MUX
225
ØCW
R/WZ
223
up
CLK
APX
Forty-eight-ary counter

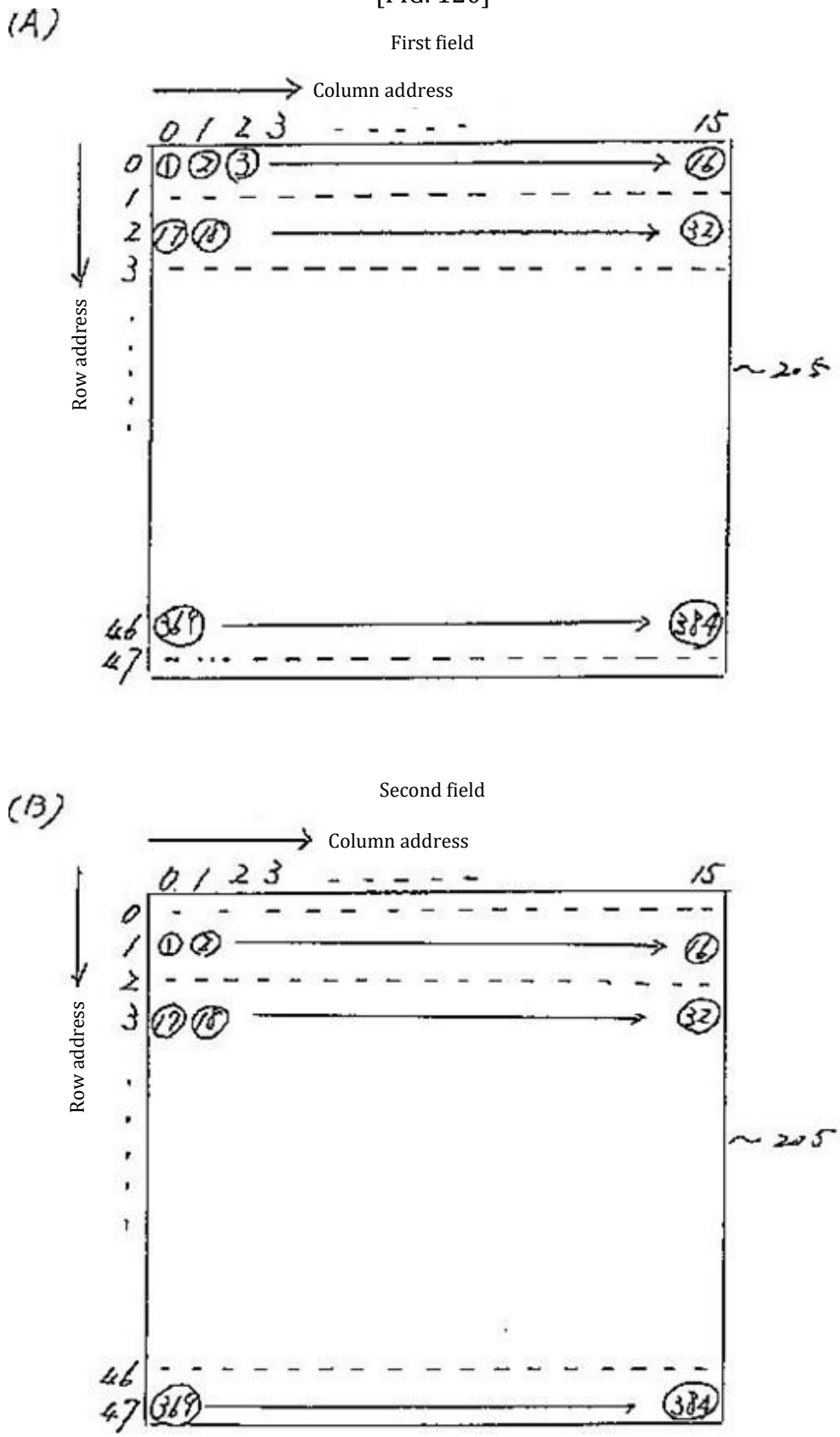
[FIG. 118]



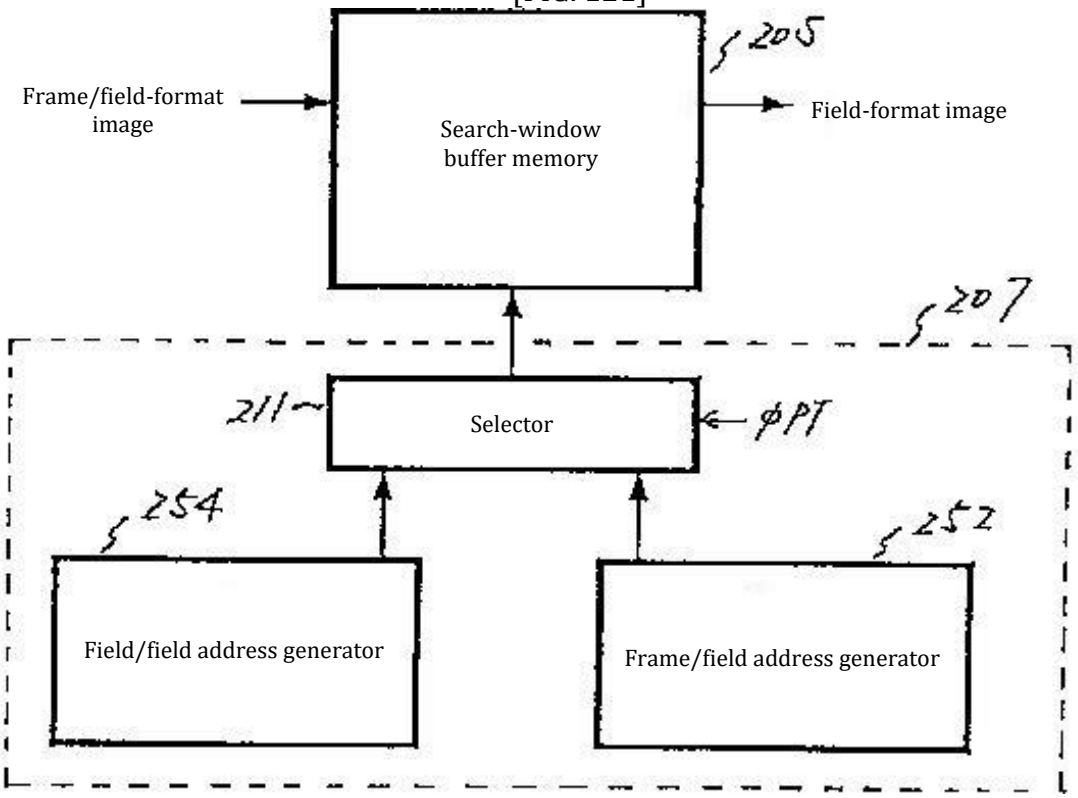
[FIG. 119]



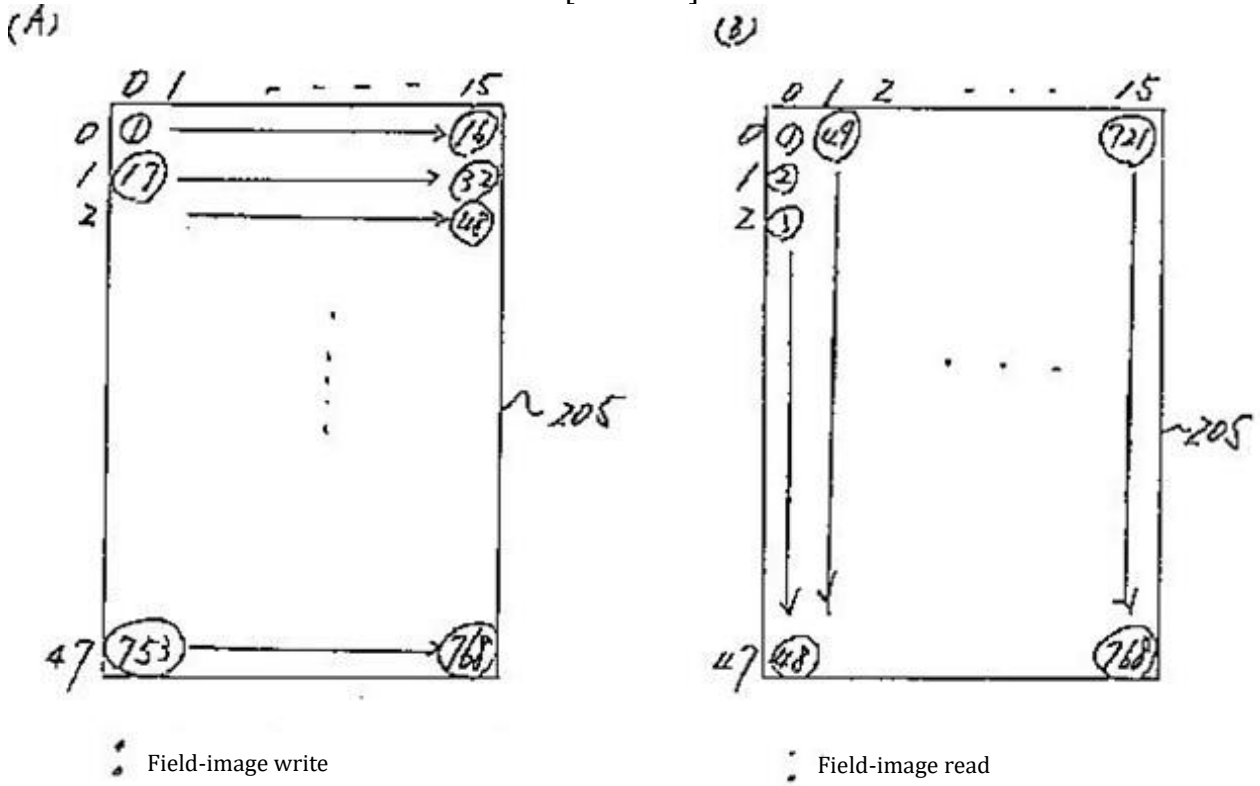
[FIG. 120]


(A)
First field
Column address
Row address
0 1 2 3 15
0
1
2
3
46
47
205
(B)
Second field
Column address
Row address
0 1 2 3 15
0
1
2
3
46
47
205

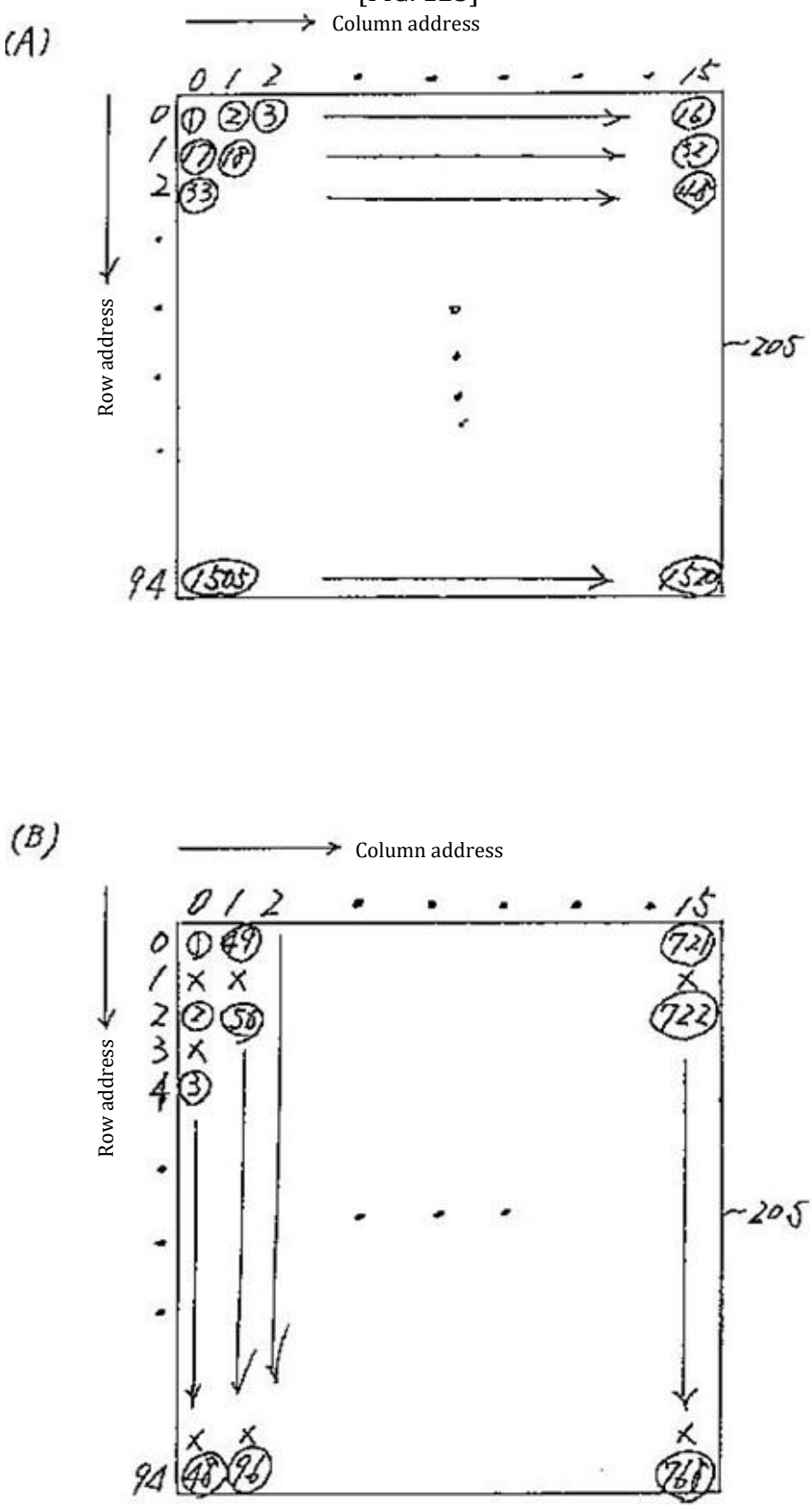
[FIG. 121]



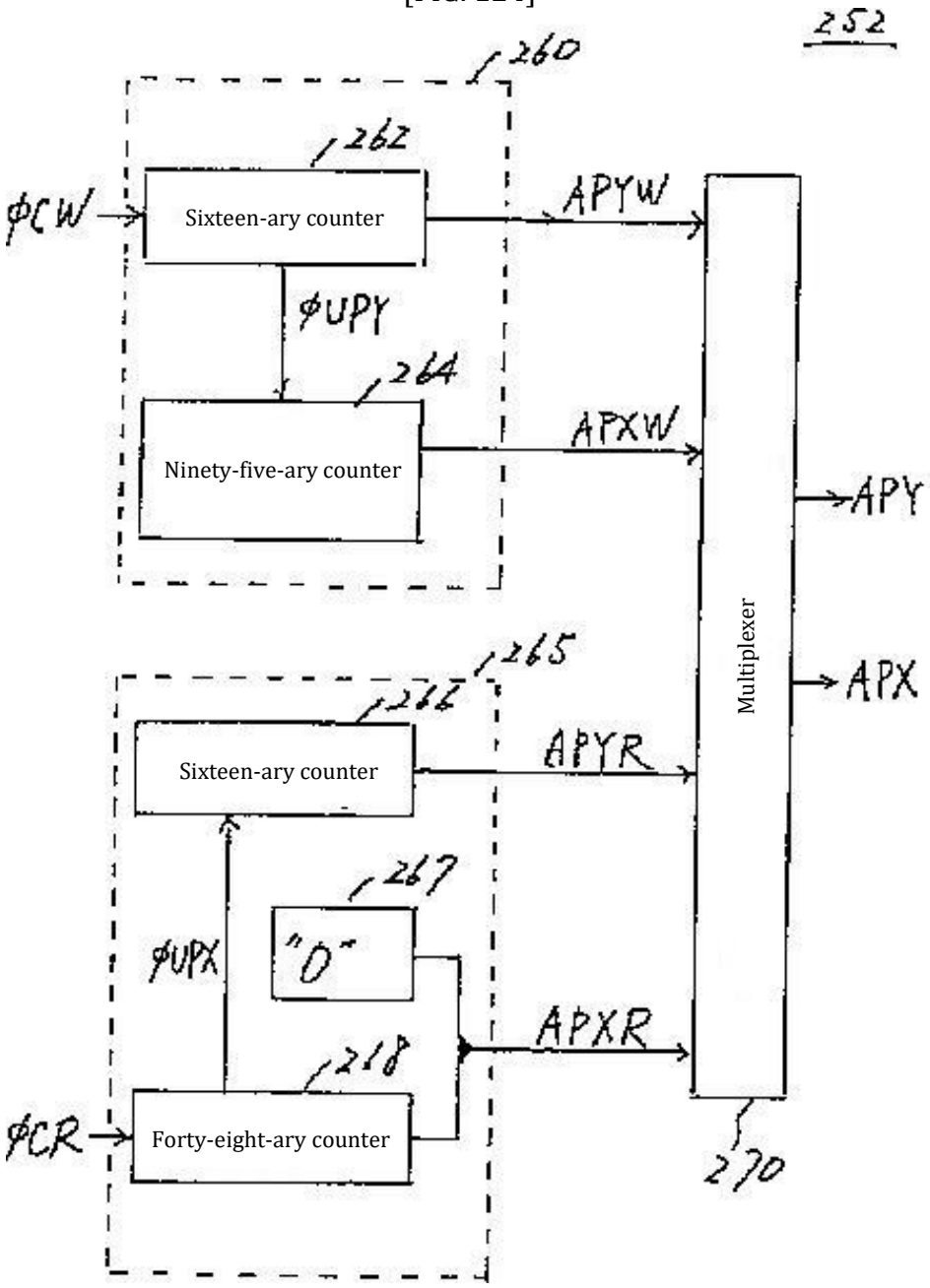
[FIG. 122]



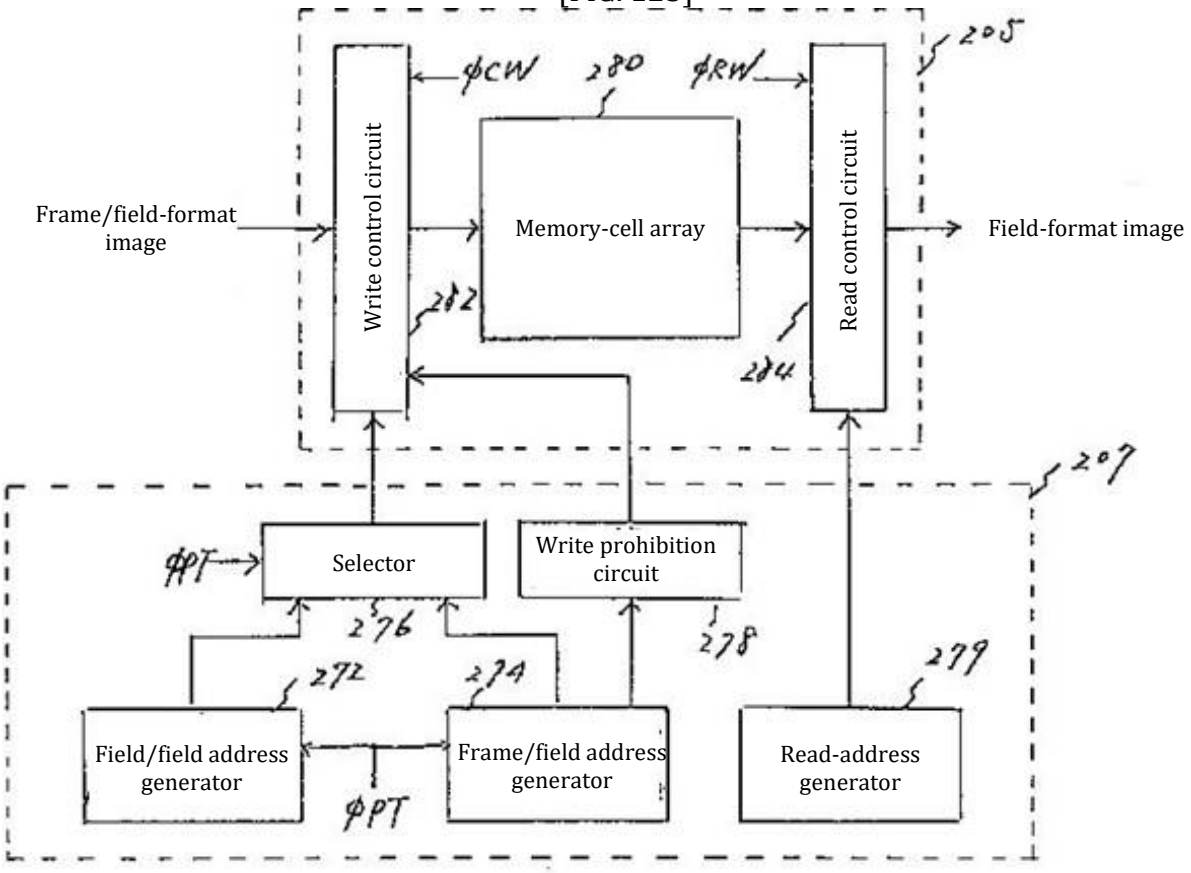
[FIG. 123]


(A)
Column address
Row address
205
(B)
Column address
Row address
205

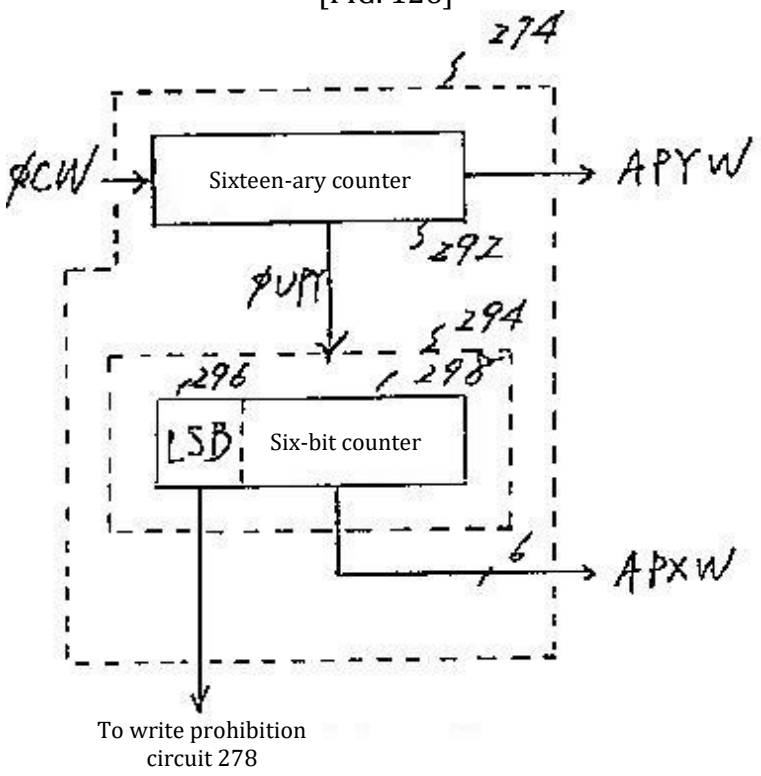
[FIG. 124]


252
260
262
φCW
Sixteen-ary counter
APYW
φUPY
264
Ninety-five-ary counter
APXW
APY
Multiplexer
APX
265
266
Sixteen-ary counter
APYR
267
φUPX
"0"
APXR
218
φCR
Forty-eight-ary counter
270

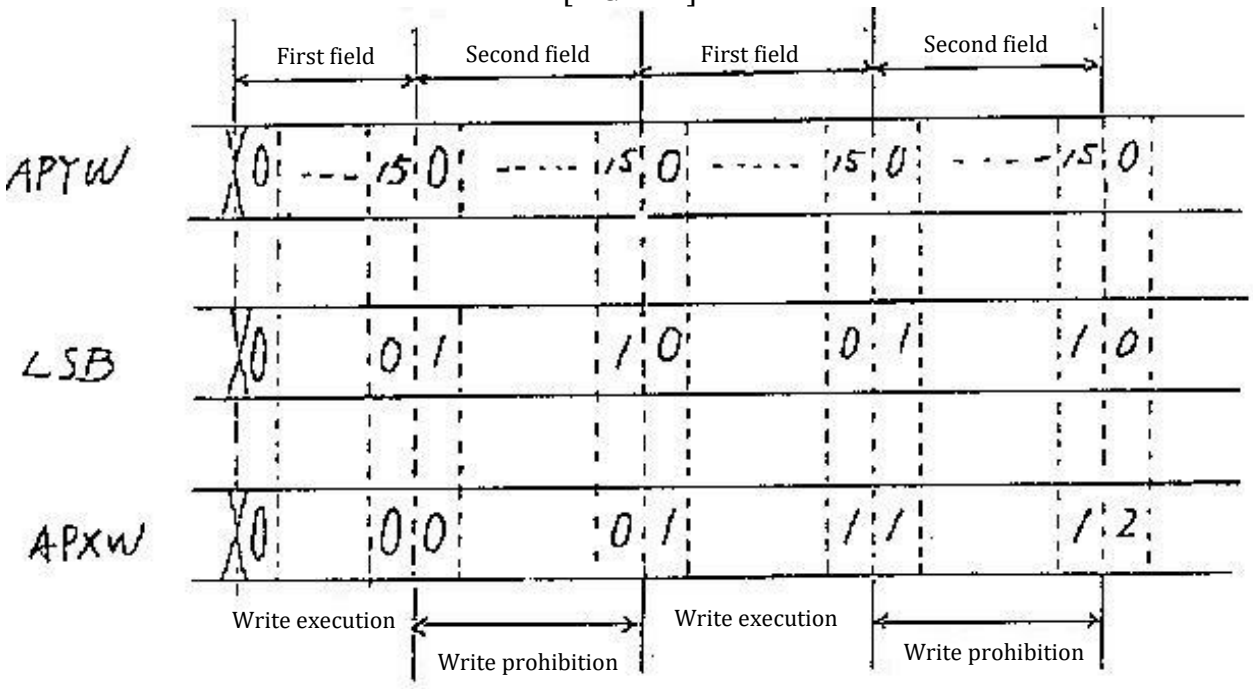
[FIG. 125]


205
φCW
280
φRW
Write control circuit
Memory-cell array
Read control circuit
Frame/field-format image
Field-format image
282
284
207
φPT
Selector
Write prohibition circuit
276
278
272
274
279
Field/field address generator
Frame/field address generator
Read-address generator
φPT

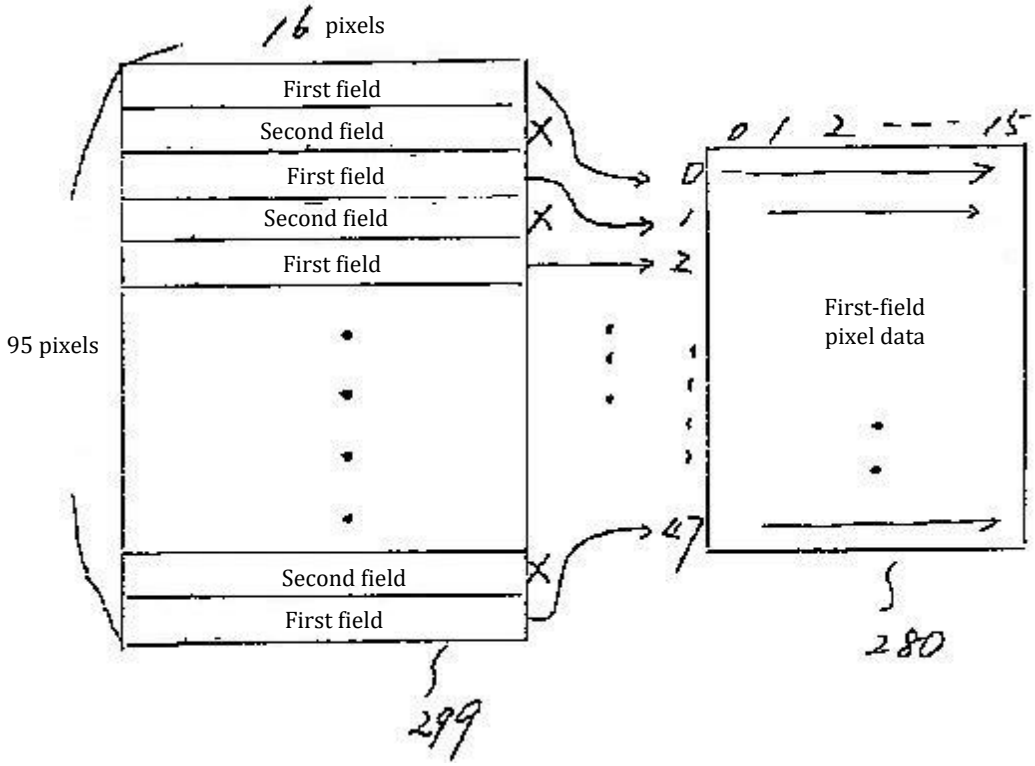
[FIG. 126]



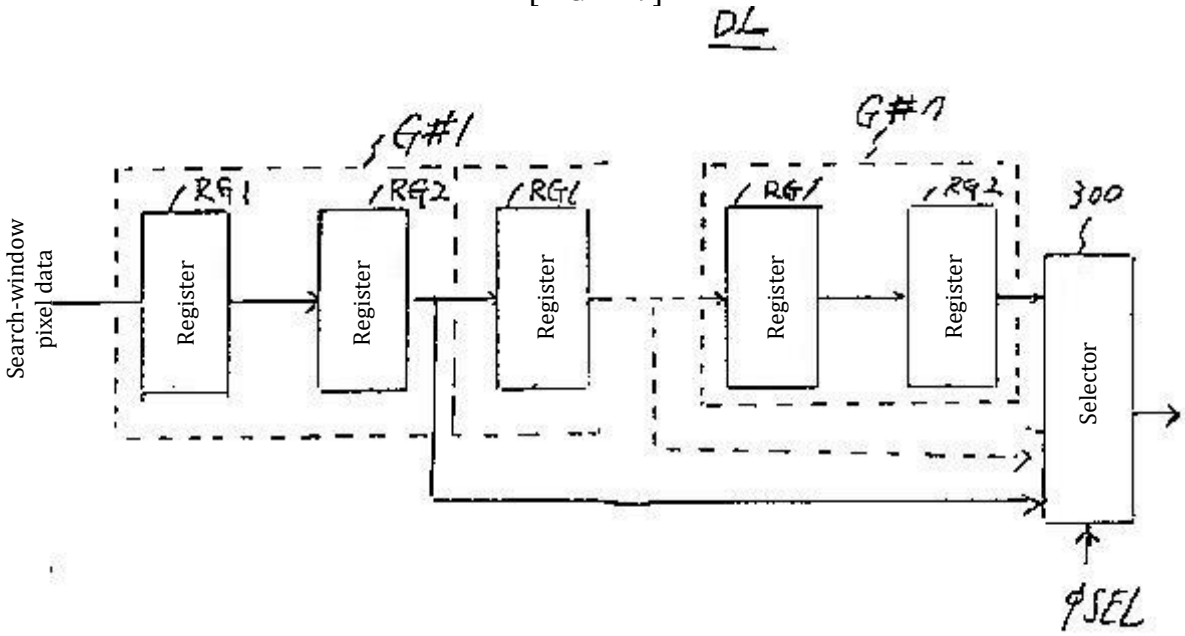
[FIG. 127]



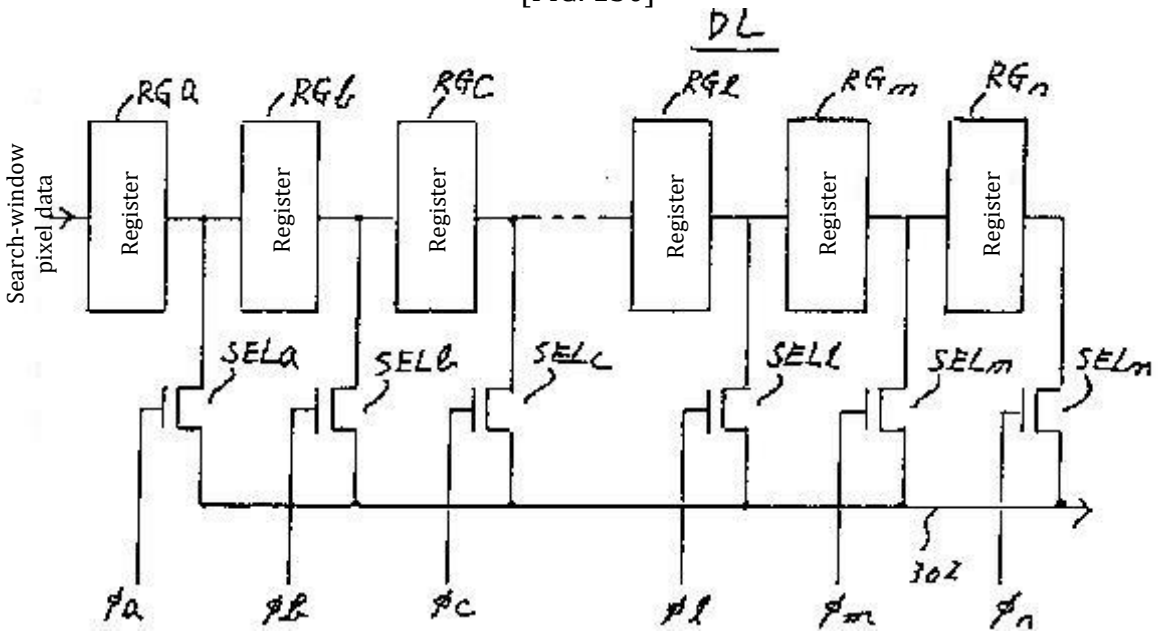
[FIG. 128]



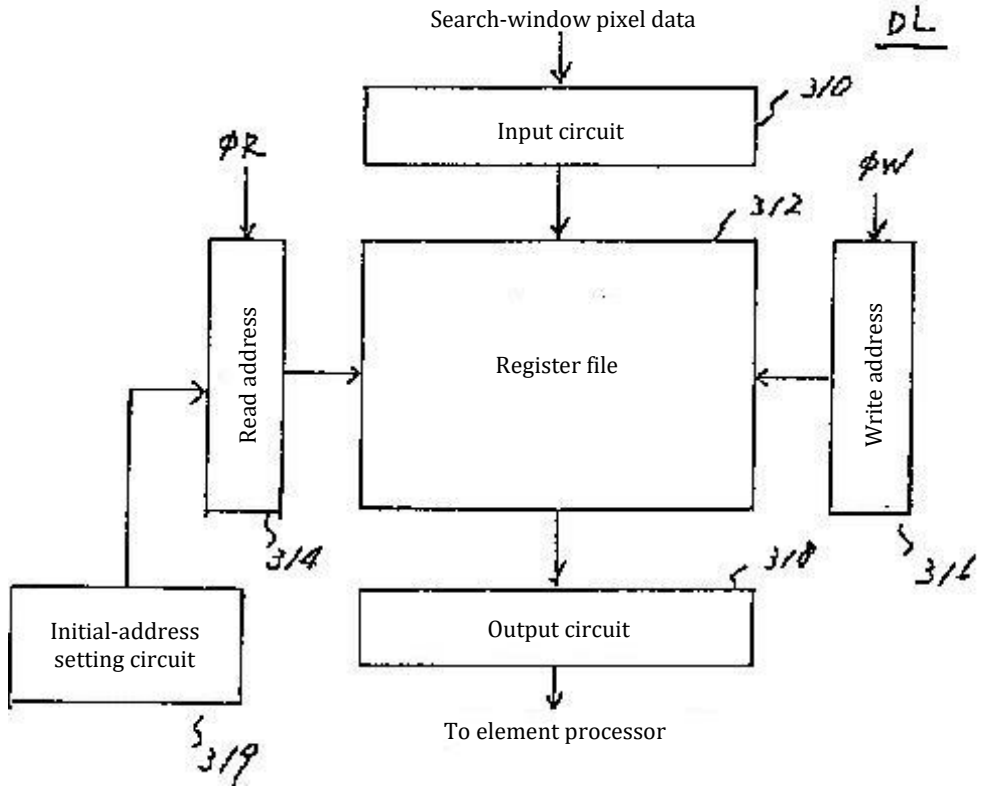
[FIG. 129]



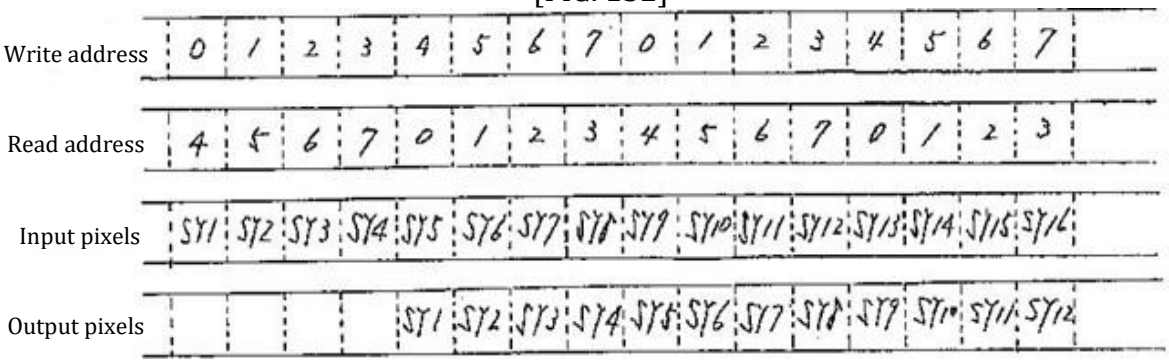
[FIG. 130]



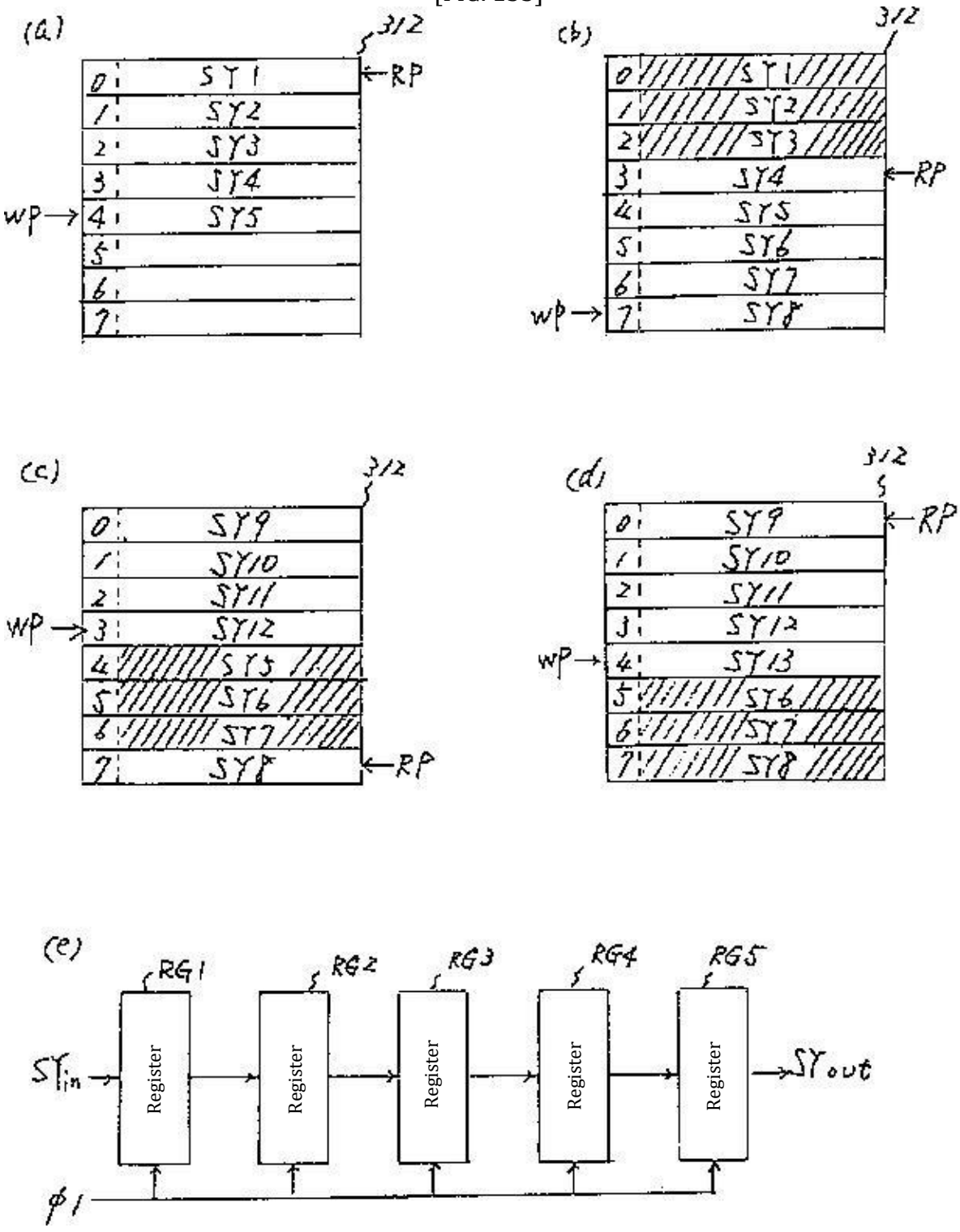
[FIG. 131]



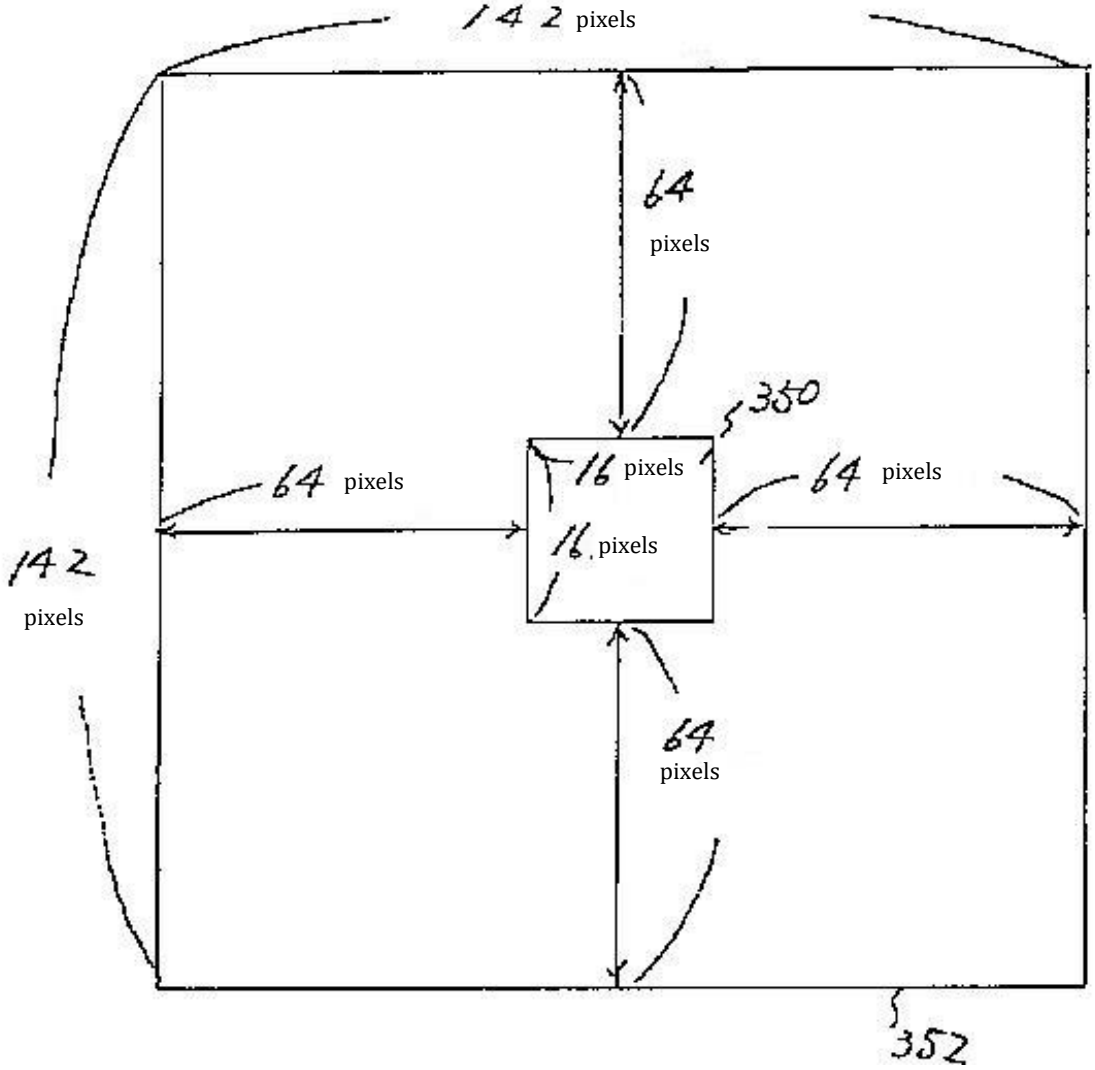
[FIG. 132]


Write address
0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7
Read address
4 5 6 7 0 1 2 3 4 5 6 7 0 1 2 3
Input pixels
SY1 SY2 SY3 SY4 SY5 SY6 SY7 SY8 SY9 SY10 SY11 SY12 SY13 SY14 SY15 SY16
Output pixels
SY1 SY2 SY3 SY4 SY5 SY6 SY7 SY8 SY9 SY10 SY11 SY12

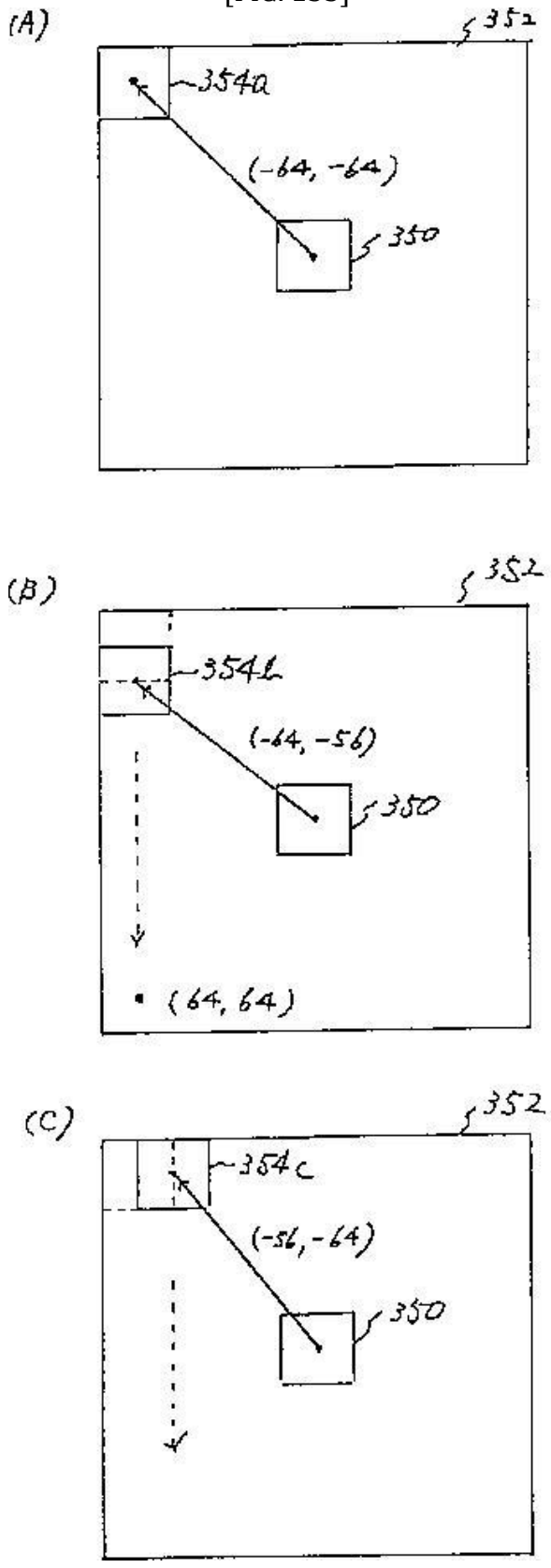
[FIG. 133]


(a)
312
RP
wp
ST1
ST2
ST3
ST4
ST5
(b)
312
RP
wp
ST1
ST2
ST3
ST4
ST5
ST6
ST7
ST8
(c)
312
wp
RP
ST9
ST10
ST11
ST12
ST5
ST6
ST7
ST8
(d)
312
RP
wp
ST9
ST10
ST11
ST12
ST13
ST6
ST7
ST8
(e)
RG1
RG2
RG3
RG4
RG5
STin
Register
STout
φ1

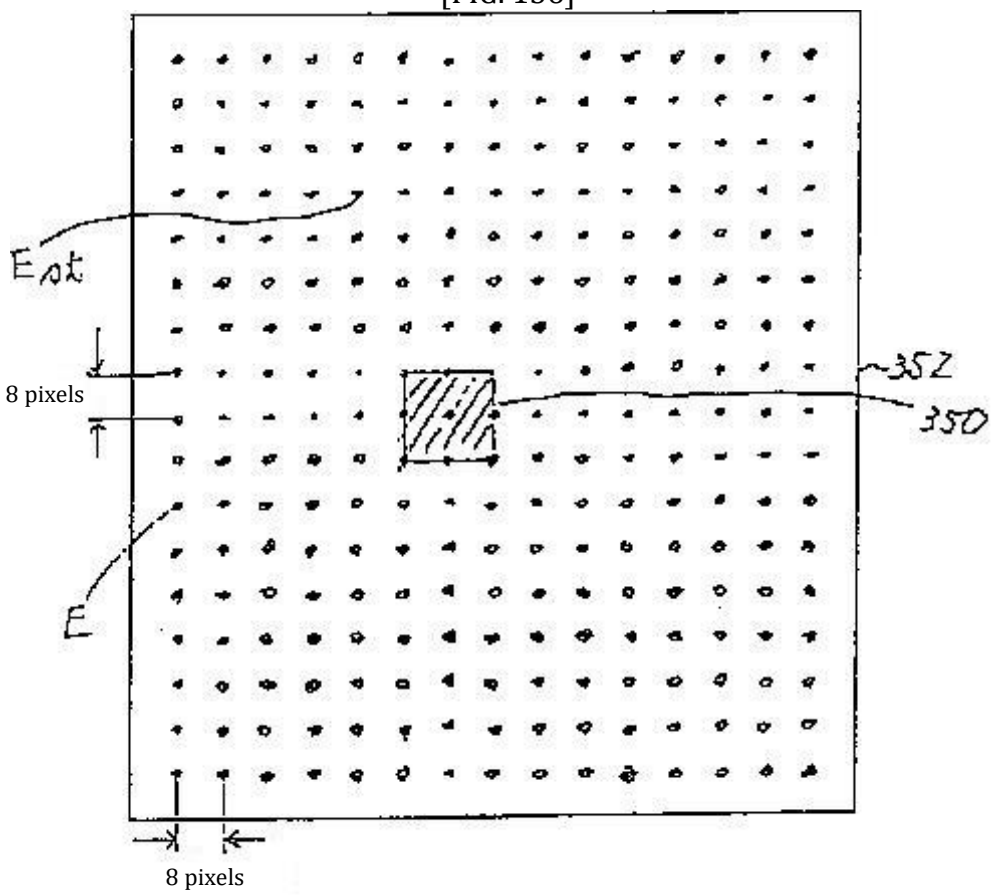
[FIG. 134]


142 pixels
64 pixels
350
64 pixels
16 pixels
64 pixels
16 pixels
142 pixels
64 pixels
352

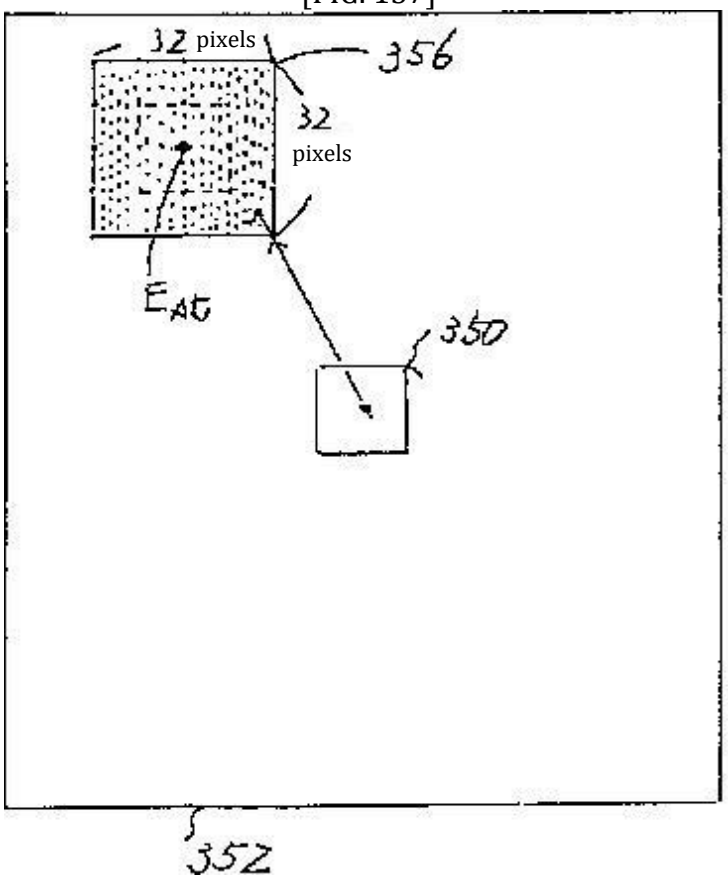
[FIG. 135]



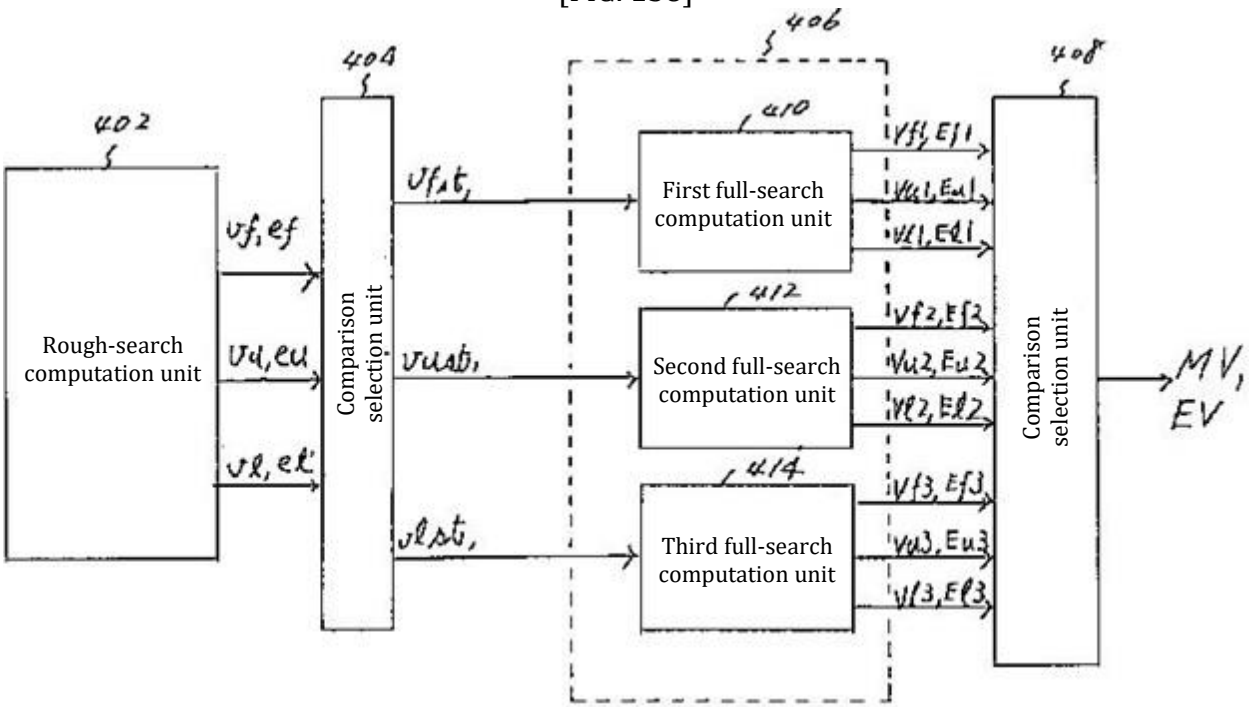
[FIG. 136]



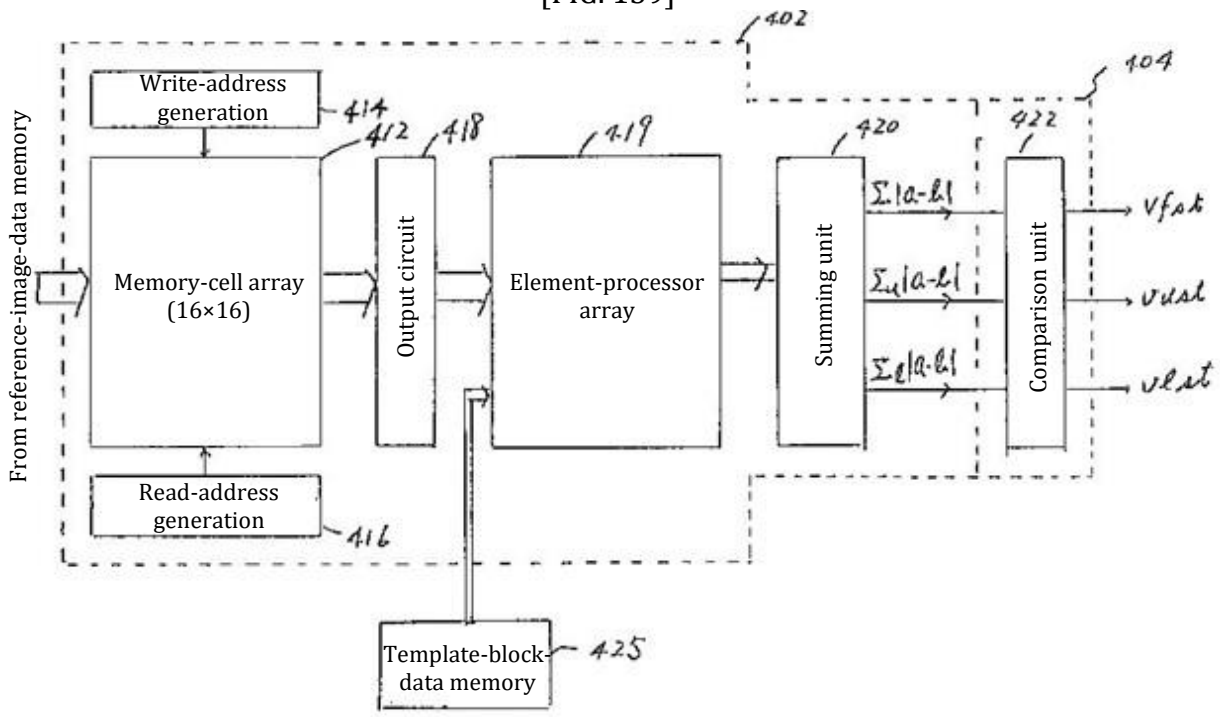
[FIG. 137]



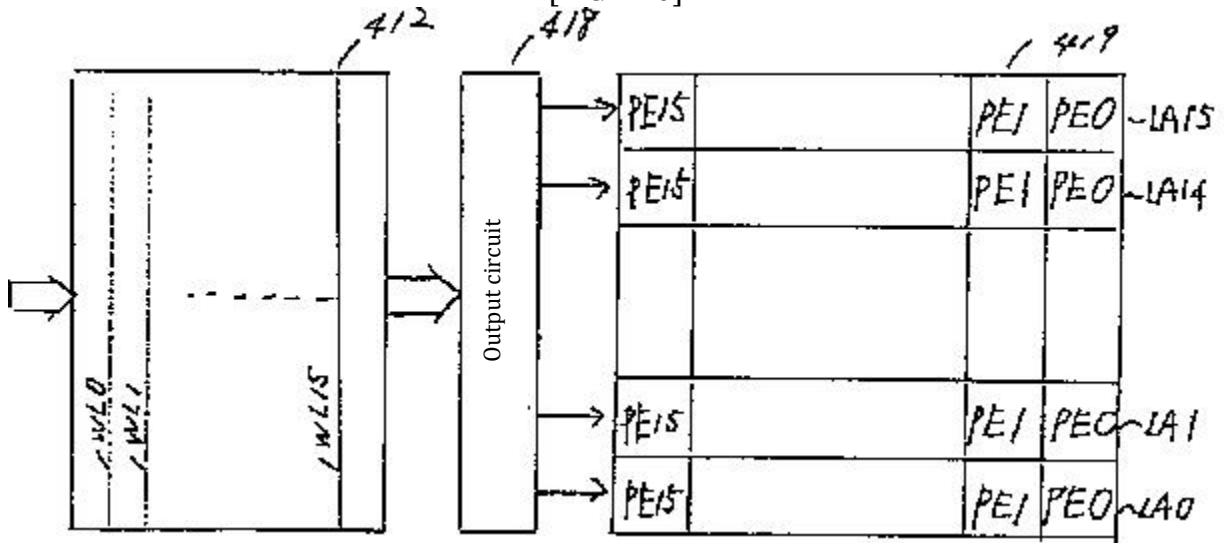
[FIG. 138]



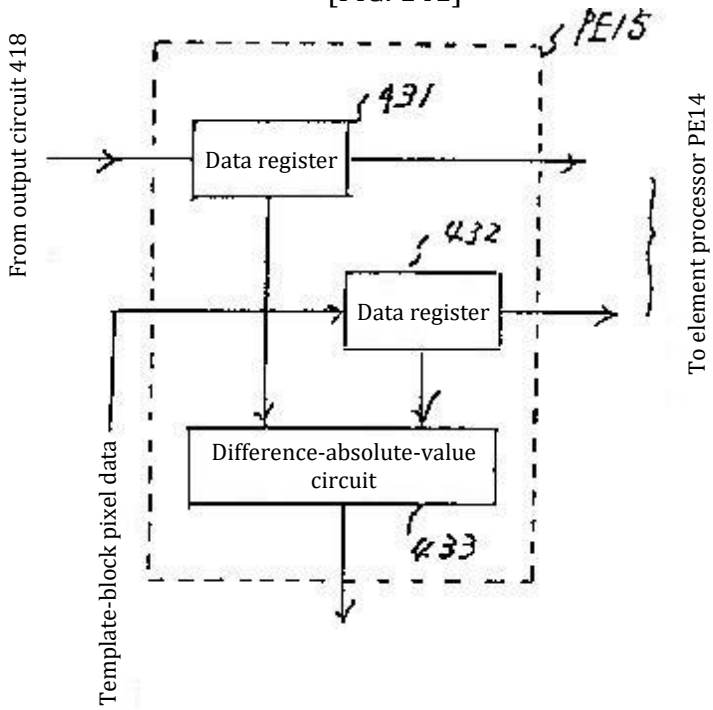
[FIG. 139]



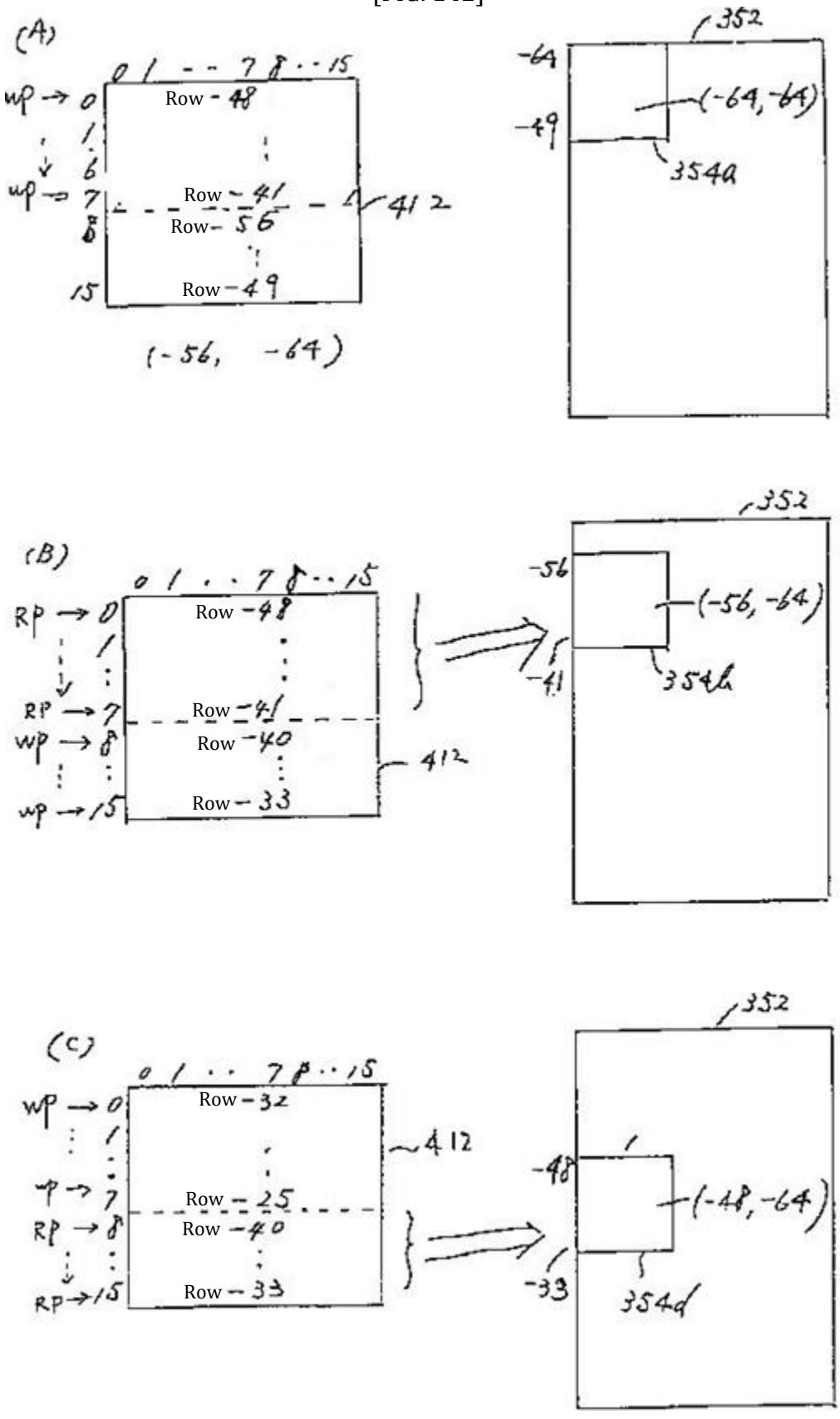
[FIG. 140]



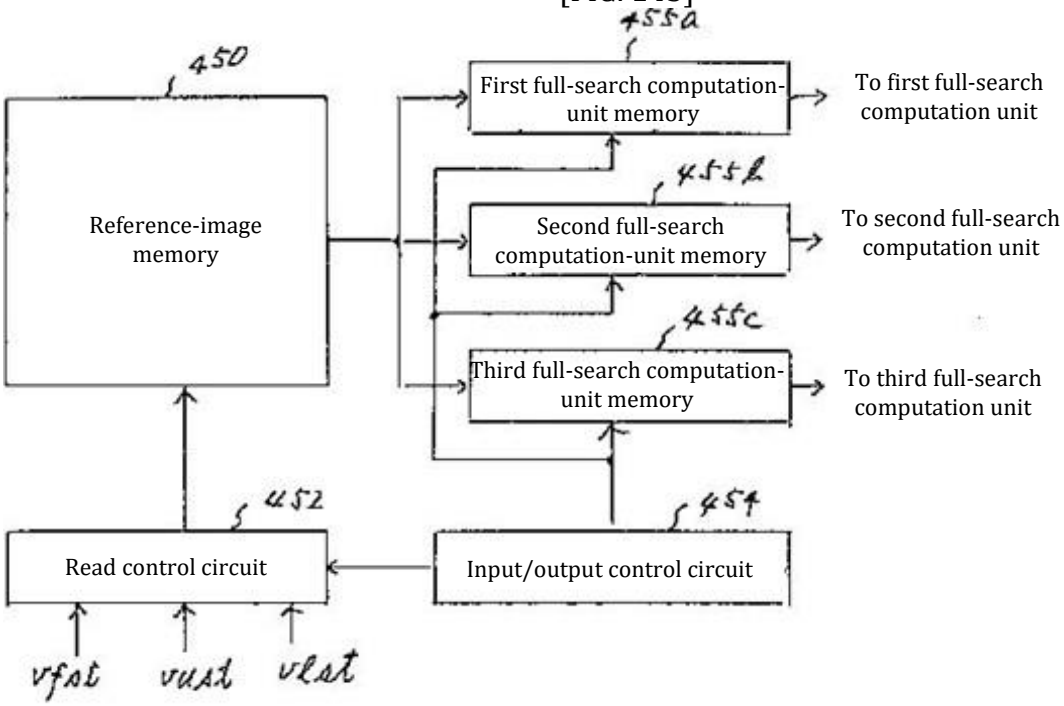
[FIG. 141]


PE15
431
Data register
432
Data register
From output circuit 418
To element processor PE14
Difference-absolute-value circuit
433
Template-block pixel data

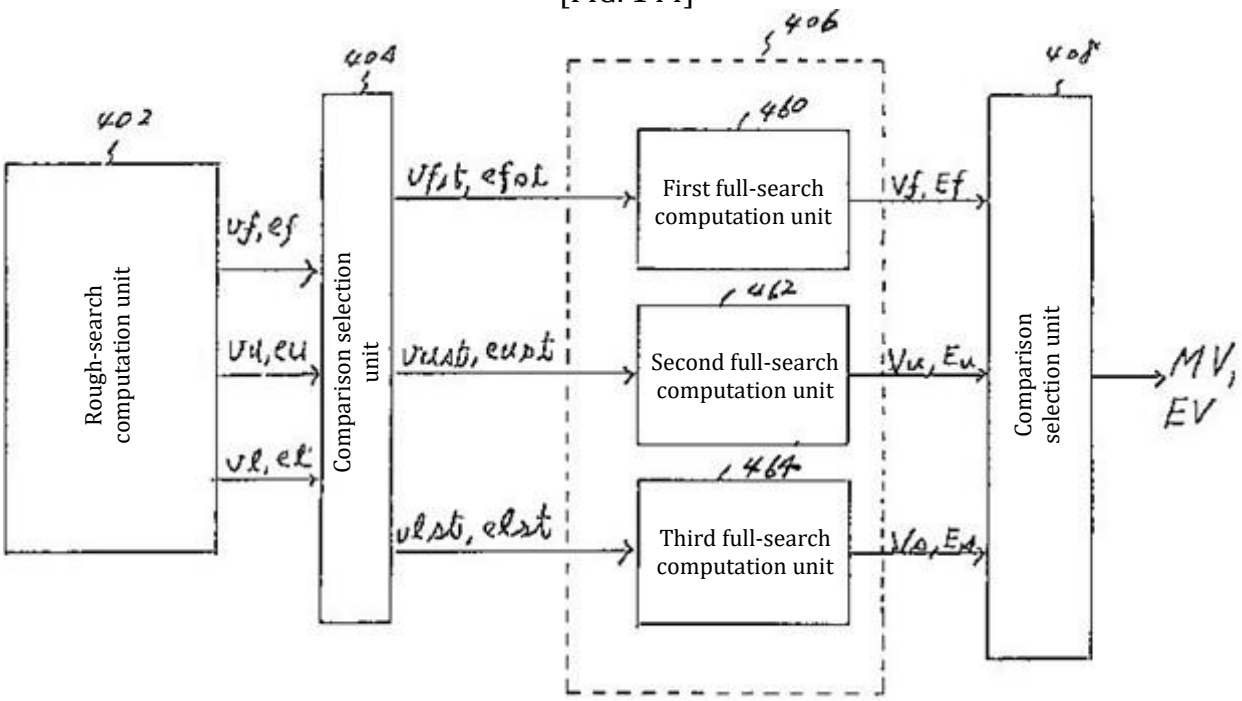
[FIG. 142]


(A)
0 1 - - 7 8 - - 15
WP → 0
Row - 48
1
6
WP → 7
Row - 41
Row - 56
8
15
Row - 49
412
(-56, -64)
352
-64
(-64, -64)
-49
354a
(B)
0 1 - - 7 8 - - 15
RP → 0
Row - 48
1
RP → 7
Row - 41
WP → 8
Row - 40
412
WP → 15
Row - 33
352
-56
(-56, -64)
-41
354b
(C)
0 1 - - 7 8 - - 15
WP → 0
Row - 32
1
412
WP → 7
Row - 25
RP → 8
Row - 40
RP → 15
Row - 33
352
1
-48
(-48, -64)
-33
354d

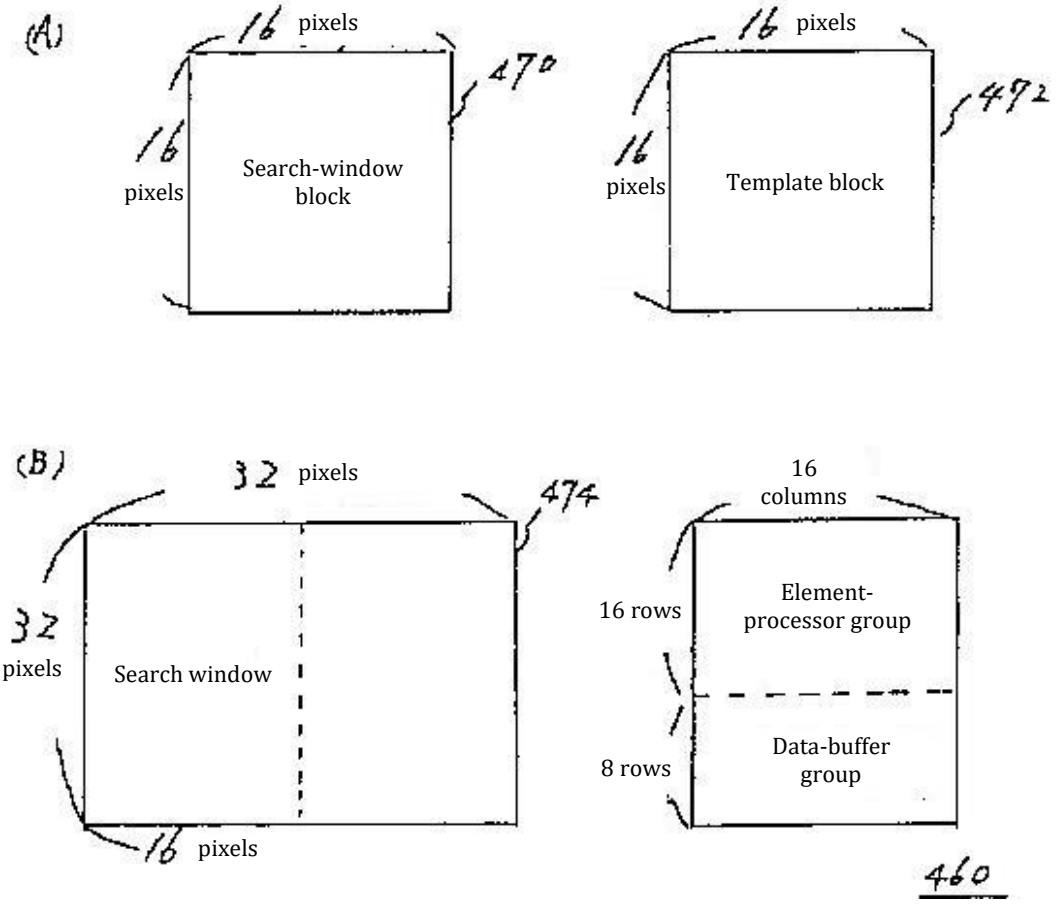
[FIG. 143]



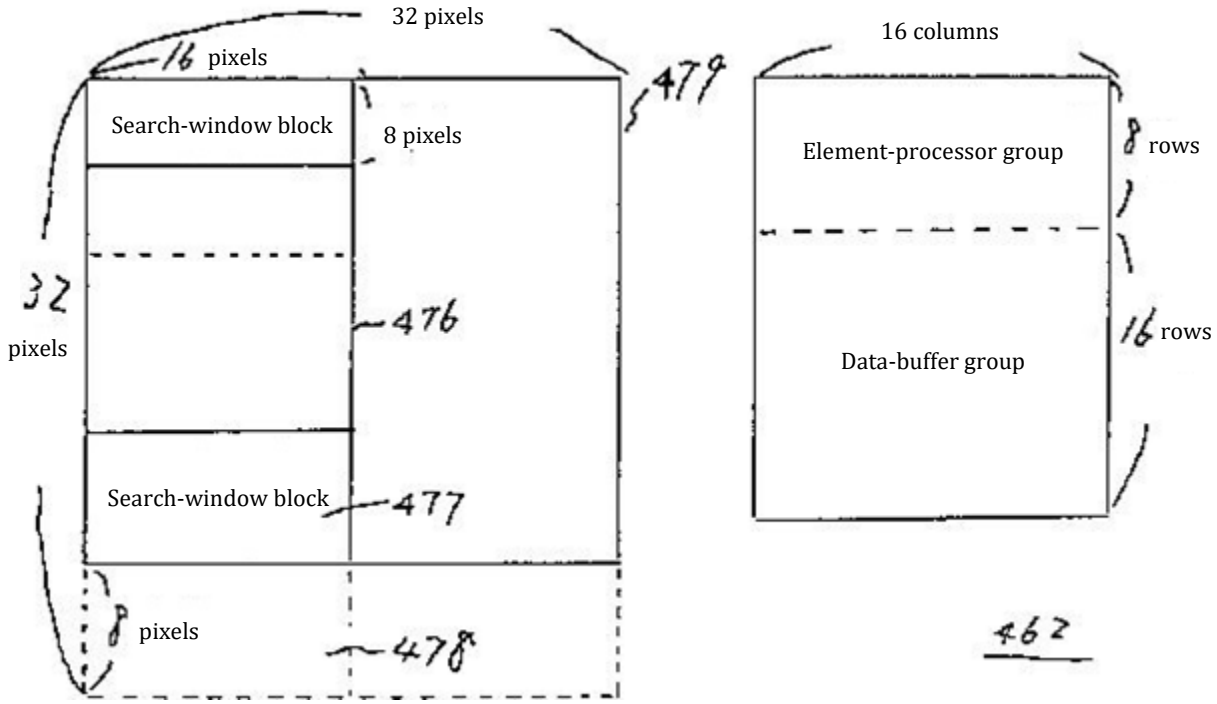
[FIG. 144]



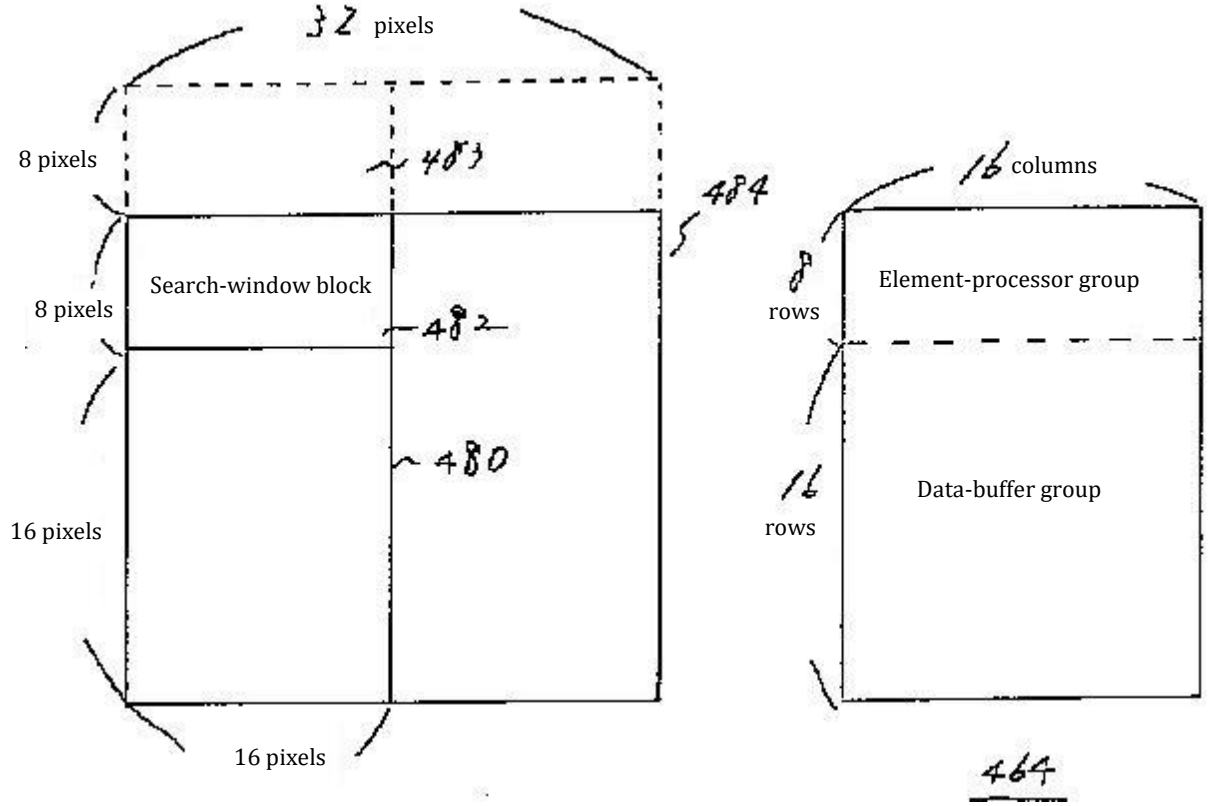
[FIG. 145]



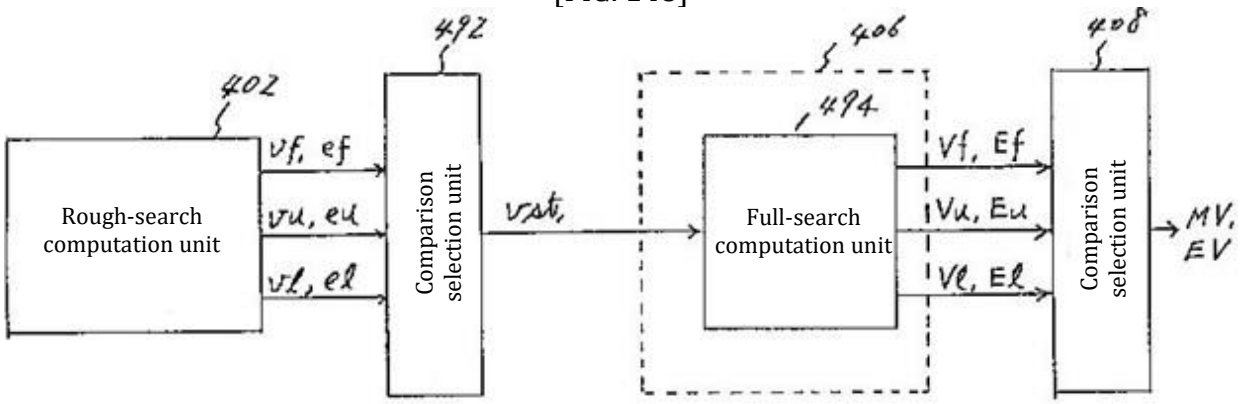
[FIG. 146]



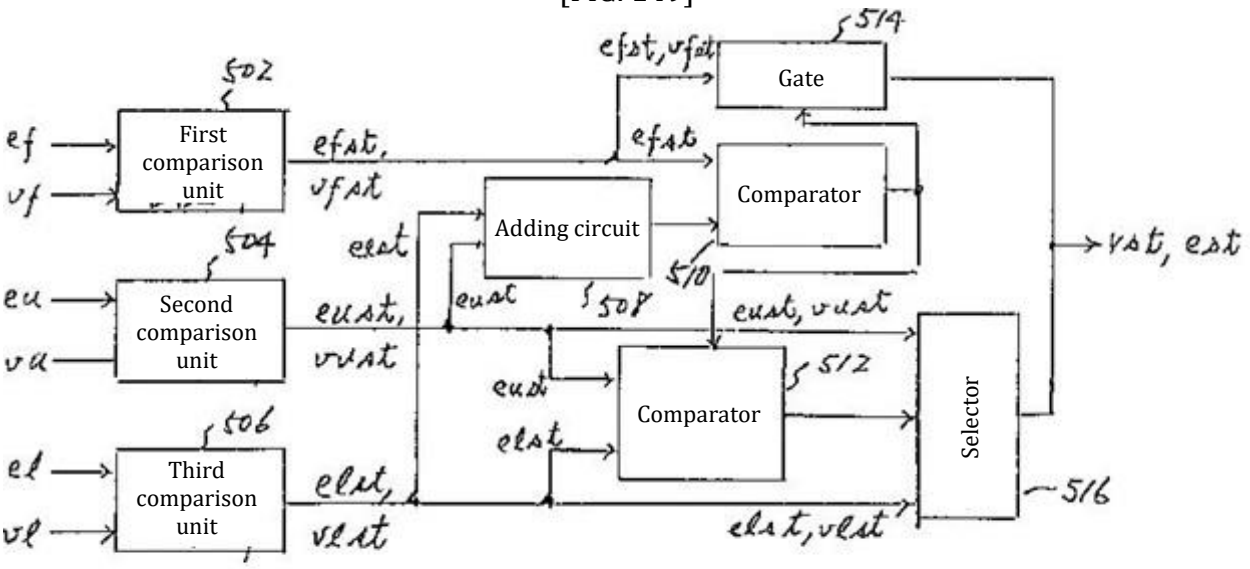
[FIG. 147]



[FIG. 148]

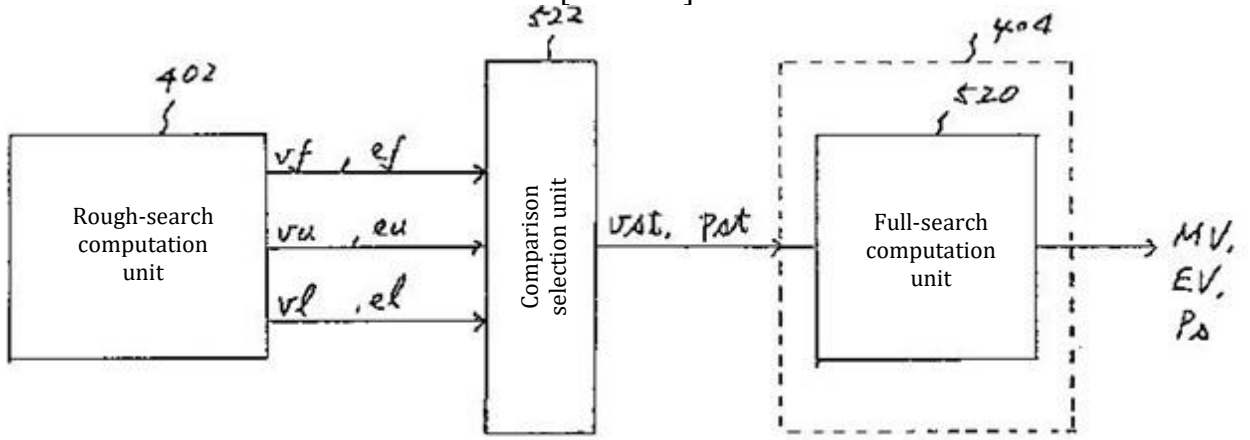


[FIG. 149]

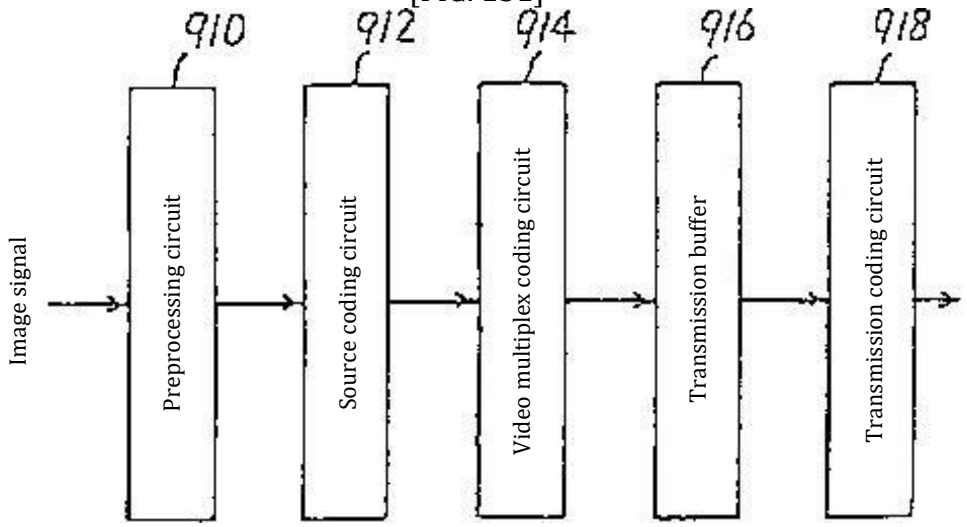


492

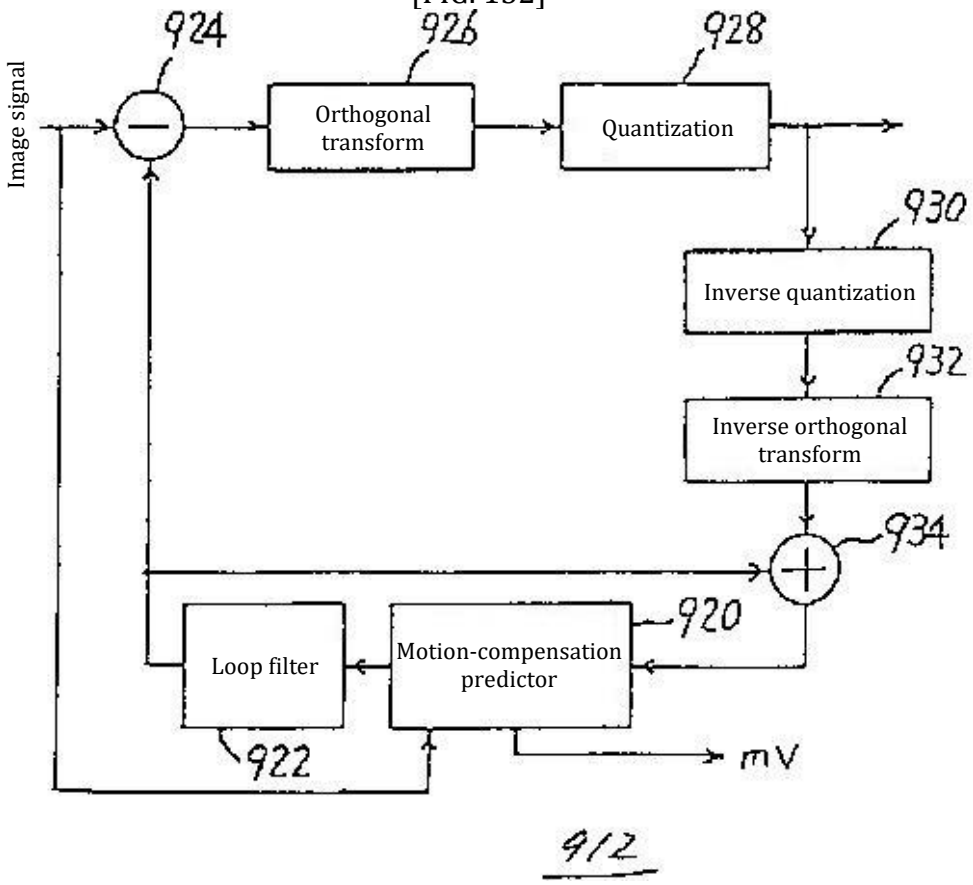
[FIG. 150]



[FIG. 151]

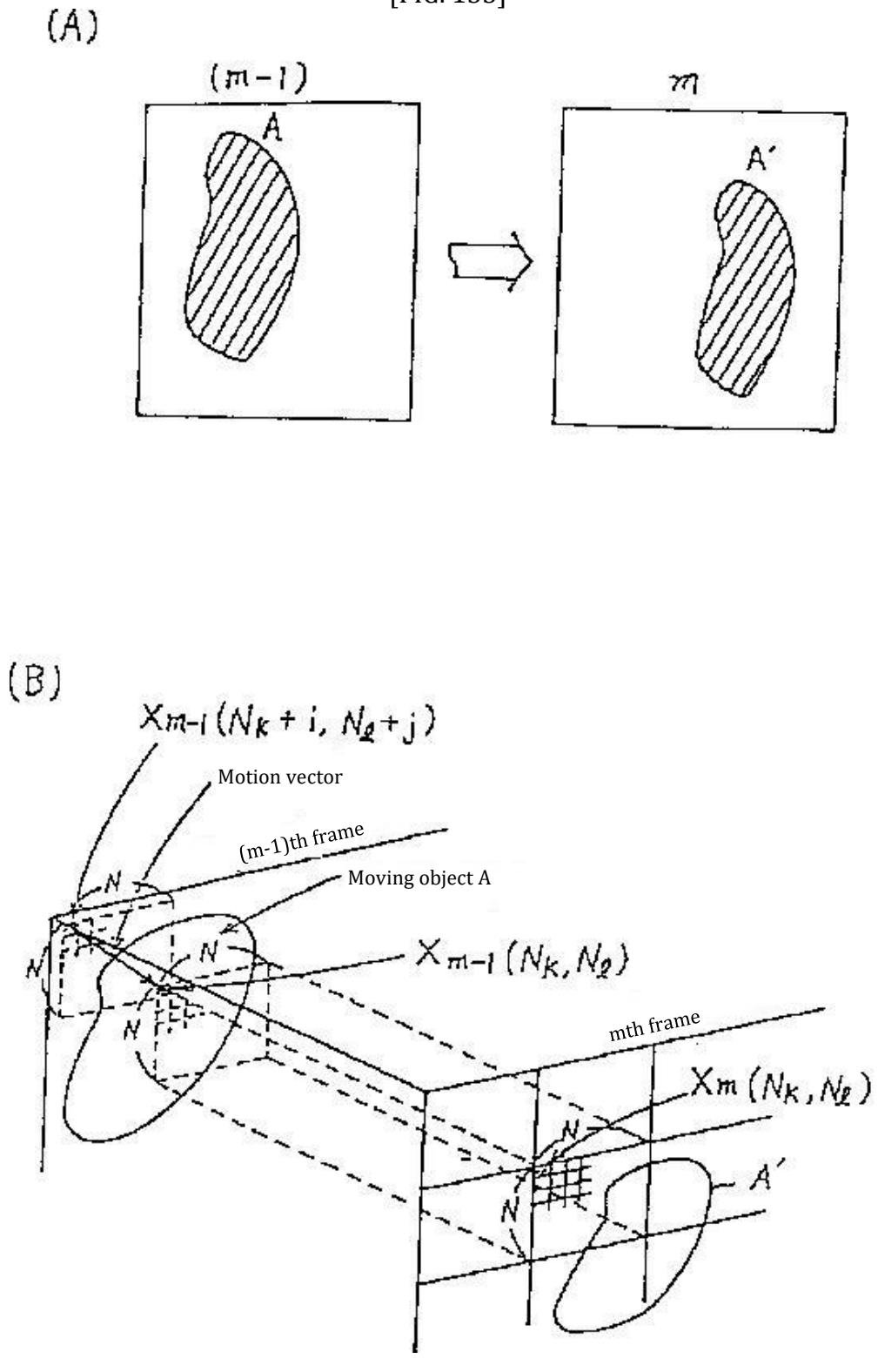


[FIG. 152]

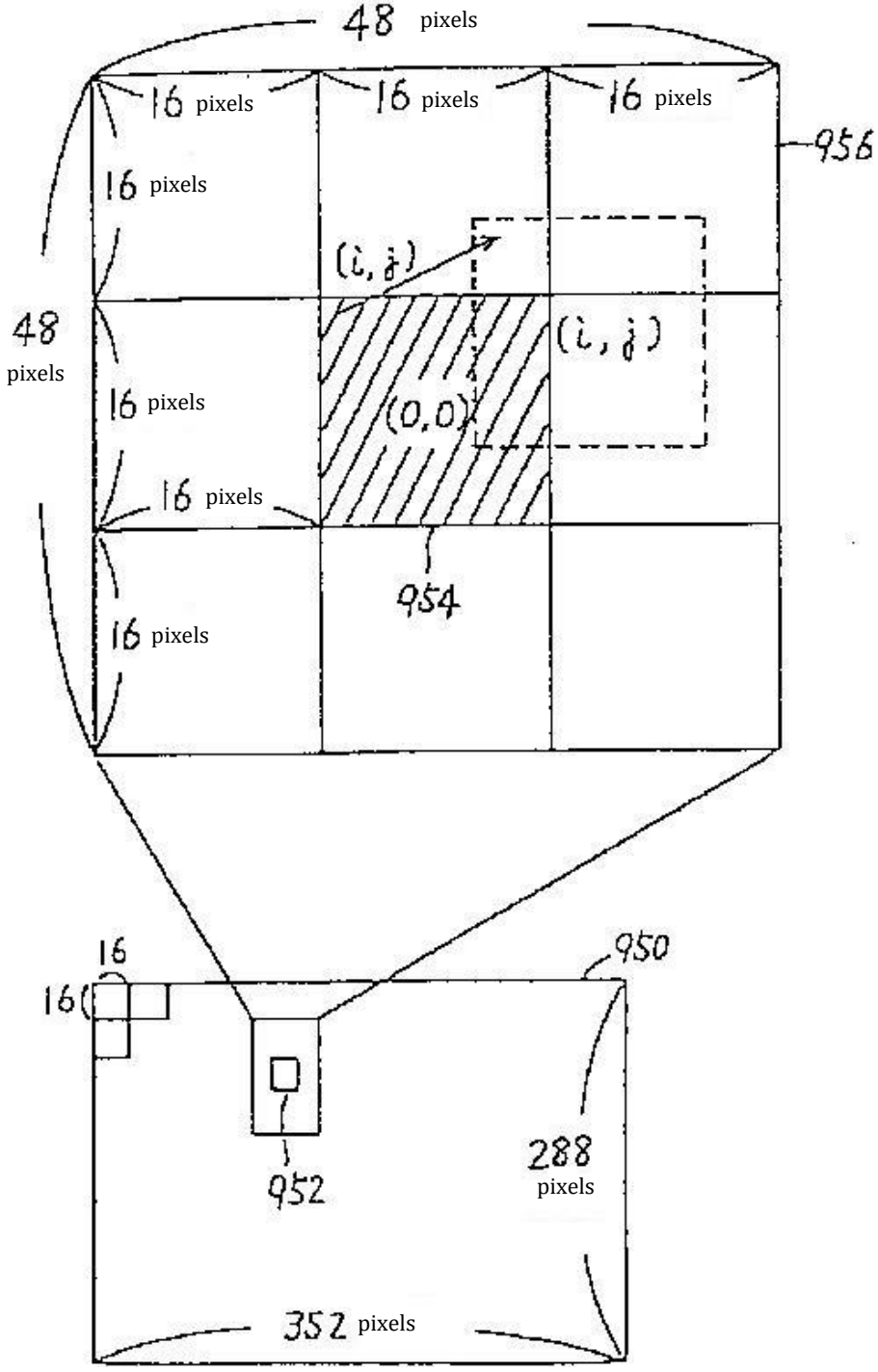


912

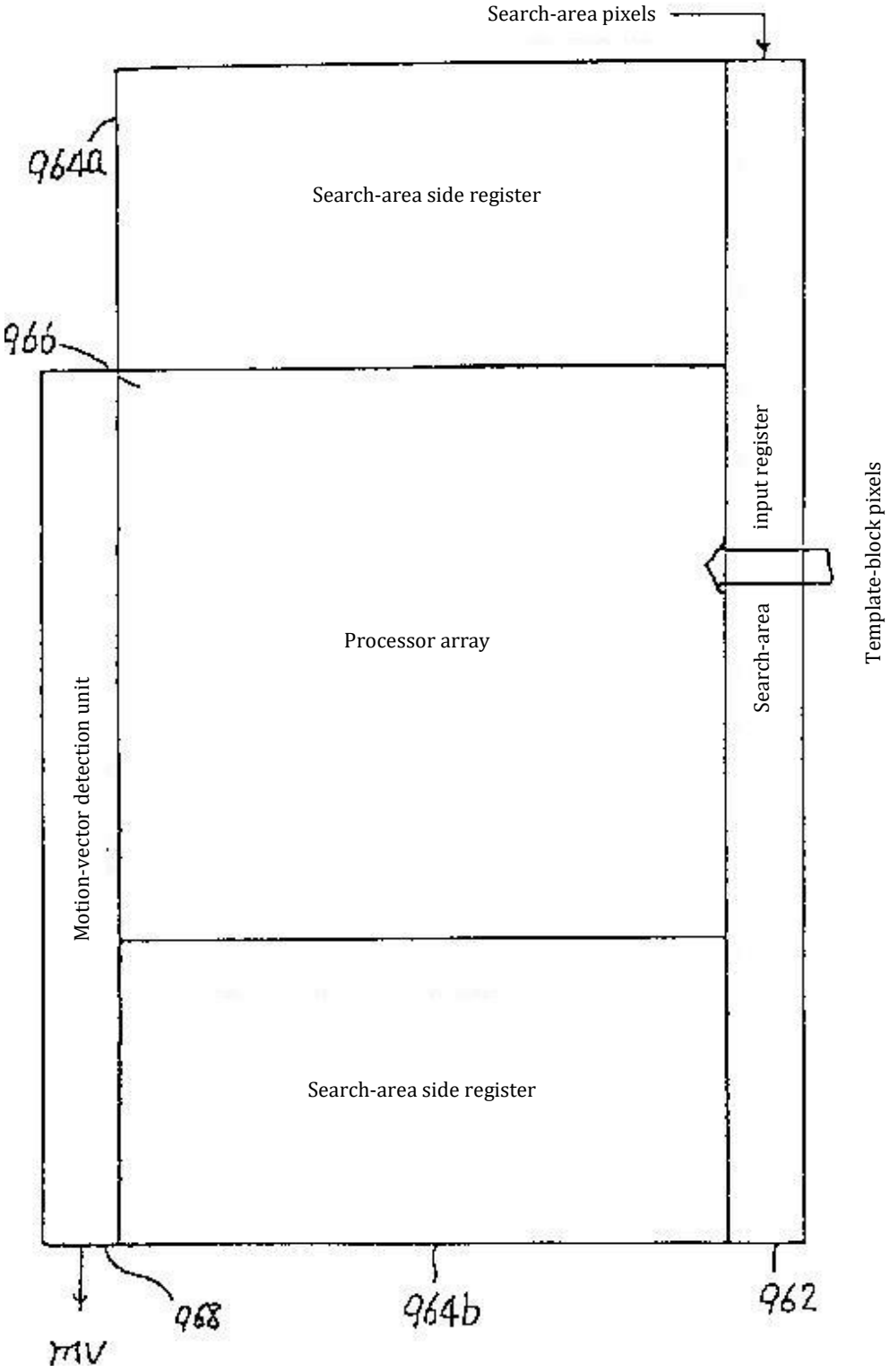
[FIG. 153]


(A)
(m-1)
A
m
A'
(B)
Xm-1 (Nk + i, Nℓ+j)
Motion vector
(m-1)th frame
Moving object A
N
Xm-1 (Nk, Nℓ)
mth frame
Xm (Nk, Nℓ)
A'

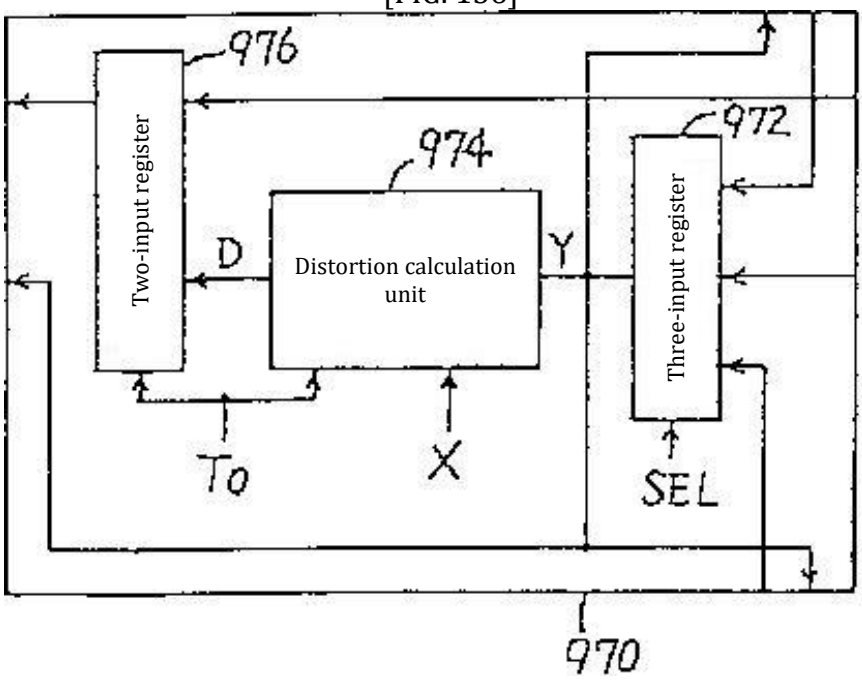
[FIG. 154]


48 pixels
16 pixels
16 pixels
16 pixels
956
16 pixels
(i, j)
48 pixels
16 pixels
(i, j)
(0,0)
16 pixels
954
16 pixels
16
16
950
952
288 pixels
352 pixels

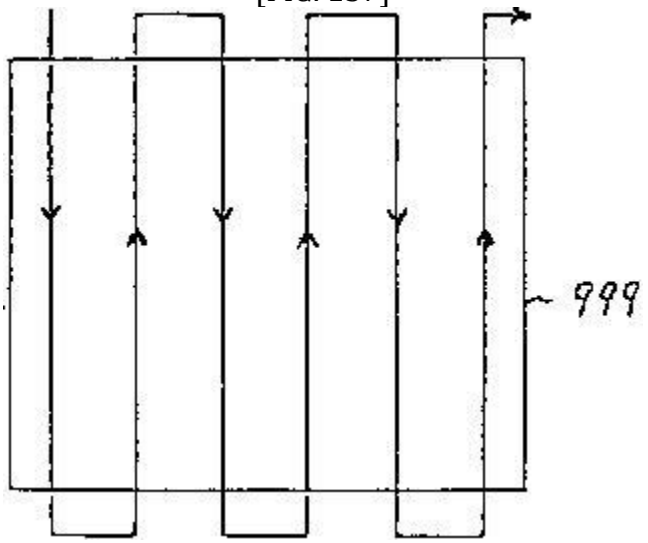
[FIG. 155]


Search-area pixels
964a
Search-area side register
966
input register
Template-block pixels
Processor array
Search-area
Motion-vector detection unit
Search-area side register
MV
968
964b
962

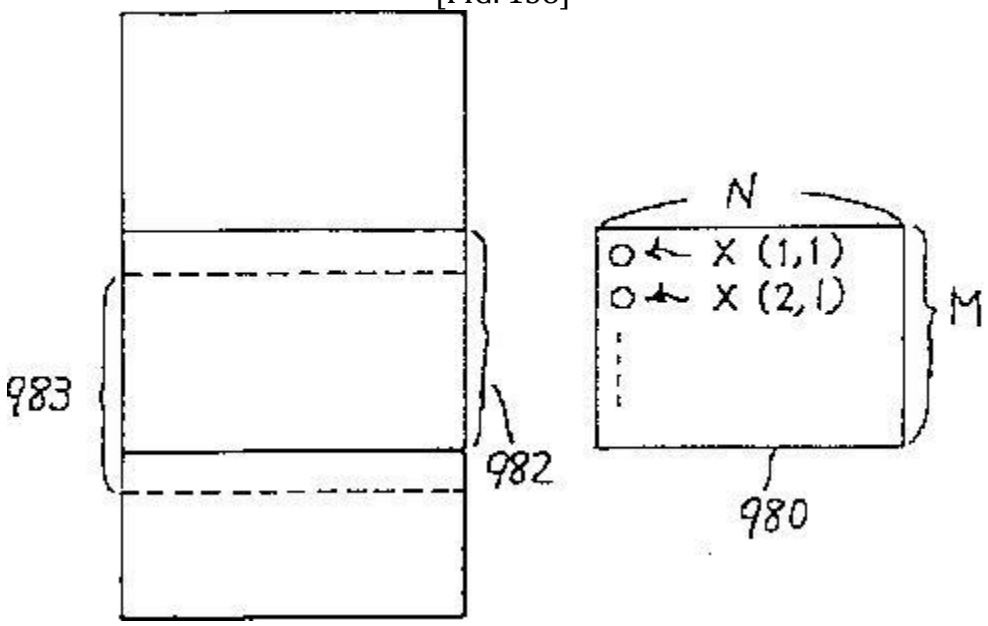
[FIG. 156]



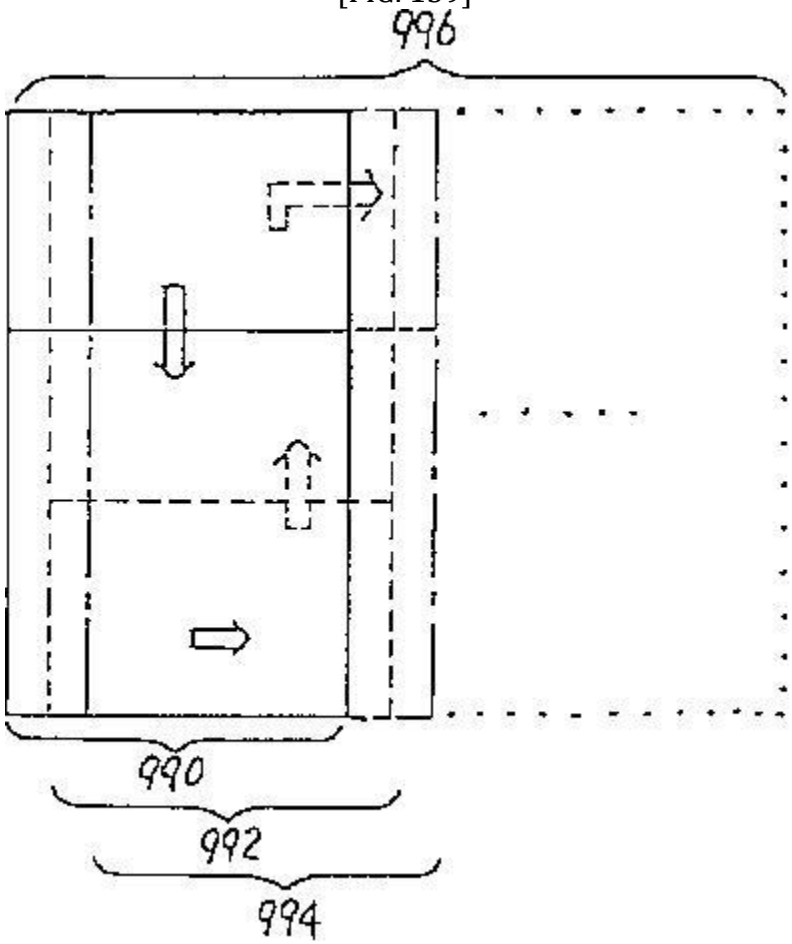
[FIG. 157]



[FIG. 158]



[FIG. 159]



Continued from Front Page

(72) Inventor Shin'ichi Nakagawa
c/o Mitsubishi Electric Corp. System LSI Development Research Laboratory
4-1 Mizuhara, Itami-shi, Hyōgo-ken
(72) Inventor Tetsuya Matsumura
c/o Mitsubishi Electric Corp. System LSI Development Research Laboratory
4-1 Mizuhara, Itami-shi, Hyōgo-ken
(72) Inventor Satoshi Kumaki
c/o Mitsubishi Electric Corp. System LSI Development Research Laboratory
4-1 Mizuhara, Itami-shi, Hyōgo-ken
(72) Inventor Atsuo Hanami
c/o Mitsubishi Electric Corp. System LSI Development Research Laboratory
4-1 Mizuhara, Itami-shi, Hyōgo-ken

(19)日本国特許庁（ＪＰ）　(12) **公開特許公報（Ａ）**　(11)特許出願公開番号

特開平7－250328

(43)公開日　平成７年(1995)９月26日

(51)Int.Cl.[6]	識別記号	庁内整理番号	ＦＩ	技術表示箇所
Ｈ０４Ｎ　7/32				
			Ｈ０４Ｎ　7/ 137	Ｚ

審査請求　未請求　請求項の数28　ＦＤ　（全103頁）

(21)出願番号　特願平6－321525

(22)出願日　平成６年(1994)11月29日

(31)優先権主張番号　特願平6－5432

(32)優先日　平６(1994)１月21日

(33)優先権主張国　日本（ＪＰ）

(71)出願人　000006013
三菱電機株式会社
東京都千代田区丸の内二丁目２番３号

(72)発明者　石原　和哉
兵庫県伊丹市瑞原４丁目１番地　三菱電機株式会社システムエル・エス・アイ開発研究所内

(72)発明者　浦本　紳一
兵庫県伊丹市瑞原４丁目１番地　三菱電機株式会社システムエル・エス・アイ開発研究所内

(74)代理人　弁理士　深見　久郎　（外３名）

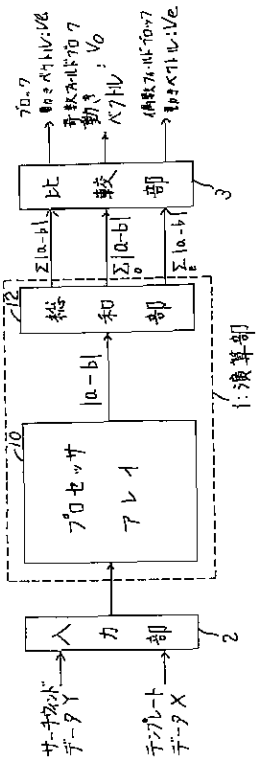
最終頁に続く

(54)【発明の名称】　動きベクトル検出装置

(57)【要約】

【目的】　少ないハードウェア量で複数の予測モードに従って高速に動き画像予測補償に用いられる動きベクトルを検出することのできる動きベクトル検出装置を提供する。

【構成】　プロセッサアレイ１０は、現画像画素ブロックであるテンプレートブロックの各画素に対応してマトリックス状に配列され、各々が対応の参照画像画素ブロックであるサーチウィンドウブロック画素データを格納し、テンプレートブロック画素データとの評価関数値成分を求める要素プロセサを含む。総和部１２は各要素プロセサから与えられた評価関数値成分を複数の予測モードそれぞれに応じて分類し、各分類ごとに総和して各予測モードに対する評価関数値を生成する。比較部３は総和部から与えられた評価関数値に従って最も良い類似度を与える評価関数値に対応する変位ベクトルを各予測モードそれぞれによる動きベクトルとして決定する。これにより複数の予測モードに従う動きベクトルを同時に検出することができる。



【特許請求の範囲】
【請求項１】　現画面画像と参照画面画像とのブロックマッチング処理により、動き補償付予測符号化に用いられる動きベクトルを求めるための動きベクトル検出装置であって、
前記現画面内の動きベクトル検出対象となる所定のサイズの現画像ブロックと前記参照画面内の前記現画像ブロックに関連する領域内の複数の参照画像ブロック各々との類似度を示すとともに予め定められた複数の予測モード各々に従って決定される評価関数値を各参照ブロックごとにかつ前記複数の予測モード並列に求めるための評価手段、および前記評価手段からの各参照ブロックごとおよび各前記複数の予測モードごとの評価関数値を受け、前記現画像ブロックに対する各前記予測モードの動きベクトルを並列に決定する動きベクトル決定手段を備える、動きベクトル検出装置。
【請求項２】　前記評価手段は、
前記現画像ブロックに含まれる画素に対応して配置され、参照画像ブロック内の対応の画素データと前記現画像ブロック内の対応の画素データとを一時的に格納する格納手段を含み、該格納手段に格納された画素データに前記評価関数の一部をなす演算を行なう複数の第１の演算手段と、
前記複数の第１の演算手段の出力を前記複数の予測モード各々に対応して分類し、各分類ごとに前記第１の演算手段の出力に対し前記評価関数の残りの演算を並列に行なう第２の演算手段を含む、請求項１記載の動きベクトル検出装置。
【請求項３】　前記複数の第１の演算手段は、各々が一方方向に沿ってのみ画素データを転送するように配置されかつ前記現画像ブロックに対応する行および列のマトリックス状に配置される複数のプロセッサを備える、請求項２記載の動きベクトル検出装置。
【請求項４】　前記現画面および参照画面は、偶数フィールドと奇数フィールドとを含むフレームで構成され、前記複数の予測モードは、前記現画像ブロック内の奇数フィールドに含まれる画素を用いる動き予測モードと、偶数フィールドに含まれる画素を用いる動き予測モードと、フレームに含まれる画素を用いる動き予測モードとを含む、請求項１ないし３のいずれかに記載の動きベクトル検出装置。
【請求項５】　前記現画面および参照画面の各々は、１つのフィールドにより構成され、
前記複数の予測モードは、前記現画像ブロックの画面垂直方向に沿って２等分し、該２等分された上半分のブロックの画素を用いる動き予測モードと、２等分された現画像ブロックの下半分のブロックの画素を用いた動き予測モードと、前記現画像ブロックの画素すべてを用いる動き予測モードとを含む、請求項１ないし３のいずれかに記載の動きベクトル検出装置。
【請求項６】　前記第２の演算手段は、
前記現画像ブロックをフレーム画面に関して垂直方向に２等分した上下ブロックおよび偶数フィールドの画素ブロックおよび奇数フィールドの画素ブロックの４ブロックに分割し、各分割ブロックごとに対応の第１の演算手段の出力を並列に受け、モード指定信号に従って前記４分割ブロックの構成を切換えてフレーム内画素を用いるフレーム予測モード、奇数フィールド内画素を用いる奇数フィールド予測モードおよび偶数フィールド内の画素を用いる偶数フィールド動き予測モードまたはフィールド内画素を用いる動き予測モード、現画像ブロックの上半分の画素を用いる上半分動き予測モードおよび現画像ブロックの下半分の画素を用いる下半分動き予測モードのいずれかの３分類を実現する切換え手段を含む、請求項１ないし５のいずれかに記載の動きベクトル検出装置。
【請求項７】　画面の形式を指定する形式指定信号に応答して、入力画像データの画面形成形式を前記形式指定信号が指定する形式に設定して前記評価手段へ与える入力変換手段をさらに含む、請求項１ないし６のいずれかに記載の動きベクトル検出装置。
【請求項８】　前記複数の第１の演算手段は、実質的に１次元状アレイを構成するように配置され、入力した現画像画素データおよび参照画像ブロック画素データをともに一方方向に沿ってのみ伝達する複数のプロセッサを備え、
前記複数のプロセッサの所定数ごとに配置され、参照画像画素データを格納するとともに格納した画素データを格納した順に順次出力するとともにその有効格納画素数が可変なバッファ手段をさらに備える、請求項１ないし７のいずれかに記載の動きベクトル検出装置。
【請求項９】　前記関連する領域内の所定の条件を満足する参照画像ブロックに対して前記複数の予測モード各々に従って各評価関数値を並列に求める第３の演算手段と、
前記第３の演算手段の出力に従って前記評価手段が演算すべき参照画像ブロックの領域を前記複数の予測モードそれぞれに対して指定する領域指定手段をさらに備え、前記評価手段は指定された領域に対し複数の予測モードで評価値を算出する、請求項１ないし８のいずれかに記載の動きベクトル検出装置。
【請求項１０】　前記領域指定手段は、前記複数の予測モード各々に対して演算すべき参照画像領域を指定する、請求項９記載の動きベクトル検出装置。
【請求項１１】　前記領域指定手段は、前記複数の予測モードに対し共通に１つの参照画像領域を指定する、請求項９記載の動きベクトル検出装置。
【請求項１２】　前記領域指定手段は、前記複数の予測モードに対し共通に１つの参照画像領域を指定しかつ前記評価手段に対し１つの予測モードに従う演算のみをイ

ネーブルする手段を含む、請求項９記載の動きベクトル検出装置。

【請求項１３】　参照フレーム画像と現フレーム画像とのブロックマッチング処理により、動き補償付予測符号化処理に用いられる動きベクトルを求めるための動きベクトル検出装置であって、前記フレームは偶数フィールドと奇数フィールドとを含み、

動きベクトル検出対象となる現フレーム画像ブロックの各画素に対応して配置される複数のプロセッサを含み、前記現フレーム画像ブロックの画素データと前記参照フレーム画像内の前記現フレーム画像ブロックに関連する領域内の参照画像ブロックの画素データとを入力し、両ブロックの対応の画素データに対し所定の演算を行なう演算手段と、

前記演算手段の出力を前記偶数フィールド内の画素に対する演算結果と前記奇数フィールド内の画素に対する演算結果と前記フレーム内の画素に対する演算結果とに分類し、各分類ごとに前記演算手段の出力を加算して総和を求める総和手段と、

前記総和手段の出力に従って、前記現画像ブロックの奇数フィールド画素ブロックに対する動きベクトル、偶数フィールド画素ブロックに対する動きベクトルおよびフレーム画素ブロックに対する動きベクトルを並列態様で決定する動きベクトル決定手段を備える、動きベクトル検出装置。

【請求項１４】　参照フィールド画像と現フィールド画像とのブロックマッチング処理により、動き補償付予測符号化処理に用いられる動きベクトルを求めるための動きベクトル検出装置であって、

前記現フィールド画像内の動きベクトル検出対象となる現フィールド画像ブロックの各画素に対応して配置される複数のプロセッサを含み、前記現フィールド画像ブロックの画素データと前記参照フィールド画像内の現フィールド画像ブロックに関連する領域内の参照フィールド画像ブロックの画素データとを入力し、両入力ブロックの対応の画素データに対し所定の演算を行なう演算手段と、

前記演算手段の出力を前記現フィールド画像ブロックのフィールド内垂直方向に沿って２分割して上半分ブロックと下半分ブロックとに分割し、前記演算手段の出力を前記上半分ブロックの画素に対する演算結果と、前記下半分のブロックの画素に対する演算結果と前記現画像ブロック内の画素すべてに対する演算結果とに分類し、各分類ごとに前記演算手段の出力を加算して総和を求める総和手段と、

前記総和手段の出力に従って、前記上半分ブロックに対する動きベクトル、前記下半分ブロックに対する動きベクトル、および前記現画像ブロックに対する動きベクトルを並列態様で決定する動きベクトル決定手段を備える、動きベクトル検出装置。

【請求項１５】　参照画面画像と現画面画像とのブロックマッチング処理により、動き補償付予測符号化に用いられる動きベクトルを求めるための動きベクトル検出装置であって、

前記現画面画像の動きベクトルのの検出対象となる現画像ブロックの各画素に対応して配置される複数のプロセッサを含み、前記現画像ブロックの画素データと前記参照画面画像内の前記現画像ブロックに関連する領域内の参照画像ブロックの画素データとを入力し、該入力した両ブロックの対応の画素のデータに対し所定の演算を行なう演算手段を備え、前記複数のプロセッサは、前記画面がフレームで構成されるとき、前記現画像ブロックの画面垂直方向の上半分のブロックの画素に対応し、かつ奇数フィールドの画素に対応して配置される第１のプロセッサ群と、前記上半分のブロックの画素に対応しかつ偶数フィールドの画素に対応して配置される第２のプロセッサ群と、前記現画像ブロックの画面垂直方向の下半分のブロックの画素に対応しかつ奇数フィールドの画素に対応して配置される第３のプロセッサ群と、前記下半分のブロックの画素に対応しかつ偶数フィールドの画素に対応して配置される第４のプロセッサ群とを含み、

前記第１ないし第４のプロセッサ群の出力を並列に受け、前記第１および第２のプロセッサ群の出力を組合わせかつ前記第３および第４のプロセッサ群の出力を組合わせて２組の出力信号を生成する第１の組合わせ手段と、前記第１および第３のプロセッサ群の出力を組合わせかつ前記第２および第４のプロセッサ群の出力を組合わせて２組の出力信号を生成する第２の組合わせ手段とを含み、モード指定信号に応答して前記第１および第２の組合わせ手段の一方を活性化し、活性化された組合わせ手段からの２組の出力を伝達する分類手段と、

前記分類手段の２組の出力をそれぞれ各組ごとに加算総和し、各組に対する評価値および前記現画像ブロック全体に対する評価値を並列態様で生成する評価値生成手段と、

前記評価値生成手段の出力に従って、前記現画像ブロックの動きベクトルを各組および現画像ブロックに対して並列態様で決定する動きベクトル決定手段を備える、動きベクトル検出装置。

【請求項１６】　現画面画像と参照画面画像とのブロックマッチング処理により、動き補償付予測符号化に用いられる動きベクトルを検出するための動きベクトル検出装置であって、

画像データの入力順序と出力順序とを異ならせる手段を含み、入力参照画面画像データを一時的に格納するバッファ手段と、

前記バッファ手段から読出された画像データと前記現画面画像データとから動きベクトル検出対象となる現画像ブロックの動きベクトルを算出する手段とを備える、動きベクトル検出装置。

【請求項１７】　現画面画像と参照画面画像とのブロックマッチング処理により、動き補償付予測符号化に用いられる動きベクトルを検出するための動きベクトル検出装置であって、
動きベクトル検出対象となる現画像ブロックの画素に対応して配置され、各々が対応の現画像ブロック画素データと前記現画像ブロックに対応する前記参照画面画像内の参照画像ブロックの対応の画素データとを格納する格納手段と、前記格納手段に格納された画素データに所定の演算処理を行なう演算手段とを含み、前記格納手段に格納されたデータを一方方向のみに沿って伝達する複数のプロセッサと、
前記複数のプロセッサの所定数ごとに配置され、前段のプロセッサまたは入力部から伝達された参照画素データを一時的に格納し、順次次段のプロセッサへ伝達するとともに、その有効格納画素データ数が変更可能なバッファ手段と、
前記複数のプロセッサ手段の演算結果出力に従って前記現画像ブロックの動きベクトルを決定する動きベクトル決定手段とを備える、動きベクトル検出装置。
【請求項１８】　現画面画像と参照画面画像とのブロックマッチング処理により、動き補償付予測符号化に用いられる動きベクトルを検出するための動きベクトル検出装置であって、
動きベクトル検出対象となる現画像ブロックと前記現画像ブロックに関連する前記参照画面内の探索領域内のすべての評価点のうち代表評価点に対応する参照画像ブロックとのブロックマッチング処理を複数の予め定められた予測モードに従って実行し、前記複数の予測モードに対して最も良い相関度を示す最適代表評価点を求める第１の算出手段と、
前記複数の予測モードそれぞれに対応して設けられ、求められた最適代表評価点に従って、該対応の最適代表評価点を中心とする所定の大きさの領域内の全評価点に対して前記現画像ブロックと参照画像ブロックとのブロックマッチング処理を前記複数の予測モードに従って行ない、各々が前記複数の予測モードそれぞれに対して最もよい相関度を示す最適評価値およびベクトルを求める複数の第２の算出手段と、
前記複数の第２の算出手段の出力に従って、前記複数の予測モードのうちの最適予測モードを求め、かつ該求められた最適予測モードに従って決定された最適ベクトルを前記現画像ブロックに対する動きベクトルとして決定する動きベクトル決定手段とを備える、動きベクトル検出装置。
【請求項１９】　現画面画像と参照画面画像とのブロックマッチング処理により、動き補償付予測符号化に用いられる動きベクトルを検出するための動きベクトル検出装置であって、
動きベクトル検出対象となる現画像ブロックと前記現画像ブロックに関連する参照画面内の探索領域のすべての評価点のうち代表評価点により表現される参照画像ブロックとのブロックマッチング処理を複数の予測モードに従って実行し、前記複数の予測モード各々に対して最もよい相関度を示す最適代表評価点を求める第１の算出手段と、
前記複数の予測モード各々に対応して設けられ、前記第１の算出手段からの対応の予測モードの最適代表評価点に従って、該最適代表評価点を中心とする所定の大きさの領域内の全評価点に対して前記現画像ブロックと参照画像ブロックとのブロックマッチング処理を前記複数の予測モードに従って実行し、最もよい相関度を示す評価点を表わす最適ベクトルおよび最適評価値を算出する複数の第２の算出手段と、
前記複数の第２の算出手段の出力に従って、前記複数の予測モードのうちの最適予測モードを決定し、かつ該決定された最適予測モードに従って決定された最適ベクトルを前記現画像ブロックに対する動きベクトルとして決定する動きベクトル決定手段とを備える、動きベクトル検出装置。
【請求項２０】　現画面画像と参照画面画像とのブロックマッチング処理により、動き補償付予測符号化に用いられる動きベクトルを検出するための動きベクトル検出装置であって、
動きベクトル検出対象となる現画像ブロックと前記現画像ブロックに関連する参照画面内の探索領域のすべての評価点のうち代表評価点により表現される参照画像ブロックとのブロックマッチング処理を複数の予測モードに従って実行し、前記複数の予測モード各々に対して最もよい相関度を示す予測モードに対応する最適代表評価点を求める第１の算出手段と、
前記第１の算出手段からの最適代表評価点に従って、前記最適代表評価点を中心とする所定の大きさの領域内の全評価点に対し前記現画像ブロックと参照画像ブロックとのブロックマッチング処理を前記複数の予測モード各々に従って実行し、各前記予測モードに対して最適ベクトルおよび最適評価値を算出する複数の第２の算出手段と、
前記第２の算出手段の出力に従って複数の予測モードのうちの最適予測モードを決定し、それに対応する最適ベクトルを前記現画像ブロックに対する動きベクトルとして決定する動きベクトル決定手段とを備える、動きベクトル検出装置。
【請求項２１】　現画面画像と参照画面画像とのブロックマッチング処理により、動き補償付予測符号化に用いられる動きベクトルを検出するための動きベクトル検出装置であって、
動きベクトル検出対象となる現画像ブロックと前記現画像ブロックに関連する参照画面内の探索領域のすべての評価点のうち代表評価点に対応する参照画像ブロック各

々とのブロックマッチング処理を複数の予測モードに従って行ない、前記複数の予測モードのうち最もよい相関度を示す最適予測モードおよびそれに対応する最適代表評価点を求める第１の算出手段と、
前記第１の算出手段からの最適予測モードを示す情報および前記最適代表評価点情報に従って、前記最適評価点を含む前記探索領域内の所定の大きさの領域内の全評価点に対し前記最適予測モードに従って前記現画像ブロックと参照画像ブロックとのブロックマッチング処理を実行し、各参照画像ブロックに対する評価関数値を算出する第２の算出手段と、
前記第２の算出手段の出力に従って、前記現画像ブロックに対する動きベクトルを決定する動きベクトル決定手段を備える、動きベクトル検出装置。
【請求項２２】　現画面画像と参照画面画像とのブロックマッチング処理により、動き予測符号化処理に用いられる動きベクトルを求めるための動きベクトル検出装置であって、
前記現画面画像のＱ行Ｐ列の画素からなる現画面ブロックの各画素に対応して配置されるプロセッサと、前記Ｐ列の各々に対応して配置され、各々がＲ個の参照画面画素データを格納するデータ格納手段とを有するプロセッサアレイを備え、前記プロセッサとデータ格納手段とが１次元的に一方向に沿って参照画面画像画素データを転送するように相互接続され、かつ前記プロセッサの各々は、対応の現画面画像画素データを格納する第１のレジスタ手段と、与えられた参照画面画像画素データを格納する第２のレジスタ手段と、前記第１および第２のレジスタ手段の格納する画素データに所定の演算処理を施して評価関数値成分を出力する演算手段とを含み、
前記プロセッサアレイへ１サイクル当たり１画素データの速度で、前記参照画面画像画素データを与えるデータ印加手段と、
前記プロセッサアレイから出力される評価関数値成分を加算して、１サイクル当たり１評価関数値を生成する評価値生成手段と、
前記評価値生成手段から所定数のサイクルにわたって与えられる評価値に従って前記現画面ブロックの動きベクトルを検出する動きベクトル決定手段と、
前記所定数のサイクルを、Ａを任意の自然数として、Ａ・（Ｐ・（Ｑ＋Ｒ））サイクルに設定する手段を備える、動きベクトル検出装置。
【請求項２３】　現画面画像と参照画面画像とのブロックマッチング処理により、動き予測符号化処理に用いられる動きベクトルを求めるための動きベクトル検出装置であって、
各々が、Ｑ行Ｐ列の画素からなる現画面画像ブロックの各画素に対応して配置されるプロセッサと、前記Ｐ列の各々に対応して配置され、かつ各々がＲ個の参照画面画素データを格納するデータ格納手段とを有するＡ個のプロセッサアレイを備え、前記Ａは自然数であり、かつ前記Ａ個のプロセッサアレイの各々は、プロセッサとデータ格納手段とが１次元的に一方方向に沿って参照画面画像画素データを転送するように相互接続され、かつ前記プロセッサの各々は、対応の現画面画像画素データを格納する第１のレジスタ手段と、与えられた参照画面画像画素データを格納する第２のレジスタ手段と、前記第１および第２のレジスタ手段に格納された画素データに所定の演算処理を施して評価関数値成分を出力する演算手段とを含み、
前記Ａ個のプロセッサアレイへ同じ参照画面画像画素データを１サイクル当たり１画素データの割合で与える手段と、
前記Ａ個のプロセッサアレイに対応して設けられ、対応のプロセッサアレイから出力される評価関数値成分から現画面ブロックと参照画面ブロックとの類似度を示す評価関数値を生成するＡ個の評価値生成手段と、
前記Ａ個の評価値生成手段各々に対応して設けられ、対応の評価値生成手段からＡ・Ｐ（Ｑ＋Ｒ）サイクルにわたって与えられる評価値に従って各プロセッサアレイそれぞれについての動きベクトルを決定するＡ個の動きベクトル決定手段と、
前記Ａ個のプロセッサアレイの現画面画像ブロック画素データ全体を前記Ａ・Ｐ（Ｑ＋Ｒ）サイクルを単位サイクルとして順次更新する現画面ブロック更新手段を備える、動きベクトル検出装置。
【請求項２４】　動き予測符号化処理に用いられる動きベクトルを、現画面画像と参照画面画像とのブロックマッチング処理により求めるための動きベクトル検出装置であって、
前記現画面画像のＱ行Ｐ列の画素に対応して配置されるプロセッサと、前記Ｐ列の各列に対応して配置され、各々がＲ個の参照画面画像画素データを格納するデータ格納手段とを備えるプロセッサアレイを含み、前記プロセッサの各々は、対応の現画面画素データを格納する第１のレジスタ手段と、与えられた参照画面画像画素データを格納する第２のレジスタ手段と、前記第１および第２のレジスタ手段の格納する画素データに所定の演算処理を施して評価関数値成分を出力する演算手段とを含み、
前記プロセッサアレイは、各々が（Ｐ／ｂ）列（Ｑ／ｃ）行の画素に対応して配置される複数のサブブロックに分割され、前記サブブロックは行列状に配置され、
前記プロセッサの列それぞれに対応して設けられ、各々がＲ個の参照画面画像画素データを格納する複数のデータ格納手段を備え、前記データ格納手段は同じ列に配置されたサブブロックに共有されかつサブブロック列それぞれにおいて、データ格納手段は１つのサブブロックのプロセッサと１次元的に参照画面画像画素データを一方方向に沿って転送するように相互接続されかつ残りのサブブロックに対しては同じ列に配置されたプロセッサに

対してのみ接続され、かつ前記残りのサブブロックにおけるプロセッサは、同一列においてのみ一方方向に沿ってのみ与えられた参照画面画像画素データを転送するように相互接続され、
前記複数のサブブロックそれぞれのプロセッサから（Ｐ／ｂ）・｛（Ｑ／ｃ）＋Ｒ｝・ｂ・ｃサイクルにわたって各サイクルに与えられる評価関数値成分に従って、各サブブロックについて関連の現画面画像ブロックの動きベクトルを決定する手段と、
前記複数のサブブロックの各々へ互いに異なる現画面画像画素ブロックの画素データを格納するとともに、前記複数のサブブロックに格納される現画面画像画素データを（Ｐ／ｂ）・｛（Ｑ／ｃ）＋Ｒ｝サイクルごとに順次更新する手段を含む、動きベクトル検出装置。
【請求項２５】　現画面画像と参照画面画像とのブロックマッチング処理により、動き予測符号化に用いられる動きベクトルを求めるための動きベクトル検出装置であって、
現画面画像のＱ行Ｐ列の画素からなる現画面ブロックの各画素に対応して行列状に配置されるプロセッサと、前記画素列の各々に対応して配置され、各々がＲ個の参照画面画素データを格納するデータ格納手段とを有するプロセッサアレイを備え、前記複数のプロセッサとデータ格納手段とは１次元的に一方方向に沿って参照画面画像画素データを転送するように相互接続され、かつ前記複数のプロセッサの各々は、第１の現画面ブロックの画素データを格納する第１のレジスタ手段と、第２の現画面ブロックの対応の画素データを格納する第２のレジスタ手段と、与えられた参照画面画像画素データを格納する第３のレジスタ手段と、前記第１および第２のレジスタ手段の一方の格納する画素データを選択する選択手段と、前記選択手段の選択した画素データと前記第３のレジスタ手段に格納された画素データとに所定の演算処理を施して評価関数値成分を生成する演算手段とを含み、
前記プロセッサアレイから出力される評価関数値成分を総和して、１サイクル当たり１評価関数値を生成する評価値生成手段と、
前記評価値生成手段の出力する評価関数値に従って現画面ブロックの動きベクトルを決定する動きベクトル決定手段と、
前記第１および第２の現画面ブロックの一方についての動きベクトル検出サイクルの間に他方の現画面ブロックに対応するレジスタ手段の内容を更新する更新手段と、前記選択手段を選択するレジスタを前記検出サイクルごとに切換える切換制御手段とを備える、動きベクトル検出装置。
【請求項２６】　現画面画像と参照画面画像とのブロックマッチング処理により、動き補償付予測符号化に用いられる動きベクトルを求めるための動きベクトル検出装置であって、
前記現画面内の動きベクトル検出対象となる所定のサイズの現画像ブロックと前記現画像ブロックに関連する前記参照画面内の領域内の複数の参照画像ブロック各々との類似度を示す評価関数値を生成する評価値生成手段と、
前記評価値生成手段から与えられた評価関数値に従って前記現画像ブロックの動きベクトルを決定する決定手段とを備え、
前記決定手段は、参照画像ブロックの前記元画像ブロックについての水平および垂直方向の位置をベクトル（Ｈ，Ｖ）で表わすとき、複数の参照画像ブロックの評価関数値が同じとき、
（ｉ）　｜Ｈ｜＋｜Ｖ｜の小さいほうの参照画像ブロックのベクトルを動きベクトル候補として選択する第１の選択手段と、
（ii）　｜Ｈ｜＋（－Ｖ＋α）の小さい参照画像ブロックのベクトルを前記動きベクトルの候補として選択する第２の選択手段と、
（iii）　｜Ｈ｜＋（Ｖ＋α）の小さいほうの参照画像ブロックのベクトルを前記動きベクトルの候補として選択する第３の選択手段と、
モード指定信号に応答して、前記第１ないし第３の選択手段のいずれか１つを活性状態とする手段とを備え、前記αは、動きベクトルのＹ方向の探索範囲に応じて決定される定数である、動きベクトル検出装置。
【請求項２７】　現画面画像と参照画面画像とのブロックマッチング処理により、動き補償付予測符号化に用いられる動きベクトルを求めるための動きベクトル検出装置であって、
前記現画面画像の所定のサイズの現画面ブロックと前記現画面ブロックに関連する前記参照画面内領域の複数の参照画面ブロックそれぞれとを複数の予測モードそれぞれに従ってブロックマッチング処理を施して整数精度での動きベクトルを前記複数の予測モード各々について決定する整数精度ベクトル決定手段と、
前記現画面ブロック画素データと前記整数精度ベクトル決定手段により決定された複数の予測モード各々の動きベクトルで示される複数の参照画面ブロック画素データをそれぞれ前記整数ベクトル決定手段から受けて格納する画素データ格納手段と、
前記画素データ格納手段に格納された現画面ブロック画素データと複数の参照画面ブロック各々の画素データとを受け、ブロックマッチング処理を分数精度で行なって、前記複数の予測モード各々の分数精度での動きベクトルを決定する分数精度ベクトル決定手段とを備える、動きベクトル検出装置。
【請求項２８】　現画面画像と参照画面画像とのブロックマッチング処理により、動き補償付予測符号化に用いられる動きベクトルを検出するための動きベクトル検出装置であって、

前記現画面の所定のサイズの現画面ブロックと所定のサイズの探索領域内の前記参照画面画像に関連する複数の参照ブロック各々との類似度を示す評価関数値を求める評価値算出手段と、
前記探索領域の有効領域を前記現画面ブロックについて設定する設定手段と、
前記設定手段により設定された有効領域に含まれる参照ブロックに対する、前記評価値算出手段の評価値に従って前記現画面ブロックの動きベクトルを決定する手段とを備える、動きベクトル検出装置。
【発明の詳細な説明】
【０００１】
【産業上の利用分野】この発明は、予測符号化処理において動画像の動き補償のために用いられる動きベクトルを検出するための装置に関する。
【０００２】
【従来の技術】膨大なデータ量を有する画像信号の伝送または蓄積のためには、データ量を削減するデータ圧縮技術が必要不可欠となる。画像データは、近接画素間の相関関係および人間の知覚特性などに起因するかなりの冗長度を備える。このデータ冗長度を抑圧して伝送データ量を削減するデータ圧縮技術は高能率符号化と呼ばれる。このような高能率符号化方式の１つに、フレーム間予測符号化方式がある。このフレーム間予測符号化方式においては、以下の処理が実行される。
【０００３】現在符号化しようとする現画面（フレームまたはフィールド）の各画素データと参照する前画面の同じ位置にある各画素データとの差分である予測誤差を算出する。算出された予測誤差を以後の符号化に用いる。この方法では、動きの少ない画像に関しては画面間の相関が大きいため、高能率で符号化を行なうことができる。しかしながら、動きの大きな画像については、画面間の相関が小さいため誤差が大きくなり、逆に伝送データ量が増加するという欠点が生じる。
【０００４】上述の問題点を解決する方法として、動き補償付フレーム（フィールド）間予測符号化方式がある。この方法では、以下の処理が実行される。まず、予測誤差を算出する前に、予め現画面（フレームまたはフィールド）と前画面の画素データを用いて動きベクトルを算出する。算出された動きベクトルに従って前画面の予測画像を移動させる。前画面の動きベクトルだけずれた位置の画像データを参照画素とし、参照画素を予測値として用いる。次いで、この移動後の前画面と現画面との各画素の予測誤差を算出し、予測誤差と動きベクトルとを伝送する。
【０００５】図１５１は、従来の動き補償付予測符号化方式に従って画像データを符号化する符号器の全体構成を概略的に示すブロック図である。図１５１において符号器は、入力された画像信号に対し所定の前処理を実行する前処理回路９１０と、この前処理回路９１０により前処理された信号に対し冗長度の除去および入力信号の量子化を実行するソース符号化回路９１２と、ソース符号化回路９１２からの信号に対し所定のフォーマットに従った符号化および予め定められたデータ構造の符号列に多重化するビデオマルチプレクス符号化回路９１４を含む。
【０００６】前処理回路９１０は、時間および空間フィルタを用いて入力画像信号を共通の中間フォーマット（ＣＩＦ）に変換し、かつノイズ除去のためのフィルタ処理を実行する。
【０００７】ソース符号化回路９１２は、与えられた信号に対するたとえば離散コサイン変換（ＤＣＴ）などの直交変換処理を行なうとともに、入力信号に対する動き補償を行ない、かつ直交変換された画像データを量子化する。
【０００８】ビデオマルチプレクス符号化回路９１４は、与えられた画像信号に対し２次元可変長の符号化を行なうとともに、データ処理単位であるブロックの各種属性（動きベクトルなど）も可変長符号化した後に、予め定められたデータ構造の符号列に多重化する。
【０００９】符号器はさらに、ビデオマルチプレクス符号化回路９１４からの画像データをバッファ処理する伝送バッファ９１６と、伝送バッファ９１６からの画像データを伝送チャネルに適合させるための伝送符号化回路９１８を含む。
【００１０】伝送バッファ９１６は、情報発生速度を一定速度に平滑化する。伝送符号化回路９１８は誤り訂正ビットの付加および音声信号データの付加などを実行する。
【００１１】図１５２は、図１５１に示すソース符号化回路の具体的構成の一例を示す図である。図１５２においてソース符号化回路９１２は、前処理回路からの与えられる入力画像信号に対し動きベクトルを検出しかつ動きベクトルに従って動き補償された参照画像を生成する動き補償予測器９２０と、動き補償予測器９２０からの参照画像画素データに対しフィルタ処理を行なうループフィルタ９２２と、ループフィルタ９２２の出力と入力画像信号との差分を求める減算器９２４と、減算器９２４の出力を直交変換する直交変換器９２６と、直交変換器９２６により直交変換されたデータを量子化する量子化器９２８を含む。
【００１２】動き補償予測器９２０の構成は後に詳細に説明するが、１フレーム前の画素データを格納するフレームメモリを含み、入力画像信号データとこのフレームメモリ内の画素データとに従って動きベクトルの検出および動き補償された参照画像画素データの生成を行なう。ループフィルタ９２２は画質改善のために設けられる。
【００１３】直交変換器９２６は、減算器９２４からのデータに対し、所定のサイズのブロック（通常８×８画

素）を１つの単位としてＤＣＴ変換などの直交変換を行なう。量子化器９２８は、この直交変換された画素データを量子化する。

【００１４】動き補償予測器９２０および減算器９２４により動き補償付フレーム間予測が実行され、動画像における時間的な冗長性が除去される。また、直交変換器９２６による直交変換により動画像信号における空間的な冗長性が除去される。

【００１５】ソース符号化回路９１２はさらに、量子化器９２８で量子化されたデータを量子化前の信号状態に変換するための逆量子化器９３０と、この逆量子化器９３０の出力に対し逆直交変換を行なう逆直交変換器９３２と、ループフィルタ９２２の出力と逆直交変換器９３２の出力とを加算する加算器９３４を含む。逆量子化器９３０および逆直交変換器９３２により次のフレームに対するフレーム間予測に用いる画像が生成される。この生成された画像データは動き補償予測器９２０に含まれるフレームメモリに書込まれる。加算器９３４により画像信号（フレーム間差分データ）がループフィルタ９２２の出力に加算されるため、現フレームの画像データが再生される。通常、この逆量子化処理、逆直交変換処理および加算処理は一般に局部復号過程と呼ばれる。次に、動きベクトルの算出について具体的に説明する。動きベクトルの算出には一般にブロックマッチング法が用いられる。

【００１６】今、図１５３（Ａ）に示すように、第（ｍ－１）フレームにおける画像Ａが第ｍフレームのおいてはＡ′に移動した状態を考える。ブロックマッチング法においては、画面（この場合１フレーム）をＰ×Ｑ画素のブロックに分割する（一般にはＰ＝Ｑ）。現フレームにおいて着目するブロックに最も近似するブロックを前フレームから捜し出す。この着目ブロックから最も近似する前フレームにおけるブロックへのずれを動きベクトルとを称す。以下に、より詳細に説明する。

【００１７】図１５３（Ｂ）に示すように、今、第ｍフレームを符号化対象フレームとする。フレームはＮ×Ｎ画素のブロックに分割される（Ｐ＝Ｑ＝Ｎ）。今第ｍフレームにおけるＮ×Ｎ画素のブロックにおける１番左上の画素位置（Ｎｋ，Ｎ１）における画素データの値をＸｍ（Ｎｋ，Ｎ１）とする。画素位置をベクトル（ｉ，ｊ）だけずらした前フレームにおけるブロックと現フレームにおけるブロックの対応の画素のデータの差分の絶対値和を求める。次に、このずれベクトル（ｉ，ｊ）をさまざまな値に代え、それぞれの差分絶対値和を求める。この差分絶対値和は一般に評価関数値と呼ばれる。この最小の差分絶対値和を与える位置（ｉ，ｊ）を動きベクトルと定義する。

【００１８】動きベクトルは１ブロック画素当たり１個伝送する必要がある。ブロックサイズを小さくすると伝送情報が増加し、効果的なデータ圧縮ができなくなる。一方、ブロックサイズを大きくすると効果的な動き検出が困難となる。そこで、ブロックサイズは１６×１６画素、動きベクトル探索範囲（ｉ，ｊの最大変化幅）は－１５～＋１５とするのが一般的である。以下に具体的にブロックマッチング法による動きベクトル算出について説明する。

【００１９】図１５４は、ブロックマッチング法による動きベクトルの算出の具体的方法を示す図である。今、図１５４に示すように、３５２ドット（画素）×２８８ラインからなる画像９５０を考える。画像９５０を、１６×１６の画素群を１ブロックとして複数のブロックに分割する。このブロック単位で動きベクトルの検出が実行される。動きベクトル検出処理の対象となるブロック（以下、テンプレートブロックと称す）９５２と同じ位置にある前フレームにおけるブロック９５４を基準として、画面上水平方向および垂直方向に±１６画素大きなブロック９５６を探索ブロック（以下、サーチエリアと称す）とする。テンプレートブロック９５２に対する動きベクトル探索はこのサーチエリア９５６内において実行される。ブロックマッチング法に従った動きベクトルの探索方法は以下の処理ステップを備える。

【００２０】動きベクトル候補に対応する変位を有するブロック（図１５４においてベクトル（ｉ，ｊ）で示す）を求める。この求められたブロックの各画素とテンプレートブロックの対応位置にある画素の差分絶対値和（または差分二乗和）のような評価関数値を求める。

【００２１】上述の動作をベクトル（ｉ，ｊ）として（－１６，－１６）～（＋１６，＋１６）のすべての変位に対して実行する。すべての予測画像ブロック（サーチエリア９５６内のすべての画像ブロック）に対して評価関数値（評価値）を求めた後、この評価関数値が最小となる予測画像ブロックを検出する。テンプレートブロックと同じ位置（以下、真裏と称す）のブロック（図１５４においてベクトル（０，０）で示すブロック９５４）から評価関数値が最小となる予測画像ブロックに向かうベクトルをこのテンプレートブロック９５２に対する動きベクトルと決定する。

【００２２】図１５５は、従来のハードウエアで実現される動きベクトル検出装置の全体の構成を示す図であり、たとえば１９８９　ＩＥＥＥ、ＩＣＡＳＳＰ’８９のプロシーディング”、第２４５３頁ないし第２４５６頁においてＡ・アルティエリ等により示されている。図１５５において、動きベクトル検出装置は、サーチエリアの画素データをサーチエリアの１列分入力するためのサーチエリア入力レジスタ９６２と、テンプレートブロックの評価点（動きベクトルの候補）と同一サイズの行および列のマトリックス状に配置された複数のプロセッサを含むプロセッサアレイ９６６と、プロセッサアレイに対しサーチエリアにおける同一列のデータを格納するサーチエリアレジスタ９６４ａおよび９６４ｂと、プロ

セッサアレイ９６６の演算結果に従って動きベクトルを検出する動きベクトル検出部９６８を含む。

【００２３】プロセッサアレイ９６６においては、評価点すなわち変位ベクトル（ずれベクトル（ｉ，ｊ））各々に対応してプロセッサが配置される。すなわち第ｉ行、第ｊ列に配置されたプロセッサＰｉｊは、変位ベクトルＤ（ｉ，ｊ）を計算する。

【００２４】図１５６は、図１５５に示すプロセッサアレイに含まれるプロセッサの構成を示す図である。図１５６においてプロセッサ９７０は、アレイ９６６内の水平および垂直方向の３方向のプロセッサから伝達されるサーチエリア画素データを受け、選択信号ＳＥＬに応答してその１つの入力を通過させる３入力レジスタ９７２と、３入力レジスタ９７２からのサーチエリア画素データＹと外部から与えられるテンプレートブロック画素データＸとに基づいて歪み（差分絶対値和）を計算する歪み計算部９７４と、歪み計算部９７４からの歪みＤとアレイ９６６内の水平方向に隣接するプロセッサからの歪みとを受け、選択信号Ｔｏに従って一方を選択して通過させる２入力レジスタ９７６を含む。

【００２５】図１５６に示すプロセッサ９７０は、図１５６に示すプロセッサアレイ９６６において、サーチエリアにおける評価点、すなわち動きベクトルの候補となるすべての変位ベクトルに対応して２次元状に配置される。プロセッサアレイ９６６（図１５５参照）の各プロセッサへは共通に同じテンプレート画素データＸが与えられる。このとき、またプロセッサ９７０へは、サーチエリアブロック内の対応の画像データが与えられる。たとえば、テンプレートブロック画素データＸが、Ｘ（ｍ，ｎ）の場合、プロセッサＰｉｊに対してはサーチエリアブロック画素データＹ（ｉ＋ｍ，ｊ＋ｎ）が与えられる。サーチウィンドウデータは、図１５５に示すサーチエリアサイドレジスタ９６４ａおよび９６４ｂならびにプロセッサアレイ９６６内の各プロセッサ９７０を介して転送される。外部から与えられるテンプレートブロック画素データＸ（ｍ，ｎ）に対し、正確にサーチエリアブロック画素データＹ（ｉ＋ｍ，ｊ＋ｎ）を各プロセッサに与えるために、テンプレートブロックおよびサーチエリアブロックはある規則性を持ってスキャンされなければならない。

【００２６】図１５７は、上述の動きベクトル検出装置におけるテンプレートブロックのデータのスキャン態様を示す図である。図１５７において、テンプレートブロック９９９を、図１５７の矢印で示すように、まず同一列に沿って上から下に向かってスキャンし、次いで１列隣の画素データを下から上に向かってスキャンすることによりテンプレートブロック画素データが生成されて、順次動きベクトル検出装置へ与えられる。このスキャン方法は「スネークスキャン」と呼ばれる。テンプレートブロック画素データの「スネークスキャン」に従って、プロセッサアレイ９６６へ与えられるサーチエリアブロック画素データも同様にスキャンされる。プロセッサ９７０は、アレイ内のその配置位置に応じてサーチエリア画素データを図１５６の上下方向または左方向へ転送する必要がある。この３方向転送を実現するために３入力レジスタ９７２が設けられる。２入力レジスタ９７６（図１５６参照）は、各変位ベクトルの評価関数値が計算された後、動きベクトル検出部９６８（図１５５参照）において最小の歪み（評価関数値）を与える変位ベクトルを求めるために、プロセッサ９７０で計算された歪みを動きベクトル検出部９６８へ伝達するために設けられる。動きベクトル検出部９６８は、プロセッサアレイ９６６内の各プロセッサからの歪みのうち最小の歪みを検出し、その最小の歪みを与えるプロセッサの位置を求め、そのプロセッサの位置を動きベクトルとして決定する。次にこの図１５５に示す動きベクトル検出装置の動作について簡単に説明する。

【００２７】プロセッサアレイ９６６において第ｉ行、第ｊ列に配置されたプロセッサＰｉｊは、

Ｄ（ｉ，ｊ）＝Σ｜Ｘ（ｍ，ｎ）－Ｙ（ｍ＋ｉ，ｎ＋ｊ）｜

で表される歪みＤ（ｉ，ｊ）を算出する。ここで総和Σはｍおよびｎに関して行なわれる。ｍおよびｎの変化範囲はテンプレートブロックのサイズにより決定される。

【００２８】今、図１５８に示すように、テンプレートブロック９８０としてｍ行ｎ列に配置された画素を考える。最初のサイクルにおいてはプロセッサアレイ９６６における各プロセッサには参照符号９８２で示すサーチエリアブロック画素データが格納される。外部からはテンプレートブロック９８０における第１行第１列の画素Ｘ（１，１）がプロセッサアレイ９６６内のすべてのプロセッサに共通に与えられる。アレイ９６６内の各プロセッサ９７０はそこに格納されているサーチエリアブロック（サーチウィンドウ）画素データＹと与えられたテンプレートブロック画素データＸとの差分絶対値を求めて累算する。

【００２９】次のサイクルにおいては、プロセッサアレイ９６６においてサーチエリアブロックが図１５８の１行下方向へシフトされる。プロセッサアレイ９６６内には参照符号９８３で示すサーチエリアブロック（サーチウィンドウ）画素データが格納される。この状態で、次いでテンプレートブロック９８２の次の画素データＸ（２，１）が与えられる。プロセッサＰｉｊにおいて確保されたサーチウィンドウ画素データは、Ｙ（ｍ＋ｉ，ｎ＋ｊ＋１）である。これらの画素データを用いて再び差分絶対値がとられかつ累算される。この動作がM回繰返される。

【００３０】M回上述の動作を繰返すと、外部からサーチエリアの１列のサーチエリア画素データが図１５８に示すサーチエリア入力レジスタ９６２を介して書込まれ

る。不要となったサーチエリアの１列の画像データは放出される。これにより、新しいサーチエリア画素データがサーチエリアサイドレジスタ９６４ａおよび９６４ｂおよびプロセッサアレイ９６６に格納される。この動作が各列ごとに繰返し実行される。

【００３１】すなわち、図１５９に示すように、最初はサーチウィンドウ（サーチエリア内のすべての行を含むブロック）を用いて差分絶対値和の計算が実行される。Ｍサイクル完了後次のサーチウィンドウ（サーチエリア内の１列右隣のブロック）の画素データを用いて再び同様の計算が実行される。以降、サーチウィンドウ９９４、…と同様の動作が実行される。最終的にサーチエリア９９６のすべての画素データに対する計算が実行されると、プロセッサＰｉｊにおいては、歪みＤ（ｉ，ｊ）が求められて保持される。このプロセッサＰｉｊにおいて求められた歪みＤ（ｉ，ｊ）が動きベクトル検出部９６８へ伝達され、そこで最小の歪みを与える変位ベクトルが動きベクトルとして検出される。

【００３２】

【発明が解決しようとする課題】動き補償付フレーム（またはフィールド）間予測符号化方式における予測画像検出方法には、いくつかの方式が提案されている。よりよい符号化効率を得るためには、複数の予測画像検出方式に従って動き検出処理を行なった後、最適な予測画像検出方式を選択し、この最適な予測画像検出方式に従って動きベクトルを検出する必要がある。画面の構成単位として、フィールドを単位とする場合とフレームを単位とする場合がある。１フレームは２フィールド（偶数フィールドと奇数フィールド）で構成される。このそれぞれの場合に対し、一例として以下のような予測画像検出方式がある。

【００３３】（Ａ）　フィールドを単位として画素データを符号化する場合：

（ａ）　フィールド画像をＰ×Ｑ画素を単位として複数のブロックに分割し、各ブロックごとに１つの動きベクトルを検出する（１つの予測画像を生成する）。

【００３４】（ｂ）　分割ブロックをさらに画面垂直方向に関して上下に２分割し、２分割ブロック各々に対して１つの動きベクトルを検出する。したがって、Ｐ×Ｑ画素のブロックに対しては上半分のブロックに対する動きベクトルと、下半分のブロックに対する動きベクトルを検出する（２つの予測画像を生成する）。

【００３５】（Ｂ）　フレームを単位として画素を符号化する場合：

（ａ）　フレーム画像をＰ×Ｑ画素のブロックを単位として複数のブロックに分割し、各ブロックに対し１つの動きベクトルを検出する（１つの予測画像を生成する）。

【００３６】（ｂ）Ｐ×Ｑ画素のブロックを同一フィールドに存在する画素、すなわち奇数フィールドに属する画素と偶数フィールドに属する画素との２つの画素グループに分割し、各画素グループに対して１つの動きベクトルを検出する。したがって、Ｐ×Ｑ画素のブロックに対して、偶数フィールドの画素グループに対する動きベクトルと奇数フィールドに属する画像グループに対する動きベクトルとが検出される（２つの予測画像が生成される）。

【００３７】上述の動きベクトル検出装置の構成の場合、プロセッサアレイ内の各プロセッサは動きベクトルの候補である変位ベクトル（評価点）に対応して配置される。また各プロセッサは同じテンプレートブロック画素データを受ける。したがってプロセッサはすべて同じ予測画素方式に従って歪み（評価関数値）を求めることができるだけであり、複数の予測画像検出方式に従って並列態様で複数の動きベクトルを検出することはできない。前述のプロセッサＰｉｊが算出する歪みＤ（ｉ，ｊ）の式から明らかなように、与えられるテンプレートブロック画素データＸ（ｍ，ｎ）が決定されれば、それに応じて与えられるサーチウィンドウ画素データＹ（ｍ＋ｉ，ｎ＋ｊ）も一意的に決定されるためである。したがって符号化効率向上のために複数の予測画像検出方式を実行するためには、少なくとも複数の予測画像検出方式それぞれに対応して動きベクトル検出装置を設け、これらの動きベクトル検出装置を並列に動作させる必要があり、装置規模が増大するとともに消費電力が増大するという問題が生じる。

【００３８】また上述の動きベクトル検出装置においては、プロセッサアレイ内のすべてのプロセッサに対し同じテンプレートブロック画素データが共通に与えられるため、テンプレートブロックの画素データを書込むための回路には大きな駆動力が必要とされ、このテンプレートブロック画素データ書込回路における電流消費が大きくなり、装置全体としての電力消費が大きくなるという問題も生じる。

【００３９】また、上述の動きベクトル検出装置の場合、各プロセッサは動きベクトル候補となる変位ベクトル（評価点）に対応して配置される。サーチエリアが垂直方向＋１６～－１６、水平方向－１６～＋１６であれば、動きベクトルの候補となる変位ベクトルの数は３３×３３＝１０８９となり、プロセッサの数が非常に大きくなり、装置占有面積が大きくなるという問題が生じる。

【００４０】さらに、演算の各サイクルにおいて、プロセッサアレイ内におけるデータ転送はプロセッサを介して行なわれる。このとき、プロセッサ間のデータ転送方向を決定するために３入力レジスタが用いられており、この３入力レジスタのいずれかを駆動する必要があり、データ転送時における消費電力が増大するという問題も生じる。

【００４１】また、動きベクトル探索方式には、全探索

方式と階層探索方式とがある。全探索方式は、サーチエリア内のすべての変位ベクトル（評価点）に対して評価関数値（差分絶対値和または差分二乗和等）を求め、すべの評価点の評価関数値から動きベクトルを求める方式である。階層探索方式は、その一例は特公平３－６８５９７号公報に示されているが、以下の探索ステップを含む。サーチエリア内の評価点のうち代表評価点を決定し（評価点の間引き）、代表評価点について評価関数値を求める。代表評価点の評価関数値から最小の評価関数値を与える代表評価点を求め、最適代表評価点とする。次いで、この最適代表評価点を中心とする所定の大きさの領域内の全ての評価点について評価関数値を求め、これらの評価関数値から最適評価点を求め、動きベクトルを決定する。

【００４２】階層探索方式の場合、サーチエリア内のすべての評価点について評価関数値を求める必要がなく、装置規模を小さくすることができ、また動きベクトル検出時間を短くすることができる。しかしながら、評価点を間引いて動きベクトルの検出を行なうため、全探索方式に比べて精度が劣るという欠点がある。逆に、全探索方式は精度は階層探索方式に比べて優れているという利点がある反面、装置規模が大きくなるという欠点がある。すなわち、単位時間当たりの演算回数が一定であると仮定し、同一量のハードウエアを用いて装置を構成する場合、階層探索方式を適用すれば広いサーチエリアを対象として予測画像を求めることができるものの、精度が劣るという欠点が生じ、また全探索方式を適用した場合、サーチエリアが狭くなるものの、動きベクトルの精度が優れているという利点が得られる。

【００４３】特に現在ＩＳＯ（国際標準化機構）で標準化が進められている動画像符号化方式においては現画面画像に対し時間的に前後する画面を用いて予測を行なうため、時間的に数画面（フレームまたはフィールド）離れた位置にある画面を用いて動き検出を行なうことが必要とされる。このような例としては、蓄積メディア符号化技術におけるＢピクチャーの動きベクトル検出がある。蓄積メディア符号化技術においては、画像データは蓄積メディアに格納されるため時間軸の制約がない。したがって、Ｂピクチャーと呼ばれる画像データに対しては、過去の画像を用いて予測画像を検出する順方向予測、未来の画像を用いて予測画像を検出する逆方向予測、および前後両画面を用いて予測を行なう内挿的（補間的）予測がある。このような前後画面を用いて予測を行なう場合、速い動きに追従するためには、サーチエリアを広くする必要がある（サーチエリアが狭ければ、画像がサーチエリアからはみ出し、正確な動きベクトル検出を行なうことができなくなる）。テンプレートブロックを中心として水平および垂直方向とも±６４～±１２８画素の範囲をサーチエリアとして探索が行なうのがよいとされている。このような広いサーチエリアを、装置規模を増大させることなく実現するためには、上述の階層探索方式を利用することが必要となる。

【００４４】しかしながら、従来の階層探索方式を前提とする動き予測方式については、単一予測モードに従った動き検出方式に従って予測画像を検出することができるだけであり、前述のような複数の予測モードを備える動き検出方式においてどのような階層探索方式を採用するかは未だ提案されていない。

【００４５】それゆえ、この発明の目的は、装置規模を増大させることなく複数の予測モードに対応して予測画像を検出することのできる動きベクトル検出装置を提供することである。

【００４６】この発明の他の目的は消費電流の増大および装置の複雑化をもたらすことなく複数の予測モードに従って高速に予測画像（動きベクトル）を検出することのできる動きベクトル検出装置を提供することである。

【００４７】この発明のさらに他の目的は、装置規模を増大させることなく高精度で階層探索方式に従って動きベクトルを検出することのできる動きベクトル検出装置を提供することである。

【００４８】この発明のさらに他の目的は、階層探索方式に従って、複数の予測モード（予測画像検出方式）各々に対する動きベクトルを高速かつ簡易な回路構成で検出することのできる動きベクトル検出装置を提供することである。

【００４９】

【課題を解決するための手段】請求項１に係る動きベクトル検出装置は、現画面画像内の動きベクトル検出対象となる所定のサイズの現画像ブロックと参照画面内の現画像ブロックに関連する領域内の複数の参照画像ブロック各々の類似度を示すとともに複数の予測モード各々に従って決定される評価関数値を各参照ブロックごとにかつ複数の予測モード並列に求めるための評価手段と、この評価手段からの各参照ブロックごとおよび各複数の予測モードそれぞれについての評価関数値を並列に受け、現画像ブロックに対する各予測モードについての動きベクトルを並列に決定する動きベクトル決定手段を備える。

【００５０】請求項２に係る動きベクトル検出装置は、請求項１に係る評価手段が、現画像ブロックに含まれる画素に対応して配置され、参照画像ブロック内の対応の画素データと現画像ブロック内の対応の画素データとを一時格納する格納手段を含み、かつ格納手段に格納された画素データに評価関数の一部をなす演算を行なう複数の第１の演算手段と、これら複数の第１の演算手段の出力を複数の予測モード各々に対応して分類し、各分類ごとに第１の演算手段の出力に評価関数の残りの演算を並列態様で行なう第２の演算手段を含む。

【００５１】請求項３に係る動きベクトル検出装置は、請求項１の複数の第１の演算手段が、各々が一方方向に

沿ってのみ格納手段に格納された画素データを転送するように配置されるとともに、現画像ブロックに対応する行および列のマトリックス状に配置される複数のプロセッサを備える。

【００５２】請求項４に係る動きベクトル検出装置は、現画面および参照画面が偶数フィールドおよび奇数フィールドを含むフレームで構成され、複数の予測モードは、奇数フィールドに含まれる画素を用いる動き予測モード（予測画像検出方式）と、偶数フィールドに含まれる画素を用いる動き予測モードと、フレームに含まれる画素を用いる動き予測モードとを含む。

【００５３】請求項５に係る動きベクトル検出装置では、上記現画面および参照画面の各々が１つのフィールドにより構成され、複数の予測モードが、現画像ブロックの垂直方向に２等分して得られる上半分のブロックに含まれる画素を用いる動き予測モードと、下半分のブロックに含まれる画素を用いる動き予測モードと、これら上半分および下半分のブロック両者に含まれる画素を用いる動き予測モードを含む。

【００５４】請求項６に係る動きベクトル検出装置は、請求項１ないし５のいずれかの動きベクトル検出装置における第２の演算手段が、現画像ブロックをフレーム画面構成時において垂直方向の上下２ブロックに分割し、かつ偶数フィールドの画素を含む偶数フィールドブロックおよび奇数フィールドに含まれる画素を含む奇数フィールドブロックの４ブロックに分割し、これら４分割ブロックそれぞれに対して対応の第１の演算手段の出力を並列に受け、予測モード指定信号に従って、４分割ブロックの構成を変更して３分類を実現する切換手段を含む。この切換手段が実現する３分類は、フレームに含まれる画素を用いるフレーム予測モード、奇数フィールドに含まれる画素を用いる奇数フィールド予測モード、および偶数フィールドに含まれる画素を用いる偶数フィールド予測、または画面がフィールドを単位として構成され、このフィールド内の画素を用いて動き予測を行なうフィールド予測モード、現画像ブロック内の上半分に含まれる画素を用いる上半分予測モードおよび現画像ブロックの下半分のブロックに含まれる画素を用いる下半分予測モードのいずれかである。

【００５５】請求項７に係る動きベクトル検出装置は、画面の形式を指定する形式指定信号に応答して、入力画像データの画面形式をこの形成指定信号が指定する形式に設定して評価手段へ与える入力設定手段をさらに含む。

【００５６】請求項８に係る動きベクトル検出装置は、請求項１ないし７のいずれかの動きベクトル検出装置における複数の第１の演算手段が、実質的に１次元アレイを構成するように配置され、入力した現画像データを一方方向に沿って伝達する複数のプロセッサと、これら複数のプロセッサの所定数ごとに配置され、入力部または前段プロセッサから与えられる参照画像ブロック画素データを格納するとともにその格納画素データを格納した順に出力するその有効格納画素数が変更可能なバッファ手段を備える。

【００５７】請求項９に係る動きベクトル検出装置は、請求項１ないし８のいずれかに記載の動きベクトル検出装置がさらに、現画像画素ブロックに関連する領域内の所定の条件を満足する代表評価点に対応する参照画像ブロックに対して複数の予測モードに従って評価関数値を複数の予測モードそれぞれに対して並列に求める第３の演算手段と、この第３の演算手段の出力に基づいて第１の演算手段が演算すべき参照画像ブロックの領域を各予測モードに対して指定する領域指定手段をさらに備える。第１の演算手段は、この領域指定手段により指定された領域に対応して配置され、領域指定された参照画像ブロック領域内において複数の予測モードに従って評価関数値を並列に求める。

【００５８】請求項１０に係る動きベクトル検出装置は、請求項９における動きベクトル検出装置において、領域指定手段が複数の予測モードそれぞれ別々に参照画像ブロック領域を指定する。

【００５９】請求項１１に係る動きベクトル検出装置は、請求項９の領域指定手段が、複数の予測モードに対し１つの参照画像領域を指定する。

【００６０】請求項１２に係る動きベクトル検出装置は、請求項９における領域指定手段が、複数の予測モードに対し共通に１つの参照画像ブロック領域を設定しかつ第１の演算手段に対し１つの予測モードのみに従って評価関数値を算出する演算動作のみをイネーブルする。

【００６１】請求項１３に係る動きベクトル検出装置は、動きベクトル検出対象となる現フレーム画像ブロックの各画素に対応して配置される複数のプロセッサを含み、現フレーム画像ブロックの画素データと、対応の参照画像ブロックの画素データとを入力し、入力した両ブロックの対応の画素データに対し所定の演算を行なう演算手段と、この演算手段の出力を偶数フィールド内の画素に対する演算結果と、奇数フィールド内の画素に対する演算結果とに分類し、各分類ごとに演算手段の出力を加算して総和を求める総和手段と、この総和手段の出力に従って、各分類に対する動きベクトルを並列態様で設定する動きベクトル決定手段を備える。これにより、奇数フィールド画素を用いた奇数フィールド予測モードに従う動きベクトル、偶数フィールド画素を用いる偶数フィールド予測モード、およびフレーム画素を用いるフレーム予測モードそれぞれに従う動きベクトルを並列に決定する。

【００６２】請求項１４に係る動きベクトル検出装置は、動きベクトル検出対象となる現フィールド画像ブロック内の各画素に対応して配置される複数のプロセッサを含み、この現フィールド画像ブロック内の画素データ

と対応の参照画像ブロック内の画素データとを入力し、入力した両ブロックの対応の画素データに対し所定の演算を行なう演算手段と、この演算手段の出力を、現画像ブロックの垂直方向上半分のブロックの画素に対する演算結果と、現画像ブロック垂直方向下半分のブロックの画素に対する演算結果と、現画像ブロック画素に対する演算結果とに分類し、各分類ごとに演算手段の出力を加算して総和を求める総和手段と、この総和手段の出力に従って各分類ごとに動きベクトルを並列態様で決定する動きベクトル決定手段を備える。これにより現フィールド画像ブロックの上半分のブロックの画素を用いる上半分ブロック予測モード、現画像ブロックの下半分のブロックの画素を用いる下半分ブロック予測モードおよび現画像ブロックの画素を用いるフィールド予測モードそれぞれに従う動きベクトルを並列に決定する。

【００６３】請求項１５に係る動きベクトル検出装置は、現画面画像の動きベクトル検出対象となる現画像ブロックの各画素に対応して配置される複数のプロセッサを含みかつこの現画像ブロックの各画素データと参照画面画像内とこの現画面画像ブロックに関連する領域内の参照画像ブロックとを入力し、入力した両ブロックの対応の画素データに対し所定の演算を行なう演算手段を含む。複数のプロセッサは、画面がフレームで構成されると仮定して、この現画像ブロックの画面上垂直方向の上半分ブロックの画素に対応しかつ奇数フィールドの画素に対応する第１のプロセッサ群と、この上半分のブロックの画素に対応しかつ偶数フィールドの画素に対応して配置される第２のプロセッサ群と、現画像ブロックの画面垂直方向の下半分の画素に対応しかつ奇数フィールドの画素に対応して配置される第３のプロセッサ群と、下半分の画素に対応しかつ偶数フィールドの画素に対応して配置される第４のプロセッサ群とを含む。

【００６４】請求項１５に係る動きベクトル検出装置はさらに、モード指定信号に応答して、この演算手段の出力を３つの類に分類する分類手段を含む。この分類手段は、第１および第２のプロセッサ群を組合わせかつ第３および第４のプロセッサ群の出力を組合わせて２つの類を生成する第１の組合わせ手段と、第１および第３のプロセッサ群を組合わせかつ第２および第４のプロセッサ群の出力を組合わせる第２の組合わせ手段と、モード指定信号に応答してこれら第１および第２の組合わせ手段の一方の出力を選択して出力する手段を含む。

【００６５】請求項１５に係る動きベクトル検出装置はさらに、この分類手段の出力を各類ごとに加算し、かつ総和して各分類に対する評価関数値および現画像ブロック全体に対する評価関数値を並列態様で生成する評価値生成手段と、この評価値生成手段の出力に従って、現画像ブロックの動きベクトルを各分類および現画像ブロックに対して並列態様で決定する動きベクトル決定手段を備える。

【００６６】請求項１６に係る動きベクトル検出装置は、参照画像データの入力順序と出力順序とを異ならせる手段を含み、入力参照画面画像データを一時的に格納するバッファ手段と、このバッファ手段から読出された参照画像データと現画面画像データとから動きベクトル検出対象となる現画像ブロックの動きベクトルを算出する手段を備える。

【００６７】請求項１７に係る動きベクトル検出装置は、動きベクトル検出対象となる現画像ブロックの画素に対応して配置され、各々が対応の現画像ブロックの画素データとこの現画像ブロックに関連する参照画面画像内の参照画像ブロックの対応の画素データとを格納する格納手段と、この格納手段に格納されたデータを一方方向のみに沿って伝達するとともに格納手段に格納された画素データに所定の演算処理を行なう演算手段とを各々が含む複数のプロセッサと、これら複数のプロセッサの所定数ごとに配置され、入力部または前段のプロセッサから伝達された参照画像ブロック画素データを一時的に格納しかつ順次格納画素データを格納順に次段のプロセッサへ伝達するその有効格納画素数が変更可能なバッファ手段と、複数のプロセッサ手段の演算結果出力から現画像ブロックの動きベクトルを決定する動きベクトル決定手段を備える。

【００６８】請求項１８に係る動きベクトル検出装置は、動きベクトル検出対象となる現画面ブロックとこの現画像ブロックに関連する参照画面内の探索領域のすべての評価点のうち代表評価点に対応する参照画像ブロックとのブロックマッチング処理を複数の予測モードに従って行ない、複数の予測モード各々に対して最もよい相関度を示す最適代表評価点を並列に求める第１の算出手段と、この第１の算出手段からの複数の予測モード各々に対する最適代表評価点に従って、各最適代表評価点が示す評価点を中心とする所定の大きさの参照画面内の領域内に含まれる評価点すべてに対し現画像ブロックと参照画像ブロックとのブロックマッチング処理を行ない、複数の予測モード各々に対する動きベクトルを算出する第２の算出手段と、この第２の算出手段の出力に従って、複数の予測モードのうちの最適予測モードの決定およびそれに対応する最適ベクトルを動きベクトルと決定する手段を含む。

【００６９】請求項１９に係る動きベクトル検出装置は、動きベクトル検出対象となる現画像ブロックとこの現画像ブロックに関連する参照画面内の所定の大きさを有する探索領域内のすべての評価点のうちの代表評価点に対応する参照画像ブロックとのブロックマッチング処理を複数の予測モードに従って並列態様で実行し、これら複数の予測モード各々に対する最もよい類似度を示す最適代表評価点を求める第１の算出手段と、複数の予測モード各々に対応して設けられ、第１の算出手段からの対応の予測モード各々に対する最適代表評価点に従っ

て、対応の最適代表評価点を含む探索領域内の所定の大きさの領域に含まれる評価点すべてに対し対応の予測モードに従って現画像ブロックと参照画像ブロックとのブロックマッチングを行ない、対応の予測モードに対する最適ベクトルおよび最適評価関数値を算出する第２の算出手段と、この第２の算出手段の出力に従って、最適ベクトルのうちから前記現画像ブロックに対する動きベクトルを決定する動きベクトル決定手段を含む。

【００７０】請求項２０に係る動きベクトル検出装置は、動きベクトル検出対象となる現画像ブロックとこの現画像ブロックに関連する参照画面内の所定の大きさの探索領域内のすべての評価点のうちの代表評価点に対応する参照画像ブロックとのブロックマッチング処理を複数の予測モードに従って並列態様で実行し、これら複数の予測モード各々に対し、最もよい相関度を示す最適代表評価点を求める第１の算出手段と、この第１の算出手段の出力に従って複数の予測モードそれぞれに対する最適代表評価点のうち最も高い類似度を示す第１の最適評価点を決定する第１の最適評価点決定手段と、この第１の最適評価点決定手段からの最適ベクトルに従ってこの最適ベクトルが示す評価点を含む探索領域内の所定のサイズの領域内のすべての評価点に対し複数の予測モードそれぞれに従って変位ベクトルを求め、これらの求めた変位ベクトルから最適変位ベクトルを各予測モードに対して求める第２の算出手段と、この第２の算出手段からの最適変位ベクトルを受け、最も高い相関度を示す最適変位ベクトルを現画像ブロックの動きベクトルと決定する動きベクトル決定手段を含む。

【００７１】請求項２１に係る動きベクトル検出装置は、動きベクトル検出対象となる現画像ブロックとこの現画像ブロックに関連する参照画面内の所定の大きさの探索領域のすべての評価点のうちの代表評価点に対応する参照画像ブロックとのブロックマッチング処理を複数の予測モードに従って並列態様で実行し、複数の予測モード各々に対する最もよい相関度を示す最適代表評価点を求める第１の算出手段と、この第１の算出手段の出力に従って最も高い相関度を示す最適予測モードおよび対応の最適代表評価点を決定する第１の決定手段と、この第１の決定手段からの最適予測モードおよび最適代表評価点に従って、この最適評価点を含む探索領域内の所定の領域内のすべての評価点に対し最適予測モードに従ってブロックマッチング処理を行ない、各参照画像ブロックに対する評価関数値を求める第２の算出手段と、この第２の算出手段からの評価関数値に従って前記現画像ブロックに対する動きベクトルを決定する動きベクトル決定手段を含む。

【００７２】請求項２２に係る動きベクトル検出装置は、現画面画像のＱ行Ｐ列の画素からなる現画面ブロックの各画素に対応して行列状に配置されるプロセッサと、Ｐ列の各列に対応して配置され、各々がＲ個の参照画面画像画素データを格納するデータ格納手段とを有するプロセッサアレイを含む。プロセッサとデータ格納手段とは１次元的に一方向に沿って参照画面画像画素データを転送するように相互接続される。プロセッサの各々は、対応の現画面画像画素データを格納する第１のレジスタと、与えられた参照画面画像画素データを格納する第２のレジスタ手段と、この第１および第２のレジスタ手段の格納する画素データに所定の演算処理を施して各プロセッサに格納された参照画面ブロックと現画面ブロックの類似度を示す評価関数値の成分を出力する演算手段を含む。

【００７３】請求項２２に係るベクトル検出装置は、さらに、１サイクル当たり１画素データの速度で、参照画面画像画素データをプロセッサアレイへ与えるデータ印加手段と、プロセッサアレイから出力される評価関数値成分を総和して１サイクル当たり１評価関数値を生成する評価値生成手段と、この評価値生成手段から所定数のサイクルにわたって与えられる評価値に従って現画面ブロックの動きベクトルを検出する動きベクトル決定手段と、所定数サイクルをＡ・（Ｐ・（Ｑ＋Ｒ））サイクルに設定する手段を備える。Ａは、任意の自然数である。請求項２３に係るベクトル検出装置は、各々が、Ｑ行Ｐ列の画素を含みかつ異なる現画面画像ブロックに対応して配置されるＡ個のプロセッサアレイを含む。Ａ個のプロセッサアレイの各々は、対応の現画面ブロックの各画素に対応して行列状に配置されるプロセッサと、プロセッサの各列に対応して配置され、各々がＲ個の参照画面画像画素データを格納するデータ格納手段とを有する。プロセッサとデータ格納手段とは１次元的に一方向に沿って参照画面画像画素データを転送するように相互接続される。プロセッサの各々は、対応の現画面ブロックの画素データを格納する第１のレジスタ手段と、与えられた参照画面画像画素データを格納する第２のレジスタ手段と、これら第１および第２のレジスタ手段に格納された画素データに所定の演算処理を施して対応の現画面ブロックと与えられた参照画面画像ブロックとの類似度を示す評価関数値の成分を出力する演算手段とを含む。

【００７４】請求項２３に係るベクトル検出装置はさらに、Ａ個のプロセッサアレイ各々に対応して設けられ、各々が対応のプロセッサの出力する評価関数値成分から評価関数値を生成するＡ個の評価値生成手段と、Ａ個の評価値生成手段対応して設けられ、対応の評価値生成手段のＡ・Ｐ（Ｑ＋Ｒ）サイクルにわたって与えられる評価関数値に従ってそれぞれ動きベクトルを決定するＡ個の動きベクトル決定手段と、Ａ個のプロセッサアレイの現画面ブロック画素データ全体をＡ・Ｐ（Ｑ＋Ｒ）サイクルごとに順次更新する手段とを備える。請求項２４に係るベクトル検出装置は、Ｑ行Ｐ列の画素に含む現画面ブロックの各画素に対応して行列状に配置される複数のプロセッサを含む。複数のプロセッサの各々は、対応の

現画面画像画素データを格納する第１のレジスタ手段と、与えられた参照画面ブロック画像画素データを格納する第２のレジスタ手段と、これら第１および第２のレジスタ手段の格納する画素データに所定の演算処理を施して参照画面ブロックと現画面ブロックとの類似度を示す評価値成分を出力する演算手段とを備える。

【００７５】請求項２４に係るベクトル検出装置は、さらに、プロセッサの各列に対応して配置され、各々がＲ個の参照画面画像画素データを格納するデータ格納手段を含む。プロセッサアレイは各々が（Ｐ／ｂ）・（Ｑ／ｃ）の画素に対応する複数のサブブロックに分割され、これらサブブロックは行列状に配置される。同じ列に配置されたサブブロックに対しては同じ参照画面ブロックの画素データが与えられ、また各列において、１つのサブブロックはプロセッサとデータバッファが１次元的に参照画面ブロック画素データを転送するように相互接続され、同じ列における残りのサブブロックは各列が一方方向にのみ沿って参照画面ブロック画素データを転送するようにプロセッサが接続され、同じ列に配置されたサブブロックはデータバッファを共有し、各サブブロックはデータ格納手段を介して参照画面画像画素データを入力する。

【００７６】請求項２４に係るベクトル検出装置は、さらに、サブブロック各々から（Ｐ／ｂ）・｛（Ｑ／ｃ）＋Ｒ｝・ｂ・ｃサイクルにわたって与えられる評価関数値成分を各サイクルごとに加算して評価関数値をそれぞれ生成し、かつ生成した評価関数値に従って対応のサブブロックそれぞれについて動きベクトルを決定する手段と、複数のサブブロック各々に互いに異なる現画面ブロックの画素データを格納するとともに、（Ｐ／ｂ）・｛（Ｑ／ｃ）＋Ｒ｝ｂ・ｃサイクルごとに現画面ブロックを順次更新する手段を備える。請求項２５に係るベクトル検出装置は、現画面ブロックの画素に対応して行列状に配置される複数のプロセッサと、各プロセッサの列に対応して配置される参照画面画像画素データを格納するデータ格納手段とを含む。このプロセッサの各々は、第１の現画面ブロックの画素データを格納する第１のレジスタと、第２の現画面ブロックの対応の画素データを格納する第２のレジスタ手段と、与えられた参照画面ブロックの画素データを格納する第３のレジスタ手段と、第１および第２のレジスタ手段の出力の一方の選択する選択手段と、選択手段が出力する画素データと第３のレジスタ手段に格納された画素データとに所定の演算処理を施して評価関数値成分を生成する演算手段を含む。

【００７７】請求項２５に係るベクトル検出装置は、さらに、このプロセッサアレイから出力される評価関数値成分に従って評価関数値を生成する評価値生成手段と、この評価値生成手段から与えられる評価関数値に従って動きベクトルを決定する動きベクトル決定手段と、第１および第２の現画面ブロックの一方についての動きベクトル検出サイクルの間に他方の現画面ブロック画素データを格納するレジスタを別の現画面ブロックの画素データで更新する手段と、動きベクトル検出サイクルごとに選択手段の選択するレジスタを切換える手段とを備える。請求項２６に係るベクトル検出装置は、現画面内の動きベクトル検出対象となる所定のサイズの現画像ブロックと最小画面内の現画像ブロックに関連する領域内の複数の参照画像ブロック各々との類似度を示す評価関数値を生成する評価値生成手段と、この評価値生成手段から与えられる評価関数値に従って現画像ブロックの動きベクトルを決定する決定手段を含む。この決定手段は、評価関数値が同じとき、参照画面ブロックの現画面ブロックについてのベクトル（Ｈ，Ｖ）に従って（ｉ）｜Ｈ｜＋｜Ｖ｜の小さいほうの参照画像ブロックのベクトルを動きベクトル候補として選択する第１の選択手段と、（ii）｜Ｈ｜＋（－Ｖ＋α）の小さいほう参照画像ブロックのベクトルを動きベクトルの候補として選択する第２の選択手段と、（iii ）｜Ｈ｜＋（Ｖ＋α）の小さいほうの参照画像ブロックのベクトルを動きベクトルの候補として選択する第３の選択手段と、動作モード指定信号に応答して、第１ないし第３の選択手段のいずれか１つを活性状態とする手段を備える。ここで、αはＹ方向の参照画面内の現画像ブロックに関連する領域の範囲に応じて決定される定数である。

【００７８】請求項２７に係るベクトル検出装置は、現画面画像所定のサイズの現画面ブロックとこの現画面ブロックに関連する参照画面内の所定領域に含まれる複数の参照画面ブロックのそれぞれとを複数の予測モードそれぞれに従ってブロックマッチング処理を施して整数精度での動きベクトルを複数の予測モード各々について決定する整数精度ベクトル決定手段と、この整数精度ベクトル決定手段からの現画面ブロック画素データとこの整数精度ベクトル決定手段により決定された複数の予測モード各々の動きベクトルで示される複数の参照画面ブロックそれぞれの画素データを格納する格納手段と、この格納手段に格納された現画面ブロック画素データと複数の参照画面ブロック各々とを分数精度でブロックマッチング処理を行なって複数の予測モード各々の分数精度での動きベクトルを決定する分数制度ベクトル決定手段とを備える。請求項２８に係るベクトル検出装置は、現画面の所定のサイズの現画面ブロックと、所定のサイズの探索領域内の複数の参照画面画像に対応する複数の参照画面ブロック各々との類似度を示す評価関数値を求める評価値算出手段と、探索領域の有効領域を設定する設定手段と、この設定手段により設定された有効領域に含まれる参照画面ブロックに対する、評価値算出手段の評価関数値に従って現画面ブロックの動きベクトルを決定する手段とを備える。

【００７９】

【作用】請求項１に係る発明においては、評価手段が複

数の予測モード（予測画像検出方式）各々に対応する評価関数値を並列に算出し、動きベクトル決定手段がこの評価手段の出力に従って各予測モードに対する動きベクトルを決定しており、複数の予測モードに対応して動きベクトルを高速かつ装置規模を増大させることなく検出することができる。

【００８０】請求項２に係る発明においては、評価手段が現画像ブロックの画素に対応して配置されたプロセッサを有し、これらのプロセッサが対応の現画像ブロックの画素と対応の参照画像ブロックの対応の画素との演算を行なって評価関数値成分を出力しているので、複数の予測モードに共通の演算を同じ回路構成を用いて実行することができ、装置規模を低減することができる。

【００８１】請求項３に係る発明においては、第１の演算手段が一方方向に沿ってのみ画素データを転送することが要求されるだけであり、画素データ転送時の電流消費を低減することができる。

【００８２】請求項４に係る発明においては、複数の予測モードが奇数フィールド予測モード、偶数フィールド予測モードおよびフレーム予測モードを含んでおり、これらの複数の予測モード各々に対応して動きベクトルを高速で検出することができる。

【００８３】請求項５に係る発明においては、複数の予測モードが、上半分ブロック予測モード、下半分ブロック予測モード、およびフィールド予測モードを含んでおり、画面形式がフィールド画面の場合、これら複数の予測モードに対応して高速に動きベクトルを検出することができる。

【００８４】請求項６に係る発明においては、分類手段により第１の演算手段の出力が分類され、かつ切換手段によりこの分類の構成を変更しているため、任意の予測モードに容易にかつ柔軟に対応することができる。

【００８５】請求項７に係る発明においては、画面形式指定信号に従って入力画像データの画面形式を設定して評価手段へ与えているため、任意の画面形式の画像データに対しても所望の予測モードに従って動きベクトルを検出することができる。

【００８６】請求項８に係る発明においては、プロセッサ間に配置されたバッファ手段の有効格納画素数が変更可能なため、動きベクトル探索範囲となるサーチエリアの大きさを容易に変更することができる。

【００８７】請求項９に係る発明においては、第３の演算手段はサーチエリアの評価点を間引いて評価関数値の算出を行なって最適代表評価点を複数の予測モードそれぞれに対して求め、第１の演算手段の動きベクトル検索範囲を決定し、第１の演算手段は決定されたサーチエリア内で全評価点を用いて評価関数値を求めているため、すべての評価点に対し評価関数値を求める必要がなく、装置規模を増大させることなく高速で複数の予測モードに従って動きベクトルを検出することができる。

【００８８】請求項１０に係る発明においては、第１の演算手段の評価関数値の演算領域が複数の所定モード各々に対応して領域設定手段により決定され、第１の演算手段は決定された演算領域内で複数の予測モードに従って評価関数値を算出しており、装置規模を増大させることなく高速で複数の予測モードに従って最適な動きベクトルを決定することができる。

【００８９】請求項１１に係る発明においては、第１の演算手段は領域設定手段により１つの演算領域内で複数の予測モードに従って評価関数値を算出しているので、装置規模をさらに低減して複数の予測モードに従って最適な動きベクトルを検出することができる。

【００９０】請求項１２に係る発明においては、領域設定手段により、複数の予測モードのうちの最適予測モードおよび演算領域が決定され、第１の演算手段はこの最適予測モードに従って演算領域内の評価点に対する評価関数値を算出し、この算出結果に従って動きベクトルを検出しているため、装置規模をさらに低減することができるとともに、広い範囲のサーチエリアに対して高速でかつ高精度で動きベクトルを決定することができる。

【００９１】請求項１３に係る発明においては、プロセッサが現画像ブロックの画素に対応して配置されており、プロセッサ出力を複数の予測モードに従って容易に分類することができる。したがって、複数の予測モード（偶数フィールド予測モード、奇数フィールド予測モードおよびフレーム予測モード）に従う評価関数値の算出を並列態様で実行することができ、装置規模を増大させることなく高速で動きベクトルを検出するすることができる。

【００９２】請求項１４に係る発明においては、プロセッサが現画像ブロックの画素に対応して配置されており、プロセッサの出力を複数の予測モードに従って容易に分類することができる。したがって、複数の予測モード（上半分ブロック予測モード、下半分ブロック予測モード、およびフィールド予測モード）に従って評価関数値を並列に算出することができ、装置規模を増大させることなく高速で動きベクトルを容易に検出することができる。

【００９３】請求項１５に係る発明においては、プロセッサが現画像ブロックの画素に対応して配置されており、プロセッサの出力を複数の予測モードに従って容易に分類することができる。また、分類手段がのプロセッサの出力を４つの属性にしたがって分類し、この４分類を構成を切換えて３分類に変更しているため、異なる予測モードに対しても装置構成を何ら変更することなく対応することができ、汎用性の高い動きベクトル検出装置を実現することができる。

【００９４】請求項１６に係る発明においては、バッファ手段が入力画像データの入出力順序を変更して評価手段へ与えているため、さまざまな画面構成の入力画像デ

ータに対しても所定の複数の予測モードに従って動きベクトルを何ら装置構成を変更することなく検出することができる。

【００９５】請求項１７に係る発明においては、プロセッサ間に配置されたバッファ手段の有効格納画素数は変更可能であるため、動きベクトルのサーチエリアの大きさを容易に変更することができる。

【００９６】請求項１８に係る発明においては、第１および第２の算出手段により階層探索方式に従ってかつ複数の予測モードに従って各予測モードに対する動きベクトルを検出し、次いでベクトル決定手段によりこれら複数の予測モードに対応する動きベクトルから最適動きベクトルを決定しているため、装置規模を増大させることなく効率的に動きベクトルを検出することができる。

【００９７】請求項１９に係る発明においては、第１の算出手段により評価点を間引いて最適評価点候補を決定し、次いで第２の算出手段により複数の予測モードそれぞれに対して最適ベクトルを算出し、最終的に動きベクトルを決定しているため、装置規模を何ら増大させることなく高速で動きベクトルを検出することができる。

【００９８】請求項２０に係る発明においては、第１の算出手段により動きベクトルサーチエリア内の評価点を間引いて評価関数値の演算を実行して最適評価点の候補を算出し、この最適評価点候補に従って第２の算出手段により複数の予測モードに従って評価関数値をその周辺領域に含まれる評価点に対して算出し、この算出結果に従って動きベクトルを検出しているため、第２の算出手段の評価点探索領域を複数の予測モードに対して共通とすることができ、装置規模を増大させることなく高速で動きベクトルを検出することができる。

【００９９】請求項２１に係る発明においては、第１の算出手段により評価点を間引いて複数の予測モードに従って評価関数値を求め、この第１の算出手段の出力に従って最適予測モードおよび対応の最適評価点を求め、この最適予測モードおよび最適評価点に従って最適評価点近傍領域の評価点全てを用いて評価関数値を求め、これらの評価関数値から動きベクトルを最終的に決定しているため、第２の算出手段の規模を何ら増大させることなく複数の予測モードに従って動きベクトルを高速で検出することができる。請求項２２に係る動きベクトル検出装置においては、所定サイクルＡ・（Ｐ・（Ｑ＋Ｒ））の係数Ａを変更することにより、動きベクトル探索範囲を変更することができ、複数の探索範囲に容易に対応することのできる動きベクトル検出装置が得られる。

【０１００】請求項２３に係る動きベクトル検出装置において、Ａ個のプロセッサアレイに異なる現画面ブロックの画素データが格納され、Ａ・Ｐ（Ｑ＋Ｒ）サイクルの処理単位サイクルごとに順次プロセッサアレイの格納する現画面ブロックの画素データが全体として更新され、かつ各プロセッサアレイには同じ参照画面ブロック画素データが与えられており、パイプライン的にＡ個のプロセッサアレイにおいて動きベクトルを決定することができ、高速かつ効率的に動きベクトルを検出する装置が得られる。請求項２４に係る動きベクトル検出装置においては、複数のサブブロック各々が異なる現画面ブロックに対応して配置されて、各サブブロックが現画面ブロックの動きベクトルを検出し、同じ列のサブブロックへは同じ参照画面ブロックの画素データを与えかつ処理単位サイクルごとに順次サブブロックの現画面ブロックの画素データを更新しているため、大きな現画面ブロックを複数のサブブロックに分割して各サブブロックについてパイプライン的に動きベクトルを効率的に検出することができる。

【０１０１】請求項２５に係る動きベクトル検出装置においては、プロセッサの第１および第２のレジスタ手段の画素データが処理単位サイクルごとに更新されかつ選択手段によりこれらの第１および第２のレジスタ手段が切換えられており、パイプライン的に現画面画像の各ブロックに対する動きベクトルを順次検出することができる。請求項２６に係る動きベクトル検出装置においては、動作モードに応じて第１ないし第３の選択手段それぞれの探索範囲の中央部、中央部下端部および中央部上端に近いベクトルのいずれかを動きベクトル候補として選択しており、この動きベクトル検出装置は単体、および複数で用いる構成に応じて最適な動きベクトル候補を選択することができ、様々なシステム構成に容易に対応することのできる動きベクトル検出装置を実現することができる。

【０１０２】請求項２７に係る動きベクトル検出装置においては、整数精度ベクトル決定手段による複数の予測モード各々の動きベクトルに従ってさらに分数精度での動きベクトルを各予測モードについて検出しており、高精度の動きベクトルを複数の予測モードそれぞれに対して検出することができる。請求項２８に係る動きベクトル検出装置においては、探索領域の有効領域が設定され、その有効領域内の参照場面ブロックについての評価関数値に従って動きベクトルが検出されており、たとえば現画面端に位置する現画面ブロックに対して必要な参照画面ブロックについての評価関数値のみを用いて動きベクトルを検出することができ、正確な動きベクトルを検出することができる。また、処理用途に応じて各現画面ブロックについて探索範囲を制限することができ、処理用途に応じた探索範囲を各現画面ブロックについて設定することのできる汎用性の高い動きベクトル検出装置を実現することができる。

【０１０３】

【実施例】

［全探索方式の動きベクトル検出装置］図１はこの発明の一実施例である動きベクトル検出装置の全体の構成を概略的に示すブロック図である。図１においては、フレ

ームを単位として画素を符号化する場合の構成が示される。

【０１０４】図１において、動きベクトル検出装置は、サーチエリア内の所定の領域に含まれるサーチウィンドウ（サーチウィンドウについて後に説明する）画素データＹとテンプレートブロック画素データＸとを受け、受けたデータＸおよびＹをそれぞれ所定のタイミングで出力する入力部２と、入力部２から与えられたデータＸおよびＹに基づいて、１つのテンプレートブロックに対する変位ベクトルについての３つの評価関数値（差分絶対値和）を算出する演算部１と、演算部１で求められた３つの評価関数値Σ｜ａ－ｂ｜、Σ_o｜ａ－ｂ｜およびΣ_e｜ａ－ｂ｜を受け、それぞれの評価関数値について最小の評価関数値を求め、それぞれの最小評価関数値に対応する変位ベクトルを動きベクトルと決定する比較部３を含む。ここでΣ｜ａ－ｂ｜は、現画像ブロック（テンプレートブロック）の全画素を用いて評価関数値を求めるフレーム予測モードの評価関数値に対応する。ａおよびｂはテンプレートブロック画素データおよび参照画像ブロック（サーチウィンドウブロック）の対応の画素データを示す。Σ_o｜ａ－ｂ｜は奇数フィールドに含まれる画素を用いて予測画像を検出する奇数フィールド予測モードに従う評価関数値に対応する。Σ_e｜ａ－ｂ｜は、偶数フィールドに含まれる画素を用いて予測画像を検出する偶数フィールド予測モードに従って算出された評価関数値を表わす。

【０１０５】比較部３からそれぞれの予測モードに従って決定された動きベクトル、すなわちフレーム予測モードに従って決定されたブロック動きベクトルＶｂ、奇数フィールド予測モードに従って決定された奇数フィールドブロック動きベクトルＶｏ、偶数フィールド予測モードに従って決定された偶数フィールドブロック動きベクトルＶｅが並列に出力される。

【０１０６】演算部１は、後にその構成は詳細に説明するが、テンプレートブロックの各画素に対応して２次元アレイ状に配置される複数の要素プロセッサを含むプロセッサアレイ１０と、プロセッサアレイ１０の各要素プロセッサが出力する演算結果値（本実施例においては差分絶対値）を予測モードに従って分類して各分類ごとに加算して総和を求める総和部１２を含む。総和部１２から各予測モードに対応する評価関数値Σ｜ａ－ｂ｜、Σ_o｜ａ－ｂ｜、およびΣ_e｜ａ－ｂ｜が並列に出力される。

【０１０７】プロセッサアレイ１０に含まれる要素プロセッサは、互いに異なるテンプレートブロック画素データを格納し、このテンプレートブロックに関連するサーチエリア内の参照画像ブロックの画素データが要素プロセッサを介して一方方向に沿って順次伝達される。プロセッサアレイ内の要素プロセッサでは、テンプレートブロックの画素データと１つの参照画像ブロック（以下、サーチウィンドウブロックと称す）の画素データとの演算が実行される。要素プロセッサの出力は並列に出力される。したがって、各要素プロセッサは、１つのテンプレートブロックと１つのサーチウィンドウブロックとの相関度を示す評価関数値の成分を算出する。プロセッサアレイ１０においては、テンプレートブロック画素データはこのテンプレートブロックについての動きベクトルを求めるサイクル中、常時格納されている。サーチウィンドウブロック画素データは、１演算サイクル（評価関数算出サイクル）ごとにこのプロセッサアレイ１０内の要素プロセッサ間を１画素分シフトされる。

【０１０８】要素プロセッサはテンプレートブロックの各画素に対応して配置される。奇数フィールド予測モードおよび偶数フィールド予測モードでは、それぞれテンプレートブロック（およびサーチウィンドウブロック）の奇数フィールドに属する画素および偶数フィールドに属する画素を用いて評価関数値を算出する。フレームは、偶数フィールドと奇数フィールドとを含んでおり、画面上１水平走査線ごとに奇数フィールド画素と偶数フィールド画素とが交互に配置される。したがって、要素プロセッサの出力も、偶数フィールド画素に対応して配置される要素プロセッサの出力は偶数フィールド予測モードに従う評価関数値の成分を表わし、奇数フィールド画素に対応して配置される要素プロセッサの出力は奇数フィールド予測モードに従う評価関数値の成分を表わす。したがってこれらの要素プロセッサの出力をそのプロセッサアレイ内の配置位置に応じて分類することにより、複数の予測モードそれぞれに対応する評価関数値を算出することができる。フレーム予測モードの場合には偶数フィールドおよび奇数フィールドの評価関数値を単に加算すれば得られる。次に、図１に示す演算部の具体的構成について説明する。

【０１０９】図２は、この実施例において利用されるテンプレートブロックおよびサーチエリアの大きさを示す図である。テンプレートブロック２０は、Ｑ行Ｐ列に配置された画素を含む。サーチエリア２２は、水平方向における探索範囲が＋ｔ１～－ｔ２であり、垂直方向の探索範囲が＋ｒ１～－ｒ２である。すなわち、サーチエリア２２は、（ｔ２＋Ｐ＋ｔ１）×（ｒ２＋Ｑ＋ｒ１）の画素を含む。

【０１１０】図３は、図１に示すプロセッサアレイに含まれる要素プロセッサの構成を概略的に示す図である。図３において、要素プロセッサＰＥは、テンプレートブロック画素データＸを格納するための縦続接続されたＭ個のデータレジスタ２５－１～２５－Ｍを含む。データレジスタ２５－１～２５－Ｍには、互いに異なるテンプレートブロック画素データが格納される。要素プロセッサＰＥは、また、サーチウィンドウ画素データを格納するためのＮ段の縦続接続されたデータレジスタ２６－１～２６－Ｎを含む。ＮはＭの整数倍（ｎ倍）である。ま

た図２に示すテンプレートブロック２０の行の数Ｑは、データレジスタ２５－１～２５－Ｍの段数Ｍの整数倍（ｍ倍）である。

【０１１１】要素プロセッサＰＥにおいては、Ｍ個のデータレジスタ２５－１～２５－Ｍに格納されたテンプレートブロック画素データを用いて所定の演算（たとえば差分絶対値演算）が実行される。この演算の場合、サーチウィンドウ画素データ格納用のデータレジスタ２６－１～２－６Ｎがテンプレートブロック画素データ格納用データレジスタ２５－１～２５－Ｍと１対１に対応し（Ｎ＝Ｍ）、各対応のレジスタの格納データを利用して所定の演算が実行されてもよい。別の組合わせが用いられてもよい。

【０１１２】要素プロセッサＰＥは、Ｍ個のテンプレートブロック画素データ（以下、単にテンプレートデータと称す）に関する演算を実行する。要素プロセッサＰＥにおける演算手段（図３には示さず）は、このＭ個のデータレジスタ２５－１～２５－Ｍに対して多重化態様で利用される。したがって、この場合には、演算手段は要素プロセッサＰＥ１つについて１つのみ設けられる。要素プロセッサＰＥ内にテンプレートデータ格納用のデータレジスタを複数個設けることにより要素プロセッサの数を低減する。また、複数のテンプレートデータ格納用レジスタの出力を多重化態様で選択して演算手段へ与えて演算を実行することにより、サーチウィンドウブロック画素データ（以下、サーチウィンドウデータと称す）の転送周期内において複数回演算を行なうことにより、１つの要素プロセッサＰＥから評価関数値の複数の成分が多重化態様で出力されるため、プロセッサアレイ１０の要素プロセッサの出力を加算する総和部の構成を小さくすることができる（総和部に対する入力信号線の数を少なくすることができるためである）。サーチウィンドウデータＹおよびテンプレートデータＸは、図３に矢印で示すように一方方向にのみ伝達される。

【０１１３】図４は、図１に示すプロセッサアレイ１０の構成を示す図である。図４において、プロセッサアレイ１０はＰ列に配置された線形プロセッサアレイＬＡ１～ＬＡＰを含む。線形プロセッサアレイＬＡ１～ＬＡＰは同一構成を備え、各々縦続形態で配置されるｍ個の要素プロセッサＰＥ１～ＰＥｍと、Ｒ（＝ｒ１＋ｒ２）個のサーチウィンドウデータを格納するとともに遅延手段としても機能するデータバッファＤＬを含む。

【０１１４】要素プロセッサＰＥ１～ＰＥｍは、それらが属する線形プロセッサアレイＬＡ（ＬＡ１～ＬＡＰを総称的に示す）内においては一方方向（図４における垂直方向）に沿ってサーチウィンドウデータおよびテンプレートデータを伝達する。隣接する線形プロセッサアレイへのデータ転送時においては、最上流の要素プロセッサＰＥ１はサーチウィンドウデータを上流の線形プロセッサアレイ（図４において左側）に含まれるデータバッファＤＬへ与え、テンプレートデータは上流の線形プロセッサアレイの最下流の要素プロセッサＰＥｍへ伝達する。すなわち、サーチウィンドウデータＹは要素プロセッサＰＥおよびデータバッファＤＬを介してプロセッサアレイ内を順次一方方向に沿って伝達され、テンプレートデータは要素プロセッサＰＥのみを介して一方方向に沿って伝達される。図４に示す構成においては、要素プロセッサＰＥは行および列のマトリックス状に配列されているが、線形プロセッサアレイは前段の線形プロセッサアレイの最上流の要素プロセッサが出力するサーチウィンドウデータおよびテンプレートデータを受けている。したがって線形プロセッサアレイＬＡ１～ＬＡＰが縦続形態で接続されており、プロセッサアレイは、シストリックアレイを構成し、実質的には１次元プロセッサアレイと等価である。

【０１１５】データバッファＤＬは、前述のごとく遅延機能またはバッファ機能を備えており、与えられたサーチウィンドウデータをファーストイン・ファーストアウト（ＦＩＦＯ）態様で出力する機能を備える。データバッファＤＬとしては、Ｒ個のシフト機能付データラッチ（Ｒ段のシフトレジスタ）が用いられてもよく、またＲ個のデータを格納するレジスタファイルが用いられてもよい。

【０１１６】今、図５（Ａ）に示すように、１フレームの画像３６を８×８の６４個のマクロブロックに分割し、各マクロブロックごとに動きベクトルを検出する場合を考える。１フレームの画像３６において斜線で示すマクロブロックをテンプレートブロックＴＢ１とする。このテンプレートブロックＴＢ１に対し３つの予測モードそれぞれに従って動きベクトルを検出する。

【０１１７】図５（Ｂ）に示すように、今テンプレートブロックＴＢ１が１６行１６列に配置されたテンプレートデータａ（０，０）～ａ（１５，１５）を含むとする。テンプレートブロックＴＢ１では、各行ごとに偶数フィールドに属する画素と奇数フィールドに属する画素が交互に配置される。図５（Ｂ）に示すテンプレートブロック画素ａ（０，０）～ａ（１５，１５）をすべて用いて評価関数値を算出することにより動きベクトルを検出する方式は、フレーム予測方式に対応し、ブロック動きベクトルＶｂが決定される。

【０１１８】図６（Ａ）に示すように、テンプレートブロックＴＢ１における第０行、第２行、…第１４行の偶数行に配置された偶数フィールドに含まれる画素ａ（２ｋ，０）～ａ（２ｋ，１５）を用いて評価関数値を算出する予測モードは、偶数フィールドブロック動きベクトルＶｅを与える。ただし、ｋ＝０～７である。

【０１１９】また、図６（Ｂ）に示すように、テンプレートブロックＴＢ１における第１列、…第１５列に配置された画素ａ（２ｋ＋１，０）～ａ（２ｋ＋１，１５）を用いて評価関数値を算出する予測モードは奇数フィー

ルド予測モードであり、奇数フィールドブロック動きベクトルＶｏを与える。

【０１２０】予測画像を検出するための参照画像探索のために、プロセッサアレイ１０においては、図７に示すように前フレーム画像３５における斜線で示す３つのマクロブロックＭＢ１、ＭＢ２およびＭＢ３が格納される。図５（Ａ）に示すテンプレートブロックＴＢ１に含まれる画素のデータが要素プロセッサＰＥの各データレジスタ２５－１～２５－Ｍに格納される。１個の要素プロセッサＰＥには、Ｑ／ｍ個の垂直方向に配置されたテンプレートブロック画素データが格納される。一方、サーチウィンドウブロック画素データについては、１つの要素プロセッサＰＥに、Ｑ・ｎ／ｍ個の垂直方向に隣接する画素データが格納される。したがって、図８に示すように、プロセッサアレイの要素プロセッサには、垂直方向Ｑ個、水平方向Ｐ個の合計Ｐ・Ｑ個のサーチウィンドウ画素データが格納される。Ｐ・Ｑ個の画素データを以下の説明おいてはサーチウィンドウブロック画素データと称す。残りのＲ（＝ｒ１＋ｒ２）・Ｐ個のサーチウィンドウ画素データはデータバッファＤＬに格納される。このデータバッファＤＬに格納される画素データが構成するブロックを以下の説明おいてはサイドウィンドウブロック４４と称す。

【０１２１】この図８に示すサーチウィンドウブロック４２およびサイドウィンドウブロック４４を併わせてサーチウィンドウ４０と以下の説明では称する。

【０１２２】一般に、図９に示すように、１個の要素プロセッサＰＥには、Ｍ個のテンプレートブロック画素データとＮ個のサーチウィンドウ画素データとが格納される。１つの線形プロセッサアレイの要素プロセッサＰＥ１～ＰＥｍによりＱ個の垂直方向に隣接するサーチウィンドウ画素データおよびテンプレート画素データが格納される。次に具体的な動作について説明する。

【０１２３】以下の説明を簡単にするために、以下の条件を仮定する。

テンプレートブロックサイズ：Ｐ＝Ｑ＝１６

動きベクトルの探索範囲：ｒ１＝ｒ２＝１６、ｔ２＝０，ｔ１＝１５，ｍ＝１６，Ｍ＝Ｎ＝１

すなわち、上述の仮定では、図１０に示すように、動きベクトル探索範囲２２は、フレーム画像について（１６＋１６＋１６）×（１６＋１５）＝１４８８個の画素を含む。テンプレートブロック２０のサイズは１６×１６＝２５６画素である。またＭ＝Ｎ＝１の条件から、要素プロセッサＰＥは、１つのサーチウィンドウブロック画素データと１つのテンプレートブロック画素データを格納する。

【０１２４】図１１は、上述の仮定の下での奇数フィールドまたは偶数フィールドの画素についてのサーチエリアの構成を示す。図１１に示すように、フレーム構成のテンプレートブロックは、偶数フィールドの画素と奇数フィールドの画素を含む。したがって、偶数フィールドまたは奇数フィールドの画素で構成されるサブテンプレートブロック２０ａは、１６画素×８画素の構成を備える。同様にサーチエリアも、偶数フィールドの画素と奇数フィールドの画素とを含むため、偶数フィールドまたは奇数フィールドの画素に対しては、サブサーチエリア２２ａは、（８＋８＋８）×（１６＋１５）＝７４４画素となる。すなわち、フレーム単位での予測モードとフィールド単位での予測モードでは、変位ベクトルが異なる（フレーム単位のマクロブロックの行数が、フィールド単位のマクロブロックの行数の２倍となるためである）。

【０１２５】図１２に、動きベクトル検出動作における最初の動作サイクルにおけるプロセッサアレイ内のデータ格納状況を示す。図１２において、１６行×１６列の画素データからなるテンプレートブロック４３の画素データがプロセッサアレイ内の各対応の要素プロセッサＰＥに格納される。これに対応して、１６行×１６列のサーチウィンドウブロック４２の画素データがプロセッサアレイ内の各要素プロセッサＰＥに格納される。この状態は、テンプレートブロック２２に対しては、フレーム変位ベクトル（０，－１６）に対応する。ここでフレーム変位ベクトルは、フレーム予測モードに従う予測動作を行なう際に用いられる変位ベクトルを示す。奇数フィールドの画素で構成される奇数サブテンプレートブロック２０ａに対しては、このサーチウィンドウブロック４２内の奇数フィールドの画素に対する変位ベクトルが用いられるため、奇数フィールド変位ベクトル（０，－８）に対応し、同様、偶数フィールドに含まれる画素のみを含む偶数サブテンプレートブロック２０ａに対しても同様、偶数フィールドに対する変位ベクトル、すなわち偶数フィールド変位ベクトル（０，－８）に対応する。

【０１２６】この状態において、各要素プロセッサＰＥが格納したテンプレートブロック画素データと対応のサーチウィンドウブロック画素データとの評価関数の成分である差分絶対値を求める。各要素プロセッサＰＥにおいて求められた差分絶対値は図１に示す総和部１２へ伝達される。総和部１２は、奇数サブテンプレートブロックに対応する要素プロセッサから伝達された差分絶対値の総和（奇数サブブロック評価関数値）と、偶数サブテンプレートブロックに対応して配置された要素プロセッサから伝達された差分絶対値の総和（偶数サブブロック評価関数値）とを別々にかつ並列に計算する。この偶数サブブロック評価関数値および奇数サブブロック評価関数値を計算した後両者を加算する。この加算演算は、プロセッサアレイのすべての要素プロセッサから伝達された差分絶対値の総和を求めたことに対応し、これによりテンプレートブロックに対応するフレーム予測モードに従う評価関数値（ブロック評価関数値）が求められる。

上述の動作により、３つの予測モードに従うそれぞれの変位ベクトルに対する評価関数値が求められる。各予測モードは単に用いられる画素が異なっているだけであり、したがって要素プロセッサＰＥの出力を分類することにより、複数の予測モードに対する評価関数値を並列に算出することができる。

【０１２７】次いで、テンプレートブロック画素データはプロセッサアレイ内の各要素プロセッサＰＥ内に保持した状態で、サーチウィンドウブロック画素データのみを１画素分転送する。

【０１２８】この状態においては、図１３（Ａ）に示すように、サーチウィンドウブロック４２における最上行のデータが隣接列のデータバッファＤＬへ転送され、応じてこのサーチウィンドウブロック４２の最初のデータがシフトアウトされる。このシフトアウトと並行して、新たにサーチウィンドウブロック画素データが入力される。シフトアウトされるサーチウィンドウブロック画素データおよび新たにシフトインされるサーチウィンドウ画素データを図１３（Ａ）において二重斜線領域で示す。

【０１２９】この状態においては、図１３（Ｂ）に示すように、プロセッサアレイ１０内の要素プロセッサＰＥ内には、サーチウィンドウ４０における１行下方向にずれたサーチウィンドウブロック４２ａの画素データが格納される。すなわち、図１３（Ｂ）に示すように、線形プロセッサアレイＬＡ１～ＬＡ１６各々において１画素分サーチウィンドウ画素データが図の上方向にシフトされる。線形プロセッサアレイＬＡ１～ＬＡ１６において、新たにシフトインされるサーチウィンドウ画素データはデータバッファＤＬに格納される。プロセッサ群（ＰＥ群）に新たにシフトインされるデータは対応のデータレジスタから与えられる。すなわち、要素プロセッサ群（ＰＥ群）においては、この１画素のシフト動作により、サーチウィンドウの１画素行下方向にずれたサーチウィンドウブロックが格納される。

【０１３０】この状態においては、図１３（Ｃ）に示すように、テンプレートブロック４３とサーチウィンドウブロック４２ａとのブロック変位ベクトルは（０，－１５）となる。

【０１３１】このシフト動作においては、サブウィンドウブロック４２ａでは、偶数フィールドの画素が１行シフトアウトされかつシフトインされている。すなわち偶数フィールドについてその変位ベクトルが（０，－１）移動している。奇数フィールドについては、サーチウィンドウブロック４２ａ内において含まれており、この奇数フィールドに対する変位ベクトルは変化しない。したがって、図１４（Ａ）にしめすように、偶数サブテンプレートブロック４３ａ（斜線領域にプロセッサが含まれる）は、奇数フィールドＯＦに対する変位ベクトル（０，－８）となる。

【０１３２】一方、図１４（Ｂ）に示すように、奇数サブテンプレートブロック４３ｂ（斜線領域）は、偶数フィールドＥＦに対する変位ベクトル（０，－７）となる。この状態において再び上述と同様の差分絶対値の総和演算が行なわれ、テンプレートブロックの評価関数値、偶数サブテンプレートブロックの評価関数値、および奇数サブテンプレートブロックの評価関数値が出力される。偶数サブテンプレートブロック４３ａの評価関数値としては、したがって偶数フィールドに対する評価関数値と奇数フィールドに対する評価関数値とが交互に出力される。同様に、奇数サブテンプレートブロック４３ｂに対しても奇数フィールドに対する評価関数値と偶数フィールドに対する評価関数値が交互に出力される。偶数フィールド予測モードにおいては、この偶数サブテンプレートブロックの画素を用い、奇数フィールドおよび偶数フィールド両者から動きベクトルを求め、同様奇数サブテンプレートブロックに対しても偶数フィールドおよび奇数フィールド両フィールドから動きベクトルを検出する。

【０１３３】上述の動作を繰返し、テンプレートブロック４３に対する変位ベクトルが（０，０）となったとき、偶数サブテンプレートブロック４３ａに対しては偶数フィールドに対する変位ベクトル（０，０）であり、奇数サブテンプレートブロック４３ｂに対しては奇数フィールドに対する変位ベクトル（０，０）となる。この状態では、図１５（Ａ）に示すように、サーチウィンドウ４０におけるサーチウィンドウブロック４２ｂは、テンプレートブロック４３の真裏に対応する。この状態においては、図１５（Ｂ）に示すように、プロセッサアレイ内に格納されるサーチウィンドウ４０のデータはその上部１６画素×１６画素の領域において１列図の右方向へずれた位置となっている。要素プロセッサ群にはサーチウィンドウブロック４２ｂおよびテンプレートブロック４３の画素データが格納され、残りのサイドウィンドウブロック４４の画素データはデータバッファ群内に格納される。このように、評価関数値算出動作にとって不要となるデータを１画素シフトアウトし、このシフトアウト動作と並行して新たに１画素分サーチウィンドウ画素データをシフトインすることにより、評価関数値算出動作と並行して新たに次の列のサーチウィンドウ画素データを格納することができる。

【０１３４】さらに評価関数値算出動作を繰返し実行し、テンプレートブロック４３に対する変位ベクトルが（０，＋１６）の状態になったとき、偶数サブテンプレートブロック４３ａに対しては偶数フィールドに対する変位ベクトル（０，＋８）であり、奇数サブテンプレートブロック４３ｂに対しては奇数フィールドに対する変位ベクトル（０，＋８）となる。この状態では、図１６（Ａ）に示すように、サーチウィンドウブロック４２ｃは、サーチウィンドウ４０の一番下の領域に配置され

る。この状態での３つの予測モードそれぞれの変位ベクトルに対する評価関数値が算出された後、プロセッサアレイからは１画素分不要となったサーチウィンドウデータＰＹ１がシフトアウトされ、新たにサーチウィンドウ画素データＰＹ２がシフトインされる。

【０１３５】この状態においては、図１６（Ｂ）に示すように斜線で示す領域５０の画素データがプロセッサアレイ内の要素プロセッサに格納されている。次のステップにおいては、サーチエリアにおいて１列図の右方向にずれたサーチウィンドウに対する評価関数値の算出を行なう。この次のステップに必要とされるサーチウィンドウの画素データをプロセッサアレイ内に準備するために、サーチウィンドウ画素データＰＹ２をシフトインした後さらに１５回シフト動作を繰返す。これにより図１６（Ｂ）に示す斜線領域５０において画素データＰＹ１と同一列に配置される１５個の画素データが順次シフトアウトされ、これと並行して画素データＰＹ２に続く１５個の画素データがシフトインされる。データバッファ群内には図１６（Ｂ）に示す３２行×１６列の画素データが格納されている。

【０１３６】領域５０には、図１７（Ａ）に示すように、要素プロセッサ内に画素データ群Ａ１～Ｇ１が格納され、残りの画素データ群Ａ２～Ｇ２はデータバッファ内に格納される。線形プロセッサアレイは画素データＡ１およびＡ２、Ｂ１およびＢ２というように同一のアルファベットで示される画素データ群を格納している。データバッファはファーストインファーストアウト態様でデータの入出力を行なう。したがって図１７（Ａ）の斜線領域で示す１６画素がシフトインされたとき、画素データＡ１がすべてシフトアウトされる。この状態では、図１７（Ｂ）に示すように、データレジスタに格納された画素データＡ２～Ｇ２は２分割されてその上半分のブロックに属する画素データＡ２ａ～Ｇ２ａが要素プロセッサ群（ＰＥ群）に格納され、残りの画素データＡ２ｂ～Ｇ２ｂと画素データＢ１～Ｇ１がデータレジスタ群（ＤＬ群）に格納される。線形プロセッサアレイＬＡ１６においては、シフトインされた画素データ（斜線で示す）が格納される。

【０１３７】上述のように、プロセッサアレイ内において実質的に線形アレイを構成するように要素プロセッサおよびデータレジスタを配置すれば、単にサーチウィンドウ画素データを連続してシフトインすることにより、図１８に示すように、１列図の右方向にずれたサーチウィンドウ４０ａの画素データをプロセッサアレイ内に格納することができるとともに、要素プロセッサ群にサーチウィンドウブロック４２ｄの画素データを格納することができる。この状態では、テンプレートブロック４３に対するブロック変位ベクトル（１，－１６）に対応する（奇数サブテンプレートブロックについては、奇数フィールドに対する変位ベクトル（１，－８）に対応し、偶数サブテンプレートブロックに対しては偶数フィールドに対する変位ベクトル（１，－８）に対応する）サーチウィンドウブロック４２ｄ～が要素プロセッサに格納されている。

【０１３８】この状態において、再び上述の動作、すなわち差分絶対値の算出および総和の算出を実行し、各予測モードに対する評価関数値を導出する。

【０１３９】上述の動作を（１６＋１６＋１６）×１５＋（１６＋１６）＝７５２回実行すると、図１９に示すように、テンプレートブロック４３に対する変位ベクトル（１５，１６）に対応するサーチウィンドウブロック４２ｅを用いた評価関数値の算出が行なわれる。このサーチウィンドウブロック４２ｅは、偶数サブテンプレートブロックに対しては、偶数フィールドに対する変位ベクトル（１５，＋８）に対応し、奇数サブテンプレートブロックに対しては奇数フィールドに対する変位ベクトル（１５，＋８）に対応する。すなわちテンプレートブロック４３に対し、サーチエリア４５における右下隅のサーチウィンドウブロック４２ｅに対する評価関数値の算出が行なわれ、このテンプレートブロック４３に対するすべての評価関数値の算出が完了する。

【０１４０】このサーチウィンドウブロック４２ｅに対する評価関数値の算出の実行後、算出されたすべての評価関数値に対し比較部（図１参照）において３つの予測モードそれぞれに対する最小の評価関数値が求められ、それらに対応する変位ベクトルがこのテンプレートブロック４３に対する動きベクトル、奇数サブテンプレートブロックに対する動きベクトル、および偶数サブテンプレートブロックに対する動きベクトルとして決定される。

【０１４１】［各回路の具体的構成］図２０は、要素プロセッサの具体的構成を示す図である。図２０において、要素プロセッサＰＥは、テンプレートブロック画素データ格納のためのデータレジスタ２５と、サーチウィンドウ画素データを格納するためのデータレジスタ２６と、データレジスタ２５およびデータレジスタ２６に格納された画素データａおよびｂの差分絶対値を求める差分絶対値回路２７を含む。データレジスタ２５は、クロック信号φ２に応答してその格納データを一方方向に沿って伝達し、データレジスタ２６は、クロック信号φ１に従って隣接要素プロセッサからまたはデータバッファからの画素データを転送する。データレジスタ２５および２６の構成としては、クロック信号に応答してデータを転送しかつラッチするシフトレジスタを用いることができる。またこれに代えてデータレジスタ２５および２６としては、書込経路と読出経路が異なるメモリを用いることができる。差分絶対値回路２７の構成としては、画素データａおよびｂの表現形式（２の補数表示、絶対値表示等）に従って適当な構成が用いられる。画素データａおよびｂの差を求め、その差が負のとき差分値の符

号が反転される構成であれば任意の形式の回路構成を利用することができる。

【０１４２】図２１は、図２０に示す要素プロセッサの動作を示す波形図である。図２１に示すように、テンプレートブロックの動きベクトル検出サイクルにおいては、クロック信号φ２は発生されず非活性状態（図２１においては“Ｌ”レベル）に固定される。クロック信号φ１はサーチウィンドウ画素データＰＹの転送周期を決定する。このクロック信号φ１に従ってサーチウィンドウ画素データＰＹがデータレジスタ２６に格納されかつ転送される。データレジスタ２５におけるテンプレートブロック画素データＰＸは何ら転送されず、データレジスタ２５は、この要素プロセッサＰＥに対応するテンプレートブロック画素データａを格納する。クロック信号φ１の１サイクルごとにデータレジスタ２６に格納される画素データｂが順次ｂ１、ｂ２、…と変化し、この各サイクルごとに差分絶対値回路２７から差分絶対値｜ａ－ｂ１｜、｜ａ－ｂ２｜、…が出力される。この構成により、サーチウィンドウ転送周期内において要素プロセッサそれぞれにおいて評価関数値の成分が算出される。

【０１４３】なお、図２１に示す構成においては、テンプレートブロック画素データおよびサーチウィンドウ画素データ格納のためにそれぞれ１つのデータレジスタが設けられている。複数のデータレジスタが設けられており、１つの要素プロセッサＰＥが、複数のテンプレートブロック画素データに対応して配置される構成が利用されてもよい。

【０１４４】図２２は、図１に示す総和部１２の構成を示す図である。図２２において、総和部１２は、プロセッサアレイに含まれる要素プロセッサＰＥのすべての出力を受けてその総和を求める総和回路１２ａと、プロセッサアレイに含まれる要素プロセッサＰＥのうち、奇数フィールドの画素に対応して配置された要素プロセッサＰＥｏ（奇数サブテンプレートブロックを構成する画素に対応する要素プロセッサ）の出力を受けてその総和を求める総和回路１２ｂと、プロセッサアレイに含まれる要素プロセッサＰＥのうち、偶数フィールドの画素に対応して配置される要素プロセッサＰＥｅ（偶数サブテンプレートブロックを構成する画素に対応して配置される要素プロセッサ）の出力を受けて総和を求める総和回路１２ｃを含む。総和回路１２ａ、１２ｂおよび１２ｃは並列に動作し、それぞれ評価関数値Σ｜ａ－ｂ｜、Σ_{o}｜ａ－ｂ｜、およびΣ_{e}｜ａ－ｂ｜を出力する。総和回路１２ａ～１２ｃを各予測モードに対応して設けて並列に動作させることにより、各予測モードに対応する評価関数値を並列に出力することができる。要素プロセッサＰＥの偶数フィールド要素プロセッサＰＥｅおよび奇数フィールド要素プロセッサＰＥｏへの分類は、単に配線接続により実現される。

【０１４５】図２３は、総和部１２の他の構成を示す図である。図２３に示す総和部１２では、すべての要素プロセッサＰＥの出力の総和を求める総和回路１２ａに代えて、奇数サブテンプレートブロックの評価関数値を導出する総和回路１２ｂの出力と偶数サブテンプレートブロックの評価関数値を導出する総和回路１２ｃの出力を加算する加算回路１２ｄが設けられる。加算回路１２ｄからテンプレートブロックの評価関数値Σ｜ａ－ｂ｜が出力される。奇数サブテンプレートブロックと偶数サブテンプレートブロックを組合わせればテンプレートブロックが得られる。したがって奇数サブテンプレートブロック評価関数値Σ_{o}｜ａ－ｂ｜および偶数サブテンプレートブロック評価関数値Σ_{e}｜ａ－ｂ｜を加算することによりテンプレートブロックに対する評価関数値Σ｜ａ－ｂ｜が得られる。加算回路１２ｄは、２入力加算回路である。したがって、プロセッサアレイ内の要素プロセッサの出力のすべてを加算して総和を求める総和回路１２ａ（図２２参照）よりも大幅にその装置規模が小さくなる。これにより、総和部１２の装置規模を増大させることなく複数の予測モード（テンプレートブロック、奇数サブテンプレートブロック、および偶数サブテンプレートブロック）の評価関数値を高速かつ同時に算出することができる。

【０１４６】［変更例］図２４は、図１に示す総和部の変更例を示す図である。図２４において、総和部１２は、奇数サブテンプレートブロックの画素に対応する要素プロセッサＰＥｏの出力を受け、クロック信号φ３に応答して加算および総和を行なう総和回路１２ｂａと、奇数サブテンプレートブロックの画素に対応して配置される要素プロセッサＰＥｅの出力を受け、クロック信号φ３に応答して総和演算を行なう総和回路１２ｃａと、クロック信号／φ３に従って総和回路１２ａの出力と奇数サブテンプレートブロックの画素に対応して配置される要素プロセッサＰＥｏの出力とを受けて総和を行なう総和回路１２ｅと、クロック信号／φ３に従って偶数サブテンプレートブロックの要素プロセッサＰＥｅの出力と総和回路１２ｃａの出力とを受けて総和演算を行なう総和回路１２ｆと、総和回路１２ｅおよび１２ｆの出力の加算を行なう加算回路１２ｄを含む。総和回路１２ｂａから評価関数値Σ_{oo}｜ａ－ｂ｜が出力される。この評価関数値は偶数サブテンプレートブロックとサーチウィンドウブロックに含まれる奇数フィールドの画素の差分絶対値の総和である。総和回路１２ｃａは偶数サブテンプレートブロックとサーチウィンドウブロックに含まれる偶数フィールドの画素との差分絶対値和Σ_{ee}｜ａ－ｂ｜を評価関数値として出力する。加算回路１２ｄからはテンプレートブロックの評価関数値Σ｜ａ－ｂ｜を出力する。次にこの図２４に示す総和部の動作をその動作波形図である図２５を参照して説明する。

【０１４７】クロック信号φ３はサーチウィンドウ画素データ転送サイクルを規定するクロック信号φ１の２倍

の周期を備える。総和回路１２ｂａおよび１２ｃａは、このクロック信号φ３が“Ｈ”のときに活性化されて総和演算を行ない、クロック信号φ３が非活性状態の“Ｌ”のときには、その総和演算結果を保持しかつ出力する。総和回路１２ｅおよび１２ｆは、クロック信号φ３の反転信号（位相が１８０°ずれている）すなわちクロック信号／φ３が“Ｈ”のときに、与えられたデータに対する総和演算を行ない、クロック信号／φ３が“Ｌ”のときには、その総和結果を保持しかつ出力し続ける。次に動作について図２５を参照して説明する。

【０１４８】今プロセッサアレイのすべての要素プロセッサＰＥの出力をＰＥ１とする。このときクロック信号φ３が“Ｈ”となり、総和回路１２ｂａおよび１２ｃａがこの与えられた差分絶対値Ｐ１（ＰＥｏおよびＰＥｅ）を取込んで総和演算を行ない、その総和演算結果Σｏ（１）およびΣｅ（１）をそれぞれ出力する。このとき総和回路１２ｅおよび１２ｆは総和演算を行なわず、先のサイクルにおける総和演算結果を保持し出力している。

【０１４９】クロック信号φ３が“Ｌ”に立下がると、このときにはプロセッサアレイの要素プロセッサから差分絶対値Ｐ２が与えられる。この状態では総和回路１２ｂａおよび１２ｃａは演算を行なわず、先の総和演算結果を保持している。一方、総和回路１２ｅおよび１２ｆが演算を実行し、総和回路１２ｂａおよび１２ｃａの出力とそのとき与えられた要素プロセッサの出力Ｐ２との総和演算を行なう。したがって加算回路１２ｄからは評価関数値Σｏ（１）＋Σｅ（１）＋Ｐ２が出力される。

【０１５０】次いでクロック信号φ１が“Ｈ”に立上がると、要素プロセッサから差分絶対値Ｐ３が出力される。総和回路１２ｂａおよび１２ｃａが総和演算を行ない、それぞれ総和値Σｏ（２）（＝Σｏ（１）＋Ｐ３ｏ）およびΣｅ（２）（＝Σｅ（１）＋Ｐ３ｅ）を出力する。ここでＰ３ｏおよびＰ３ｅは、それぞれ奇数サブテンプレートブロックの画素に対応して設けられた要素プロセッサの出力を示し、Ｐ３ｅは、偶数サブテンプレートブロックの画素に対応して配置される要素プロセッサの出力を示す。次いで要素プロセッサから差分絶対値Ｐ４が与えられると、総和回路１２ｅおよび１２ｆが総和演算を行ない、加算回路１２ｄからはΣｏ（２）＋Σｅ（２）＋Ｐ４が出力される。以降この動作が繰返される。

【０１５１】すなわち、総和回路１２ｂａおよび１２ｃａは１画素ずつ間引いて総和演算を行なっており、総和回路１２ｅおよび１２ｆは、この間引かれた画素データを補って総和演算を行なっている。したがって、図２６に示すように、サブテンプレートブロック（テンプレートブロック４３における斜線領域で構成されるブロック）は、サーチウィンドウブロック４２における同じフィールドに含まれる画素データと差分絶対値をとることになる。すなわち、奇数サブテンプレートブロックは、サーチウィンドウブロックにおける奇数フィールドに属する画素のみを用いて差分絶対値和が求められ、偶数サブテンプレートブロックに対してはサーチウィンドウブロック内の偶数フィールドの画素を用いてのみ差分絶対値和演算が行なわれる。このように（ａ）フレーム単位の場合にテンプレートブロックの全画素データおよびサーチエリア内の全画素データを用いることによる動きベクトル検出、（ｂ）奇数サブテンプレートブロックとサーチエリア内の奇数フィールドに属する画素のみ等を用いて動きベクトルを決定するモード、および（ｃ）偶数サブテンプレートブロックとサーチエリア内の偶数フィールドに属する画素のみを用いて動きベクトルを検出するモードを実現することができる。

【０１５２】［比較部］図２７は図１に示す比較部の構成を概略的に示す図である。図２７において、比較部３は、テンプレートブロックについての動きベクトル、すなわちブロック動きベクトルＶｂを求めるための比較回路３ａと、奇数サブテンプレートブロックについての動きベクトルすなわち奇数フィールドブロック動きベクトルＶｏを求めるための比較回路３ｂと、偶数サブテンプレートブロックについての動きベクトル、すなわち偶数フィールドブロック動きベクトルＶｅを求めるための比較回路３ｃを含む。比較回路３ａ～３ｃは同一構成を備え、図２７においてはテンプレートブロックについての動きベクトルすなわちブロック動きベクトルＶｂを求めるための比較回路の構成のみを具体的に示す。

【０１５３】比較回路３ａは、総和部から与えられるテンプレートブロックについての評価関数値Σ｜ａ－ｂ｜とレジスタラッチ１３０に格納された評価関数値とを比較する比較器１３２と、比較器１３２の出力に従ってカウンタ１３８から与えられるカウント値を格納するレジスタラッチ１３６を含む。比較器１３２は、レジスタラッチ１３０に格納されている評価関数値よりも新たに総和部から与えられた評価関数値Σ｜ａ－ｂ｜が小さい場合にレジスタラッチ１３０および１３６へデータ更新指示信号を発生する。カウンタ１３８はクロック信号φ１をカウントしており、変位ベクトルを表わすカウント値を発生する。したがって新たに総和部から与えられた評価関数値Σ｜ａ－ｂ｜が先にレジスタラッチ１３０に格納されている評価関数値よりも小さい場合にはレジスタラッチ１３０の記憶内容が新たに与えられた評価関数値で更新され、またレジスタラッチ１３６はそのときにカウンタ１３８から与えられているカウント値を格納する。カウンタ１３８のカウント値は比較回路３ｂおよび３ｃへ共通に与えられる。この構成により、サーチエリア内において最小の評価関数値を与える変位ベクトルを検出するとともに最小評価関数値を保持することができる。このレジスタラッチ１３６に格納されている変位ベクトル値はデコーダ１３７によりデコードされ、具体的

な動きベクトルＶｂを表わす値に変換されてもよい。レジスタラッチ１３６の保持データが直接動きベクトルとして出力されてもよい。またレジスタラッチ１３０に格納されているデータをこの動きベクトルに対する評価関数値Ｅｂとして出力する構成が利用されてもよい。

【０１５４】比較回路３ｂおよび３ｃは、それぞれ総和部から与えられる評価関数値の値が異なっているだけであり、その動作は比較回路３ａのそれと同じである。

【０１５５】［実施例２］図２８は、この発明の第２の実施例である動きベクトル検出装置の全体の構成を概略的に示す図である。この図２８に示す構成においては、総和部１２からはテンプレートブロックについての評価関数値Σ｜ａ－ｂ｜、このテンプレートブロックの上半分のブロックについての評価関数値Σ_n ｜ａ－ｂ｜、およびテンプレートブロックの下半分のブロックについての評価関数値Σ_1 ｜ａ－ｂ｜が出力される。図１に示す構成とは、総和部１２の構成は同じであるが、プロセッサアレイ１０に含まれる要素プロセッサと総和部１２に含まれる総和回路との接続が異なるだけである。

【０１５６】この図２８に示す動きベクトル検出装置の構成では、画面は、図２９に示すように１つのフィールドで構成される。テンプレートブロック４３は垂直方向に２分割され、上半分のブロックの画素で構成される上サブテンプレートブロック４３ｕと、下半分のブロックに含まれる画素で構成される下サブテンプレートブロック４３１に分割される。

【０１５７】同様に、サーチウィンドウブロック４２も、上半分ブロック４２ｕおよび下半分ブロック４２１に分割される。上サブテンプレートブロック４３ｕは、サーチウィンドウブロック４２ｕにおける上半分ブロック４２ｕとの変位ベクトルｆｕに従って評価関数値が算出される。下サブテンプレートブロック４３１は、サーチウィンドウブロック４２の下半分ブロック４２１に対しての変位ベクトルｆ１に対する評価関数値が算出される。またテンプレートブロック４３とサーチウィンドウブロック４２についての変位ベクトルｆに対する評価関数値が算出される。

【０１５８】プロセッサアレイにおいては要素プロセッサがそれぞれテンプレートブロック４３内の対応の画素データとサーチウィンドウブロック４２内の対応の画素データとの差分絶対値を求める。したがって、１つの演算サイクルにおいては、上サブテンプレートブロック４３ｕ、下サブテンプレートブロック４３１およびテンプレートブロック４３の変位ベクトルは同じとなる。

【０１５９】この要素プロセッサの出力を上サブテンプレートブロック４３ｕおよび下サブテンプレートブロック４３１に分類することにより、１つのサイクルで、テンプレートブロック４３に対する評価関数値、上サブテンプレートブロック４３ｕに対する評価関数値、および下サブテンプレートブロック４３１に対する評価関数値を同時に算出することができる。このプロセッサアレイ１０の構成、画素データの転送順序は、図１に示す第１の実施例と同じである。異なるのは、以下に説明する総和部における要素プロセッサの分類態様である。

【０１６０】図３０は、この第２の実施例における総和部１２の構成を示す図である。図３０において、総和部１２は、プロセッサアレイ１０に含まれる要素プロセッサＰＥのうち、上サブテンプレートブロック４３ｕに対応して配置される要素プロセッサの出力ＰＥｕを加算する総和回路１２ｇと、プロセッサアレイに含まれる要素プロセッサのうち下サブテンプレートブロック４３１に対応して配置される要素プロセッサの出力ＰＥ１の総和を求める総和回路１２ｈと、総和回路１２ｇおよび１２ｈの出力を加算する加算回路１２ｄを含む。総和回路１２ｇから上サブテンプレートブロック４３ｕに対する評価関数値Σ_u ｜ａ－ｂ｜が出力され、総和回路１２ｈから下サブテンプレートブロック４３１に対する評価関数値Σ_1 ｜ａ－ｂ｜が出力される。加算回路１２ｄからテンプレートブロック４３についての評価関数値Σ｜ａ－ｂ｜が出力される。要素プロセッサＰＥの上半分ブロックおよび下半分ブロックへの分類は、配線接続により実現される。

【０１６１】比較部３の構成は第１の実施例のそれと同様であり、評価関数値Σ_u ｜ａ－ｂ｜、Σ_1 ｜ａ－ｂ｜、およびΣ｜ａ－ｂ｜に従って上ブロック動きベクトルＶｕ、下ブロック動きベクトルＶ１、およびブロック動きベクトルＶｂが決定される。

【０１６２】上述の構成により、フィールドを単位として符号化を行なう符号化システムにおいて、テンプレートブロックに対する動きベクトル検出と、上サブテンプレートブロックに対する動きベクトル検出と下サブテンプレートブロックに対する動きベクトル検出を同時に実行することができ、複数の予測モードに従って最適な動きベクトルを検出することができる。

【０１６３】［実施例３］図３１は、この発明の第３の実施例である動きベクトル検出装置における要素プロセッサ群とテンプレートブロックの画素との対応関係を示す図である。図３１においては、１６行×１６列に配置された画素によりテンプレートブロック４３が構成される場合を一例として示す。テンプレートブロック４３は、画面上垂直方向に沿って２分割され、まず上半分のブロックの画素データで構成される上サブテンプレートブロック４３ｕと、下半分のブロックの画素を含む下サブテンプレートブロック４３１に分割される。サブテンプレートブロック４３ｕおよび４３１に対応して、要素プロセッサ群も同様にグループ分割される。

【０１６４】次いで、テンプレートブロック４３が更に偶数行の画素と奇数行の画素に分割される。図３１においては、偶数行は、第０行、第２行、…、第１４行を示し、奇数行の画素は、第１行、第３行、…、第１５行の

画素を示す。これに応じて要素プロセッサも偶数行画素に対応する要素プロセッサと奇数行画素に対応する要素プロセッサに分割される。

【０１６５】したがって、要素プロセッサＰＥは、上サブテンプレートブロック４３ｕに属しかつ偶数行に配置される画素に対応する要素プロセッサ群ＰＥ＃０Ｕ、上サブテンプレートブロック４３ｕに属しかつ奇数行に配列される画素に対応する要素プロセッサ群ＰＥ＃１Ｕ、下サブテンプレートブロック４３１に属しかつ偶数行に配置される画素に対応する要素プロセッサ群ＰＥ＃０Ｌ、および下サブテンプレートブロック４３１に属しかつ奇数行に配列される画素に対応する要素プロセッサ群ＰＥ＃１Ｌの４つのグループに分割される。

【０１６６】画面構成がフレーム単位の場合、要素プロセッサ群ＰＥ＃０ＵおよびＰＥ＃０Ｌにより、偶数フィールドサブテンプレートブロックに対する評価関数値の算出が行なわれ、要素プロセッサ群ＰＥ＃１ＵおよびＰＥ＃１Ｌにより奇数フィールドサブテンプレートブロックに対する評価関数値算出が行なわれる。画面構成がフィールド単位の場合、要素プロセッサ群ＰＥ＃０ＵおよびＰＥ＃１Ｕにより、上サブテンプレートブロック４３ｕに対する評価関数値算出が行なわれ、要素プロセッサ群ＰＥ＃０ＬおよびＰＥ＃１Ｌにより、下サブテンプレートブロック４３１に対する評価関数値算出が実行される。したがって図３１に示すように、要素プロセッサ群を４つの群ＰＥ＃０Ｕ、ＰＥ＃１Ｕ、ＰＥ＃０Ｌ、およびＰＥ＃１Ｌに分類すれば、画面構成に応じてこの要素プロセッサの組合せの形態を変更することにより各画面構成に対して定義された予測モードに従って動きベクトル算出を行なうことができる。

【０１６７】図３２はこの発明の第３の実施例である動きベクトル検出装置の全体の構成を示す図である。図３２において、演算部１に含まれるプロセッサアレイ１０においては、要素プロセッサは４つの群ＰＥ＃０Ｕ、ＰＥ＃１Ｕ、ＰＥ＃０Ｌ、およびＰＥ＃１Ｌに分割されている。したがってプロセッサアレイ１０からは、動作時においては各群に対応して差分絶対値｜ａ－ｂ｜ｕｅ、｜ａ－ｂ｜ｕｏ、｜ａ－ｂ｜１ｅ、および｜ａ－ｂ｜１ｏが出力される。

【０１６８】総和部１２は、モード指定信号Ｓに従って、このプロセッサアレイ１０から与えられた４つの差分絶対値群の組合せを変更し、テンプレートブロック単位およびサブテンプレートブロック単位の差分絶対値にグループ化する分類変更部１２２と、分類変更部１２２から与えられた差分絶対値を各類ごとに総和演算を行なう総和演算部１２０を含む。総和演算部１２０からは、この分類変更部１２２から与えられた差分絶対値の分類に従ってそれぞれ対応の評価関数値が導出され比較部３へ与えられる。比較部３は、この総和演算部１２０から与えられた評価関数値に従ってテンプレートブロックの動きベクトルＶｂおよびサブテンプレートブロックに対応する動きベクトルＶｅおよびＶｏまたはＶｕおよびＶ１を出力する。

【０１６９】分類変更部１２２は、モード指定信号Ｓが、フレーム単位での符号化を指定する場合には、この４つの群の差分絶対値からテンプレートブロックに対する差分絶対値群｜ａ－ｂ｜、奇数サブテンプレートブロックに対する差分絶対値｜ａ－ｂ｜ｏ、および偶数フィールドに対する差分絶対値群｜ａ－ｂ｜ｅを生成する。モード指定信号Ｓがフィールド単位での符号化を示すとき、分類変更部Ｓは、テンプレートブロック単位での差分絶対値群｜ａ－ｂ｜、上半分のブロックを用いる上サブテンプレートブロックからの差分絶対値群｜ａ－ｂ｜ｕ、および下半分の画素群からなる下サブテンプレートブロックからの差分絶対値群｜ａ－ｂ｜１を生成する。

【０１７０】図３２に示すように、分類変更部１２２においてモード指定信号Ｓによりこのプロセッサアレイ１０から出力される４つの差分絶対値群の組合せを変更することにより、フレーム単位での予測画像検出およびフィールド単位での予測画像検出それぞれに用いられる動きベクトルを容易に作成することができる。１つの動きベクトル検出装置で異なる画面形式の符号化に対応することができ、汎用性の高い動きベクトル検出装置を実現することができる。

【０１７１】［変更例］図３３は、図３２に示す総和部の変更例を示す図である。図３３において、総和部１２は、プロセッサアレイの要素プロセッサの各群ＰＥ＃１Ｕ、ＰＥ＃１Ｌ、ＰＥ＃０Ｕ、およびＰＥ＃０Ｌに対応して設けられ、各対応の群の出力｜ａ－ｂ｜ｕ０、｜ａ－ｂ｜１０、｜ａ－ｂ｜ｕｅ、および｜ａ－ｂ｜１ｅの総和を求める総和回路１２０ａ、１２０ｂ、１２０ｃ、および１２０ｄと、４つの総和回路のうちの２つの総和回路、すなわち総和回路１２０ｂおよび１２０ｃの出力の伝達経路を切換える分類変更部１２２を含む。分類変更部１２２は、総和回路１２０ｂおよび１２０ｃの伝達経路を切換えるセレクタ１２２ａ、および１２２ｂを含む。セレクタ１２２ａは、モード指定信号Ｓがフレーム単位での符号化を示すとき、総和回路１２０ｂの出力を選択し、セレクタ１２２ｂは、モード指定信号Ｓがフレーム単位での符号化を示すとき総和回路１２０ｃの出力を選択する。

【０１７２】総和部１２は、さらに、総和回路１２０ａの出力とセレクタ１２２ａの出力とを加算する加算回路１２０ｅと、総和回路１２０ｄの出力とセレクタ１２２ｂの出力とを加算する加算回路１２０ｆと、加算回路１２０ｅおよび１２０ｆの出力を加算する加算回路１２０ｇを含む。加算回路１２０ｅから奇数サブテンプレートブロックの評価関数値Σｏ｜ａ－ｂ｜または上サブテンプレートブロックに対する評価関数値Σｕ｜ａ－ｂ｜が出力される。加算回路１２０ｇからはテンプレートブロ

ックに対する評価関数値Σ｜ａ－ｂ｜が出力される。加算回路１２０ｆからは、奇数サブテンプレートブロックについての評価関数値Σｅ｜ａ－ｂ｜または下サブテンプレートブロックの評価関数値Σｌ｜ａ－ｂ｜が出力される。

【０１７３】総和部１２において入力部に要素プロセッサの各群に対応して総和回路１２０ａ～１２０ｄを設けかつ２つの総和回路の出力の伝達経路をモード指定信号により切換える構成は以下の利点を与える。単純に図３２に示す総和部の構成を実現して、要素プロセッサの出力をモード指定信号に応じて伝達経路を切換える場合、この伝達経路を切換えるためのセレクタを各要素プロセッサに対して設ける必要が生じる。この結果、セレクタの数が多くなり、装置規模が増大する。一方、総和回路１２０ａ～１２０ｄの各々は、その担当する要素プロセッサの数は図２３に示す総和回路１２ｂおよび１２ｃのそれの半分であり、装置規模を小さくすることができる。したがって、実質的にこの図３３に示す構成において総和回路の数が図２３に示す構成に比べて増大しているものの占有面積はそれほど大きくはならない。総和回路１２０ｂおよび１２０ｃの伝達経路を切換えるためにセレクタを設ければ、必要とされるセレクタの数は２つであり、大幅に素子数を低減することができ、装置規模を小さくすることができる。加算回路１２０ｅ、１２０ｆ、および１２０ｇは2入力加算回路であり、その規模は総和回路に比べて小さく、この加算回路１２０ｅおよび１２０ｆが設けられてもそれほど装置占有面積は増大しない。したがって図３３に示すように、入力部に要素プロセッサの各群に対応して総和回路を設け、４つの総和回路のうち２つの総和回路の出力の伝達経路を切換える構成とすることにより、大幅に装置占有面積を低減することができる。

【０１７４】なお、図３３に示す構成に代えて、要素プロセッサ群ＰＥ＃１ＬおよびＰＥ＃０Ｕの出力を受けるようにセレクタ１２２ａおよび１２２ｂを設け、一方のセレクタの出力と要素プロセッサ群ＰＥ＃１Ｕの出力の総和をとる総和回路と、他方のセレクタの出力と残りの要素プロセッサ群ＰＥ＃０Ｌの出力との総和をとる総和回路が設けられる構成が利用されてもよい。この構成はほぼ実質的に図２３に示す構成に対応する。

【０１７５】［実施例４］上で説明したように、プロセッサアレイにおいてサーチウィンドウデータを（１６＋１６＋１６）・１５＋（１６＋１６）＝７５２回シフトさせると（シフトインおよびシフトアウトを繰返す）、図３４（Ａ）に示すように、１つのテンプレートブロック（ブロックＮ）に対するサーチエリア（探索範囲）についてすべての変位ベクトル（評価関数値）が求められ、このテンプレートブロックＴＢ（ブロックＮ）に対する動きベクトルを決定することができる。このとき、図３４（Ａ）に示すように、領域５０３ａがプロセッサアレイに存在しており、それに隣接するサーチエリア５０２のサーチウィンドウ画素データもプロセッサアレイ内のプロセッサおよびデータバッファに格納されている。したがってこの領域５０３ａに含まれる１６画素をシフトアウトさせ、同時に領域５０２の続く１６画素の参照画素（サーチウィンドウデータ）をシフトインさせると、プロセッサアレイ内においては、図３４（Ｂ）に示すように、このテンプレートブロックＴＢに対する変位ベクトル（１６，－１６）のサーチウィンドウブロック５０４の画素データが格納される。したがって、７５２＋１６＝７６８サイクルを１処理単位とすれば、水平方向１６変位×垂直方向３３変位の範囲で動きベクトルを探索することができるため、この処理単位を複数回繰返すことにより、水平方向探索範囲を拡張することができ、水平方向におけるサーチエリアを広くして動きベクトルを探索することができる。

【０１７６】一般に、Ｑ行Ｐ列のサーチウィンドウ画素にそれぞれ１対１対応でプロセッサが格納されており、データバッファがＲ個の画素データを格納する場合、Ｐ・（Ｑ＋Ｒ）サイクルでサーチエリアの１単位での動きベクトル探索が実行される。このとき、探索される変位ベクトルは、サーチウィンドウの１列の画素データのシフトアウト当りの変位ベクトルが（Ｒ＋１）であり、したがってＰ・（Ｒ＋１）の変位ベクトルについて評価を行なうことができる。この場合、水平方向の探索範囲はＰとなる。さらに、図３５に示すように、サーチウィンドウブロック５０６のサイズをＰ列Ｑ行とし、サーチウィンドウ５０５ａのサイズを（Ｑ＋Ｒ）・Ｐとし、サーチエリアを領域５０５ａおよび５０５ｂとした場合、このサーチエリア５０５ａおよび５０５ｂにおける変位ベクトルをすべて評価するためには、サーチウィンドウ画素データを（Ｐ－１）・（Ｑ＋Ｒ）＋Ｒ回、参照画像画素データ（サーチウィンドウ画素データ）をシフトさせる必要がある。この後、さらにＲ回参照画像画素データをシフトインすれば、プロセッサアレイには領域５０５ｃに含まれる参照画像画素データが格納される。したがって、領域５０５ｃのサーチウィンドウの画素データをプロセッサアレイに格納するためには合計（Ｐ－１）・（Ｑ＋Ｒ）＋Ｒ＝Ｐ・（Ｑ＋Ｒ）サイクル必要とされる。サーチエリアの各列において（Ｒ＋１）の変位ベクトルが評価されるため、評価される変位ベクトルの数はＰ・（Ｒ＋１）となる。

【０１７７】図３６に水平方向の探索範囲の具体的例を示す。図３６において、ケース（Ｉ）は、水平方向探索範囲は－８～７である。ケース（ＩＩ）の場合、水平方向探索範囲は－１６～＋１５である。この場合、１つの処理単位サイクルにおいて水平方向－１６～－１の範囲における動きベクトル探索が行なわれ、もう１つの処理単位サイクルにおいて水平方向０～＋１５の範囲について動きベクトル探索が行なわれる。ケース（ＩＩＩ）に

おいては、水平方向探索範囲は－３２～＋３１とされる。この場合においては、４つの処理単位サイクルが必要とされ、各処理単位サイクルにおいて－３２～－１７、－１６～－１、０～＋１５、および１６～＋３１の水平方向範囲についてそれぞれ動きベクトル探索が行なわれる。

【０１７８】図３７は、様々な水平探索範囲における動きベクトル探索動作シーケンスを示す図である。図３７においては、水平探索範囲として、－８～＋７、－１６～＋１５、および－３２～＋３１の一例として示される。図３７（Ａ）に示すように、水平探索範囲が－８～＋８の場合、処理単位時間（７６８サイクル）Ｔごとにテンプレートブロックが更新される。すなわち各処理単位時間Ｔごとにテンプレートブロックの動きベクトルが決定される。図３７（Ａ）においては、テンプレートブロックＮ、Ｎ＋１、…について動きベクトルが順次決定される状態が示される。水平探索範囲が－１６～＋１５の場合、図３７（Ｂ）に示すように、２処理単位時間、２Ｔが必要とされる。２Ｔのうち、最初の処理単位時間において水平方向－１６～－１の範囲において動きベクトル探索が行なわれ、残りのＴにおいて水平方向０～＋１５において動きベクトル探索が行なわれる。２Ｔごとに１つのテンプレートブロックに対する動きベクトルが決定される。

【０１７９】図３７（Ｃ）に示すように、水平探索範囲が－３２～＋３１の場合、１つの処理単位時間Ｔにおいて１６の水平範囲について動きベクトルが検出されるため、１つのテンプレートブロックについて４Ｔの処理時間が必要とされる。上述のように、垂直探索範囲が－１６～＋１６の場合、処理単位時間を７６８サイクルとすれば、水平１６変位・垂直３３変位の範囲での動きベクトルが探索可能であり、この処理単位時間を把握することにより、水平探索範囲を容易に拡張することができる。なお、図３７においては、テンプレートブロックの更新サイクル数は示していない。単に１つのテンプレートブロックについての動きベクトル決定に必要とされる処理時間のみを考慮している。

【０１８０】図３８は、この第４の実施例に用いられる動きベクトル検出装置の全体の構成を示す図である。動きベクトル検出装置６００は、先の実施例において説明したように、サーチウィンドウ画素データおよびテンプレートブロック画素データを入力する入力部２と、入力部２から与えられたサーチウィンドウ画素データおよびテンプレートブロック画素データに所定の演算処理を施して変位ベクトルに対する評価関数値を生成する演算部１と、演算部１からの評価関数値に従ってこのテンプレートブロックに対する動きベクトルを決定するための比較部３を含む。図３８においては、３つの予測モードに従って評価関数値および動きベクトルが各々生成されるが、１つの動きベクトルのみが生成されるように示される。この動きベクトル検出装置６００は、さらに、参照画面画像画素データを格納する参照画像メモリ６０２と、現画面画素データを格納する現画像メモリ６０４と、参照画像メモリ６０２および現画像メモリ６０４の画素データ読出動作を制御する読出制御回路６０６と、比較部３の比較動作すなわち動きベクトル検出動作を制御する比較制御回路６０８と、この動きベクトルの水平方向の探索範囲を設定して設定された水平方向探索範囲情報を読出制御回路６０６および６０８へ与える探索範囲設定部６０７を含む。

【０１８１】比較制御回路６０８は、この探索範囲設定部６０７からの水平方向探索範囲設定情報に従って比較部３における動きベクトル検出動作サイクルを設定する。読出制御回路６０６は、またこの探索範囲設定部６０７からの水平方向探索範囲設定情報に従って参照画像メモリ６０２からの画素データの読出領域の設定および現画像メモリ６０４からの現画像画素データ読出タイミングを調整する。図３９は、図３８に示す比較部３の構成の一例を示す図である。図３９において、比較部３は、動きベクトル候補の評価関数値を格納するレジスタ６１０と、レジスタ６１０に格納された候補評価関数値と演算部１から与えられる評価関数値とを比較する比較器６１１と、評価関数値それぞれの変位ベクトルを示す値を発生するカウンタ６１３と、比較器６１１の出力信号に従ってカウンタ６１３のカウント値を動きベクトル候補として格納するレジスタ６１２を含む。比較器６１１が、演算部１から与えられた評価関数値がレジスタ６１０に格納された評価関数値よりも小さいことを示すとき、レジスタ６１０および６１２の内容が更新される。すなわち、レジスタ６１０はそのときに演算部１から与えられた評価関数値を格納し、レジスタ６１２はカウンタ６１３から与えられたカウント値を格納する。

【０１８２】カウンタ６１３は、比較制御回路６０８から与えられるクロック信号φをカウントし、変位ベクトル情報を発生し、処理サイクル終了時に発生されるリセット信号Ｒｅに従ってそのカウント値がリセットされる。比較制御回路は、この探索範囲設定部６０７からの水平方向探索範囲設定情報に従ってリセット信号Ｒｅの発生タイミングを変更する。図４０は、この比較制御回路６０８の動作を例示する図である。図４０に示すように、比較制御回路６０８からは、３２サイクルの間クロック信号φがカウンタ６１３へ与えられ、カウンタ６１３はカウント動作を行なう。次の１６サイクルにおいては、クロック信号φは発生されず非活性状態とされる。すなわち、サーチウィンドウの最下部にまで相対的に移動したテンプレートブロックを再度上部（垂直方向－１６）の位置へ相対的に移動させるためである。この４８サイクルを１つの小単位として７５２サイクルシフト動作が繰返される。この７５２サイクルにより、水平方向１６変位についての動きベクトルの探索が行なわれる。

次の１６サイクルにおいては再びクロック信号φの発生が禁止される（非活性状態とされる）。この７５２サイクル＋１６サイクルが１処理単位サイクルＴに対応する。比較制御回路６０８は、探索範囲設定部６０７により設定された水平方向探索範囲情報に従って動きベクトル探索範囲を設定する。この設定されたサイクル内においては、１列の動作サイクルすなわち４８サイクルを単位として行なわれる動作が繰返される。

【０１８３】図４１は、カウンタ６１３のカウント値のリセットタイミングを示す図である。図４１に示すように、水平探索範囲が－８～＋７の場合、７６８サイクルごとにリセット信号Ｒｅが発生される（活性状態とされる）。水平探索範囲が－１６～＋１５の場合には、７６８・２サイクルごとにリセット信号Ｒｅが発生され、水平探索範囲が－３２～＋３１の場合には、７６８・４サイクルごとにリセット信号Ｒｅが発生される。このリセット信号Ｒｅの発生タイミングを処理時間に合わせて異ならせることにより、動きベクトルの探索範囲を容易に変更することができる。カウンタ６１３の構成としては、水平方向の位置Ｈを表わすカウンタと垂直方向の位置Ｖを示すカウンタが別々に設けられてもよく、また単に２進カウンタにより、そのカウント値に従ってＨ情報およびＶ情報両者を含むように発生する構成が利用されてもよい。

【０１８４】なお、比較動作禁止時においては、当然、レジスタ６１０および６１２（図３９参照）の更新動作も禁止される。これは、この比較動作禁止期間において比較器６１１の出力を強制的にレジスタ非更新状態に設定する構成が利用されてもよい。このとき、また比較器６１１（図３９参照）の比較動作が禁止されるように構成されてもよい。図４２は、図３８に示す比較制御回路の構成の一例を示す図である。図４２において、比較制御回路６０８は、参照画像メモリからプロセッサアレイ（演算部１に含まれる）への画素データ印加時に発生されるクロック信号ＣＬＫをカウントするカウンタ６１５および６１６と、探索範囲設定部６０７からの水平方向探索範囲設定情報Ａとカウンタ６１５のカウントアップ信号φＣＵＰとを比較する比較器６１７と、カウンタ６１６からの３２カウント指示信号φ３２および４８カウント指示信号φ４８に従ってクロック信号φを発生するクロック発生器６１８を含む。カウンタ６１５は、その１処理単位サイクルに含まれるクロックサイクル数をカウントしたときにカウントアップ信号φＣＵＰを発生する。水平方向探索範囲設定情報Ａは、水平方向の探索範囲が単位探索範囲の何倍であるかを指定する。比較器６１７は、このカウンタ６１５からのカウントアップ信号φＣＵＰと水平方向探索範囲設定情報Ａとを比較し、これらが一致したときにリセット信号Ｒｅを発生する（活性状態とする）。

【０１８５】カウンタ６１６は、クロック信号ＣＬＫを３２カウントしたときにカウントアップ信号φ３２を発生し、また４８カウントしたときにカウントアップ信号φ４８を発生する。カウンタ６１６は、その最大カウント値が４８に設定され、サーチウィンドウの１列の画素データが処理されたときにカウントアップ信号φ４８を発生する。クロック発生器６１８は、カウンタ６１６からカウントアップ信号φ３２が与えられると次にカウントアップ信号φ４８が与えられるまでクロック信号φを非活性状態に設定する。カウントアップ信号φ３２が発生される（活性状態とされる）までは、クロック発生器６１８はクロック信号ＣＬＫに従って（同期して）クロック信号φを発生する。このクロック発生器６１８からのクロック信号φにより比較部３に含まれるカウンタ６１３（図３９参照）のカウント動作が制御される。

【０１８６】図４２に示す比較制御回路６０８の構成を用いることにより、比較部３における動きベクトル探索範囲の水平方向の範囲を容易に制御することができる。図４３は、図３８に示す読出制御回路６０６の構成の一例を示す図である。図４３において、読出制御回路６０６は、参照画像メモリ６０２の読出アドレスを発生するアドレスカウンタ６２０と、水平方向探索範囲設定情報Ａに従ってこのアドレスカウンタ６２０の初期カウント値およびカウント範囲を設定するアドレス制御回路６２１と、現画像メモリ６０４の読出アドレスを発生するアドレスカウンタ６２２と、水平方向探索範囲設定情報Ａに従ってアドレスカウンタ６２２の動作タイミングおよび初期アドレスを設定するアドレス制御回路６２３を含む。アドレス制御回路６２１は、水平方向探索範囲情報Ａに従ってアドレスカウンタ６２０から発生されるアドレスのうち水平方向の画素データに関するアドレスの範囲を設定する。アドレス制御回路６２３は、テンプレートブロックの位置を示すポインタを発生してアドレスカウンタ６２２へ与える。テンプレートブロックの位置を示すポインタ情報に従ってアドレスカウンタ６２２が読出あしアドレスを生成し、テンプレートブロック画素データが現画像メモリ６０４から読出される。アドレス制御回路６２３は、この水平方向探索範囲設定情報Ａに従って、このアドレスカウンタ６２２へのテンプレートブロックの位置を示すポインタの発生タイミングがＡ・Ｔに設定される。アドレスカウンタ６２０は、一方、アドレス制御回路６２１の制御の下に、参照画像メモリ６０２における列アドレス範囲がＡ倍に設定される。これにより、指定された水平方向探索範囲に応じて参照画像画素データおよびテンプレートブロック画素データをそれぞれ所望のタイミングで読出すことができる。

【０１８７】なお、この図４０に示す比較部の動作から明らかなように、水平方向探索範囲は、処理単位時間Ｔの整数倍で設定することもできるが、これはまた水平方向探索範囲を１列ずつ増加させることも可能である。各列における動作が３２サイクルの動きベクトル評価動作

および１６サイクルの次の列ベクトル評価のための参照画像画素データ入力動作が行なわれ、この１列の動作を単位として動きベクトル検出動作が行なわれているため、水平方向探索範囲を１列ずつ増減することもまた可能である。この場合には処理単位を１列とし、水平方向探索範囲情報Ａを水平方向探索範囲を示す値とすることにより、上述の構成をそのまま利用することができる。以上のように、この発明の第４の実施例に従えば、動きベクトルの水平方向探索範囲を変更することができ、処理用途に応じて最適な水平方向探索範囲を設定することができ、汎用性の高い画像符号化システムを容易に構築することができる。

【０１８８】［実施例５］図４４は、この発明の第５の実施例である動きベクトル検出装置の全体の構成を概略的に示す図である。図４４において、動きベクトル検出装置は、並列に設けられるＡ側演算部１ａおよびＢ側演算部１ｂを含む。Ａ側演算部１ａおよびＢ側演算部１ｂは、ともに先の実施例において説明したプロセッサアレイ１０および総和部１２を含む（図１参照）。動きベクトル検出装置は、さらに、Ａ側演算部１ａが３つの予測モードそれぞれに従って生成する評価関数値ｅｖａ、ｅｖｂおよびｅｖｃを受け、それぞれについて最適動きベクトルを決定する比較部３ａと、Ｂ側演算部１ｂが複数の予測モードに従ってそれぞれ生成した評価関数値ｅｖａ、ｅｖｂおよびｅｖｃをそれぞれ受けてそれぞれについて最適動きベクトルを決定する比較部３ｂと、比較部３ａおよび３ｂの出力する動きベクトルを交互に選択してＡ側演算部１ａおよびＢ側演算部１ｂそれぞれの動きベクトルＭＶＡ、ＭＶＢおよびＭＶＣを出力するセレクタ６３２と、外部から与えられるテンプレートブロック画素データＰＸを処理単位サイクルごとに交互にＡ側演算部１ａおよびＢ側演算部１ｂへ与えるとともに、外部から与えられるサーチウィンドウ画素データＰＹを共通にＡ側演算部１ａおよびＢ側演算部１ｂへ与える入力部６３０を含む。Ａ側演算部１ａおよびＢ側演算部１ｂは、その水平方向探索範囲が－１６～＋１５に設定される。したがって、Ａ側演算部１ａおよびＢ側演算部１ｂは、それぞれ１つのテンプレートブロックの処理に２処理単位時間、２Ｔ必要とする。したがって、入力部６３０は、この１処理単位時間ごとに交互にテンプレートブロック画素データを更新するとともにセレクタ６３２は、この１処理単位時間Ｔごとに比較部３ａおよび３ｂの出力する動きベクトルを切換えて出力する。

【０１８９】図４５に示すように、サーチウィンドウブロック６３５は、テンプレートブロックＮに関しては変位ベクトル（０，－１６）を有し、テンプレートブロックＮ＋１については変位ベクトル（－１６，－１６）を有する。テンプレートブロックＮをたとえばＡ側演算部１ａのプロセッサアレイに格納し、テンプレートブロックＮ＋１の画素データをＢ側演算部１ｂに格納する。このＡ側演算部１ａおよびＢ側演算部１ｂに共通にサーチウィンドウ画素データＰＹを印加すれば、並列に２つのテンプレートブロックについて動きベクトル探索動作を行なうことができる。異なるのは、Ａ側演算部１ａとＢ側演算部１ｂにおける動きベクトル検出動作開始タイミングである。図４６にＡ側演算部１ａおよびＢ側演算部１ｂの処理シーケンスを示す。図４６に示すように、Ａ側演算部１ａにおいてはテンプレートブロックＮの水平方向－１６～－１についての動きベクトル探索動作が行なわれているとき、Ｂ側演算部１ｂにおいては、テンプレートブロックＮ－１について水平方向０～＋１５の範囲において動きベクトル探索動作が行なわれる。テンプレートブロックＮ－１の水平方向０～＋１５の動きベクトル探索動作が完了すると、このテンプレートブロックＮ－１についての動きベクトルが決定される。一方、Ａ側演算部１ａにおいては、次の処理単位時間Ｔにおいてブロックについて水平方向０～＋１５において動きベクトル検出動作が行なわれる。Ｂ側演算部１ｂにおいては、入力部６３０を介してテンプレートブロックＮ＋１の画素データが格納され、このテンプレートブロックＮ＋１の水平方向－１６～－１についての動きベクトルの検出動作が行なわれる。以降、処理単位時間ＴごとにＡ側演算部１ａおよびＢ側演算部１ｂのテンプレートブロック画素データが交互に更新される。比較部３ａおよび３ｂにおいては、したがって処理単位時間Ｔごとに各予測モードに応じた動きベクトルを交互に決定して出力する。セレクタ６３２はこの処理単位時間ごとに選択態様を切換えて比較部３ａおよび３ｂから出力された動きベクトルを選択して出力する。

【０１９０】上述のように、２つの演算部および２つの比較部を設け、異なったテンプレートブロックの画素データを各演算部へ格納するとともに、同じサーチウィンドウ画素データを共通に演算部へ与えることにより、パイプライン的にテンプレートブロックの動きベクトル検出動作を行なうことができ、高速で動きベクトル検出を行なうことができる。すなわち、動きベクトル検出に実際には２Ｔ必要とされても、見かけ上動きベクトルはＴごとに決定されるためである。図４７は、演算部および比較部の組が４組並列に設けられた場合の動作を示す図である。この図４７においては、水平方向探索範囲が－３２～＋３１に設定され、１つのテンプレートブロックについての動きベクトル検出には４Ｔの処理時間が必要とされる。各演算部には異なるテンプレートブロックの画素データが格納され、それぞれの演算部には共通に同じサーチウィンドウ画素データが与えられる。この場合、４つのテンプレートブロック画素データの処理がパイプライン的に行なわれ、処理時間Ｔごとに演算部から順次動きベクトルが決定されて出力される。したがってこの場合においても、処理時間が４Ｔとなった場合においても動きベクトルを単位時間Ｔごとに出力することが

できる。

【０１９１】すなわち、処理単位時間をＴとしてその水平方向探索範囲がＡ倍された場合、演算部および比較部をＡ組設け、各演算部に異なるテンプレートブロック画素データを格納することにより、Ａ個のテンプレートブロックについての動きベクトル検出動作をパイプライン的に行なうことができる。なお、このパイプライン態様の動作について、テンプレートブロック画素データの更新時間は無視している。これは各処理単位時間ごとにテンプレートブロック画素データ更新時間の待合せを行ない、演算部においてテンプレートブロック画素データが格納された後にすべての演算部において動きベクトル検出動作が同じタイミングで開始／再開されるように構成されれば、容易にこの図４６および図４７に示す動きベクトル検出動作シーケンスを実現することができる。

【０１９２】図４４に示す入力部６３０およびセレクタ６３２の構成としては、単に処理対象となるテンプレートブロックの数に応じてそのテンプレートブロック画素データの伝達経路が順次書換えられる構成が入力部６３０において設けられ、セレクタ６３２においては、その処理単位時間Ｔごとに比較部の出力を順次選択する構成が用いられればよく、この構成は容易に実現することができる。またこの場合、テンプレートブロック画素データを現画像メモリから読出す構成は何ら処理単位時間Ｔでテンプレートブロック画素データを更新する場合と変更はなく、単に参照画像メモリからサーチウィンドウ画素データを出力する場合、その処理時間に応じてサーチウィンドウ画素データを読出す範囲が変更されるだけであり、これは第４の実施例の構成を利用することができる。

【０１９３】以上のように、この発明の第５の実施例の構成に従えば、複数の組の演算部および比較部を並列に設け、これらの演算部および比較部の各組へ異なるテンプレートブロック画素データを与えるとともに、共通にサーチウィンドウ画素データを与え、各組で処理単位時間ごとにテンプレートブロックを更新しているため、高速で複数の予測モードに従って動きベクトルを検出することができる。また２組の演算部および比較部の構成が用いられる場合においても、水平方向のベクトル探索範囲を変更することができる（２Ｔ処理時間ごとにテンプレートブロック画素を更新すればよい）ため、動きベクトルの水平方向探索範囲を容易に変更することができる。さらに、演算部および比較部を１つのチップで構成した場合、この１つのチップにおいて処理単位時間Ｔが設定されている場合、Ａ個のチップを並列に設けておけば、容易に水平方向探索範囲が異なる画像符号化システムへ適用することができる。各チップにおいて実施例４の構成を設けておけばよいためである。

【０１９４】［実施例６］図４８は、１６行×１６列に配置された要素プロセッサを含むプロセッサアレイの全体の構成を概略的に示す図である。図４８においては、要素プロセッサＰＥは、１６行×１６列に配置された画素を有するテンプレートブロックの各画素に１対１対応で配置される。すなわち、要素プロセッサＰＥは、サーチウィンドウ画素データを格納する１つのレジスタと、テンプレートブロック画素データを格納する１つのレジスタを含む（図３においてＮ＝Ｍ＝１）。図４８において、テンプレートブロックの各列に対応して配置される線形プロセッサアレイＬＡ０～ＬＡ１５は、１６個の要素プロセッサＰＥと、３２画素のデータを各々が格納するデータバッファＤＬ（ＤＬ０～ＤＬ１５）を含む。図４８においては、後の説明の便宜上、線形プロセッサアレイＬＡ０～ＬＡ１５の各々は、上流側（図４８の上部）に設けられる８個の要素プロセッサを含む要素プロセッサグループＰＧｉＵと、下流側に設けられた８個の要素プロセッサを含む要素プロセッサグループＰＧｉＬを含む（ただし、ｉ＝０～１５）。線形プロセッサアレイＬＡ０～ＬＡ１５において要素プロセッサは、１次元的にテンプレートブロック画素データを一方方向に伝達するように相互接続され、また線形プロセッサアレイＬＡ０～ＬＡ１５はサーチウィンドウ画素データＰＹを１次元的に一方方向に沿って伝達するように相互接続される。すなわち、要素プロセッサおよびデータバッファはシストリックアレイ状に配置される。

【０１９５】この図４８に示す構成の場合、先に説明した実施例４と同様の動作を行なうことができる。すなわち、１６行・１６列に配置されたテンプレートブロックに対し垂直方向探索範囲を－１６～＋１６として動きベクトル探索を行なうことができる。このとき、水平方向の探索範囲は先の実施例４と同様変更可能である。図４９は、８行８列に配置された画素からなるテンプレートブロックに対応して図４８に示すプロセッサアレイを再配置したときのプロセッサアレイ（演算部）の全体の構成を概略的に示す図である。図４９において、プロセッサアレイ（演算部）は、各々が８行・８列に配置された要素プロセッサを含む４つのプロセッサブロックＰＢＡ～ＰＢＤを含む。プロセッサブロックＰＢＡは、８列のプロセッサグループＰＧ８Ｕ～ＰＧ１５Ｕを含み、プロセッサブロックＰＢＢは、プロセッサグループＰＧ０Ｕ～ＰＧ７Ｕを含む。プロセッサブロックＰＢＣは、プロセッサグループＰＧ８Ｌ～ＰＧ１５Ｌを含み、プロセッサブロックＰＢＤは、プロセッサグループＰＧ０Ｌ～ＰＧ７Ｌを含む。

【０１９６】プロセッサアレイは、２行・２列に配置されたプロセッサブロックを含み、プロセッサブロックの各列に対応してデータバッファＤＬ０～ＤＬ１５が分割される。すなわち、データバッファＤＬ０～ＤＬ７がプロセッサブロックＰＢＢおよびＰＢＤに対応して設けられ、データバッファＤＬ８～ＤＬ１５がプロセッサブロックＰＢＡおよびＰＢＣに対応して配置される。データ

バッファＤＬ０～ＤＬ１５の各々は、３２画素のデータを格納する。同じ列に配置されたプロセッサグループはすべて対応のデータバッファからサーチウィンドウ画素データを入力する。たとえば、プロセッサグループＰＧ０ＵおよびＰＧ０Ｌは、データバッファＤＬ０からサーチウィンドウ画素データを入力する。上流側に設けられたプロセッサブロックＰＢＡおよびＰＢＢのプロセッサグループＰＧ０Ｕ～ＰＧ１５Ｕの各々は、対応のデータバッファＤＬ０～ＤＬ１５から与えられるサーチウィンドウ画素データを、その内部の要素プロセッサを介して順次転送し、各最上流の要素プロセッサからシフトアウトする。すなわち、プロセッサグループＰＧ０Ｕ～ＰＧ１５Ｕの各々は、サーチウィンドウ画素データについては、単に、直列にサーチウィンドウ画素データを転送する動作を行ない、シストリックアレイは構成しない。一方、下流側のプロセッサブロックＰＢＣおよびＰＢＤについては、プロセッサグループＰＧ０Ｌ～ＰＧ７Ｌと対応のデータバッファＤＬ０～ＤＬ７がサーチウィンドウ画素データを１次元的に一方方向にのみ転送するようにシストリックアレイを構成するように相互接続される。すなわち、プロセッサグループの最上流の要素プロセッサからシフトアウトされるサーチウィンドウ画素データは上流側の隣接列のデータバッファへ与えられる。同様、プロセッサブロックＰＢＣにおいても、プロセッサグループＰＧ８Ｌ～ＰＧ１５ＬおよびデータバッファＤＬ８～ＤＬ１５がサーチウィンドウ画素データについては１次元的に一方方向に沿ってのみ転送動作を行なうシストリックアレイを構成する。

【０１９７】テンプレートブロック画素データについては、プロセッサブロックＰＢＡ～ＰＢＤの各々は、１次元方向に沿ってのみ画像データ転送を行なうようにシストリックアレイ状に要素プロセッサが相互接続される。プロセッサブロックＰＢＡ～ＰＢＤの各々へは、異なるテンプレートブロックの画素データが与えられてそこに格納される。図４９においては、異なるテンプレートブロックを示すために、テンプレートブロック画素データＰＸＡ～ＰＸＤがそれぞれプロセッサブロックＰＢＡ～ＰＢＤへ与えられるように示される。次に動作について説明する。テンプレートブロックのサイズが８行・８列であり、またデータバッファＤＬ０～ＤＬ１５それぞれは、３２画素のデータを格納するため、図５０に示すように、処理単位として、水平方向８変位、垂直方向３３変位をとる（Ｐ・（Ｑ＋Ｒ）が処理単位時間である）。

【０１９８】今、図５１に示すように、プロセッサブロックＰＢＡ～ＰＢＤそれぞれにテンプレートブロックＭＡ、ＭＢ、ＭＣおよびＭＤそれぞれの画素データが格納されており、またサーチウィンドウブロック６５０が格納された状態を考える。サーチウィンドウブロック６５０の変位は、テンプレートブロックＭＡに対しては変位（８，－８）であり、テンプレートブロックＭＢに対しては変位（０，－８）であり、テンプレートブロックＭＣに対しては、変位（－８，－８）であり、テンプレートブロックＭＤに対しては、変位（－１６，－８）である。したがって、プロセッサブロックＰＢＡ～ＰＢＤそれぞれが並列にパイプライン的に動きベクトル検出動作を行なうために、４処理単位時間を処理時間とする。すなわち、図５２に示すように水平方向３３変位、および垂直方向３３変位のサーチエリア（探索領域）において動きベクトル検出を行なう。

【０１９９】今、１つのプロセッサブロックにおける１つのサーチウィンドウの動きベクトル検出動作を考える。今、図５３（Ａ）に示すように、サーチウィンドウは４０行・８列に配置された画素を備える。最上部のサーチウィンドウブロックの変位がたとえば（－１６，－１６）であり、３２回シフト動作を行ない、変位（－１６，＋１６）に到達した状態を考える。この場合、プロセッサアレイにおいては、図５３（Ｂ）に斜線で示す８画素のみが動きベクトル検出動作を受けるサーチウィンドウ画素データとして残る。この斜線領域で示す８画素をシフトアウトすれば、次の列における動きベクトルすなわち（－１５，－１６）に対する評価値を算出することができる。４処理単位時間の各処理単位時間において水平方向変位－１６～－９、－８～－１、０～＋７、および＋８～＋１５についてそれぞれ動きベクトル検出動作が行なわれる。この４処理単位時間における各処理単位時間の異なる位相での動きベクトル検出動作をプロセッサブロックＰＢＡ～ＰＢＤそれぞれにおいて実行する。

【０２００】図５４にプロセッサブロックにおける処理動作シーケンスの一例を示す。処理単位時間Ｔは（３２＋８）×８＝３２０サイクルである。期間ｔ１においては、プロセッサブロックＰＢＡにおいてテンプレートブロックＮの水平変位－１６～－９についての動きベクトル検出が行なわれる。プロセッサブロックＰＢＤにおいては、テンプレートブロックＮ－３に対し水平方向変位＋８～＋１５の範囲においての動きベクトル検出が行なわれる。プロセッサブロックＰＢＣにおいては、テンプレートブロックＮ－２に対し水平方向変位０～＋７の範囲においての動きベクトル検出が行なわれる。プロセッサブロックＰＢＤにおいては、テンプレートブロックＮ－１に対して水平方向変位－８～－１における動きベクトル検出が行なわれる。ここで、ブロックＮ－３、Ｎ－２、Ｎ－１、およびＮ、Ｎ＋１、…は順次水平方向に沿って配置される。プロセッサブロックＰＢＡにおいては、４処理単位時間経過後の時間ｔ４においてはブロックＮに対する水平方向変位＋８～＋１５における動きベクトル検出動作が行なわれ、このテンプレートブロックＮに対する動きベクトルが決定される。プロセッサブロックＰＢＢにおいては、時間ｔ１においてテンプレートブロックＮ－３の水平方向範囲＋８～＋１５における動

きベクトル検出動作が行なわれ、その結果、テンプレートブロックＮ－３についての動きベクトルが決定される。次の期間ｔ２から新しいテンプレートブロックＮ＋１についての動きベクトル検出動作が行なわれる。

【０２０１】プロセッサブロックＰＢＣにおいては、期間ｔ２においてテンプレートブロックＮ－２の水平方向変位＋８～＋１５における動きベクトル探索動作が完了し、このテンプレートブロックＮ－２についての動きベクトルが決定される。次の期間ｔ３から新しいテンプレートブロックＮ＋２についての動きベクトル探索が行なわれる。プロセッサブロックＰＢＤについては、期間ｔ３においてテンプレートブロックＮ－１についての動きベクトル探索動作が完了し、このテンプレートブロックＮ－１に対する動きベクトルが決定される。次の期間ｔ４から新しいテンプレートブロックＮ＋３についての動きベクトル検出動作が行なわれる。すなわち、プロセッサブロックＰＢＡ～ＰＢＤにおいて１処理単位時間ずらして新しいテンプレートブロックについての動きベクトル検出動作が開始される。各処理単位時間完了時において１つのテンプレートブロックについての動きベクトルがプロセッサブロックＰＢＡ～ＰＢＤのいずれかの処理結果により決定される。

【０２０２】上述のように、プロセッサアレイを４分割することにより、また処理時間を４処理単位時間とすることにより、４つのテンプレートブロックについてパイプライン的に動きベクトル検出動作を行なうことができる。なお上述の実施例においては、１６行１６列に配置されたテンプレートブロック画素に対応して配置される要素プロセッサを各々が８行・８列の要素プロセッサを有する４つのプロセッサブロックに分割している。一般に、要素プロセッサが１画素に対応する場合、Ｐ列・Ｑ行に配置された要素プロセッサを（Ｐ／ｂ）列・（Ｑ／ｃ）行の要素プロセッサを有するプロセッサブロックに分割し、プロセッサブロックをｂ列、ｃ行に配置した場合、データバッファがＲ個の画素データを格納する場合、処理単位時間として（Ｐ／ｂ）・｛（Ｑ／ｃ）＋Ｒ｝を用いれば、プロセッサブロックの各々は、ｂ・ｃ・（Ｐ／ｂ）・（Ｒ＋１）の変位ベクトルに対して評価を行なって動きベクトルを検出することができる。このときの処理時間はｂ・ｃ・（Ｐ／ｂ）・｛（Ｑ／ｃ）＋Ｒ｝サイクルとなる。

【０２０３】［具体的な構成］図５５は、この発明の第６の実施例に用いられる動きベクトル検出装置の要部の構成を示す図である。図５５においては、動きベクトル検出装置における演算部６４２の構成を示す。図５５において、演算部６４２は４つのプロセッサブロックＰＢＡ～ＰＢＤと、データバッファ群ＤＬＡおよびＤＬＢを含む。プロセッサブロックＰＢＡ～ＰＢＤはそれぞれ図４９に示すものと同じ構成を備える。データバッファ群ＤＬＡは、図４８および図４９に示すデータバッファＤＬ８～ＤＬ１５を含み、データバッファ群ＤＬＢは、データバッファＤＬ０～ＤＬ７を含む。テンプレートブロックのサイズに従ってこの演算部６４２の内部構成を変更するために、接続制御回路６４４の出力する制御信号に従ってテンプレートブロック画素データおよびサーチウィンドウ画素データの転送経路を切換えるセレクタＳＬａ～ＳＬｇが設けられる。セレクタＳＬａ～ＳＬｇの内部構成は後に詳細に説明する。接続制御回路６４４はこの動きベクトル検出装置の演算部外部から与えられるモード指定信号（テンプレートブロックサイズ指定信号）φＭに従ってセレクタＳＬａ～ＳＬｇの接続を確立する。この演算部６４２には、また入力部６４０から与えられるサーチウィンドウ画素データＰＸを接続制御回路６４４の制御の下にプロセッサブロックＰＢＡ～ＰＢＤへ選択的に分配するマルチプレクサ６４６が設けられる。接続制御回路６４４は、この指定信号φＭが８行８列のテンプレートブロックサイズを示すときには、処理単位時間（Ｔ）ごとにそのマルチプレクサ６４６の出力経路を切換える。これにより処理単位時間Ｔごとに入力部６４０から与えられる８行８列のテンプレートブロック画素データＰＸが、処理単位時間ごとに順次プロセッサブロックＰＢＡ～ＰＢＤへ与えられる。モード指定信号φＭが１６行・１６列のテンプレートブロックサイズを指定するときには、接続制御回路６４４は、このマルチプレクサ６４６の所定の１つの出力のみを選択状態としてプロセッサブロックＰＢＣへ与える。入力部６４０へは、図示しない参照画像メモリおよび現画像メモリからサーチウィンドウ画素データＰＹおよびテンプレートブロック画素データＰＸが与えられる。テンプレートブロックが１６行１６列のサイズを有するとき、外部からは、４処理単位時間（４Ｔ）ごとにテンプレートブロック画素データが与えられてプロセッサブロックＰＢＡ～ＰＢＤのテンプレートブロック画素データの更新が行なわれる。テンプレートブロックが８行・８列のサイズを有する場合、処理単位時間Ｔごとに８行・８列のテンプレートブロック画素データが与えられ、プロセッサブロックＰＢＡ～ＰＢＤがそれぞれ順次処理単位時間ごとにそのテンプレートブロック画素データの更新を受ける。

【０２０４】図５６は、図５５に示すセレクタＳＬｂの構成を概略的に示す図である。図５６においては、スイッチング素子は機械的スイッチで構成されるように示されるが、これらのスイッチング素子はトランスファゲートまたはトランスミッションゲートなどのトランジスタ素子で構成される。また、図５６においては、テンプレートブロック画素データを転送する経路に関連する構成のみを示す。サーチウィンドウ画素データ転送のためのセレクタ部の構成は後に示す。セレクタブロックＳＬｂは、プロセッサブロックＰＢＡにおいて隣接する２つのプロセッサグループの入力と出力とを信号／φＬＡの活性化時に接続するスイッチング素子ＳＷａｕと、信号／

φＬＡの活性化時にプロセッサグループ１５Ｕの入力（このグループ１５Ｕの最下流の要素プロセッサ）へテンプレートブロック画素データＰＸＡを与えるスイッチング素子ＳＷＸａと、プロセッサブロックＰＢＣにおいて隣接する２つのプロセッサグループの入力と出力とを信号／φＬＡの活性化時に接続するスイッチング素子ＳＷａ１と、プロセッサブロックＰＢＡおよびＰＢＣの同一列に配置されたプロセッサグループの出力と入力とを信号φＬＡの活性化時に接続するスイッチング素子Ｗｂｂと、プロセッサブロックＰＢＡのプロセッサグループの出力をプロセッサブロックＰＢＣの１列上位のプロセッサグループの入力へ接続するスイッチング素子ＳＷｂａとを含む。

【０２０５】信号φＬＡは、１６行・１６列のテンプレートブロックサイズが指定されたときに活性状態とされ、信号／φＬＡは８行・８列のテンプレートブロックサイズが指定されたときに活性状態とされる。プロセッサグループＰＧ８Ｕに対し、さらに信号／φＬＡの活性化時に導通状態とされるスイッチング素子ＳＷｄが設けられる。プロセッサグループＰＧ８Ｕの出力は、またスイッチング素子ＳＷｂｂを介してプロセッサブロックＰＢＣのプロセッサグループＰＧ７Ｌの入力へ接続される。プロセッサブロックＰＢＢのプロセッサグループＰＧ７ＵおよびプロセッサブロックＰＢＣのプロセッサグループＰＧ７Ｌへは、また信号／φＬＡに応答してテンプレートブロック画素データＰＸＢおよびＰＸＤをそれぞれ伝達するスイッチング素子ＳＷＸｃおよびＳＷＸｄが設けられる。スイッチング素子ＳＷＸｃおよびＳＷＸｄは、信号／φＬＡの活性化時に導通状態とされる。

【０２０６】プロセッサグループＰＧ８Ｕの出力部にさらに、信号φＬＡに応答して導通するスイッチング素子ＳＷｄが設けられる。このスイッチング素子ＳＷｄと同様の構成のスイッチング素子がまたプロセッサグループＰＧ８Ｌの出力部にも設けられる（図５６に示す）。このスイッチング素子ＳＷｄは、次段の回路でテンプレートブロック画素データをさらに処理する場合に、直接このテンプレートブロック画素データをその次段回路へ与えるために設けられる。この図５６に示すようにプロセッサグループの入力部および出力部それぞれにスイッチング素子を設けることにより、容易にテンプレートブロックのサイズに応じて要素プロセッサの配置を変更することができる。すなわち、テンプレートブロックのサイズに従って、プロセッサブロックＰＢＡ～ＰＢＤそれぞれにおいてプロセッサグループがシストリックアレイ状に接続されるかまたはプロセッサブロックＰＢＡ～ＰＢＤ全体として１つのシストリックアレイを構成することができる。

【０２０７】図５７は、図５５に示すセレクタのサーチウィンドウ画素データ転送に関連する部分の構成を概略的に示す図である。図５７においても、スイッチング素子は機械的スイッチで構成されるように示されるが、また、トランジスタ素子で構成される。セレクタＳＬａは、プロセッサブロックＰＢＡに含まれるプロセッサグループＰＧ８Ｕ～ＰＧ１５Ｕの出力部にそれぞれ設けられ、信号／φＬＡに応答して対応のプロセッサグループの出力する画素データを出力ノードＯＴＮへ伝達するスイッチング素子ＳＴａと、信号φＬＡに応答してプロセッサグループＰＧ８Ｕ～ＰＧ１５Ｕの出力する画素データを１列上流のデータバッファＤＬ７～ＤＬ１４の入力部へ伝達するスイッチング素子ＳＴｂを含む。信号／φＬＡは、先の図５６において説明した信号／φＬＡと同様、８行・８列のテンプレートブロックサイズが指定されたときに活性状態とされてスイッチング素子ＳＴａを導通状態とする。信号φＬＡは、１６行・１６列のテンプレートブロックサイズが指定されたときに活性状態とされ、スイッチング素子ＳＴｂを導通状態とする。

【０２０８】セレクタＳＬｂは、プロセッサブロックＰＢＡおよびＰＢＣにおいて、同一列に配置されたプロセッサグループの出力と入力を信号φＬＡの活性化時に接続するスイッチング素子ＳＴｃと、プロセッサグループＰＧ８Ｌ～ＰＧ１５Ｌの出力を１列上流に設けられたデータバッファＤＬ７～ＤＬ１４の入力に接続するスイッチング素子ＳＴｄを含む。セレクタＳＬｃは、データバッファＤＬ８～ＤＬ１５それぞれの出力部に設けられ、信号／φＬＡの活性化時に導通状態とされて対応のデータバッファＤＬ８～ＤＬ１５の出力する画素データを同一列に配置されたプロセッサブロックＰＢＡ内のプロセッサグループＰＧ８Ｕ～ＰＧ１５Ｕの入力（該グループにおける最下流の要素プロセッサ）へ伝達するスイッチング素子ＳＴｅを含む。

【０２０９】ここで、プロセッサグループＰＧ８Ｌの出力部にはスイッチング素子ＳＴｄは設けられていない。プロセッサグループＰＧ８Ｌとデータバッファ ＤＬ７とは、異なるテンプレートブロックに対応するプロセッサブロックに含まれており、したがってこのプロセッサグループＰＧ８Ｌからデータバッファ ＤＬ７へサーチウィンドウ画素データを転送する必要がないためである。このプロセッサグループＰＧ８Ｌの出力部には、信号／φＬＡの活性化時に導通状態とされるスイッチング素子ＳＴａと同様のスイッチング素子が設けられる（図５７には示さず）。プロセッサブロックＰＢＢおよびＰＢＤにおいても、プロセッサブロックＰＢＡおよびＰＢＣにおけるセレクタＳＬａおよびＳＬｂおよびＳＬｃと同様の構成が設けられる。このとき、データバッファＤＬ７には、セレクタＳＬｇとして、信号／φＬＡの活性化時に導通し、サーチウィンドウ画素データＰＹをそのデータバッファＤＬ７の入力へ伝達するスイッチング素子が設けられる。

【０２１０】図５７においては、８行・８列のテンプレートブロックサイズが指定されたときのスイッチング素

子の導通／非導通状態が一例として示される。これらスイッチング素子の導通／非導通状態を信号φＬＡおよび／φＬＡにより確立することにより、テンプレートブロックサイズに対応するプロセッサ配置を得ることができる。なお、図５７に示す構成においては、スイッチング素子ＳＴｂはセレクタＳＬａに含まれるように示される。しかしながら、このスイッチング素子ＳＴｂはセレクタＳＬｂに設けられてもよい。図５８は、この第６の実施例において用いられる動きベクトル決定部の構成を示す図である。図５８においては、１つの予測モードに関連する部分の構成のみを示す。各予測モードに対応して図５８に示す構成が基本的に設けられる（幾つかの回路部分の共用などの変更を行なうことはできる）。

【０２１１】図５８において、動きベクトル決定部は、プロセッサブロックＰＢＡ～ＰＢＤそれぞれに対応して設けられ、対応のプロセッサブロックから与えられる評価関数値成分の総和をそれぞれ求めることにより、評価関数値を導出する総和部６５０ａ～６５０ｄと、総和部６５０ａ～６５０ｄそれぞれに対応して設けられる１入力２出力のセレクタ６５２ａ～６５２ｄを含む。セレクタ６５２ａ～６５２ｄの各々は、信号φＬＡが非活性状態にあり、８行・８列のテンプレートブロックサイズが指定されたとき対応の総和部から与えられる評価関数値をその第１の出力へ伝達し、信号φＬＡが活性状態であり１６行・１６列のテンプレートブロックが指定されたときには与えられた評価関数値を第２の出力へ伝達する。動きベクトル検出部は、さらに、セレクタ６５２ａ～６５２ｄのそれぞれの第１の出力に対応して配置される比較部６５４ａ～６５４ｄと、セレクタ６５２ａ～６５２ｄのそれぞれの第２の出力から与えられる評価関数値を加算する加算部６５５と、加算部６５５の出力データを比較することにより１６行・１６列のテンプレートブロックの動きベクトルを決定する比較部６５６と、比較部６５４ａ～６５４ｄおよび６５６の出力データを、信号φＬＡＴに従って選択するセレクタ６５８を含む。比較部６５４ａ～６５４ｄおよび６５６は、それぞれ与えられた評価関数値の最小値を選択して対応のテンプレートブロックについての動きベクトルを示すデータｍｖＡ～ｍｖＤおよびｍｖＬを出力する。

【０２１２】比較部６５４ａ～６５４ｄが、信号φＬＡの活性化時、すなわち１６行・１６列のテンプレートブロックサイズが指定されたときには非活性状態とされ、比較部６５６のみが活性状態とされる構成が用いられてもよい。また、信号φＬＡが非活性状態にあり、８行・８列のテンプレートブロックサイズが指定されたときには加算部６５５および比較部６５６の動作が禁止されるように構成されてもよい。セレクタ６５８は、比較部６５４ａ～６５４ｄおよび６５６の出力する動きベクトルデータを受ける。セレクタ６５８へ与えられる信号φＬＡＴは多ビット信号である。この信号φＬＡＴにより、セレクタ６５８は、１６行×１６列のテンプレートブロックが指定されたときには比較部６５６の出力する動きベクトルデータをｍｖＬを選択して出力する。８行・８列のテンプレートブロックサイズが指定されたとき、セレクタ６５８は比較部６５４ａ～６５４ｄの出力する動きベクトルデータｍｖＡ～ｍｖＤを、順次、処理単位時間ごとに選択して出力する。このセレクタ６５８へ与えられる多ビット信号φＬＡＴは、外部の制御装置から与えられる。この外部制御装置は、図５５に示すモード指定信号φＭを発生し、参照画像メモリおよび現画像メモリからそれぞれ所定のタイミングでサーチウィンドウ画素データおよびテンプレートブロック画素データを読出して演算部へ与えるとともに、このセレクタ６５８における選択動作を制御する。この外部制御装置を備える動きベクトル検出装置の全体の構成は、先に図３８を参照して説明した構成と同様の構成となり、ここでは示さない。

【０２１３】［変更例］今、図５９（Ａ）に示すように、１６行・１６列のサイズを有するテンプレートブロックにおいて水平および垂直方向いずれにおいても画素をサブサンプリングする場合を考える。すなわち、図５９（Ａ）において○印で示す画素について評価関数値の成分導出が行なわれて、×印で示される画素については評価関数値成分は導出されない。この場合、評価関数値成分を導出する画素のみを抽出すれば、図５９（Ｂ）に示すように８行・８列に配置された画素からなるテンプレートブロックが得られる。サーチウィンドウデータについても同様水平および垂直方向両方向に沿ってサブサンプリングが行なわれる。サーチウィンドウ６６０は、４０行・８列に配置された画素を備える。このサーチウィンドウ６６０に含まれる画素はすべて評価関数値成分が導出されるべき画素である。今、サーチウィンドウブロック６６１の変位ベクトルが（Ｈ，Ｖ）のとき、次のサイクルにおいては、図５９（Ｄ）に示すようにサーチウィンドウブロック６６２の変位ベクトルは（Ｈ，Ｖ＋２）となる。すなわち、図５９（Ａ）において、×印で表わされる画素が省略されているためである。この変位ベクトルの変化量は、列方向に沿っても２となる。したがって、図５９（Ｂ）に示すようなサブサンプリング画像を用いて動きベクトルを検出する場合、その水平および垂直方向の動きベクトル探索範囲が水平方向１５変位、垂直方向６５変位となり、等価的に動きベクトル探索範囲が拡張される。これによりサブサンプリング画素を用いている場合においても、動きベクトル探索範囲を拡張してより正確な動きベクトル検出を行なうことができる。

【０２１４】なお、この２：１サブサンプリング画像は、言うまでもなく一般のサブサンプリング画像に拡張可能である。また、図５６および図５７に示す構成を一般の構成に拡張する場合、単に図の上方向にプロセッサ

ブロックＰＢＡの構成がそのまま拡張されればよい。またさらに、上述の構成においては、テンプレートブロックのサイズに応じてプロセッサアレイを制御信号により再配置可能としているが、適用される用途が予め固定的に設定される場合には、以下の構成を利用することができる。すなわち、プロセッサグループおよびデータバッファのみを配置し、その用途に応じてサーチウィンドウ画素データおよびテンプレートブロック画素データを伝達する経路をマスク配線により設定する。同一チップを用いて複数のテンプレートブロックサイズに容易に対応することができる。

【０２１５】また、単に、１６行・１６列のテンプレートブロックを８行・８列のテンプレートブロックに分割するだけであり、動きベクトル検出部の構成のみを図５８に示す構成とすれば、４つの８行・８列のテンプレートブロックに対し同時に動きベクトル検出動作を行なうことができる（この場合、すべてのテンプレートブロックについての変位ベクトルは同じ値となる）。以上のように、この発明の第６の実施例の構成に従えば、テンプレートブロックのサイズに従って演算部に含まれる要素プロセッサの配置を変更可能となるように構成したため、自由度の高い画像符号化システムを構成することが可能となる。

［実施例７］図６０は、この発明の第７の実施例である動きベクトル検出装置の要部の構成を示す図である。図６０においては、１つの要素プロセッサＰＥの構成を示す。図６０において、要素プロセッサＰＥは、それぞれ、異なるテンプレートブロックの画素データを格納するレジスタ（Ａ）６６０ａおよびレジスタ（Ｂ）６６０ｂと、制御信号φＡに従ってレジスタ（Ａ）６６０ａおよびレジスタ（Ｂ）６６０ｂの出力する画素データの一方を選択して隣接する要素プロセッサまたはデータバッファへ伝達するセレクタ６６２と、信号φＡに応答して隣接要素プロセッサまたは入力部から与えられるテンプレートブロック画素データをレジスタ（Ａ）６６０ａおよびレジスタ（Ｂ）６６０ｂの一方へ与えるマルチプレクサ６６１と、信号φＡに応答してレジスタ（Ａ）６６０ａおよびレジスタ（Ｂ）６６０ｂが格納しかつ出力する画素データを選択して出力するセレクタ６６３と、隣接要素プロセッサまたはデータバッファから与えられるサーチウィンドウ画素データを格納するとともに図示しないクロック信号に応答して隣接する要素プロセッサまたはデータバッファへ伝達するレジスタ６６４と、セレクタ６６３の出力する画素データａとレジスタ６６４の出力する（格納された）画素データｂとの減算を行なう減算器６６５と、減算器６６５の出力データの絶対値をとる絶対値演算器６６６を含む。

【０２１６】信号φＡがレジスタ（Ａ）６６０ａを指定する場合には、隣接要素プロセッサまたは入力部から与えられたテンプレートブロック画素データＰＸがマルチプレクサ６６１を介してレジスタ（Ａ）６６０ａへ与えられ、またこのレジスタ（Ａ）６６０ａの格納するデータまたは出力する画素データはセレクタ６６２により選択されて隣接要素プロセッサまたは出力部へ与えられる。このときには、セレクタ６６３は、レジスタ（Ｂ）６６０ｂの格納する画素データ（または出力する画素データ）を選択する状態にある。絶対値演算器６６６から差分絶対値｜ａ－ｂ｜が評価関数値成分として出力され、図示しない総和部において総和されて評価関数値が生成される。レジスタ（Ａ）６６０ａ、レジスタ（Ｂ）６６０ｂおよびレジスタ６６４は、その内部に転送ゲートを備えるシフトレジスタの構成を備えていてもよく、単にデータをラッチする構成を備え、データ転送部は要素プロセッサ外部に配置される構成であってもよい。

【０２１７】この図６０に示す構成においては、レジスタ（Ａ）６６０ａおよびレジスタ（Ｂ）６６０ｂの一方に格納されたテンプレートブロック画素データを用いて演算が行なわれる。他方のレジスタに対し別のテンプレートブロックの画素データが与えられて格納される。すなわち、１つのテンプレートブロックについての動きベクトル検出動作の間に他方のレジスタに対し別のテンプレートブロックの画素データのロード動作が行なわれる。図６１は、この図６０に示す要素プロセッサを含む動きベクトル検出装置における処理動作シーケンスの一例を示す図である。以下、図６１を参照してこの発明の第７の実施例の処理動作について説明する。時間Ｔ１においては、レジスタ（Ａ）に格納されたテンプレートブロックＮの画素データを用いて演算操作（差分絶対値演算）が行なわれ、図６０に示すレジスタ６６４に格納されたサーチウィンドウ画素データＰＩを順次シフトすることにより、このテンプレートブロックＮに対する動きベクトル検出動作が行なわれる。すなわち、図６０に示すセレクタ６６３は信号／φＡによりレジスタ（Ａ）に格納された画素データを選択する状態に設定される。マルチプレクサ６６１およびセレクタ６６２は、信号φＡによりレジスタ（Ｂ）を選択する状態に設定される。これにより、次に指示されるべきテンプレートブロック（Ｎ＋１）の画素データが順次シフト動作によりレジスタ（Ｂ）６６０ｂに格納される。

【０２１８】期間Ｔ１が完了し、テンプレートブロックＮの動きベクトル検出動作が完了すると、そのときには次に処理されるべきテンプレートブロックＮ＋１の画素データのロードは完了している。マルチプレクサ６６１およびセレクタ６６２が、信号φＡに従ってレジスタ（Ａ）６６０ａの選択状態に設定される。セレクタ６６３はレジスタ（Ｂ）６６０ｂを選択する状態に設定される。この状態においては、レジスタ（Ｂ）６６０ｂに格納されたテンプレートブロックＮ＋１の画素データを用いて動きベクトル検出動作が行なわれる。動きベクトル検出動作時においては、参照画像メモリへのアクセスは

行なわれる（サーチウィンドウ画素データを演算部へシフトインする必要があるため）。しかしながらテンプレートブロック画素データを格納する現画像メモリへはアクセスは行なわれない。したがって、この間、次に処理されるべきテンプレートブロックの画素データを現画像メモリから読出して演算部へ与えることができる。すなわち、期間Ｔ２においてテンプレートブロックＮ＋２の画素データのロードが行なわれ、この与えられた画素データがレジスタ（Ａ）６６０ａに順次格納される。

【０２１９】以降、期間Ｔ３、Ｔ４、Ｔ５およびＴ６において、レジスタ（Ａ）６６０ａに対してはテンプレートブロックＮ＋２についての動きベクトル探索、テンプレートブロックＮ＋４の画素データのロード、テンプレートブロックＮ＋４の動きベクトルの探索およびテンプレートブロックＮ＋６の画素データのロードが行なわれる。レジスタ（Ｂ）６６０ｂに対しては、期間Ｔ３～Ｔ６それぞれにおいて、テンプレートブロックＮ＋３の画素データのロード、テンプレートブロックＮ＋３の動きベクトルの探索、テンプレートブロックＮ＋５の画素データのロードおよびテンプレートブロックＮ＋５の動きベクトルの検出が行なわれる。以上のように、テンプレートブロック画素データを格納するためのレジスタを各要素プロセッサ内に２つ設け、一方のレジスタに格納されたテンプレートブロック画素データを用いて動きベクトル検出動作を行なっている間に他方のレジスタへ次に処理されるべきテンプレートブロック画素データを格納する。これにより、テンプレートブロックの画素データの入力とテンプレートブロックについての動きベクトル検出処理を並行して実行することができ、処理時間を大幅に短縮することができる。

【０２２０】図６２は、この発明の第７の実施例の動きベクトル検出装置の全体の構成を概略的に示す図である。動きベクトル検出装置６７０は、サーチウィンドウ画素データおよびテンプレートブロック画素データを入力する入力部６７１と、この入力部６７１から与えられたテンプレートブロック画素データおよびサーチウィンドウブロック画素データに所定の演算を施して評価関数値ｅｍｖを生成する演算部６７２と、演算部６７２からの評価関数値ｅｍｖに従って動きベクトルＭＶを決定する比較部６７３を含む。図６２においては、複数の予測モードのうち１つの予測モードに従って決定される評価関数値ｅｍｖおよび動きベクトルＭＶのみを示す。演算部６７２および比較部６７３は、それぞれ複数の予測モードに従って評価関数値および動きベクトルを検出する。この演算部６７２におけるテンプレートブロック画素データの格納を制御するために制御装置６７４が設けられる。制御装置６７４は、単位処理時間ごとに信号φＡおよび／φＡを交互に活性状態とし、演算部６７２においてテンプレートブロック画素データを入力するレジスタを切換える。この制御装置６７４は、またサーチウィンドウ画素データを格納する参照画像メモリ６７５およびテンプレートブロック画素データを格納する現画像メモリ６７６に対するデータの読出動作をも制御する。この制御装置６７４の制御の下に、１つのテンプレートブロックについて参照画像メモリ６７５からサーチエリア内の画素データが入力部６７１へ与えられて演算部６７２に入力されている間、現画像メモリ６７６から次に処理されるべきテンプレートブロックの画素データが読出されて入力部６７１を介して演算部６７２へ与えられる。

【０２２１】［変更例１］図６３は、この発明の第７の実施例の第１の変更例の構成を示す図である。図６３においては、テンプレートブロック画素データを入力部６７１から受ける初段の要素プロセッサの構成を示す。しかしながら、他の要素プロセッサの構成もこの図６３に示す要素プロセッサの構成と同じである。図６３に示す要素プロセッサＰＥにおいては、テンプレートブロック画素データを振り分けるためのマルチプレクサおよびセレクタは設けられない。他の構成は図６０に示す構成と同じであり、対応する部分には同一の参照番号を付す。入力部６７１と初段の要素プロセッサＰＥの間に、テンプレートブロック画素データを信号φＡに従ってレジスタ（Ａ）６６０ａおよびレジスタ（Ｂ）６６０ｂの一方へ伝達するマルチプレクサ６７５が設けられる。レジスタ（Ａ）６６０ａの出力するテンプレートブロック画素データは、シフト動作時においては隣接する要素プロセッサのレジスタ（Ａ）へ与えられる。同様、レジスタ（Ｂ）６６０ｂに格納されるテンプレートブロック画素データは、シフト動作時には隣接する要素プロセッサのレジスタ（Ｂ）６６０ｂに与えられる。すなわち、入力部６７１の次段に設けられたマルチプレクサ６７５により、レジスタ（Ａ）６６０ａに格納されるテンプレートブロック画素データの伝達経路とレジスタ（Ｂ）６６０ｂに格納されるテンプレートブロック画素データの経路を切換える。要素プロセッサ間においては、レジスタ（Ａ）はレジスタ（Ａ）に接続され、レジスタ（Ｂ）は同様レジスタ（Ｂ）に接続される。この図６３に示す構成の場合、要素プロセッサＰＥ内にマルチプレクサおよび隣接要素プロセッサ間のテンプレートブロック画素データ転送のためのセレクタを設ける必要がなく、要素プロセッサの規模を低減することができる。

【０２２２】なお図６３に示す構成においてはレジスタ（Ａ）６６０ａおよびレジスタ（Ｂ）６６０ｂは、単にデータを格納する機能のみを備えるように示しているが、これはシフトレジスタのようにデータシフト機能を備えていてもよい。シフトレジスタ構成の場合、データ転送時において一方のレジスタにおいてのみ転送動作を行なうようにクロック信号が与えられ、他方のレジスタへは画素データ転送のためのクロック信号は与えられない。

［変更例２］図６４は、この発明の第７の実施例の第２の変更例の構成を示す図である。図６４においては、１つの要素プロセッサのテンプレートブロック画素データ格納用のレジスタ６６０ａおよび６６０ｂの構成を示す。他の構成は図６０および図６３に示す要素プロセッサのそれと同じである。

【０２２３】図６４（Ａ）において、レジスタ（Ａ）６６０ａは、クロック信号φ１Ａに応答して下流の隣接要素プロセッサまたは入力部から与えられるテンプレートブロック画素データを伝達するトランスファゲート６７６ａと、トランスファゲート６７６ａを介して与えられたテンプレートブロック画素データをラッチするラッチ６７７ａと、クロック信号／φ１Ａに応答してこのラッチ回路６７７ａの格納する画素データを上流の隣接要素プロセッサまたは出力部へ伝達するトランスファゲート６７８ａを含む。レジスタ（Ｂ）６６０ｂは、クロック信号φ１Ｂに応答して下流の隣接要素プロセッサまたは入力部から与えられたテンプレートブロック画素データを伝達するトランスファゲート６７６ｂと、このトランスファゲート６７６ｂを介して与えられた画素データをラッチするラッチ回路６７７ｂと、ラッチ回路６７７ｂのラッチする画素データをクロック信号／φ１Ｂに応答して上流の隣接要素プロセッサまたは出力部へ伝達するトランスファゲート６７８ｂを含む。演算部（減算器および絶対値回路）へテンプレートブロック画素データを与えるセレクタ６６３へは、ラッチ回路６７７ａおよび６７７ｂがそれぞれラッチする画素データが与えられる。信号φＡがレジスタ（Ａ）６６０ａを指定するときセレクタ６６３はラッチ６７７ａがラッチする画素データを選択する。信号φＡがレジスタ（Ｂ）６６０ｂを指定するとき、セレクタ６６３はラッチ６７７ｂのラッチする画素データを選択する。

【０２２４】図６４に示すレジスタ（Ａ）６６０ａおよびレジスタ（Ｂ）６６０ｂの構成は、本質的にシフトレジスタである。これらのレジスタ６６０ａおよび６６０ｂの入力部を相互接続し、またその出力部を相互接続する。テンプレートブロック画素データをロードすべきレジスタに対してのみクロック信号φ１Ａおよび／φ１Ａ（またはφ１Ｂおよび／φ１Ｂ）が与えられる。ラッチする画素データが演算処理される場合には、クロック信号φ１Ｂおよび／φ１Ｂ（またはφ１Ａおよび／φ１Ａ）はともに非活性状態のローレベルとされ、その内部のラッチはレジスタ外部と分離される。これにより、一方のレジスタのテンプレートブロック画素データ転送動作が他方のレジスタに格納された画素データに影響を及ぼすのを防止することができる。次にこの図６４（Ａ）に示す要素プロセッサの動作をその動作波形図である図６４（Ｂ）を参照して説明する。

【０２２５】レジスタ（Ａ）にテンプレートブロック画素データをロードする際には、信号φ１Ａおよび／φ１Ａが順次ハイレベルとされる。クロック信号φＡおよび／φＡは互いに重なり合わない２相のクロック信号である。レジスタ（Ｂ）６６０ｂに対するクロック信号φ１Ｂおよび／φ１Ｂがともにローレベルとされ、トランスファゲート６７６ｂおよび６７８ｂはともにオフ状態とされる。セレクタ６６３は、信号φＡに従って、ラッチ６７７ｂのラッチする画素データを選択して演算部へ与える。これにより、ラッチ６７７ｂすなわちレジスタ（Ｂ）に格納されたテンプレートブロック画素データを用い動きベクトル検出動作が行なわれる。この動きベクトル検出動作中に、クロック信号φ１Ａおよび／φ１Ａが順次活性状態とされて別のテンプレートブロック画素データのシフトイン／シフトアウト動作が行なわれる。クロック信号φ１Ａがハイレベルとなると、トランスファゲート６７６ａがオン状態とされ、一方、このとき、信号／φ１Ａはローレベルであり、トランスファゲート６７８ａはオフ状態にある。したがって下流の隣接要素プロセッサまたは入力部から与えられたテンプレートブロック画素データがラッチ６７７ａにラッチされる。次いでクロック信号φ１Ａがローレベルとなり、トランスファゲート６７６ａがオフ状態とされ、クロック信号／φ１Ａがハイレベルとされ、トランスファゲート６７８ａがオン状態とされ、ラッチ６７７ａに格納された画素データが上流の隣接要素プロセッサまたは出力部へ伝達される。これにより、レジスタ（Ａ）６６０ａを介して順次次に処理されるべきテンプレートブロック画素データを転送して対応のレジスタ（Ａ）６６０ａにテンプレートブロック画素データを格納することができる。

【０２２６】レジスタ（Ｂ）６００ｂに格納された画素データを用いて動きベクトル検出動作が完了すると、このときにレジスタ（Ａ）６６０ａには既に次の処理されるべきテンプレートブロック画素データは格納されている。この状態において、クロック信号φ１Ａおよび／φ１Ａをともにローレベルとし、トランスファゲート６７６ａおよび６７８ａをともにオフ状態とする。同時にセレクタ６６３を信号φＡによりレジスタ（Ａ）６６０ａの選択状態に設定する。これにより、レジスタ（Ａ）６６０ａに格納されたテンプレートブロック画素データを用いた動きベクトル検出動作が行なわれる。このとき、今度はクロック信号φ１Ｂおよび／φ１Ｂが順次ハイレベルとされて次に処理されるべきテンプレートブロック画素データのレジスタ（Ｂ）６６０ｂへの格納が行なわれる。

【０２２７】この図６４に示す構成の場合、単にシフトレジスタ構成を備えるレジスタを用いているだけであり、テンプレートブロック画素データの転送経路を切換えるためのマルチプレクサおよびセレクタを設ける必要はない。したがって、要素プロセッサの規模をより低減することができる。以上のように、この発明の第７の実施例に従えば、要素プロセッサ内に異なるテンプレート

ブロックの画素データを格納する２つのレジスタを設け、これら一方のレジスタに格納された画素データを用いての動きベクトル検出動作と並行して他方のレジスタへ次に処理されるべきテンプレートブロック画素データを格納する（ロードする）ように構成したため、テンプレートブロック画素データ入力と動きベクトル検出動作を並行して行なうことができ、テンプレートブロック画素データの更新に要する時間を見かけ上なくすことができ、動きベクトル検出の処理時間を大幅に短縮することができる。

【０２２８】［実施例８］図６５は、この発明の第８の実施例の動きベクトル検出装置の要部の構成を示す図である。図６５においては、動きベクトル検出装置の動きベクトルを検出する比較部の構成を示す。図６５において、比較部は、演算部で算出された評価関数値ｅｖのうち最小の評価関数値を格納するレジスタ６８０と、レジスタ６８０に格納された評価関数値と与えられた評価関数値を比較する比較器６８２と、与えられた評価関数値を有する変位ベクトルを表わす情報を発生するカウンタ６８４と、最小の評価関数値に対応する変位ベクトルを動きベクトル候補として格納するレジスタ６８６と、比較器６８２からの等値指示信号φｅｑに応答して活性化され、カウンタ６８４の出力するカウント値およびレジスタ６８６に格納された変位ベクトル値を受けて優先順位を判定する優先順位判定回路６９０と、比較器６８２からの与えられた評価関数値が小さいことを示す信号φｓｍ１と優先順位判定回路６９０からの更新指示信号φｒｗに応答してレジスタ６８０および６８６の内容の更新する更新制御回路６８８を含む。更新制御回路６８８は信号φｓｍ１およびφｒｗの一方が活性状態のときにレジスタ６８０および６８６の内容を更新する。

【０２２９】カウンタ６８４は、通常の２進カウンタであってもよく、また水平方向ベクトルＨおよび垂直方向ベクトルＶを表わすカウント値をそれぞれ別々に発生するカウンタであってもよい。水平方向および垂直方向のベクトル値をそれぞれ別々に示すカウンタの場合、垂直方向探索範囲の大きさに従ってカウント値が設定され、その垂直方向ベクトルカウンタが所定値をカウントアップしたときに水平方向ベクトルカウンタのカウント値が１増分される。いずれの構成が利用されてもよい。優先順位判定回路６９０は、３つの優先順位判定基準を実現可能なように構成され、用いられる用途に応じてこれらの３つの優先範囲順位判定基準のうちの１つが選択され、選択された優先順位基準に従って優先順位判定を行なう。

【０２３０】図６６および図６７は、図６５に示す優先順位判定回路６９０において利用される第１の優先順位判定基準を一覧として示す図である。図６６および図６７においては、水平方向探索範囲が－１６～＋１５、垂直方向探索範囲が－１６～＋１６の場合の優先順位判定基準が示される。図６６には水平方向のＨベクトルが０または負の領域の優先範囲判定基準が示され、図６７においては、水平方向変位ベクトルＨの正の値のときの優先順位判定基準が示される。この図６６および図６７に示す第１の優先順位判定基準の場合、変位ベクトルの優先順位は、Ｐ（Ｈ，Ｖ）＝｜Ｈ｜＋｜Ｖ｜に従って設定される。評価関数値が同じ値のとき、この優先順位評価値Ｐ（Ｈ，Ｖ）が小さいときに優先順位が高いとされ、高い方の優先順位を有する変位ベクトルが動きベクトル候補として選択される。

【０２３１】この図６６および図６７に示す優先順位判定基準の場合、優先順位判定基準において真裏の領域（ベクトル０，０）が最も優先順位が高くされ、この中央部から離れるに従って優先順位が低くされる。図６８および図６９は、図６５に示す優先順位判定回路の第２の優先順位判定基準を示す図である。図６８および図６９においても、水平方向－１６～＋１５および垂直方向－１６～＋１６を動きベクトル探索範囲とする場合の優先順位評価値が一覧にして示される。この図６８および図６９に示す優先順位判定基準の場合、変位ベクトル（Ｈ，Ｖ）の優先順位評価値Ｐ（Ｈ，Ｖ）は、Ｐ（Ｈ，Ｖ）＝｜Ｈ｜＋（－Ｖ＋１６）で表される。優先順位評価基準値Ｐ（Ｈ，Ｖ）が小さいほど高い優先順位が与えられる。

【０２３２】この図６８および図６９に示す優先順位判定基準の場合、変位ベクトル（０，１６）が最も高い優先順位が与えられ、そこから離れるに従って優先順位が低くされる。すなわち、サーチエリアにおいて中央最下端部が最も優先順位が高くされる。図７０および図７１は、図６５に示す優先順位判定回路の第３の優先順位判定基準を示す図である。図７０においては、この第３の優先順位判定基準における水平方向が負の領域の優先順位判定基準値を示し、図７１には、探索範囲の水平方向の正の領域における優先順位評価基準値を示す。この第３の優先順位判定基準においては、優先順位評価値Ｐ（Ｈ，Ｖ）は、

Ｐ（Ｈ，Ｖ）＝｜Ｈ｜＋（Ｖ＋１６）

で表される。優先順位評価値Ｐ（Ｈ，Ｖ）の値が小さい変位ベクトルに対しより高い優先順位が与えられる。この第３の優先順位判定基準に従えば、水平方向中央部において垂直方向最上端部（垂直方向変位ベクトル：Ｖベクトルの値が最も小さな変位ベクトル）に対し最も高い優先順位が与えられる。優先順位判定基準として上述のように３つの判定基準を設けることにより、以下に述べるように、動きベクトル検出のための垂直方向の探索範囲の拡張に容易に対応することができる。

【０２３３】図７２（Ａ）は、１つのチップＣＨ１で動きベクトル検出装置が構成される状態を示す。この状態においては、第１の優先順位判定基準に従って変位ベクトルの優先順位の判定が行なわれる。この場合、図７２

（Ｂ）に示すように、動きベクトル探索範囲ＳＡＲにおいては、優先順位が｜Ｈ｜＋｜Ｖ｜に従って判定されるため、その中央部の変位ベクトル（ベクトル（０，０））が最も高い優先順位を与えられる。したがって、テンプレートブロックの真裏のサーチウィンドウブロックに対し最も高い優先順位が与えられることになる。動きベクトル探索範囲を垂直方向に２倍に拡張する場合には、図７３（Ａ）に示すように、２つの動きベクトル検出装置チップＣＨ２およびＣＨ３を用いる。動きベクトル検出装置チップＣＨ２へは、探索範囲の上半分のサーチウィンドウ画素データＰＹＵが与えられ、この探索範囲の上半分について動きベクトル検出はこの動きベクトル検出装置チップＣＨ２において行なわれる。動きベクトル検出装置チップＣＨ３に対しては、探索範囲の下側半分のサーチウィンドウ画素データＰＹＬが与えられ、探索範囲の下側半分の領域についての動きベクトル検出が行なわれる。動きベクトル検出装置チップＣＨ２およびＣＨ３に対しては共通に同じテンプレートブロック画素データＰＸが与えられる。動きベクトル検出装置チップＣＨ２においては、第２の優先順位判定基準に従って変位ベクトルの優先順位判定が行なわれ、動きベクトル検出装置チップＣＨ３においては、第３の動きベクトル判定基準に従って変位ベクトルの優先順位判定が行なわれる。

【０２３４】この場合、図７３（Ｂ）に示すように、動きベクトル検出装置チップＣＨ２においては、その探索範囲の水平方向中央部でかつ垂直方向最下端部の変位ベクトルに対し最も高い優先順位が与えられる。一方、動きベクトル検出装置チップＣＨ３においては、探索範囲において水平方向中央部、垂直方向最上部の変位ベクトルに対し最も高い優先順位が与えられる。したがって、図７３（Ａ）に示す比較部ＣＭＰで動きベクトル検出装置ＣＨ２および動きベクトル検出装置ＣＨ３が検出した動きベクトルからその評価関数値に従ってさらに最適ベクトルを判定することにより、拡張された動きベクトル探索範囲に対しても、真裏の位置の変位ベクトルに対し最も高い優先順位を与えた優先順位判定を行なった動きベクトル検出を行なうことができる。したがって動きベクトル探索範囲の垂直方向拡張時においても常に変位ベクトル（０，０）を中心とする優先順位に従って動きベクトル検出を行なうことができ、テンプレートブロックにより近いサーチウィンドウブロックを予測画像ブロックとして指定する動きベクトル選択を行なうことができる。また、２つの動きベクトル検出装置チップを用いることにより、垂直方向の探索範囲が２倍に拡張された場合においても、２つの動きベクトル検出装置チップを並列に動作させることによりその動きベクトル検出に要す時間は拡張前において必要とされた処理時間と同じであり、高速処理が可能となる。

【０２３５】ここで、垂直方向についての動きベクトル探索範囲は、２倍に限らず、３倍、４倍と容易に拡張することができる。この動きベクトルの垂直方向探索範囲を拡張する場合、単位となる垂直方向の探索範囲が$-\alpha$～$+\alpha$の場合、第２の優先順位判定基準はＰ（Ｈ，Ｖ）＝｜Ｈ｜＋（$-$Ｖ$+\alpha$）と表現され、第３の優先順位判定基準は、｜Ｈ｜＋（Ｖ$+\alpha$）で表される。この３つの優先順位判定基準それぞれを備える優先順位判定回路を、チップの用途に応じて別々に形成することはできるが、以下に説明するように、単に制御信号に従って、これらの３つの優先順位判定基準のうちの１つを利用することもできる。図７４は、図６５に示す優先順位判定回路の構成の一例を示す図である。

【０２３６】図７４において、優先順位判定回路６９０は、処理中のサーチウィンドウブロックの変位ベクトルを示すデータｍｖを出力するカウンタ６８４および動きベクトル候補を格納するレジスタ６８６からのベクトルＭＶａをタイミング信号ϕｔｉｍに従って順次時分割的に選択するセレクタ７０１と、セレクタ７０１から与えられたベクトルのうち水平方向ベクトル成分を抽出してその絶対値を生成するＨ絶対値回路７０２と、セレクタ７０１から与えられたベクトルデータから垂直方向成分Ｖの絶対値を示すデータを生成するＶ絶対値回路７０３と、セレクタ７０１から与えられたベクトルの垂直方向ベクトル成分Ｖを抽出するＶ抽出回路７０４と、セレクタ７０１から与えられたベクトルの垂直方向成分の符号を反転した値、$-$Ｖを生成する$-$Ｖ抽出回路７０５を含む。これらの回路７０２～７０５は、カウンタ６８４の出力するベクトル情報が、単に２進カウント値である場合、それぞれ内部にＨ成分およびＶ成分を生成するデコーダ回路を備えていてもよい。またカウンタ６８４が、Ｈ成分およびＶ成分をそれぞれ別々に生成する構成の場合には、単にこれらの回路７０２～７０５は対応の成分を受けて所望の演算処理を行なって必要とされるデータを生成する。また回路７０３および７０５は、Ｖ抽出回路７０４の出力を受けるように構成されてもよい。この場合、Ｖ絶対値回路７０３は、Ｖ抽出回路７０４の出力するデータを受ける絶対値回路で置き換えられ、$-$Ｖ抽出回路７０５は、Ｖ抽出回路７０４の出力するデータの符号反転を行なう符号反転回路で置き換えられる。

【０２３７】優先順位判定回路６９０は、さらに、Ｖ抽出回路７０４の出力する垂直成分データＶと$-$Ｖ抽出回路７０５の出力する符号反転された垂直方向成分$-$Ｖの一方をモード指定信号ϕｍｏｄａに従って選択するセレクタ７０６と、セレクタ７０６の出力するデータと一定値１６（＝α）を加算する加算器７０７と、Ｖ絶対値回路７０３の出力するデータ｜Ｖ｜と加算器７０７の出力するデータの一方をモード指定信号ϕｍｏｄｂに従って選択するセレクタ７０８と、セレクタ７０８の出力するデータとＨ絶対値回路７０２の出力するデータを加算する加算器７０９を含む。セレクタ７０６は、モード指定

信号φｍｏｄａが第２の優先順位判定基準を指定するときには－Ｖ抽出回路７０５の出力するデータ－Ｖを選択し、信号φｍｏｄａが第３の優先順位判定基準を指定するときにはＶ抽出回路７０４の出力するデータＶを選択する。セレクタ７０８は、モード指定信号φｍｏｄｂが第１の優先順位判定基準を指定するときにはＶ絶対値回路７０３の出力するデータ｜Ｖ｜を選択し、第２または第３の優先順位判定基準を指定するときには加算器７０７の出力するデータを選択する。

【０２３８】優先順位判定回路６９０は、さらに、タイミング信号φｔｉｍに従って加算器７０９の出力をその２出力の一方または他方へ伝達するマルチプレクサ７１０と、マルチプレクサ７１０の一方および他方出力にそれぞれ対応して設けられるレジスタ７１１および７１２と、レジスタ７１１および７１２の格納するデータの大小を比較する比較器７１３を含む。比較器７１３から更新制御信号φｒｗが生成される。このマルチプレクサ７１０は、タイミング信号φｔｉｍに従って、時分割的に、この加算器７０９の出力データをレジスタ７１１および７１２へ格納する。すなわち、タイミング信号φｔｉｍがカウンタ６８４のカウント値を選択する状態に設定されたとき、セレクタ７０１がカウンタ６８４のカウント値を選択し、一方、マルチプレクサ７１０は加算器７０９の出力をレジスタ７１２へ伝達する。一方、このタイミング信号φｔｉｍがレジスタ６８６の出力するデータＭＶａを選択する状態に設定されたときには、マルチプレクサ７１０は、このタイミング信号φｔｉｍに従って加算器７０９の出力するデータをレジスタ７１１に格納する。これにより、レジスタ７１１には、レジスタ６８６の格納する動きベクトル候補の優先順位判定基準値が格納され、レジスタ７１２には、カウンタ６８４の出力する現在処理中のサーチウィンドウブロックの変位ベクトルの優先順位評価基準値データが格納される。レジスタ７１１および７１２にそれぞれ優先順位評価基準値データが格納された後に比較器７１３が大小比較を行なう。処理中のサーチウィンドウブロックの変位ベクトルの優先順位評価基準値が、レジスタ６８６に格納されている動きベクトル候補の変位ベクトルの優先順位評価基準値よりも小さい場合には、比較器７１３から活性状態の更新制御信号φｒｗが出力される。

【０２３９】図７５は、図７４に示す優先順位判定回路の動作シーケンスの一例を示す図である。図７５においては、１つの優先順位判定動作サイクルのみを示す。モード指定信号φｍｏｄａおよびφｍｏｄｂは、それぞれ予め所定の状態に設定され、対応の動きベクトル検出装置が従うべき優先順位判定基準が設定される。レジスタ６８６の格納された動きベクトル候補の評価関数値とカウンタ６８４の出力する変位ベクトルの評価関数値とが等しいことが示されたとき、この優先順位判定回路が活性状態とされる。この活性状態とされたときにタイミング信号φｔｉｍが活性状態とされる。タイミング信号φｔｉｍが時刻Ｔａにおいてハイレベルに立上がると、セレクタ７０１がレジスタ６８６に格納されたそれまでに与えられている最小の評価関数値を備える動きベクトルデータを選択する。このレジスタ６８６からのデータに従って所定の優先順位評価基準値算出動作が行なわれ、時刻Ｔａにおいてレジスタ７１１の格納データが確定状態とされる。

【０２４０】時刻Ｔｂにおいてタイミング信号φｔｉｍがローレベルになると、セレクタ７０１がカウンタ６８４の出力するデータを選択する。このカウンタ６８４の出力するデータに従って所定の優先順位評価基準値算出判定動作が行なわれ、その算出された優先順位評価基準値がレジスタ７１２に時刻Ｔｂにおいて格納される。レジスタ７１２の内容が確定すると、時刻Ｔｃにおいて比較器７１３の比較動作が行なわれ、時刻Ｔｃにおいてこの比較器７１３から出力される更新制御信号φｒｗの状態が決定される。時分割多重で優先順位評価基準値を生成することにより、装置規模を低減することができる。

図７６は、タイミング信号発生部の構成を示す図である。図７６に示すように、タイミング信号発生回路７１５は、図６５に示す比較器６８２からの等値指示信号φｅｑに応答して所定の期間活性状態／非活性状態となるタイミング信号φｔｉｍを出力する。このタイミング信号発生回路７１５は、たとえばこの等値指示信号φｅｑに応答して動作可能状態とされる発振回路（たとえばリングオシレータ）で構成される。図７６には、さらにこの等値指示信号φｅｑに応答して優先順位判定回路の各構成要素を活性状態とする活性制御信号φａｃｔを発生する活性制御回路７１６を示す。この活性制御回路７１６により、優先順位判定回路６９０は、優先順位判定動作が必要とされるときのみ動作状態とされ、消費電流を低減する。

【０２４１】なお図７４に示す構成において、優先順位評価基準値を格納するレジスタ７１１および７１２の入力部に設けられるマルチプレクサ７１０を用いずに、レジスタ７１１および７１２へタイミング信号φｔｉｍが直接与えられる構成が利用されてもよい（図６４（Ａ）の構成参照）。以上のように、この発明の第８の実施例の構成に従えば、動きベクトル探索範囲に応じて変位ベクトルの優先順位が変更可能となるように構成したため、垂直方向について動きベクトル探索範囲が拡張された場合においても、変位ベクトル（０，０）に対する優先順位を最も高くして動きベクトル検出動作を行なうことが可能となり、最適な動きベクトルを、拡張された動きベクトル探索範囲においても選択することができる。

【０２４２】［実施例９］図７７は、この発明の第９の実施例である動きベクトル検出装置の全体の構成を概略的に示す図である。この図７７に示す構成においては、フレームを単位として画素を符号化する場合の構成が示

される。図７７において、動きベクトル検出装置は、サーチウィンドウ画素データＰＹとテンプレートブロック画素データＰＸとを受け、与えられたデータＰＹおよびＰＸを、それぞれ所定のタイミングで出力する入力部２と、入力部２から与えられた画素データに基づいて複数の予測モード（本実施例では３つ）に従って複数の評価関数値（差分絶対値）を算出する整数精度演算部１と、この演算部１から与えられた評価関数値に従って複数の予測モードそれぞれに対して最適な動きベクトルを決定する比較部３を含む。整数精度演算部１は、先のたとえば図１等において説明した演算部と同じ構成を備える。この整数精度演算部１は、ブロック動きベクトル予測モード、奇数フィールドブロック予測モード、および偶数フィールドブロック予測モードに従って各サーチウィンドウブロックの変位ベクトルについての評価関数値を算出する。比較部３は、この整数精度演算部１から与えられた各予測モードに対する評価関数値それぞれの最小値に対応する変位ベクトルを予測モードそれぞれの動きベクトルと決定する。

【０２４３】動きベクトル検出装置は、さらに、比較部３における最小値検出に応答して、その最小値の評価関数を与える変位ベクトルにより表現されるサーチエリア内の領域（予測画像領域）を検出し、この予測画像領域を水平および垂直両方向に１画素ずつ拡張された分数精度（１／２画素精度）評価用予測画像領域を格納する分数精度予測画像メモリ７２２と、整数精度演算部１から与えられるテンプレートブロック画素データを格納するテンプレートブロックメモリ７２４を含む。予測画像メモリ７２２は、複数の予測モードそれぞれに対する予測画像領域画素データを格納する。すなわち、予測画像メモリ７２２においては、テンプレートブロックおよびサーチウィンドウブロックのサイズが１６行・１６列の画素であるとき、ブロック予測モードに従って決定された動きベクトルにより指定されるサーチウィンドウブロックを含む１８行・１８列の画素データを格納する領域と、奇数フィールドブロック予測モードおよび偶数フィールドブロック予測モードそれぞれに従って決定された動きベクトルが指定する領域の奇数フィールド画素および偶数フィールド画素それぞれに対応する１６列・８行を囲む１８列・１０行の予測画像領域の画素データを格納する領域を備える。

【０２４４】動きベクトル検出装置は、さらに、テンプレートブロックメモリ７２４に格納されたテンプレートブロック画素データと予測画像メモリ７２２に格納されたブロック予測モードにより決定された予測画像ブロックに含まれる画素データとを受け、分数精度（１／２画素精度）で所定の演算処理を行ない分数精度（１／２画素精度）での各変位ベクトルに対する評価関数値を生成する１／２画素精度演算部７２６と、テンプレートブロックメモリ７２４に格納されたテンプレートブロック画素データと予測画像メモリ７２２に格納された奇数フィールドブロック予測モードに対応する予測画像領域の画素データおよび偶数フィールドブロック予測モードにより決定された予測画像領域に含まれる画素データとを受けてそれぞれ１／２画素精度で所定の演算を行なって評価関数値を生成する１／２画素精度演算部７３８と、１／２画素精度演算部７２６から出力される評価関数値の最小値を求めることによりブロック動きベクトル予測モードによる分数精度での動きベクトルを決定する比較部７３０と、１／２画素精度演算部７３８からの評価関数値に従って最小値を検出し、奇数フィールドブロック予測モードおよび偶数フィールドブロック予測モードそれぞれに対する動きベクトルを検出する比較部７３２を含む。１／２画素精度演算部７３８は、奇数フィールドブロック予測モードによる動き分数精度での動きベクトル検出動作と偶数フィールドブロック予測モードに従う動きベクトル検出動作を時分割的に実行する。この動作については後に詳細に説明する。動きベクトル検出装置は、さらに比較部３から出力される複数の予測モードそれぞれに従って決定された動きベクトルと比較部７３０および７３２から与えられる分数精度での複数の予測モードそれぞれに対応する動きベクトルとを受け、ブロック予測モードに従うブロック動きベクトルＶＢ、奇数フィールドブロック予測モードに従って決定された動きベクトルＶｏ、および偶数フィールドブロック予測モードに従って決定された動きベクトルＶｅを出力する。セレクタ７３４が設けられているのは、比較部７３２から時分割的に奇数フィールドブロック予測モードによる動きベクトルＶｏおよび偶数フィールドブロック予測モードに従う分数精度の動きベクトルが与えられるためである。

【０２４５】図７８に示すように、１／２画素精度での動きベクトル検出時においては、整数精度演算部１により複数の予測モードそれぞれに対して決定された動きベクトルを中心とするその８近傍点の評価関数値を算出する。この整数精度での動きベクトル位置の評価関数値を含む９近傍の評価関数値のうち最小の評価関数値を与えるベクトルを動きベクトルと決定する。すなわち、評価点（－１／２，－１／２）、（０，－１／２）、（＋１／２，－１／２）、（－１／２，０）、（０，０）、（１／２，０）、（－１／２，１／２）、（０，１／２）、および（１／２，１／２）の９評価点を用いて動きベクトル決定動作を行なう。図７９は、この１／２画素精度で動きベクトル検出操作を行なう際に用いられる予測画像領域の画素の分布を示す図である。図７９に示すように、動きベクトルが指定する予測画像領域７４２から正および負の水平方向ならびに正および負の垂直方向それぞれに１画素ずつ拡張された領域７４０に含まれる画素データが用いられる。この領域７４０に格納される画素データ（図７９において○印で示す）から１／２

画素精度での演算精度に用いられるための画素データが生成される（図７９に×印で示す）。１／２画素精度での変位ベクトルの評価関数値算出時においては、図７９に示す×印で表わされる画素データのみが用いられる。この×印で示す画素データとテンプレートブロック画素データとにより図７８に示す変位ベクトルそれぞれに対する評価関数値が生成される。

【０２４６】ここで、領域７４０は、テンプレートブロックのサイズが１６行・１６列の画素を含む場合、１８行・１８列の画素を含む。偶数フィールドブロック予測モードおよび奇数フィールド予測モード時においては、テンプレートブロックは8行・１６列のサイズを備える。したがってこの場合には領域７４０は、１０行・１６列の画素を含む。図８０は、図７７に示す予測画像メモリ７２２の１つの予測画像領域の画素データを格納するメモリの構成を示す図である。以下の説明においては、ブロック予測モードに従って決定された動きベクトルに対応する予測画像領域の画素データを格納する記憶装置の構成および動作について説明する。これは、容易に偶数フィールド予測モードおよび奇数フィールド予測モードの予測画像領域の画素データを格納する記憶装置の構成に適用することができる。

【０２４７】図８０において、予測画像メモリ７２２は、整数精度演算部１から与えられるサーチウィンドウ画素データＰＹを所定時間遅延する遅延回路７２１と、この遅延回路７２１から出力される画素データを格納するメモリ７７２と、メモリ７７２のデータの書込および読出を制御する書込／読出制御回路７７４を含む。メモリ７７２の出力ノードＤＯは、図７７に示す１／２画素精度演算部７２６に接続される。メモリ７７２は、スタティック・ランダム・アクセスメモリであってもよく、またダイナミック・ランダム・アクセスメモリであってもよい。ダイナミック・ランダム・アクセス・メモリの場合、ページモードまたはスタティックコラムモードなどの高速アクセスモードが用いられる。書込／読出制御回路７７４は、メモリ７７２への書込アドレスを発生する書込アドレス発生回路７８１と、メモリ７７２の読出アドレスを発生する読出アドレス発生回路７８３と、このメモリ７７２の読出モードおよび書込モードを指定する信号を発生する制御回路７８６と、この制御回路７８６の制御の下に、書込アドレスおよび読出アドレスの一方を選択してメモリ７７２のアドレス入力ノードＡへ与えるセレクタ７８４を含む。

【０２４８】書込アドレス発生回路７８１は、図７７に示す比較部３の出力するラッチ指示信号Ｒφに応答して書込アドレスを発生する。このラッチ指示信号Ｒφは、比較部３において、そこに格納されている動きベクトル候補の評価関数値およびベクトルの更新が行なわれるときに活性状態とされる（たとえば図６５の比較部に含まれる更新制御回路６８８の更新制御信号に対応する）。このラッチ指示信号Ｒφが活性状態とされたときには、書込アドレス発生回路７８１の書込アドレスは初期値にリセットされる。書込アドレス発生回路７８１は、たとえば０番地から順次書込アドレスをクロック信号φｃｋに応答して発生する。このクロックφｃｋは整数精度演算部１におけるサーチウィンドウ画素データのシフト動作を行なうために用いられる信号である。読出アドレス発生回路７８３も、同様に、０番地から順次読出アドレスを発生する。セレクタ７８４は、制御回路７８６がデータ書込を指示している場合には、書込アドレス発生回路７８１からの書込アドレスを選択してメモリ７７２へ与える。制御回路７８６が、読出モードを指示している場合には、セレクタ７８４は読出アドレス発生回路７８３からの読出アドレスを選択してメモリ７７２へ与える。

【０２４９】制御回路７８６は、サーチウィンドウ画素データ転送クロック信号φｃｋに応答してメモリ７７２へのデータ書込のタイミングおよびデータの読出タイミングを決定する信号を発生する。制御回路７８６は、また、動作モード指示信号φＲＷに応答して、メモリ７７２の書込動作モードおよび読出動作モードを指定し、かつメモリ７７２を選択状態とするチップイネーブル信号（チップセレクト信号）を発生する。制御回路７８６へ与えられる動作モード指示信号φＲＷは、外部に設けられたコントローラから与えられてもよい。また、たとえば比較部３に含まれる変位ベクトル情報生成用のカウンタがカウントアップを行なったときにカウントアップ信号を発生し、このカウントアップ信号を動作モード指定信号φＲＷとして利用してもよい。比較部３に含まれる変位ベクトル情報生成のためのカウンタのカウント値が所定のカウント値に達するまでは１つのテンプレートブロックに対する評価関数値の導出が持続的に実行されているためである。動きベクトル検出動作期間中は、メモリ７７２への書込動作を行ない、１つのテンプレートブロックについての動きベクトルが決定したとき、整数精度演算部においては新たなテンプレートブロック画素データが格納されるため、これと並行して次のテンプレートブロックに対するサーチウィンドウの画素データが格納されるため（１つのテンプレートブロックについての評価関数値のみが生成される場合）、このとき並行してメモリ７７２から画素データを読出す構成とすれば、整数精度演算部への所望のデータロード時にこの予測画像メモリ７２２から予測画像領域のデータを読出すことができる。次に動作について説明する。

【０２５０】今、図８１に示すように、サーチウィンドウが、４８画素行・１６列画素の大きさを備え、マクロブロック（テンプレートブロックおよびサーチウィンドウブロック）が１６画素行・１６画素列の大きさを備える場合を考える。サーチウィンドウブロック７４２について評価関数値算出動作が行なわれているとする。この

サーチウィンドウブロック７４２に対する分数精度の動きベクトルを求めるために必要とされる領域は、この領域７４２を含む１８画素・１８画素の領域７４８である。ここでは、ブロック予測モードに従った動きベクトル検出動作について説明する。領域７４８は、サーチウィンドウ画素データＰ０～Ｐ３２５を含む。図８２に示すように、サーチウィンドウデータ転送用クロック信号φｃｋは、サーチウィンドウ画素データの各転送時に発生される。クロック信号φｃｋが１つ発生されると、サーチウィンドウ画素データが、１画素分シフトアウトされる。サーチウィンドウブロック７４２に対する評価関数値算出動作時においては、図８０に示す遅延回路７２１の出力データは、画素Ｐ０に対応するデータである。ここで、遅延回路７２１は、サーチウィンドウ７４０ａの１列の画素転送に要する時間に等しい遅延時間を与える。サーチウィンドウブロック７４２の評価関数値がそれまでに得られている評価関数値のうちで最小である場合には、比較部からラッチ指示信号Ｒφ（評価関数値更新指示信号）が発生される（活性状態とされる）。このラッチ指示信号Ｒφに応答して、図８０に示す書込アドレス発生回路の書込アドレスが初期値０にリセットされる。このリセットされた初期値０のアドレスに画素データＰ０が書込まれる。以降、信号Ｒφが活性状態とされない限り、連続して１８画素のデータ、すなわちＰ１…Ｐ１７がメモリ７７２のアドレス１～１７の位置に格納される。次に、不要データの書込を禁止するために、書込アドレス発生回路７８１は、３０クロック期間、すなわち、３０φｃｋの期間休止状態となり、メモリ７７２は書込禁止状態となる。この書込禁止の休止期間は、制御回路７８６が、活性状態のラッチ指示信号Ｒφが与えられてから１８回クロック信号φｃｋをカウントした後３０φｃｋ期間メモリ７７２を非選択状態とする構成が利用される。ただしこの間においては、ラッチ指示信号Ｒφは活性状態とされていないものとする。このいずれかの期間においてラッチ指示信号Ｒφが活性状態とされた場合には、先に説明したように、書込アドレス発生回路７８１の書込アドレスは初期値０にリセットされる。

【０２５１】３０クロック期間（３０φｃｋサイクル期間）が経過すると、再び書込アドレス発生回路７８１が書込アドレスを発生する。このときの書込アドレスは、アドレス１８であり、このアドレス１８の位置に画素データＰ１８が格納される。この画素データＰ１８が遅延回路７２１（図８０参照）から与えられる場合には、サーチウィンドウ７４０ａが、次の列のサーチウィンドウ７４０ｂに移行している。テンプレートブロック変更時には、サーチエリアは、１マクロブロックサイズ（１６列）正の水平方向へシフトする。したがって、動きベクトルを与えるマクロブロックがサーチエリアの境界に接している場合においても、分数精度の動きベクトル検出に必要な画素データはすべて得ることができる。この場合、後に説明するようにサーチエリア外部の画素データは無視される構成が利用されてもよい。また、テンプレートブロック変更時においては、同様サーチウィンドウ画素データがシフトアウトされる。このとき、メモリ７７２への書込動作を禁止しておけば、次のテンプレートブロックの最初の変位ベクトル動作検出完了後、遅延回路７２１からは、サーチエリアの１列前の先頭の画素データが出力される。したがって、特に遅延回路７２１においては、その内容をテンプレートブロック更新時にリセットする必要なく、整数精度演算部から出力されるサーチウィンドウ画素データをそのまま利用することができる。

【０２５２】上述の一連の動作により、メモリ７７２には、常に、動きベクトル候補となる変位ベクトルに対応するサーチウィンドウブロックの画素データのみが格納される。奇数フィールド画素データおよび偶数フィールド画素データを格納する場合には、同様の構成を用いて、図８３に示すように、クロック信号信号φｃｋの１サイクルおきにクロック信号φｃｋａおよびφｃｋｂが発生されて、それぞれが奇数フィールド画素データおよび偶数フィールド画素データを書込むためのクロック信号として利用されればよい。偶数フィールド用書込アドレス発生回路と奇数フィールド画素データのための書込アドレス発生回路へは、それぞれ、奇数フィールド予測モードに従って動きベクトル検出動作を行なう比較部からのリセット信号Ｒφが与えられ、また偶数フィールド予測モードに従って動きベクトル検出を行なう比較部からリセット信号Ｒφが与えられる。この構成により、複数の予測モードそれぞれに従って所望の分数精度での動きベクトル検出に必要な画素データを格納することができる。

【０２５３】テンプレートブロックメモリ７２４へは、整数精度演算部１からそのテンプレートブロック画素データ更新時に順次出力されるテンプレートブロック画素データが順次格納される。図８４は、図７７に示す１／２画素精度演算部７２６および７３８の構成の一例を示す図である。図８４においては、１／２画素精度演算部７２６および７３８が同じ構成を備えるため、１／２画素精度演算部７２６の構成のみを示す。図８４において、１／２画素精度演算部７２６は、予測画像メモリ７２２から与えられるサーチウィンドウ画素データを受けて分数精度の評価値算出に必要とされる予測画像を生成する分数精度予測画像生成回路８０２と、分数精度予測画像生成回路８０２で生成された予測画像の画素データとテンプレートブロックメモリ７２４から与えられるテンプレートブロック画素データＰＸとの差分絶対値和を求める差分絶対値和回路８０４と、差分絶対値和回路８０４の出力値のうち最小の差分絶対値和を与える変位ベクトルを検出する比較部８０６を含む。

【０２５４】分数精度予測画像生成回路８０２は、複数

の予測画像データを並列に生成する。この構成については後に詳細に説明する。差分絶対値和回路８０４も、また、動きベクトル候補となる変位ベクトルに対する評価値を並列態様で生成する。比較部８０６は、差分絶対値和回路８０４から与えられる複数の差分絶対値和と図７７に示す比較部３における整数精度での動きベクトルの評価関数値のうち最小の値に対応する変位ベクトルを動きベクトルと決定する。次に、この図８４に示す各回路の具体的構成について説明する。１／２画素精度での動きベクトル検出のためには、図８５に示すように、着目画素Ｐに対し、その８近傍Ｑ１～Ｑ４およびＱ６～Ｑ９の画素データを補間により求める必要がある。サーチウィンドウ画素データをＰ１～Ｐ９とし、この水平方向隣接列画素間の転送期間をＴｈとし、隣接行間における遅延時間をＨｖ（１８Ｔｈ：分数精度におけるサーチウィンドウブロックサイズが１８画素×１８画素の場合）とする。分数１／２画素精度演算部７２６および７３８に対しては、図８５に示す画素Ｐ１～Ｐ９に対応するデータが与えられ、内部で補間データＱ１～Ｑ９が生成される。

【０２５５】図８６は、図７７に示す１／２画素精度演算部の具体的構成を示す。１／２画素精度演算部７２６（または７３８）は、与えられたサーチウィンドウ画素データを１Ｈｖ期間遅延する遅延回路８３５ａと、この遅延回路８３５ａの出力するデータをさらに１Ｈｖ期間遅延する遅延回路８３５ｂを含む。この２段の縦続接続された遅延回路８３５ａおよび８３５ｂにより、図８５に示す各行に対応するデータを発生する経路が形成される。１／２画素精度演算部７２６（または７３８）は、さらに、入力サーチウィンドウ画素データＰを１Ｔｈ期間遅延する遅延回路８３６ａと、この遅延回路８３６ａの出力データをさらに１Ｔｈ期間遅延する遅延回路８３６ｄと、１Ｈｖ遅延回路８３５ａの出力データを１Ｔｈ期間遅延する遅延回路８３６ｂと、遅延回路８３６の出力データを１Ｔｈ期間遅延する遅延回路８３６ｅと、１Ｈｖ遅延回路８３５ｂの出力データを１Ｔｈ期間遅延する遅延回路８３６ｃと、遅延回路８３６ｃの出力データを１Ｔｈ期間さらに遅延する遅延回路８３６ｆを含む。これらの１Ｔｈ遅延回路３６ａ～８３６ｆにより、補間に必要とされるサーチウィンドウ画素データが生成される。

【０２５６】１／２画素精度演算部７２６（または７３８）は、さらに、入力サーチウィンドウ画素データＰと１Ｔｈ遅延回路８３６ａの出力データを加算しかつ係数（１／２）を乗算する加算シフト回路８３０ａを含む。加算シフト回路８３０ａは、係数（１／２）の乗算を、１ビット下位方向への画素データビットのシフト動作により実現する。この１／２画素精度演算部７２６（または７３８）は、さらに、１Ｔｈ遅延回路８３６ａの出力データと１Ｔｈ遅延回路８３６ｅの出力データに対し加算シフト動作を実行する加算シフト回路８３０ｂと、１Ｔｈ遅延回路８３６ｂの出力データと１Ｔｈ遅延回路８３６ｅの出力データとに対し加算シフト動作を実行する加算シフト回路８３６ｃと、１Ｔｈ遅延回路８３５ａの出力データと１Ｔｈ遅延回路８３６ｂの出力データとに対し加算シフト動作を実行する加算シフト回路８３０ｄと、１Ｔｈ遅延回路８３５ｂの出力データと１Ｔｈ遅延回路８３６ｃの出力データとに対し加算シフト動作を実行する加算シフト回路８３０ｅと、１Ｔｈ遅延回路８３５ｃの出力データと１Ｔｈ遅延回路８３６ｆの出力データとに対し加算シフト動作を実行する加算シフト回路８３０ｆとを含む。これらの加算シフト回路８３０ａ～８３０ｆにより、４画素間の補間データを生成するためのデータが生成される。

【０２５７】１／２画素精度演算部７２６（または７３８）は、さらに、加算シフト回路８３０ａの出力データと加算シフト回路８３０ｄの出力データとに対し加算シフト動作を実行する加算シフト回路８３０ｇと、１Ｔｈ遅延回路８３６ａの出力データと１Ｔｈ遅延回路８３６ｂの出力データとに対し加算シフト動作を実行する加算シフト回路８３０ｈと、加算シフト回路８３０ｂの出力データと加算シフト回路８３０ｃの出力データとに対し加算シフト動作を実行する加算シフト回路８３０ｉと、１Ｔｈ遅延回路８３６ｂの出力データと１Ｔｈ遅延回路８３５ａの出力データとに対し加算シフト動作を実行する加算シフト回路８３０ｊと、１Ｔｈ遅延回路８３６ｅの出力データと１Ｔｈ遅延回路８３６ｂの出力データとに対し加算シフト動作を実行する加算シフト回路８３０ｋと、加算シフト回路８３０ｄの出力データと加算シフト回路８３０ｅの出力データとに対し加算シフト動作を実行する加算シフト回路８３０１と、１Ｔｈ遅延回路８３６ｂの出力データと１Ｔｈ遅延回路８３６ｃの出力データとに対し加算シフト動作を実行する加算シフト回路８３０ｍと、加算シフト回路８３０ｃの出力データと加算シフト回路８３０ｆの出力データとに対し加算シフト動作を実行する加算シフト回路８３０ｎとを含む。これら加算シフト回路８３０ｇないし８３０ｎから、図８５に示す補間画素データＱ９～Ｑ６およびＱ４～Ｑ１の位置にある画素データが生成される。

【０２５８】差分絶対値和回路８０４は、加算シフト回路８３０ｇ～８３０１の出力Ｑ９～Ｑ６およびＱ４～Ｑ１とテンプレートブロック画素データＡとを受け、与えられた信号の差分絶対値和を求める差分絶対値和回路８０４ａ～８０４ｈを含む。サーチウィンドウブロック画素データＰとテンプレートブロック画素データＡとは真裏の状態の関係にある。差分絶対値和回路８０４ａ～８０４ｈはそれぞれ変位ベクトル（分数精度での）に対応しており、それぞれ固有のコードが付されている。比較部８０６へ、また整数精度での検出された動きベクトルに対応する評価関数値ｅｖｍが与えられる。比較部８０

６は、この差分絶対値和回路８０４ａ～８０４ｈの出力および評価関数値ｅｖｍを受けて、最小の値を与える差分絶対値和を検出し、この最小の差分絶対値和に対応する変位ベクトルを分数精度での動きベクトルと決定する。

【０２５９】図８６に示す構成において、差分絶対値和回路が、１または４つだけ設けられ、時分割的に活性化されて加算シフト回路８３０ｇ～８３０ｎの出力データが順次加算および累算される構成が利用されてもよい。また、上述の構成においては、１／２画素精度で動きベクトルを検出している。１／４画素精度などのより細かな分数精度の動きベクトルを検出する構成が利用されてもよい。また、評価点は９点であるが、さらに多くの評価点が利用される構成が用いられてもよい。図８７は、この発明の第９の実施例における動きベクトル検出装置の動作シーケンスを示す図であり、横軸に時間を示す。整数精度演算部１および比較部３においてＮ番目のブロック（テンプレートブロック）についての処理を行ない、動きベクトル検出動作が行なわれる。このテンプレートブロックＮの整数精度での検出された動きベクトルに従って、１／２画素精度での動きベクトル検出が行なわれる。テンプレートブロックＮについてのフレームの分数精度での動きベクトル検出動作が行なわれ、これと並行してテンプレートブロックＮ（サブテンプレートブロック）の奇数フィールドについての分数精度での動きベクトル検出が行われる。奇数フィールドの分数精度での動きベクトル検出が完了すると、次いでこのブロックＮの偶数フィールドの動きベクトルが分数精度で検出される。

【０２６０】一方、整数精度演算部１においては、次のテンプレートブロックＮ＋１についての整数精度での動きベクトル検出動作が実行される。このブロックＮ＋１についての動きベクトル検出動作が完了すると、再びブロックＮ＋１についてのフレームおよび奇数フィールドについての分数精度での動きベクトル検出動作が実行される。奇数フィールドの分数精度での動きベクトル検出が完了すると、偶数フィールドについての分数精度での動きベクトル検出動作が実行される。上述のように整数精度演算部１の動作と１／２画素精度演算部７２６および７３８の動作をパイプライン化することにより整数精度演算部１の動作と１／２画素精度演算部７２６および７３８の演算動作を互いに時間的に切離して実行することができ、演算操作に対するタイミング要件に余裕を持って、分数精度での動きベクトル検出を実行することができる。また、この場合においても、整数精度演算部１と、１／２画素精度演算部７２６および７３８とは互いに並列に動作しており、高速で分数精度での動きベクトル検出を行なうことができる。

【０２６１】なお、整数精度演算部１における整数精度での動きベクトル検出動作時においては、サーチウィンドウ画素データがそこから出力される。この場合、１／２画素精度演算部７３８が評価関数値算出動作を行なっているときに、予測画像メモリ７２２のサーチウィンドウ画素データが整数精度演算部１から与えられるサーチウィンドウ画素データにより書替えられることが考えられる。この場合、整数精度演算部１と予測画像メモリ７２２との間にバッファメモリを設けておけば、予測画像メモリ７２２の格納データが１／２画素精度演算部７３８において利用されているときに、整数精度演算部１から与えられるサーチウィンドウ画素データによりこの予測画像メモリ７２２の格納画素データが書替えるられるのが防止され、確実に分数精度での演算操作を行なうことができる。またこのようなバッファを設けておけば、整数精度での演算と分数精度での演算とのタイミング条件を緩和することができ、余裕を持って整数精度の演算および分数精度の演算をそれぞれ並列態様で実行することができる。

【０２６２】なお、上述の実施例においては、予測モードとしてはブロック予測モード、偶数フィールド予測モードおよび奇数フィールド予測モードを想定している。しかしながら、フィールド単位での画素の符号化が行なわれる場合、ブロック予測モード、上半分予測モードおよび下半分予測モードに従って動きベクトルを検出することができ、このような予測モードに対してもこの第９の実施例の構成はそのまま利用することができる。上半分ブロック予測モードおよび下半分ブロック予測モードに対する画素データの予測画像メモリへのデータの書込の期間が変更を受けるだけである。以上のように、この発明の第９の実施例の構成に従えば、整数精度演算部から与えられるテンプレートブロック画素データおよびサーチウィンドウ画素データを用いて分数精度での動きベクトル検出を行なうように構成しているため、複数の予測モードそれぞれに対して分数精度での動きベクトルを行なうことができ、より最適な動きベクトルを選択することができ、精度の高い動きベクトル検出を行なうことができる。

【０２６３】［実施例１０］図８８は、現画面を５×５の２５個のマクロブロック（テンプレートブロック）ＭＢ（０，０）～ＭＢ（４，４）に分割した際のテンプレートブロックの位置を示す図である。テンプレートブロックの位置を、符号ＭＢの下に、左上のテンプレートブロックを原点として水平および垂直方向に正の方向に沿ったベクトルで示す。テンプレートブロックＭＢ（０，０）～ＭＢ（４，４）はそれぞれ１６画素行・１６画素列の構成を備える。動きベクトル検出動作時においては、現画面画像のテンプレートブロックすべてについて動きベクトル検出が行なわれる。各テンプレートブロックについて動きベクトル探索範囲は同じに設定される。この場合、たとえば図８９に示すように、テンプレートブロックＭＢ（０，０）に対しては、その破線で囲む領

域１０には、サーチウィンドウ画素が存在しない。したがってこの領域８１０に含まれる画素を対象とする動きベクトル検出動作は禁止する必要がある。以下にこの発明の第１０の実施例である、動きベクトル探索範囲を限定する方法および構成について説明する。

【０２６４】図９０は、処理単位として利用される探索範囲の構成を示す図である。単位探索範囲は、水平方向１６変位、垂直方向３３変位に設定される。この単位探索範囲における動きベクトル評価関数値算出に必要とされるサイクルは１処理時間Ｔ（３３・１６＝７６８サイクル）である。この単位探索範囲において、全禁止信号Ｉａ１が与えられたとき、その探索範囲における変位ベクトルの評価が禁止される（水平方向および垂直方向いずれについても）。図９１（Ａ）に示すように、上半分禁止信号Ｉｕｐｈが活性状態とされたとき、単位探索範囲８１２における上半分の領域、すなわち垂直方向のベクトル（Ｖベクトル）が－の値（－１～－１６）の領域における変位ベクトル評価が禁止される。また下半分禁止信号Ｉ１ｗｈが活性状態とされたとき、単位探索範囲８１２における下半分の領域８１２１、すなわち垂直方向変位が１～＋１６の正の領域における評価が禁止される。

【０２６５】図９１（Ｂ）に示すように、１／２画素精度の評価時においては、上半分禁止信号Ｉｕｐｈが与えられたとき、その垂直方向のベクトル－１／２を有する変位ベクトルが探索範囲から除外される。同様、下半分禁止信号Ｉｎｗｈが活性状態とされたとき、１／２画素精度動きベクトル評価時において、垂直方向変位ベクトルが＋１／２の領域は評価対象から排除される。図９２に示すように、左半分禁止信号Ｉｌｆｈが与えられたとき、整数精度での動きベクトル評価時においては、この単位探索範囲における変位ベクトルの評価はすべて行なわれる。しかしながら、１／２画素精度での動きベクトル算出時において、整数精度での動きベクトルがこの単位探索範囲８１２の一番左端の領域８１２１に存在する場合、水平方向－１／２のベクトル成分（Ｈベクトル）を有する変位ベクトルの評価が除外される。

【０２６６】垂直成分－１／２を生成するための画素はこの探索範囲領域８１２の外部に存在し、所望の補間画素データを生成することができないためである。同様、図９３に示すように、右半分禁止信号Ｉｒｔｈが活性状態とされたとき、探索範囲８１２において水平方向について左端の領域８１２ｒのベクトルのみを評価対象として動きベクトル検出を行なう。領域８１２ｒのベクトルが最適ベクトルすなわち動きベクトルとして選択された場合、分数精度（１／２画素精度）演算部において水平正方向の分数精度ベクトル（１／２画素精度ベクトル）は評価対象から除外される。５種類の禁止信号Ｉａｌ、Ｉｕｐｈ、Ｉ１ｗｈ、Ｉｌｆｈおよびｉｒｔｈを用いることにより、１つのテンプレートブロックについての動きベクトル探索範囲を正確な画素データのみが存在する領域に設定することができる。次に、動きベクトル探索範囲制限の具体的動作について説明する。

【０２６７】まず、水平方向１６変位、垂直方向３３変位の範囲を単位探索範囲とする。整数精度演算部の構成は第４の実施例の場合と同様とする。すなわち１６行・１６列に配置された要素プロセッサと、各々が３２画素のデータを格納するデータバッファ列とを用いて整数精度での動きベクトル探索動作を行なう。この場合、処理単位時間Ｔは、１６×（１６＋３２）＝７６８サイクルである。今、水平方向探索範囲を上述のごとく－１６～＋１５．５、垂直方向探索範囲を－１６～＋１６とする。この場合、整数精度変位ベクトルは水平方向成分Ｈとして－１６～＋１５を含み、これらの整数精度変位ベクトルに対する評価には、２処理単位時間、２Ｔ必要とされる。また、このとき、分数精度演算部は、１／２画素精度で動きベクトル検出動作を行なうものとする。

【０２６８】今、図８８に示す現画面左上端のテンプレートブロックＭＢ（０，０）に対する動きベクトル検出を行なう場合を考える。この場合、テンプレートブロックＭＢ（０，０）の左側および上側領域には画素は存在しない。したがってこれらの領域に対する動きベクトル評価を以下の様にして排除する。図９４に示すように、最初の処理単位時間Ｔにおいて、水平方向－１６～－１の範囲において動きベクトル探索が行なわれる。この処理時間内においては、図９５（Ａ）に示すように、領域８１５には、画素データが存在しないため、この領域８１５における動きベクトル探索動作は禁止される。すなわち、全禁止信号が発生され、領域８１５における変位ベクトルの評価が禁止される。次の処理単位時間Ｔにおいては、水平方向変位０～＋１５の領域における変位ベクトルの評価が行なわれる。この場合、図９５（Ｂ）に示すように、領域８１６において、垂直方向の変位が－１～－１６の領域８１７ａには画素は存在しない。したがってこの場合にはまず、領域８１７ａにおける変位ベクトルが評価対象から除外される。また、１／２画素精度での動きベクトル検出を行なう場合、領域８１７ｂにおける画素のみを用いる必要がある。したがって、１／２画素精度での予測画像作成時において、水平方向変位が－１／２の変位ベクトルに対しては画素データが存在しないため、この領域に対する分数精度での変位ベクトル評価は禁止される。したがって、このときには、左禁止信号が活性状態とされる。これにより、全探索範囲において、現画面内に存在する画素データのみを用いて動きベクトル検出を行なうことができる。

【０２６９】図９６は、テンプレートブロックＭＢ（０，０）に対する分数精度（１／２画素精度）での動きベクトル探索時の動作を示すフロー図である。まず、整数精度での動きベクトル（Ｈ，Ｖ）が検出される（ステップＳ１）、次いで、ステップＳ２およびＳ３におい

て、この動きベクトル（Ｈ，Ｖ）の水平成分ＨおよびＶが０であるか否かの判別が行なわれる。ステップＳ２において、整数精度動きベクトルの水平成分Ｈが０であると判定されたとき、左禁止信号が与えられているため、ステップＳ４において、分数精度予測時において、水平方向負成分（水平方向－１／２）の変位ベクトルは評価対象から除外される。一方、ステップＳ３において、垂直成分Ｖが０であると判定されたとき、現画面の最上端部のマクロブロックを中心として分数精度予測画像が生成されるため、このとき現画面に含まれない画素を含む垂直方向負成分（－１／２）の変位ベクトルは、評価対象から除外される（ステップＳ５）。水平成分Ｈおよび垂直成分Ｖがともに０でない場合には、整数精度の動きベクトルを中心として分数精度の変位ベクトルすべてが評価対象とされる（ステップＳ６）。

【０２７０】現画面上端のブロックをテンプレートブロックＭＢ（１，０）、ＭＢ（２，０）、およびＭＢ（３，０）に対しては、垂直成分Ｖが負の変位ベクトルに対する評価が禁止される。したがって、図９７に示すように、全評価期間において、上禁止信号のみが活性状態とされる。現画面の上右端のテンプレートブロックＭＢ（４，０）に対しては、図９８（Ａ）に示すように、最初の処理単位時間において、上禁止信号が活性状態とされる。この状態においては、図９８（Ｂ）に示すように、最初の探索領域８１５において、上半分の領域８１７ｃ（垂直方向変位が負の領域）における動きベクトル探索動作が禁止される。次の処理単位時間においては、上禁止信号および右禁止信号がともに活性状態とされる。この状態においては、図９８（Ｃ）に示すように、次の探索領域８１６において、上半分領域８１７ａにおける変位ベクトルの評価が禁止される。下側の領域（垂直成分Ｖが０以上の領域）８１７ｂにおいては、この領域８１７ｂの左端の領域に存在する変位ベクトルに対してのみ評価が行なわれる。すなわち、Ｈ＝０かつＶ≧０の変位ベクトル（Ｈ，Ｖ）に対する評価動作のみが行なわれる。これにより、テンプレートブロックＭＢ（４，０）に対する整数精度での、画面画像画素データのみを用いた動きベクトルが決定される。次いで、この決定された整数精度の動きベクトルに従って、分数精度での動きベクトルが決定される。この分数精度での動きベクトル検出動作は、図９９に示す処理ステップを含む。

【０２７１】図９９のステップＳ２０～Ｓ２２に示すように、最初の処理単位時間において、垂直方向の変位ベクトルを除外して動きベクトル探索が行なわれ、次の処理単位時間において、水平成分Ｈが０でありかつ垂直成分Ｖが０以上の変位ベクトルに対する評価が行なわれる。このステップＳ２０およびＳ２２により、整数精度での動きベクトル（Ｈ，Ｖ）が決定される（Ｓ２２）。次いで分数精度での動きベクトル検出が行なわれる。ステップＳ２３およびＳ２４において、この整数精度の動きベクトル（Ｈ，Ｖ）の水平成分Ｈおよび垂直成分Ｖが０に等しいか否かの判定が行なわれる。ステップＳ２３において、水平成分Ｈが０であると判定された場合、現画面右端に位置するサーチウィンドウブロックを予測画像として用いる必要がある。この場合、正の分数精度の水平成分すなわち、水平成分＋１／２の変位ベクトルを除外して分数精度での動きベクトルの探索が行なわれる（ステップＳ２５）。一方、垂直成分Ｖが０であると判定された場合、Ｖ＝－１の動きベクトルに対応する画素は現画面に含まれていないため、分数精度で負の垂直成分（－１／２）の変位ベクトルを除外して分数精度での動きベクトルの探索が行なわれる。水平成分Ｈおよび垂直成分Ｖがともに０の場合には、ステップＳ２５およびＳ２６の動作がともに行なわれる。整数精度動きベクトル（Ｈ，Ｖ）の水平成分Ｈおよび垂直成分Ｖがともに０と異なる場合には、整数精度の動きベクトル（Ｈ，Ｖ）を中心として、分数精度の全探索範囲において動きベクトルの探索が行なわれる（ステップＳ２７）。

【０２７２】現画面左端のテンプレートブロックＭＢ（０，１）、ＭＢ（０，２）、およびＭＢ（０，３）に対しては、最初の処理単位時間においては、全禁止信号が活性状態とされ、水平方向変位－１６～－１の領域における変位ベクトルは評価対象から除外される。次の処理単位時間においては、左禁止信号が活性状態とされる。整数精度での動きベクトル探索時においては、水平方向変位０～＋１５におけるすべての変位ベクトルに対する評価が行なわれる。分数精度での動きベクトル検出時の動作時には、図９６に示すステップＳ２、Ｓ４、およびＳ６が行なわれる。現画面の中央部のテンプレートブロックＭＢ（ｉ，ｊ（ｉ≠０，ｊ≠０））に対しては、図１０１に示すように、２処理単位時間の間、探索範囲におけるすべての変位ベクトルに対する評価が行なわれる。

【０２７３】現画面の右端のテンプレートブロックＭＢ（４，１）、ＭＢ（４，２）、およびＭＢ（４，３）に対しては、図１０２に示すように、最初の処理単位時間Ｔにおいては、探索範囲のすべての変位ベクトルに対する評価関数値が生成される。次の処理単位時間においては、右禁止信号のみが活性状態とされる。すなわち、水平方向変位０の変位ベクトルのみが評価対象とされて動きベクトル探索が行なわれる。この右端のテンプレートブロックに分数精度での動きベクトル探索動作では、図９９に示すステップＳ２３、Ｓ２５およびＳ２７の処理が実行される。現画面の下左端のテンプレートブロックＭＢ（０，４）に対しては、図１０３に示すように、最初の処理単位時間においては、全禁止信号が活性状態とされる。この状態においては、図１０３（Ｂ）に示すように、最初の探索領域８１５における変位ベクトルがすべて評価対象から除外される。

【０２７４】次の処理単位時間においては、下禁止信号

および左禁止信号がともに活性状態とされる。この状態においては、図１０３（Ｃ）に示すように、探索領域８１６における下半分の領域（Ｂ＝１～１６）８１７ｄにおける変位ベクトルが評価対象から除外される。すなわち、領域８１７ｅにおける変位ベクトルのみを用いてテンプレートブロックＭＢ（０，４）の動きベクトル探索が行なわれる。図１０４は、このテンプレートブロックＭＢ（０，４）の分数精度での動きベクトル探索動作を示すフロー図である。すなわち、図１０４のステップＳ３０に示すように、水平成分０以上および垂直成分０以下の範囲の領域で整数精度での動きベクトル探索が行なわれ、その探索結果に従って整数精度での動きベクトルが決定される（ステップＳ３１）。この整数精度で決定された動きベクトル（Ｈ，Ｖ）に従って、水平成分Ｈおよび垂直成分Ｖが０であるか否かの判定が行なわれる（ステップＳ３２およびＳ３３）。ステップＳ３２において、水平成分Ｈが０であると判定されたとき、水平方向の分数精度で負のベクトルをすべて除外して変位ベクトルの評価が行なわれる（ステップＳ３４）。一方、垂直成分Ｖが０であるとステップＳ３３において判定されたとき、分数精度で、垂直方向正の変位ベクトルを除外して、分数精度で変位ベクトルの評価が行なわれる（ステップＳ３５）。整数精度の動きベクトル（Ｈ，Ｖ）の水平成分Ｈおよび垂直成分Ｖがともに０でないと判定された場合には、この分数精度での探索範囲内のすべての変位ベクトルがすべて評価対象とされる（ステップＳ３６）。

【０２７５】現画面下端のテンプレートブロックＭＢ（１，４）、ＭＢ（２，４）、およびＭＢ（３，４）に対しては図１０５に示すように、処理時間全体にわたって下禁止信号が活性状態とされる。この下禁止信号活性時における整数精度ゼロ動きベクトル探索時においては垂直成分Ｖが０以下の範囲で動きベクトル探索が行なわれる。分数精度での動きベクトル探索時においては、図１０４に示すステップＳ３３、Ｓ３５およびＳ３６に示す処理動作が実行される。現画面下右端のテンプレートブロックＭＢ（４，４）に対しては、図１０６（Ａ）に示すように、最初の処理単位時間Ｔにおいては下禁止信号が活性状態とされる。この状態においては、図１０６（Ｂ）に示すように最小探索領域８１５における下半分の領域（Ｂ＝１～１６）８１７ｆにおける変位ベクトルが評価対象から除外される。次の処理単位時間においては、図１０６（Ａ）に示すように、下禁止信号および右禁止信号がともに活性状態とされる。この状態においては、図１０６（Ｃ）に示すように、後半の探索領域８１６における水平成分Ｈが０の領域８１７ｇに含まれる変位ベクトルのみが評価対象とされる。下禁止信号活性化時および右禁止信号活性化時における分数精度での動きベクトル検出時の動作は、先ＮＯテップＳ３３、Ｓ３５およびＳ３６の処理動作、ならびにステップＳ２３、Ｓ２５およびＳ２７の処理動作と同じである。

【０２７６】上述のような処理されるべきテンプレートブロックの位置に応じて禁止信号を選択的に活性状態とすることにより現画面画像に含まれる画素データのみを用いて動きベクトルを決定することができ、精度の高い動きベクトルを検出することができ、正確な予測画像を生成することができる。なお上述の説明においては、明確に示していないが、整数精度による動きベクトル検出時においては、複数の予測モードそれぞれに従って動きベクトルが決定される。したがって、分数精度での動きベクトル検出時においても、複数の予測モードそれぞれに従って決定された動きベクトルの水平成分および垂直成分の値とそのときの活性状態とされている禁止信号の組合わせに従って、複数の予測ベクトルそれぞれに対して分数精度での動きベクトル探索範囲が決定される。

【０２７７】図１０７は、この発明の第１０の実施例に従う動きベクトル検出装置の全体の構成を概略的に示す図である。図１０７において、動きベクトル検出装置は、参照画像画素データを格納する参照画像メモリ８２０と、現画面画像画素データを格納する現画像メモリ８２１と、参照画像メモリ８２０からのサーチウィンドウ画素データＰＹおよび現画像メモリ８２１からテンプレートブロック画素データＰＸを受けて整数精度で動きベクトルを検出する整数精度動きベクトル検出部８３０と、整数精度動きベクトル検出部８３０から与えられるサーチウィンドウ画素データＰＹ、テンプレート画素データＰＸ、整数精度動きベクトルデータＭＶＩ、およびこの整数精度動きベクトルＭＶＩの評価関数値ｅｖｍを受けて分数精度で動きベクトルを検出する分数精度動きベクトル検出部８３２を含む。図１０７においては、図面を簡略化するために、１つの予測モードに従って決定される動きベクトルのみを代表的に示す。複数の予測モードそれぞれに従って整数精度および分数精度での動きベクトルが検出される。分数精度動きベクトル検出部８３２からの分数精度動きベクトルはベクトルＭＶＦで代表的に示される。

【０２７８】動きベクトル検出装置は、さらに、参照画像メモリ８２０および現画像メモリ８２１の読出アドレスを発生するとともにこれらのメモリ８２０および８２１における画素データ読出動作を制御する読出制御回路８３８と、この読出制御回路８３８に含まれる現画像メモリ８２１に対する読出アドレス、すなわちテンプレートブロック画素位置情報を受けて動きベクトル探索範囲を設定する探索範囲設定部８３４と、探索範囲設定部８３４からの動きベクトル探索範囲情報に従って整数精度動きベクトル検出部８３０および分数精度動きベクトル検出部８３２における動きベクトル探索動作、すなわち変位ベクトル評価動作を制御する探索動作制御部８３６を含む。この探索動作制御部８３６は、探索範囲設定部８３４からの探索範囲情報に従って整数精度動きベクト

ル検出部８３０における変位ベクトル評価動作を制御する。この探索動作制御部８３６は、また、整数精度動きベクトル検出部８３０からの整数精度動きベクトル情報ＭＶＩと探索範囲設定部８３４からの探索範囲設定情報すなわち禁止信号とに従って分数精度動きベクトル検出部８３２における変位ベクトル評価動作を制御する。

【０２７９】図１０８は、図１０７に示す探索動作制御部８３６に含まれる整数精度動きベクトル選出部の評価動作制御部の構成を示す図である。図１０８においては、整数精度動きベクトル検出部８３０に含まれる比較部３の構成を併せて示す。整数精度動きベクトル検出部８３０に含まれる比較部３は、評価関数値を格納するレジスタ６８０と、演算部から与えられた評価関数値ｅｖｍｉとレジスタ６８０の格納する評価関数値とを比較する比較器６８２と、演算部から与えられる評価関数値に関連する変位ベクトル情報を発生するカウンタ６８４と、最小の評価関数値を与える変位ベクトル情報を格納するレジスタ６８６を含む。このレジスタ６８０、比較器６８２、カウンタ６８４およびレジスタ６８６の構成は、図６５に示す比較部の構成とほぼ同様である。ただ、図面の簡略化のため、図１０８においては図６５に示す優先順位判定回路６９０を示していない。更新制御回路８３９は、比較器６８２からの更新指示信号が与えられると、レジスタ６８０および６８６の内容を更新する。この更新制御回路８３９は、また、整数精度探索動作制御部８４０からの禁止信号Ｉａの活性化時、比較器６８２の出力信号にかかわらずレジスタ６８０および６８６の更新動作を禁止する。

【０２８０】整数精度探索動作制御部８４０は、カウンタ６８４からの変位ベクトル情報ｄｅｐｖを受けて水平成分Ｈおよび垂直成分Ｖを抽出するＨＶ抽出回路８４１と、上半分禁止信号Ｉｕｐｈの活性化時に作動状態とされて、ＨＶ抽出回路８４１から与えられる垂直成分Ｖが負のときに活性状態の信号を出力する上半分検出器８４２と、下半分禁止信号Ｉｌｗｈの活性化時に作動状態とされて、ＨＶ抽出回路８４１からの垂直成分Ｖが正の領域にあるときに活性状態の信号を出力する下半分検出器８４３と、右半分禁止信号Ｉｒｈｔの活性化時にＨＶ抽出回路８４１から最初に出力される水平成分Ｈデータをラッチするラッチ回路８４４と、右半分禁止信号Ｉｒｈｔの活性化時に作動状態とされて、ＨＶ抽出回路８４１の出力する水平成分Ｈとラッチ回路８４４がラッチする水平成分とが一致したときに活性状態の信号を出力する左端検出器８４５を含む。全禁止信号Ｉａｌ、上半分検出器８４２の出力、下半分検出器８４３の出力信号、および左端検出器８４５の出力信号はたとえばワイヤードＯＲ接続されて禁止信号Ｉｈを生成して更新制御回路８３９へ与えられる。ワイヤードＯＲ接続に替えて、信号Ｉａｌ、右半分検出器８４２の出力信号、下半分検出器８４３の出力信号、および左端検出器８４５の出力信号を受けるＯＲゲートにより、禁止信号Ｉｈが出力される構成が利用されてもよい。

【０２８１】全禁止信号Ｉａｌが活性状態のときには、更新制御回路８３９が、更新指示信号を常時非活性状態とし、レジスタ６８０および６８６における内容更新を禁止する。上半分検出器８４２は、カウンタ６８４から出力される変位ベクトル情報ｄｅｐｖが垂直方向負の変位ベクトルを示しかつ上半分禁止信号Ｉｕｐｈが活性状態のときに活性状態の信号を出力する。したがって上半分禁止信号Ｉｕｐｈが活性状態のとき負の垂直成分を有する変位ベクトルの評価関数値はその比較器６８２の出力信号にかかわらずレジスタ６８０に格納されず、またレジスタ８６６の変位ベクトルの更新は行なわれない。下半分禁止信号Ｉｌｗｈが活性状態のとき、下半分検出器８４３は、カウンタ６８４から出力される変位ベクトル情報ｄｅｐｖが正の垂直成分を有するとき活性状態の信号を出力する。これにより、禁止信号Ｉｈが活性状態とされ、レジスタ６８０および６８６の更新が禁止される。ラッチ回路８４４は、右半分禁止信号Ｉｒｈｔが活性状態のとき、その処理単位時間の最初に発生されるカウンタ６８４の出力する変位ベクトル情報の水平成分をラッチする。これにより、その右半分禁止信号Ｉｒｈｔが活性状態とされる処理単位時間における負のベクトル探索範囲における左端の領域、すなわちその処理単位時間おける水平方向の最初の水平成分情報がラッチされる。左端検出器８４５は、活性化時、カウンタ６８４が出力する変位ベクトル情報ｄｅｐｖの水平成分がラッチ回路８４４に格納された水平成分と異なるときに、活性状態の信号を出力する。これより処理単位時間における探索範囲における左端の列に位置する変位ベクトルに対してのみ評価動作が行なわれる。

【０２８２】上述の構成により整数精度での動きベクトル検出時において動きベクトル探索範囲を禁止信号に従って決定することができる。この整数精度探索動作制御部８４０へ左半分禁止信号Ｉｌｆｔが与えられていないのは、この信号が与えられたとき、水平方向全範囲において変位ベクトルの評価動作が行なわれるためである。また、図１０８に示す構成において、禁止信号Ｉｈは比較器６８０に与えられて比較器の比較動作が禁止される構成が利用されてもよい。比較動作が禁止された比較器６８２は、更新指示信号を非活性状態に維持する。図１０９は、図１０７に示す探索動作制御部８３６に含まれる分数精度探索動作制御部の構成を概略的に示す図である。図１０９において、分数精度探索動作制御部８５０は、図１０８に示す比較部３から出力される整数精度の動きベクトル情報ＭＶＩから水平成分Ｈおよび垂直成分Ｖを抽出するＨＶ抽出器８５１と、ＨＶ抽出器８５１からの垂直成分Ｖが０のときに活性状態の信号を出力するＶ０検出器８５２と、上半分禁止信号ＩｕｐｈとＶ０検出器８５２の出力信号がともに活性状態のときに、分数

精度での垂直方向負成分の変位ベクトルの評価を禁止する垂直負成分禁止信号ＩＶＮＦを活性状態とするゲート回路８５３と、下半分禁止信号ＩｌｗｈおよびＶ０検出器８５２の出力信号がともに活性状態のときに分数精度での垂直方向正の変位ベクトルの評価を禁止する垂直正成分禁止信号ＩＶＰＦを活性状態とするゲート回路８５４と、左半分禁止信号Ｉｌｆｔの活性化時、その処理単位時間の最初にカウンタ６８４（図１０８参照）から与えられる変位ベクトルの水平成分ｄｅｐｖ（Ｈ）をラッチするラッチ回路８５５と、左半分禁止信号Ｉｌｆｔの活性化時に活性化されてラッチ回路８５５のラッチ水平成分とＨＶ抽出器８５１から出力される動きベクトルの水平成分Ｈが等しいときに分数精度での水平方向負の変位ベクトルの評価を禁止する水平負成分禁止信号ＩＨＮＦを活性状態とする一致回路８５６と、右半分禁止信号Ｉｒｈｔの活性化時、その処理単位時間の最初に与えられた変位ベクトルの水平成分ｄｅｐｖ（Ｈ）をラッチするラッチ回路８５７と、右半分禁止信号Ｉｒｈｔの活性化時に活性化され、ラッチ回路８５７がラッチする水平成分データとＨＶ抽出器８５１からの整数精度動きベクトルＭＶＩの水平成分が一致したときに分数精度での正の変位ベクトルの評価を禁止する水平正成分禁止信号ＩＨＰＦを活性状態とする一致回路８５８を含む。

【０２８３】ゲート回路８５３および８５４は、それぞれ、その両入力に与えられた信号が活性状態のとき活性状態の出力信号を生成するたとえばＡＮＤゲートで構成することができる（活性状態がハイレベルのとき）。一致回路８５６および８５８は、たとえば、ＥＸＮＯＲゲートおよびＡＮＤゲートを用いて構成することができる（ＡＮＤゲートを用いるのは、水平成分データは多ビット信号であるため各ＥＸＮＯＲゲートの出力信号の論理積をとるために用いる。ＨＶ抽出器８５１の構成は、この動きベクトル情報ＭＶＩの構成により種々の構成が可能である。整数精度動きベクトル情報ＭＶＩが通常の２進カウント値の場合には、そのカウント値をデコードする回路が利用されればよい。また、この整数精度動きベクトル情報ＭＶＩが水平成分および垂直成分それぞれを備える場合には、その水平成分および垂直成分がそのまま利用されればよい。

【０２８４】また、全禁止信号Ｉａｌは、分数精度動きベクトル検出部の全変位ベクトルに対する評価動作を禁止する信号ＩＡＬＦとして利用される。この図１０９に示す禁止信号ＩＡＬＦ、ＩＶＮＦ、等は、図１０７に示す分数精度動きベクトル検出部８３２に含まれる比較部へ与えられる。この比較部の構成に応じて禁止信号の印加態様は種々の修正を受ける。１／２画素精度での変位ベクトル評価が、各変位ベクトルに対し逐次実行される場合には、図１０８に示す構成と同様の構成により分数精度での動きベクトル検出範囲を制御することができる。図１１０は、分数精度動きベクトル検出部８３２の要部の構成の一例を示す図である。この図１１０に示す分数精度動きベクトル検出装置は、図８６に示す分水精度（１／２画素程度）動きベクトル検出装置の構成に対応する。この図１１０に示す分水精度動きベクトル検出装置は、８近傍の評価点の評価関数値を生成する差分絶対値和回路８０４ａ～８０４ｇと、これら差分絶対値和回路８０４ａ～８０４ｇおよび整数精度動きベクトルの評価関数値ｅｖｍから最小の評価関数値を検出し、その最小の評価関数値に対応する変位ベクトルを示す情報を分数精度動きベクトルＭＶＦとして出力する比較部８０６を含む。差分絶対値和回路８０４ａ～８０４ｇはそれぞれ分数精度の８近傍の変位ベクトル（評価点）に一意的に対応づけられる。差分絶対値和回路８０４ａは変位ベクトル（－１／２，－１／２）に対応し、差分絶対値和回路８０４ｂは変位ベクトル（０，－１／２）に対応し、差分絶対値和回路８０４ｃは変位ベクトル（１／２，－１／２）に対応する。

【０２８５】差分絶対値和回路８０４ｄは、変位ベクトル（－１／２，０）に対応し、差分絶対値和回路８０４ｅは、変位ベクトル（１／２，０）に対応し、差分絶対値和回路８０４ｆは、変位ベクトル（－１／２，１／２）に対応し、差分絶対値和回路８０４ｇは、変位ベクトル（０，１／２）に対応する。評価関数値ｅｖｍは変位ベル（０，０）に対応する。差分絶対値和回路８０４ａには、禁止信号ＩＨＮＦ、ＩＶＮＦおよびＩＡＬＦが与えられる。差分絶対値和回路８０４ｂには、禁止信号ＩＶＮＦおよびＩＡＬＦが与えられる。差分絶対値和回路８０４ｃには、禁止信号ＩＨＰＦ、ＩＶＮＦおよびＩＡＬＦが与えられる。差分絶対値和回路８０４ｄへは、禁止信号ＩＨＮＦおよびＩＡＬＦが与えられる。差分絶対値和回路８０４ｅへは、禁止信号ＩＨＰＦおよびＩＡＬＦが与えられる。差分絶対値和回路８０４ｆへは、禁止信号ＩＨＮＦ、ＩＶＰＦ、およびＩＡＬＦが与えられる。差分絶対値和回路８０４ｇは、禁止信号ＩＨＮＦおよびＩＡＬＦが与えられる。これら差分絶対値和回路８０４ａ～８０４ｇは、与えられた禁止信号が活性状態となったときに、その出力値を最大値に設定する。すなわち、たとえば差分絶対値和回路８０４ａ～８０４ｇそれぞれの出力データのビット数がＮビットのとき、差分絶対和回路８０４ａ～８０４ｇは、対応の禁止信号の少なくとも１つが活性状態とされたとき、そのＮビットの出力をすべて“１”に設定する。これにより、比較部８０６において禁止信号が活性状態とされた領域に含まれる変位ベクトルは非選択状態とされ、動きベクトルの評価対象から排除される。禁止信号が活性状態とされたときに差分絶対値和回路８０４（８０４ａ～８０４ｇ）の出力データを最大値に設定する構成は、たとえば禁止信号と差分絶対値和回路８０４の出力ビットのＯＲ演算を行なうゲートが各出力ビットに対して設けられればよい。ただし、禁止信号の活性状態時はハイレベルの信号であ

り、ビット“１”をハイレベルの信号に対応させる。

【０２８６】上述の構成において、分数精度での変位ベクトルの評価対象領域を容易に設定することができる。この図１０８ないし図１１０に示す構成は、複数の予測モードそれぞれに対応して配置される。図１１１は、図１０７に示す探索範囲設定回路８３４および読出制御回路８３８の構成を概略的に示す図である。図１１１において、読出制御回路８３８は、現画面におけるテンプレートブロックの位置を示す情報を出力するカウンタ８６２と、動きベクトル探索時間を、処理単位時間を単位として設定する処理単位設定回路８６３と、現画像メモリ８２１の読出アドレスを発生するリードアドレス発生器８６０と、処理単位設定回路８６３により設定された処理時間情報に従ってカウンタ８６２の更新およびリードアドレス発生器８６０のアドレス更新および現画像メモリ８２１のデータ読出動作を制御するリード制御器８６１を含む。

【０２８７】リードアドレス発生器８６０は、カウンタ８６２のカウント値を、先頭アドレス情報に変換し、その先頭アドレスから順次現画像メモリ８２１の読出アドレスを発生する。通常、現画像メモリ８２１は、行および列のマトリクス状に配列されたメモリセルを含む。各行が画面上の水平方向１行の画素に対応するように現画像メモリ８２１が画素データを格納する構成の場合、リードアドレス発生器８６０は、リード制御器８６１の制御のもとに、１行において１６画素のデータを読出した後、次の行の同じ列に配置された画素データを読出す。すなわち、カウンタ８６２のテンプレートブロック情報に従って生成された先頭アドレスが指定する行から１６画素のデータを読出した後、次の行において先頭アドレスと同じ列から始まって１６画素のデータを読出す。このアドレス変換機能をまたリードアドレス発生器８６０は備える。このアドレス発生器の構成は、たとえば、先頭アドレス（行および列アドレスを含む）において先頭行において１６画素のデータを読出した後、再び先頭アドレスが指定する列から１６画素のデータを読出す構成が利用されればよい。リード制御器８６１は、処理単位設定回路８６３の設定する処理時間情報に従って、カウンタ８６２のカウント値の更新を行なうとともに、現画像メモリ８２１を、新しいテンプレートブロックについての動きベクトル検出動作時においてデータ読出状態に設定する。図示しないが、外部から処理時間情報が与えられるかまたは、使用用途に応じて、この処理単位設定回路８６３の格納処理時間情報が固定的に設定される。

【０２８８】探索範囲設定回路８３４は、カウンタ８６２からのカウント値を受けて処理を受けるテンプレートブロックの位置を検出するテンプレートブロック位置検出器８７１と、処理単位設定回路８６３に設定された処理時間とサーチウィンドウ画素データ転送クロック信号φとに従って、処理単位時間を単位として、実行されているサイクルがいずれのサイクルであるかを検出するサイクル検出回路８７２と、テンプレートブロック位置検出器８７１の検出したテンプレートブロック位置情報とサイクル検出回路８７２の検出したサイクル情報とに従って全禁止信号Ｉａｌ、上半分禁止信号Ｉｕｐｈなどの禁止信号を発生する禁止信号発生器８７３を含む。この禁止信号発生器８７３は、先に図９２ないし図１０６に示したシーケンスに従って各禁止信号を発生状態とする。たとえば、テンプレートブロック位置検出器８７１が、画面左上端のテンプレートブロックＭＢ（０，０）を検出したとき、禁止信号発生器８７３は、第１サイクルにおいては、全禁止信号Ｉａｌを活性状態とし、次の処理サイクルにおいては、上半分禁止信号Ｉｕｐｈおよび左半分禁止信号Ｉｌｗｈを活性状態とする。

【０２８９】探索範囲設定回路８３４はさらに、動きベクトル探索範囲の制限を受けるべきテンプレートブロックを指定する情報を生成するテンプレートブロック指定回路８７４と、テンプレートブロック指定回路８７４が指定するテンプレートブロックと、テンプレートブロック位置検出器８７１の指定するテンプレートブロック位置情報とが一致するか否かを判断する一致検出器８７６と、変位ベクトルの評価動作が禁止されるべき領域を設定する禁止区域設定回路８７５と、一致検出器８７６からの一致検出信号と禁止区域設定回路８７５からの禁止区域情報とに従って、禁止信号を発生する禁止信号発生器８７７と、禁止信号発生器８７３および８７３の出力の一方をモード指定信号φＳＭＬに従って選択するセレクタ８７８を含む。このセレクタ８７８から探索動作制御部８３６（図１０７）に対し禁止信号が与えられる。テンプレートブロック指定回路８７４および禁止区域設定回路８７５へは、図示しない外部コントローラからそれぞれ所定の情報が設定される。それにより、所望のテンプレートブロックに対してのみ動きベクトル探索範囲を制限することができる。禁止信号発生器８７７は、一致検出器８７６からの一致信号が活性状態のとき、この禁止区域設定回路８７５が設定された禁止区域における動きベクトル探索動作（変位ベクトル評価動作）を禁止するように禁止信号を活性状態とする。デフォルトモードとしては、禁止信号発生器８７３からの禁止情報がセレクタ８７８により選択される。この画面上のテンプレートブロックの位置に従って決定される動きベクトル探索範囲と異なる探索範囲を設定する場合には、モード指定信号φＳＭＬにより、セレクタ８７８を禁止信号発生器８７７の出力する禁止信号を選択する状態に設定する。このテンプレートブロック指定回路８７４、禁止区域設定回路８７５、一致検出器８７６および禁止信号発生器８７７を設けることにより、処理モードに応じて動きベクトル探索範囲を設定することができ、処理されるべき画像の構成に応じた動きベクトル探索範囲を設定することが可能となる。

【０２９０】たとえば、図１１２（Ａ）に示すように、画面８８０の中央部に位置するテンプレートブロック８８１に対し、図１１２（Ｂ）に示すように、探索範囲８８２における４つの領域８８２ａ、８８２ｂ、８８２ｃ、および８８２ｄの領域における動きベクトル探索を禁止することができる。たとえば画面が一方方向に移動している場合、動きベクトルとしては、その移動方向と反対方向にのみ動きベクトルが検出される確率が高い。この場合、その他方方向に対してのみ動きベクトル探索を行なうことにより、より相関度の高い動きベクトルを検出することができる。また、Ｉピクチャーのようなフレーム間予測を行なう画像と異なり、Ｐピクチャーのように、フレーム内予測を行なう画像が挿入されるとき、フレーム間予測で動きベクトルを検出する場合、このＰピクチャーにおけるフレーム内予測モードで形成された画面領域を用いた動きベクトルの精度はフレーム間予測の場合に比べて劣化するため、このような領域を用いたフレーム間予測による動きベクトル検出動作を禁止することにより、精度の高い動きベクトルを検出することができる。

【０２９１】以上のように、この発明の第１０の実施例に従えば、テンプレートブロックに対し、その動きベクトル探索範囲を制限することができるように構成したため、より精度の高い動きベクトル検出を行なうことができる。特に分数精度の動きベクトル検出動作に対してもそのベクトル探索範囲を制限するように構成したため、実際に存在する画素のデータのみを用いて動きベクトルを検出することができ、より精度の高い動きベクトルを検出することができ、高精度の画像符号化システムを構築することができる。

［入力部の構成］図１１３は、動きベクトル検出装置の入力部の構成を概略的に示す図である。図１１３において、動きベクトル検出装置２００は、現画面画像データを格納する現画像メモリ２０４から与えられるテンプレートブロックデータと参照画面画像データを格納する参照画像メモリ２０２から読出されたサーチエリア画素データとに従って所定の演算を行なって動きベクトルを検出する。現画像メモリ２０４からは、動きベクトル検出の１サイクルごとにテンプレートブロック画素データが読出されて演算部１へ与えられる。この場合、演算部１に含まれる要素プロセッサは直線状に実質的に配列されているため、前のサイクルで処理されたテンプレートブロックに隣接する列にありかつ同一行に配列された画素データを所定数読出すことにより次のテンプレートブロックの画素データを演算部１のプロセッサアレイ内に配置させることができる。

【０２９２】参照画像メモリ２０２は、サーチエリア内の画素データを順次読出して入力部２へ与える。この参照画像メモリ２０２に格納されるデータは、図１５２に示す加算回路９３４の出力によりその記憶内容が変更される。

【０２９３】入力部２は、参照画像メモリ２０２からのサーチエリア画素データを格納するためのサーチウィンドウバッファメモリ２０５と、このサーチウィンドウバッファメモリ２０５に格納された画像データの画面形式を指定された画面形式に変換するための画面形式変換回路２０７を含む。サーチウィンドウバッファメモリ２０５は、例えばサーチエリア領域における演算部１のプロセッサアレイに格納されたサーチウィンドウに１列隣接するサーチウィンドウの画素データを格納する。参照画像メモリ２０２から読出される画像データの画面形式はフレーム形式またはフィールド形式のいずれかである。この場合、動きベクトル検出装置２００の予測モードがフレーム形式に対応する構成の場合、この内部の予測モードに合わせて処理すべき画素データの画面形式を変更する。外部処理装置がこの動きベクトル検出装置の予測モードに合わせて画素データ列を変換して画面形式を変換する必要がなく、外部処理装置の負荷が軽減されるとともに、画面形式変換に必要とされる回路を外部に設ける必要がなく、外部のハードウェア量を削減することができる。また処理対象となる画像データの画面形式がフィールド形式およびフレーム形式いずれであってもフレームを単位として動き検出を実行することができ、汎用性の高い動きベクトル検出装置を実現することができる。

【０２９４】図１１４は図１１３に示す画面形式変換回路の構成を概略的に示す図である。図１１４において、画面形式変換回路２０７は、入力画像信号の画面形式がフレーム形式の場合に書込および読出アドレスを発生するフレーム／フレーム用アドレス生成器２１３と、入力画像データの画面形式がフィールド形式の場合に書込および読出アドレスを発生するフィールド／フレーム用アドレス生成器２１５と、画面形式指定信号φＰＴに従ってアドレス生成器２１３および２１５の出力の一方を選択してサーチウィンドウバッファメモリ２０５へ与えるセレクタ２１１を含む。

【０２９５】図１１５は、図１１４に示すフレーム／フレーム用アドレス生成器の構成の一例を示す図である。図１１５においては、サーチウィンドウバッファメモリが４８行×１４列の構成を備える場合のアドレス発生のための構成が一例として示される。サーチウィンドウバッファメモリ２０５の記憶容量に合わせてこの図１１５に示す構成が適宜修正される。

【０２９６】図１１５において、フレーム／フレーム用アドレス生成器２１３は、画像データ書込時に書込アドレスを発生するライトアドレス発生器２２０と、このバッファメモリ２０５からデータを読出して演算部１へ与えるための読出アドレスを発生するためのリードアドレス発生器２２２を含む。ライトアドレス発生器２２０は、書込時に発生される画像データ書込のタイミングを

与える書込クロック信号φＣＷをカウントし、そのカウント値を列アドレスポインタＡＹＰＷとして発生する１６進カウンタ２２０ａと、１６進カウンタ２２０ａのカウントアップ信号φＵＰＹをカウントしそのカウント値を行アドレスポインタＡＸＰＷとして発生する４８進カウンタ２２０ｂを含む。４８進カウンタ２２０ｂは、６ビット２進カウンタで構成することができ、その最大カウント値を４８に設定し、カウント値が４８に到達したときには次のクロックサイクルでそのカウント値が１に初期設定される。

【０２９７】リードアドレス発生器２２２も同様の構成を備え、データ読出時に発生されるリードクロック信号φＣＲをカウントしそのカウント値をデータ読出時の行アドレスポインタＡＸＰＲとして発生する４８進カウンタ２２２ｂと、４８進カウンタ２２２ｂのカウントアップ信号φＵＰＸをカウントし、そのカウント値をデータ読出時の列アドレスポインタＡＹＰＲとして発生する１６進カウンタ２２２ａを含む。

【０２９８】フレーム／フレーム用アドレス生成器２１３はさらに、データ書込／読出を示すリード／ライト信号Ｒ／ＷＺに従ってデータ書込時に発生される列アドレスポインタＡＹＰＷとデータ読出時に発生される列アドレスポインタＡＹＰＲの一方を列アドレスポインタＡＹＰとして通過させるマルチプレクサ（ＭＵＸ）２２４ａと、リード／ライト指示信号Ｒ／ＷＺに従って書込時に発生される行アドレスポインタＡＸＰＷとデータ読出時に発生される行アドレスポインタＡＸＰＲの一方を行アドレスポインタＡＸＰとして通過させるマルチプレクサ２２４ｂを含む。マルチプレクサ２２４ａおよび２２４ｂからのアドレスポインタＡＹＰおよびＡＸＰは図３５に示すセレクタ２１１の一方入力へ与えられる。次に動作について説明する。

【０２９９】まず、図１１６（Ａ）を参照して、画像データ書込時の動作について説明する。この場合には、図１１５に示すライトアドレス発生器２２０が動作し、リードアドレス発生器２２０は動作しない。ライトアドレス発生器２２０において、１６進カウンタ２２０ａおよび２２０ｂは初期値にリセットされる。まず最初の行アドレスポインタＡＸＰＷおよび列アドレスポインタＡＹＰＷはともに０となり、バッファメモリ２０５のＸアドレスおよびＹアドレス（Ｘ，Ｙ）＝（０，０）が指定される。これにより参照画像メモリから伝達されたデータのアドレス（０，０）に書込まれる。次いでライトクロック信号φＣＷが発生され、１６進カウンタ２２０ａのカウント値が１増分される。これにより列アドレスポインタＡＹＰＷが１増分される。行アドレスポインタＡＸＰＷは変化しない。したがってアドレス（０，１）が指定され、２番目に与えられた画素データが書込まれる。以降１６進カウンタ２２０ａのカウント値が１５に到達するまでバッファメモリ２０５において第０行に順次画素データが格納される。このバッファメモリ２０５において１６個の画素データが格納された後、１６進カウンタ２２０ａからカウントアップ信号φＵＰＹが発生され、４８進カウンタ２２０ｂのカウント値が１増分される。これにより行アドレスポインタＡＸＰＷが１となり、１６進カウンタ２２０ａのカウント値は０となる。したがってアドレス（１，０）に次に与えられる１７番目の画素データが書込まれる。以降この動作を繰返すことにより、１６個の画素データが書込まれるごとに４８進カウンタ２２０ｂのカウント値すなわち行アドレスポインタＡＸＰＷが１増分され、各行ごとに順次データが書込まれる。通常、参照画像メモリに与えられる画像データはラスタスキャン走査方式で走査された画素データであり、水平方向に１行に配置される画素データが順次与えられる。したがってデータ書込時にこのラスタスキャン走査方式に合わせて順次データを格納する（参照画像メモリにおいてもラスタスキャン走査方式に従ってデータが書込まれており、ラスタスキャン方式に従って同様データの読出が行なわれるためである）。

【０３００】次に図１１６（Ｂ）を参照して図１１５に示すサーチウィンドウバッファメモリ２０５から順次画素データを読出して演算部１へ与える場合の動作について説明する。画素データ読出時においては、リードアドレス発生器２２２が動作し、ライトアドレス発生器２２０は動作しない。１６進カウンタ２２２ａおよび４８進カウンタ２２２ｂはともに初期値が０に設定されており、最初に読出されるアドレスは（０，０）である。次いでリードクロック信号φＣＲが発生されると４８進カウンタ２２２ｂのカウント値が１増分され行アドレスポインタＡＸＰＲの値が１増分される。カウントアップ信号φＵＰＸは発生されないため、１６進カウンタ２２２ａのカウント値は１を維持する。したがって２番目にはアドレス（１，０）の画素データが読出される。以降リードクロック信号φＣＲに従って４８個のデータが読出される。４８進カウンタ２２２ｂのカウント値が４７に到達すると次のクロックサイクルにおいてカウントアップ信号φＵＰＸが発生され、１６進カウンタ２２２ａのカウント値が１増分され、１となる。４８進カウンタ２２２ｂはこのカウントアップ信号の発生と同期してそのカウント値が初期値にリセットされる。したがって、４９番目の画素データはアドレス（０，１）に格納される。以降この動作を繰返すことにより、順次、行アドレスが増分する方向に沿って画素データが読出されていく。この行アドレスを順次増分して読出す構成により、サーチエリア内の画素を行アドレス増加方向に沿ってスキャンすることができ、演算部１においては単に与えられたデータを順次シフトすることにより演算に必要なデータがすべて与えられることになる。

【０３０１】［フレーム／フレームアドレス生成器の変更例］図１１７は、フレーム／フレームアドレス生成器

の変更例を示す図である。図１１７において、フレーム／フレームアドレス生成器２１３は、クロック入力ＣＬＫへ与えられるクロック信号をカウントし、そのカウント値が１５に到達しかつ初期値のリセット時にカウントアップ信号をそのカウントアップ出力端子ｕｐから出力する１６進カウンタ２２１と、クロック入力端子ＣＬＫに与えられる信号をカウントし、そのカウント値が４７に到達したとき次のクロックサイクル移行時にカウントアップ信号を出力端子ｕｐから出力する４８進カウンタ２２３を含む。１６進カウンタ２２１のクロック入力端子ＣＬＫへは、マルチプレクサ２２７を介してライトクロック信号φＣＷまたは４８進カウンタ２２３のカウントアップ信号が与えられる。４８進カウンタ２２３のクロック入力端子ＣＬＫにはマルチプレクサ２２５を介してリードクロック信号φＣＲおよび１６進カウンタ２２１のカウントアップ信号の一方が与えられる。マルチプレクサ２２５および２２７の選択はリード／ライト指示信号Ｒ／ＷＺにより設定される。リードライト指定信号Ｒ／ＷＺがデータ書込モードを示すとき、マルチプレクサ２２５は１６進カウンタ２２１のカウントアップ信号を４８進カウンタ２２３のクロック入力端子ＣＬＫへ与える。マルチプレクサ２２７はライトクロック信号φＣＷを１６進カウンタ２２１のクロック入力端子ＣＬＫへ与える。図１１７に示す構成の場合には、１６進カウンタおよび４８進カウンタはそれぞれ１つ必要とされるだけであり、装置規模を低減することができる。

【０３０２】図１１８は、図１１４に示すフィールド／フレーム用アドレス生成器の具体的構成を示す図である。図１１８において、フィールド／フレーム用アドレス生成器２１５は画素データ書込時における行および列アドレスポインタＡＰＸＷおよびＡＰＹＷを発生するためのライトアドレス発生器２３０と、画素データ読出時に行および列アドレスポインタＡＰＸＲおよびＡＰＹＲを発生するリードアドレス発生器２３５を含む。

【０３０３】ライトアドレス発生器２３０は、画素データ書込時に発生されるライトクロック信号φＣＷをカウントし、そのカウント値をライト列アドレスポインタＡＰＷとして発生する１６進カウンタ２３２からのカウントアップ信号φＵＰＹをカウントし、そのカウント値をライトアドレスポインタＡＰＸＷとして発生するライトアドレスポインタ発生器２３４を含む。ライトアドレスポインタ発生器２３４は、１６進カウンタ２３２からのカウントアップ信号φＵＰＹをカウントする２４進カウンタ２３３と、この２４進カウンタ２３３のカウントアップ信号φＵａに従ってその出力の論理が変化するＴ型フリップフロップ２３１を含む。Ｔ型フリップフロップ２３１の出力はライト行アドレスポインタＡＰＸＷの最下位ビットとして用いられ、２４進カウンタ２３３の出力がライト行アドレスポインタＡＰＸＷの残りの上位ビットとして利用される。

【０３０４】リードアドレス発生器２３５は、画素データ書込時に発生されるリードクロック信号φＣＲをカウントし、そのカウント値をリード行アドレスＡＰＸＲとして発生する４８進カウンタ２３８と、４８進カウンタ２３８からのカウントアップ信号φＵＰＸをカウントし、そのカウント値をリード列アドレスＡＰＹＲとして出力する１６進カウンタ２３６を含む。ライトアドレス発生器２３０からのアドレスポインタＡＰＹＷおよびＡＰＸＷならびにリードアドレス発生器２３５からのアドレスポインタＡＰＹＲおよびＡＰＸＲはマルチプレクサ２４０へ与えられる。マルチプレクサ２４０は、リード／ライト指定信号Ｒ／ＷＺに従ってライトアドレスポインタＡＰＹＷおよびＡＰＸＷとリードアドレスポインタＡＰＹＲおよびＡＰＸＲとの一方を選択してアドレスポインタＡＰＹおよびＡＰＲとしてセレクタ２１１（図１１４参照）へ伝達する。次に動作について説明する。

【０３０５】まず図１１９を参照して、ライトアドレス発生器２３０の動作について説明する。１６進カウンタ２３２および２４進カウンタ２３３はそれぞれ最大カウント値１５および０に初期設定されている。Ｔ型フリップフロップ２３１は０出力状態に初期設定されている。ライトクロック信号φＣＷが与えられると、この１６進カウンタ２３２の出力が０となる。このときにカウントアップ信号φＵＰＹは発生されない（先のサイクルにおいて発生されて２４進カウンタを０に初期設定するために利用されているかまたは初期値設定時にこのカウントアップ信号φＵＰＹの発生が禁止される）。したがって、第１回目にライトクロック信号φＣＷが与えられた場合、行および列アドレスポインタＡＰＸＷおよびＡＰＹＷはともに０となり、アドレス（０，０）が指定される。次にライトクロック信号φＣＷが与えられると、１６進カウンタ２３２のカウント値が１増分する。リードアドレス発生器２３４からのリードアドレスポインタＡＰＸＷは変化せず、０を維持する。したがって第０行に対して画素が順次アドレス指定される。

【０３０６】ライトクロック信号φＣＷが１６回カウントされ、１６進カウンタ２３２のカウント値すなわちライト列アドレスポインタＡＰＹＷが１５となると、所定のタイミングでカウントアップ信号φＵＰＹが発生される。アドレス（０，１５）が指定され、画素データが書込まれた後このカウントアップ信号φＵＰＹに従って２４進カウンタ２３３のカウント値が１増分される。一方、ライトクロック信号φＣＷが与えられると１６進カウンタ２３２のカウント値が０に復帰する。ここで、２４進カウンタ２３３の出力の変化時点はライトクロック信号φＣＷに同期するように構成されてもよい。これにより、アドレス（２，０）が指定されて次いでライトクロック信号φＣＷが与えられると順次第２行の画素データを格納するためのアドレス指定が行なわれる。

【０３０７】上述の動作を繰返し、２４進カウンタ２３

３のカウント値が最大値（２３）となるとカウントアップ信号φＵａが発生され、Ｔ型フリップフロップ２３１の出力状態が変化する。すなわちＴ型フリップフロップ２３１の出力が“１”となる。２４進カウンタ２３３の出力は、１６進カウンタ２３２からのカウントアップ信号に従って０に復帰する。２４進カウンタ２３３の出力が０であり、Ｔ型フリップフロップ２３１の出力が１であるため、このリード行アドレスポインタＡＰＸＷは１を指定する。したがって、次にライトクロック信号φＣＷが与えられるとアドレス（１，０）が指定され、以降この第１行の画素データ格納位置が順次アドレス指定された後、２４進カウンタ２３３のカウント値が１６進カウンタ２３２のカウントアップ信号φＵＰＹに従って変化するとリード行アドレスポインタＡＰＸＷは３となり、第３行の画素データ格納位置が順次アドレス指定される。

【０３０８】したがって、図１２０（Ａ）に示すように第１フィールド（偶数フィールドまたは奇数フィールド）に含まれる画素データはサーチウィンドウバッファメモリ２０５の偶数行に順次格納され、第２フィールド（奇数フィールドまたは偶数フィールド）に属する画素データは図１２０（Ｂ）に示すように奇数行に順次格納される。この第１フィールドおよび第２フィールドと偶数フィールドおよび奇数フィールドとの対応関係は任意であるが、先の説明との対応で言えば、第１フィールドは偶数フィールドに対応し、第２フィールドは奇数フィールドに対応する。したがって、フィールド単位で与えられる画素データ（たとえば一般のＮＴＳＣ方式に従う画像データ）を２フィールドを単位としてバッファメモリに格納することにより１フレームの画像に含まれる画素データがフレーム形式でサーチウィンドウバッファメモリ２０５内に格納される。

【０３０９】リードアドレス発生器２３５の動作は、先に図１１５を参照して説明したリードアドレス発生器２２２の動作と同じであり、リードクロック信号φＣＲに従って第０列の画素データが第０行から第４７行まで順次読出され、次に第１列の画素データが読出され、以降この動作が繰返される。この読出動作時における画素データの読出シーケンスは図１１６（Ｂ）に示すものと同じである。

【０３１０】上述のように、フレーム形式またはフィールド形式いずれの形式の画像データが入力されてもすべてフレーム形式の画像データに変換して演算部１へ伝達することができる。したがって、演算部１は、入力画像データの画面形式にかかわらずフレームを単位として動きベクトル検出を実行することが可能となる。

【０３１１】［入力部の第２の構成］図１２１は、入力部の第２の構成を概略的に示す図である。この図１２１に示す構成においては、フレーム形式またはフィールド形式の画像画素データ列をフィールド形式画像の画素データ列に変換して出力する。

【０３１２】画面形式変換部２０７は、フィールド形式の画像画素データ列が与えられたときにサーチウィンドウバッファメモリ２０５に対するアドレスを発生するフィールド／フィールド用アドレス生成器２５４と、フレーム形式の画像画素データ列が与えられたときにフィールド形式の画像画素データ列に変換するアドレスを発生するフレーム／フィールド用アドレス生成器２５２と、画面形式指定信号φＰＴに従ってフィールド／フィールド用アドレス生成器２５４およびフレーム／フィールド用アドレス生成器２５２が出力するアドレスポインタの一方を選択してサーチウィンドウバッファメモリ２０５へ与えるセレクタ２１１を含む。フィールド／フィールド用アドレス生成器２５４の構成は図１１５に示すフレーム／フレーム用アドレス生成器２１３の構成と同じである。処理対象となる画素データ列の画面形式がフレーム形式であるかフィールド形式であるかが異なるだけである。

【０３１３】図１２２（Ａ）にフィールド／フィールド用アドレス生成器２５４が発生するアドレスポインタの内容およびサーチウィンドウバッファメモリ２０５に格納される画素データ列の対応関係を示す。図１２２（Ａ）に示すように、フィールド形式画像の画素データ列が与えられるとき、フィールド／フィールド用アドレス生成器２５４は、まず第０行を指定しこの第０行において第０列ないし第１５列を順次指定する。第０行において第１５列に画素データが書込まれた後次いで第１行が指定され、再び第１行において１６個の画素データが書込まれる。以降この動作を第４７行に対するまで繰返し実行する。ただしここで演算部１において利用されるサーチウィンドウのサイズは４８画素（行）×１６画素（列）としている。

【０３１４】画素データ読出時においてはフィールド／フィールド用アドレス生成器２５４は、図１２２（Ｂ）に示すように、まず第０列を選択状態とし、順次第０行ないし第４７行を指定する。これにより第０列の画素データが順次読出される。第０列の画素データ（４８画素）が読出されると、次いで第１列における４８個の画素が順次読出される。この動作を繰返し実行する。これにより、演算部１においてプロセッサアレイ内に格納するサーチウィンドウブロック画素データの順序でサーチウィンドウバッファメモリ２０５から画素データ列を読出す。

【０３１５】図１２３に、フレーム／フィールド用アドレス発生器２５２が発生するアドレスポインタの発生順序を示す。フレーム／フィールド用アドレス生成器２５２は、フレーム形式の画像画素データ列が与えられたときにサーチウィンドウバッファメモリ２０５に対する画素データ格納位置を示すアドレスポインタを発生する。画素データ書込時においては、図１２３（Ａ）に示すよ

うに、フレーム／フィールド用アドレス生成器２５２は、フィールド形式画像入力時と同様に、１行ずつ順次画素データを格納するようにアドレスポインタを発生する。この画素データ書込時においてはフレーム／フィールド用アドレス生成器２５２は９５画素（行）×１６画素（列）の画素データを単位として格納する。フレーム形式画像は、偶数フィールドおよび奇数フィールド（第１フィールドおよび第２フィールド）を含んでおり、フィールド形式の画像作成のためにはこのフレーム画像から１枚のフィールド画像の画素データを抽出する必要があるためである。画素データ読出時においては、フレーム／フィールド用アドレス生成器２５２は、図１２３（Ｂ）に示すように、各列ごとに偶数行の画素データのみを読出すようにアドレスポインタを発生する。これにより第１フィールドの画素データのみが抽出されて４８画素行×１６画素列のサーチウィンドウの画素データが得られる。

【０３１６】図１２４は、図１２１に示すフレーム／フィールド用アドレス発生器の具体的構成を示す図である。図１２４において、フレーム／フィールドアドレス発生器２５２は、画素データ書込時にアドレスポインタを発生するライトアドレス発生器２６０と、画素データ読出時にアドレスを発生するリードアドレス発生器２６５を含む。ライトアドレス発生器２６０は、画素データ書込時に発生されるライトクロック信号φＣＷをカウントし、そのカウント値を列アドレスポインタＡＰＹＷとして出力する１６進カウンタ２６２と、１６進カウンタ２６２のカウントアップ信号φＵＰＹをカウントしそのカウント値を行アドレスポインタＡＰＸＷとして出力する９５進カウンタ２６４を含む。このライトアドレス発生器２６０は、先に図１１５を参照して説明したライトアドレス発生器と同様の動作を行ない、１６進カウンタ２６２により１行の画素データ格納位置がすべて指定されたときに次の行のアドレスが指定される。９５進カウンタ２６４は、９５行のアドレスを順次指定する。

【０３１７】リードアドレス発生器２６５は、画素データ読出時に発生されるリードクロック信号φＣＲをカウントする４８進カウンタ２６８と、４８進カウンタ２６８のカウントアップ信号φＵＰＸをカウントし、そのカウント値を列アドレスポインタＡＰＹＲとして出力する１６進カウンタ２６６と、固定値“０”を発生する１ビットアドレス発生器２６７を含む。４８進カウンタ２６８の出力と１ビットアドレス発生器２６７の出力とが行アドレスポインタＡＰＸＲとして利用される。１ビットアドレス発生器２６７からの固定ビット値“０”は画素データ読出時に利用される行アドレスポインタＡＰＸＲの最下位ビットとして利用される。したがって行アドレスポインタＡＰＸＲは偶数行のみを指定する。このリードアドレス発生器２６５の構成により、画素データ読出時に第０行、第２行、第４行、…、第９４行を順次指定して画素データを読出すことができ、１列の画素データを読出した後１６進カウンタ２６６のカウント値を１増分して次の列の画素データを順次読出すことができる。

【０３１８】この図１２１に示すような構成を利用すれば、入力画像画素データの画面形式がフィールド形式およびフレーム形式いずれであってもフィールド形式の画像画素データ列を生成することができ、入力画像の画面形式にかかわらずフィールドを単位として動きベクトルを検出することができる。また外部にフレーム形式の画像画素データ列をフィールド形式の画像画素データ列に変換する回路を設ける必要がなく、装置規模を増大させることなく任意の画面形式に対応することのできる汎用性の高い動きベクトル検出装置を実現することができる。

【０３１９】［入力部の第３の構成］図１２５は、この発明に用いられる動きベクトル検出装置の入力部の第３の構成を示す図である。図１２５において、画面形式変換回路２０７は、フィールド形式の画像画素データ列が与えられたときに書込アドレスポインタを発生するフィールド／フィールド用アドレス発生器２７２と、フレーム形式の画像画素データ系列が与えられたときに書込アドレスポインタを発生するフレーム／フィールド用アドレス発生器２７４と、画面形式指定信号φＰＴに従ってアドレス発生器２７２および２７４の一方の出力アドレスを選択するセレクタ２７６と、フレーム／フィールド用アドレス発生器２７４から発生されるアドレスポインタの最下位ビットの値に従ってサーチウィンドウバッファメモリ２０５へのデータの書込を禁止する書込禁止回路２７７と、バッファメモリ２０５からの画素データ読出時の読出アドレスを発生するリードアドレス発生器２７９を含む。フィールド／フィールド用アドレス発生器２７２およびフレーム／フィールド用アドレス発生器２７４は画面形式指定信号φＰＴに従って一方のみが活性状態とされる。この画面形式指定信号φＰＴに従ってアドレス発生器２７２および２７４の一方が活性化される構成は、先の第１および第２の画面形式変換回路の構成においても同様である。

【０３２０】サーチウィンドウバッファメモリ２０５は、行および列のマトリックス状に配列された複数のメモリセル（たとえばＳＲＡＭセル）を有するメモリセルアレイ２８０と、データ書込時に画面形式変換回路２０７のセレクタ２７６を介して与えられる書込アドレスポインタに従ってメモリセルアレイ２８０内のメモリセルを選択し、選択されたメモリセルへ入力画像画素データを書込む書込制御回路２８２と、データ読出時にリードアドレス発生器２７９から与えられたアドレスポインタに従ってメモリセルアレイ２８０から対応のメモリセルを選択し、その記憶データを読出す読出制御回路２８４を含む。書込制御回路２８２および読出制御回路２８４はそれぞれクロック信号φＣＷおよびφＲＷに従って動

作する。書込制御回路２８２は、書込禁止回路２７８が書込禁止を示すとき外部から画素データが与えられていてもデータの書込を実行しない。

【０３２１】フィールド／フィールド用アドレス発生器２７２は図１１５に示すライトアドレス発生器２２０と同じ構成を備える。メモリセルアレイ２８０は、サーチウィンドウのデータを格納するために４８行１６列に配置されたメモリセルを備えると仮定する。フィールド形式の画像画素データ列が与えられた場合には、フィールド／フィールド用アドレス発生器２７２が活性化され、書込制御回路２８２の制御の下にフィールド形式画像の画素データ列がメモリセルアレイ２８０内へ順次先に説明したものと同様にして書込まれる。フレーム形式の画像の画素データ列が与えられた場合、フレーム／フィールド用アドレス発生器２７４の発生するアドレスポインタに従って画素データの書込が実行される。このときフレーム／フィールド用アドレス発生器２７４の発生するアドレスに従って１行に対する画素データの書込が行なわれた場合、次の行に対する書込はフレーム内の別のフィールドの画素データであり、データの書込が禁止される。すなわち、フレーム／フィールド用アドレス発生器２７４が発生するアドレスに従ってフレーム形式画像のうち１つのフィールド（偶数フィールドまたは奇数フィールド；第１フィールドまたは第２フィールド）の画素データが抽出され、この１つのフィールドに対してのみ書込制御回路２８２によりメモリセルアレイ２８０へのデータ書込が行なわれ、別のフィールドに対しては書込禁止回路２７８の制御の下に画素データの書込が禁止される。したがって、メモリセルアレイ２８０においては入力される画像の形式がフレーム形式およびフィールド形式いずれであってもフィールド形式の画像の画素データが格納される。

【０３２２】リードアドレス発生器２７９は図１１８に示すリードアドレス発生器２３５と同様の構成を備え、メモリセルアレイ２８０において、各列ごとに順次メモリセルが選択されるようにアドレスを発生する。読出制御回路２８４は読出クロック信号φＲＷに従ってリードアドレス発生器２７９からのアドレスに従ってメモリセルアレイ２８０内のメモリセルを選択し、その選択されたメモリセルが有する画素データを読出す。したがって読出制御回路２８４からはフィールド形式の画像の画素データ列が順次出力される。

【０３２３】図１２６は図１２５に示すフレーム／フィールド用アドレス発生器２７４の具体的構成を示す図である。図１２６において、フレーム／フィールド用アドレス発生器２７４は、ライトクロック信号φＣＷをカウントしそのカウント値を列アドレスポインタＡＰＹＷとして発生する１６進カウンタ２９２と、１６進カウンタ２９２のカウントアップ信号φＵＰＹをカウントする７ビットカウンタ（９５進カウンタ）２９４を含む。７ビットカウンタ２９４は、最下位ビット（ＬＳＢ）のみを含む１ビットカウンタ２９６と、残りの上位６ビットカウント値を出力する６ビットカウンタ２９８を含む。１ビットカウンタ２９６のカウント値は書込禁止回路２７８へ書込禁止指示信号として与えられる。この１ビットカウンタ２９６のカウント値（ＬＳＢ）が“１”のとき書込禁止回路２７８は書込禁止が指定されたとして書込制御回路２８２によるデータ書込動作を禁止する。６ビットカウンタ２９８のカウント値が行アドレスポインタＡＰＸＷとして出力される。次にこの図４７に示すフレーム／フィールド用アドレス発生器２７４の動作をその動作タイミング図である図１２７を参照して説明する。

【０３２４】初期時においては１６進カウンタ２９２および７ビットカウンタ２９４のカウント値はともに０に設定される。１６進カウンタ２９２はライトクロック信号φＣＷをカウントして列アドレスポインタＡＰＹＷを発生する。この１６進カウンタ２９２のカウント値が１５に到達すると１行すなわち第０行の１６画素のデータの書込が完了したことになる。この書込まれた画素データは第１フィールドに属している。

【０３２５】次に１６進カウンタ２９２のカウント値が０に戻る。この１６進カウンタ２９２のカウント値の初期値への復帰に応答して７ビットカウンタ２９４のカウント値が１増分される。このときには、最下位ビットＬＳＢを出力する１ビットカウンタ２９６のカウント値が１となり、６ビットカウンタ２９８のカウント値は変化しない。したがって行アドレスポインタＡＰＸＷは０を維持する。このとき、１ビットカウンタ２９６のカウント値（ＬＳＢ）が１であるため、書込禁止回路２７８は書込制御回路２８２の書込動作を禁止する。これにより１６進カウンタ２９２のカウント値が１５に到達するまでの期間画素データの書込が禁止され、第２フィールドに属する画素データの書込が禁止される。

【０３２６】次いで再び１６進カウンタ２９２のカウント値が０に復帰すると、カウントアップ信号φＵＰＹに従って７ビットカウンタ２９４のカウント値が１増分される。これにより１ビットカウンタ２９６のカウント値（ＬＳＢ）が０となり、６ビットカウンタ２９８のカウント値が１となる。この１ビットカウンタ２９６のカウント値（ＬＳＢ）が０であるため、書込禁止回路２７８は書込禁止を解除する。これにより書込制御回路２８２はフレーム形式の画像の画素データ列を順次書込む。このとき書込まれる画素データは第１フィールドに属しており、第１行に書込まれる。

【０３２７】次いで再び１６進カウンタ２９２のカウント値が１５に到達し、１行の画素データの書込が完了するとカウントアップ信号φＵＰＹが発生される。これにより７ビットカウンタ２９４のカウント値が１増分され、１ビットカウンタ２９６のカウント値が１となり、６ビットカウンタ２９８のカウント値は１を維持する。

したがってこの場合には、１ビットカウンタ２９６のカウント値１に従って画素データの書込が禁止される。以降この動作を繰返すことによりフレーム形式の画像の画素データ列のうち第２フィールドの画素データに対する書込が禁止され、第１フィールドの画素データのみがメモリセルアレイ２８０内に書込まれる。

【０３２８】すなわち、図１２８に示すようにフレーム形式画像２９０から必要とされる第１フィールドの画素データのみを選択してメモリセルアレイ２８０内へ書込むことにより、第１フィールド画素データのみでサーチウィンドウを形成することができ、フレーム形式の画像の画素データ列からフィールド形式画像の画素データを得ることができる。

【０３２９】データの読出は、先に示したフィールド／フィールド画像のデータ読出動作（図１２２（Ｂ）参照）と同じである。

【０３３０】上述のようにフレーム形式画像から必要とされるフィールドの画像データのみを抽出してメモリセル内へ格納することにより、入力画像データの画像形式にかかわらず常にフィールド単位で動きベクトルを検出することが可能となる。

【０３３１】［データバッファの構成］図１２９はプロセッサアレイ内に含まれるサイドウィンドウブロックデータを格納するためのデータレジスタの構成を示す図である。図１２９において、データレジスタ（バッファ）ＤＬは、複数の縦続接続されたデータ転送機能を備えるレジスタを含む。複数のレジスタは2つずつグループに分割される。図１２９においては2つのレジスタグループＧ＃１、およびＧ＃ｎを代表的に示す。レジスタグループＧ＃１、…、Ｇ＃ｎはそれぞれ縦続接続されたレジスタＲＧ１およびＲＧ２を含む。これらのレジスタグループＧ＃１～Ｇ＃ｎは一方方向に沿って図示しないクロック信号に応答してサーチウィンドウ画素データを伝達する機能を備える。

【０３３２】データバッファＤＬはさらに、各レジスタグループＧ＃１～Ｇ＃ｎの出力のいずれかを選択信号φＳＥＬに従って選択するセレクタ３００を含む。この選択信号φＳＥＬは外部から与えられる。このセレクタ３００により、データバッファＤＬのレジスタの段数を変更することができる。たとえばレジスタグループＧ＃１が選択された場合、このデータバッファＤＬのレジスタの段数は2であり、サーチウィンドウ画素データを２画素格納することができる。レジスタグループＧ＃ｎが選択された場合には、このデータバッファＤＬはそこに含まれるレジスタの段数だけサーチウィンドウ画素データを格納することができる。

【０３３３】データバッファＤＬは、サイドウィンドウブロックの画素データを格納する（図8参照）。したがって、データバッファに含まれるレジスタの段数を変更することによりサイドウィンドウブロックのサイズを変更することができ、応じてサーチウィンドウのサイズ、すなわちサーチエリアの範囲を変更することができる。これにより仕様に応じてサーチエリアが変更されても容易に対応することができる。なお、レジスタグループに含まれるレジスタの数およびレジスタグループの数は任意である。

【０３３４】［データバッファの変更例１］図１３０はデータバッファの変更例を示す図である。図１３０において、データバッファＤＬは縦続接続された複数のレジスタＲＧａ～ＲＧｎを含む。これらのレジスタＲＧａ～ＲＧｎを介してデータ転送が可能である。レジスタＲＧａ～ＲＧｎの出力部に活性化信号φａ～φｎに応答して導通する選択ゲートＳＥＬａ～ＳＥＬｎがそれぞれ設けられる。外部から与えられる選択信号φａ～φｎのうちのいずれか１つが活性状態とされる。導通状態とされた選択ゲートＳＥＬｉは、対応のレジスタＲＧｉの出力をデータ線３０２へ伝達する。このデータ線３０２は要素プロセッサアレイ内における同じ線形プロセッサアレイの最下流の要素プロセッサＰＥｍの入力部に接続される。

【０３３５】図１３０に示す構成に従えば、選択信号φａ～φｎに従ってデータバッファのレジスタ段数をレジスタ単位で設定することができ、より柔軟にサーチエリアの仕様変更に対処することができる。

【０３３６】［データバッファの他の構成例］図１３１は、データバッファの他の具体的構成を示す図である。図１３１において、データバッファＤＬは、サーチウィンドウ画素データを入力する入力回路３１０と、入力回路３１０から与えられるサーチウィンドウ画素データを各ファイルに格納する複数のファイルを備えるレジスタファイル３１２と、レジスタファイル３１２へのデータ書込時におけるライトアドレスポインタを発生するライトアドレス発生回路３１６と、レジスタファイル３１２からのデータ読出時に用いられるリードアドレスポインタを発生するリードアドレス発生回路３１４と、レジスタファイル３１２からのリードアドレス発生回路３１４により指定されたファイルの内容を読出す出力回路３１８と、リードアドレス発生回路３１４の初期アドレスを設定する初期アドレス設定回路３１９を含む。レジスタファイル３１２は、たとえばＳＲＡＭセルを用いて構成される、１ファイルに１画素データを格納する。ライトアドレス発生回路３１６およびリードアドレス発生回路３１４は、それぞれライトクロック信号φＷおよびリードクロック信号φＲに従ってアドレスを順次１ずつ増分または減分する。このデータバッファＤＬの構成においては、データの読出が行なわれた後にデータの書込が実行される。読出すべきデータが新たに与えられたデータで変更されるのを防止するためである（ただしリードアドレスとライトアドレスが同じアドレスを指定するように設定される場合）。次に図１３１に示すデータバッフ

ァの動作を図１５２を合わせて参照して説明する。

【０３３７】図１５２においては、レジスタファイル３１２のファイル数が８であり、８個のファイルにそれぞれアドレス０ないし７が割り当てられている構成が一例として示される。また図１５２においては、初期アドレス設定回路３１９は、リードアドレス発生器３１４の初期アドレスとしてアドレス４を指定している。

【０３３８】ライトアドレス発生回路３１６は、ライトクロック信号φＷに従ってアドレス０から順次書込アドレスを指定する。このライトアドレス発生回路３１６からのライトアドレスポインタに従って入力回路３１０は与えられたサーチウィンドウ画素データをレジスタファイル３１２における対応のファイルに書込む。したがってレジスタファイル３１２においてアドレス０、１、…に対し入力されたサーチウィンドウ画素データＳＹ１、ＳＹ２、…が順次書込まれる。リードアドレス発生回路３１４は、リードクロック信号φＲに従って、初期アドレス設定回路３１９により設定された初期アドレス４から順次アドレスを指定する。リードアドレス発生回路３１４が発生するリードアドレスポインタも０になると、レジスタファイル３１２のアドレス０に格納された画素データＳＹ１が出力回路３１８により読出される。以降順次レジスタファイル３１２に格納された画素データＳＹ２、ＳＹ３、…がリードクロック信号φＲに従って読出される。したがってこの構成においては、入力サーチウィンドウ画素データは５クロックサイクル遅れて出力されることになる。したがってこの初期アドレス設定回路３１９によるリードアドレスの初期アドレスを変更することにより、このデータバッファの遅延時間を変更することができ、このデータバッファに格納される有効サーチウィンドウ画素データの数は変更することができ、サーチエリア（サーチウィンドウ）の垂直方向のサイズを変更することができる。データバッファの遅延時間を変更することによりサーチエリアの垂直方向のサイズが変更されることについて以下に具体的に説明する。

【０３３９】図１３３（Ａ）に示すように、ライトアドレスポインタＷＰがアドレス４を示しているとき、リードアドレスポインタＲＰがアドレス０を示す状態を考える。この状態においては、レジスタファイル３１２においてはサーチウィンドウ画素データＳＹ１～ＳＹ５が格納されている。

【０３４０】図１３３（Ｂ）に示すようにライトアドレスＷＰが７を示すとき、リードアドレスＲＰはアドレス３を示す。この状態においてはアドレス０、１および２のファイルに格納された画素データＳＹ１～ＳＹ３は既に読出されている。したがって画素データＳＹ４を読出す前には、レジスタファイル３１２においては画素データＳＹ４～ＳＹ８の画素データが有効画素データ（これから読出されるべき画素データ）として格納されている。

【０３４１】図１３３（Ｃ）に示すようにさらにクロックサイクルが進み、ライトアドレスポインタＷＰがアドレス３を示すとき、リードアドレスポインタＲＰはアドレス７を示している。この状態においては、アドレス４ないし６に格納された画素データＳＹ５～ＳＹ７は既に読出されており、消費された画素データである。この状態においても、レジスタファイル３１２に格納された画素データＳＹ８～ＳＹ１２が有効画素データとなる。

【０３４２】次のサイクルにおいては、図１３３（Ｄ）に示すようにライトアドレスポインタＷＰがアドレス４を指定し、リードアドレスポインタＲＰがアドレス０を指定する。この状態においては画素データＳＹ９～ＳＹ１３が有効画素データとなる。

【０３４３】すなわち、入力サーチウィンドウ画素データはレジスタファイル３１２内のすべてのファイル（図示の例では８個のファイル）に格納されるが、有効サーチウィンドウ画素データの数はすべてのレジスタファイルの数以下となる（図示の例では５）。このレジスタファイル３１２内に格納される有効サーチウィンドウ画素データの数は初期アドレス設定回路３１９が指定する初期アドレスに１加えた値と等しくなる。したがってこのレジスタファイル３１２は図１３３（Ｅ）に示すように５段のレジスタＲＧ１～ＲＧ５を縦続接続したシフトレジスタと等価な構成となる。データバッファ内には図８に示すサイドウィンドウブロックの画素データが格納される。この図１３３（Ｅ）に示すレジスタの段数はサイドウィンドウブロックの垂直方向の画素数に等しい。したがって図１３１に示す構成を利用することにより、サイドウィンドウブロックの垂直方向の長さを変更することができ、応じてサーチウィンドウの垂直方向の長さを調節することができる。

【０３４４】上述の構成により、任意のサーチエリアのサイズに対して柔軟に対処することが可能となる。

【０３４５】［階層化動きベクトル検出方式］図１３４はこの発明に従う階層化動きベクトル検出方式に用いられるサーチエリアおよびテンプレートブロックの構成を示す図である。図１３４において、テンプレートブロック３５０は、先に説明した全探索方式動きベクトル検出装置において用いられるものと同様、１６画素（行）×１６画素（列）の画素で構成される。動きベクトル探索範囲は水平および垂直方向ともに±６４画素である。したがってサーチエリア３５２は、（６４＋１６＋６４）×（６４＋１６＋６４）＝１４２×１４２画素で構成される。

【０３４６】階層化動きベクトル検出方式においては、２段階で動きベクトルの探索が行なわれる。第１段階ではサーチエリア３５２内の評価点（変位ベクトルに対応し、評価関数値が求められる点：（－６４，－６４）～（６４，６４）の各点）を間引いてブロックマッチング処理を行なって評価関数値を算出する。算出された評価

関数値から最も良い相関度を示す最適評価関数値および対応の変位ベクトルを求める。この最適評価関数値および対応の変位ベクトルの検出を複数の予測モードそれぞれに対して並列に実行する。次に具体的に第１段階のブロックマッチング処理について説明する。

【０３４７】図１３５（Ａ）に示すように、まず、評価点（－６４，－６４）のサーチウィンドウブロック３５４ａのテンプレートブロック３５０に対する評価関数値が算出される。この評価関数値の算出動作は先の全探索方式の動きベクトル検出装置において述べたものと同様、対応の画素間の差分絶対値の算出および求められた差分絶対値の総和が求められる。この評価関数値としては差分絶対値和に代えて差分自乗和などの他の評価関数値が用いられてもよい。

【０３４８】次いで、図１３５（Ｂ）に示すように、評価点（－６４，－６３）～（－６４，－５７）の7評価点がスキップされ（間引かれ）、評価点（－６４，－５６）に対応するサーチウィンドウブロック３５４ｂの評価関数値が算出される。７つの評価点をスキップしての評価関数値算出を繰返し、サーチエリア３５２の下部の評価点（６４，６４）の評価関数値の算出が完了すると、再度サーチエリア３５２の図の上部側からの評価関数値算出が行なわれる。評価点列変換時においては、図１３５（Ｃ）に示すように、水平方向に７画素分評価点がスキップされ、次に評価関数値が算出される評価点は（－５６，－６４）となる。以降垂直方向に７画素分ずつ評価点をスキップして評価関数値を算出し、サーチエリア３５２の下部に評価点に到達すると、水平方向に７画素分の評価点をスキップして再度サーチエリア３５２の上部から順次評価関数値を算出する。

【０３４９】図１３５（Ａ）～（Ｃ）に示す一連の動作を繰返すと、図１３６に示すように、サーチエリア３５２内のすべての代表評価点Ｅ（評価偶数値が算出される評価点であり、水平および垂直方向の８評価点ごとに配置された評価点）に対する評価関数値が算出される。これらの代表評価点Ｅの評価関数値から最も高い相関度を示す評価関数値を検出し、対応の評価点Ｅｓｔが決定される。この最適代表評価点Ｅｓｔは複数の予測モードそれぞれに対して求められる。以上の一連の動作が第１段階の概略探索動作シーケンスであり、次に第２段階の探索動作が行なわれる。

【０３５０】第２段階の探索動作においては、図１３７に示すように、最適代表評価点Ｅｓｔの周辺領域（全探索領域）３５６をサーチエリアとし、この周辺領域３５６内のすべての評価点に対する探索が行なわれる。周辺領域３５６は、少なくともスキップされた評価点を包含する。図１３７に示す構成では、周辺領域３５６は探索範囲を最適代表評価点を基準として水平および垂直±１６とする。したがって周辺領域３５６は３２×３２評価点を含む。この周辺領域３５６内の３２×３２＝９３１の評価点すべてに対し評価関数値の算出を行ない、この算出された評価関数値に従って最適評価点を求め、この最適評価点に対応する変位ベクトルを動きベクトルとして決定する。この全探索動作が複数の予測モードそれぞれに対して並列に実行される。

【０３５１】第１段階での評価点の数は（８＋１＋８）×（８＋１＋８）＝１７×１７＝２８９であり、第2段階での評価点の数は３２×３２＝１０２４であり、合計１３１３となる。したがって全探索方式に比べて広い画像領域をサーチエリアとして動きベクトル検出を行なうことができ、また複数の予測モードそれぞれに対し並列に動きベクトルが決定されるため、装置規模を増大させることなく効率的に動きベクトルを決定することができる。次に具体的構成について説明する。

【０３５２】［階層探索方式動きベクトル検出装置の第１の実施例］図１３８は、階層探索方式動きベクトル検出装置の第１の実施例の構成を概略的に示す図である。図１３８において、階層探索方式動きベクトル検出装置は、サーチエリア内の代表評価点に対する評価関数値をフィールド予測モード、上サブテンプレートブロック予測モード、および下サブテンプレートブロック予測モードそれぞれに従って同時に算出し、これら算出した評価関数値ｅｆ、ｅｕおよびｅｌを対応の変位ベクトル（評価点位置を表わす）ｖｆ、ｖｕおよびｖｌとともに比較選択部４０４へ与える粗探索演算部４０２を含む。この粗探索演算部４０２は、その具体的構成は後に説明するが、図１３５（Ａ）ないし（Ｃ）および図１３６を参照して説明した第１段階の探索動作を実行する。ここで、動きベクトル検出装置はフィールドを単位として動きベクトルを検出する。フィールド予測モードに従って算出される評価関数値および変位ベクトルをｅｆおよびｖｆで示し、上サブテンプレートブロック予測モードに従って算出される評価関数値および対応の変位ベクトルをｅｕおよびｖｕで表わし、下サブテンプレートブロック予測モードに従って算出される評価関数値および対応の変位ベクトルをｅｌおよびｖｌで表わす。

【０３５３】比較選択部４０４は、粗探索演算部４０２からの評価関数値ｅｆ、ｅｕおよびｅｌならびに変位ベクトルｖｆ、ｖｕおよびｖｌを受け、フィールド予測モード、上サブテンプレートブロック予測モード、および下サブテンプレートブロック予測モードそれぞれに対する最適代表評価点を決定し、各予測モードの最適代表評価点を示す変位ベクトルｖｆｓｔ、ｖｕｓｔおよびｖｌｓｔを出力する。最適変位ベクトルｖｆｓｔ、ｖｕｓｔおよびｖｌｓｔは、それぞれフィールド予測モードによる最適変位ベクトル、上サブテンプレートブロック予測モードにおける最適変位ベクトルおよび下サブテンプレートブロック予測モードにおける最適変位ベクトルを表わす。

【０３５４】動きベクトル検出装置はさらに、比較選択

部４０４から与えられた最適変位ベクトルｖｆｓｔ、ｖｕｓｔおよびｖｌｓｔに従って最適代表評価点周辺領域を各最適変位ベクトルに対して抽出し、抽出された周辺領域において全探索を行なう密探索演算部４０６を含む。密探索演算部４０６は、最適変位ベクトルｖｆｓｔ、ｖｕｓｔおよびｖｌｓｔそれぞれに対して設けられた第１ないし第３の全探索演算部４１０、４１２および４１４を含む。第１の全探索演算部４１０は、最適変位ベクトルｖｆｓｔに従って最適代表評価点周辺領域を抽出し、フィールド予測モード、上サブテンプレートブロック予測モード、下サブテンプレートブロック予測モードそれぞれに従って最適評価点を算出し、最適評価点を表わす変位ベクトルＶｆ１、Ｖｕ１およびＶｌ１と対応の評価関数値Ｅｆ１、Ｅｕ１およびＥｌ１を比較選択部４０８へ与える。

【０３５５】第２の全探索演算部４１２は、上サブテンプレートブロック予測モードに従って求められた最適変位ベクトルｖｕｓｔに従って最適代表評価点周辺領域を抽出し、この最適代表評価点周辺領域に対し３つの予測モードに従って評価関数値を求め、対応の変位ベクトルとともに比較選択部４０８へ与える。

【０３５６】第３の全探索演算部４１４は、下サブテンプレートブロック予測モードに従って決定された最適変位ベクトルｖｌｓｔに従って最適代表評価点周辺領域を抽出し、この最適代表評価点周辺領域に対し全探索方式で評価関数値算出を行ない、予測モードそれぞれに対する評価関数値Ｅｆ３、Ｅｕ３およびＥｌ３を対応の変位ベクトルＶｆ３、Ｖｕ３およびＶｌ３とともに比較選択部４０８へ与える。

【０３５７】比較選択部４０８は、合計９種類の評価関数値Ｅｆ１、Ｅｕ１、Ｅｌ１、Ｅｆ２、Ｅｕ２、Ｅｌ２、Ｅｆ３、Ｅｕ３およびＥｌ３から最も相関度の高い評価関数値（最小の評価関数値を求め、対応の予測モード、変位ベクトルＭＶ、および評価関数値ＥＶを出力する。

【０３５８】図１３９は、図１３８に示す粗探索演算部４０２および比較選択部４０４の構成を概略的に示す図である。図１３９において、粗探索演算部４０２は、外部のプレーンメモリに格納された参照画像データからサーチウィンドウブロックサイズの画素データを格納するメモリセルアレイ４１２と、メモリセルアレイへの画素データ書込時のライトアドレスポインタを発生するライトアドレス発生器４１４と、メモリセルアレイ４１２からの画素データ読出時にリードアドレスポインタを発生するリードアドレス発生器４１６と、メモリセルアレイ４１２から読出された１行のデータをラッチしかつ転送する出力回路４１８と、テンプレートブロックの画素に対応して行および列のマトリックス状に配列される要素プロセッサを含む要素プロセッサアレイ４１９と、要素プロセッサアレイ４１９の要素プロセッサから出力される差分絶対値をそれぞれ予測モードに従って総和し、各予測モードに対応する評価関数値Σ｜ａ－ｂ｜、Σｕ｜ａ－ｂ｜およびΣｌ｜ａ－ｂ｜を出力する総和部４２０とを含む。要素プロセッサアレイ４１９の要素プロセッサのテンプレートブロック画素データ格納部へはテンプレートブロックデータメモリ４２５からテンプレートブロック画素データが転送される。

【０３５９】比較選択部４０４は、総和部４２０からの評価関数値を並列に受け、各予測モードに対して最適評価点を表わす変位ベクトルｖｆｓｔ、ｖｕｓｔおよびｖｌｓｔを出力する比較部４２２を含む。この比較選択部４０４の構成は先に図３０および図２７を参照して説明した比較部の構成と同様であり、この動作説明は繰返さない。

【０３６０】図１４０は、要素プロセッサアレイ４１９の具体的構成を示す図である。図１４０において要素プロセッサアレイ４１９は、行および列のマトリックス状に配列された要素プロセッサＰＥを含む。図１４０の水平方向に配列された１６個の要素プロセッサＰＥ０～ＰＥ１５は線形プロセッサアレイＬＡを構成する。１６個の線形プロセッサアレイＬＡ０～ＬＡ１５が配列される。要素プロセッサＰＥ（ＰＥ０～ＰＥ１５）は自身が属する線形プロセッサアレイＬＡ（ＬＡ０～ＬＡ１５）において隣接（図１４０の右側）の要素プロセッサへサーチウィンドウブロック画素データおよびテンプレートブロック画素データを転送することができる。

【０３６１】メモリセルアレイ４１２は、１６行１６列に配列されたメモリセルを含む。図１４０においては、メモリセルアレイ４１２において１６行のメモリセルに対応して配置された１６本のワード線ＷＬ０～ＷＬ１５を示す。メモリセルアレイ４１２からのデータ読出時には１行のメモリセルのデータが同時に選択されて出力回路４１８へ与えられる。外部のプレーンメモリからのサーチウィンドウ画素データ書込時においては、装置の外部から画素データを受けるため、先の図１１４ないし図１２７に示すように１本のワード線を選択し、この選択されたワード線へ順次画素データが書込まれる。このメモリセルアレイ４１２は、好ましくはＳＲＡＭセルのアレイで構成される（高速アクセスを可能とするためである）。

【０３６２】図１４１は、要素プロセッサアレイ４１９に含まれる要素プロセッサＰＥ１５の構成を示す図である。図１４１において、要素プロセッサＰＥ１５は、出力回路４１８からのサーチウィンドウ画素データを格納しかつ転送するデータレジスタ４３１と、テンプレートブロックデータメモリ４２５からのテンプレートブロック画素データを格納するためのデータレジスタ４３２と、データレジスタ４３１および４３２の格納データの差分絶対値を求める差分絶対値回路４３３を含む。この図１４１に示す要素プロセッサＰＥ１５の構成は、図２

０に示す要素プロセッサの構成と同じであり、単にデータバッファからサーチウィンドウ画素データを受ける代わりに出力回路４１８から画素データを受ける点が異なっているだけであり、それ以外の動作は同様である。データバッファが設けられていないため、サーチウィンドウ画素データは線形プロセッサアレイＬＡ（ＬＡ０～ＬＡ１５）内においてのみ転送される。次にこの図１４０に示すメモリセルアレイから要素プロセッサへのデータ転送動作について図１４２を併せて参照して説明する。

【０３６３】図１４２（Ａ）に示すように、要素プロセッサアレイ４１９においては、変位ベクトル（－６４，－６４）のサーチウィンドウブロック３５４ａが格納されている状態を考える。このとき図１３９に示すライトアドレス発生器４１４は、行アドレス０から行アドレス７まで順次変化させ、行アドレス０ないし７の各行にサーチエリア３５２内の第－４８行ないし第－４１行の１６個の画素データを格納する。この状態では、メモリセルアレイ４１２においては、変位ベクトル（－５６，－６４）の画素データが格納されている。

【０３６４】変位ベクトル（－６４，－６４）に対する評価関数値の算出が完了すると、リードアドレス発生器４１６からのリードアドレスポインタＲＰが０から７まで順次発生される。行アドレス０には第－４８行の画素データが格納されている。１行の画素データを読出して出力回路４１８を介して要素プロセッサアレイ４１９へ伝達することにより、第－４８行ないし第－４１行の画素データが順次要素プロセッサアレイ４１９内に格納される。このとき要素プロセッサアレイ４１９内においては、サーチウィンドウブロック画素データの隣接要素プロセッサ間の転送が実行されている。したがって行アドレス７の第－４１行の画素データが要素プロセッサアレイ４１９に格納されたとき、要素プロセッサアレイ４１９においては変位ベクトル（－５６，－６４）の画素データが全て格納されている。１行の画素データの要素プロセッサアレイ４１９への格納と同時に１行ずつ画素データが要素プロセッサアレイ内をシフトされているためである。

【０３６５】このメモリセルアレイ４１２からの画素データ読出と並行してライトアドレス発生器４１４からのライトアドレスポインタＷＰが行アドレス８から１５へ変化し、第－４０行ないし第－３３行の画素データが格納される。この変位ベクトル（－５６，－６４）のサーチウィンドウブロック３５４ｂに対する評価関数値の算出が完了すると、次いでメモリセルアレイ４１２からはリードアドレスポインタＲＰに従って行アドレス８ないし１５の画素データが行単位で順次読出されて要素プロセッサアレイ４１９へ与えられる。これにより要素プロセッサアレイ４１９においては変位ベクトル（－４８，－６４）に対応する画素データが格納される。このデータ読出動作と並行してライトアドレスポインタＷＰに従って行アドレス０ないし７の位置に第－３２行ないし第－２５行の画素データが格納される。以降、上述の動作を繰返すことによりサーチエリア３５２内の代表評価点に対する評価関数値の算出が実行される。

【０３６６】画素データ読出を行単位で実行し、要素プロセッサアレイ４１９内へシフトさせることによりサーチウィンドウブロックの半分の画素データを読出すだけで次の代表評価点に対するサーチウィンドウブロックを要素プロセッサアレイ４１９内に格納することができる。

【０３６７】図１４３は比較選択部４０４からの最適代表評価点に対応する変位ベクトルｖｆｓｔ、ｖｕｓｔおよびｖｌｓｔに従って密探索演算部４０６に対する周辺領域画素データを転送するための構成を示す図である。図１４３において、データ転送制御系は、粗探索演算部４０２における最適代表評価点検出動作完了を検出すると（サーチエリア内の代表評価点の数は予め定められており、粗探索動作サイクルを規定するクロック信号をカウントすることにより容易に識別することができる）、読出制御回路４５２を活性化するとともに、第１の全探索演算部用メモリ４５５ａ、第２の全探索演算部用メモリ４５５ｂ、および第３の全探索演算部用メモリ４５５ｃのデータ書込タイミングおよびアドレスを発生する入出力制御回路４５４を含む。

【０３６８】読出制御回路４５２は、活性化時には、最適代表評価点に対応する変位ベクトルｖｆｓｔ、ｖｕｓｔおよびｖｌｓｔを順次デコードし、この変位ベクトル周辺領域を含むアドレスを発生し、参照画像メモリ４５０から変位ベクトルｖｆｓｔ、ｖｕｓｔおよびｖｌｓｔそれぞれの周辺領域画素データを順次読出す。この参照画像メモリ４５０から読出された周辺領域画素データは入出力制御回路４５４の制御の下にメモリ４５５ａ、４５５ｂおよび４５５ｃへ順次格納される。このメモリ４５５ａ～４５５ｃへの画素データ格納動作と並行してメモリ４５５ａ～４５５ｃにおいてはその格納データの読出が実行され、それぞれ対応の第１の全探索演算部、第２の全探索演算部および第３の全探索演算部へ周辺領域画素データが転送される。メモリ４５５ａ～４５５ｃは周辺領域の画素データすべてを格納する（３２×３２評価点画素データであり、それほど大きな記憶容量は要求されない）。この全探索演算部用メモリ４５５ａ～４５５ｃへ格納されたデータは、密探索演算部４０６における探索動作に同期してその格納画素データが読出されて対応の全探索演算部の入力部へ伝達される。第１の全探索演算部４１０、第２の全探索演算部４１２、および第３の全探索演算部４１４の構成は先に図２８を参照して説明した動きベクトル検出装置のそれと同じである。単にサーチエリアのサイズが異なっているだけであり、同様の動作を行なって複数の予測モード（フィールド予測モード、上サブテンプレートブロック予測モード、およ

び下サブテンプレートブロック予測モード）に従って評価関数値算出を実行する。これにより、それぞれ複数の予測モードに対応する変位ベクトルが出力される。

【０３６９】比較選択部４０８は、この密探索演算部４０６から与えられる９種類の最適変位ベクトルに対応する評価関数値の最小値を求め、この最小評価関数値およびそれに対応する変位ベクトルを動きベクトルＭＶおよび評価関数値ＥＶとして出力する。この構成は、単に９入力のうちの最小値を求めるだけであり、容易に実現することができる。

【０３７０】［階層探索方式動きベクトル検出装置の第２の実施例］図１４４は、階層探索方式動きベクトル検出装置の第２の実施例の構成を示す図である。図１４４に示す動きベクトル検出装置は、密探索演算部４０６が、３つの予測モードそれぞれについての評価関数値および最適変位ベクトルのみ、すなわち３種類の評価関数値および最適変位ベクトルのみを出力する点が図１３８に示す構成と異なっている。

【０３７１】密探索演算部４０６は、並列に設けられた第１、第２および第３の全探索演算部４６０、４６２および４６４を含む。第１の全探索演算部４６０は、比較選択部４０４から与えられるフィールド予測モードに従って決定された最適代表評価点に対応する変位ベクトルｖｆｓｔを中心評価点としてその周辺領域に含まれるすべての評価点に対してフィールド予測モードに従ってブロックマッチング処理を行ない、最適評価点に対応する変位ベクトルｖｆおよびその評価関数値Ｅｆを求める。

【０３７２】第２の全探索演算部４６２は、比較選択部４０４から与えられる上サブテンプレートブロック予測モードに従って決定された最適代表評価点に対応する変位ベクトルｖｕｓｔを受け、その最適代表評価点を中心とする周辺領域に含まれるすべての評価点に対し上サブテンプレートブロック予測モードに従って最適評価点を求め、この求めた最適評価点に対応する変位ベクトルＶｕ、および評価関数値Ｅｕを出力する。

【０３７３】第３の全探索演算部４６４は、比較選択部４０４からの下サブテンプレートブロック予測モードに従って決定された最適代表評価点に対応する変位ベクトルｖｌｓｔに従って、この変位ベクトルｖｌｓｔが示す評価点を中心とする周辺領域に含まれる評価点すべてに対し下サブテンプレートブロック予測モードに従って最適評価点を求め、対応の変位ベクトルＶｓ、および評価関数値Ｅｓを出力する。

【０３７４】比較選択部４０８は、密探索演算部４０６から出力される最適変位ベクトルＶｆ、ＶｕおよびＶｓならびに評価関数値Ｅｆ、ＥｕおよびＥｓを受け、これら３つの評価関数値Ｅｆ、ＥｕおよびＥｓのうち最も高い類似度を示す（最も小さい）評価関数値を求め、その最小評価関数値ＥＶおよび対応の変位ベクトルを動きベクトルＭＶとして出力する。

【０３７５】図１４５は、第１の全探索演算部４６０の構成を説明するための図である。図１４５（Ａ）に示すように、フィールド予測モードに従って評価関数値を決定する場合、テンプレートブロック４７２のサイズが１６画素×１６画素であり、サーチウィンドウブロック４７０のサイズも同様１６画素×１６画素となる。サーチウィンドウブロック４７０およびテンプレートブロック４７２の対応の画素の差分絶対値の総和が求められる。

【０３７６】図１４５（Ｂ）に示すように、探索範囲は３２画素×３２画素である（探索範囲が水平および垂直方向±８のとき）。サーチウィンドウのサイズは３２画素×１６画素である。第１の全探索演算部４６０は、図４に示す構成と同様のプロセッサアレイを含む。要素プロセッサ群は１６行１６列に配列され、データバッファ群が８行１６列の画素データを格納するように配列される。順次サーチエリア４７４内のサーチウィンドウ画素データをシフトさせて評価関数値の算出を行なう。

【０３７７】図１４６は、第２の全探索演算部の構成を概略的に示す図である。第２の全探索演算部４６２はテンプレートブロックは８画素（行）×１６画素（列）のサイズを備える。１６画素×１６画素のテンプレートブロックの上半分のブロックに含まれる画素データのみが用いられるためである。この場合、サーチウィンドウブロックも同様８画素（行）×１６画素（列）のサイズとなる。探索範囲が３２画素×３２画素の場合、上サブテンプレートブロック予測モードにおいては、３２画素×３２画素の領域において、下側８行の画素データは利用されない（１６画素×１６画素のブロックを単位として評価関数値の算出が行なわれるためである）。すなわちサーチウィンドウ４７６において最下部にサーチウィンドウブロック４７７が到達した場合、この３２画素×３２画素の探索範囲における残りの領域４７８の画素データは利用されない。したがってこの上サブテンプレートブロック予測モードにおけるサーチエリア４７９は、２４画素行×３２画素列のサイズを備える。第２の全探索演算部４６２は、図４に示す構成と同様の構成を備えるが、要素プロセッサ群が８行１６列に配列され、データバッファ群が１６行１６列の画素データを格納するように配置される。このサーチエリア４７９内の画素データを順次シフトして第２の全探索演算部４６２は評価関数値の算出、および最適変位ベクトルの検出を行なう。

【０３７８】図１４７は第３の全探索演算部の構成を説明するための図である。下サブテンプレートブロック予測モードにおいては、１６画素×１６画素のテンプレートブロックのうちの下半分のブロックに含まれる画素データが利用される。したがってサーチウィンドウブロック４８２のサイズは８画素行×１６画素列となる。３２画素行×３２画素列の周辺領域の画素データのうち、上部８行の画素データは利用されない。たとえば図１４７においてブロック４８２および４８３の位置に対する評

価関数値算出が行なわれる場合、ブロック４８３の画素データは利用されない。ブロック４８２に含まれる画素データのみが利用される。したがってこの下サブテンプレートブロック予測モードにおけるサーチエリア４８４は、３２画素行×３２画素列のうち、下側２４行×３２画素の画素データで構成される。第３の全探索演算部４６４はしたがって図４に示す構成と同様の構成を備えるが、８行１６列に配列される要素プロセッサ群と、１６行１６列に配列される画素データを格納するデータバッファ群とを含む。サーチエリア４８４内の画素データを順次１画素分ずつシフトして評価関数値の算出を行なう。

【０３７９】この第２の実施例に示す構成においては、密探索演算部４０６においては、３つの演算部が設けられているが、これらの３つの演算部はそれぞれ異なる予測モードに従って最適変位ベクトルを算出しており、装置規模を低減することができるとともに、並列に複数の予測モードに従って最適変位ベクトルを検出し、これらの検出された最適変位ベクトルに従って動きベクトルを決定しているため、高速で最適な動きベクトルを検出することができ、最適を予測画像を得ることができる。

【０３８０】なお、全探索演算部４６０、４６２および４６４へのそれぞれのサーチエリアの画素データの伝達は図６４に示す構成と同様の構成を利用することができる。

【０３８１】［階層探索方式動きベクトル検出装置の第３の実施例］図１４８は階層探索方式動きベクトル検出装置の第３の実施例を示す図である。図１４８において、比較選択部４９２は、粗探索演算部４０２から与えられる評価関数値ｅｆ、ｅｕおよびｅｌを対応の変位ベクトルｖｆ、ｖｕおよびｖｌを受け、最適代表評価点を求めるとともに、最適予測モードを検出し、この検出された最適予測モードに対応する最適代表評価点に対応する変位ベクトルｖｓｔおよび評価関数値ｅｓｔを出力する。

【０３８２】密探索演算部４０６は、１つの全探索演算部４９４を含む。全探索演算部４９４は、比較選択部４９２から与えられた最適代表評価点に対応する変位ベクトルｖｓｔに従ってその周辺領域に含まれる画素データ（または評価点）のすべてに対しフィールド予測モード、上サブテンプレートブロック予測モード、および下サブテンプレートブロック予測モードに従って最適変位ベクトルＶｆ、ＶｕおよびＶｌと対応の評価関数値Ｅｆ、ＥｕおよびＥｌを決定する。この全探索演算部４９４の構成としては、図１に示す動きベクトル検出装置の構成を利用することができる。

【０３８３】比較選択部４０８は、この全探索演算部４９４から与えられた評価関数値Ｅｆ、ＥｕおよびＥｌから最適予測モードを決定し、その最適予測モードに対応する変位ベクトルおよび評価関数値を動きベクトルＭＶ、および評価関数値ＥＶとして出力する。

【０３８４】図１４９は図１４８に示す比較選択部４９２の構成の一例を示す図である。図１４９において、比較選択部４９２は、フィールド予測モードに従って算出された評価関数値ｅｆおよび対応の変位ベクトルｖｆを受けて最適代表評価点に対応する変位ベクトルｖｆｓｔおよび評価関数値ｅｆｓｔを出力する第１の比較部５０２と、上サブテンプレートブロック予測モードに従って算出された評価関数値ｅｕおよび対応の変位ベクトルｖｕを受け、最適代表評価点に対応する評価関数値ｅｕｓｔおよび対応の変位ベクトルｖｕｓｔを出力する第２の比較部５０４と、下サブテンプレートブロック予測モードに従って算出された評価関数値ｅｌおよび対応の変位ベクトルｖｌを受け、最適代表評価点に対応する変位ベクトルｖｌｓｔおよび対応の評価関数値ｅｌｓｔを出力する第３の比較部５０６を含む。第１、第２および第３の比較部５０２、５０４および５０６の構成は図２７に示す構成と同様であり、最小の評価関数値が最適代表評価点に対応する評価関数値として決定される。

【０３８５】比較選択部４９２はさらに、第２および第３の比較部５０４および５０６から出力される最適評価関数値ｅｕｓｔおよびｅｌｓｔを加算する加算回路５０８と、加算回路５０８の出力と第１の比較部５０２からの最適評価関数値ｅｆｓｔを比較する比較器５１０と、比較器５１０の出力が最適評価関数値ｅｆｓｔが加算回路５０８の出力よりも小さいことを示すとき第１の比較部５０２からの最適評価関数値ｅｆｓｔおよび変位ベクトルｖｆｓｔを伝達するゲート回路５１４と、比較器５１０の出力が加算回路５０８の出力の方が大きいことを示すとき活性化され、第２および第３の比較部５０４および５０６からの評価関数値ｅｕｓｔおよびｅｌｓｔを比較する比較器５１２と、比較器５１２の出力に従って第２の比較部５０４からの評価関数値ｅｕｓｔおよび変位ベクトルｖｕｓｔを第３の比較部５０６からの評価関数値ｅｌｓｔおよび変位ベクトルｖｌｓｔの一方を通過させるセレクタ５１６を含む。

【０３８６】図１４９に示す比較選択部４９２は、以下の判断動作を行なっている：

```
ｉｆ　　ｅｆｓｔ＜ｅｕｓｔ＋ｅｌｓｔ
ｅｓｔ＝ｅｆｓｔ、ｖｓｔ＝ｖｆｓｔ
ｅｌｓｅ　ｉｆ　　ｅｕｓｔ＜ｅｌｓｔ
ｅｓｔ＝ｅｕｓｔ、ｖｓｔ＝ｖｕｓｔ
ｅｌｓｅ　　ｅｓｔ＝ｅｌｓｔ、ｖｓｔ＝ｖｌｓｔ
```

図１４８に示す比較選択部４０８も図１４９に示す比較選択部４９２と同様の構成を備える。すなわち図１４８に示す比較選択部４０８は、図１４９に示す構成から入力段の比較部５０２、５０４および５０６を除いた構成を備える。

【０３８７】この第３の実施例の動きベクトル検出装置の場合、密探索演算部において用いられる全探索演算部

は１つであり、装置規模を低減することができる。

【０３８８】［階層探索方式動きベクトル検出装置の第４の実施例］図１５０は、階層探索方式動きベクトル検出装置の第４の実施例の構成を示す図である。図１５０に示す動きベクトル検出装置においては、比較選択部５２２は、粗探索演算部４０２から与えられた評価関数値ｅｆ、ｅｕおよびｅｌと対応の変位ベクトルｖｆ、ｖｕおよびｖｌを受けて最適代表評価点に対応する変位ベクトルｖｓｔおよびその最適代表評価点を与えた予測モードを示す最適予測モード指示データｐｓｔを出力する。

【０３８９】密探索演算部４０４は、１つの全探索演算部５２０を含む。全探索演算部５２０は、比較選択部５２２から与えられた最適代表評価点に対応する変位ベクトルｖｓｔを中心とする周辺領域をサーチエリアとして全探索動作を行なって動きベクトルを検出する。このとき全探索演算部５２０は、比較選択部５２２から与えられた最適予測モード指示データｐｓｔが指定する予測モードに従ってのみブロックマッチング処理を実行して動きベクトルを決定する。比較選択部５２２の構成としては、図１５０に示す比較選択部４９２の構成を利用することができる。すなわち、図１５０に示す第１の比較部５０２、第２の比較部５０４、および第３の比較部５０６は、それぞれを示す識別ビットを予測モード指定データとして出力する構成が追加されればよい。これらの予測モード識別ビットはゲート回路５１４およびセレクタ５１６へ与えられて対応の評価関数値および変位ベクトルとともに選択されて出力される。これにより容易に最適予測モード指示データを生成することができる。

【０３９０】全探索演算部５２０は、最適予測モード指定データｐｓｔに従ってその従うべき予測モードが決定される。全探索演算部５２０の構成としては、図１に示す動きベクトル検出装置の構成をそのまま利用することができる。すなわち、最適予測モード指示データｐｓｔに従って、作成された３つの予測モードに従う変位ベクトルおよび評価関数値から１つを選択する構成が利用されればよい。またこれに代えて、全探索演算部５２０の構成としては図１３８に示す構成を利用することができる。この図１３８に示す密探索演算部の構成を利用する場合には、最適予測モード指示データｐｓｔに従って第１ないし第３の全探索演算部のうちの１つが動作状態とされる。全探索演算部５２０からはまた予測モード指示データＰｓが出力される。この場合最適予測モード指示データｐｓｔが利用されてこの予測モード指示信号Ｐｓが発生される。

【０３９１】なお、第１ないし第３の実施例の階層探索方式動きベクトル検出装置においては図面には予測モード指示データＰｓが出力されるようには示していない。しかしながらこの予測モード指示データＰｓが各実施例において出力されるように構成されてもよい。

【０３９２】また比較選択部５２２が最適予測モードを決定し、その最適予測モードに従って決定された最適変位ベクトルを検出する際に用いられる判別方法としては図１４９に示す比較選択部と別の判別ステップが利用されてもよい。

【０３９３】さらに密探索演算部４０４においては、整数精度での動きベクトルの検出が実行されている。ハーフペル精度などの分数精度での動きベクトルを検出する構成が利用されてもよい。これらの分数精度での動きベクトルを検出する構成としては、たとえば本願発明者のグループが既に出願している特願平５－１０５４３２号の構成または図８６に示す構成を利用することができる。さらに上記第１ないし第４の階層探索方式動きベクトル検出装置においては、フィールド単位で符号化すなわち予測画像の検出を行なう場合の動きベクトル検出を行なっている。この場合フレーム単位での符号化を行なう構成が利用されてもよい。この場合には、先に図１以降において説明した全探索方式の動きベクトル検出装置においてフレーム予測モード、奇数フィールド予測モード、および偶数フィールド予測モードに従って動きベクトルを検出する構成を用いることができる。

【０３９４】さらに階層探索方式動きベクトル検出装置において粗探索演算部が探索を行なう代表評価点の位置は８画素ごとではなく、他の条件に従って評価点が間引かれて代表評価点が決定されてもよい。また密探索演算部の探索範囲は間引かれて評価点を最少限含めばよく、そのサイズは任意である。

【０３９５】以上のように、複数の予測モードについて階層探索を行なって動きベクトルを検出するように構成したため、広範囲の領域をサーチエリアとして少ないハードウェア量または演算回数で高精度で探索して動きベクトルを検出することができ、正確に予測画像を作成することができ、高精度の符号化を実現することができる。

【０３９６】

【発明の効果】請求項１に係る発明においては、評価手段が複数の予測モードそれぞれに対応する評価関数値を並列に算出し、動きベクトル決定手段がこの評価手段の出力に従って各予測モードに対する動きベクトルを同時に決定するように構成したため、複数の予測モードに従って動きベクトルをハードウェア量を増大させることなく高速で検出することが可能となる。

【０３９７】請求項２に係る発明においては、評価手段が現画像ブロック（テンプレートブロック）の各画素に対応して配置されたプロセッサを有し、これらのプロセッサが現画像ブロックの画素と対応の参照画像ブロック（サーチウィンドウブロック）の画素との所定の演算（評価関数の成分値を求める）を行なって評価関数値を出力するように構成したため、複数の予測モードに共通の演算を１つの回路構成を用いて実行することができ、装置規模を低減することが可能となる。

【０３９８】請求項３に係る発明においては、複数の要素プロセッサは一方方向に沿ってのみ格納画素データを転送するように構成したため、１回のサーチウィンドウ画素データの転送動作によりサーチウィンドウブロックを書換えることができ、効率的に評価関数値を算出することができるとともに画素データ転送時における電流消費を低減することができる。

【０３９９】請求項４に係る発明においては、奇数フィールド予測モード、偶数フィールド予測モードおよびフレーム予測モードに従って動きベクトルを検出するように構成したため、フレーム単位での予測画像生成に対し効率的に動きベクトルを決定することができる。

【０４００】請求項５に係る発明においては、上半分ブロック予測モード（上サブテンプレートブロック予測モード）、下半分ブロック予測モード（下サブテンプレートブロック予測モード）およびフィールド予測モードに従って動きベクトルを検出するように構成したため、フィールド単位での符号化時において高速かつ効率的に動きベクトルを検出することができる。

【０４０１】請求項６に係る発明においては、現画像ブロック（テンプレートブロック）画素データを格納するプロセッサを上半分ブロックに属しかつ偶数フィールドに属する画素に対応するプロセッサ群、上半分ブロックに属しかつ奇数フィールドに属する画素に対応するプロセッサ群、下半分ブロックに属しかつ偶数フィールドに属する画素に対応するプロセッサ群、および下半分ブロックに属しかつ奇数フィールドに属する画素に対応するプロセッサ群の４つのグループに分割し、モード指定信号に従ってこれらのプロセッサ群のグループ構成を変更して複数の予測モードに対応するように構成したため、フレーム単位での符号化およびフィールド単位での符号化いずれにも柔軟に対応することのできる汎用性の高い動きベクトル検出装置を実現することができる。

【０４０２】請求項７に係る発明においては、画面形式指定信号に従って入力画像データを画面形式を指定された画面形式に設定した後評価関数値を生成する評価手段へ与えるように構成したため、任意の画面形式の画像データが与えられても、容易に所望の複数の予測モードに従って動きベクトルを検出することができる。

【０４０３】請求項８に係る発明においては、プロセッサアレイ内に配置されたサイドウィンドウブロック画素データの格納のためのバッファ手段の有効格納画素数を変更可能としたため、サーチエリアの大きさを容易に変更することができる。

【０４０４】請求項９に係る発明においては、サーチエリア内の評価点を間引いて代表評価点に対する複数の予測モードに従ってブロックマッチング処理を行なって最適代表評価点を求め、次いでこの求められた最適代表評価点の周辺領域に含まれる評価点すべてを対象としてブロックマッチング処理を行なって動きベクトルを検出するように構成したため、広い範囲をサーチエリアとして最適な予測画像を生成するための動きベクトルを高速で決定することができる。

【０４０５】請求項１０に係る発明においては、全探索を行なう第１の演算手段のサーチエリアが複数の予測モードそれぞれに対して決定され、各決定されたサーチエリア内で複数の予測モードそれぞれに従って評価関数値の算出が行なわれて最適な動きベクトルが検出されるように構成しているため、複数の予測モードに従って最適な動きベクトルを決定することができる。

【０４０６】請求項１１に係る発明においては、最適代表評価点が複数の予測モードに共通に１つ定められ、この最適代表評価点に従って複数の予測モードに従って全探索動作を行なって動きベクトルを検出するように構成したため、装置規模をさらに低減して複数の予測モードに従って広い範囲をサーチエリアとして最適な動きベクトルを容易に検出することができる。

【０４０７】請求項１２に係る発明においては、最適評価点を与える予測モードに従ってこの最適代表評価点を含む周辺領域内の全評価点に対する評価関数値の演算および動きベクトルの検出を行なっているため、さらに装置規模を低減して広い範囲を探索して最適な動きベクトルを決定することができる。

【０４０８】請求項１３に係る発明においては、現画像ブロックの各画素に対応してマトリックス状に配列されるプロセッサを利用して評価関数値成分を生成し、プロセッサアレイの各プロセッサの出力を偶数フィールド、奇数フィールドおよびフレーム内画素に対応するプロセッサ群に分類し、各分類ごとにプロセッサの出力の総和を求め、この総和結果に従って奇数フィールド、偶数フィールドおよびフレームそれぞれに対する動きベクトルを並列に決定するように構成しているため、少ないハードウェア量で高速でかつ効率的に動きベクトルを検出することができる。

【０４０９】請求項１４に係る発明においては、現画像ブロック（テンプレートブロック）の各画素に対応してマトリックス状に配列されるプロセッサを利用して評価関数値成分を並列に生成し、これらのプロセッサの出力をフィールド内画素、現画像ブロックの上半分ブロック（上サブテンプレートブロック）、下半分ブロック（下サブテンプレートブロック）それぞれの画素に対応するプロセッサの出力に分類し、各分類ごとにプロセッサの出力の総和を求めて動きベクトルを検出しているため、少ないハードウェア量で高速かつ効率的にフィールド予測モード、上半分ブロック予測モード（上サブテンプレートブロック予測モード）、および下半分ブロック予測モード（下サブテンプレートブロック予測モード）それぞれに対する動きベクトルを検出することができる。

【０４１０】請求項１５に係る発明においては、現画像ブロックの各画素に対応してマトリックス状に配列され

るプロセッサを利用して参照画像ブロック（サーチウィンドウブロック）と現画像ブロック（テンプレートブロック）の評価関数値成分を並列に生成し、プロセッサマトリックスを上半分ブロックおよび偶数フィールドに属する画素に対応するプロセッサ群、上半分ブロックおよび偶数フィールドに属する画素に対応するプロセッサ群、下半分ブロックおよび偶数フィールドの画素に対応するプロセッサ群、および下半分ブロックおよび奇数フィールドの画素に対応するプロセッサ群に４分割し、モード指定信号に従ってこのプロセッサ群の組合せを変更して３つの予測モードに対応する評価関数値を生成するように構成したため、異なる予測モードに対しても装置構成を変更することなく容易に対応することができ汎用性の高い動きベクトル検出装置を得ることができる。

【０４１１】請求項１６に係る発明においては、入力参照画像データの入力順序と出力順序とが異ならせるように入力部のバッファ手段を構成したため、様々な画面形式の入力参照画像データが与えられても所望の予測モードに従う画面形式に変更してブロックマッチング処理を行なって動きベクトルを検出することができ、汎用性の高い動きベクトル検出装置を得ることができる。

【０４１２】請求項１７に係る発明においては、現画像ブロック（テンプレートブロック）と参照画像ブロック（サーチウィンドウブロック）の各画素に対応して配置されるプロセッサアレイにおいて所定数のプロセッサごとに配置されるバッファ手段の有効格納画素数を変更可能としたため、動きベクトルのサーチエリアを容易に変更することができる。

【０４１３】請求項１８に係る発明においては、まず広範囲のサーチエリア内の代表評価点に対し複数の予測モードそれぞれに対する最適代表評価点を決定し、次いで各予測モードそれぞれに対する最適代表評価点周辺領域のすべての評価点を探索して複数の予測モードそれぞれに対する動きベクトル候補（最適変位ベクトル）を決定し、これらの決定された最適変位ベクトルから動きベクトルを決定するように構成したため、複数の予測モードに従いかつ階層探索方式に従って動きベクトルをハードウェア量を増大させることなく効率的に検出することができる。

【０４１４】請求項１９に係る発明においては、広い範囲のサーチエリア内の代表評価点のみを用いて複数の予測モードに従って最適代表評価点を決定し、次いで各予測モードごとに最適評価点を中心とする周辺領域のすべての評価点に対する探索動作を行なって最適変位ベクトルを算出し、最終的に最適変位ベクトルから動きベクトルを決定するように構成したため、装置規模を増大させることなく高速で複数の予測モードに従って動きベクトルを検出することができる。

【０４１５】請求項２０に係る発明においては、広い範囲内のサーチエリアにおける代表評価点のみを用いて複数の予測モードに従って最適代表評価点を決定し、この最適代表評価点周辺領域の評価点すべてに対する全探索を複数の予測モードに従って行なって各予測モードごとにブロックマッチング処理を行ない最適変位ベクトルを決定し、これらの最適変位ベクトルから動きベクトルを決定しているため、ハードウェア量を増大させることなく高速で最適な動きベクトルを検出することができる。

【０４１６】請求項２１に係る発明においては、広い範囲内の代表評価点のみを用いて複数の予測モードに従ってブロックマッチング処理を行なって最適代表評価点およびこの最適代表評価点を与える最適予測モードを決定し、この最適代表評価点を含む周辺領域に含まれる評価点それぞれに対する探索動作を最適予測モードに従ってブロックマッチング処理を行なっているので、装置規模を増大させることなく複数の予測モードに従って最適な動きベクトルを高速に決定することができる。請求項２２に係る発明においては、Ｑ行Ｐ列の画素に対応して配置される要素プロセッサとＲ個の画素データをそれぞれが格納する各要素プロセッサ列に対応して配置されるデータ格納手段とを備えこの要素プロセッサおよびデータ格納手段を参照画面画像データを一次元的一方方向に沿って転送するようにシストリックアレイ状に配置し、Ａを変更可能な定数としてＡ・（Ｐ・（Ｑ＋Ｒ））サイクル毎に動きベクトルを検出するように構成したため、水平方向における動きベクトル探索範囲をその装置構成を変更することなく変更することができ複数の探索範囲に容易に対応することのできる動きベクトル検出装置を実現することができる。

【０４１７】請求項２３に係る発明に従えば、各々が、Ｐ列・Ｑ行の画素に対応して行列状に配列される要素プロセッサと各々がＲ個の画素データを格納することのできるデータバッファをプロセッサアレイの各列に対応して配置した構成を備えるＡ個のプロセッサアレイを配置し、このＡ個のプロセッサアレイにそれぞれ異なる現画面ブロックの画素データを格納し、Ａ・（Ｑ＋Ｒ）・Ｐサイクル毎に順次プロセッサアレイの格納する現画面ブロック画素データを更新するように構成し、かつＡ個のプロセッサアレイに同一の参照画面画素データを与えるように構成したため、パイプライン的に現画面ブロックについて動きベクトルを検出することができ、高速で動きベクトルを検出することのできる動きベクトル検出装置を実現することができる。

【０４１８】請求項２４に係る発明に従えば、Ｐ列・Ｑ行の画素に対応して行列状に配置される要素プロセッサと要素プロセッサアレイの各列に対応して配置され各々がＲ個の画素データを格納するデータバッファとからなるプロセッサアレイにおいて要素プロセッサを複数のサブブロックに分割し、各サブブロックに対し異なる現画面ブロックの画素データを格納するとともにこれらのサブブロックに同じ参照画面画素データを与えるように構

成し、処理単位サイクル毎に順次サブブロックの現画面ブロックの画素データを更新するように構成したため、現画面画素ブロックがサイズが小さくされた場合においても容易に装置規模を変更することは対応の動きベクトルを検出することができるとともに、複数の現画面ブロックの動きベクトルをパイプライン的に検出することができ、高速で各現画面ブロックの動きベクトルを決定することのできる動きベクトル検出装置を得ることができる。

【０４１９】請求項２５に係る発明に従えば、Ｐ列・Ｑ行の現画面画素に対応して行列状に配置される要素プロセッサそれぞれにおいて第１の現画面画像画素データ格納用の第１のレジスタと第２の現画面画像画素データ格納のための第２のレジスタとを設け、一方のレジスタを用いて評価関数値生成時に他方のレジスタへ別の全画面画像画素データを格納するように構成したため、現画面ブロックの動きベクトル検出をパイプライン的に実行することができ、装置規模を増大させることなく高速で動きベクトルを検出することのできる動きベクトル検出装置を実現することができる。請求項２６に係る発明に従えば、評価関数値の値に従って最適変位ベクトルを動きベクトルと決定する比較部において、同じ評価関数値を与える変位ベクトルに対する優先順位として、探索範囲中央領域に最も高い優先順位を与える優先順位判定基準、探索範囲の水平方向中央部垂直方向最下端部の変位ベクトルに最も高い優先順位を与える優先順位判定基準および探索範囲において水平方向中央部、垂直方向最上端部の変位ベクトルに最も高い優先順位を与える優先順位判定基準の３つを準備し、用いられる用途に応じてこれらの優先順位判定基準のうちの１つを選択するように構成したため、用いられる用途にかかわらず常に真裏に対応する変位ベクトルに近い変位ベクトルを動きベクトルとして決定することができ、より精度の高い動きベクトルを検出することのできる動きベクトル検出装置を実現することができる。

【０４２０】請求項２７に係る発明に従えば、複数の予測モードそれぞれについて整数精度での動きベクトルを検出する整数精度ベクトル決定手段と、この整数精度ベクトル決定手段から与えられる参照画面画像画素データと現画面画像画素データとを受け、決定された整数精度の動きベクトルが複数の予測モードそれぞれについての参照画面ブロック画素データを格納する格納手段と、この格納手段に格納された参照画面ブロック画素データと整数精度ベクトル決定手段から与えられる現画面ブロック画素データとを受けて複数の予測モードそれぞれについて分数精度での動きベクトルを決定するように構成したため、分数精度での動きベクトル決定時に新たに参照画像メモリおよび現画像メモリをアクセスする必要がなく、高速で複数の予測モードそれぞれに従って分数精度での動きベクトルを決定することができ、高速で精度の高い動きベクトルを決定することができる。また、整数精度の動きベクトル決定と分数精度での動きベクトル決定とを並列態様で実行することができ、高速で分数精度および整数精度の動きベクトルを決定することができる。

【０４２１】請求項２８に係る発明に従えば、現画面画像内の所定のサイズのブロックそれぞれに対し、動きベクトル探索範囲の有効領域を設定し、この有効領域内においてのみ動きベクトル探索を行なうように構成したため、必要とされる領域内の変位ベクトルに対してのみ評価を行なって動きベクトルを決定することができ、より最適な動きベクトルを決定することができ、高性能の画像符号化システムを構築することができる。

【図面の簡単な説明】

【図１】　この発明の第１の実施例である全探索方式動きベクトル検出装置の全体の構成を概略的に示す図である。

【図２】　図１に示す動きベクトル検出装置に利用されるテンプレートブロックおよびサーチエリアの大きさを示す図である。

【図３】　図１に示す動きベクトル検出装置のプロセッサアレイ内に含まれる要素プロセッサの構成を概略的に示す図である。

【図４】　図１に示すプロセッサアレイの構成を示す図である。

【図５】　（Ａ）は画面の分割構成およびブロックマッチング処理単位となるマクロブロックの構成を示し、（Ｂ）はテンプレートブロックの構成の一例を示す図である。

【図６】　（Ａ）は偶数フィールドサブテンプレートブロックの構成を示し、（Ｂ）は奇数フィールドサブテンプレートブロックの構成を示す図である。

【図７】　参照画像画面の分割構成およびプロセッサアレイ内に格納されるマクロブロック画素データの存在領域を示す図である。

【図８】　図１に示すプロセッサアレイ内に格納される参照画像画素データの格納状態を例示的に示す図である。

【図９】　図１に示すプロセッサアレイ内の線形プロセッサアレイに格納される参照ブロック画像画素データおよび現画像ブロック画素データの格納状態を例示的に示す図である。

【図１０】　動きベクトル探索範囲および現画像ブロックの具体例を示す図である。

【図１１】　奇数フィールドサブテンプレートブロックまたは偶数フィールドサブテンプレートブロックを用いる際の現画像ブロックおよびサーチエリアの具体例を示す図である。

【図１２】　図１に示すプロセッサアレイ内に格納される参照画像ブロック画素データおよびテンプレートブロ

ック画素データの格納状況を示す図である。
【図１３】　（Ａ）はサーチウィンドウが１列水平方向に移動する際のサーチウィンドウ画素データのプロセッサアレイ内の転送状態を示し、（Ｂ）はプロセッサアレイ内におけるサーチウィンドウ画素データの移動状況を例示的に示し、（Ｃ）はサーチウィンドウ画素データ入力後のプロセッサアレイ内に格納されているサーチウィンドウ画素データのサーチエリア内の分布状態を示す図である。
【図１４】　（Ａ）は偶数フィールドサブテンプレートブロックの奇数フィールドに対する変位ベクトルを例示的に示す図であり、（Ｂ）は奇数フィールドサブテンプレートブロックの偶数フィールドに対する変位ベクトルを例示的に示す図である。
【図１５】　評価関数値算出動作におけるプロセッサアレイ内の格納データおよびサーチエリア内の位置を示す図である。
【図１６】　（Ａ）はサーチウィンドウブロックがサーチウィンドウの最下部に到達したときの状態を示し、（Ｂ）は（Ａ）に示す状態におけるプロセッサアレイ内に格納されるサーチウィンドウ画素データのサーチエリア内の位置を示す図である。
【図１７】　（Ａ）は１つのサーチウィンドウに対する評価関数値算出後のプロセッサアレイ内のサーチウィンドウ画素データ格納状態を示す図であり、（Ｂ）は１６画素シフトイン動作後のプロセッサアレイ内におけるサーチウィンドウ画素データ格納状態を示す図である。
【図１８】　図１７（Ｂ）に示す状態におけるテンプレートブロックとサーチウィンドウブロックとの位置関係を示す図である。
【図１９】　サーチウィンドウブロックがサーチエリアにおいて最終位置に到達したときの状態を示す図である。
【図２０】　図１に示すプロセッサアレイに含まれる要素プロセッサの構成の一例を示す図である。
【図２１】　図２０に示す要素プロセッサの動作を示すタイミング図である。
【図２２】　図１に示す総和部の具体的構成を示す図である。
【図２３】　図１に示す総和部の他の構成を示す図である。
【図２４】　図１に示す総和部のさらに他の構成を示す図である。
【図２５】　図２４に示す総和部の動作を示すタイミング図である。
【図２６】　図２４に示す総和部の動作を例示的に示す図である。
【図２７】　図１に示す比較部の構成を示す図である。
【図２８】　この発明の全探索方式動きベクトル検出装置の第２の実施例の構成を示す図である。
【図２９】　図２８に示す動きベクトル検出装置において用いられるテンプレートブロックおよびサーチウィンドウブロックの構成を示す図である。
【図３０】　図２８に示す総和部の構成を示す図である。
【図３１】　この発明の第３の実施例である全探索方式動きベクトル検出装置において用いられるテンプレートブロックの分割構成を例示的に示す図である。
【図３２】　この発明の全探索方式動きベクトル検出装置の第３の実施例の構成を示す図である。
【図３３】　図３２に示す総和部の構成を概略的に示す図である。
【図３４】　この発明の第４の実施例の動作を説明するための図である。
【図３５】　この発明の第４の実施例における水平方向探索範囲拡張動作を説明するための図である。
【図３６】　この発明の第４の実施例における水平方向探索範囲拡張の具体的例を示す図である。
【図３７】　この発明の第４の実施例の具体的動作シーケンスを示す図である。
【図３８】　この発明の第４の実施例の動きベクトル検出装置の全体の構成を概略的に示す図である。
【図３９】　図３８に示す比較部の構成の一例を示す図である。
【図４０】　図３９に示す比較部の動作を説明するための図である。
【図４１】　図３９に示す比較部の動作を説明するための図である。
【図４２】　図３８に示す比較制御回路の構成の一例を示す図である。
【図４３】　図３８に示す読出制御回路の構成を概略的に示す図である。
【図４４】　この発明の第５の実施例の動きベクトル検出装置の全体の構成を概略的に示す図である。
【図４５】　この発明の第５の実施例の動きベクトル検出装置の動作を説明するための図である。
【図４６】　この発明の第５の実施例の動きベクトル検出装置の具体的動作シーケンスを示す図である。
【図４７】　この発明の第５の実施例において処理時間を４処理単位時間にした場合の動きベクトル検出動作の具体的シーケンスを示す図である。
【図４８】　この発明の第６の実施例における動きベクトル検出装置のプロセッサアレイの構成を示す図である。
【図４９】　図４８で示すプロセッサアレイをテンプレートブロックサイズに応じて再配置した際のプロセッサアレイの構成を概略的に示す図である。
【図５０】　図４９に示すプロセッサアレイにおける単位探索範囲を示す図である。
【図５１】　図４９に示す動きベクトル検出装置の動作

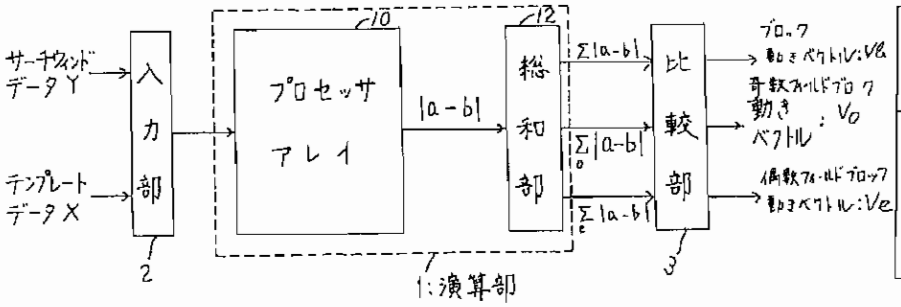
を説明するための図である。
【図５２】　この発明の第６の実施例において用いられる動きベクトル探索範囲の構成を示す図である。
【図５３】　この発明の第６の実施例における処理単位時間におけるプロセッサアレイにおける参照画面画素データの転送態様を示す図である。
【図５４】　図４９に示す動きベクトル検出装置の動きベクトル検出動作の具体的シーケンスを示す図である。
【図５５】　この発明の第６の実施例における動きベクトル検出装置におけるプロセッサアレイ部の配置を示す図である。
【図５６】　図５５に示すセレクタのテンプレートブロック画素データに関連する部分の構成を概略的に示す図である。
【図５７】　図５５に示すセレクタのサーチウィンドウ画素データに関連する部分の構成を概略的に示す図である。
【図５８】　この発明の第６の実施例における動きベクトル検出装置の演算部および比較部の構成を概略的に示す図である。
【図５９】　この発明の第６の実施例において利用されるサブサンプリング画像の構成を示す図である。
【図６０】　この発明の第７の実施例の要素プロセッサの構成を概略的に示す図である。
【図６１】　図６０に示す要素プロセッサの動作を説明するための図である。
【図６２】　この発明の第７の実施例の動きベクトル検出装置の概略構成を示す図である。
【図６３】　この発明の第７の実施例の要素プロセッサの第１の変更例を示す図である。
【図６４】　この発明の第７の実施例の要素プロセッサの第２の変更例およびその動作波形を示す図である。
【図６５】　この発明の第８の実施例における比較部の構成を概略的に示す図である。
【図６６】　図６５に示す優先順位判定回路の第１の優先順位判定基準を示す図である。
【図６７】　図６５に示す優先順位判定回路の第１の優先順位判定基準を示す図である。
【図６８】　図６５に示す優先順位判定回路の第２の判定基準を示す図である。
【図６９】　図６５に示す優先順位判定回路の第２の判定基準を示す図である。
【図７０】　図６５に示す優先順位判定回路の第３の優先順位判定基準を示す図である。
【図７１】　図６５に示す優先順位判定回路の第３の優先順位判定基準を示す図である。
【図７２】　第１の優先順位判定基準を用いる際の動きベクトル検出装置の構成を示す図である。
【図７３】　第２および第３の優先順位判定基準を用いる際の動きベクトル検出装置の構成および対応の優先順位判定基準を示す図である。
【図７４】　図６５に示す優先順位判定回路の構成の一例を概略的に示す図である。
【図７５】　図７４に示す優先順位判定回路の動作を説明するための図である。
【図７６】　図７４に示すセレクタおよびマルチプレクサ切換信号を発生するための構成を概略的に示す図である。
【図７７】　この発明の第９の実施例の動きベクトル検出装置の構成を概略的に示す図である。
【図７８】　図７７に示す動きベクトル検出装置の分数精度演算部の動作を説明するための図である。
【図７９】　分数精度動きベクトル検出時に生成される予測画像の構成を概略的に示す図である。
【図８０】　図７７に示す予測画像メモリおよびテンプレートブロックメモリの構成の一例を示す図である。
【図８１】　図８０に示す予測画像メモリの動作を説明するための図である。
【図８２】　図８０に示す予測画像メモリの動作を示す信号波形図である。
【図８３】　図８０に示す予測画像メモリにおけるブロックサイズ予測モード、奇数フィールド予測モードおよび偶数フィールド予測モードそれぞれに用いられるメモリ部へ与えられるクロック信号の発生態様を示す図である。
【図８４】　図７７に示す１／２画素精度演算部の構成を概略的に示す図である。
【図８５】　図８４に示す分数精度予測画素生成回路の動作を説明するための図であ。
【図８６】　図８４に示す１／２画素精度演算部の具体的構成の一例を示す図である。
【図８７】　この発明の第９の実施例の動作をシーケンスの一例を示す図である。
【図８８】　現画面上のテンプレートブロックの位置を示す図である。
【図８９】　この発明の第１０の実施例の動作原理を説明するための図である。
【図９０】　この発明の第１０の実施例において用いられる単位探索範囲の構成を示す図である。
【図９１】　この発明の第１０の実施例において用いられる上半分禁止信号および下半分禁止信号の効果を説明するための図である。
【図９２】　この発明の第１０の実施例における左半分禁止信号の効果を説明するための図である。
【図９３】　この発明の第１０の実施例における右半分禁止信号の効果を説明するための図である。
【図９４】　現画面左上端のテンプレートブロックにおける探索範囲制限動作を説明するための図である。
【図９５】　現画面左上端のテンプレートブロックにおける探索範囲制限動作を説明するための図である。

【図９６】　現画面左上端のテンプレートブロックにおける分数精度の動きベクトル検出時の動作を説明するためのフロー図である。
【図９７】　現画面上端テンプレートブロックにおける禁止信号の発生態様を示す図である。
【図９８】　現画面上右端のテンプレートブロックにおける禁止信号発生態様および動きベクトル探索範囲制限の動作を説明するための図である。
【図９９】　現画面上右端のテンプレートブロックに対する動きベクトル検出動作を示すフロー図である。
【図１００】　現画面左端テンプレートブロックにおける禁止信号発生態様を示す図である。
【図１０１】　現画面中央部のテンプレートブロックにおける禁止信号発生態様を示す図である。
【図１０２】　現画面右端のテンプレートブロックに対する禁止信号発生態様を示す図である。
【図１０３】　現画面下左端のテンプレートブロックに対する禁止信号発生態様および探索範囲制限態様を説明するための図である。
【図１０４】　現画面下左端のテンプレートブロックの分数精度での動きベクトル検出動作時を説明するフロー図である。
【図１０５】　現画面下端のテンプレートブロックに対する禁止信号発生態様を示す図である。
【図１０６】　現画面下右端のテンプレートブロックに対する禁止信号発生態様および動きベクトル探索範囲制限態様を示す図である。
【図１０７】　この発明の第１０の実施例の動きベクトル検出装置の全体の構成を概略的に示す図である。
【図１０８】　図１０７に示す探索動作制御および整数精度動きベクトル検出部に含まれる比較部の構成の一例を示す図である。
【図１０９】　図１０７に示す探索動作制御部の分数精度に対する探索範囲制御部の構成を概略的に示す図である。
【図１１０】　図１０７に示す分数精度動きベクトル検出部に含まれる演算部および比較部の構成およびそこへ与えられる禁止信号を概略的に示す図である。
【図１１１】　図１０７に示す探索範囲設定部および読出制御回路の構成を概略的に示す図である。
【図１１２】　図１１１に示すテンプレートブロック指定回路および禁止範囲設定回路の効果を説明するための図である。
【図１１３】　この発明に従う動きベクトル検出装置の参照画像画素データ入力部の構成を概略的に示す図である。
【図１１４】　図１１３に示す画面形式変換回路の構成を概略的に示す図である。
【図１１５】　図１１４に示すフレーム／フレーム用アドレス生成器の具体的構成を示す図である。
【図１１６】　（Ａ）は図１１５に示すアドレス発生器のデータ書込時のアドレス発生順序を示し、（Ｂ）は画素データ読出時における図１１５に示すアドレス発生器のアドレス発生シーケンスを示す図である。
【図１１７】　図１１５に示すアドレス発生器の変更例を示す図である。
【図１１８】　図１１５に示すフィールド／フレーム用アドレス生成器の構成の一例を示す図である。
【図１１９】　図１１８に示すアドレス発生器の動作を示すタイミング図である。
【図１２０】　図１１８に示すアドレス発生器のアドレス発生シーケンスを示す図であり、（Ａ）は第１フィールドの画素データ書込時のアドレス発生シーケンスを示し、（Ｂ）は第２フィールドの画素データ書込時のアドレス発生シーケンスを示す図である。
【図１２１】　図１１３に示す画面形式変換回路の他の構成を示す図である。
【図１２２】　図１２１に示すフィールド／フィールド用アドレス生成器のアドレス発生シーケンスを示す図であり、（Ａ）はフィールド画像画素データ書込時のアドレス発生シーケンスを示し、（Ｂ）はフィールド画像画素データ読出時のアドレス発生シーケンスを示す図である。
【図１２３】　図１２１に示すフレーム／フィールド用アドレス生成器のアドレス発生シーケンスを示す図であり、（Ａ）はフレーム画像画素データ書込時のアドレス発生シーケンスを示し、（Ｂ）は画素データ読出時のアドレス発生シーケンスを示す図である。
【図１２４】　図１２１に示すフレーム／フィールド用アドレス生成器の構成の一例を示す図である。
【図１２５】　図１１３に示す画面形式変換回路およびサーチウィンドウバッファメモリのさらに他の構成を示す図である。
【図１２６】　図１２５に示すフレーム／フィールド用アドレス発生器の構成の一例を示す図である。
【図１２７】　図１２６に示すアドレス発生器の動作を示すタイミング図である。
【図１２８】　図１２７に示すアドレス発生器によるフレーム画像からフィールド画像を生成する方法を例示的に示す図である。
【図１２９】　図１１９に示すデータバッファの変更例を示す図である。
【図１３０】　図１２９に示すデータバッファの変更例を示す図である。
【図１３１】　図4に示すデータバッファのさらに他の構成を示す図である。
【図１３２】　図１３１に示すデータバッファの動作を説明するための図である。
【図１３３】　図１３１に示すデータバッファの動作を具体的に説明するための図である。

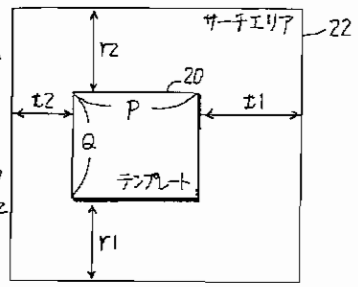
【図１３４】　階層探索方式動きベクトル検出動作において用いられるテンプレートブロックおよびサーチエリアの構成の一例を示す図である。
【図１３５】　階層探索方式動きベクトル検出の第１段階の動作を説明するための図である。
【図１３６】　階層探索方式動きベクトル検出動作における第１段階動作完了後の代表評価点の分布を例示的に示す図である。
【図１３７】　階層探索方式動きベクトル検出における第２段階の動作を説明するための図である。
【図１３８】　階層探索方式動きベクトル検出装置の第１の実施例の構成を概略的に示す図である。
【図１３９】　図１３８に示す粗探索演算部の構成を概略的に示す図である。
【図１４０】　図１３９に示す要素プロセッサアレイおよびメモリセルアレイの構成を概略的に示す図である。
【図１４１】　図１４０に示す最下流要素プロセッサの構成を概略的に示す図である。
【図１４２】　図１４０に示すメモリセルアレイから要素プロセッサアレイへのデータ転送動作を例示的に示す図である。
【図１４３】　図１３８に示す密探索演算部への参照画像画素データ転送制御系の構成を概略的に示す図である。
【図１４４】　この発明の階層探索方式動きベクトル検出装置の第２の実施例の構成を概略的に示す図である。
【図１４５】　図１４４に示す第１の全探索演算部の構成を概略的に示す図である。
【図１４６】　図１４４に示す第２の全探索演算部の構成を概略的に示す図である。
【図１４７】　図１４５に示す第３の全探索演算部の構成を概略的に示す図である。
【図１４８】　全探索方式動きベクトル検出装置の第３の実施例の構成を概略的に示す図である。
【図１４９】　図１４８に示す最適代表評価点を求める比較選択部の構成を概略的に示す図である。
【図１５０】　この発明に従う階層探索方式動きベクトル検出装置の第４の実施例の構成を示す図である。
【図１５１】　従来の画像信号符号化回路の全体の構成を示す図である。
【図１５２】　図１５１に示すソース符号化回路の構成を示す図である。
【図１５３】　画像の動き補償の操作を説明するための図である。
【図１５４】　ブロックマッチング法による動き補償を行なう際のサーチエリアおよびテンプレートブロックの配置例および動きベクトルの関係を示す図である。
【図１５５】　従来の動きベクトル検出装置の全体の構成を概略的に示す図である。
【図１５６】　図１５５に示すプロセッサアレイ内に含まれる要素プロセッサの構成を示す図である。
【図１５７】　図１５５に示す動きベクトル検出装置におけるテンプレートブロックのスキャンおよびサーチウィンドウのスキャン方法を示す図である。
【図１５８】　図１５５に示す動きベクトル検出装置の動作を例示的に示す図である。
【図１５９】　図１５５に示す動きベクトル検出装置の動作を説明するための図である。
【符号の説明】
１，１ａ，１ｂ　演算部、２　入力部、３，３ａ，３ｂ　比較部、１０　プロセッサアレイ、１２　総和部、２０　テンプレートブロック、２２　サーチエリア、ＰＥ　要素プロセッサ、２５－１～２５－Ｍ　データレジスタ、２６－１～２６－Ｎ　データレジスタ、ＬＡ１～ＬＡＰ　線形プロセッサアレイ、２７　差分絶対値回路、１２ａ，１２ｂ，１２ｃ　総和回路、１２ｄ　加算回路、１２ｂａ，１２ｃａ，１２ｅ，１２ｆ　総和回路、１３２　比較器、１３８　カウンタ、１２ｇ，１２ｈ　総和回路、１２２　分類変更部、１２０　総和演算部、１２０ａ，１２０ｂ，１２０ｃ，１２０ｄ　総和回路、１２０ｅ，１２０ｆ，１２０ｇ　加算回路、１２２ａ，１２２ｂ　セレクタ、２０５　サーチウィンドウバッファメモリ、２０７　画面形式変換回路、２１１　セレクタ、２１３　フレーム／フレーム用アドレス生成器、２１５　フィールド／フレーム用アドレス生成器、２００　動きベクトル検出装置、２５２　フレーム／フィールド用アドレス生成器、２５４　フィールド／フィールド用アドレス生成器、２７２　フィールド／フィールド用アドレス発生器、２７４　フレーム／フィールド用アドレス発生器、２７８　書込禁止回路、２８９　リードアドレス発生器、２８２　書込制御回路、２８４　読出制御回路、ＤＬ　データバッファ、ＲＧ１，ＲＧ２　レジスタ、ＲＧａ～ＲＧｎ　レジスタ、３１２　レジスタファイル、３１４　リードアドレス発生器、３１６　外部アドレス発生器、３１９　初期アドレス設定回路、４０２　粗探索演算部、４０４　比較選択部、４０６　密探索演算部、４０８比較選択部、４１０　第１の全探索演算部、４１２　第２の全探索演算部、４１４　第３の全探索演算部、４１２　メモリセルアレイ、４１９　要素プロセッサアレイ、４２０　総和部、４２２　比較部、４２５　テンプレートブロックデータメモリ、４６０　第１の全探索演算部、４６２　第２の全探索演算部、４６４第３の全探索演算部、４９２　比較選択部、４９４　全探索演算部、５２０全探索演算部、５２２　比較選択部、６００　動きベクトル検出装置、６０２参照画像メモリ、６０４　現画像メモリ、６０６　読出制御回路、６０８　比較制御回路、６０７　探索範囲設定部、６１０　レジスタ、６１１　比較器、６１２　レジスタ、６１３　カウンタ、６３０　入力部、６３２　セレクタ、ＰＧ０Ｕ～ＰＧ１５Ｕ，ＰＧ０Ｌ～ＰＧ１５Ｌ

プロセッサグループ、ＰＢＡ，ＰＢＢ，ＰＢＣ，ＰＢＤ　プロセッサブロック、ＳＬａ，ＳＬｂ，ＳＬｃ，ＳＬｄ，ＳＬｅ，ＳＬｆ，ＳＬｇ　セレクタ、ＤＬＡ，ＤＬＢ　データバッファ群、６４４接続制御回路、ＳＷａｕ，ＳＷＸａ，ＳＷＸｂ，ＳＷＸｃ，ＳＷａ１，ＳＷｂ１，ＳＷｂｂ　スイッチング素子、ＳＴａ，ＳＴｂ，ＳＴｄ，ＳＴｃ，ＳＴｅスイッチング素子、６５０ａ～６５０ｄ　総和部、６５２ａ～６５２ｄ　セレクタ、６５４ａ～６５４ｄ　比較部、６５５　加算部、６５６　比較部、６６０ａレジスタ（Ａ）、６６０ｂ　レジスタ（Ｂ）、６６１　マルチプレクサ、６６２，６６３　セレクタ、６６４　レジスタ、６６５　減算器、６６６　絶対値回路、６７２　演算部、６７３　比較部、６７４　制御装置、６７５　参照画像メモリ、６７６　現画像メモリ、６７９　マルチプレクサ、６７７ａ，６７７ｂ　ラッチ、６８０　レジスタ、６８２　比較器、６８４　カウンタ、６８６　レジスタ、６８８　更新制御回路、６９０　優先順位判定回路、ＣＨ１，ＣＨ２，ＣＨ３　動きベクトル検出装置チップ、７０１　セレクタ、７０２　Ｈ絶対値回路、７０３　Ｖ絶対値回路、７０４　Ｖ抽出回路、７０５　－Ｖ抽出回路、７０６セレクタ、７０７　加算器、７０８　セレクタ、７０９　加算器、７１０　マルチプレクサ、７１１，７１２　レジスタ、７１３　比較器、７１５　タイミング信号発生回路、７１６　活性制御回路、７２２　予測画像メモリ、７２４　テンプレートブロックメモリ、７２６，７３８　１／２画素精度演算部、７３０，７３２　比較部、７３４　セレクタ、８０２　分数精度予測画像生成回路、８２０　参照画像メモリ、８２１　現画像メモリ、８２１　現画像メモリ、８３０整数精度動きベクトル検出部、８３２　分数精度動きベクトル検出部、８３４探索範囲設定部、８３６　探索動作制御部、８３８　読出制御回路。

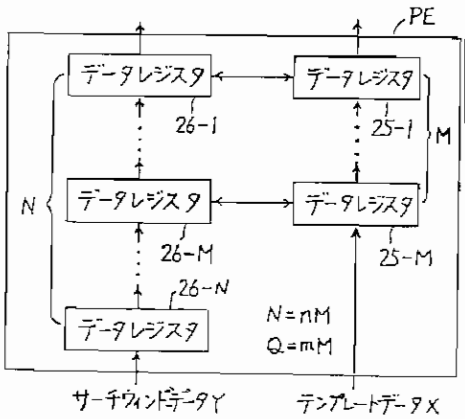
【図１】



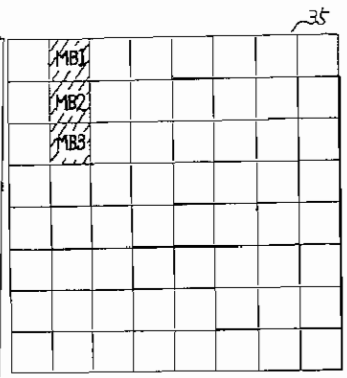
【図２】



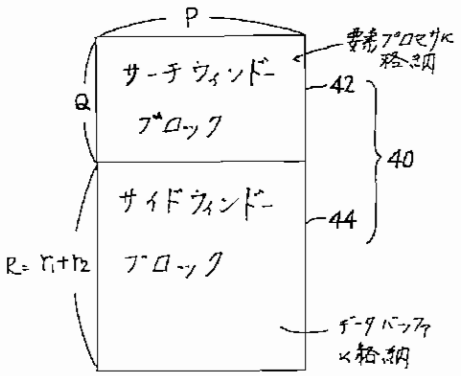
【図３】



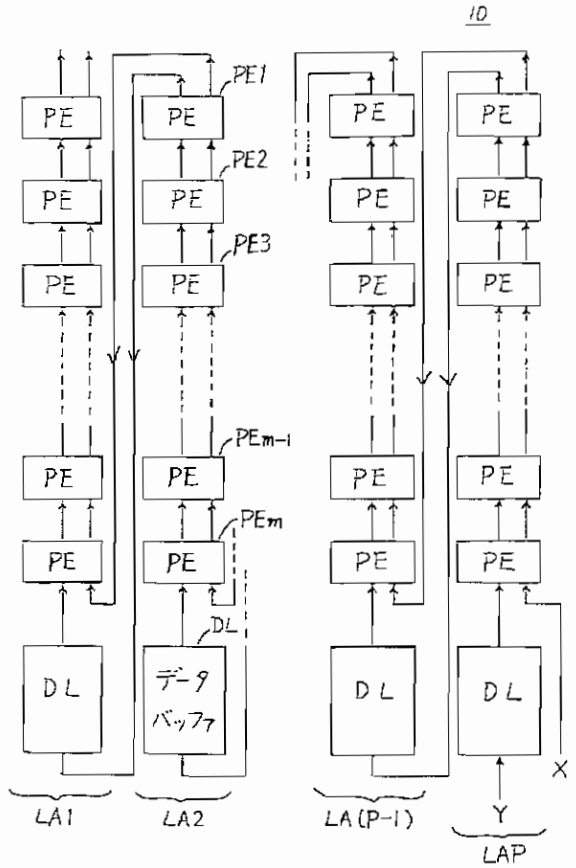
【図７】



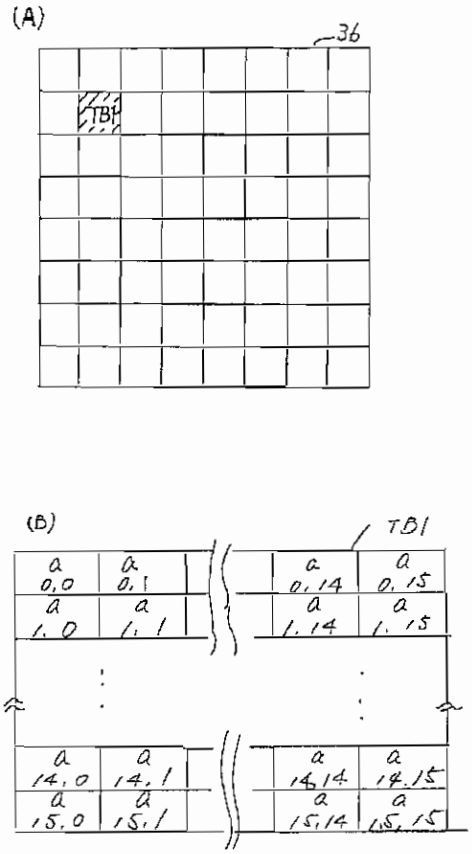
【図８】



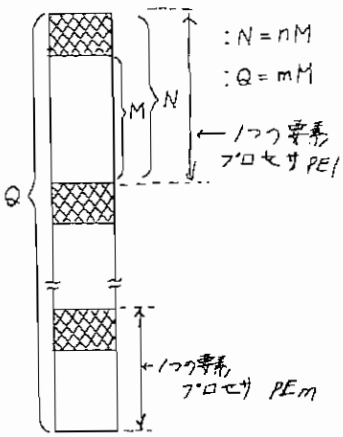
【図４】



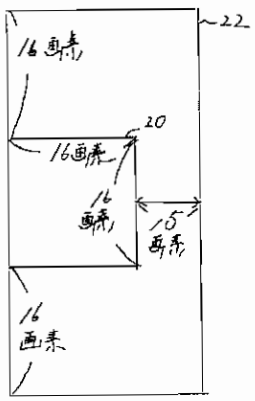
【図５】



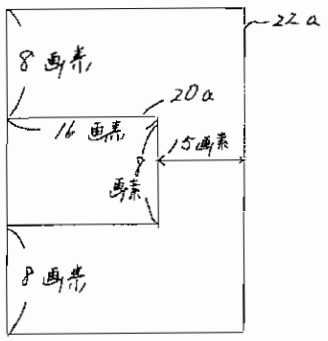
【図９】



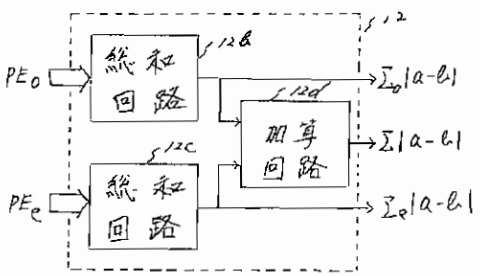
【図１０】



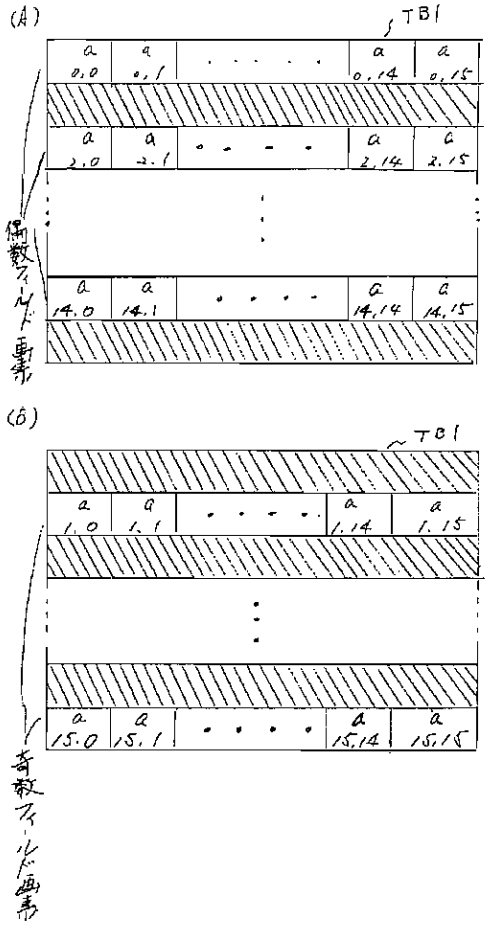
【図１１】



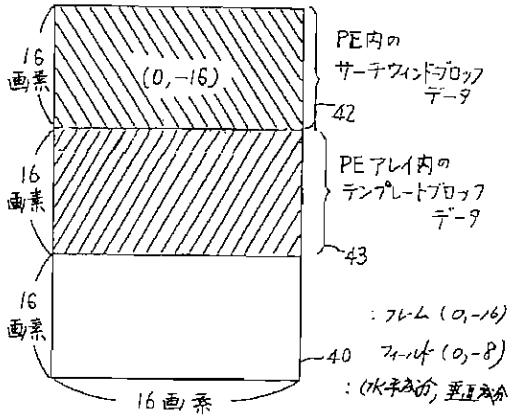
【図２３】



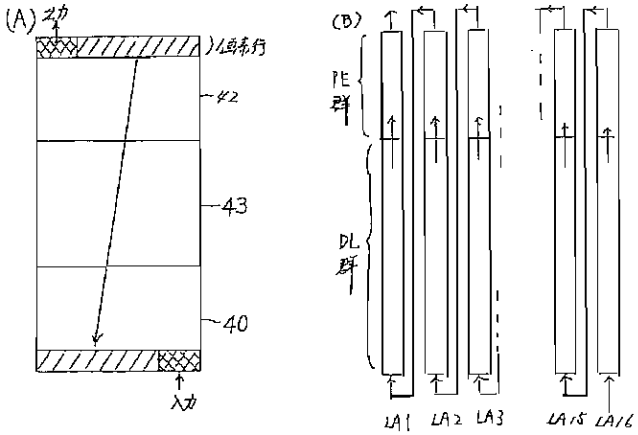
【図６】



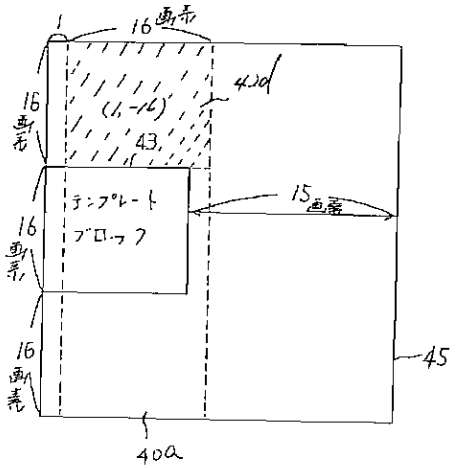
【図１２】



【図１３】



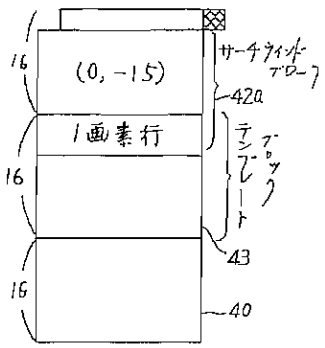
【図１８】



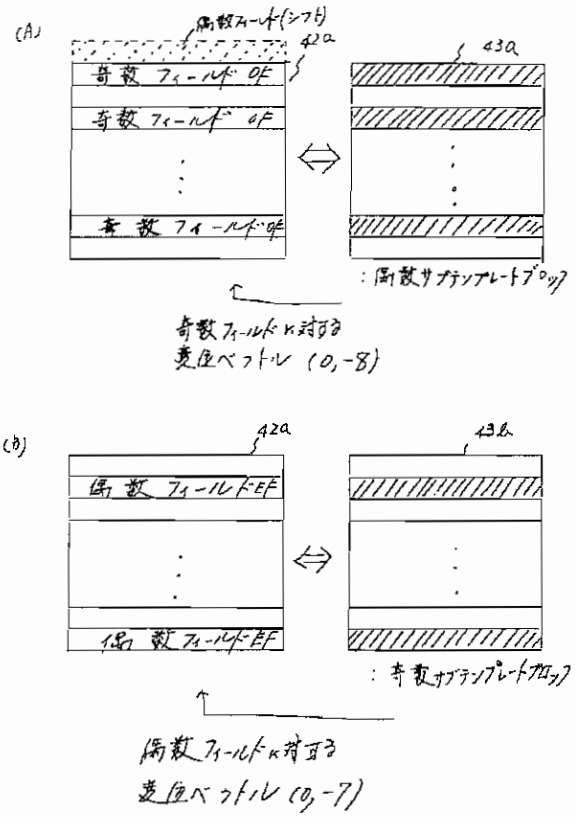
：奇数サブテンプレートブロック
：奇数フィールドに対して (1,-8)

：偶数サブテンプレートブロック
偶数フィールドに対して (1,-8)

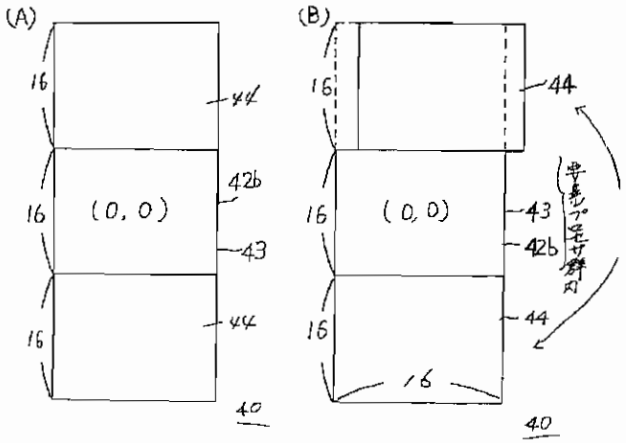
(C)



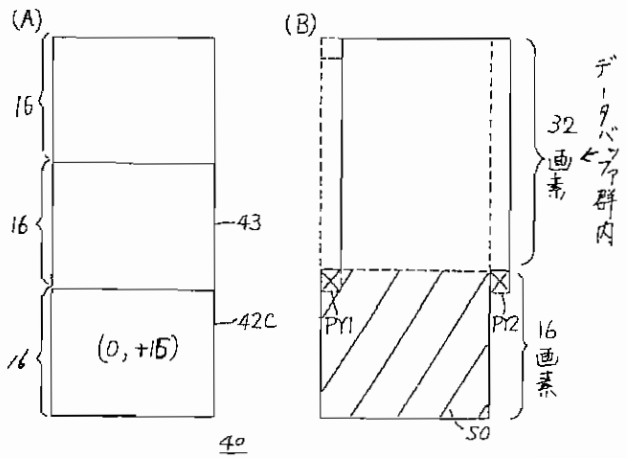
【図１４】



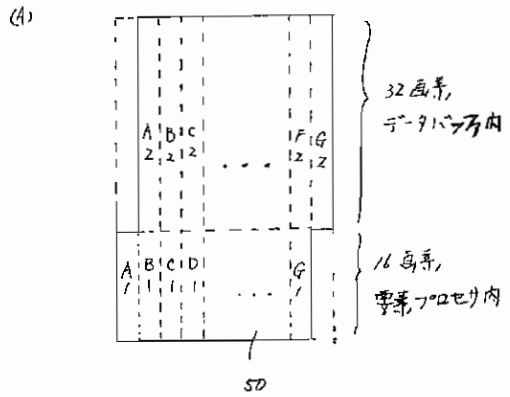
【図１５】

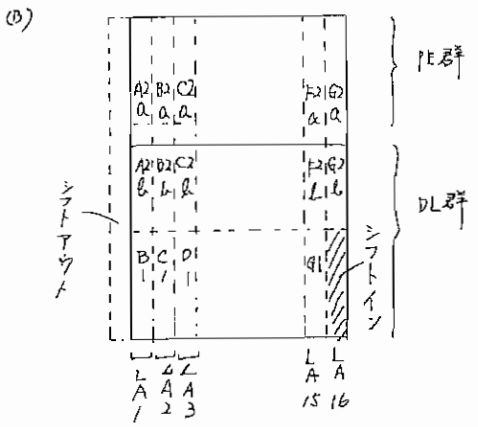


【図１６】

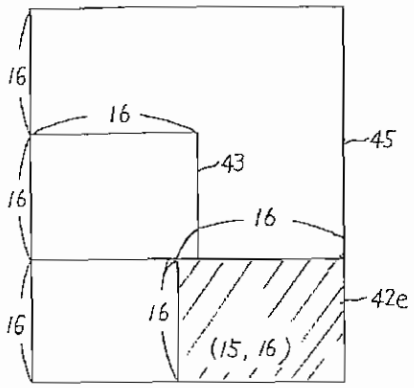


【図１７】

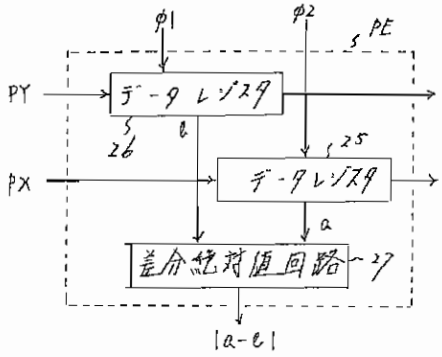




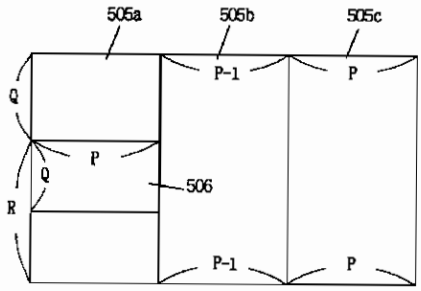
【図19】



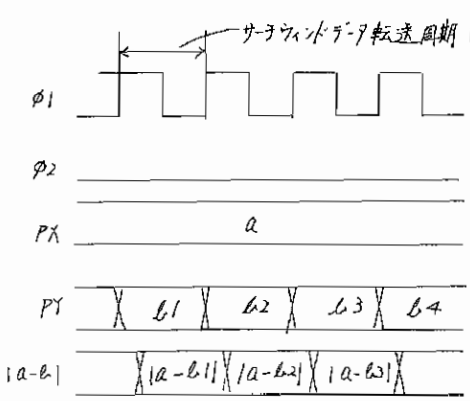
【図20】



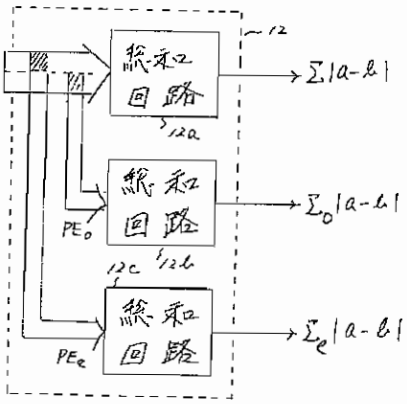
【図35】



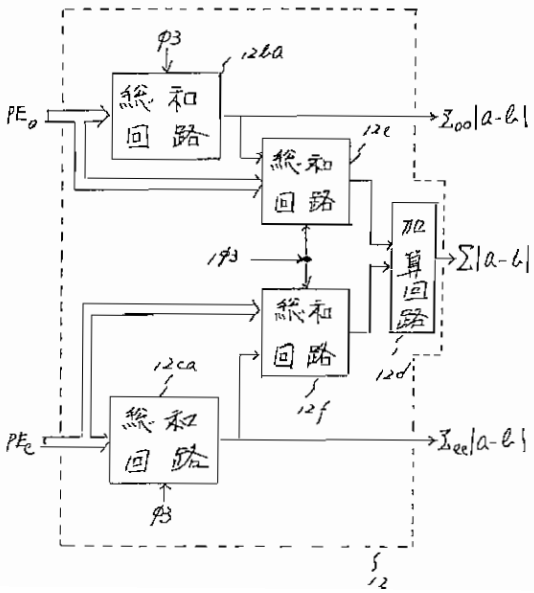
【図21】



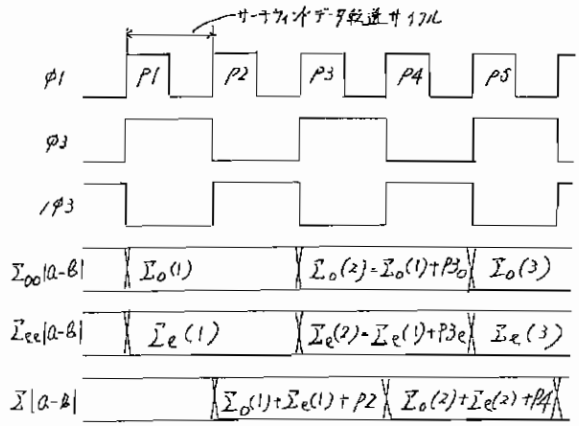
【図22】



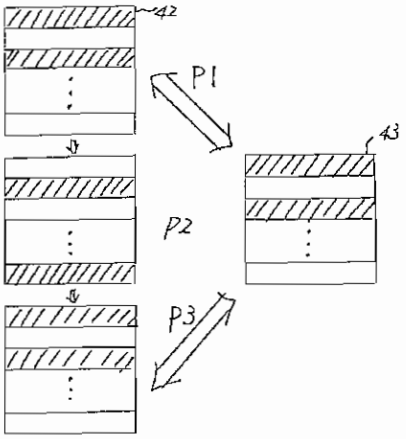
【図24】



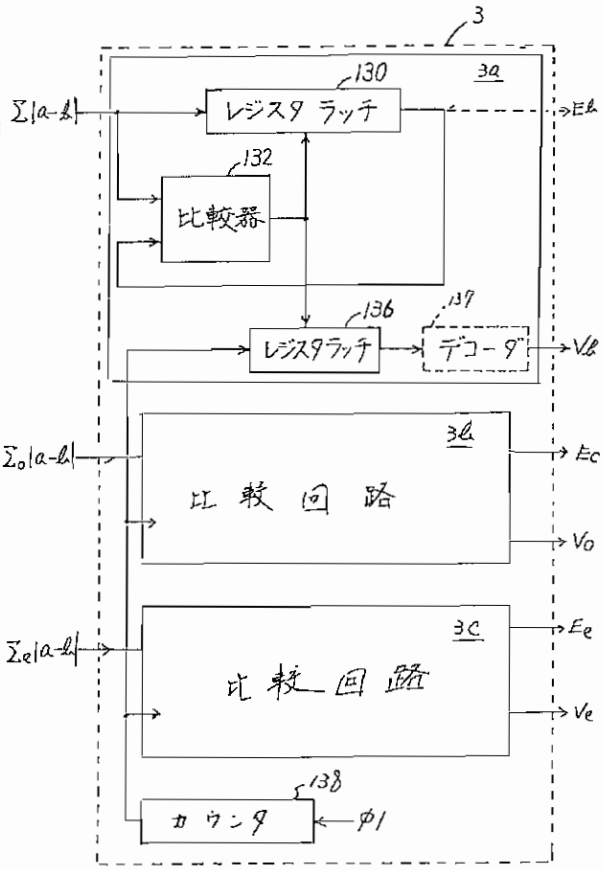
【図25】



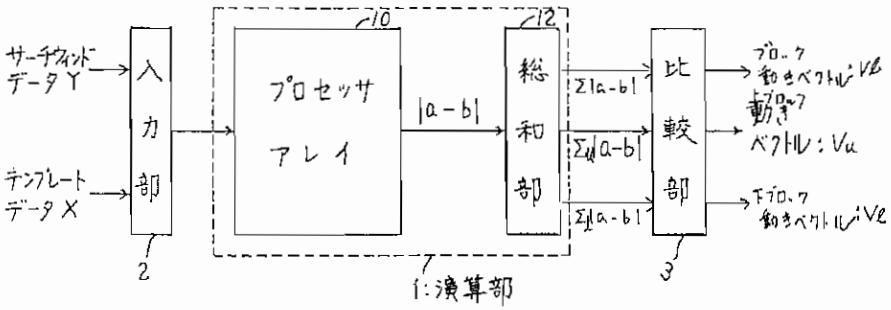
【図26】



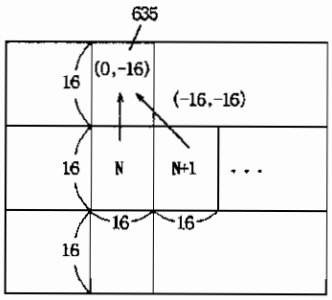
【図27】



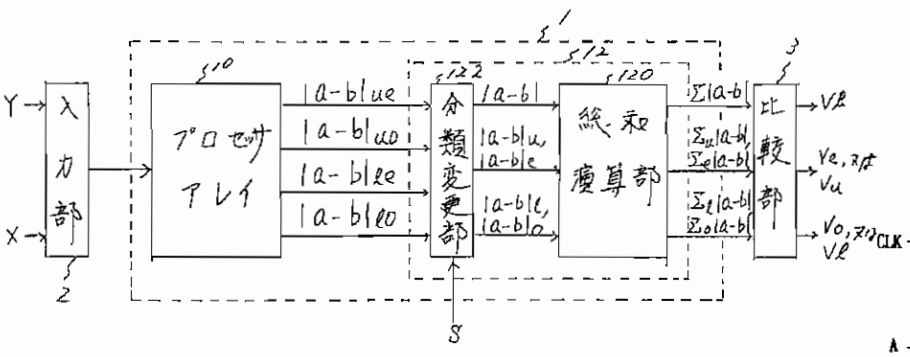
【図28】



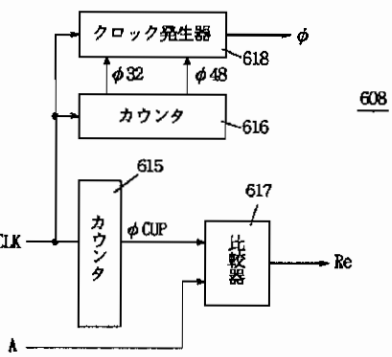
【図45】



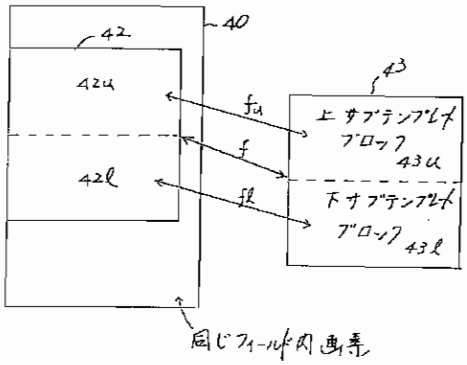
【図32】



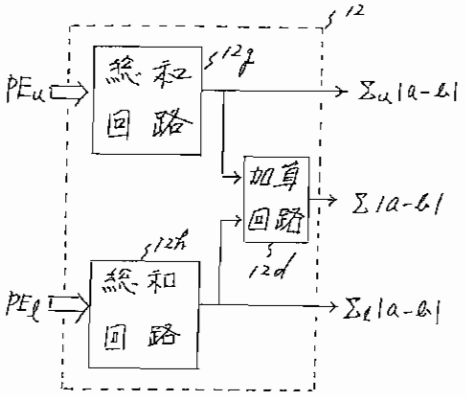
【図42】



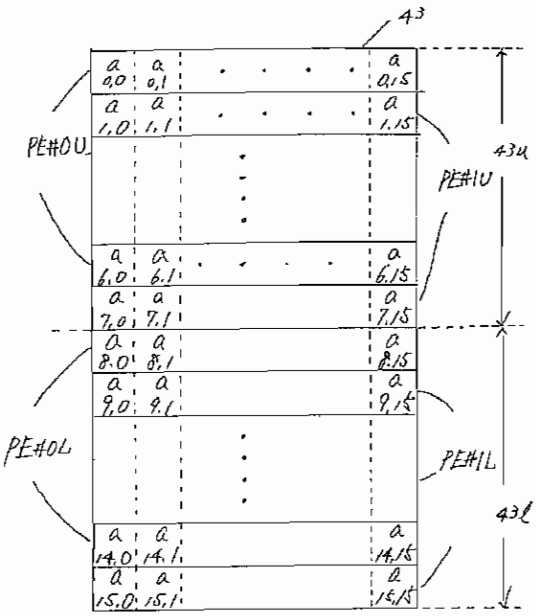
【図２９】



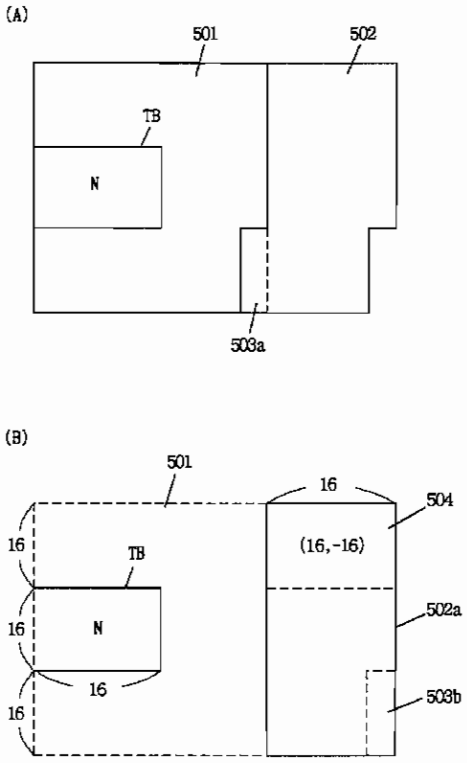
【図３０】



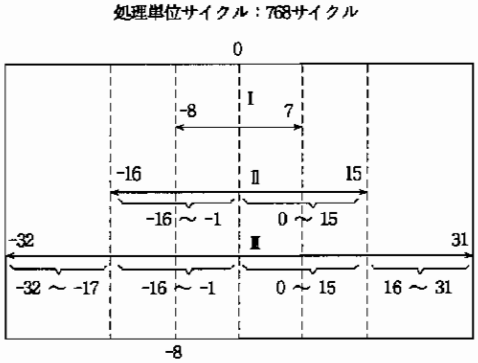
【図３１】



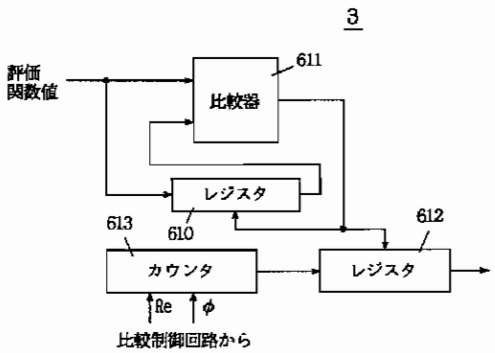
【図３４】



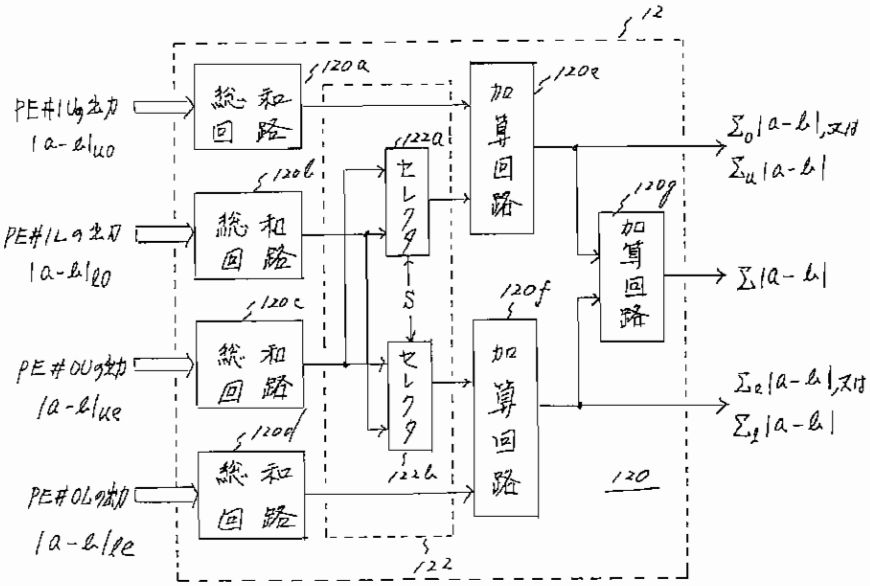
【図３６】



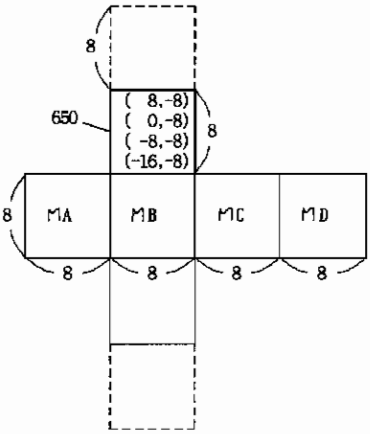
【図３９】



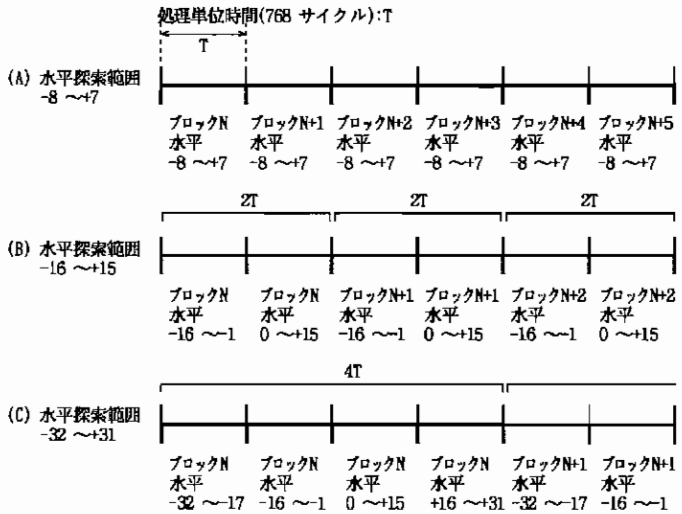
【図33】



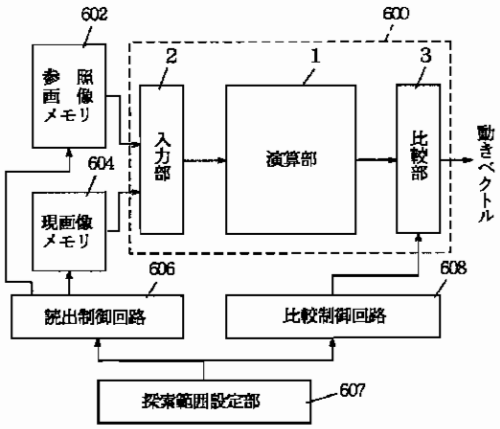
【図51】



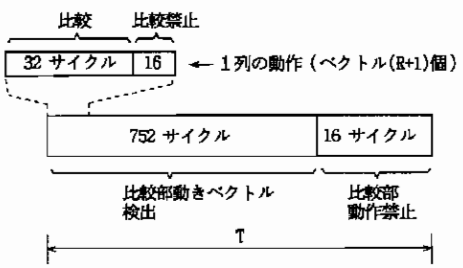
【図37】



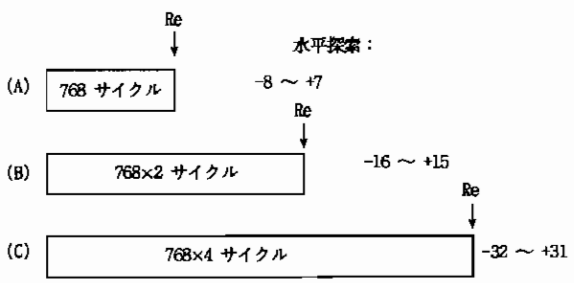
【図38】



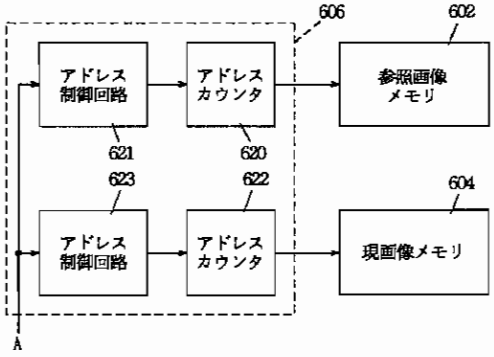
【図40】



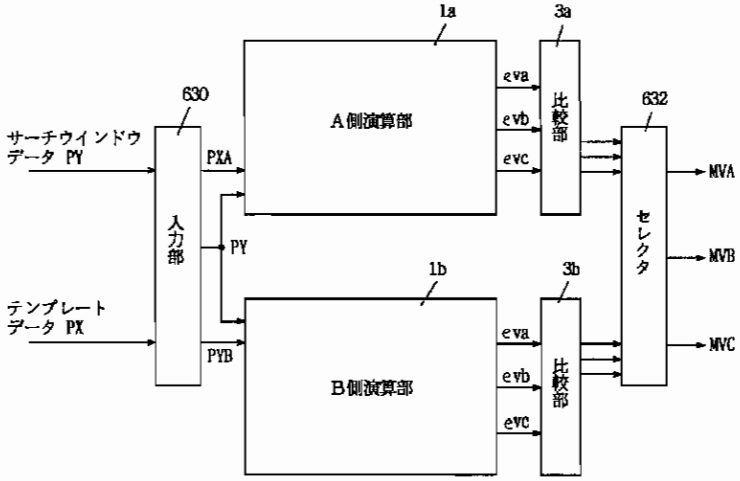
【図41】



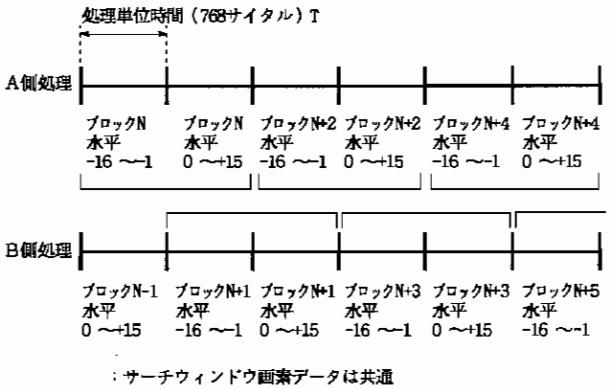
【図４３】



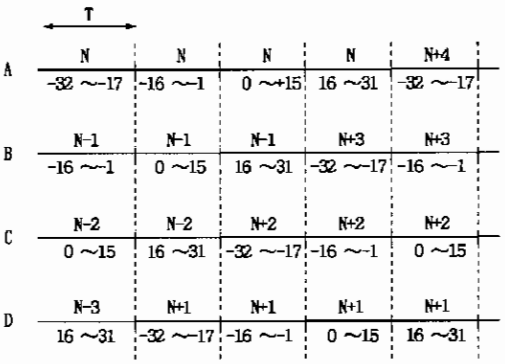
【図４４】



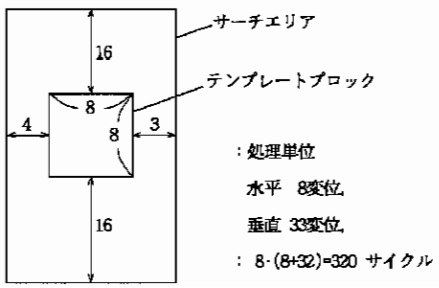
【図４６】



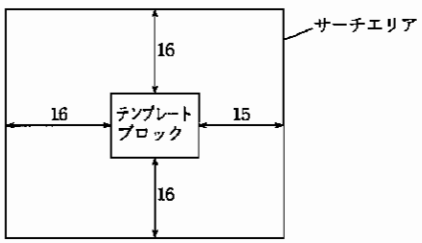
【図４７】



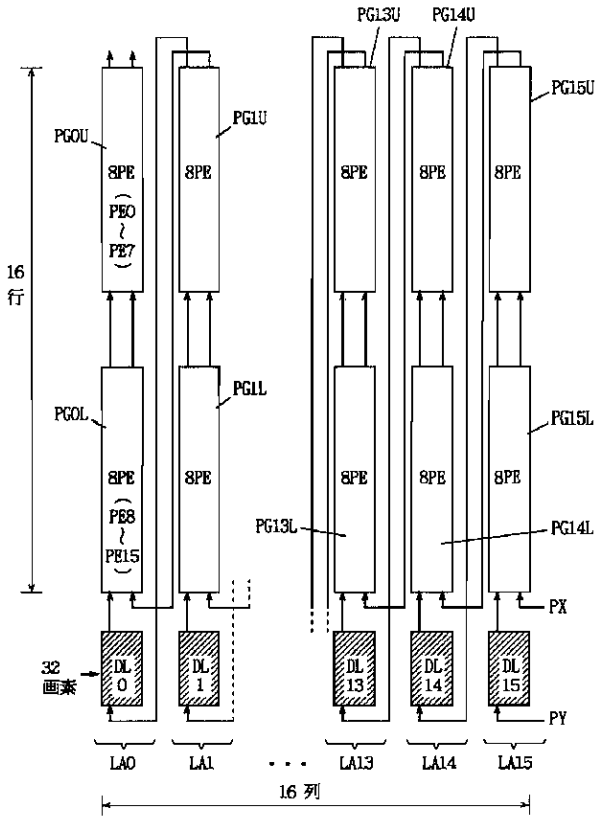
【図５０】



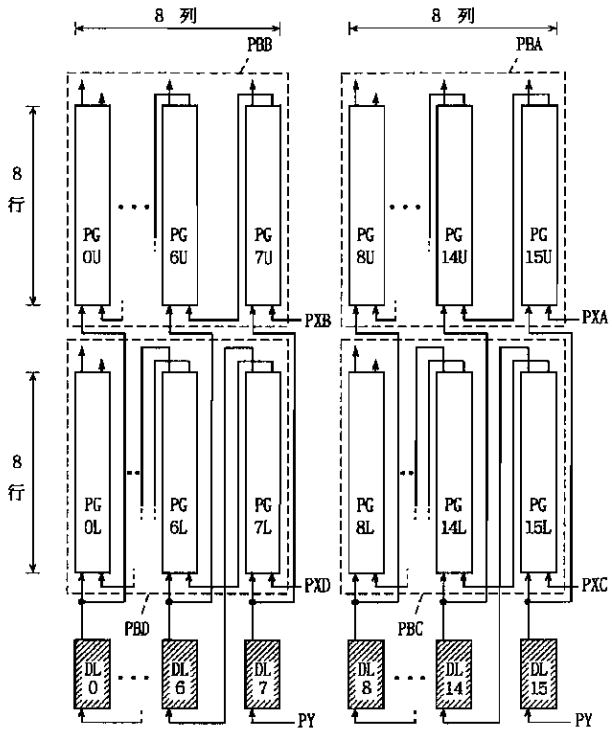
【図５２】



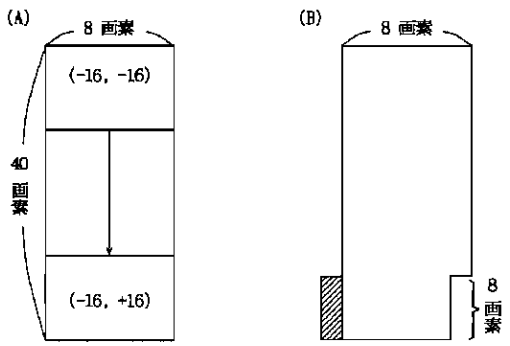
【図４８】



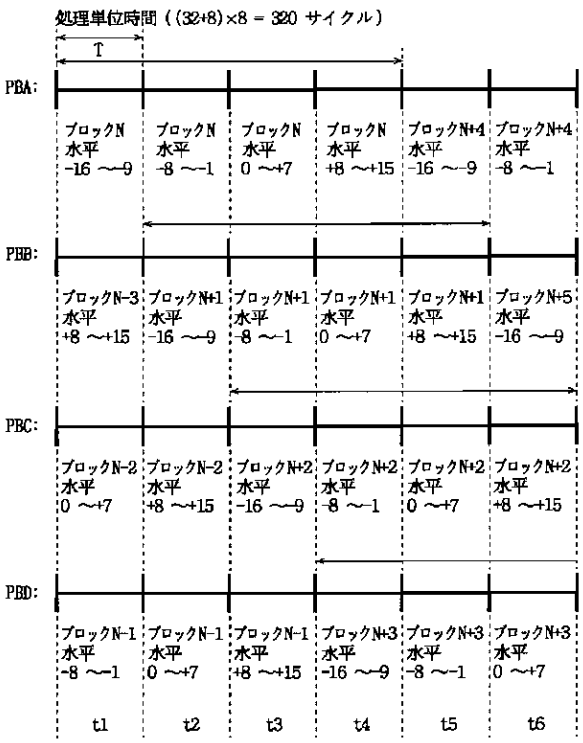
【図４９】



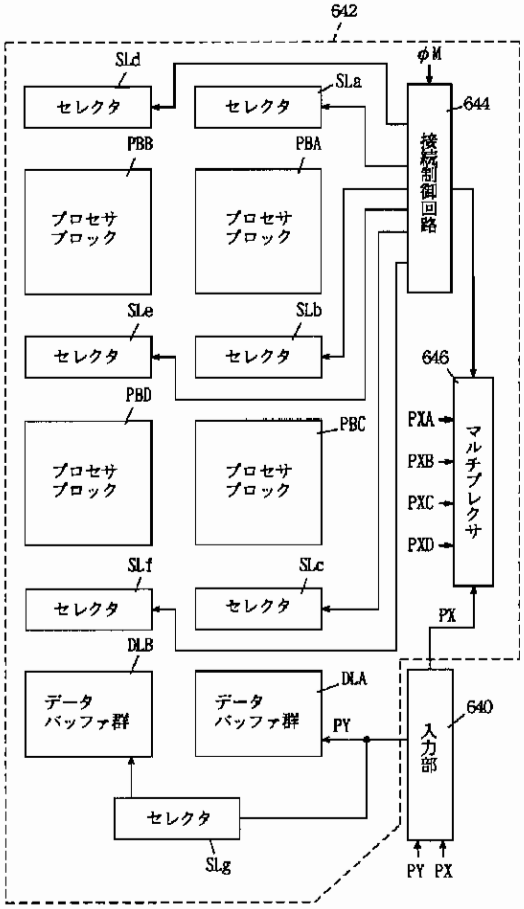
【図５３】



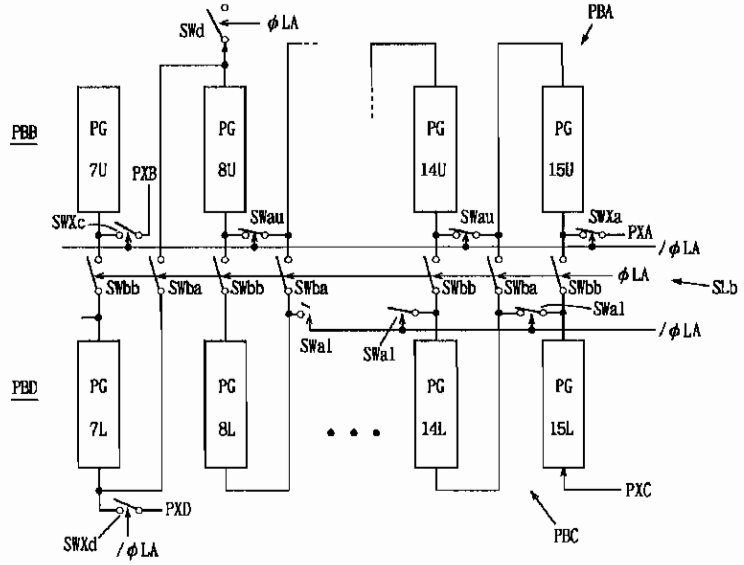
【図５４】



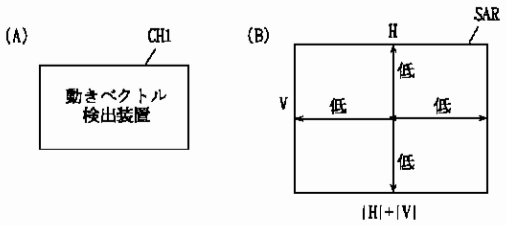
【図55】



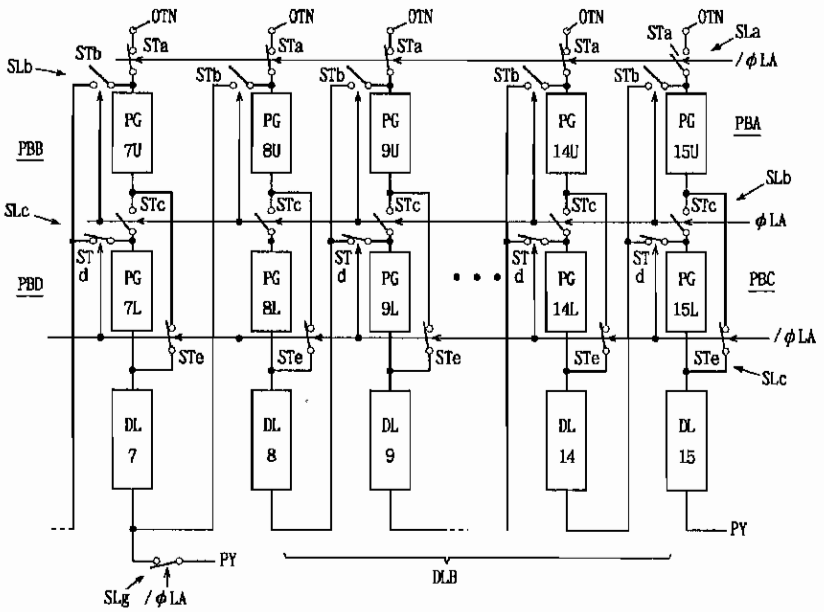
【図56】



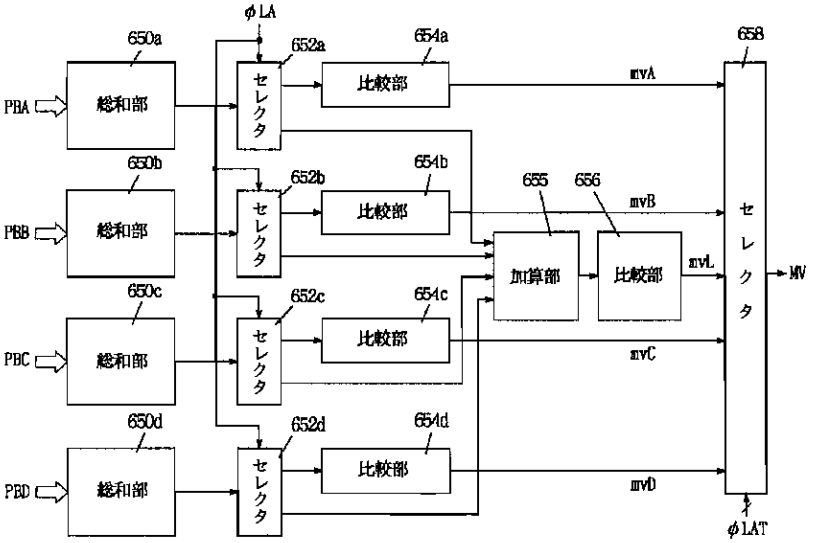
【図72】



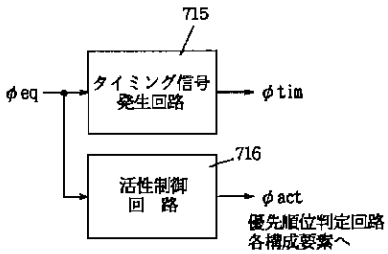
【図57】



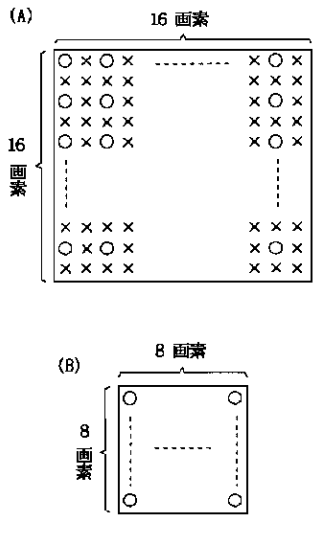
【図５８】



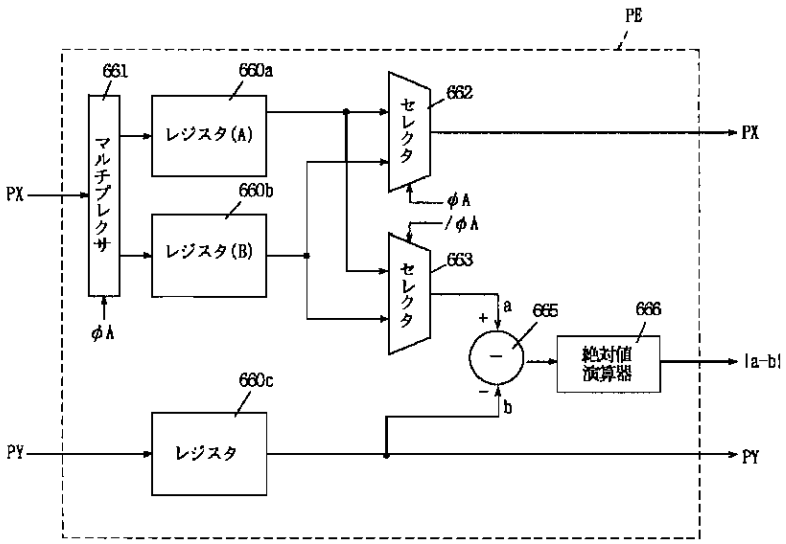
【図７６】



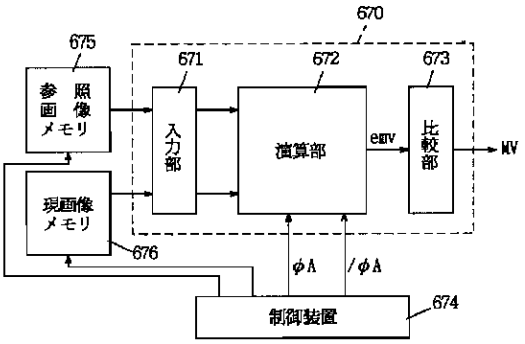
【図５９】



【図６０】



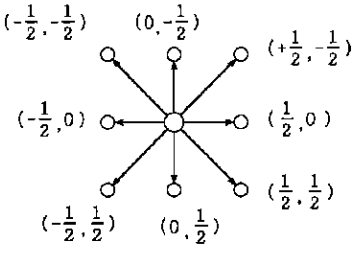
【図６２】



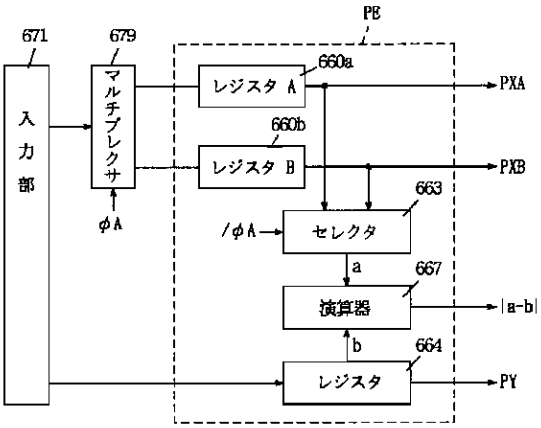
【図６１】

	T1	T2	T3	T4	T5	T6
レジスタ(A)	テンプレートブロックNの探索	テンプレートブロックN+2のロード	テンプレートブロックN+2の探索	テンプレートブロックN+4のロード	テンプレートブロックN+4の探索	テンプレートブロックN+6のロード
レジスタ(B)	テンプレートブロックN+1のロード	テンプレートブロックN+1の探索	テンプレートブロックN+3のロード	テンプレートブロックN+3の探索	テンプレートブロックN+5のロード	テンプレートブロックN+5の探索

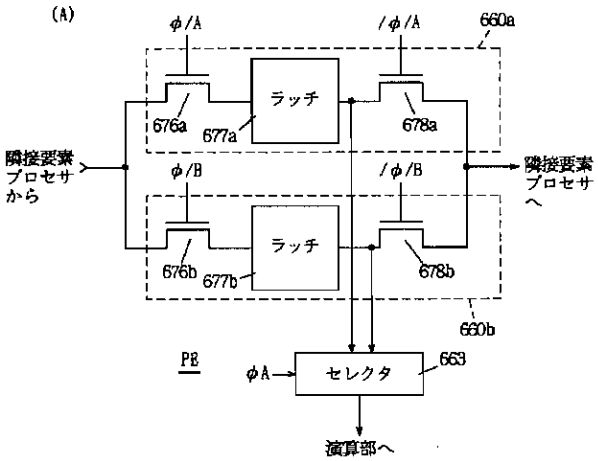
【図７８】



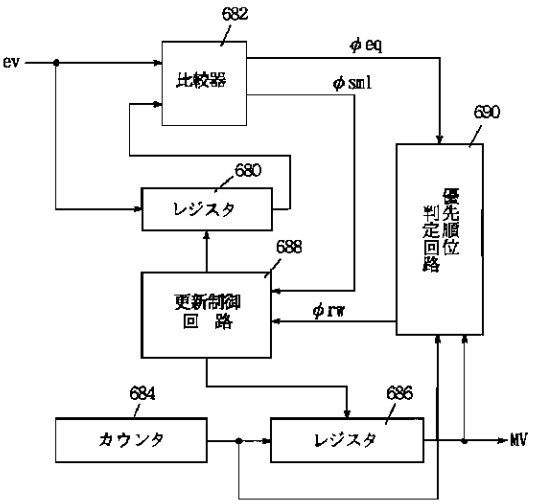
【図６３】

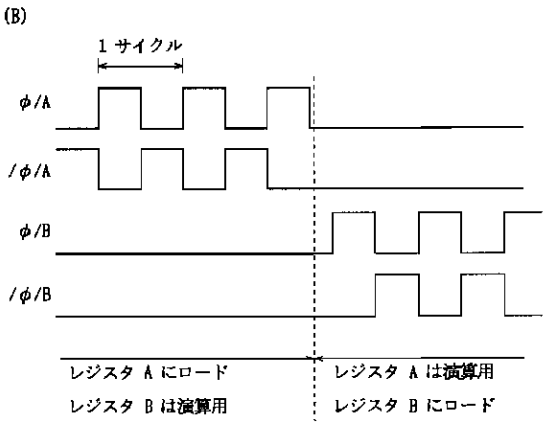


【図６４】

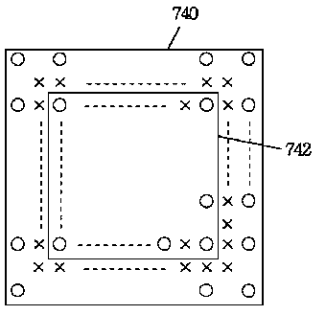


【図６５】





【図７９】



【図６６】

P(H,V) = |H|+|V|

Vベクトル ＼ Hベクトル	-16	-15	-14	-13	-12	-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1	0
-16	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
-15	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15
-14	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14
-13	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13
-12	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12
-11	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11
-10	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10
-9	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9
-8	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8
-7	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7
-6	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6
-5	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5
-4	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4
-3	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3
-2	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2
-1	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
0	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
2	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2
3	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3
4	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4
5	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5
6	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6
7	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7
8	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8
9	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9
10	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10
11	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11
12	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12
13	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13
14	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14
15	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15
16	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16

【図６７】

P(H,V) = |H|+|V|

V ＼	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
-16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	32
-15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	31
-14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	30
-13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	29
-12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	28
-11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	27
-10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	26
-9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	25
-8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	24
-7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	23
-6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	22
-5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	21
-4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	20
-3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	19
-2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	18
-1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	17
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	16
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	17
2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	18
3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	19
4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	20
5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	21
6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	22
7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	23
8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	24
9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	25
10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	26
11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	27
12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	28
13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	29
14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	30
15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	31
16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	32

【図６８】

P(H,V) = |H|+(-V+16)

Vベクトル ＼ Hベクトル	-16	-15	-14	-13	-12	-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1	0
-16	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
-15	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32	31
-14	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32	31	30
-13	45	44	43	42	41	40	39	38	37	36	35	34	33	32	31	30	29
-12	44	43	42	41	40	39	38	37	36	35	34	33	32	31	30	29	28
-11	43	42	41	40	39	38	37	36	35	34	33	32	31	30	29	28	27
-10	42	41	40	39	38	37	36	35	34	33	32	31	30	29	28	27	26
-9	41	40	39	38	37	36	35	34	33	32	31	30	29	28	27	26	25
-8	40	39	38	37	36	35	34	33	32	31	30	29	28	27	26	25	24
-7	39	38	37	36	35	34	33	32	31	30	29	28	27	26	25	24	26
-6	38	37	36	35	34	33	32	31	30	29	28	27	26	25	24	26	22
-5	37	36	35	34	33	32	31	30	29	28	27	26	25	24	26	22	21
-4	36	35	34	33	32	31	30	29	28	27	26	25	24	26	22	21	20
-3	35	34	33	32	31	30	29	28	27	26	25	24	26	22	21	20	19
-2	34	33	32	31	30	29	28	27	26	25	24	26	22	21	20	19	18
-1	33	32	31	30	29	28	27	26	25	24	26	22	21	20	19	18	17
0	32	31	30	29	28	27	26	25	24	26	22	21	20	19	18	17	16
1	31	30	29	28	27	26	25	24	26	22	21	20	19	18	17	16	15
2	30	29	28	27	26	25	24	26	22	21	20	19	18	17	16	15	14
3	29	28	27	26	25	24	26	22	21	20	19	18	17	16	15	14	13
4	28	27	26	25	24	26	22	21	20	19	18	17	16	15	14	13	12
5	27	26	25	24	26	22	21	20	19	18	17	16	15	14	13	12	11
6	26	25	24	26	22	21	20	19	18	17	16	15	14	13	12	11	10
7	25	24	26	22	21	20	19	18	17	16	15	14	13	12	11	10	9
8	24	26	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8
9	26	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7
10	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6
11	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5
12	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4
13	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3
14	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2
15	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
16	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

【図６９】

P(H,V) = |H|+(-V+16)

V ＼	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
-16	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47
-15	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46
-14	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45
-13	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44
-12	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43
-11	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42
-10	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41
-9	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40
-8	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39
-7	26	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38
-6	22	26	24	25	26	27	28	29	30	31	32	33	34	35	36	37
-5	21	22	26	24	25	26	27	28	29	30	31	32	33	34	35	36
-4	20	21	22	26	24	25	26	27	28	29	30	31	32	33	34	35
-3	19	20	21	22	26	24	25	26	27	28	29	30	31	32	33	34
-2	18	19	20	21	22	26	24	25	26	27	28	29	30	31	32	33
-1	17	18	19	20	21	22	26	24	25	26	27	28	29	30	31	32
0	16	17	18	19	20	21	22	26	24	25	26	27	28	29	30	31
1	15	16	17	18	19	20	21	22	26	24	25	26	27	28	29	30
2	14	15	16	17	18	19	20	21	22	26	24	25	26	27	28	29
3	13	14	15	16	17	18	19	20	21	22	26	24	25	26	27	28
4	12	13	14	15	16	17	18	19	20	21	22	26	24	25	26	27
5	11	12	13	14	15	16	17	18	19	20	21	22	26	24	25	26
6	10	11	12	13	14	15	16	17	18	19	20	21	22	26	24	25
7	9	10	11	12	13	14	15	16	17	18	19	20	21	22	26	24
8	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	26
9	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
10	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
11	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
12	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
13	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
14	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
15	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
16	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

【図７０】

H ベクトル

Vベクトル	-16	-15	-14	-13	-12	-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1	0
-16	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-15	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
-14	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2
-13	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3
-12	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4
-11	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5
-10	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6
-9	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7
-8	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8
-7	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9
-6	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10
-5	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11
-4	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12
-3	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13
-2	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14
-1	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15
0	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
1	33	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17
2	34	33	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18
3	35	34	33	32	31	30	29	28	27	26	25	24	23	22	21	20	19
4	36	35	34	33	32	31	30	29	28	27	26	25	24	23	22	21	20
5	37	36	35	34	33	32	31	30	29	28	27	26	25	24	23	22	21
6	38	37	36	35	34	33	32	31	30	29	28	27	26	25	24	23	22
7	39	38	37	36	35	34	33	32	31	30	29	28	27	26	25	24	23
8	40	39	38	37	36	35	34	33	32	31	30	29	28	27	26	25	24
9	41	40	39	38	37	36	35	34	33	32	31	30	29	28	27	26	25
10	42	41	40	39	38	37	36	35	34	33	32	31	30	29	28	27	26
11	43	42	41	40	39	38	37	36	35	34	33	32	31	30	29	28	27
12	44	43	42	41	40	39	38	37	36	35	34	33	32	31	30	29	28
13	45	44	43	42	41	40	39	38	37	36	35	34	33	32	31	30	29
14	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32	31	30
15	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32	31
16	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32

P(H,V) = |H|+(+V+16)

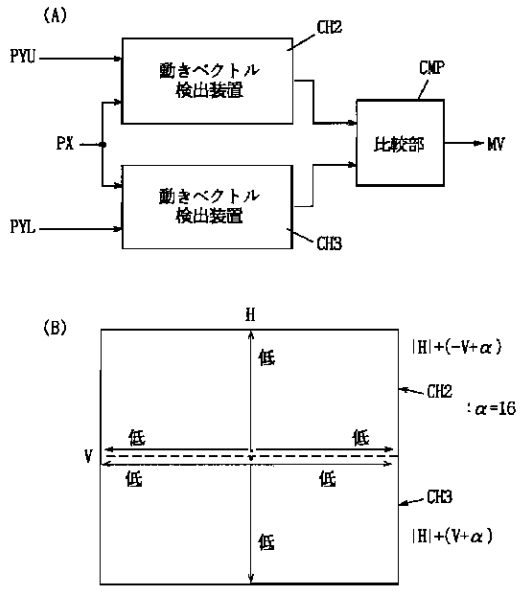
【図７１】

H ベクトル

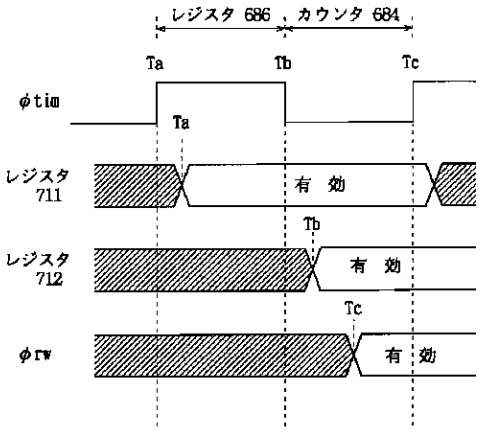
V	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
-16	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
-15	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
-14	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
-13	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
-12	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
-11	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
-10	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
-9	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
-8	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
-7	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
-6	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
-5	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26
-4	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27
-3	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28
-2	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29
-1	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
0	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
1	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
2	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33
3	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34
4	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35
5	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36
6	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37
7	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38
8	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39
9	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40
10	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41
11	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42
12	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43
13	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44
14	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45
15	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46
16	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47

P(H,V) = |H|+(+V+16)

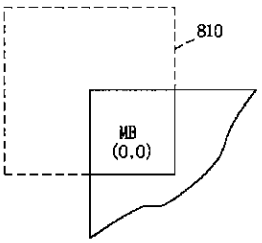
【図７３】



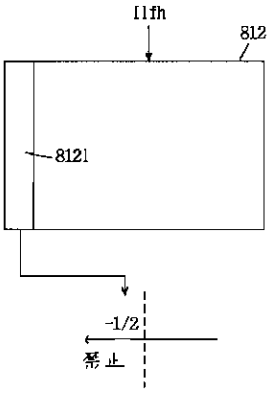
【図７５】



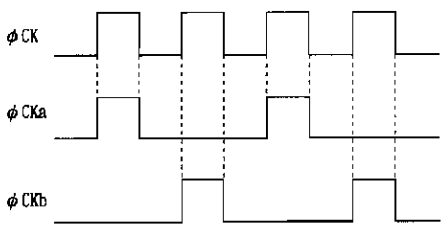
【図８９】



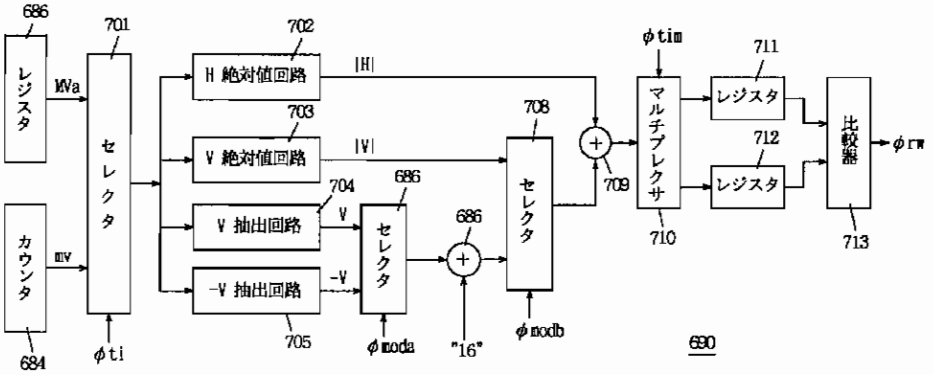
【図９２】



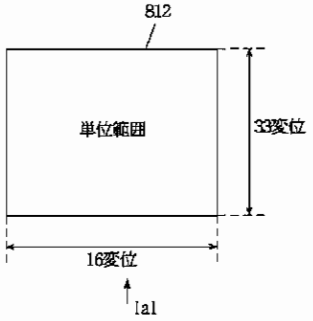
【図８３】



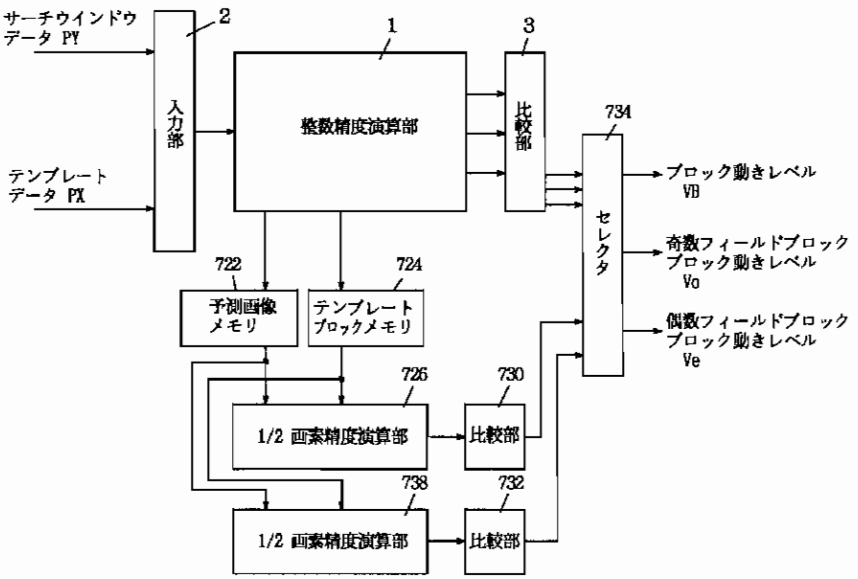
【図７４】



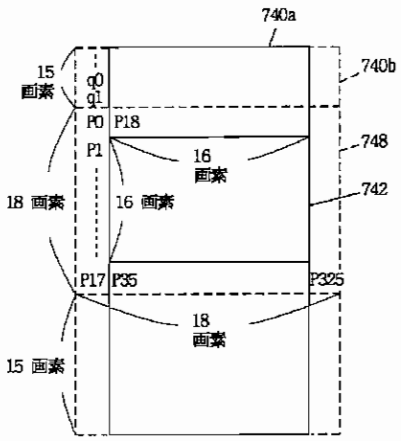
【図９０】



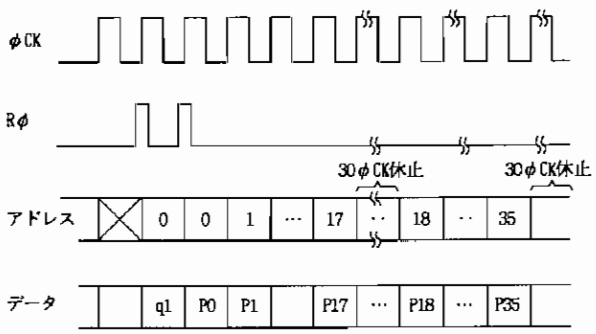
【図７７】



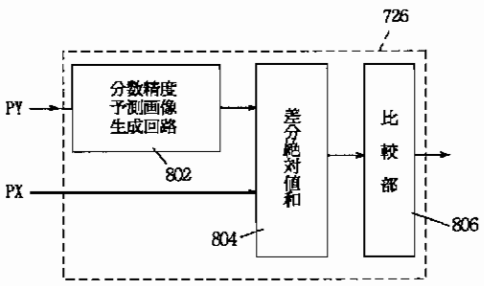
【図８１】



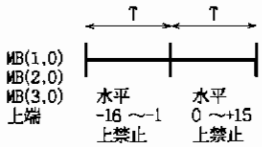
【図８２】



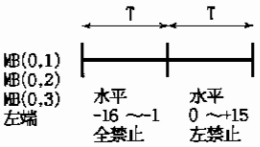
【図８４】



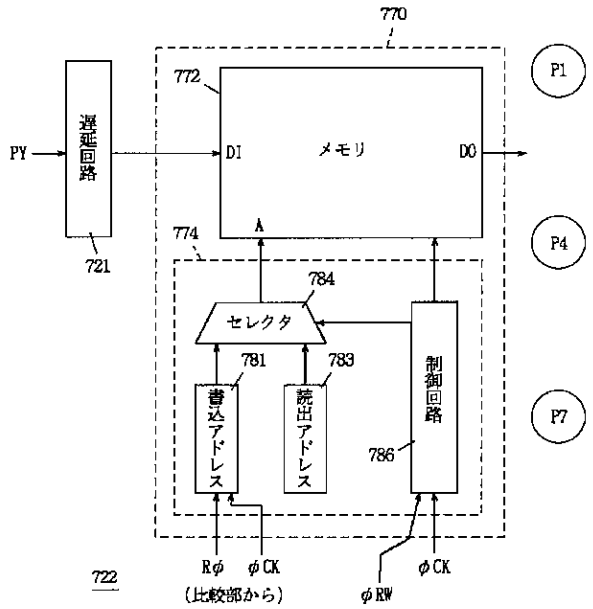
【図９７】



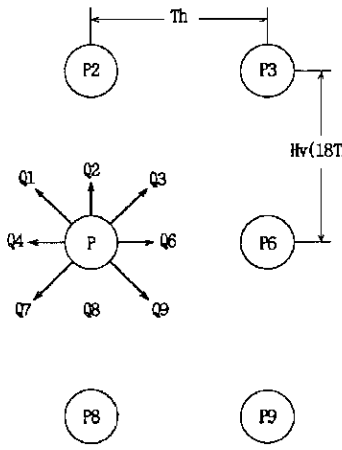
【図１００】



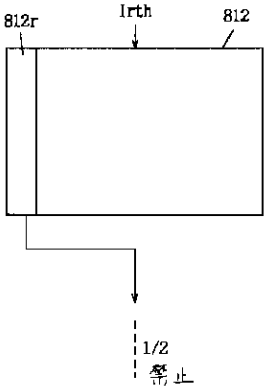
【図８０】



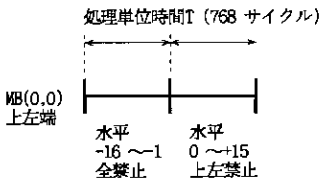
【図８５】



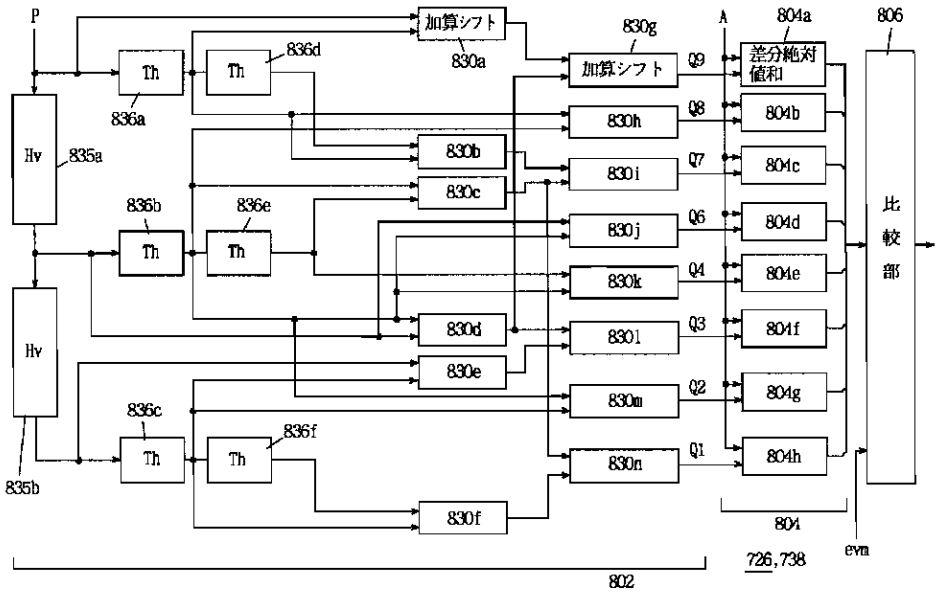
【図９３】



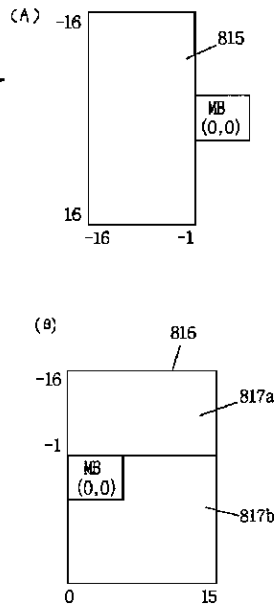
【図９４】



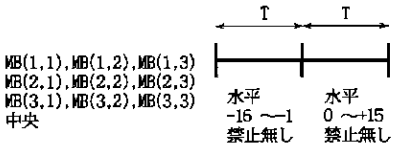
【図８６】



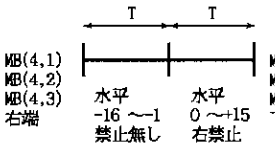
【図９５】



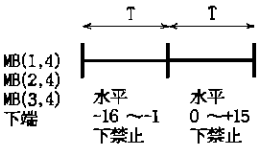
【図１０１】



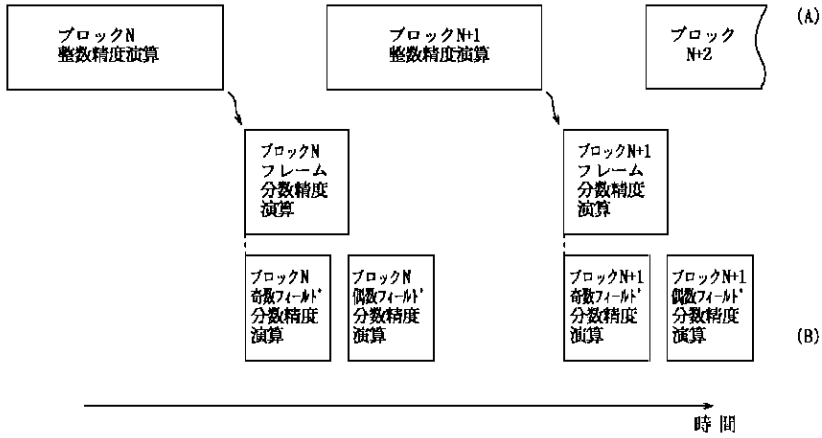
【図１０２】



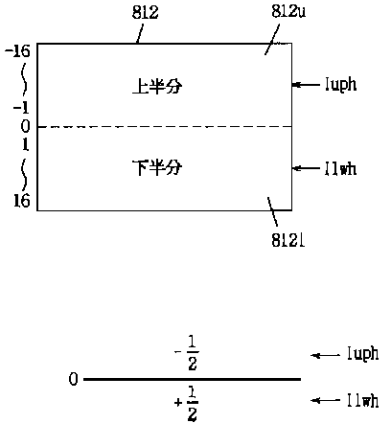
【図１０５】



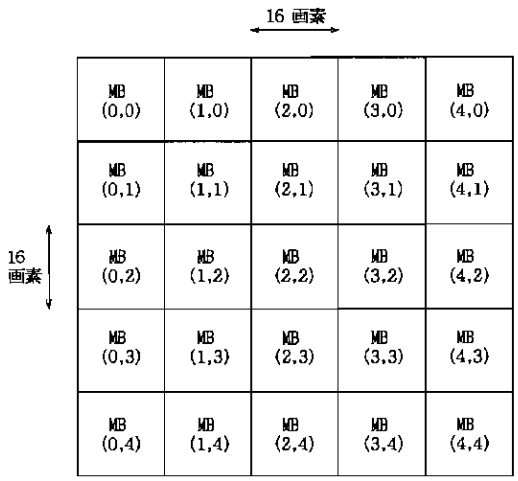
【図８７】



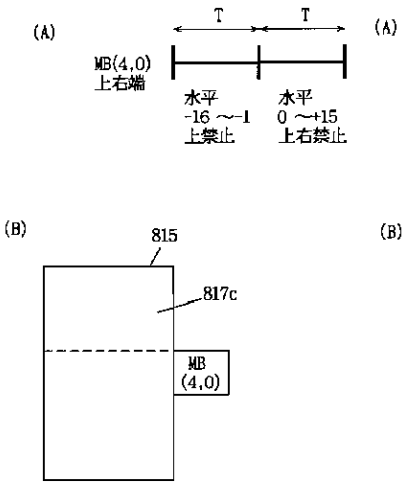
【図９１】



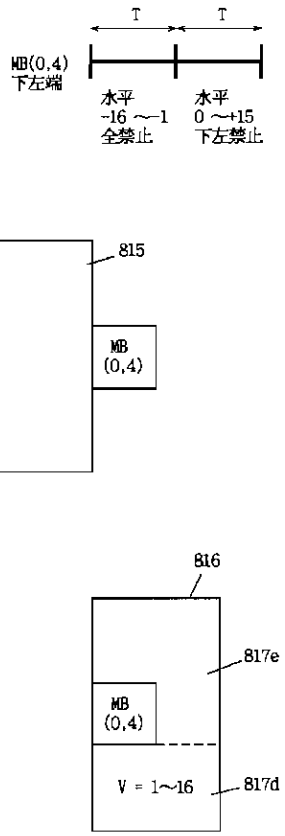
【図８８】



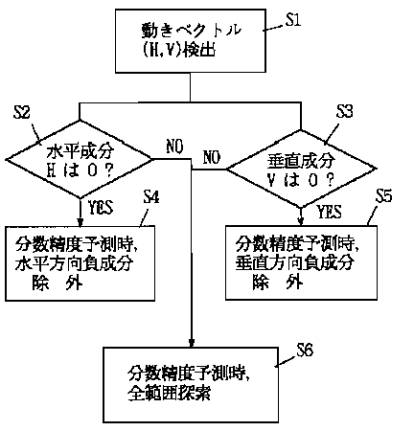
【図９８】



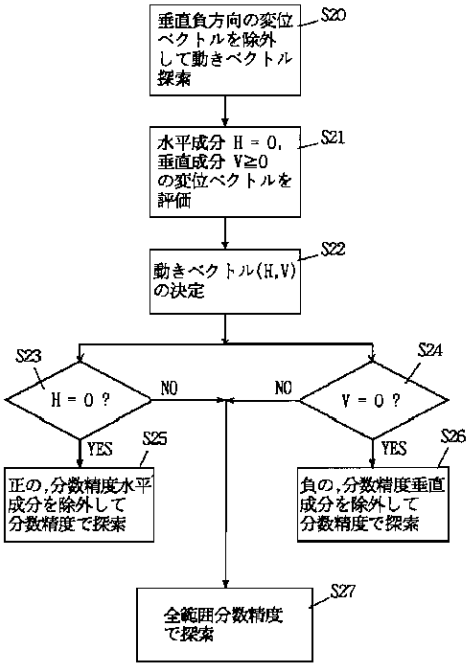
【図１０３】



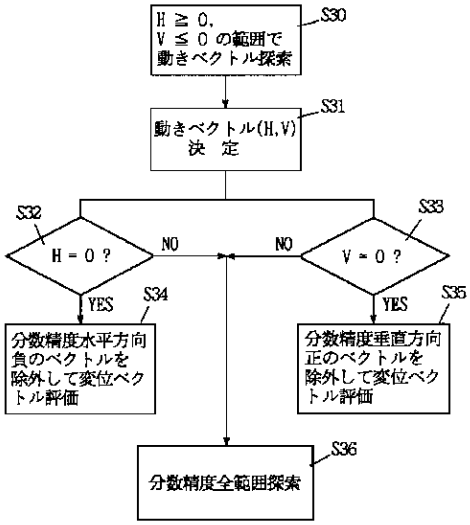
【図９６】



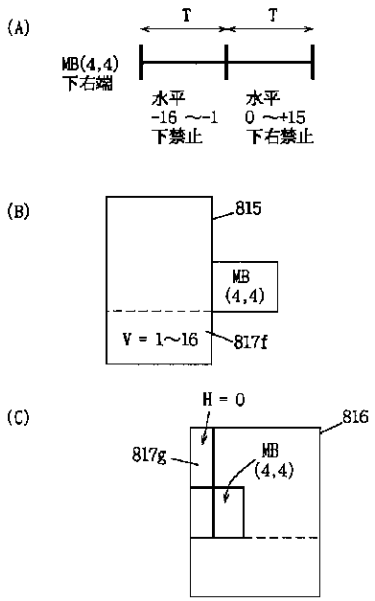
【図９９】



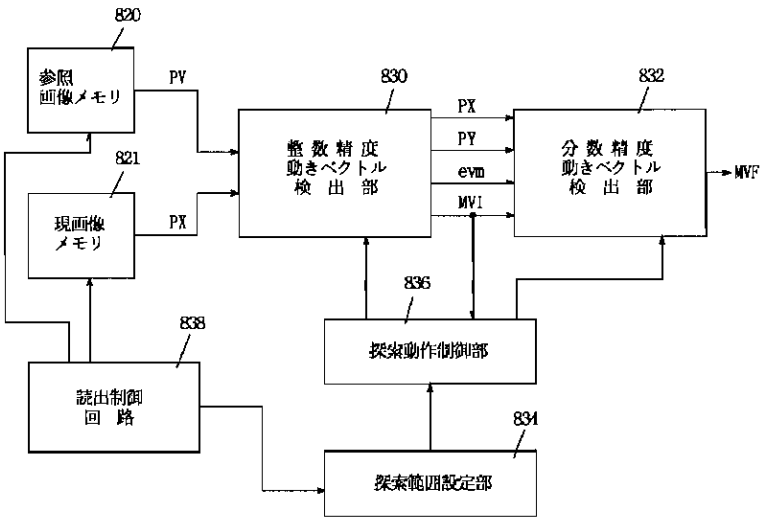
【図１０４】



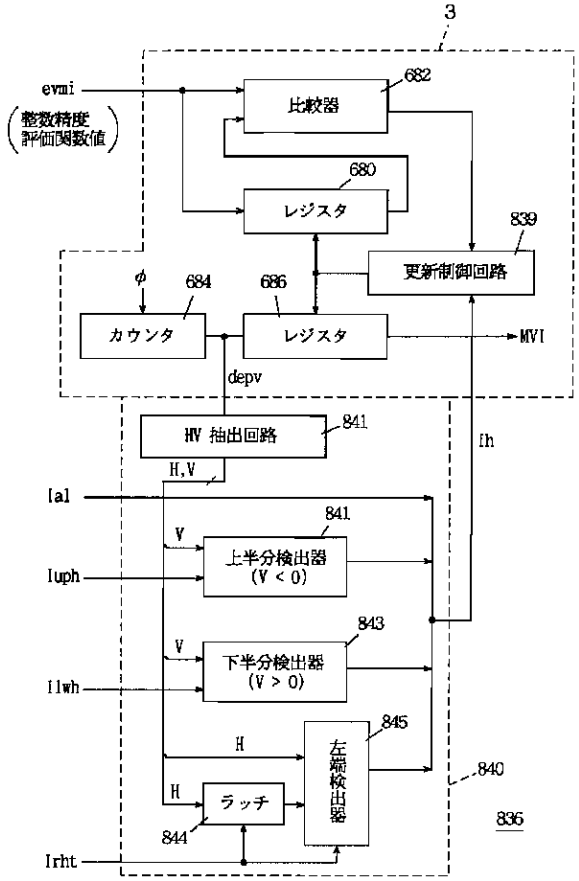
【図１０６】



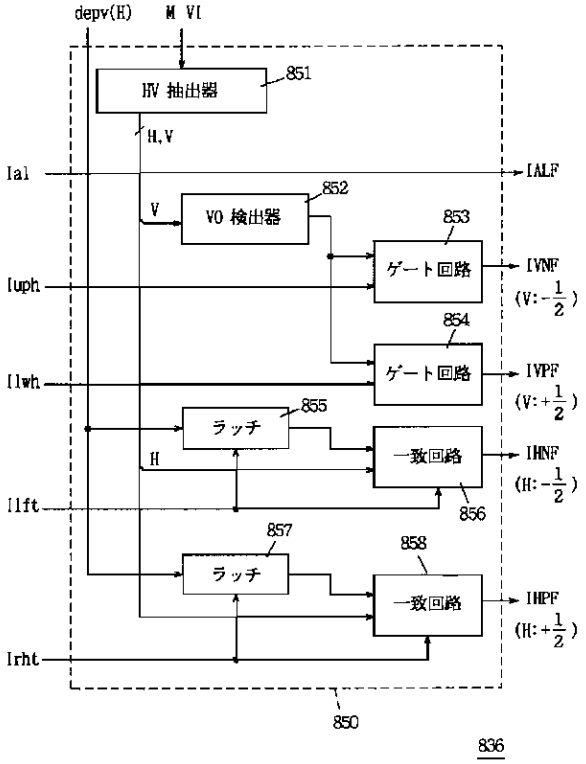
【図１０７】



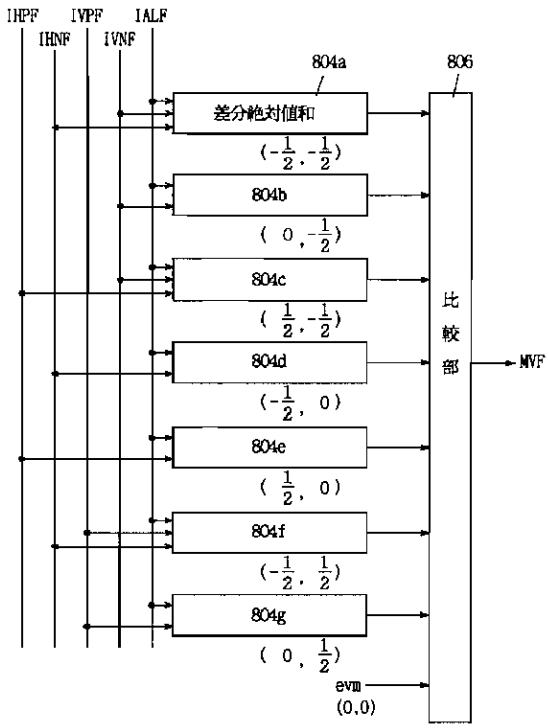
【図１０８】



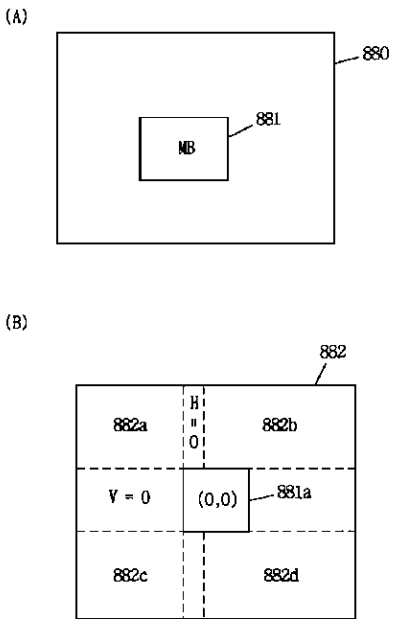
【図１０９】



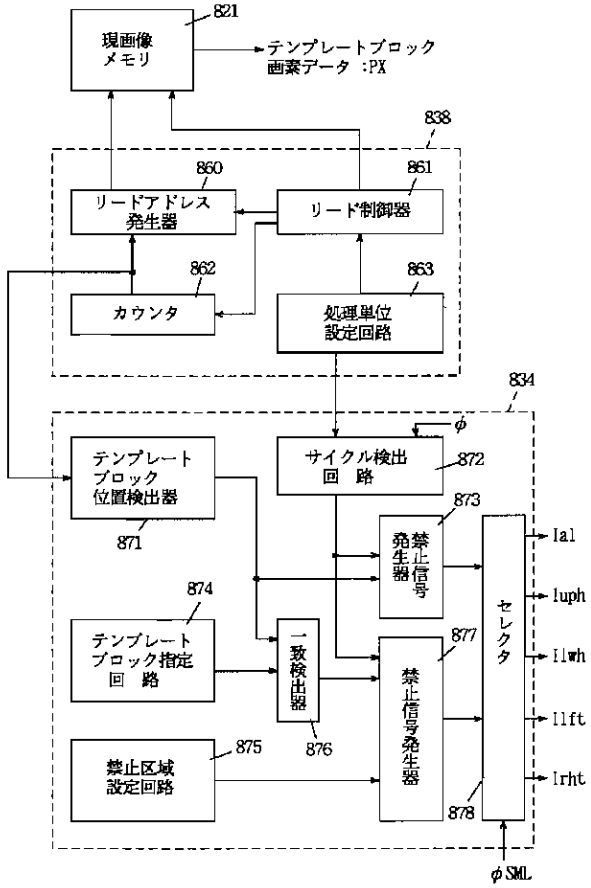
【図１１０】



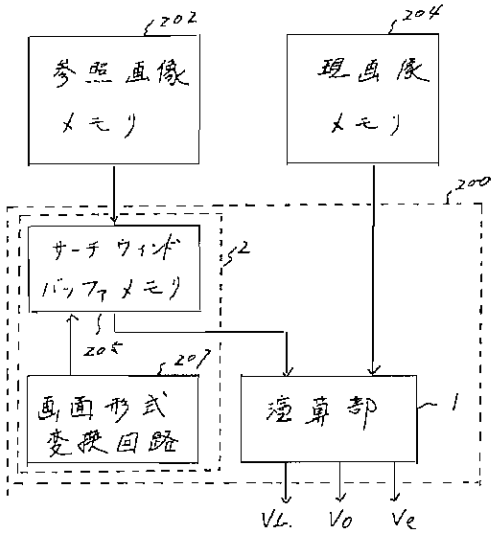
【図１１２】



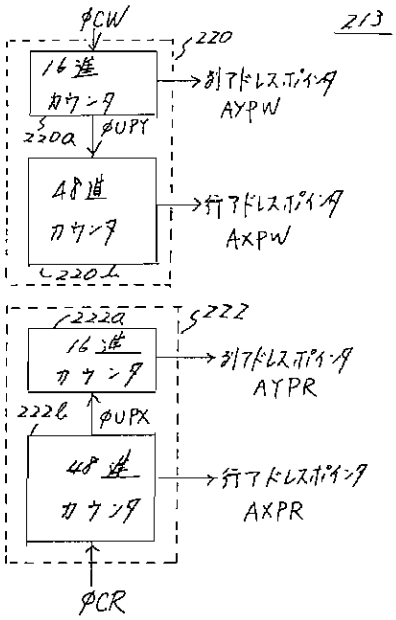
【図１１１】



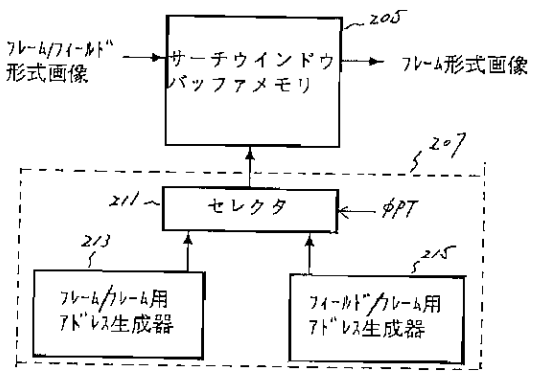
【図１１３】

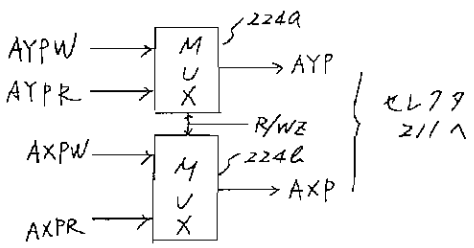


【図１１５】

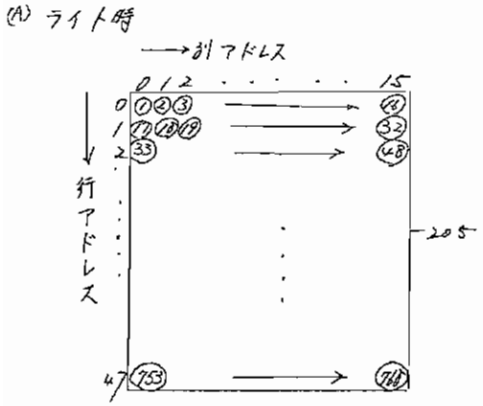


【図１１４】

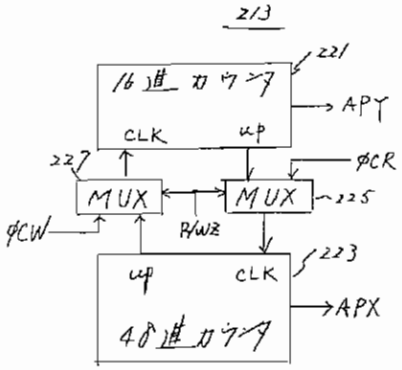




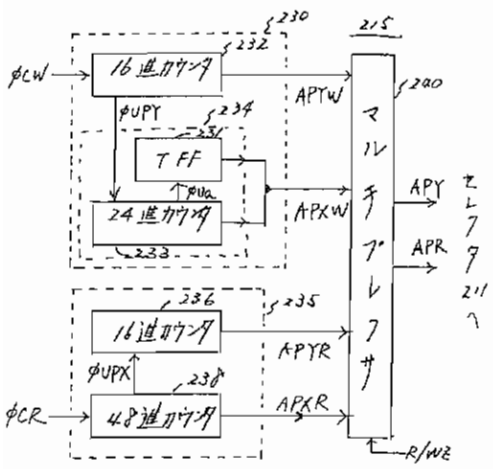
【図１１６】



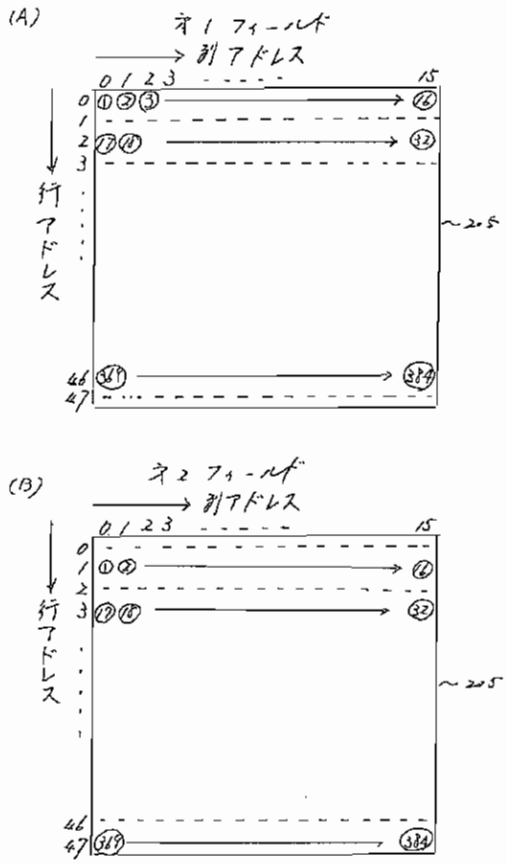
【図１１７】



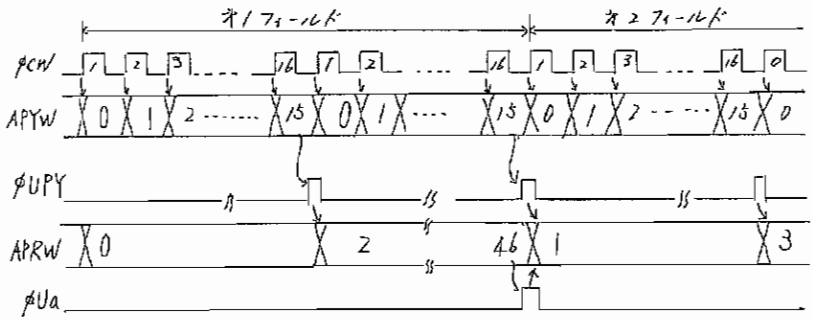
【図１１８】



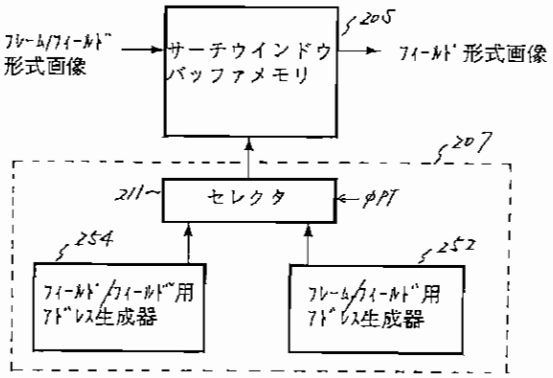
【図１２０】



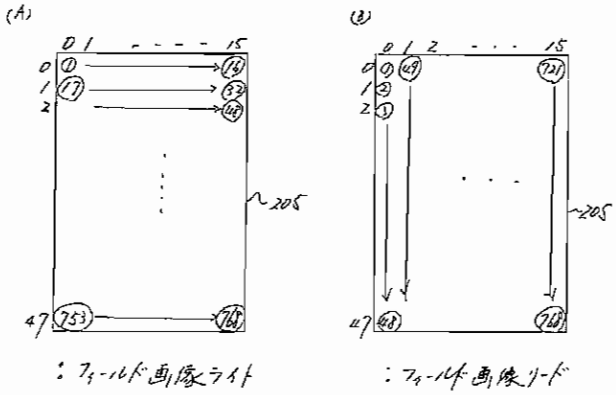
【図１１９】



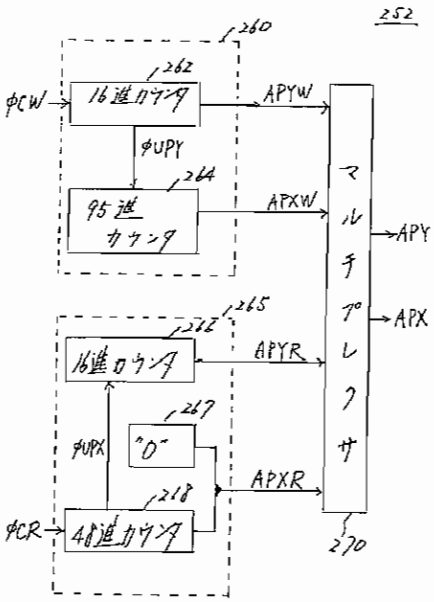
【図１２１】



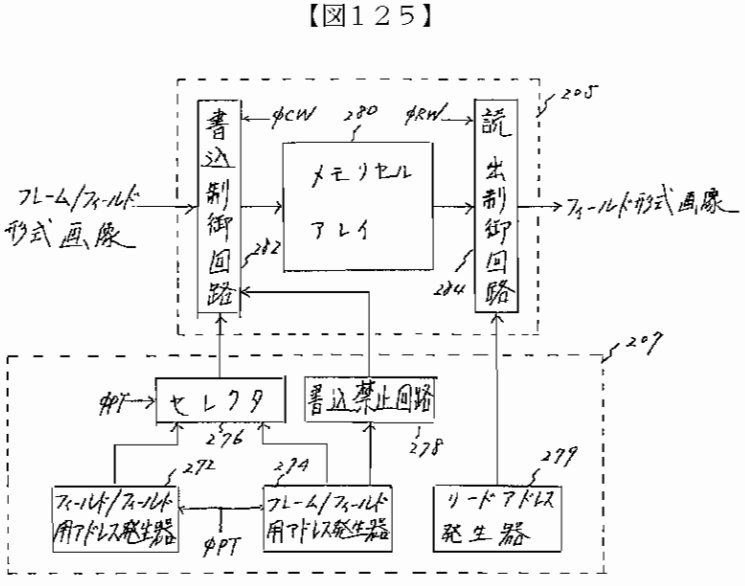
【図１２２】



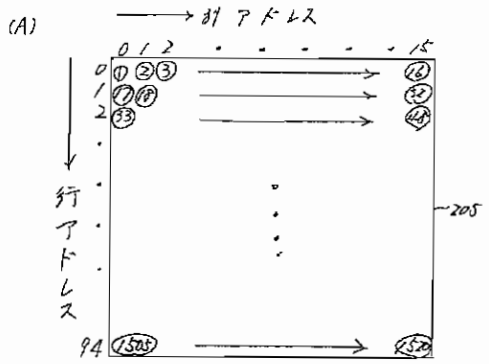
【図１２４】



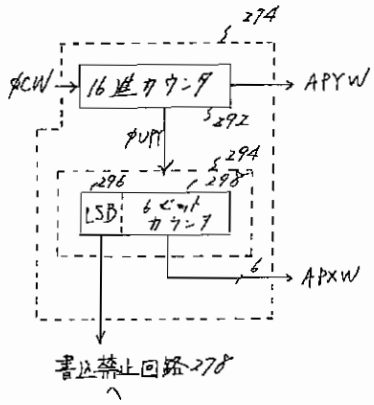
【図１２５】



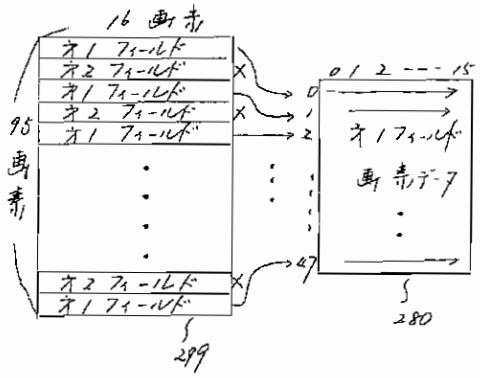
【図123】



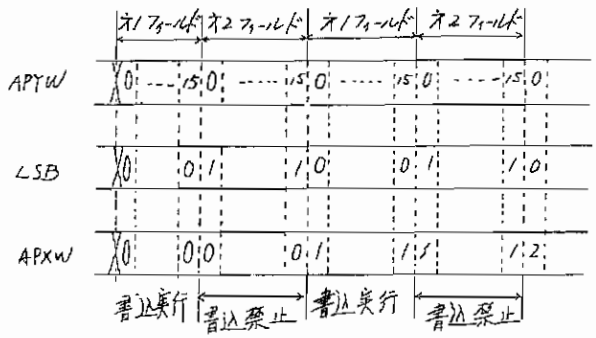
【図126】



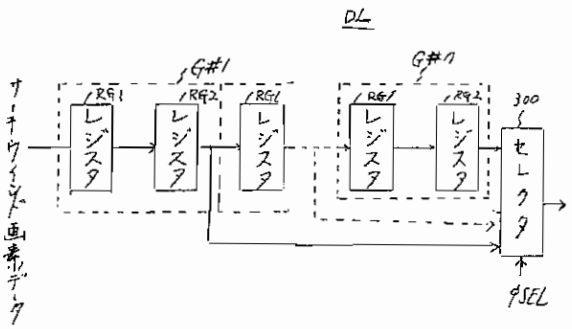
【図128】



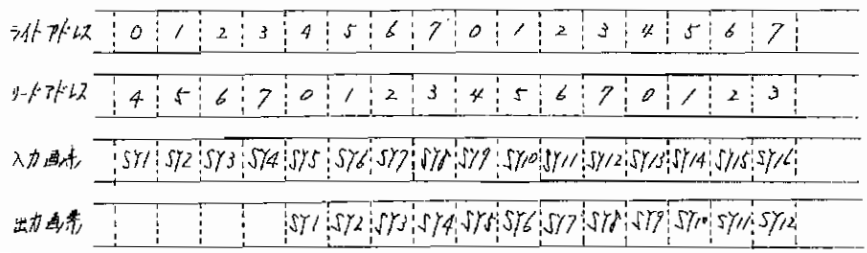
【図127】



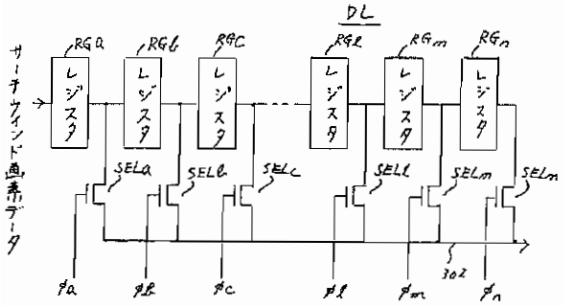
【図129】



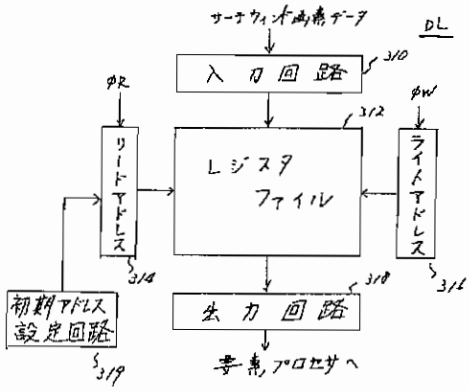
【図132】



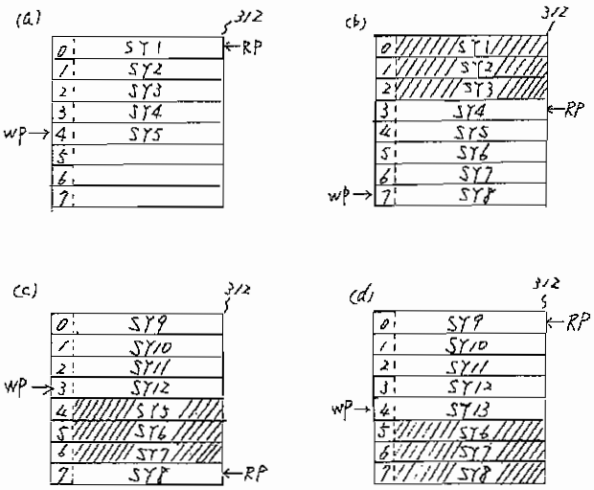
【図１３０】



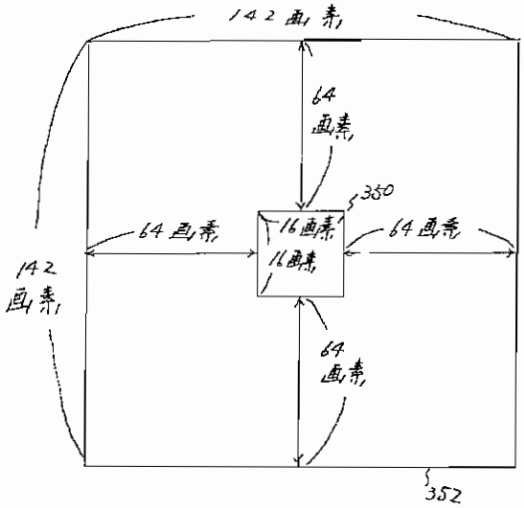
【図１３１】

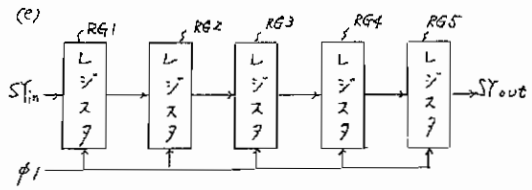


【図１３３】

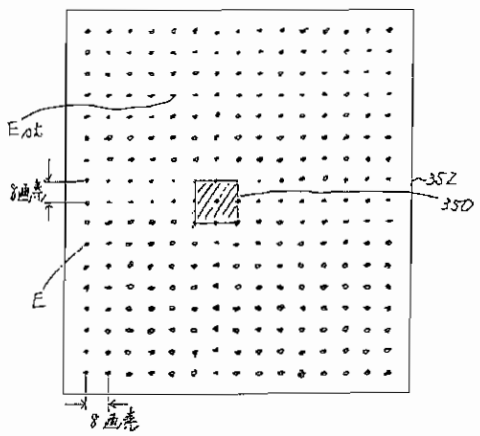


【図１３４】

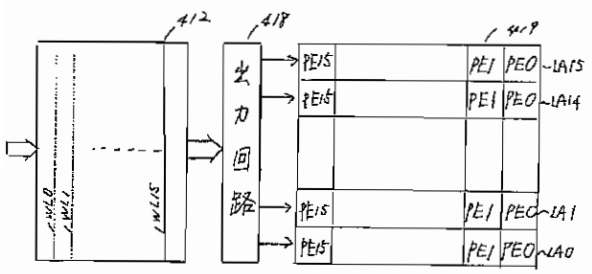




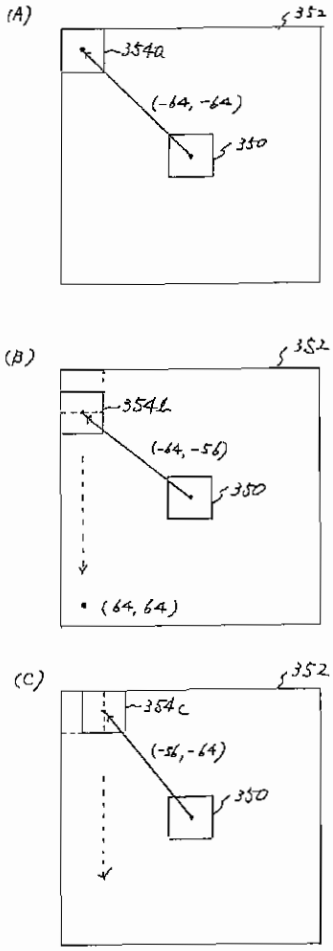
【図１３６】



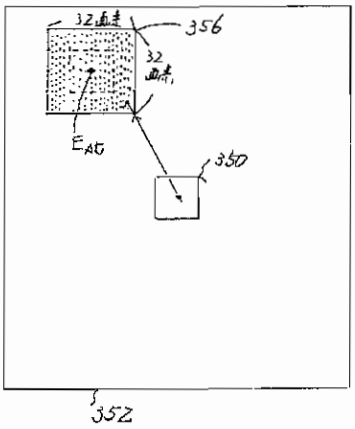
【図１４０】



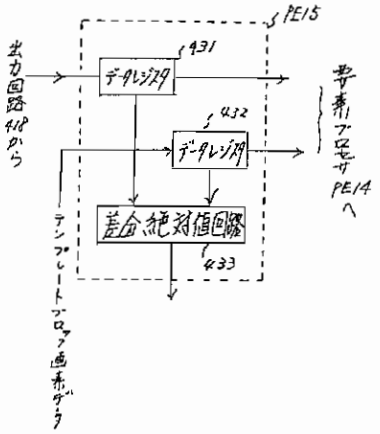
【図１３５】



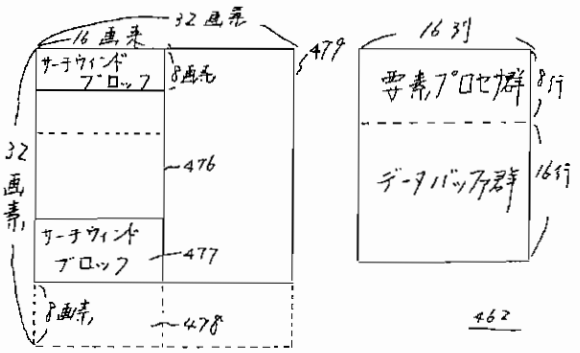
【図１３７】



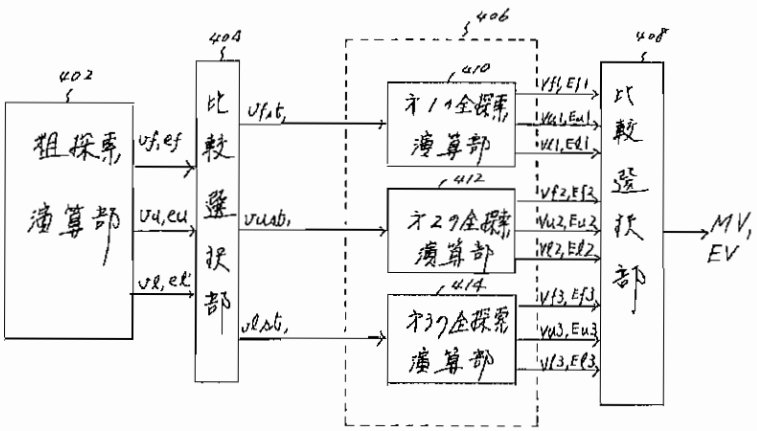
【図１４１】



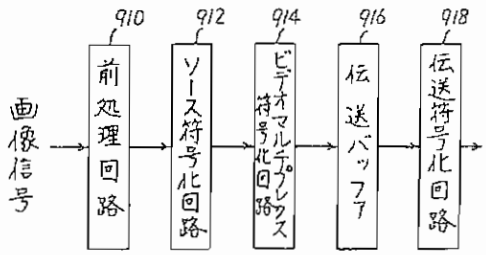
【図１４６】



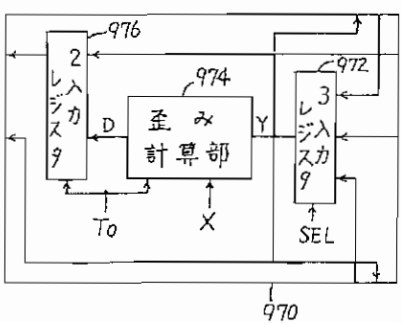
【図１３８】



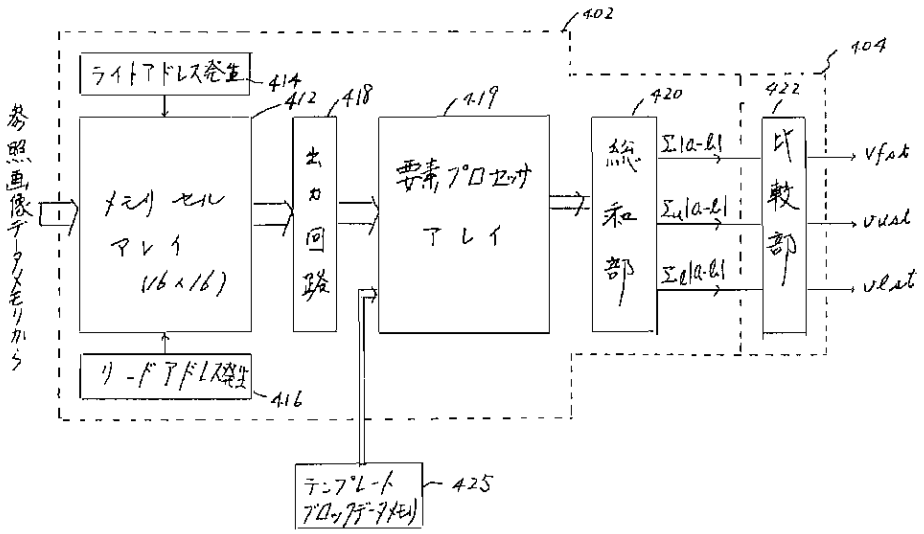
【図１５１】



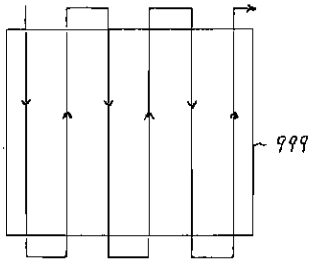
【図１５６】



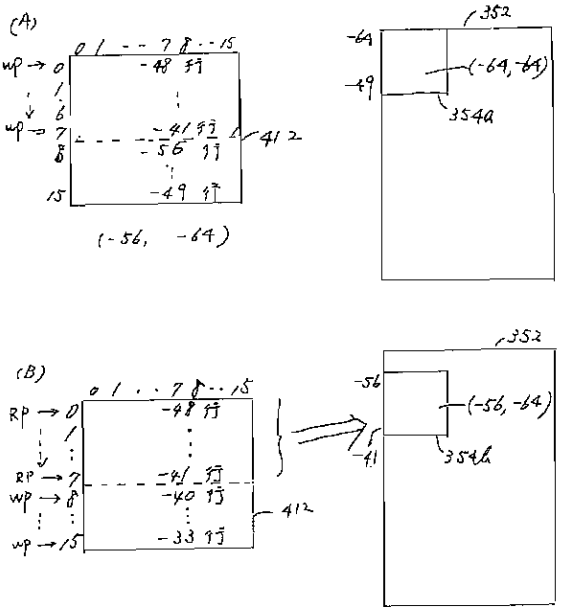
【図１３９】



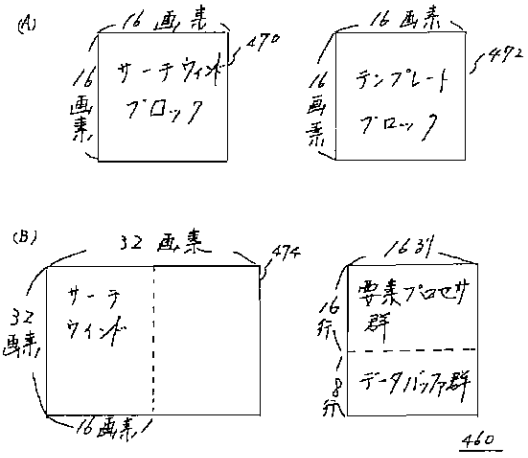
【図１５７】



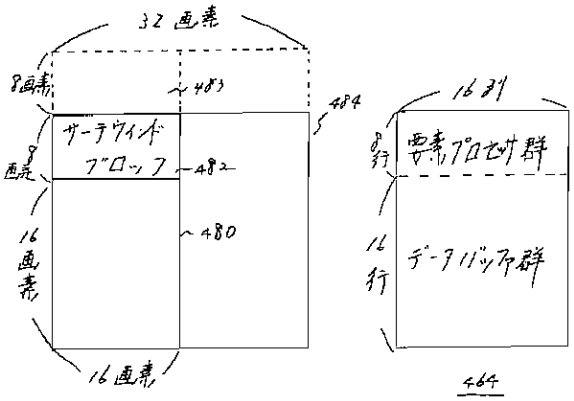
【図１４２】



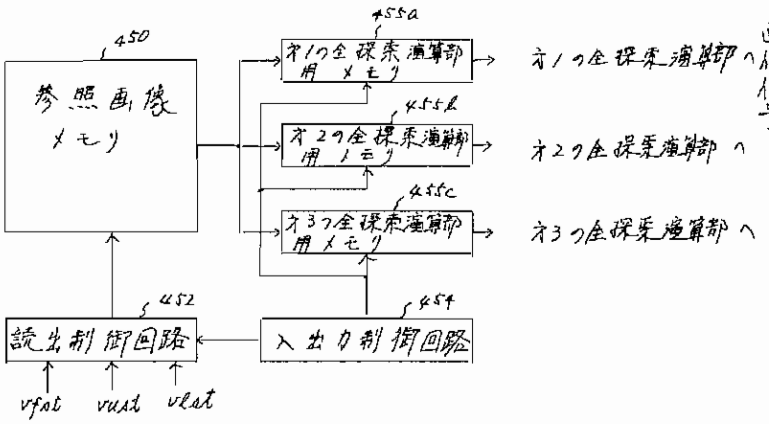
【図１４５】



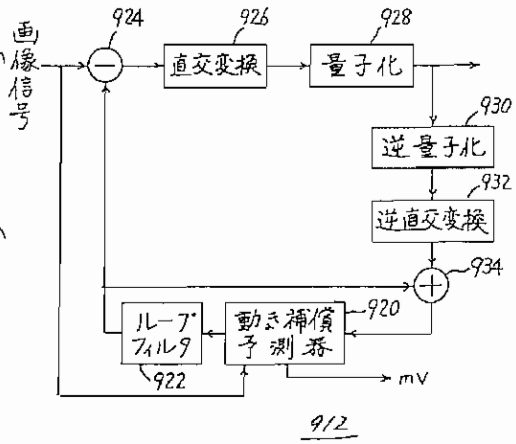
【図１４７】



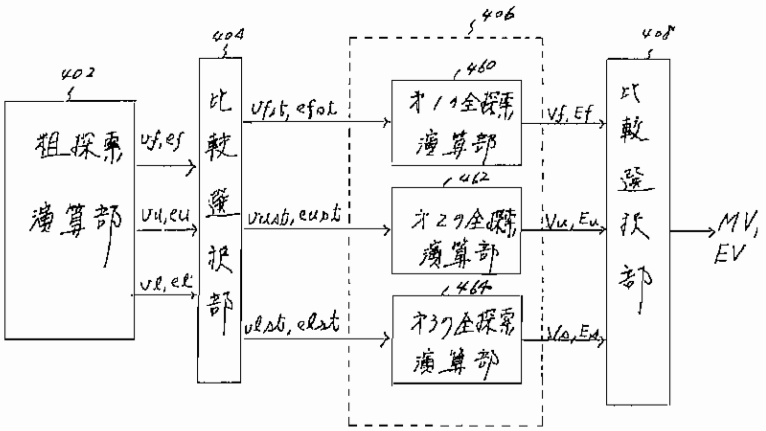
【図１４３】



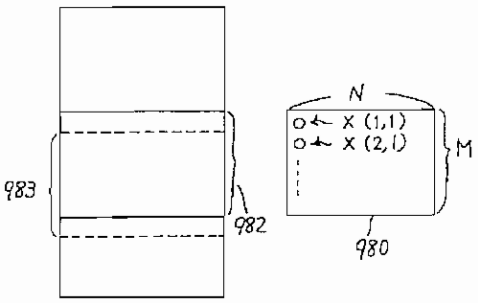
【図１５２】



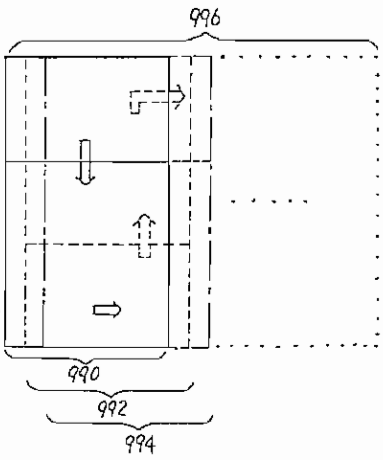
【図１４４】



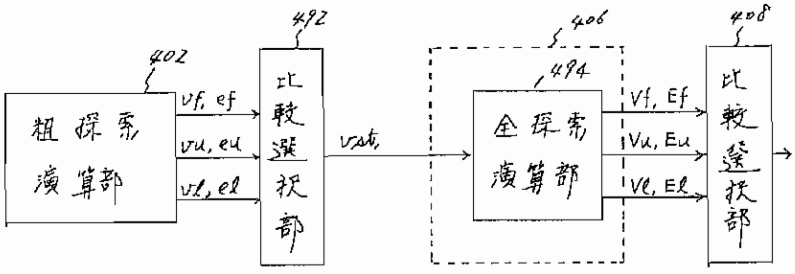
【図１５８】



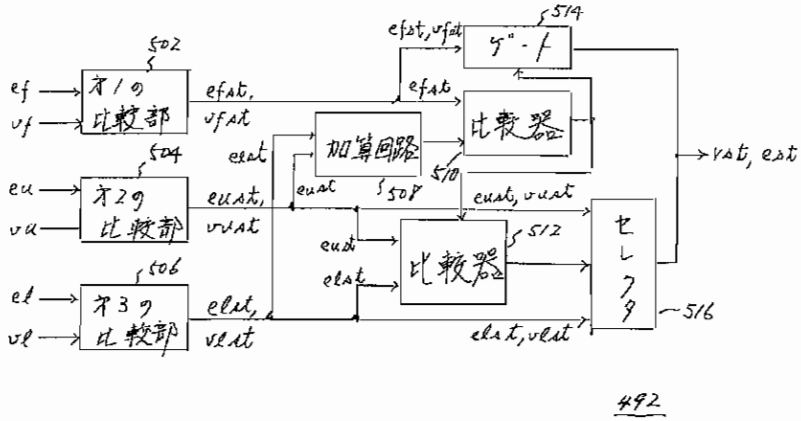
【図１５９】



【図１４８】

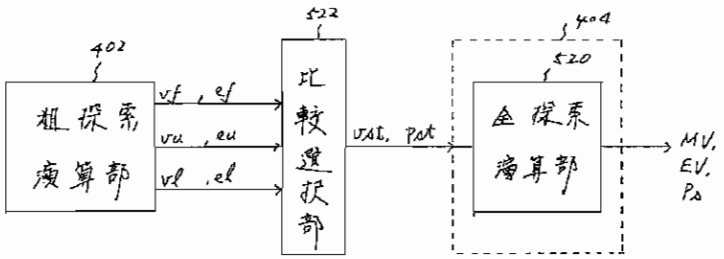


【図１４９】

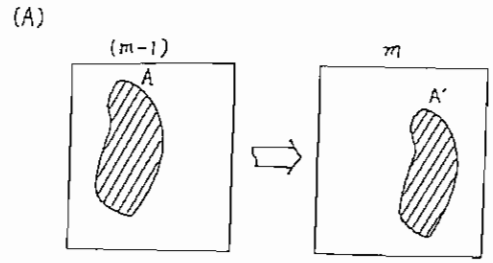


492

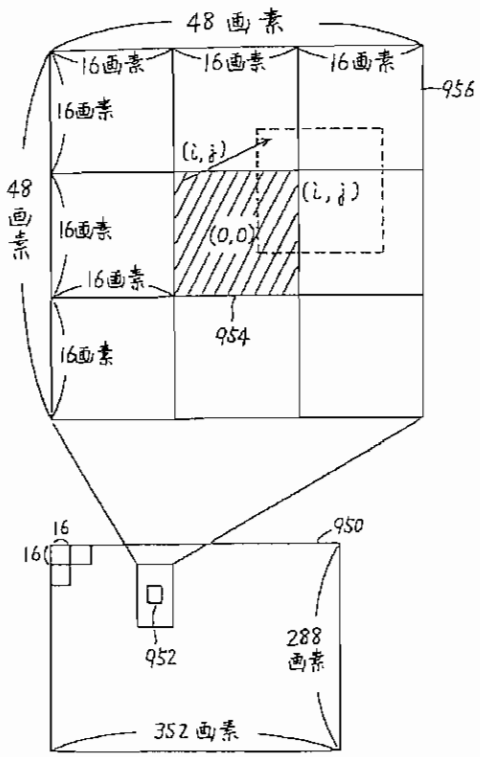
【図１５０】

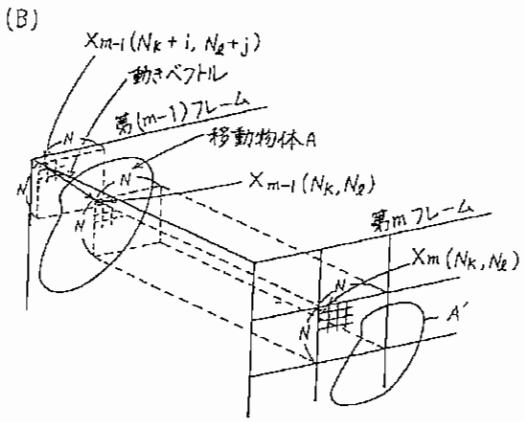


【図１５３】

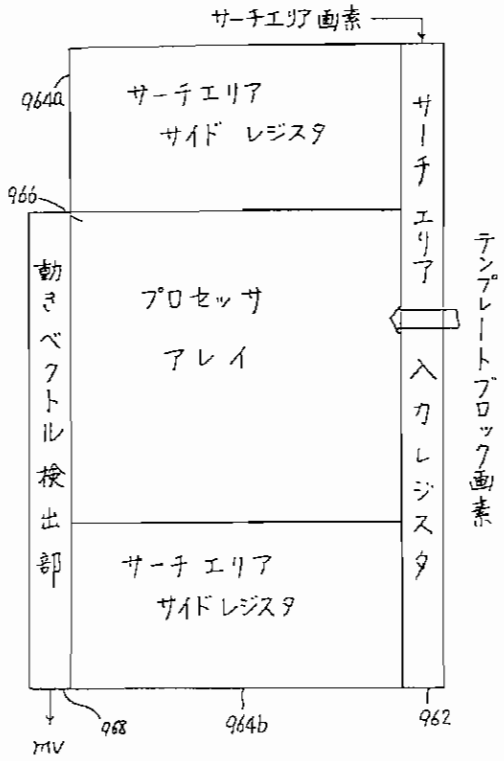


【図１５４】





【図１５５】



フロントページの続き

(72)発明者　中川　伸一
兵庫県伊丹市瑞原４丁目１番地　三菱電機株式会社システムエル・エス・アイ開発研究所内
(72)発明者　松村　哲哉
兵庫県伊丹市瑞原４丁目１番地　三菱電機株式会社システムエル・エス・アイ開発研究所内
(72)発明者　熊木　哲
兵庫県伊丹市瑞原４丁目１番地　三菱電機株式会社システムエル・エス・アイ開発研究所内
(72)発明者　花見　充雄
兵庫県伊丹市瑞原４丁目１番地　三菱電機株式会社システムエル・エス・アイ開発研究所内

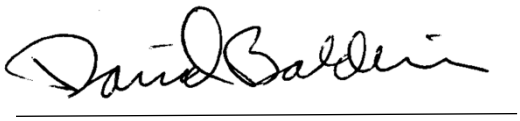
July 12, 2019

Certification

Park IP Translations

I, David Baldwin, am fluent in both the Japanese and English languages. I have translated the Japanese Patent Application Publication No. H7-250328 from Japanese to English. I hereby certify that the translation is accurate and complete to the best of my judgment and ability. I understand that willful false statements and the like are punishable by fine or imprisonment, or both, pursuant to 18 U.S.C. § 1001.

I certify under penalty of perjury under the laws of the United States that the foregoing is true and correct.



David Baldwin

Project Number: GGLA_1905_015

15 W. 37th Street 8th Floor
New York, NY 10018
212.581.8870
ParkIP.com