

Low-power H.264 video decoder with graceful degradation

Arnaud Bourge*, Joël Jung*

Philips Digital Systems Laboratories - Paris
51, rue Carnot, BP301, 92156 Suresnes Cedex, France

ABSTRACT

In this paper, we present a novel H.264 video decoder where memory transfers and energy consumption are significantly reduced. A power-friendly solution is indeed required in mobile applications, where autonomy is a key feature. Whereas low-power design is usually regarded as a pure architectural and implementation problem, our solution is based on an algorithmic approach. We first observe that in modern hardware architectures, computational complexity plays a second role in terms of energy dissipation, which is dominated by memory transfers. We thus identify the motion compensation module as the most consuming part of the standard H.264 decoder, due to its numerous memory accesses to the reference frame(s). Through an algorithmic modification that uses concurrent embedded compression techniques and relies on the data structure of the memory, we reduce memory transfers, and hence power dissipation, while maintaining the visual quality. Experimental results prove that the new method is close to the reference H.264 baseline decoder both in terms of objective and subjective measurements. In the meantime, memory transfers have been reduced by 55% in average, implying power savings which lengthen the battery life of the mobile device, increase the reliability of the chip and lower production costs.

Keywords: Low Power, Battery life, H.264, Video Decoding, Memory Transfers, Complexity, Motion Compensation, Embedded Compression, Subjective Quality.

1. INTRODUCTION

The evolution of mobile devices shows a fast integration of multimedia functionalities. Today the capture and display of color images at reasonable resolutions are common features in cellular phones. Video is already present in some products (high-end mobile phones, PDAs) and is expected to spread with the emergence of the MMS (Multimedia Messaging System). However, this integration comes at the expense of increasing power consumption. Indeed, mobile devices providing video-decoding capabilities suffer from a shortened battery life, whereas autonomy remains a key-feature in the mobile world. Advances in battery capacities only partially answer this problem, because the complexity of codecs and the requirements in picture size and frame-rate keep increasing too. Directly reducing the energy dissipation of a video decoding circuit is thus crucial. It does not only extend battery life, but also increases the reliability of the chip. Furthermore, power saving allows a significant attenuation of the cooling effort, which simplifies the hardware management and lowers production costs.

Low-power design is usually regarded as a pure architectural problem. For instance, hardware/software partitioning allows an improved management of the power consumption¹ by allocating the most consuming parts to dedicated components. Embedded memories also help because they are located close to the CPU and co-processor, so an access to their data goes through less cable and dissipates less energy². Finally, optimizing the implementation of a given algorithm for a specific platform is primordial³. In this classic approach, however, power matters are considered at the latest stage and result in a non-optimal solution.

On the contrary, we take the energy dissipation into account at the algorithmic stage. This approach, if combined with architectural considerations, leads to the most promising gains⁴ because some algorithms intrinsically have the property to enable power-efficient implementations⁵. The counterpart is that evaluating the actual power consumption at the algorithmic stage is a tricky task that can only result in qualitative estimations. In practice, we translate consumption properties of a generic platform into guidelines and constraints for the design of algorithms⁶. First of all, low power should not be regarded as a mere complexity problem. Today, power consumption is led by memory transfers, and especially by accesses to an external memory. Computations only represent a minor (though not negligible) portion of

* {arnaud.bourge ; joel.jung }@philips.com

the overall dissipation⁷. As a consequence, a low-power video-decoding algorithm should mainly minimize the number of accesses to the memory, even at the expense of a moderate complexity increase⁸. Hence, we do not measure performance through a classical rate-distortion trade-off, but through a power-rate-distortion criterion. It means that a loss in visual quality can be compensated by a gain in power consumption.

We have chosen to start from the H.264 decoding scheme - that is the state-of-the-art standard in terms of coding efficiency - at the Baseline Profile, which is intended to be used in the mobile world. In this paper, we present a novel H.264 video decoder where memory transfers and power consumption are significantly reduced. Section 2 is a preliminary study. We present the state-of-the-art in low-power video decoding, show that the motion compensation is the most costly module, explain why we choose to apply embedded compression, and highlight some difficulties inherent to this kind of scheme. In Section 3 we define our strategy, which consists in imposing some specific constraints to the embedded compression, and we describe our algorithmic solution. Section 4 is dedicated to the experimental results, where we show that the memory size can be reduced by a factor of 3, and the associated memory transfers by a factor of 2.3 while maintaining the visual quality. Results also include complexity evaluations, rate-distortion curves and subjective quality assessments. Section 5 concludes this paper.

2. PRELIMINARY STUDY

2.1. State of the art

We have seen in the introduction some examples of low-power decoder designs based on architectural optimizations^{1,2,3}. Although this is the most common approach, our method is more focused on the algorithmic side. Following the same principle, authors have built their decompression algorithm based on underlying hardware properties to minimize power dissipation. Most of them propose computational simplifications, but this method is not sufficient anymore to reduce energy consumption. Others put the strain on minimizing data transfers, both spatially and temporally. In the spatial domain, isometries were thus proposed as a way to significantly reduced a vector quantization (VQ) codebook⁹. Moreover in¹⁰, “power scalability”, depicted as the ability for the codec to change its power consumption depending on the available bandwidth and on the required video quality, is introduced through the use of a hierarchical VQ. Yet we will see that the spatial domain is not the most critical path regarding memory transfers in a conventional decoder. In the temporal domain, the Motion Estimation / Motion Compensation can be replaced by a so-called Conditional Replenishment¹¹ but at the cost of compression performance. Alternatively, an efficient trade-off between data transfers and coding efficiency is introduced in⁶ with the notion of “Collocated Motion Estimation”.

All the above solutions imply the use of a non-conventional encoder to generate the incoming bit-stream. They are not applicable in a context where interoperability and exchanges impose the use of an accepted standard, which is generally the case in mobile communications. Strictly speaking, one cannot come up with a completely new scheme only for the decoder: the corresponding encoding scheme must be accepted too. However, it is possible to decode a standard bit-stream in a manner that differs from the standard algorithm. The resulting sequence is necessarily degraded, but a sound control of the modifications can keep the loss small while some costly operations are saved. In fact this notion of “graceful degradation” introduces a trade-off between the quality of the reconstruction and the cost of the decoding¹². Each modification of the standard scheme introduces a deviation that impacts the visual result. The goal is then to find modifications that maximize the gain in complexity while minimizing the visual losses. Finally the algorithm can adapt itself to the available resources: depending on the CPU utilization, some operations are done, simplified or even skipped. For instance the Inverse DCT can be truncated, or the sub-pixel interpolation filters of the motion compensation can be approximated.

2.2. Identification of the power-costly module: Motion Compensation

We have seen that in the literature “graceful degradation” is commonly applied to computational complexity. We extend this notion to power consumption by adapting the decoding scheme to the available energy resources, thus introducing power-related graceful degradation. We start with the identification of the power-consuming part of the H.264 decoder, which we will then modify.

Memory transfers, and especially accesses to an external memory, have a major influence on power consumption⁸. Computations (arithmetic and logical operations) still represent a significant portion of the overall dissipation, but there

is a trend towards a reduction of its relative weight⁷. The design of low-power algorithms is thus primarily a matter of minimizing memory transfers, which is closely related to bandwidth reduction between the memory and the processing modules. This is especially true in hardware-only and in mixed hardware/software solutions. In software-only environments, however, the power repartition between memory transfers and computations is more balanced and difficult to measure.

In the scope of an H.264 video decoder, the Motion Compensation consumes most of the memory transfers. This is because the other parts of the decoder can generally be performed locally, on a block-by-block basis. In this paper, the term “Motion Compensation” (MC) defines all the operations (logical, arithmetic and memory) that are part of the feedback loop of the predictive scheme, as shown in Figure 1. It includes storing the decoded frame to be used as a reference for the next frame(s), calculating the Motion Vectors (MV), reading the memory to get the reference pixels, and finally performing the interpolation filters that generate the sub-pixel grid.

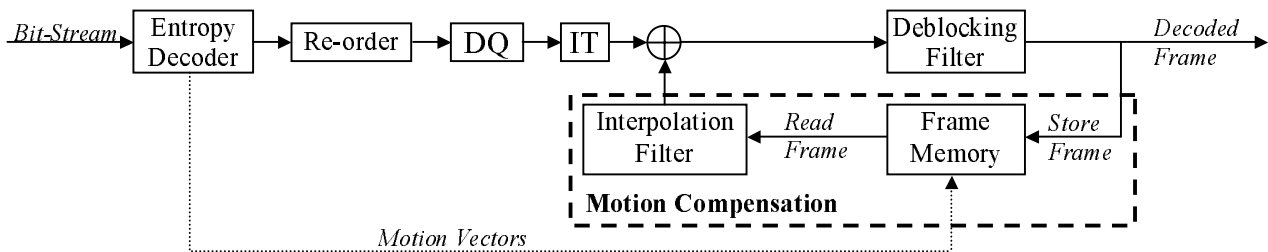


Figure 1: General scheme of a standard H.264 decoder.

The storing of the latest decoded frame is unavoidable in the standard scheme because all its pixels may serve as references for the next frame(s). In CIF, with a 4:2:0 format, this implies $352 \times 288 \times 1.5 = 152,064$ bytes to be stored. Moreover, even in the Baseline Profile, up to 5 preceding frames are used as references, so with the currently decoded frame there are 6 frames in the memory. Consequently, a 1MB external memory is required. It consumes much more power than a small internal memory, partly due to longer connections. During the decoding of one frame, around 300kB of data have to be transferred to and from this memory (read/write operations). No other module needs to store and access so much data: most of them can be performed on the fly, except the deblocking filter in case FMO (Flexible Macroblock Ordering) is enabled. We do not consider this possibility in the remaining of this paper. The Motion Compensation is thus responsible for a major part of the bandwidth between the external memory and the processing modules.

2.3. An existing technique for reducing memory size and transfers in the MC: Embedded Compression

Recent work has put the strain on reducing this bandwidth by presenting an appropriate memory management¹³. The pixels of the stored reference frame(s) are mapped to the structure of a DDR-SDRAM block-by-block instead of line-by-line, yielding excellent results. This method optimized the implementation of the MC without changing the decoding algorithm itself. It was then proposed to go further by using embedded compression techniques¹⁴. Embedded Compression (EC) consists in compressing the reference frame(s) in the feedback loop of a conventional predictive video coding scheme¹⁵ as depicted in Figure 2. The additional encoder and decoder that perform the EC have a computational cost (complexity). This cost must be moderate, because in an extreme case the associated increase in CPU dissipation could exceed the expected power savings on memory accesses.

On quality aspects, if the selected compression technique is lossless it has no impact on the reconstructed sequence, but of course the compression ratio (and thus the expected gain) is limited (two at most). On the contrary if the compression is lossy the memory savings are much higher. It comes at the expense of a quality degradation that worsens along a Group of Pictures (GoP). The reference frames are compressed with loss, so the decoding of the next frame is erroneous. The predictive structure of video coding schemes make this reconstruction error propagate and accumulate: visual quality gets inexorably poorer and poorer. This phenomenon is known as the drift effect. Designing a low-power decoder with embedded compression, we have to minimize this unwanted artifact that is temporally very annoying.

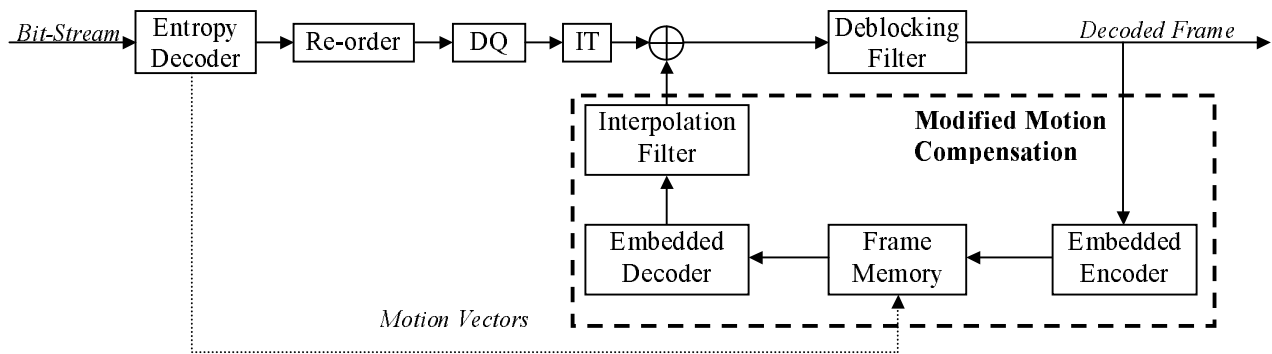


Figure 2: General scheme of an H.264 decoder with Embedded Compression.

2.4. Drawback of Embedded Compression: data overhead

When the goal of Embedded Compression (EC) is purely to reduce the required memory size for cost reasons, virtually any efficient still image compression technique can be performed. In practice, block-based schemes are convenient because they match the block-oriented structure of the incoming bit-stream, allowing on-the-fly processing. However, we also aim at reducing the bandwidth, and the relation between the Compression Ratio (CR) of the EC and the gain in memory transfers is not direct, due to overhead.

We define overhead as the ratio between the number of pixels that are actually accessed during the MC of a block and the number of pixels that are really useful in the reference block. This ratio is superior to 1 because of the block-based nature of EC techniques. For instance let us consider that the reference frames are encoded on an 8x8 block grid. The reference frame no longer has a pixel-oriented structure, but a block-oriented one (Figure 3): to access the value of one pixel we must fetch and decode the whole block it belongs to.

This is not a problem as long as the reference pixels in the MC form blocks that are aligned with the grid of the EC, but it becomes annoying when the motion vector points to a reference that overlaps several encoded blocks. It was already true with previous standards¹⁵ but it becomes even more serious with H.264. Indeed, a 16x16 macroblock can be divided in 16x8, 8x16 or 8x8 blocks during the MC. Any 8x8 block can then be sub-divided into 8x4, 4x8 or 4x4 sub-blocks. In the case depicted in Figure 4, this has a dramatic impact on the overhead. Indeed the compensated block is a 4x4 block (allowed in H.264) that is not at all aligned with the 8x8 grid. Four encoded blocks, corresponding to 256 pixels, have thus to be read and decoded just to get the useful 16 pixels. The overhead in that worst-case scenario is 16. The long 6-tap filters that perform sub-pixel interpolation also contribute to this phenomenon.

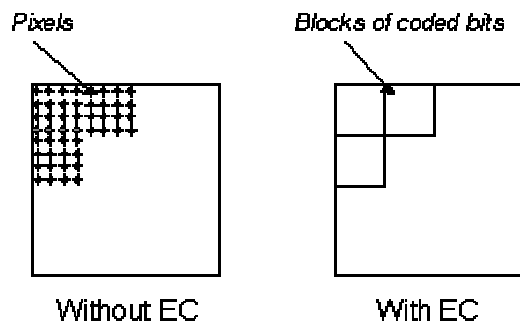


Figure 3: Block-oriented frame structure.

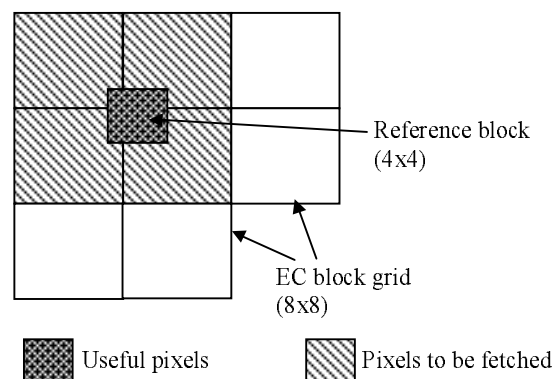


Figure 4: Overhead example.

3. PROPOSED LOW POWER H.264 DECODER

We propose a low-power H.264 decoder based on embedded compression. Its high-level scheme is similar to the one depicted in Figure 2, and we impose some extra constraints to obtain an actual reduction of the memory transfers. Two techniques that fulfill these constraints are then detailed. Our solution takes benefits from both techniques via an efficient competitive mode.

3.1. Requirements for the embedded codec and consequences

To overcome overhead, we impose three main constraints on our EC algorithm:

- 1- the compression is block-based, and each block is independently encoded;
- 2- the compression ratio is fixed for all blocks and maps the memory structure;
- 3- luminance and chrominance planes are jointly stored in the memory.

First, having a block-based EC fits well with the structure of the standard decoder since it allows on-the-fly processing and avoids extra buffering. Furthermore, independent encoding of the blocks ensures that a limited amount of data must be fetched to recover some pixels pointed by the current motion vector. This excludes any kind of prediction between blocks, in particular spatial prediction, because one would have to load and decode also the previous blocks in the reference frame, which would annihilate the desired effect of memory transfer reduction.

The second constraint enables fast and easy access to any block in the reference frame. The start and end addresses of each encoded block are known, so that they do not have to be retrieved in an extra table in memory. The full benefit in terms of transfers and power comes when this constraint is coupled with a good data mapping strategy. In a typical SDRAM, memory addresses point to words of N bits (e.g. $N=32$), where N corresponds to the bus width. Moreover, data can be accessed by bursts of M consecutive words. Long bursts dissipate less energy than small ones relatively to the amount of transferred data. Generally, the maximum burst length is $M=8$ words (256 bits in the case of a 32-bit data bus). Hence, we set a fixed compression ratio CR so that the pixels of the source block fit in the length of one (or multiple) burst(s).

The third constraint increases data locality since motion vectors are identical for luminance and chrominance. Thus, pixels of the three planes can be accessed in one transfer, saving much energy in data-bus communications. We work with 4:2:0 progressive materials, so an 8×8 block in the pixel domain contains 512 bits of luminance and 256 bits of chrominance data. It exactly fits in one 8-word burst of 256 bits with the above memory structure if we apply a compression factor of 3.

While they clearly act in favor of a limited overhead effect, our first two constraints make the efficiency of our scheme difficult in terms of compression ratio and visual quality. Indeed, we cannot benefit from correlations between adjacent blocks. Furthermore, the fixed CR implies that too few bits might be used to correctly encode complex blocks while some easy blocks can certainly be losslessly encoded with less than 256 bits. On the contrary, the third constraint gives a double advantage since it decreases power through a reduction of the memory requests and also enables a joint bit-allocation between the Y, U and V components, which allows more flexibility for coding efficiency.

3.2. A competitive embedded compression scheme

The above constraints limit the power dissipation due to memory accesses by imposing a block-based processing that fits well with the data structure of conventional memories and with the communication properties of the data bus. Once these constraints have been set, the choice of an embedded compression algorithm purely depends on coding efficiency. We have built a scheme that uses a competition between two complementary solutions through an *a priori* selection: a modified version of the Intra mode of H.264, and a zonal bit-plane coding scheme¹⁶. The former is known for its efficiency and is especially convenient to implement in an H.264 codec platform since the main functions are already available. The latter is bit-rate scalable and thus does not need any rate regulation module, resulting in an optimal use of the allocated bit budget.

Modified H.264 Intra mode

In the H.264 standard, if a macroblock is encoded in Intra mode, a spatial prediction is formed based on the previously encoded blocks. 4x4 integer transform, quantization and entropy coding are then applied to the prediction error. In our case we cannot enable the complete prediction scheme, but a restrained version is possible. As a matter of fact we encode blocks of size 8x8 pixels where there are four 4x4 subblocks, so spatial predictions between these subblocks are performed. For instance the up-right subblock can be predicted from the pixels in the right column of the first subblock, either from their average value ("DC prediction") or by horizontal padding (Horizontal prediction"). Only the first subblock cannot at all be predicted from the other ones. Instead, we extract its DC coefficient truncated on 8 bits and use it as a predictor. The DC coefficient of the first subblock is thus encoded with a fair and constant precision, independently of the chosen quantization step. It helps limiting the drift effect. Figure 5 summarizes the available prediction modes for each subblock. One can refer to the H.264 standard for further information about each mode.

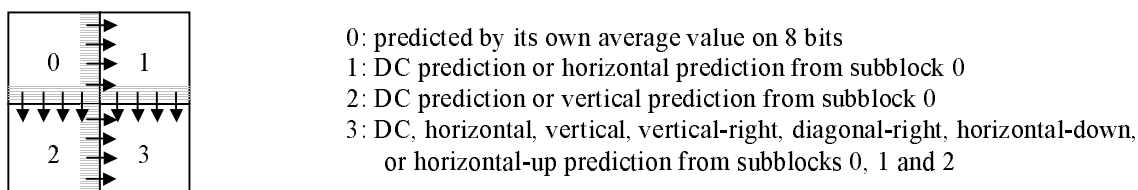


Figure 5: Spatial prediction modes available for each 4x4 subblock in the modified H.264 Intra mode.

One bit is enough to transmit the prediction mode of subblock 1, and another one for subblock 2, while 3 bits are needed for subblock 3, leading to a total of 5 bits. Chrominance pixels directly form 4x4 blocks, so we do not perform any spatial prediction there, except that their average value truncated on 8 bits is also losslessly encoded for each UV component. Then the 4x4 transform and the quantization process are similar to the standard, but with a limited number of 32 possible quantization steps. The Level and Run tables are encoded using a variable-length coding (VLC). In fact, a complex entropy coding (e.g. the Context-based Adaptive Binary Arithmetic Coding - CABAC) is not justified in this application because the context would have to be reset after each block, leading to a sub-optimal solution. VLC is just as efficient here. QP is adjusted so that the total length of the bit-stream does not exceed the chosen value (in our case 256 bits or a burst of eight 32-bit words). Finally, the bit-stream corresponding to one block is composed as depicted in Figure 6.

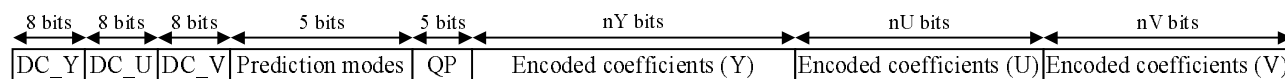


Figure 6: Bit-stream composition in the modified H.264 Intra mode.

Scalable Rmax/Cmax mode

The other scheme¹⁶ that we have studied encodes DCT coefficients bit plane by bit plane, starting with the Most Significant Bits. As in the above solution, we encode the DC coefficients losslessly to counter the drift effect. For the AC coefficients, compression is achieved through a zonal coding: for each bit plane the significance map of the previously insignificant coefficients is represented by the maximum row Rmax and the maximum column Cmax where a bit is equal to 1 (see Figure 7). The data beyond this zone does not need to be transmitted. Since the energy usually lies in the low frequency coefficients, this technique implies a good bit-rate reduction. Its great advantage is to provide a fine-grain scalability: the process stops when the bit budget is exhausted. This property, although it generally comes at the expense of coding efficiency, is important in our context since we have imposed a severe rate limitation on a block basis. With the conventional quantization/VLC solution above, some bits are always wasted, while here all the available bits can efficiently be used.

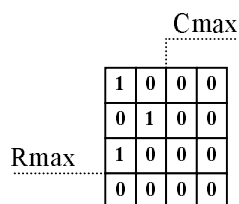


Figure 7: Zonal bit-plane coding.

Competition mode

In practice, we have observed that although the two depicted algorithms give encouraging results, none is better than the other in all conditions and for all sequences. More precisely, their respective artifacts are very dissimilar and are located in regions with different properties. The modified Intra mode tends to blur the picture and to lose details in very textured areas. On the contrary, the scalable solution gives a sharper rendering in edges and rich textures but generates an unpleasing noise/ringing effect in smoother regions, due to the bit-plane truncation. Consequently, our solution is based on the competition between these two modes, to achieve the best performance. In order to limit complexity, we select the encoding mode through an *a priori* criterion: if the activity (or energy) of the current block is higher than a given threshold, then the scalable scheme is chosen, otherwise the modified H.264 Intra is applied. *A priori* selection is important because it generates a very limited computational overhead compared to an *a posteriori* solution where both schemes are processed before one gets selected.

4. EXPERIMENTAL RESULTS

We present our results on several dimensions: the memory aspects (size and transfers), the complexity issue, the objective and the subjective quality. Our test set is composed of CIF sequences at 25 frames/s with various contents as given in Table 1. We have encoded them with the JM6 reference software of H.264, set to respect the Baseline Profile, with an Intra period of 25 frames. Bit-rates are in the range [150...512] kb/s. This spatio-temporal resolution, these sequences and the encoding conditions were chosen as representative of the targeted applications.

Foreman	Face close-up, moderate motion.
Flower	Slow camera motion, rich-textured garden, cloudy sky.
Silent	Talking person moving hands and arms, painted background.
Stefan	Tennis scene, fast motions.
Weather	Weather forecast, synthetic map (background), talking person (foreground).

Table 1: Test sequences.

4.1. Memory gain

Our primary goal is to reduce power consumption through a decrease of the memory transfers. We have seen that we apply a Compression Ratio CR of 3 so that an 8x8 block maps a data burst of 256 bits. The first consequence is that the memory occupancy of the reference frame(s) is divided by three. Hence a smaller memory can be implemented, which is very positive in terms of both costs and consumption. Indeed, smaller memories can be embedded closer to the CPU, leading to less cable dissipation during accesses. Concerning transfers (and thus bandwidth), the gain is not so direct due to overhead (see paragraph 2.4). In the following, we describe how we measure it.

Overhead: statistical prediction

The memory accesses are divided by CR/Ovh at the reading side, where Ovh is the overhead. At the storing side, this overhead issue is irrelevant and transfers are thus divided by CR. Equation 1 expresses the global gain in memory transfers (load/store) with the approximation that there are as many loads as stores in the standard decoder. One can see that overhead can theoretically counter-balance the benefits of EC regarding memory transfers. We have to measure it precisely to assess the memory bandwidth that is actually saved.

$$Transfer_Gain = \frac{Memory_Store_{EC} + Memory_Load_{EC}}{Memory_Store_{Std} + Memory_Load_{Std}} = \frac{1 + Ovh}{2 * CR} \quad \text{Equation 1}$$

Overhead strictly depends on the block size of the EC, on the size of the current motion-compensated block, and on the value of its motion vector. It is a part of our contribution to quantify their impact through a statistical analysis of the sequences of the test set.

There are four main inter prediction modes for luminance pixels in H.264: 16x16, 16x8, 8x16 and 8x8. The 8x8 mode can be further decomposed into 4x4, 8x4 and 4x8 partitions. For each mode and sub-mode, we have counted the occurrences when a reference block is aligned with the chosen EC block grid (8x8). If it is not, then there is a separation between horizontal misalignment, vertical misalignment, and both. In each case, the number of encoded blocks to retrieve (and thus the number of pixels to get) is different. Then the statistical apparition of each mode is also computed. A synthesis permits to extract the average overhead from our test set.

Figure 8 illustrates a typical case, here for the Stefan sequence, where we also indicate the equivalent results for an EC block-grid of 4x4 and 16x16. A simple analysis shows that the result is qualitatively coherent. The 4x4 grid generates fewer overheads than the other ones because it is closer to pixel accuracy. Furthermore, for all grid sizes, overheads increase with the bit-rate. This is due to the H.264 encoder that allows macroblock partitioning and large motion vectors at high rates (small quantization step) while it favors the null vector with a 16x16 partition at low rates.

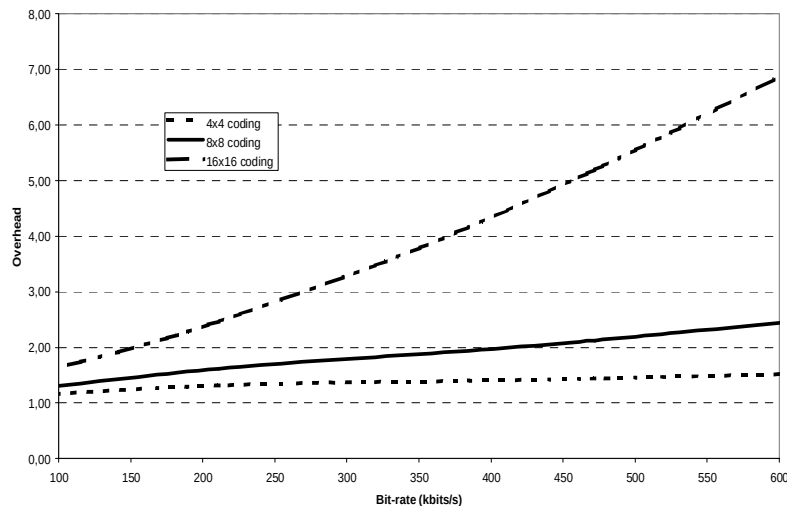


Figure 8: overhead vs bit-rate (Stefan sequence).

Sequence	4x4 EC Grid	8x8 EC Grid	16x16 EC Grid
Foreman	1.31	1.77	3.69
Flower	1.30	1.74	3.77
News	1.14	1.51	2.78
Silent	1.17	1.50	3.22
Stefan	1.51	2.44	6.95
Weather	1.17	1.49	3.18
All	1.27	1.73	3.93

Table 2: overhead vs EC block-grid for each sequence.

Table 2 summarizes the statistical analysis for each sequence of the test set. Not surprisingly, still sequences generate the smallest overhead values (News, Weather) because the motion vector is often equal to zero. On the contrary, a fast sequence like Stefan generates much overhead. In the remaining of the experiments we only present results for the 8x8 EC block grid, which turned out to offer the best trade-off between transfer overhead (that is minimized if the block grid is small) and coding efficiency (that prefers larger blocks).

Memory transfers

Thanks to the formula of Equation 1 and to the statistical results of Table 2, we can derive the average gain in memory transfers for each sequence. For the 8x8 grid, the transfers gain lies between 0.40 for the Weather sequence and 0.55 for the Stefan sequence, respectively corresponding to a 60% and a 45% reduction. The average gain for all the sequences is about 0.45 (or 55% of reduction), corresponding to a gain of 0.33 for the storing and 0.57 for the reading of a reference frame. These figures correspond to a Compression Ratio (CR) of 3 and lead to very good visual results as we

will see in section 4.4. Higher CR values, and thus higher gains, can be achieved. For instance CR=8 implies an average reduction of 80% for memory transfers at the expense of more visible degradations.

4.2. Complexity evaluation (CFDG)

Although memory transfers represent the major part of the energy dissipation in modern architecture, computational and logical operations are not negligible. By applying Embedded Compression, we have added some modules to the conventional standard decoder, and consequently increased the complexity. In this paragraph we estimate the number of added operations through a CFDG analysis.

A Control Flow Graph (CFG) is an abstract data structure, a representation of a procedure or program. It consists of nodes performing operations and exchanging data. Each node in the graph represents a basic block, i.e. a straight-line piece of code without any jump. A CFDG is a CFG where Data dependencies are highlighted. In general, CF(D)G are internally built-up by compilers to optimize binary executables. In our case, we found it was a good tool to precisely decompose an algorithm at a high-level. It allows analyzing its complexity, memory accesses, level of data dependencies and degree of parallelism. All these notions directly give indications on the ease to implement a low-power system based on the studied algorithm.

The CFDG methodology has been used to evaluate the average complexity of the reference MC and of the proposed solution with EC. The results given in Table 3 represent the number of additions (Add), multiplex (Mux, related to 'if' statements) and multiplications (Mul) for performing the MC of one frame. It takes into account statistics about the motion vectors (including sub-pixel interpolation). The numbers of memory read and write are taken from the study of the previous paragraph, and also take into account the storing and reading of the motion vectors from the previous neighboring blocks (for motion vector decoding).

	Read (x1000)	Write (x1000)	Add (x1000)	Mux (x1000)	Mul (x1000)
MC of the H.264 decoder	226	152	709	1,060	200
Proposed MC with Embedded Compression	138	53	5,017	1,938	504
Ratio	0.61	0.35	7.00	1.72	2.52

Table 3: Complexity of the proposed MC compared to the standard one.

One can notice that EC performs a lot of additions: they are multiplied by 7 compared to the conventional MC. However, additions are less costly than multiplications. In general, one considers that one multiplication is three times as complex (and power-consuming) as an addition and a multiplex in most processing units. This approximation leads to an average ratio of 3.57 more operations (Add, Mux, Mul) for the proposed method. Finally, some studies estimate that the Motion Compensation represents around 40% of the total complexity of a Baseline H.264 decoder. Hence, our solution is globally between 1.5 and 2 times as complex as the standard one, while dividing the memory transfers by more than 2.

4.3. Objective quality (PSNR)

Embedded Compression generates an unavoidable reconstruction drift at the decoder, which impacts the resulting quality. However, in the introduction we have stressed the fact that we judge our algorithms on a power-rate-distortion basis, meaning that a large gain in power can justify a slight increase in distortion. We have to quantify this distortion to assess the acceptability of our solution.

Figure 9 and Figure 10 depict the rate-distortion graphs of the standard decoder and of the proposed one, for two representative sequences of the test set (Flower and Foreman). The gap between the two methods increases with the bit-rate because the decoded frames are better preserved and contain more details at higher rates, implying a non-negligible loss of data during the Embedded Compression. On the contrary, the decoded pictures at lower rates are easier to encode by the EC module because they are already degraded anyway.

The average PSNR loss on all the sequences of the test set is 0.7 dB. It is null at the low end of the bit-rate range and can reach more than 2.0 dB around 512 kbits/s. However, some authors have remarked that the visual impact of the EC is not necessarily as high as the gaps in the rate-distortion curves¹⁴.

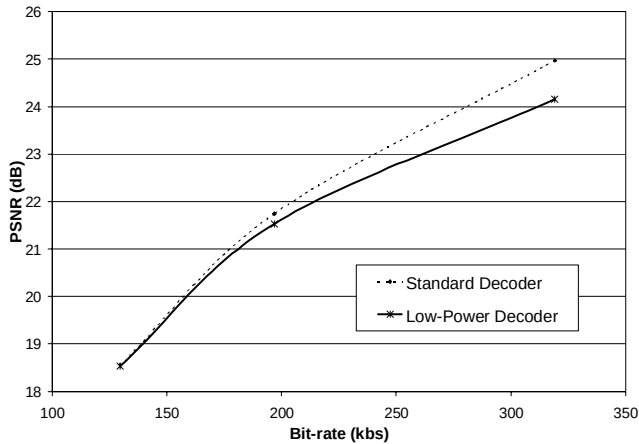


Figure 9: Flower sequence.

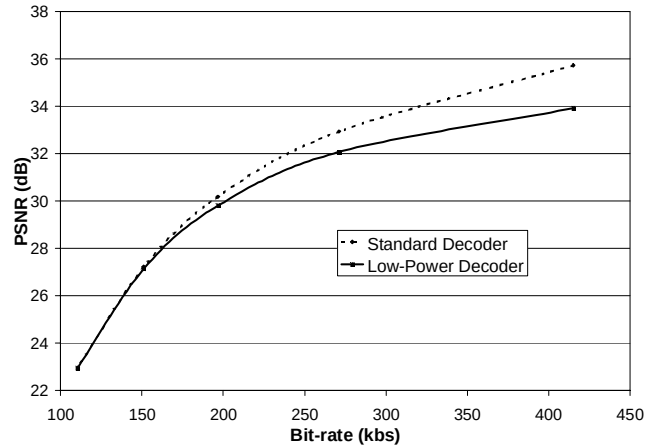


Figure 10: Foreman sequence.

4.4. Subjective quality (visual tests)

Rate-distortion graphs are a useful way of evaluating coding efficiency, especially during the process of designing and tuning algorithms. Yet, PSNR curves do not give a direct idea about the actual differences in visual quality between two solutions and can even be misleading in some special cases. Moreover, the subjective quality that viewers feel while looking at a sequence is much influenced by the viewing conditions. For instance the visibility of some artifacts (blocking, ringing, temporal drift, etc.) varies much according to whether people look at the palm-sized LCD screen of a PDA or at a professional monitor in a laboratory environment. Subjective tests are thus needed. Our solution targets mobile applications, so we performed the subjective tests on a portable device, following as much as possible the known protocols and recommendations proposed for television applications¹⁷.

Tests were designed for Double Stimulus Impairment Scale (DSIS). Each test consists of a pair of stimuli: one sequence processed with the reference solution (in our case the standard decoder) and with the proposed one. Each pair is shown in this order, twice. Let us notice that the reference is not necessarily “artifact-free”: typical coding artifacts (blocking, ringing, mosquito noise, lack of sharpness, temporal inconsistency) can be present. The viewers are asked to score the impairment level of the proposed technique regarding this reference. Table 4 shows the scoring scale.

Score	Impairment
5	Imperceptible
4	Perceptible but not annoying
3	Slightly annoying
2	Annoying
1	Very annoying

Table 4: Scoring scale.

The device is a mobile video player with a 320x240 LCD-TFT screen and a depth of 16 bits/pixel. The viewing distance is not imposed. Both expert and naïve viewers are asked to hold the mobile device at a convenient distance and to keep this distance constant during the test. The room illumination is low. Finally, the Mean Opinion Score (MOS) is computed. Results are summarized in Table 5.

Category	Number of viewers	Mean Opinion Score	Variance	Min Score	Max Score
Experts	9	4.0	0.4	2	5
Naïves	5	4.3	0.5	3	5

Table 5: Subjective Tests results.

As usual, expert viewers are both more sensitive to artifacts and more consistent in their judgment than naïve viewers, leading to a smaller MOS and a smaller variance. However, the scores are rather coherent across the two categories, and the impairment level of the low-power solution is not considered annoying although it is perceptible. The proposed scheme is thus only 1 JND (Just Noticeable Difference¹⁸) away from the reference decoder, which is an excellent result.

4.5. General interpretation

In this section, the proposed low-power H.264 decoder has been evaluated regarding memory (size and transfers), complexity, objective and subjective quality. On the one hand, the cost of Embedded Compression is quite high in terms of computational operations compared to the standard Motion Compensation, leading to an estimated 1.7 complexity factor for the whole decoder. On the other hand, memory transfers, which are predominant in the global energy dissipation, have been divided by more than 2. As stated in section 2.2, this undoubtedly leads to a global gain in power consumption, especially in a hardware-only or a mixed software/hardware environment. It is difficult to quantitatively evaluate this gain: several tools exist⁶, using various consumption models, but they need the implementation process to be well advanced. Finally, subjective tests proved that the proposed low-power decoder provides video sequences that are visually very close to the standard decoder. This scheme thus provides an excellent trade-off between power consumption and visual quality. It is possible to achieve larger gains by increasing the Compression Ratio of the EC at the expense of a degraded visual quality, which can be acceptable in some applications.

5. CONCLUSIONS

We have presented a modified H.264 decoder that gracefully degrades the output video quality in order to significantly reduce energy dissipation. We have first observed that the Motion Compensation was the most power-consuming part of the decoder, due to its numerous accesses to the frame memory. Embedded Compression (EC) was then identified as a promising technique to reduce these transfers. In order to limit the data overhead that is inherent to EC, we have imposed some requirements that help mapping the EC grid with the memory structure. Once these constraints have been set, we have applied a compression factor of 3 on the reference frames, using a combination of two complementary EC techniques. The selection criterion is *a priori* and low-cost.

Experimental results show that memory transfers, which dominate power consumption, are reduced by 55% in average for CIF sequences in mobile application conditions. The computational complexity of the decoder is globally increased by 1.7, which should not counter-balance the power gain due to memory transfers. Furthermore, subjective tests proved that our solution maintains the visual quality of a standard decoder. The proposed decoder thus provides an excellent trade-off between power consumption and visual quality, making it an advantageous solution for mobile applications.

ACKNOWLEDGEMENTS

The authors would like to acknowledge Yi Li and Carolina Miro, from the Platform & Integration group of the Philips Digital Systems Labs in Paris, for their work on the evaluation of the complexity of the different decoders used in this paper.

REFERENCES

1. P. van der Wolf, P. Lieveverse, M. Goel, D. La Hei, K. Vissers, "An MPEG-2 decoder case study as a driver for a system level design methodology", Proc. of the 7th Int. Workshop on Hardware/Software Codesign, pp.33-37, Roma, Italy, March 1999.
2. C.W. Yoon *et al.*, "An 80/20 MHz 160 mW multimedia processor integrated with embedded DRAM, MPEG-4 accelerator and 3D rendering engine for mobile applications", IEEE Journal of Solid-State Circuits, Vol. 36, Issue 11, pp. 1758–1767, November 2001.
3. L. Nachtergaele, F. Catthoor, B. Kapoor, S. Janssens and D. Moolenaar, "Low-Power Data Transfer and Exploration for H.263 Video Decoder System", IEEE Journal on Selected Areas in Communications, Vol. 16, No. 1, January 1998.
4. W. Badawy and M. Bayoumi, "Algorithm-based Low-Power VLSI Architecture for 2D-Mesh Video-Object Motion Tracking", IEEE Trans. On Circuits and Systems for Video Technology, Vol. 12, April 2002.
5. A. Chandrakasan and W. Brodersen, "Minimizing Power Consumption in Digital CMOS Circuits", Proc. Of the IEEE, Vol. 83, April 1995.

6. J. Jung and A. Bourge, "Power Scalable Video Encoder for Mobile Devices based on Collocated Motion Estimation", Proc. of VCIP, Electronic Imaging, San Jose, California, USA, January 2004.
7. B. Kapoor, "Low Power Memory Architectures for Video Applications", Proceedings of the 8th Great Lakes Symposium on VLSI, pp 2-7, Lafayette, LA, USA, 19-21 February 1998.
8. T.H. Meng, A.C. Hung, E.K. Tsem and B.M. Gordon, "Low-Power Signal Processing System for Wireless Applications", IEEE Personal Communications (USA), vol.5, no.3, pp.20-31, June 1998.
9. K. Masselos, P. Merakos, T. Stouraitis and C.E. Goutis, "A Novel Algorithm for Low-Power Image and Video Coding", IEEE Trans. On CSVT, Vol.8, No.3, pp.258-263, June 1998.
10. N. Chaddha and M. Vishwanath, "A Low Power Video Encoder with Power, Memory and Bandwidth Scalability", 9th Int. Conf. On VLSI Design, pp. 358-263, January 1996.
11. W. Namgoong, N. Chaddha and T. Meng, "Low-Power Video Encoder/Decoder using Wavelets/TSVQ with Conditional Replenishment", ICASSP'96, Vol.6, pp. 3240-3243, May 1996.
12. Y. Chen, Z. Zhong, T.H. Lan, S. Peng and K. van Zon, "Regulated Complexity-Scalable MPEG-2 video decoding for Media Processors", IEEE Trans. On Circuits and Systems for Video Technology, Vol. 12, No. 8, August 2002.
13. E. Jaspers and P.H.N. de With, "Bandwidth reduction for video processing in consumer systems", IEEE Trans. On Consumer Electronics, Vol. 47, No. 4, pp. 885-894, November 2001.
14. E. Jaspers and P.H.N. de With, "Embedded compression for memory resource reduction in MPEG systems", Proc. Of 3rd Benelux IEEE Signal Processing Symposium, Leuven, Belgium, March 2002.
15. P.H.N. de With, P.H. Frencken and M. van der Schaar, "An MPEG decoder with embedded compression for memory reduction", International Conference on Consumer Electronics, Digest of Technical Papers, pp. 60-62, June 1998.
16. R.J. van der Vleuten, "Low-complexity lossless and fine-granularity scalable near-lossless compression of color images", Proceedings of the Data Compression Conference, pp.477, April 2002.
17. "Methodology for the subjective assessment of the quality of television pictures", ITU-R BT-500.10, updated in 2000.
18. J. Lubin and D. Fibush, "Sarnof JND vision model", T1A1.5 Working Group Doc. No. 97-612, T1 Standards Committee, 1997.