Express Mailing No.: EV577684659US

PATENT APPLICATION
Kunzler & Associates Docket No.: 2082.2.1

UNITED STATES PATENT APPLICATION

of

R. DREW MAJOR,

AND

MARK B. HURST,

for

APPARATUS, SYSTEM, AND METHOD FOR ADAPTIVE-RATE SHIFTING OF STREAMING CONTENT

fuboTV Media Inc.
Exhibit 1018
fuboTV v. Dish
IPR2024-00918
Ex. 1018-001

APPARATUS, SYSTEM, AND METHOD FOR ADAPTIVE-RATE SHIFTING OF STREAMING CONTENT

CROSS-REFERENCES TO RELATED APPLICATIONS

[0001] This application claims benefit of United States Provisional Patent Application Number 60/566,831 entitled "APPARATUS, SYSTEM, AND METHOD FOR DYNAMIC RATE SHIFTING OF STREAMING CONTENT" and filed on April 30, 2004 for R. Drew Major and Mark B. Hurst, which is incorporated herein by reference.

BACKGROUND OF THE INVENTION

FIELD OF THE INVENTION

[0002] The invention relates to video streaming over packet switched networks such as the Internet, and more particularly relates to adaptive-rate shifting of streaming content over such networks.

DESCRIPTION OF THE RELATED ART

[0003] The Internet is fast becoming a preferred method for distributing media files to end users. It is currently possible to download music or video to computers, cell phones, or practically any network capable device. Many portable media players are equipped with network connections and enabled to play music or videos. The music or video files (hereinafter "media files") can be stored locally on the media player or computer, or streamed or downloaded from a server.

[0004] "Streaming media" refers to technology that delivers content at a rate sufficient for presenting the media to a user in real time as the data is received. The data may be stored in memory temporarily until played and then subsequently deleted. The user has the immediate satisfaction of viewing the requested content without waiting for the media file to completely download. Unfortunately, the audio/video quality that can be received for real time presentation is constrained by the available bandwidth of the user's network

KUNZLER & ASSOCIATES
ATTORNEYS AT LAW
8 East Broadway, SUITE 600
SALT LAKE CITY, UTAH 84101

connection. Streaming may be used to deliver content on demand (previously recorded) or from live broadcasts.

[0005] Alternatively, media files may be downloaded and stored on persistent storage devices, such as hard drives or optical storage, for later presentation. Downloading complete media files can take large amounts of time depending on the network connection. Once downloaded, however, the content can be viewed repeatedly anytime or anywhere. Media files prepared for downloading usually are encoded with a higher quality audio/video than can be delivered in real time. Users generally dislike this option, as they tend to want to see or hear the media file instantaneously.

[0006] Streaming offers the advantage of immediate access to the content but currently sacrifices quality compared with downloading a file of the same content. Streaming also provides the opportunity for a user to select different content for viewing on an ad hoc basis, while downloading is by definition restricted to receiving a specific content selection in its entirety or not at all. Downloading also supports rewind, fast forward, and direct seek operations, while streaming is unable to fully support these functions. Streaming is also vulnerable to network failures or congestion.

[0007] Another technology, known as "progressive downloads," attempts to combine the strengths of the above two technologies. When a progressive download is initiated, the media file download begins, and the media player waits to begin playback until there is enough of the file downloaded that playback can begin with the hope that the remainder of the file will be completely downloaded before playback "catches up." This waiting period before playback can be substantial depending on network conditions, and therefore is not a complete or fully acceptable solution to the problem of media presentation over a network.

[0008] Generally, three basic challenges exist with regard to data transport streaming over a network such as the Internet that has a varying amount of data loss. The first challenge is reliability. Most streaming solutions use a TCP connection, or "virtual circuit," for transmitting data. A TCP connection provides a guaranteed delivery mechanism

KUNZLER & ASSOCIATES
ATTORNEYS AT LAW
8 East Broadway, SUITE 600
SALT LAKE CITY, UTAH 84101

so that data sent from one endpoint will be delivered to the destination, even if portions are lost and retransmitted. A break in the continuity of a TCP connection can have serious consequences when the data must be delivered in real-time. When a network adapter detects delays or losses in a TCP connection, the adapter "backs off" from transmission attempts for a moment and then slowly resumes the original transmission pace. This behavior is an attempt to alleviate the perceived congestion. Such a slowdown is detrimental to the viewing or listening experience of the user and therefore is not acceptable.

[0009] The second challenge to data transport is efficiency. Efficiency refers to how well the user's available bandwidth is used for delivery of the content stream. This measure is directly related to the reliability of the TCP connection. When the TCP connection is suffering reliability problems, a loss of bandwidth utilization results. The measure of efficiency sometimes varies suddenly, and can greatly impact the viewing experience.

[0010] The third challenge is latency. Latency is the time measure form the client's point-of-view, of the interval between when a request is issued and the response data begins to arrive. This value is affected by the network connection's reliability and efficiency, and the processing time required by the origin to prepare the response. A busy or overloaded server, for example, will take more time to process a request. As well as affecting the start time of a particular request, latency has a significant impact on the network throughput of TCP.

[0011] From the foregoing discussion, it should be apparent that a need exists for an apparatus, system, and method that alleviate the problems of reliability, efficiency, and latency. Additionally, such an apparatus, system, and method would offer instantaneous viewing along with the ability to fast forward, rewind, direct seek, and browse multiple streams. Beneficially, such an apparatus, system, and method would utilize multiple connections between a source and destination, requesting varying bitrate streams depending upon network conditions.

KUNZLER & ASSOCIATES
ATTORNEYS AT LAW
8 East Broadway, SUITE 600
SALT LAKE CITY, UTAH 84101

SUMMARY OF THE INVENTION

[0012] The present invention has been developed in response to the present state of the art, and in particular, in response to the problems and needs in the art that have not yet been fully solved by currently available content streaming systems. Accordingly, the present invention has been developed to provide an apparatus, system, and method for adaptive-rate content streaming that overcome many or all of the above-discussed shortcomings in the art.

[0013] The apparatus for adaptive-rate content streaming is provided with a logic unit containing a plurality of modules configured to functionally execute the necessary steps. These modules in the described embodiments include an agent controller module configured to simultaneously request a plurality of streamlets, the agent controller module further configured to continuously monitor streamlet requests and subsequent responses, and accordingly request higher or lower quality streamlets, and a staging module configured to stage the streamlets and arrange the streamlets for playback on a content player.

[0014] The apparatus is further configured, in one embodiment, to establish multiple Transmission Control Protocol (TCP) connections with a content server, and request streamlets of varying bitrates. Each streamlet may further comprise a portion of a content file. Additionally, the agent controller module may be configured to generate a performance factor according to responses from streamlet requests.

[0015] In a further embodiment, the agent controller module is configured to upshift to a higher quality streamlet when the performance factor is greater than a threshold, and the agent controller module determines the higher quality playback can be sustained according to a combination of factors. The factors may include an amount of contiguously available streamlets stored in the staging module, a minimum safety margin, and a current read ahead margin.

[0016] The agent controller module may be configured to downshift to a lower quality streamlet when the performance factor is less than a second threshold. Also, the agent controller module is further configured to anticipate streamlet requests and pre-request

KUNZLER & ASSOCIATES
ATTORNEYS AT LAW
8 East Broadway, SUITE 600
SALT LAKE CITY, UTAH 84101

streamlets to enable fast-forward, skip randomly, and rewind functionality. In one embodiment, the agent controller module is configured to initially request low quality streamlets to enable instant playback of the content file, and subsequent upshifting according to the performance factor.

[0017] A system of the present invention is also presented to adaptive-rate content streaming. In particular, the system, in one embodiment, includes a data communications network, and a content server coupled to the data communications network and having a content module configured to process content and generate a plurality of high and low quality streams. In one embodiment, each of the high and low quality streams may include a plurality of streamlets.

[0018] In a further embodiment, the system also includes an agent controller module configured to simultaneously request a plurality of streamlets, the agent controller module further configured to continuously monitor streamlet requests and subsequent responses, and accordingly request higher or lower quality streamlets, and a staging module configured to stage the streamlets and arrange the streamlets for playback on a content player.

[0019] A method of the present invention is also presented for adaptive-rate content streaming. The method in the disclosed embodiments substantially includes the steps necessary to carry out the functions presented above with respect to the operation of the described apparatus and system. In one embodiment, the method includes simultaneously requesting a plurality of streamlets, continuously monitoring streamlet requests and subsequent responses, and accordingly requesting higher or lower quality streamlets, and staging the streamlets and arranging the streamlets for playback on a content player.

[0020] In a further embodiment, the method may include establishing multiple Transmission Control Protocol (TCP) connections with a content server, and requesting streamlets of varying bitrates. Also, the method may include generating a performance factor according to responses from streamlet requests, upshifting to a higher quality streamlet when the performance factor is greater than a threshold, and determining if the higher quality

KUNZLER & ASSOCIATES
ATTORNEYS AT LAW
8 East Broadway, SUITE 600
SALT LAKE CITY, UTAH 84101

playback can be sustained. Furthermore, the method may include downshifting to a lower quality streamlet when the performance factor is less than a second threshold.

[0021] In one embodiment, the method includes anticipating streamlet requests and pre-requesting streamlets to enable fast-forward, skip randomly, and rewind functionality. The method may also comprise initially requesting low quality streamlets to enable instant playback of a content file, and subsequent upshifting according to the performance factor.

[0022] Reference throughout this specification to features, advantages, or similar language does not imply that all of the features and advantages that may be realized with the present invention should be or are in any single embodiment of the invention. Rather, language referring to the features and advantages is understood to mean that a specific feature, advantage, or characteristic described in connection with an embodiment is included in at least one embodiment of the present invention. Thus, discussion of the features and advantages, and similar language, throughout this specification may, but do not necessarily, refer to the same embodiment.

[0023] Furthermore, the described features, advantages, and characteristics of the invention may be combined in any suitable manner in one or more embodiments. One skilled in the relevant art will recognize that the invention may be practiced without one or more of the specific features or advantages of a particular embodiment. In other instances, additional features and advantages may be recognized in certain embodiments that may not be present in all embodiments of the invention.

[0024] These features and advantages of the present invention will become more fully apparent from the following description and appended claims, or may be learned by the practice of the invention as set forth hereinafter.

KUNZLER & ASSOCIATES
ATTORNEYS AT LAW
8 East Broadway, SUITE 600
SALT LAKE CITY, UTAH 84101

BRIEF DESCRIPTION OF THE DRAWINGS

[0025] In order that the advantages of the invention will be readily understood, a more particular description of the invention briefly described above will be rendered by reference to specific embodiments that are illustrated in the appended drawings. Understanding that these drawings depict only typical embodiments of the invention and are not therefore to be considered to be limiting of its scope, the invention will be described and explained with additional specificity and detail through the use of the accompanying drawings, in which:

[0026] Figure 1 is a schematic block diagram illustrating one embodiment of a system for adaptive rate shifting of streaming content in accordance with the present invention;

[0027] Figure 2a is a schematic block diagram graphically illustrating one embodiment of a content file in accordance with the present invention;

[0028] Figure 2b is a schematic block diagram illustrating one embodiment of a plurality of streams having varying degrees of quality and bandwidth in accordance with the present invention;

[0029] Figure 2c is a schematic block diagram illustrating one embodiment of a stream divided into a plurality of streamlets in accordance with the present invention;

[0030] Figure 3 is a schematic block diagram illustrating one embodiment of a content module in accordance with the present invention;

[0031] Figure 4 is a schematic block diagram graphically illustrating one embodiment of a client module in accordance with the present invention;

[0032] Figure 5 is a schematic flow chart diagram illustrating one embodiment of a method for processing content in accordance with the present invention;

[0033] Figure 6 is a schematic flow chart diagram illustrating one embodiment of a method for playback of a plurality of streamlets in accordance with the present invention; and

KUNZLER & ASSOCIATES
ATTORNEYS AT LAW
8 East Broadway, SUITE 600
SALT LAKE CITY, UTAH 84101

[0034] Figure 7 is a schematic flow chart diagram illustrating one embodiment of a method for requesting streamlets within an adaptive-rate content streaming environment in accordance with the present invention.

KUNZLER & ASSOCIATES
ATTORNEYS AT LAW
8 East Broadway, SUITE 600
SALT LAKE CITY, UTAH 84101

DETAILED DESCRIPTION OF THE INVENTION

[0035] Many of the functional units described in this specification have been labeled as modules, in order to more particularly emphasize their implementation independence. For example, a module may be implemented as a hardware circuit comprising custom VLSI circuits or gate arrays, off-the-shelf semiconductors such as logic chips, transistors, or other discrete components. A module may also be implemented in programmable hardware devices such as field programmable gate arrays, programmable array logic, programmable logic devices or the like.

[0036] Modules may also be implemented in software for execution by various types of processors. An identified module of executable code may, for instance, comprise one or more physical or logical blocks of computer instructions which may, for instance, be organized as an object, procedure, or function. Nevertheless, the executables of an identified module need not be physically located together, but may comprise disparate instructions stored in different locations which, when joined logically together, comprise the module and achieve the stated purpose for the module.

[0037] Indeed, a module of executable code may be a single instruction, or many instructions, and may even be distributed over several different code segments, among different programs, and across several memory devices. Similarly, operational data may be identified and illustrated herein within modules, and may be embodied in any suitable form and organized within any suitable type of data structure. The operational data may be collected as a single data set, or may be distributed over different locations including over different storage devices, and may exist, at least partially, merely as electronic signals on a system or network.

[0038] Reference throughout this specification to "one embodiment," "an embodiment," or similar language means that a particular feature, structure, or characteristic described in connection with the embodiment is included in at least one embodiment of the present invention. Thus, appearances of the phrases "in one embodiment," "in an

KUNZLER & ASSOCIATES
ATTORNEYS AT LAW
8 East Broadway, SUITE 600
SALT LAKE CITY, UTAH 84101

embodiment," and similar language throughout this specification may, but do not necessarily, all refer to the same embodiment.

[0039] Reference to a signal bearing medium may take any form capable of generating a signal, causing a signal to be generated, or causing execution of a program of machine-readable instructions on a digital processing apparatus. A signal bearing medium may be embodied by a transmission line, a compact disk, digital-video disk, a magnetic tape, a Bernoulli drive, a magnetic disk, a punch card, flash memory, integrated circuits, or other digital processing apparatus memory device.

[0040] Furthermore, the described features, structures, or characteristics of the invention may be combined in any suitable manner in one or more embodiments. In the following description, numerous specific details are provided, such as examples of programming, software modules, user selections, network transactions, database queries, database structures, hardware modules, hardware circuits, hardware chips, etc., to provide a thorough understanding of embodiments of the invention. One skilled in the relevant art will recognize, however, that the invention may be practiced without one or more of the specific details, or with other methods, components, materials, and so forth. In other instances, well-known structures, materials, or operations are not shown or described in detail to avoid obscuring aspects of the invention.

[0041] Figure 1 is a schematic block diagram illustrating one embodiment of a system 100 for dynamic rate shifting of streaming content in accordance with the present invention. In one embodiment, the system 100 comprises a content server 102 and an end user 104. The content server 102 and the end user station 104 may be coupled by a data communications network. The data communications network may include the Internet 106 and connections 108 to the Internet 106. Alternatively, the content server 102 and the end user 104 may be located on a common local area network, wireless area network, cellular network, virtual local area network, or the like. The end user station 104 may comprise a

KUNZLER & ASSOCIATES
ATTORNEYS AT LAW
8 East Broadway, SUITE 600
SALT LAKE CITY, UTAH 84101

personal computer (PC), an entertainment system configured to communicate over a network, or a portable electronic device configured to present content.

[0042] In the depicted embodiment, the system 100 also includes a publisher 110, and a web server 116. The publisher 110 may be a creator or distributor of content. For example, if the content to be streamed were a broadcast of a television program, the publisher 110 may be a television or cable network channel such as NBC®, or MTV®. Content may be transferred over the Internet 106 to the content server 102, where the content is received by a content module 112. The content module 112 may be configured to receive, process, and store content. In one embodiment, processed content is accessed by a client module 114 configured to play the content on the end user station 104. In a further embodiment, the client module 114 is configured to receive different portions of a content stream from a plurality of locations simultaneously. For example, the client module 114 may request and receive content from any of the plurality of web servers 116.

[0043] Figure 2a is a schematic block diagram graphically illustrating one embodiment of a content file 200. In one embodiment, the content file 200 is distributed by the publisher 110. The content file 200 may comprise a television broadcast, sports event, movie, music, concert, etc. The content file 200 may also be live or archived content. The content file 200 may comprise uncompressed video and audio, or alternatively, video or audio. Additionally, the content file 200 may be compressed. Examples of a compressed content file 200 include, but are not limited to, DivX®, Windows Media Video 9®, Quicktime 6.5 Sorenson 3®, or Quicktime 6.5/ MPEG-4® encoded content.

[0044] Figure 2b is a schematic block diagram illustrating one embodiment of a plurality of streams 202 having varying degrees of quality and bandwidth. In one embodiment, the plurality of streams 202 comprises a low quality stream 204, a medium quality stream 206, and a high quality stream 208. Each of the streams 204, 206, 208 is a copy of the content file 200 encoded and compressed to varying bit rates. For example, the low quality stream 204 may be encoded and compressed to a bit rate of 100 kilobits per

KUNZLER & ASSOCIATES
ATTORNEYS AT LAW
8 East Broadway, SUITE 600
SALT LAKE CITY, UTAH 84101

second (kbps), the medium quality stream 206 may be encoded and compressed to a bit rate of 200 kbps, and the high quality stream 208 may be encoded and compressed to 600 kbps.

[0045] Figure 2c is a schematic block diagram illustrating one embodiment of a stream 210 divided into a plurality of streamlets 212. As used herein, streamlet refers to any sized portion of the content file 200. Each streamlet 212 may comprise a portion of the content contained in stream 210, encapsulated as an independent media object. The content in a streamlet 212 may have a unique time index in relation to the beginning of the content contained in stream 210. In one embodiment, the content contained in each streamlet 212 has a duration of two seconds. For example, streamlet 0 may have a time index of 00:00 representing the beginning of content playback, and streamlet 1 may have a time index of 00:02, and so on. Alternatively, the time duration of the streamlets 212 may be any duration smaller than the entire playback duration of the content in stream 210. In a further embodiment, the streamlets 212 may be divided according to file size instead of a time index.

[0046] Figure 3 is a schematic block diagram illustrating in greater detail one embodiment of the content module 112 in accordance with the present invention. The content module 112 may comprise a stream module 302, a streamlet module 304, an encoder module 306, a streamlet database 308, and the web server 116. In one embodiment, the stream module 302 is configured to receive the content file 200 from the publisher 110 and generate the plurality of streams 202 of varying qualities. The original content file 200 from the publisher may be digital in form and may comprise content having a high bit rate such as, for example, 2mbps. The content may be transferred from the publisher 110 to the content module 112 over the Internet 106. Such transfers of data are well known in the art and do not require further discussion herein. Alternatively, the content may comprise a captured broadcast.

[0047] In the depicted embodiment, the plurality of streams 202 may comprise the low quality stream 204, the medium quality stream 206, and the high quality stream 208. Alternatively, the plurality of streams 202 may comprise any number of streams deemed

KUNZLER & ASSOCIATES
ATTORNEYS AT LAW
8 East Broadway, SUITE 600
SALT LAKE CITY, UTAH 84101

necessary to accommodate end user bandwidth. The streamlet module 304 may be configured to receive the plurality of streams 202 from the stream module and generate a plurality of streams 312, each stream comprising a plurality of streamlets 212. As described with reference to Figure 2c, each streamlet 212 may comprise a pre-defined portion of the stream. The encoder module 306 is configured to encode each streamlet from the plurality of streams 312 and store the streamlets in the streamlet database 308. The encoding module 306 may utilize encoding schemes such as DivX®, Windows Media Video 9®, Quicktime 6.5 Sorenson 3®, or Quicktime 6.5/ MPEG-4®. Alternatively, a custom encoding scheme may be employed.

[0048] The content module 112 may also include a metadata module 312 and a metadata database 314. In one embodiment, metadata comprises static searchable content information. For example, metadata includes, but is not limited to, air date of the content, title, actresses, actors, length, and episode name. Metadata is generated by the publisher 110, and may be configured to define an end user environment. In one embodiment, the publisher 100 may define an end user navigational environment for the content including menus, thumbnails, sidebars, advertising, etc. Additionally, the publisher 110 may define functions such as fast forward, rewind, pause, and play that may be used with the content file 200. The metadata module 312 is configured to receive the metadata from the publisher 110 and store the metadata in the metadata database 314. In a further embodiment, the metadata module 312 is configured to interface with the client module 114, allowing the client module 114 to search for content based upon at least one of a plurality of metadata criteria. Additionally, metadata may be generated by the content module 112 through automated process(es) or manual definition.

[0049] Once the streamlets 212 have been received and processed, the client module 114 may request streamlets 212 using HTTP from the web server 116. Such use of client side initiated requests requires no additional configuration of firewalls. Additionally, since the client module 114 initiates the request, the web server 116 is only required to retrieve and

KUNZLER & ASSOCIATES
ATTORNEYS AT LAW
8 East Broadway, SUITE 600
SALT LAKE CITY, UTAH 84101

serve the requested streamlet. In a further embodiment, the client module 114 may be configured to retrieve streamlets 212 from a plurality of web servers 310. Each web server 116 may be located in various locations across the Internet 106. The streamlets 212 are essentially static files. As such, no specialized media server or server-side intelligence is required for a client module 114 to retrieve streamlets 212. Streamlets 212 may be served by the web server 116 or cached by cache servers of Internet Service Providers (ISPs), or any other network infrastructure operators, and served by the cache server. Use of cache servers is well known to those skilled in the art, and will not be discussed further herein. Thus, a highly scalable solution is provided that is not hindered by massive amounts of client module 114 requests to the web server 116 at any specific location.

[0050] Figure 4 is a schematic block diagram graphically illustrating one embodiment of a client module 114 in accordance with the present invention. The client module 114 may comprise an agent controller module 402, a streamlet cache module 404, and a network controller module 406. In one embodiment, the agent controller module 402 is configured to interface with a viewer 408, and transmit streamlets 212 to the viewer 408. In a further embodiment, the client module 114 may comprise a plurality of agent controller modules 402. Each agent controller module 402 may be configured to interface with one viewer 408. Alternatively, the agent controller module 402 may be configured to interface with a plurality of viewers 408. The viewer 408 may be a media player (not shown) operating on a PC or handheld electronic device.

[0051] The agent controller module 402 is configured to select a quality level of streamlets to transmit to the viewer 408. The agent controller module 402 requests lower or higher quality streams based upon continuous observation of time intervals between successive receive times of each requested streamlet. The method of requesting higher or lower quality streams will be discussed in greater detail below with reference to Figure 7.

[0052] The agent controller module 402 may be configured to receive user commands from the viewer 408. Such commands may include play, fast forward, rewind,

KUNZLER & ASSOCIATES
ATTORNEYS AT LAW
8 East Broadway, SUITE 600
SALT LAKE CITY, UTAH 84101

pause, and stop. In one embodiment, the agent controller module 402 requests streamlets 212 from the streamlet cache module 404 and arranges the received streamlets 212 in a staging module 409. The staging module 409 may be configured to arrange the streamlets 212 in order of ascending playback time. In the depicted embodiment, the streamlets 212 are numbered 0, 1, 2, 3, 4, etc. However, each streamlet 212 may be identified with a unique filename.

[0053] Additionally, the agent controller module 402 may be configured to anticipate streamlet 212 requests and pre-request streamlets 212. By pre-requesting streamlets 212, the user may fast-forward, skip randomly, or rewind through the content and experience no buffering delay. In a further embodiment, the agent controller module 402 may request the streamlets 212 that correspond to time index intervals of 30 seconds within the total play time of the content. Alternatively, the agent controller module 402 may request streamlets at any interval less than the length of the time index. This enables a "fast-start" capability with no buffering wait when starting or fast-forwarding through content file 200. In a further embodiment, the agent controller module 402 may be configured to pre-request streamlets 212 corresponding to specified index points within the content or within other content in anticipation of the end user 104 selecting new content to view.

[0054] In one embodiment, the streamlet cache module 404 is configured to receive streamlet 212 requests from the agent controller module 402. Upon receiving a request, the streamlet cache module 404 first checks a streamlet cache 410 to verify if the streamlet 212 is present. In a further embodiment, the streamlet cache module 404 handles streamlet 212 requests from a plurality of agent controller modules 402. Alternatively, a streamlet cache module 404 may be provided for each agent controller module 402. If the requested streamlet 212 is not present in the streamlet cache 410, the request is passed to the network controller module 406. In order to enable fast forward and rewind capabilities, the streamlet cache module 404 is configured to store the plurality of streamlets 212 in the streamlet cache

KUNZLER & ASSOCIATES
ATTORNEYS AT LAW
8 East Broadway, SUITE 600
SALT LAKE CITY, UTAH 84101

410 for a specified time period after the streamlet 212 has been viewed. However, once the streamlets 212 have been deleted, they may be requested again from the web server 116.

[0055] The network controller module 406 may be configured to receive streamlet requests from the streamlet cache module 404 and open a connection to the web server 116 or other remote streamlet 212 database (not shown). In one embodiment, the network controller module 406 opens a TCP/IP connection to the web server 116 and generates a standard HTTP GET request for the requested streamlet 212. Upon receiving the requested streamlet 212, the network controller module 406 passes the streamlet 212 to the streamlet cache module 404 where it is stored in the streamlet cache 410. In a further embodiment, the network controller module 406 is configured to process and request a plurality of streamlets 212 simultaneously. The network controller module 406 may also be configured to request a plurality of streamlets, where each streamlet 212 is subsequently requested in multiple parts.

[0056] In a further embodiment, streamlet requests may comprise requesting pieces of any streamlet file. Splitting the streamlet 212 into smaller pieces or portions beneficially allows for an increased efficiency potential, and also eliminates problems associated with multiple full-streamlet requests sharing the bandwidth at any given moment. This is achieved by using parallel TCP/IP connections for pieces of the streamlets 212. Consequently, efficiency and network loss problems are overcome, and the streamlets arrive with more useful and predictable timing.

[0057] In one embodiment, the client module 114 is configured to use multiple TCP connections between the client module 114 and the web server 116 or web cache. The intervention of a cache may be transparent to the client or configured by the client as a forward cache. By requesting more than one streamlet 212 at a time in a manner referred to as "parallel retrieval," or more than one part of a streamlet 212 at a time, efficiency is raised significantly and latency is virtually eliminated. In a further embodiment, the client module allows a maximum of three outstanding streamlet 212 requests. The client module 114 may maintain additional open TCP connections as spares to be available should another

KUNZLER & ASSOCIATES
ATTORNEYS AT LAW
8 East Broadway, SUITE 600
SALT LAKE CITY, UTAH 84101

connection fail. Streamlet 212 requests are rotated among all open connections to keep the TCP flow logic for any particular connection from falling into a slow-start or close mode. If the network controller module 406 has requested a streamlet 212 in multiple parts, with each part requested on mutually independent TCP/IP connections, the network controller module 406 reassembles the parts to present a complete streamlet 212 for use by all other components of the client module 114.

[0058] When a TCP connection fails completely, a new request may be sent on a different connection for the same streamlet 212. In a further embodiment, if a request is not being satisfied in a timely manner, a redundant request may be sent on a different connection for the same streamlet 212. If the first streamlet request's response arrives before the redundant request response, the redundant request can be aborted. If the redundant request response arrives before the first request response, the first request may be aborted.

[0059] Several streamlet 212 requests may be sent on a single TCP connection, and the responses are caused to flow back in matching order along the same connection. This eliminates all but the first request latency. Because multiple responses are always being transmitted, the processing latency of each new streamlet 212 response after the first is not a factor in performance. This technique is known in the industry as "pipelining." Pipelining offers efficiency in request-response processing by eliminating most of the effects of request latency. However, pipelining has serious vulnerabilities. Transmission delays affect all of the responses. If the single TCP connection fails, *all* of the outstanding requests and responses are lost. Pipelining causes a serial dependency between the requests.

[0060] Multiple TCP connections may be opened between the client module 114 and the web server 116 to achieve the latency-reduction efficiency benefits of pipelining while maintaining the independence of each streamlet 212 request. Several streamlet 212 requests may be sent concurrently, with each request being sent on a mutually distinct TCP connection. This technique is labeled "virtual pipelining" and is an innovation of the present invention. Multiple responses may be in transit concurrently, assuring that communication

KUNZLER & ASSOCIATES
ATTORNEYS AT LAW
8 East Broadway, SUITE 600
SALT LAKE CITY, UTAH 84101

bandwidth between the client module 114 and the web server 116 is always being utilized. Virtual pipelining eliminates the vulnerabilities of traditional pipelining. A delay in or complete failure of one response does not affect the transmission of other responses because each response occupies an independent TCP connection. Any transmission bandwidth not in use by one of multiple responses (whether due to delays or TCP connection failure) may be utilized by other outstanding responses.

[0061] A single streamlet 212 request may be issued for an entire streamlet 212, or multiple requests may be issued, each for a different part or portion of the streamlet. If the streamlet is requested in several parts, the parts may be recombined by the client module 114 streamlet.

[0062] In order to maintain a proper balance between maximized bandwidth utilization and response time, the issuance of new streamlet requests must be timed such that the web server 116 does not transmit the response before the client module 114 has fully received a response to one of the previously outstanding streamlet requests. For example, if three streamlet 212 requests are outstanding, the client module 114 should issue the next request slightly before one of the three responses is fully received and "out of the pipe." In other words, request timing is adjusted to keep three responses in transit. Sharing of bandwidth among four responses diminishes the net response time of the other three responses. The timing adjustment may be calculated dynamically by observation, and the request timing adjusted accordingly to maintain the proper balance of efficiency and response times.

[0063] The schematic flow chart diagrams that follow are generally set forth as logical flow chart diagrams. As such, the depicted order and labeled steps are indicative of one embodiment of the presented method. Other steps and methods may be conceived that are equivalent in function, logic, or effect to one or more steps, or portions thereof, of the illustrated method. Additionally, the format and symbols employed are provided to explain the logical steps of the method and are understood not to limit the scope of the method.

KUNZLER & ASSOCIATES
ATTORNEYS AT LAW
8 East Broadway, SUITE 600
SALT LAKE CITY, UTAH 84101

Although various arrow types and line types may be employed in the flow chart diagrams, they are understood not to limit the scope of the corresponding method. Indeed, some arrows or other connectors may be used to indicate only the logical flow of the method. For instance, an arrow may indicate a waiting or monitoring period of unspecified duration between enumerated steps of the depicted method. Additionally, the order in which a particular method occurs may or may not strictly adhere to the order of the corresponding steps shown.

[0064] Figure 5 is a schematic flow chart diagram illustrating one embodiment of a method 500 for processing content in accordance with the present invention. In one embodiment the method 500 starts 502, and the content module 112 receives 504 content from the publisher 110. Receiving content 504 may comprise receiving 504 a digital copy of the content file 200, or digitizing a physical copy of the content file 200. Alternatively, receiving 504 content may comprise capturing a radio or television broadcast. Once received 504, the stream module 302 generates 506 a plurality of streams 202, each stream 202 having a different quality. The quality may be predefined, or automatically set according to end user bandwidth, or in response to pre-designated publisher guidelines

[0065] The streamlet module 304 receives the streams 202 and generates 508 a plurality of streamlets 212. In one embodiment, generating 508 streamlets comprises dividing the stream 202 into a plurality of two second streamlets 212. Alternatively, the streamlets may have any length less than or equal to the length of the stream 202. The encoder module 306 then encodes 510 the streamlets according to a compression algorithm. In a further embodiment, the algorithm comprises a proprietary codec such as WMV9®. The encoder module 306 then stores 512 the encoded streamlets in the streamlet database 308. Once stored 512, the web server 116 may then serve 514 the streamlets. In one embodiment, serving 514 the streamlets comprises receiving streamlet requests from the client module 114, retrieving the requested streamlet from the streamlet database 308, and subsequently transmitting the streamlet to the client module 114. The method 500 then ends 516.

KUNZLER & ASSOCIATES
ATTORNEYS AT LAW
8 East Broadway, SUITE 600
SALT LAKE CITY, UTAH 84101

[0066] Figure 6 is a schematic flow chart diagram illustrating one embodiment of a method 600 for viewing a plurality of streamlets in accordance with the present invention. The method 600 starts and an agent control module 402 is provided 604 and associated with a viewer 408 and provided with a staging module 409. The agent controller module 402 then requests 606 a streamlet from the streamlet cache module 404. Alternatively, the agent controller module 402 may simultaneously request 606 a plurality of streamlets from the streamlet cache module 404. If the streamlet is stored 608 locally in the streamlet cache 410, the streamlet cache module 404 retrieves 610 the streamlet and sends the streamlet to the agent controller module 402. Upon retrieving 610 or receiving a streamlet, the agent controller module 402 makes 611 a determination of whether or not to shift to a higher or lower quality stream 202. This determination will be described below in greater detail with reference to Figure 7.

[0067] In one embodiment, the staging module 409 then arranges 612 the streamlets into the proper order, and the agent controller module 402 delivers 614 the streamlets to the viewer 408. In a further embodiment, delivering 614 streamlets to the end user comprises playing video and or audio streamlets on the viewer 408. If the streamlets are not stored 608 locally, the streamlet request is passed to the network controller module 406. The network controller module 406 then requests 616 the streamlet from the web server 116. Once the streamlet is received, the network controller module 406 passes the streamlet to the streamlet cache module 404. The streamlet cache module 404 archives 618 the streamlet. Alternatively, the streamlet cache module 404 then archives 618 the streamlet and passes the streamlet to the agent controller module 402, and the method 600 then continues from operation 610 as described above.

[0068] Referring now to Figure 7, shown therein is a schematic flow chart diagram illustrating one embodiment of a method 700 for requesting streamlets within a adaptive-rate shifting content streaming environment in accordance with the present invention. The method 700 may be used in one embodiment as the operation 611 of Figure 6. The method

KUNZLER & ASSOCIATES
ATTORNEYS AT LAW
8 East Broadway, SUITE 600
SALT LAKE CITY, UTAH 84101

700 starts and the agent controller module 402 receives 704 a streamlet as described above with reference to Figure 6. The agent controller module 402 then monitors 706 the receive time of the requested streamlet. In one embodiment, the agent controller module 402 monitors the time intervals Δ between successive receive times for each streamlet response. Ordering of the responses in relation to the order of their corresponding requests is not relevant.

[0069] Because network behavioral characteristics fluctuate, sometimes quite suddenly, any given Δ may vary substantially from another. In order to compensate for this fluctuation, the agent controller module 402 calculates 708 a performance ratio r across a window of n samples for streamlets of playback length S. In one embodiment, the performance ratio r is calculated using the equation

$$r = S \frac{n}{\sum_{i=1}^{n} \Delta_i}.$$

[0070] Due to multiple simultaneous streamlet processing, and in order to better judge the central tendency of the performance ratio r, the agent control module 402 may calculate a geometric mean, or alternatively an equivalent averaging algorithm, across a window of size m, and obtain a performance factor φ:

$$\varphi_{current} = \left(\prod_{j=1}^{m} r_j \right)^{\frac{1}{m}}.$$

[0071] The policy determination about whether or not to upshift 710 playback quality begins by comparing $\varphi_{current}$ with a trigger threshold Θ_{up}. If $\varphi_{current} \geq \Theta_{up}$, then an up shift to the next higher quality stream may be considered 716. In one embodiment, the trigger threshold Θ_{up} is determined by a combination of factors relating to the current read ahead margin (i.e. the amount of contiguously available streamlets that have been sequentially arranged by the staging module 409 for presentation at the current playback time index), and

KUNZLER & ASSOCIATES
ATTORNEYS AT LAW
8 East Broadway, SUITE 600
SALT LAKE CITY, UTAH 84101

a minimum safety margin. In one embodiment, the minimum safety margin may be 24 seconds. The smaller the read ahead margin, the larger Θ_{up} is to discourage upshifting until a larger read ahead margin may be established to withstand network disruptions. If the agent controller module 402 is able to sustain 716 upshift quality, then the agent controller module 402 will upshift 717 the quality and subsequently request higher quality streams. The determination of whether use of the higher quality stream is sustainable 716 is made by comparing an estimate of the higher quality stream's performance factor, φ_{higher}, with Θ_{up}. If $\varphi_{higher} \geq \Theta_{up}$ then use of the higher quality stream is considered sustainable. If the decision of whether or not the higher stream rate is sustainable 716 is "no," the agent control module 402 will not attempt to upshift 717 stream quality. If the end of the stream has been reached 714, the method 618 ends 716.

[0072] If the decision on whether or not to attempt upshift 710 is "no", a decision about whether or not to downshift 712 is made. In one embodiment, a trigger threshold Θ_{down} is defined in a manner analogous to Θ_{up}. If $\varphi_{current} > \Theta_{down}$ then the stream quality may be adequate, and the agent controller module 402 does not downshift 718 stream quality. However, if $\varphi_{current} \leq \Theta_{down}$, the agent controller module 402 does downshift 718 the stream quality. If the end of the stream has not been reached 714, the agent controller module 402 begins to request and receive 704 lower quality streamlets and the method 618 starts again. Of course, the above described equations and algorithms are illustrative only, and may be replaced by alternative streamlet monitoring solutions.

[0073] The present invention may be embodied in other specific forms without departing from its spirit or essential characteristics. The described embodiments are to be considered in all respects only as illustrative and not restrictive. The scope of the invention is, therefore, indicated by the appended claims rather than by the foregoing description. All changes which come within the meaning and range of equivalency of the claims are to be embraced within their scope.

[0074] What is claimed is:

KUNZLER & ASSOCIATES
ATTORNEYS AT LAW
8 East Broadway, SUITE 600
SALT LAKE CITY, UTAH 84101

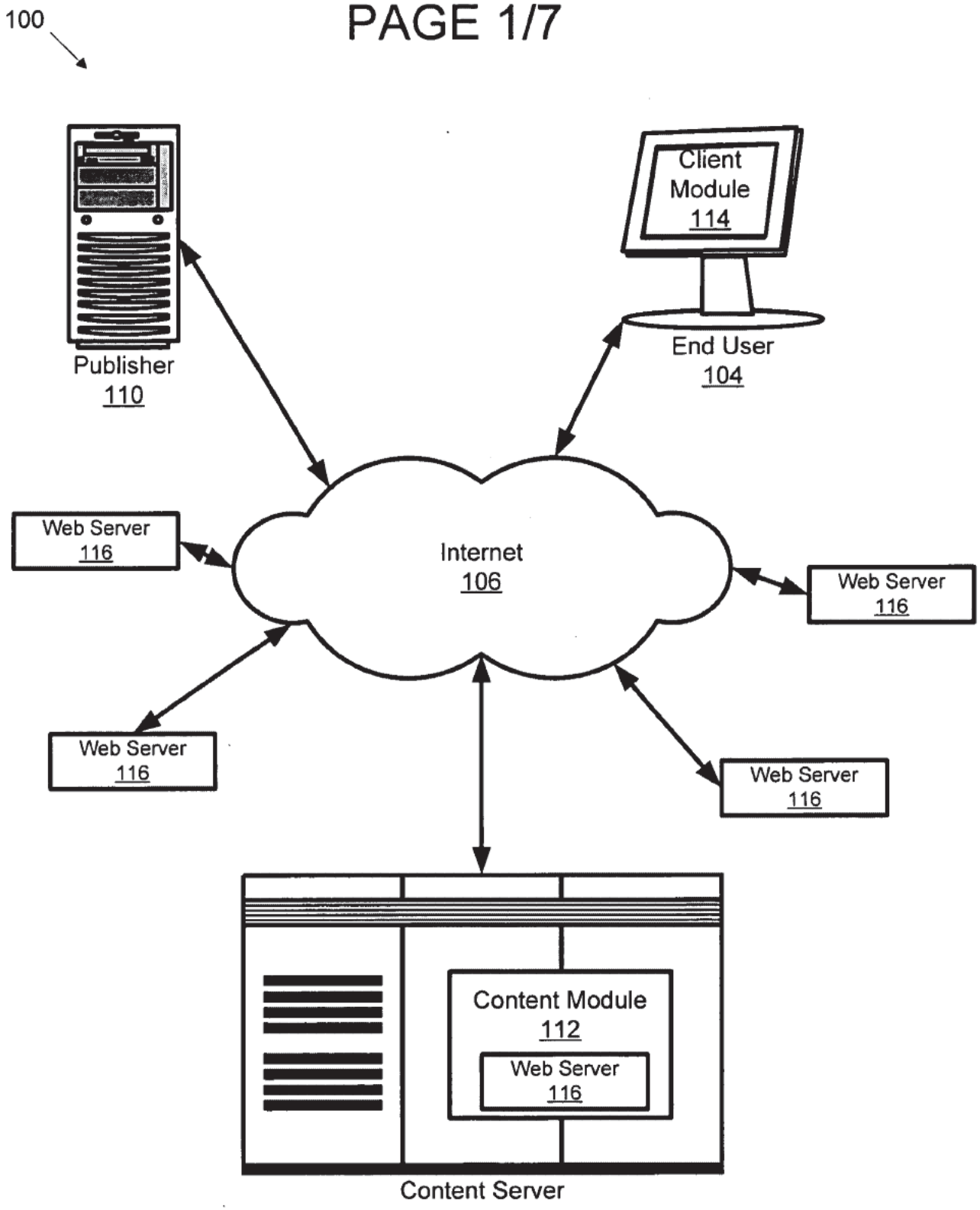

100
Publisher
110
Client
Module
114
End User
104
Internet
106
Web Server
116
Web Server
116
Web Server
116
Web Server
116
Content Module
112
Web Server
116
Content Server

FIG. 1

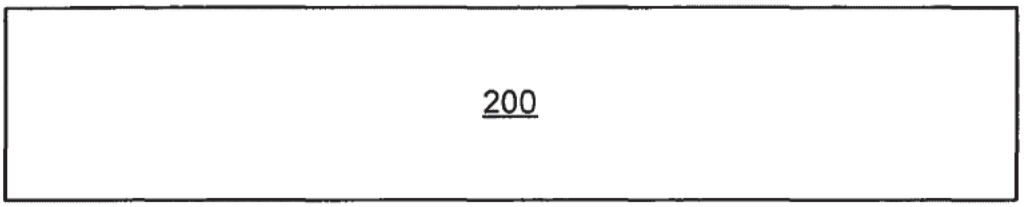


FIG. 2a

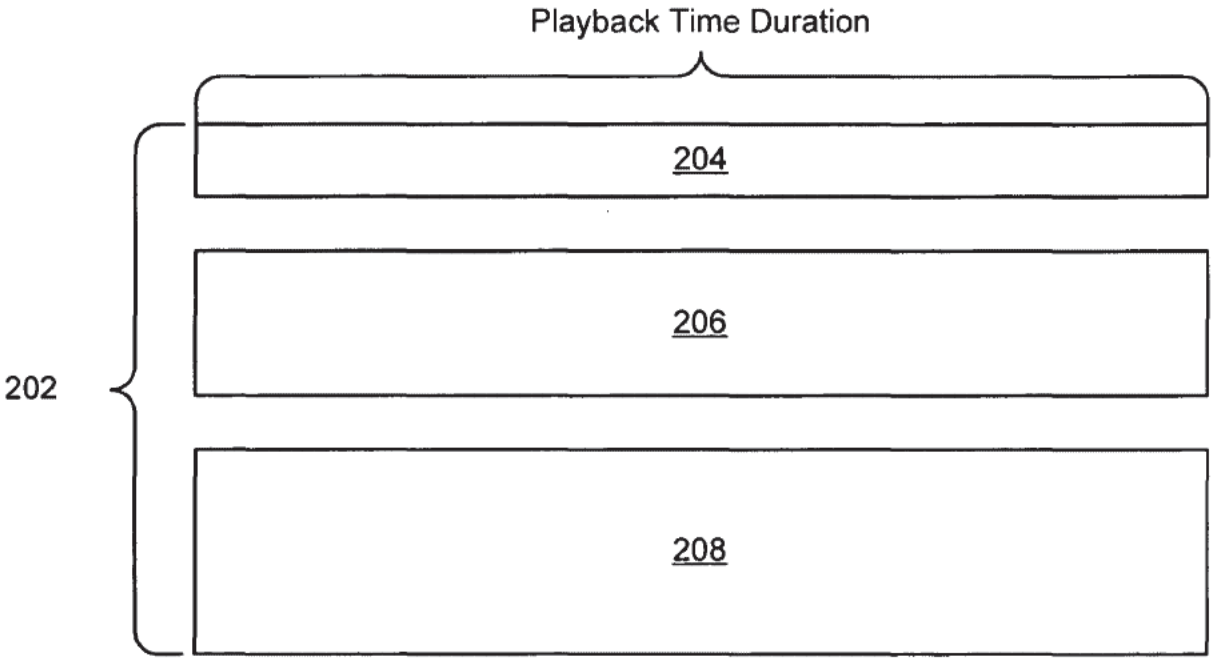


FIG. 2b

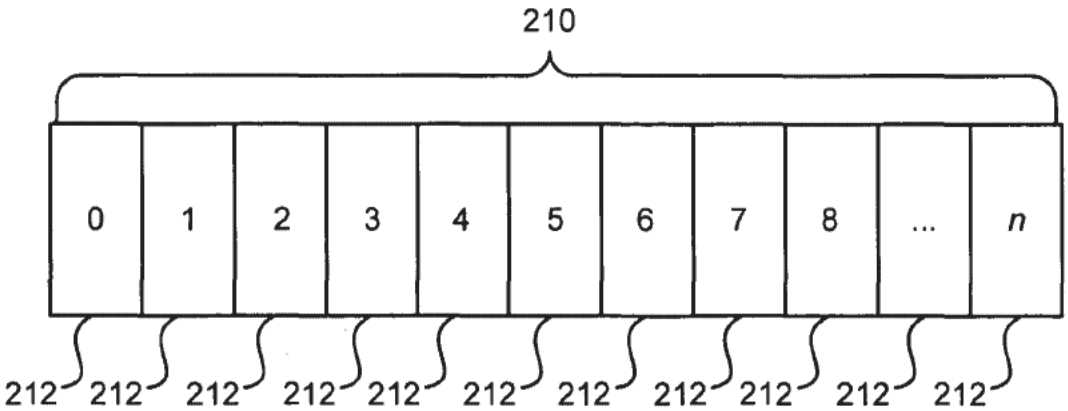


FIG. 2c

Ex. 1018-026

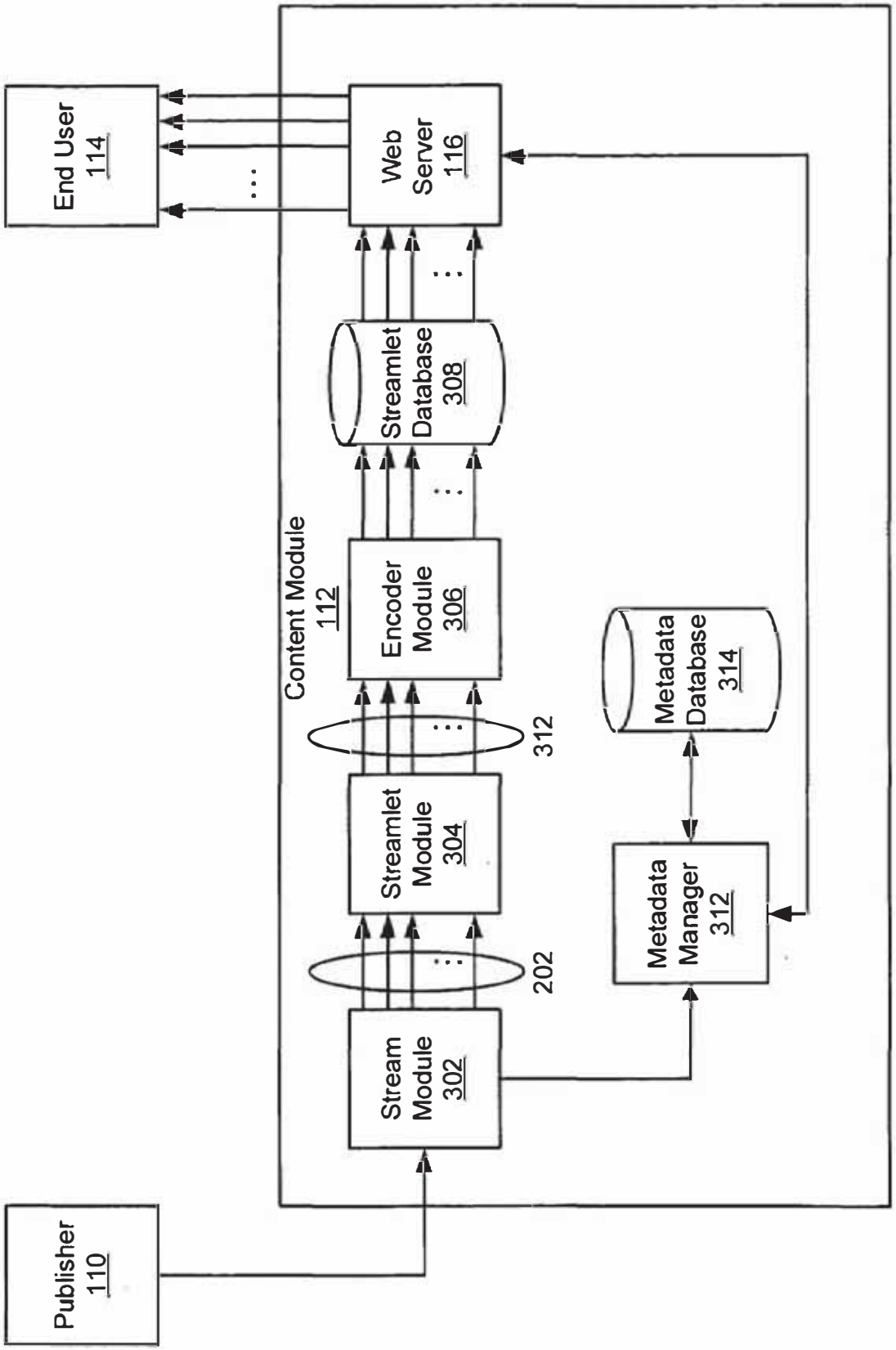


FIG. 3

Ex. 1018-027

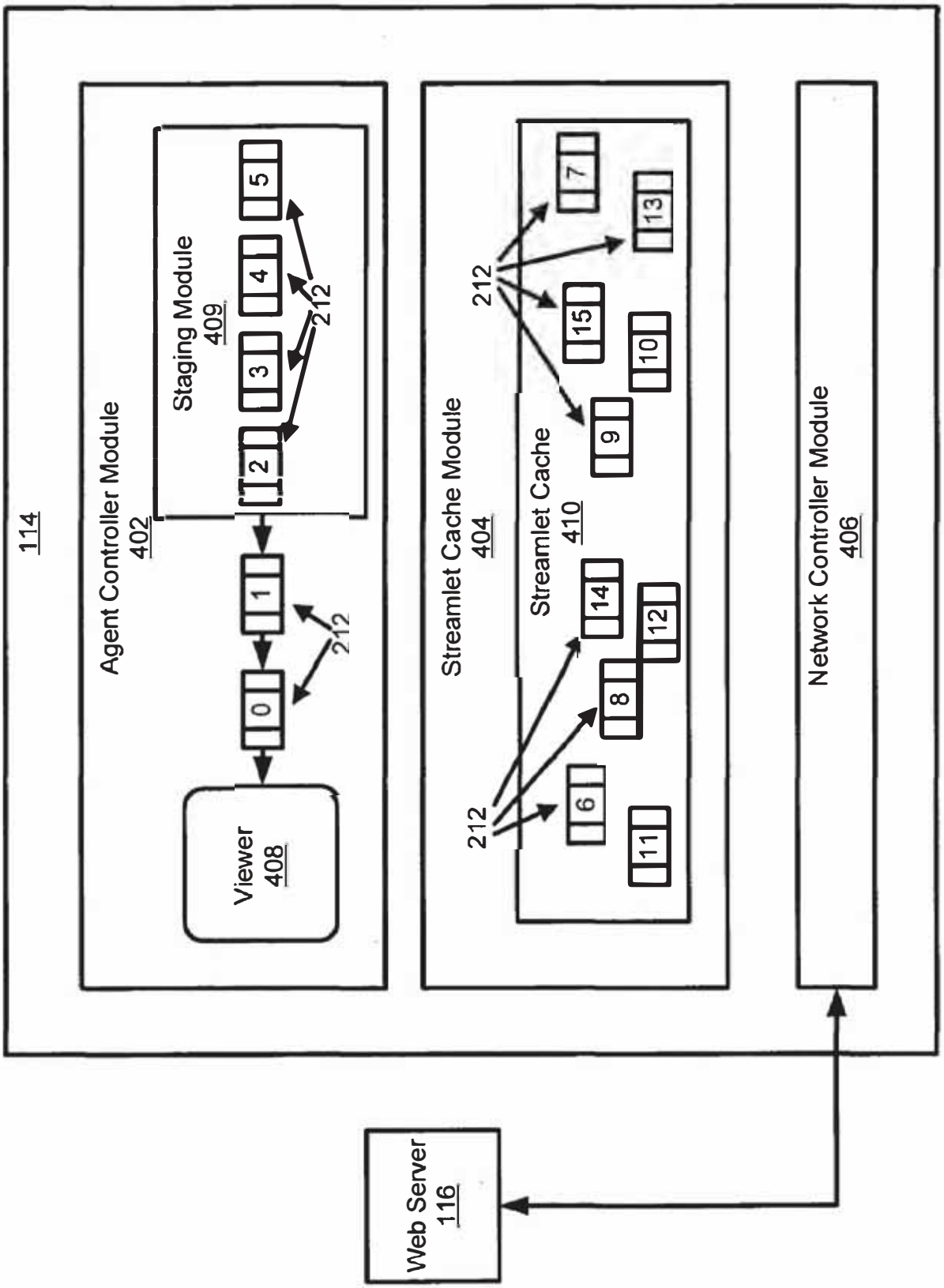


FIG. 4

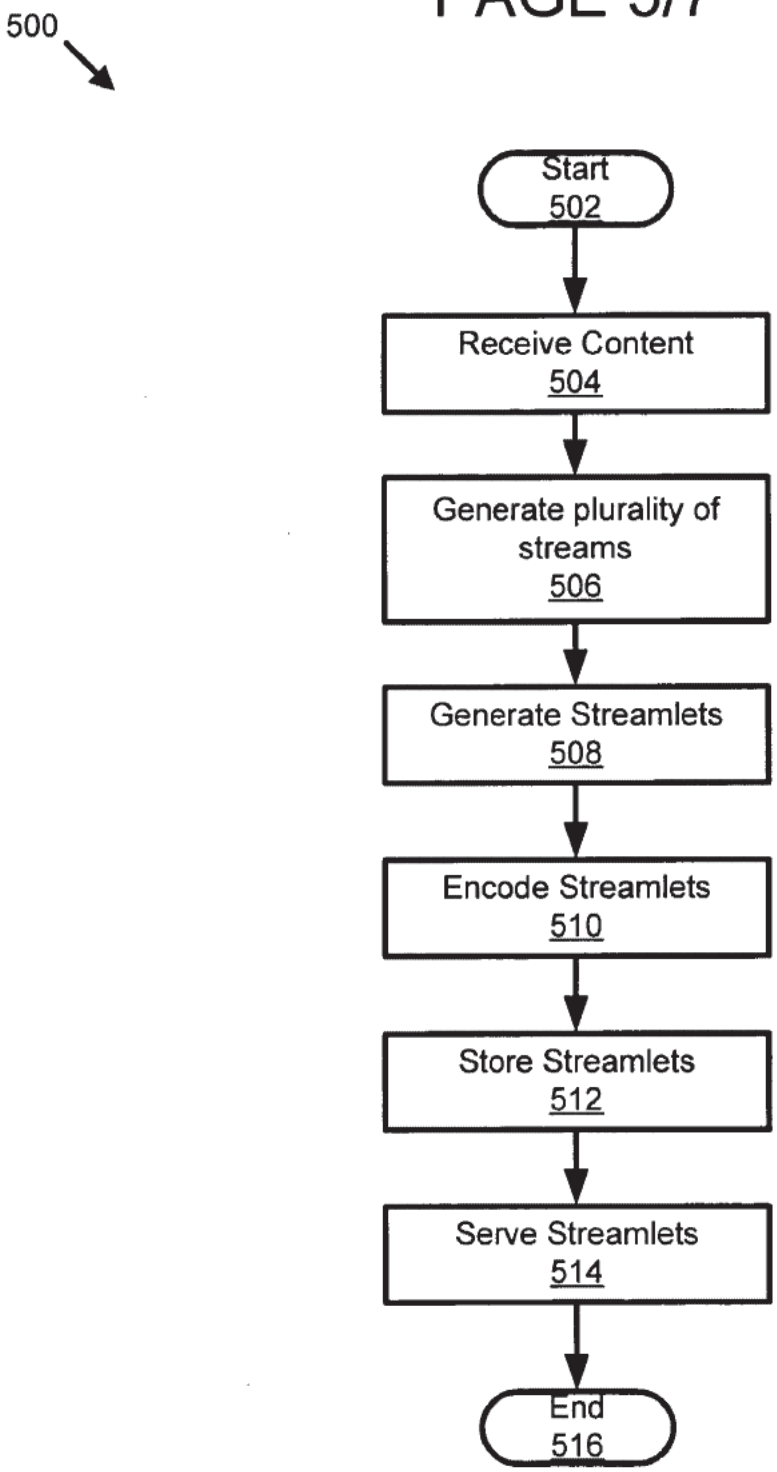

500
Start
502
Receive Content
504
Generate plurality of streams
506
Generate Streamlets
508
Encode Streamlets
510
Store Streamlets
512
Serve Streamlets
514
End
516

FIG. 5

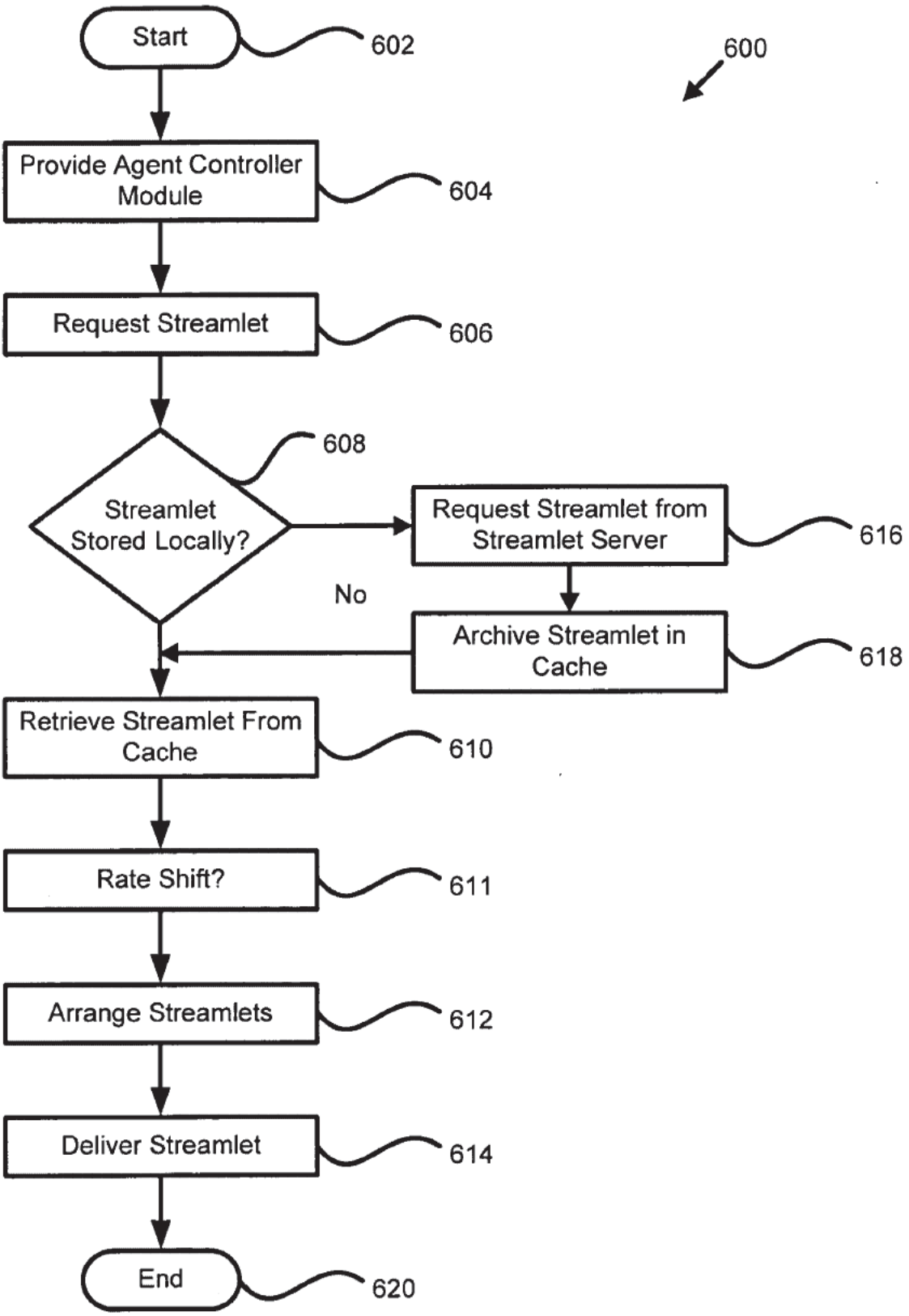

Start
602
600
Provide Agent Controller Module
604
Request Streamlet
606
608
Streamlet Stored Locally?
Request Streamlet from Streamlet Server
616
No
Archive Streamlet in Cache
618
Retrieve Streamlet From Cache
610
Rate Shift?
611
Arrange Streamlets
612
Deliver Streamlet
614
End
620

FIG. 6

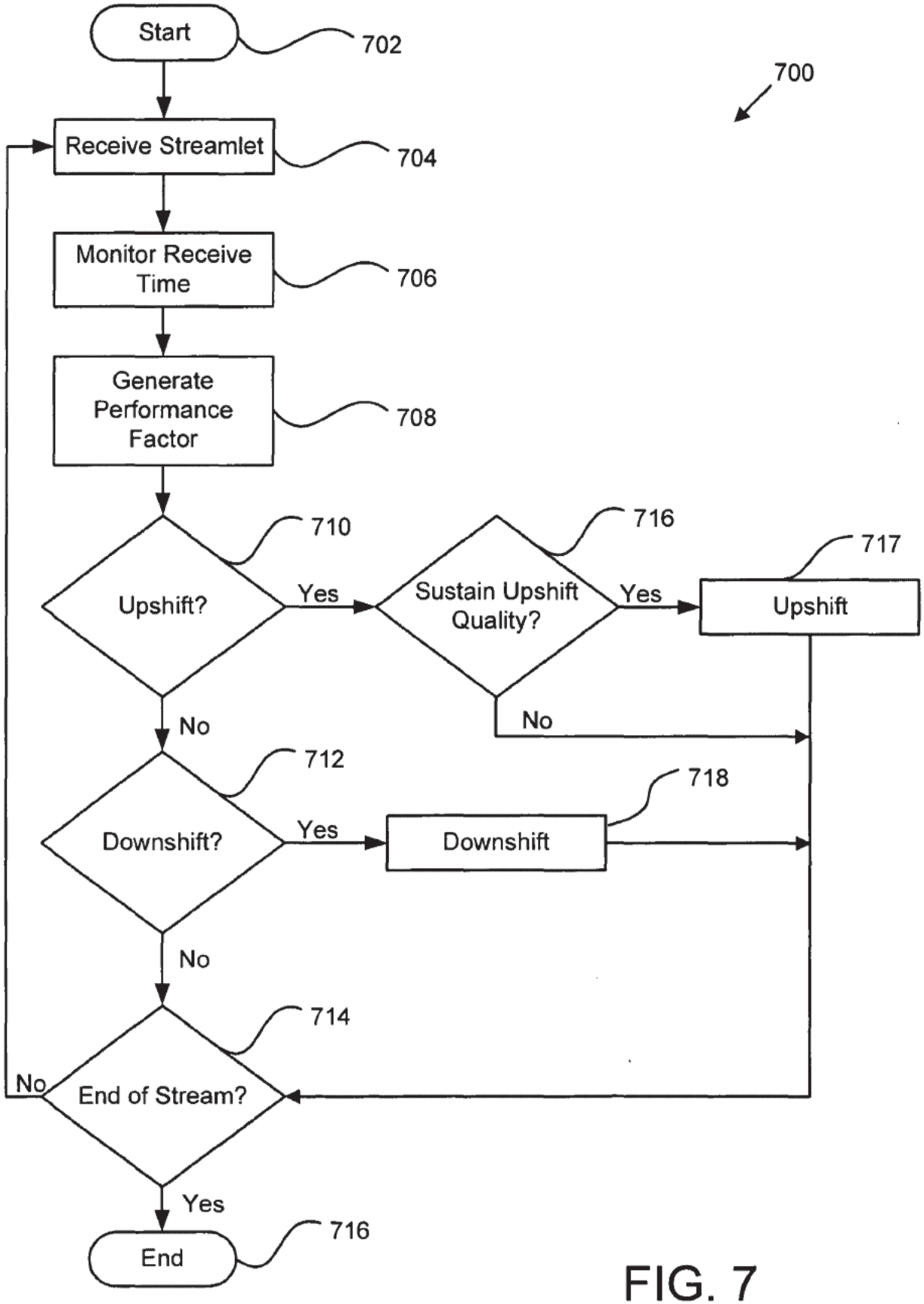

Start
702
700
Receive Streamlet
704
Monitor Receive Time
706
Generate Performance Factor
708
710
Upshift?
Yes
716
Sustain Upshift Quality?
Yes
717
Upshift
No
No
712
Downshift?
Yes
718
Downshift
No
714
No
End of Stream?
Yes
716
End

FIG. 7

APPARATUS, SYSTEM, AND METHOD FOR ADAPTIVE-RATE SHIFTING OF STREAMING CONTENT

ABSTRACT OF THE DISCLOSURE

An apparatus for adaptive-rate shifting of streaming content includes an agent controller module configured to simultaneously request at least portions of a plurality of streamlets. The agent controller module is further configured to continuously monitor streamlet requests and subsequent responses, and accordingly request higher or lower quality streamlets. A staging module is configured to stage the streamlets and arrange the streamlets for playback on a content player. A system includes a data communications network, a content server coupled to the data communications network and having a content module configured to process content and generate a plurality of high and low quality streams, and the apparatus. A method includes simultaneously requesting at least portions of a plurality of streamlets, continuously monitoring streamlet requests and subsequent responses, and accordingly requesting higher or lower quality streamlets, and staging the streamlets and arranging the streamlets for playback on a content player.

KUNZLER & ASSOCIATES
ATTORNEYS AT LAW
8 East Broadway, SUITE 600
SALT LAKE CITY, UTAH 84101

Docket No.: 2082.2.1

DECLARATION AND POWER OF ATTORNEY FOR PATENT APPLICATION

As a below named inventor, I hereby declare that:

My residence and citizenship are as stated below next to my name;

I believe I am the original, first and joint inventor of the subject matter which is claimed and for which a patent is sought on the invention entitled

APPARATUS, SYSTEM, AND METHOD FOR ADAPTIVE-RATE SHIFTING OF STREAMING CONTENT

the specification of which (check one)

__X__ is attached hereto.
_____ was filed on _____
as Application Serial No. _____
and was amended on _____ (if applicable).

I hereby state that I have reviewed and understand the contents of the above identified specification, including the claims, as amended by any amendment referred to above.

I acknowledge the duty to disclose information which is material to patentability as defined in Title 37, code of Federal Regulations, Section 1.56.

I hereby claim foreign priority benefits under Title 35, United States Code, Section 119 of any foreign application(s) for patent or inventor's certificate listed below and have also identified below any foreign application for patent or inventor's certificate having a filing date before that of the application on which priority is claimed:

Prior Foreign Application(s)			Priority Claimed
none			_____Yes _____No
(Number)	(Country)	(Day/Month/Year filed)	

I hereby claim the benefit under Title 35, United States Code, Section 120 of any United States application(s) listed below and, insofar as the subject matter of each of the claims of this application is not disclosed in the prior United States application in the manner provided by the first paragraph of Title 35, United States Code, Section 112, I acknowledge the duty to disclose information which is material to patentability as defined in Title 37, Code of Federal Regulations, Section 1.56, which occurred between the filing date of the prior application and the national or PCT international filing date of this application:

60/566,831	04/30/2004	Pending
(Application Serial No.)	(Filing Date)	(Status) (patented, pending, abandoned)

I hereby declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true; and further that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under Section 1001 of Title 18 of the United States Code and that such willful false statements may jeopardize the validity of the application or any patent issued thereon.

POWER OF ATTORNEY: As named inventor, I hereby appoint the following attorney(s) and/or agent(s) to prosecute this application and transact all business in the Patent and Trademark Office connected therewith. **Customer No. 36491**

Brian C. Kunzler: 38,527
David J. McKenzie: 46,919

Send correspondence to: **Customer No. 36491**

Full name of sole or first joint-inventor: **R. Drew Major**

Inventor's signature: R. Drew Major Date: 4/28/05

Residence: **863 West 600 South, Orem, Utah 84058**

Citizenship: **United States of America**

Post Office Address: Same

Full name of second joint-inventor: **Mark B. Hurst**

Inventor's signature: Date:

Residence: **10639 North Avalon, Cedar Hills, Utah 84062**

Citizenship: **United States of America**

Post Office Address: Same

F:\FILES\Brian\!Client Files\1200 SanJose\1259\1259-Dec-Power substitute.wpd

POWER OF ATTORNEY: As named inventor, I hereby appoint the following attorney(s) and/or agent(s) to prosecute this application and transact all business in the Patent and Trademark Office connected therewith. **Customer No. 36491**

Brian C. Kunzler: 38,527
David J. McKenzie: 46,919

Send correspondence to: **Customer No. 36491**

Full name of sole or first joint-inventor: **R. Drew Major**

Inventor's signature: Date:

Residence: **863 West 600 South, Orem, Utah 84058**

Citizenship: **United States of America**

Post Office Address: Same

Full name of second joint-inventor: **Mark B. Hurst**

Inventor's signature: Date: 28-APR-2005

Residence: **10639 North Avalon, Cedar Hills, Utah 84062**

Citizenship: **United States of America**

Post Office Address: Same

F:\FILES\Brian\Client Files\1200 SanJose\1259\1259-Dec-Power substitute.wpd

PATENT APPLICATION SERIAL NO. _____

U.S. DEPARTMENT OF COMMERCE
PATENT AND TRADEMARK OFFICE
FEE RECORD SHEET

05/02/2005 MWOLDGE1 00000052 11116783

01 FC:2011	150.00	OP
02 FC:2111	250.00	OP
03 FC:2311	100.00	OP
04 FC:2201	200.00	OP
05 FC:2202	350.00	OP

PTO-1556
(5/87)

*U.S. Government Printing Office: 2002 — 489-267/69033

BEST AVAILABLE COPY

20427 U.S. PTO 042805

Please type a plus sign (+) inside this box → [+]

PTO/SB/05 (12/97)
Approved for use through 9/30/00. OMB 0651-0032
Patent and Trademark Office: U.S. DEPARTMENT OF COMMERCE

Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.

UTILITY PATENT APPLICATION TRANSMITTAL

(Only for new nonprovisional applications under 37 CFR 1.53(b))

Attorney Docket No.	2082.2.1	*Total Pages*	41
First Inventor : R. Drew Major			
Title: APPARATUS, SYSTEM, AND METHOD FOR ADAPTIVE-RATE SHIFTING OF STREAMING CONTENT			
Express Mail Label No.	EV577684659US		

112955 U.S. PTO 11/116783 042805

APPLICATION ELEMENTS

See MPEP chapter 600 concerning utility patent application contents.

ADDRESS TO: **Commissioner for Patents**
P.O. Box 1450
Alexandria, VA 22313-1450

1. [X] Fee Transmittal Form
   (Submit an original, and a duplicate for fee processing)
2. [X] Applicant Claims Small Entity Status
3. [X] Specification *(Total Pages)* 31
   (Preferred arrangement set forth below)
   - Descriptive title of the Invention
   - Cross References to Related Application
   - Statement Regarding Fed sponsored R & D
   - Reference to Microfiche Appendix
   - Background of the Invention
   - Brief Summary of the Invention
   - Brief Description of the Drawings *(if filed)*
   - Detailed Description
   - Claim(s)
   - Abstract of the Disclosure
4. [X] Drawing(s) *(35 USC 113)* *(Total Pages)* 7
5. Oath or Declaration (Total Pages) 3
   a. [X] Newly executed (original or copy)
   b. [ ] Copy from a prior application (37 CFR 1.63(d))
   (for continuation /divisional with Box 17 completed)
   [Note Box 5 below]
   i. [ ] DELETION OF INVENTOR(S)
   Signed statement attached deleting inventor(s) named in prior application, see 37 CFR 1.63(d)(2) and 1.33(b).
6. [ ] Application Data Sheet. See 37 CFR 1.76
7. [ ] CD-Rom or CD-R in duplicate, large table or Computer Program (Appendix)
8. Nucleotide and/or Amino Acid Sequence Submission *(if applicable, all necessary)*
   a. [ ] Computer Readable Form (CRF)
   b. Specification Sequence Listing on:
   [ ] CD-ROM or CD-R (2 copies); or
   ii. [ ] Paper
   c. [ ] Statement verifying identify of above copies

ACCOMPANYING APPLICATION PARTS

9. [ ] Assignment Papers (cover sheet & document(s))
10. [ ] 37 CFR 3.73(b) Statement (when there is an assignee) [X] Power of Attorney
11. [ ] English Translation Document *(if applicable)*
12. [ ] Information Disclosure Statement (IDS)/PTO-1449 [ ] Copies of IDS Citations
13. [ ] Preliminary Amendment
14. [X] Return Receipt Postcard (MPEP 503) *(Should be specifically itemized)*
15. [ ] Certified Copy of Priority Document(s) *if foreign priority is claimed)*
16. [ ] Nonpublication Request under 35 U.S.C. 122 (b)(2) (B) (i). Applicant must attach form PTO/SB/35 or its equivalent.
17. [X] Other: Express Mail Certificate

18. **If a CONTINUING APPLICATION**, *check appropriate box and supply the requisite information:*
[ ] Continuation [ ] Divisional [ ] Continuation-in-part (CIP) of prior application No.:

19. CORRESPONDENCE ADDRESS

X *Customer Number or Bar Code Label* 36491
☐ Or Correspondence address below

NAME					
ADDRESS					
CITY		STATE		ZIP	
COUNTRY		TELEPHONE		FAX	

Name: David J. McKenzie Registration No. (Attorney/Agent): 46,919

Signature : David J. McKenzie Date: 4/28/05

Please type a plus sign (+) inside this box → [+]

PTO/SB/05 (12/97)
Approved for use through 9/30/03. OMB 0651-0032
Patent and Trademark Office: U.S. DEPARTMENT OF COMMERCE

Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.

FEE TRANSMITTAL

Note: Effective December 8, 2004.
Patent fees are subject to annual revision.

TOTAL AMOUNT OF PAYMENT	$1050

Complete If Known	
Application Number	Not yet assigned
Filing Date	April 28, 2005
First Named Inventor	R. Drew Major
Group Art Unit	
Examiner Name	
Attorney Docket Number	2082.2.1

METHOD OF PAYMENT (check one)

1. ☐ The Commissioner is hereby authorized to charge indicated fees and credit any over payments to:

Deposit Account No.: _____

Deposit Account Name: _____

☐ Charge Any Additional Fee Required Under 37 CFR 1.16 and 1.17

☐ Charge the Issue Fee In 37 CFR at the Mailing of the Notice of Allowance

2. X Payment Enclosed:

X Check ☐ Money Order ☐ Other

FEE CALCULATION

1. FILING FEE

Large Entity Fee Code	Large Entity Fee ($)	Small Entity Fee Code	Small Entity Fee ($)	Fee Description	Fee Paid
1011	300	2011	150	Utility Fling Fee	150
1111	500	2111	250	Utility Search Fee	250
1311	200	2311	100	Utility Examination Fee	100
1081	250	2081	125	Utility over 100 sheet fee per additional 50	
1085	250	2085	125	Provisional over 100 sheet fee per add. 50	
1005	200	2005	100	Provisional filing fee	
				SUBTOTAL (1)	$ 500

2. CLAIMS

			Extra		Fee from below		Fee Paid
Total Claims	34	-20 =	14	x	25	=	350
Ind. Claims	5	- 3 =	2	x	100	=	200
Multiple Dep. Claims			0	x	180	=	0

Large Entity Fee Code	Large Entity Fee ($)	Small Entity Fee Code	Small Entity Fee ($)	Fee Description
1202	50	2202	25	Claims in excess of 20
1201	200	2201	100	Independent claims in excess of 3
1203	360	2203	180	Multiple dependent claim
1204	200	2204	100	Reissue independent claims over original patent
1205	50	2205	25	Reissue claims in excess of 20 and over original patent

SUBTOTAL (2) 550

FEE CALCULATION (continued)

3. ADDITIONAL FEES

Large Entity Fee Code	Large Entity Fee ($)	Small Entity Fee Code	Small Entity Fee ($)	Fee Description	Fee Paid
1051	130	2051	65	Surcharge - late filing fee or oath	
1052	50	2052	25	Surcharge - late provisional filing fee or cover sheet	
1053	130	1053	130	Non-English specification	
1812	2520	1812	2520	For filing a request for reexamination	
1804	920*	1804	920*	Requesting publication of SIR prior to Examiner action	
1805	1840*	1805	1840*	Requesting publication of SIR after Examiner action	
1251	120	2251	60	Extension for reply within first month	
1252	450	2252	225	Extension for reply within second month	
1253	1020	2253	510	Extension for reply within third month	
1254	1590	2254	795	Extension for reply within fourth month	
1255	2160	2255	1080	Extension for reply within fifth month	
1401	500	2401	250	Notice of Appeal	
1402	500	2402	250	Filing a brief in support of an appeal	
1403	1000	2403	500	Request for oral hearing	
1451	1510	1451	1510	Petition to institute a public use proceeding	
1452	110	2452	55	Petition to revive -unavoidably	
1453	1370	2453	685	Petition to revive - unintentional	
1460	130	1460	130	Petitions to the Commissioner	
1501	1400	2501	700	Utility Issue fee	
1502	800	2502	400	Design issue fee	
1503	1100	2503	550	Plant issue fee	
8007	20	8007	20	Copy of Patent Application as filed	
1807	50	1807	50	Petitions related to provisional applications	
1806	180	1806	180	Submission of Information Disclosure Stmt	
8021	40	8021	40	Recording each patent assignment per property (times number of properties)	
1801	790	2801	395	Request for continued examination (RCE) (37 CFR 1.114)	
1810	790	2810	395	Filing a submission after final rejection (37 CFR 1.129(a))	
1810	790	2810	395	For each additional invention to be examined (37 CFR 1.129(b))	
				Other fee (specify) _____	

*Reduced by Basic Filing Fee

SUBTOTAL (3) $ 0

SUBMITTED BY				Complete (if applicable)	
Typed or Printed Name	David J. McKenzie			Reg. Number	46,919
Signature		Date	Apr 28, 2005	Deposit Account User ID	

Burden Hour Statement: This form is estimated to take 0.2 hours to complete. Time will vary depending upon the needs of the individual case. Any comments on the amount of time you are required to complete this form should be sent to the Chief Information Officer, Patent and Trademark Office, Washington,

PATENT APPLICATION
Docket No.: 2082.2.1

CERTIFICATE OF MAILING BY "EXPRESS MAIL"

"Express Mail" Mailing Label No.: EV577684659US

Date of Deposit: April 28, 2005

I hereby certify that this patent application in the name of R Drew Major and Mark B. Hurst for APPARATUS, SYSTEM, AND METHOD FOR ADAPTIVE-RATE SHIFTING OF STREAMING CONTENT, together with the drawings, a Declaration, Power of Attorney, and Petition, and Check No.3353 for $1050.00 are being deposited with the United States Postal Service "Express Mail Post Office to Addressee" service under 37 C.F.R. § 1.10 on the date indicated above in an envelope addressed to Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.

Respectfully submitted,

David J. McKenzie
Reg. No. 46,919
Attorney for Applicant

Date: April 28, 2005

Kunzler & Associates
8 East Broadway, Suite 600
Salt Lake City, Utah 84111
Telephone: 801/994-4646

1. An apparatus for adaptive-rate content streaming, the apparatus comprising:

an agent controller module configured to simultaneously request at least portions of a plurality of streamlets;

the agent controller module further configured to continuously monitor the progress of subsequent responses to streamlet requests, and in accordance with the progress of the responses, request higher or lower quality streamlets; and

a staging module configured to stage the streamlets and arrange the streamlets for playback on a content player.

2. The apparatus of claim 1, wherein the agent controller module is configured to establish multiple Transmission Control Protocol (TCP) connections with a content server, and request streamlets of varying bitrates.

3. The apparatus of claim 1, wherein each streamlet further comprises a portion of a content file.

4. The apparatus of claim1, wherein the streamlet comprises a Hyper Text Transfer Protocol file.

5. The apparatus of claim 3, wherein the agent controller module is configured to generate a performance factor according to the responses to streamlet requests.

6. The apparatus of claim 4, wherein the agent controller module is configured to upshift to a higher quality streamlet when the performance factor is greater than a selected threshold and the agent controller module determines the higher quality playback can be sustained according to a combination of factors.

KUNZLER & ASSOCIATES
ATTORNEYS AT LAW
8 East Broadway, SUITE 600
SALT LAKE CITY, UTAH 84101

7. The apparatus of claim 5, wherein the combination of factors is selected from the group consisting of an amount of contiguously available streamlets stored in the staging module, a minimum safety margin, and a current read ahead margin.

8. The apparatus of claim 4, wherein the agent controller module is configured to downshift to a lower quality streamlet when the performance factor is less than a second selected threshold.

9. The apparatus of claim 1, wherein the agent controller module is further configured to anticipate streamlet requests and pre-request streamlets to enable fast-forward, skip randomly, and rewind functionality.

10. The apparatus of claim 1, wherein the agent controller module is configured to initially request low quality streamlets to enable instant playback of the content file, and subsequent upshifting according to the performance factor.

KUNZLER & ASSOCIATES
ATTORNEYS AT LAW
8 East Broadway, SUITE 600
SALT LAKE CITY, UTAH 84101

11. A system for adaptive-rate content streaming, the system comprising:

a data communications network;

a content server coupled to the data communications network and having a content module configured to process content and generate a plurality of high and low quality streams;

each of the high and low quality streams comprising a plurality of streamlets;

an agent controller module configured to simultaneously request at least portions of a plurality of streamlets;

the agent controller module further configured to continuously monitor the progress of subsequent responses to streamlet requests, and in accordance with the progress of the responses, request higher or lower quality streamlets; and

a staging module configured to stage the streamlets and arrange the streamlets for playback on a content player.

12. The system of claim 11, wherein the agent controller module is configured to establish multiple Transmission Control Protocol (TCP) connections with the content server, and request streamlets of varying bitrates.

13. The system of claim 11, wherein each streamlet further comprises a portion of a content file.

14. The apparatus of claim 13, wherein the agent controller module is configured to generate a performance factor according to the responses to streamlet requests.

KUNZLER & ASSOCIATES
ATTORNEYS AT LAW
8 East Broadway, SUITE 600
SALT LAKE CITY, UTAH 84101

15. The system of claim 14, wherein the agent controller module is configured to upshift to a higher quality streamlet when the performance factor is greater than a selected threshold and the agent controller module determines the higher quality playback can be sustained according to a combination of factors.

16. The system of claim 15, wherein the combination of factors is selected from the group consisting of an amount of contiguously available streamlets stored in the staging module, a minimum safety margin, and a current read ahead margin.

17. The system of claim 14, wherein the agent controller module is configured to downshift to a lower quality streamlet when the performance factor is less than a second selected threshold.

18. The system of claim 11, wherein the agent controller module is further configured to anticipate streamlet requests and pre-request streamlets to enable fast-forward, skip randomly, and rewind functionality.

19. The system of claim 11, wherein the agent controller module is configured to initially request low quality streamlets to enable instant playback of the content file, and subsequent upshifting according to the performance factor.

KUNZLER & ASSOCIATES
ATTORNEYS AT LAW
8 East Broadway, SUITE 600
SALT LAKE CITY, UTAH 84101

20. A method for presenting rate-adaptive streams, the method comprising:

simultaneously requesting at least portions of a plurality of streamlets;

continuously monitoring the progress of subsequent responses to streamlet requests, and in accordance with the progress of the responses, requesting higher or lower quality streamlets; and

staging the streamlets and arranging the streamlets for playback on a content player.

21. The method of claim 20, further comprising establishing multiple Transmission Control Protocol (TCP) connections with a content server, and requesting streamlets of varying bitrates.

22. The method of claim 20, further comprising generating a performance factor according to responses from streamlet requests.

23. The method of claim 20, further comprising upshifting to a higher quality streamlet when the performance factor is greater than a selected threshold, and determining if the higher quality playback can be sustained.

24. The method of claim 20, further comprising downshifting to a lower quality streamlet when the performance factor is less than a second selected threshold.

25. The method of claim 20, further comprising anticipating streamlet requests and pre-requesting streamlets to enable fast-forward, skip randomly, and rewind functionality.

KUNZLER & ASSOCIATES
ATTORNEYS AT LAW
8 East Broadway, SUITE 600
SALT LAKE CITY, UTAH 84101

26. The method of claim 20, further comprising initially requesting low quality streamlets to enable instant playback of a content file, and subsequent upshifting according to the performance factor.

27. A computer readable storage medium comprising computer readable code configured to carry out operations for presenting rate-adaptive content streams, the operations comprising:

simultaneously requesting at least portions of a plurality of streamlets;

continuously monitoring the progress of subsequent responses to streamlet requests, and in accordance with the progress of the responses, requesting higher or lower quality streamlets; and

staging the streamlets and arranging the streamlets for playback on a content player.

28. The computer readable storage medium of claim 27, wherein the operations further comprise establishing multiple Transmission Control Protocol (TCP) connections with a content server, and requesting streamlets of varying bitrates.

29. The computer readable storage medium of claim 27, wherein the operations further comprise generating a performance factor according to responses from streamlet requests.

30. The computer readable storage medium of claim 27, wherein the operations further comprise upshifting to a higher quality streamlet when the performance factor is greater than a threshold, and determining if the higher quality playback can be sustained.

KUNZLER & ASSOCIATES
ATTORNEYS AT LAW
8 East Broadway, SUITE 600
SALT LAKE CITY, UTAH 84101

31. The computer readable storage medium of claim 27, wherein the operations further comprise downshifting to a lower quality streamlet when the performance factor is less than a second threshold.

32. The computer readable storage medium of claim 27, wherein the operations further comprise anticipating streamlet requests and pre-requesting streamlets to enable fast-forward, skip randomly, and rewind functionality.

33. The computer readable storage medium of claim 27, wherein the operations further comprise initially requesting low quality streamlets to enable instant playback of a content file, and subsequent upshifting according to the performance factor.

34. An apparatus to present rate-adaptive content streams, the apparatus comprising:

means for simultaneously requesting at least portions of a plurality of streamlets;

means for continuously monitoring the progress of subsequent responses to streamlet requests, and in accordance with the progress of the responses, requesting higher or lower quality streamlets; and

means for staging the streamlets and arranging the streamlets for playback on a content player.

KUNZLER & ASSOCIATES
ATTORNEYS AT LAW
8 East Broadway, SUITE 600
SALT LAKE CITY, UTAH 84101

PATENT APPLICATION FEE DETERMINATION RECORD
Effective December 8, 2004

11116783

CLAIMS AS FILED - PART I

	(Column 1)	(Column 2)
TOTAL CLAIMS 34		
FOR	NUMBER FILED	NUMBER EXTRA
TOTAL CHARGEABLE CLAIMS	34 minus 20=	* 14
INDEPENDENT CLAIMS	5 minus 3 =	* 2
MULTIPLE DEPENDENT CLAIM PRESENT		☐

* If the difference in column 1 is less than zero, enter "0" in column 2

SMALL ENTITY TYPE ☑ OR OTHER THAN SMALL ENTITY

RATE	FEE		RATE	FEE
BASIC FEE	150.00	OR	BASIC FEE	300.00
X$ 25=	350	OR	X$50=	
X100=	200	OR	X200=	
+180=		OR	+360=	
TOTAL		OR	TOTAL	

CLAIMS AS AMENDED - PART II

AMENDMENT A

	(Column 1) CLAIMS REMAINING AFTER AMENDMENT		(Column 2) HIGHEST NUMBER PREVIOUSLY PAID FOR	(Column 3) PRESENT EXTRA
Total	*	Minus	**	=
Independent	*	Minus	***	=
FIRST PRESENTATION OF MULTIPLE DEPENDENT CLAIM				☐

SMALL ENTITY OR OTHER THAN SMALL ENTITY

RATE	ADDITIONAL FEE		RATE	ADDITIONAL FEE
X$ 25=		OR	X$50=	
X100=		OR	X200=	
+180=		OR	+360=	
TOTAL ADDIT. FEE		OR	TOTAL ADDIT. FEE	

AMENDMENT B

	(Column 1) CLAIMS REMAINING AFTER AMENDMENT		(Column 2) HIGHEST NUMBER PREVIOUSLY PAID FOR	(Column 3) PRESENT EXTRA
Total	*	Minus	**	=
Independent	*	Minus	***	=
FIRST PRESENTATION OF MULTIPLE DEPENDENT CLAIM				☐

RATE	ADDITIONAL FEE		RATE	ADDITIONAL FEE
X$ 25=		OR	X$50=	
X100=		OR	X200=	
+180=		OR	+360=	
TOTAL ADDIT. FEE		OR	TOTAL ADDIT. FEE	

AMENDMENT C

	(Column 1) CLAIMS REMAINING AFTER AMENDMENT		(Column 2) HIGHEST NUMBER PREVIOUSLY PAID FOR	(Column 3) PRESENT EXTRA
Total	*	Minus	**	=
Independent	*	Minus	***	=
FIRST PRESENTATION OF MULTIPLE DEPENDENT CLAIM				☐

RATE	ADDITIONAL FEE		RATE	ADDITIONAL FEE
X$ 25=		OR	X$50=	
X100=		OR	X200=	
+180=		OR	+360=	
TOTAL ADDIT. FEE		OR	TOTAL ADDIT. FEE	

* If the entry in column 1 is less than the entry in column 2, write "0" in column 3.
** If the "Highest Number Previously Paid For" IN THIS SPACE is less than 20, enter "20."
*** If the "Highest Number Previously Paid For" IN THIS SPACE is less than 3, enter "3."
The "Highest Number Previously Paid For" (Total or Independent) is the highest number found in the appropriate box in column 1.

PATENT APPLICATION FEE DETERMINATION RECORD
Effective December 8, 2004

11116783

CLAIMS AS FILED - PART I

	(Column 1)	(Column 2)
TOTAL CLAIMS 34		
FOR	NUMBER FILED	NUMBER EXTRA
TOTAL CHARGEABLE CLAIMS	34 minus 20=	* 14
INDEPENDENT CLAIMS	5 minus 3 =	* 2
MULTIPLE DEPENDENT CLAIM PRESENT		☐

* If the difference in column 1 is less than zero, enter "0" in column 2

SMALL ENTITY TYPE ☑ OR OTHER THAN SMALL ENTITY

RATE	FEE		RATE	FEE
BASIC FEE	150.00	OR	BASIC FEE	300.00
X$ 25=	350	OR	X$50=	
X100=	200	OR	X200=	
+180=		OR	+360=	
TOTAL	650.	OR	TOTAL	

CLAIMS AS AMENDED - PART II

AMENDMENT A

	(Column 1) CLAIMS REMAINING AFTER AMENDMENT		(Column 2) HIGHEST NUMBER PREVIOUSLY PAID FOR	(Column 3) PRESENT EXTRA
Total	*	Minus	**	=
Independent	*	Minus	***	=
FIRST PRESENTATION OF MULTIPLE DEPENDENT CLAIM				☐

SMALL ENTITY OR OTHER THAN SMALL ENTITY

RATE	ADDITIONAL FEE		RATE	ADDITIONAL FEE
X$ 25=		OR	X$50=	
X100=		OR	X200=	
+180=		OR	+360=	
TOTAL ADDIT. FEE		OR	TOTAL ADDIT. FEE	

AMENDMENT B

	(Column 1) CLAIMS REMAINING AFTER AMENDMENT		(Column 2) HIGHEST NUMBER PREVIOUSLY PAID FOR	(Column 3) PRESENT EXTRA
Total	*	Minus	**	=
Independent	*	Minus	***	=
FIRST PRESENTATION OF MULTIPLE DEPENDENT CLAIM				☐

RATE	ADDITIONAL FEE		RATE	ADDITIONAL FEE
X$ 25=		OR	X$50=	
X100=		OR	X200=	
+180=		OR	+360=	
TOTAL ADDIT. FEE		OR	TOTAL ADDIT. FEE	

AMENDMENT C

	(Column 1) CLAIMS REMAINING AFTER AMENDMENT		(Column 2) HIGHEST NUMBER PREVIOUSLY PAID FOR	(Column 3) PRESENT EXTRA
Total	*	Minus	**	=
Independent	*	Minus	***	=
FIRST PRESENTATION OF MULTIPLE DEPENDENT CLAIM				☐

RATE	ADDITIONAL FEE		RATE	ADDITIONAL FEE
X$ 25=		OR	X$50=	
X100=		OR	X200=	
+180=		OR	+360=	
TOTAL ADDIT. FEE		OR	TOTAL ADDIT. FEE	

* If the entry in column 1 is less than the entry in column 2, write "0" in column 3.
** If the "Highest Number Previously Paid For" IN THIS SPACE is less than 20, enter "20."
*** If the "Highest Number Previously Paid For" IN THIS SPACE is less than 3, enter "3."
The "Highest Number Previously Paid For" (Total or Independent) is the highest number found in the appropriate box in column 1.

Electronic Acknowledgement Receipt

EFS ID:	1924904
Application Number:	11116783
International Application Number:	
Confirmation Number:	5535
Title of Invention:	Apparatus, system, and method for adaptive-rate shifting of streaming content
First Named Inventor/Applicant Name:	R. Drew Major
Customer Number:	36491
Filer:	Brian C. Kunzler/Emily McKenzie
Filer Authorized By:	Brian C. Kunzler
Attorney Docket Number:	2082.2.1
Receipt Date:	29-JUN-2007
Filing Date:	28-APR-2005
Time Stamp:	12:05:45
Application Type:	Utility

Payment information:

Submitted with Payment	no

File Listing:

Document Number	Document Description	File Name	File Size(Bytes)	Multi Part /.zip	Pages (if appl.)
1	Information Disclosure Statement (IDS) Filed	2082_2_1_IDS.pdf	42647	no	3

Warnings:

Information:	
This is not an USPTO supplied IDS fillable form	
Total Files Size (in bytes):	42647

This Acknowledgement Receipt evidences receipt on the noted date by the USPTO of the indicated documents, characterized by the applicant, and including page counts, where applicable. It serves as evidence of receipt similar to a Post Card, as described in MPEP 503.

New Applications Under 35 U.S.C. 111
If a new application is being filed and the application includes the necessary components for a filing date (see 37 CFR 1.53(b)-(d) and MPEP 506), a Filing Receipt (37 CFR 1.54) will be issued in due course and the date shown on this Acknowledgement Receipt will establish the filing date of the application.

National Stage of an International Application under 35 U.S.C. 371
If a timely submission to enter the national stage of an international application is compliant with the conditions of 35 U.S.C. 371 and other applicable requirements a Form PCT/DO/EO/903 indicating acceptance of the application as a national stage submission under 35 U.S.C. 371 will be issued in addition to the Filing Receipt, in due course.

New International Application Filed with the USPTO as a Receiving Office
If a new international application is being filed and the international application includes the necessary components for an international filing date (see PCT Article 11 and MPEP 1810), a Notification of the International Application Number and of the International Filing Date (Form PCT/RO/105) will be issued in due course, subject to prescriptions concerning national security, and the date shown on this Acknowledgement Receipt will establish the international filing date of the application.

CERTIFICATE OF TRANSMISSION

I hereby certify that this correspondence is being transmitted by either submission using the EFS WEB submission system, fax to the U.S. Patent and Trademark office to fax number 571-273-8300, or is being deposited with the United States Postal Service as first class mail in an envelope addressed to: Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450 to on June ___29__, 2007.

__/Brian C. Kunzler/__
Attorney for Applicant

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicant(s) R. Drew Major Serial No.: 11/116,783 Filed: April 28, 2005 Title: APPARATUS, SYSTEM, AND METHOD FOR ADAPTIVE-RATE SHIFTING OF STREAMING CONTENT Attorney Docket No.: 2082.2.1	Group Art Unit: 2155 Examiner: Saleh Najjar

Commissioner for Patents
Washington, DC. 20231

INFORMATION DISCLOSURE STATEMENT

Dear Sir:

This Information Disclosure Statement is submitted:

_X_ under 37 CFR 1.97(b), or
(Within three months of filing national application; or date of entry of international application; or before mailing date of first office action on the merits; whichever occurs last)

____ under 37 CFR 1 .97(c) together with either a:
____ Statement under 37 CFR 1.97(e), or
____ a $180.00 fee under 37 CFR 1.17(p), or
(After the CFR 1.97(b) time period, but before final action or notice of allowance, whichever occurs first)

____ under 37 CFR 1.97(d) together with a:
_____ Statement under 37 CFR 1.97(e), and
_____ a $180.00 fee set forth in 37 CFR 1 .17 (p).
(Filed after final action or notice of allowance, whichever occurs first, but before payment of the issue fee)

____ under 37 CFR 1.97(e),

_____ each item of information contained herein was first cited in a communication from a foreign patent office in a counterpart foreign application not more than three months prior to the filing of the information disclosure statement.

_____ no item of information contained herein was cited in a communication from a foreign patent office in a counterpart foreign application nor was known to any individual designated in §1.56(c) more than three months prior to the filing of the information disclosure statement.

__X__ Applicant(s) submit herewith Form PTO 1449-Information Disclosure Citation together with copies, of patents, publications or other information of which applicant(s) are aware, which applicant(s) believe(s) may be material to the examination of this application and for which there may be a duty to disclose in accordance with 37 CFR 1.56.

The relevance of the attached references is that this is the closest art of which Applicant is aware. Applicant submits that the above references taken alone or in combination neither anticipate nor render obvious the present invention, Consideration of the foregoing in relation to this application is respectfully requested.

It is requested that the information disclosed herein he made of record in this application.

Respectfully submitted,

/Brian C. Kunzler/

Brian C. Kunzler
Attorney/Agent for Applicant(s)
Reg. No. 38,527

Date: *06/29/2007*

Telephone No.: 801-994-4646

FORM PTO-1449	SERIAL NO. 11/116,783	ATTORNEY DOCKET NO. 2082.2.1
LIST OF PATENTS AND PUBLICATIONS FOR APPLICANT'S INFORMATION DISCLOSURE STATEMENT	FILING DATE April 28, 2005	GROUP ART UNIT 2155
(use several sheets if necessary)	APPLICANT(S): R. Drew Major	

REFERENCE DESIGNATION **U.S. PATENT DOCUMENTS**

EXAMINER INITIAL		DOCUMENT NUMBER	DATE	NAME	CLASS/ SUBCLASS	FILING DATE
	A1	6,195,680	02/27/2001	Goldszmidt et al.	709/203	07/23/1998
	A2	6,366,614	04/02/2002	Pian et al.	375/240.02	10/11/1996
	A3	2003/0151753	08/14/2003	Li et al.	358/1.9	06/27/2002
	A4	2005/0108414	05/19/2005	Taylor et al.	709/231	11/15/2004

FOREIGN PATENT DOCUMENTS

EXAMINER INITIAL		DOCUMENT NUMBER	DATE	COUNTRY	CLASS/ SUBCLASS	TRANSLATION	
						YES	NO

NON-PATENT DOCUMENTS

EXAMINER INITIAL		DOCUMENT (Including Author, Title, Source, and Pertinent Pages

EXAMINER	**DATE CONSIDERED**

EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609; Draw line through citation if not in conformance and not considered. Include copy of this form with next communication to applicant(s).

PTO/SB/08a (05-07)
Approved for use through 11/30/2007. OMB 0651-0031
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE
Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it contains a valid OMB control number.

INFORMATION DISCLOSURE STATEMENT BY APPLICANT

(Not for submission under 37 CFR 1.99)

Application Number	11116783
Filing Date	2005-04-28
First Named Inventor	R. Drew Major
Art Unit	2143
Examiner Name	
Attorney Docket Number	2082.2.1

U.S.PATENTS Remove

Examiner Initial*	Cite No	Patent Number	Kind Code[1]	Issue Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1	4535355		1985-08-13	Arn et al.	
	2	5953506		1999-09-14	Kalra et al.	
	3	6490627	B1	2002-12-03	Kalra et al.	
	4	6574591	B1	2003-06-03	Kleiman et al.	
	5	6604118	B2	2003-08-05	Kleiman et al.	

If you wish to add additional U.S. Patent citation information please click the Add button. Add

U.S.PATENT APPLICATION PUBLICATIONS Remove

Examiner Initial*	Cite No	Publication Number	Kind Code[1]	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1					

If you wish to add additional U.S. Published Application citation information please click the Add button. Add

FOREIGN PATENT DOCUMENTS Remove

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	11116783
	Filing Date	2005-04-28
	First Named Inventor	R. Drew Major
	Art Unit	2143
	Examiner Name	
	Attorney Docket Number	2082.2.1

Examiner Initial*	Cite No	Foreign Document Number[3]	Country Code[2] i	Kind Code[4]	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear	T[5]
	1							☐

If you wish to add additional Foreign Patent Document citation information please click the Add button Add

NON-PATENT LITERATURE DOCUMENTS Remove

Examiner Initials*	Cite No	Include name of the author (in CAPITAL LETTERS), title of the article (when appropriate), title of the item (book, magazine, journal, serial, symposium, catalog, etc), date, pages(s), volume-issue number(s), publisher, city and/or country where published.	T[5]
	1		☐

If you wish to add additional non-patent literature document citation information please click the Add button Add

EXAMINER SIGNATURE

Examiner Signature		Date Considered	

*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609. Draw line through a citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

[1] See Kind Codes of USPTO Patent Documents at www.USPTO.GOV or MPEP 901.04. [2] Enter office that issued the document, by the two-letter code (WIPO Standard ST.3). [3] For Japanese patent documents, the indication of the year of the reign of the Emperor must precede the serial number of the patent document. [4] Kind of document by the appropriate symbols as indicated on the document under WIPO Standard ST.16 if possible. [5] Applicant is to place a check mark here if English language translation is attached.

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	11116783
	Filing Date	2005-04-28
	First Named Inventor	R. Drew Major
	Art Unit	2143
	Examiner Name	
	Attorney Docket Number	2082.2.1

CERTIFICATION STATEMENT

Please see 37 CFR 1.97 and 1.98 to make the appropriate selection(s):

☐ That each item of information contained in the information disclosure statement was first cited in any communication from a foreign patent office in a counterpart foreign application not more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(1).

OR

☐ That no item of information contained in the information disclosure statement was cited in a communication from a foreign patent office in a counterpart foreign application, and, to the knowledge of the person signing the certification after making reasonable inquiry, no item of information contained in the information disclosure statement was known to any individual designated in 37 CFR 1.56(c) more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(2).

☒ See attached certification statement.

☐ Fee set forth in 37 CFR 1.17 (p) has been submitted herewith.

☐ None

SIGNATURE

A signature of the applicant or representative is required in accordance with CFR 1.33, 10.18. Please see CFR 1.4(d) for the form of the signature.

Signature	/Brian C. Kunzler/	Date (YYYY-MM-DD)	2008-04-10
Name/Print	Brian C. Kunzler	Registration Number	38,527

This collection of information is required by 37 CFR 1.97 and 1.98. The information is required to obtain or retain a benefit by the public which is to file (and by the USPTO to process) an application. Confidentiality is governed by 35 U.S.C. 122 and 37 CFR 1.14. This collection is estimated to take 1 hour to complete, including gathering, preparing and submitting the completed application form to the USPTO. Time will vary depending upon the individual case. Any comments on the amount of time you require to complete this form and/or suggestions for reducing this burden, should be sent to the Chief Information Officer, U.S. Patent and Trademark Office, U.S. Department of Commerce, P.O. Box 1450, Alexandria, VA 22313-1450. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. **SEND TO: Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.**

Privacy Act Statement

The Privacy Act of 1974 (P.L. 93-579) requires that you be given certain information in connection with your submission of the attached form related to a patent application or patent. Accordingly, pursuant to the requirements of the Act, please be advised that: (1) the general authority for the collection of this information is 35 U.S.C. 2(b)(2); (2) furnishing of the information solicited is voluntary; and (3) the principal purpose for which the information is used by the U.S. Patent and Trademark Office is to process and/or examine your submission related to a patent application or patent. If you do not furnish the requested information, the U.S. Patent and Trademark Office may not be able to process and/or examine your submission, which may result in termination of proceedings or abandonment of the application or expiration of the patent.

The information provided by you in this form will be subject to the following routine uses:

1. The information on this form will be treated confidentially to the extent allowed under the Freedom of Information Act (5 U.S.C. 552) and the Privacy Act (5 U.S.C. 552a). Records from this system of records may be disclosed to the Department of Justice to determine whether the Freedom of Information Act requires disclosure of these record s.

2. A record from this system of records may be disclosed, as a routine use, in the course of presenting evidence to a court, magistrate, or administrative tribunal, including disclosures to opposing counsel in the course of settlement negotiations.

3. A record in this system of records may be disclosed, as a routine use, to a Member of Congress submitting a request involving an individual, to whom the record pertains, when the individual has requested assistance from the Member with respect to the subject matter of the record.

4. A record in this system of records may be disclosed, as a routine use, to a contractor of the Agency having need for the information in order to perform a contract. Recipients of information shall be required to comply with the requirements of the Privacy Act of 1974, as amended, pursuant to 5 U.S.C. 552a(m).

5. A record related to an International Application filed under the Patent Cooperation Treaty in this system of records may be disclosed, as a routine use, to the International Bureau of the World Intellectual Property Organization, pursuant to the Patent Cooperation Treaty.

6. A record in this system of records may be disclosed, as a routine use, to another federal agency for purposes of National Security review (35 U.S.C. 181) and for review pursuant to the Atomic Energy Act (42 U.S.C. 218(c)).

7. A record from this system of records may be disclosed, as a routine use, to the Administrator, General Services, or his/her designee, during an inspection of records conducted by GSA as part of that agency's responsibility to recommend improvements in records management practices and programs, under authority of 44 U.S.C. 2904 and 2906. Such disclosure shall be made in accordance with the GSA regulations governing inspection of records for this purpose, and any other relevant (i.e., GSA or Commerce) directive. Such disclosure shall not be used to make determinations about individuals.

8. A record from this system of records may be disclosed, as a routine use, to the public after either publication of the application pursuant to 35 U.S.C. 122(b) or issuance of a patent pursuant to 35 U.S.C. 151. Further, a record may be disclosed, subject to the limitations of 37 CFR 1.14, as a routine use, to the public if the record was filed in an application which became abandoned or in which the proceedings were terminated and which application is referenced by either a published application, an application open to public inspections or an issued patent.

9. A record from this system of records may be disclosed, as a routine use, to a Federal, State, or local law enforcement agency, if the USPTO becomes aware of a violation or potential violation of law or regulation.

Electronic Acknowledgement Receipt

EFS ID:	3136767
Application Number:	11116783
International Application Number:	
Confirmation Number:	5535
Title of Invention:	Apparatus, system, and method for adaptive-rate shifting of streaming content
First Named Inventor/Applicant Name:	R. Drew Major
Customer Number:	36491
Filer:	Brian C. Kunzler/Heather Babb
Filer Authorized By:	Brian C. Kunzler
Attorney Docket Number:	2082.2.1
Receipt Date:	10-APR-2008
Filing Date:	28-APR-2005
Time Stamp:	18:52:54
Application Type:	Utility under 35 USC 111(a)

Payment information:

Submitted with Payment	no

File Listing:

Document Number	Document Description	File Name	File Size(Bytes) /Message Digest	Multi Part /.zip	Pages (if appl.)
1	Information Disclosure Statement Letter	2082_2_1_IDS_Letter.pdf	24932 06ed9e2ce3c3951f330d7bdfbf6f16334a3b9f35	no	2

Warnings:

Information:

2	Information Disclosure Statement (IDS) Filed	US_IDS_Form__SB_08a.pdf	574628 2a6919f66b2a72bdc5326b78293c7b47e4b60be5	no	4

Warnings:

Information:

Total Files Size (in bytes):	599560

This Acknowledgement Receipt evidences receipt on the noted date by the USPTO of the indicated documents, characterized by the applicant, and including page counts, where applicable. It serves as evidence of receipt similar to a Post Card, as described in MPEP 503.

New Applications Under 35 U.S.C. 111
If a new application is being filed and the application includes the necessary components for a filing date (see 37 CFR 1.53(b)-(d) and MPEP 506), a Filing Receipt (37 CFR 1.54) will be issued in due course and the date shown on this Acknowledgement Receipt will establish the filing date of the application.

National Stage of an International Application under 35 U.S.C. 371
If a timely submission to enter the national stage of an international application is compliant with the conditions of 35 U.S.C. 371 and other applicable requirements a Form PCT/DO/EO/903 indicating acceptance of the application as a national stage submission under 35 U.S.C. 371 will be issued in addition to the Filing Receipt, in due course.

New International Application Filed with the USPTO as a Receiving Office
If a new international application is being filed and the international application includes the necessary components for an international filing date (see PCT Article 11 and MPEP 1810), a Notification of the International Application Number and of the International Filing Date (Form PCT/RO/105) will be issued in due course, subject to prescriptions concerning national security, and the date shown on this Acknowledgement Receipt will establish the international filing date of the application.

CERTIFICATE OF TRANSMISSION

I hereby certify that this correspondence is being transmitted by either submission using the EFS WEB submission system, fax to the U.S. Patent and Trademark office to fax number 571-273-8300, or is being deposited with the United States Postal Service as first class mail in an envelope addressed to: Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450 to on April 10, 2008.

/Brian C. Kunzler/
Attorney for Applicant

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicant(s) Major et al. Serial No.: 11/116,783 Filed: April 28, 2005 Title: Apparatus, System, and Method for Adaptive-Rate Shifting of Streaming Content Attorney Docket No.: 2082.2.1	Group Art Unit: 2143 Examiner: not yet assigned

Commissioner for Patents
Washington, DC. 20231

INFORMATION DISCLOSURE STATEMENT

Dear Sir:

This Information Disclosure Statement is submitted:

_X_ under 37 CFR 1.97(b), or
(Within three months of filing national application; or date of entry of international application; or before mailing date of first office action on the merits; whichever occurs last)

____ under 37 CFR 1 .97(c) together with either a:
____ Statement under 37 CFR 1.97(e), or
____ a $180.00 fee under 37 CFR 1.17(p), or
(After the CFR 1.97(b) time period, but before final action or notice of allowance, whichever occurs first)

____ under 37 CFR 1.97(d) together with a:
_____ Statement under 37 CFR 1.97(e), and
_____ a $180.00 fee set forth in 37 CFR 1 .17 (p).
(Filed after final action or notice of allowance, whichever occurs first, but before payment of the issue fee)

____ under 37 CFR 1.97(e),

_____ each item of information contained herein was first cited in a communication from a foreign patent office in a counterpart foreign application not more than three months prior to the filing of the information disclosure statement.

_____ no item of information contained herein was cited in a communication from a foreign patent office in a counterpart foreign application nor was known to any individual designated in §1.56(c) more than three months prior to the filing of the information disclosure statement.

__X___ Applicant(s) submit herewith Form PTO/SB/08a-Information Disclosure Statement together with copies, of patents, publications or other information of which applicant(s) are aware, which applicant(s) believe(s) may be material to the examination of this application and for which there may be a duty to disclose in accordance with 37 CFR 1.56.

The relevance of the attached references is that this is the closest art of which Applicant is aware. Applicant submits that the above references taken alone or in combination neither anticipate nor render obvious the present invention, Consideration of the foregoing in relation to this application is respectfully requested.

It is requested that the information disclosed herein he made of record in this application.

Respectfully submitted,

/Brian C. Kunzler/

Brian C. Kunzler
Attorney/Agent for Applicant(s)
Reg. No. 38,527

Date: April 10, 2008

Telephone No.: 801-994-4646

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

BIB DATA SHEET

CONFIRMATION NO. 5535

SERIAL NUMBER	FILING or 371(c) DATE	CLASS	GROUP ART UNIT	ATTORNEY DOCKET NO.
11/116,783	04/28/2005 RULE	709	2144	2082.2.1

APPLICANTS
R. Drew Major, Orem, UT;
Mark B. Hurst, Cedar Hills, UT;

**** CONTINUING DATA ***************************
This appln claims benefit of 60/566,831 04/30/2004

**** FOREIGN APPLICATIONS ***************************

**** IF REQUIRED, FOREIGN FILING LICENSE GRANTED **** ** SMALL ENTITY **
05/26/2005

Foreign Priority claimed	☐ Yes ☑ No		STATE OR COUNTRY	SHEETS DRAWINGS	TOTAL CLAIMS	INDEPENDENT CLAIMS
35 USC 119(a-d) conditions met	☐ Yes ☑ No	☐ Met after Allowance				
Verified and Acknowledged	/NINOS J DONABED/ Examiner's Signature	Initials	UT	7	34	5

ADDRESS
Kunzler & McKenzie
8 EAST BROADWAY
SUITE 600
SALT LAKE CITY, UT 84111
UNITED STATES

TITLE
Apparatus, system, and method for adaptive-rate shifting of streaming content

FILING FEE RECEIVED		
1050	FEES: Authority has been given in Paper No._____ to charge/credit DEPOSIT ACCOUNT No._____ for following:	☐ All Fees ☐ 1.16 Fees (Filing) ☐ 1.17 Fees (Processing Ext. of time) ☐ 1.18 Fees (Issue) ☐ Other _____ ☐ Credit

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
11/116,783	04/28/2005	R. Drew Major	2082.2.1	5535

36491 7590 05/14/2008
Kunzler & McKenzie
8 EAST BROADWAY
SUITE 600
SALT LAKE CITY, UT 84111

EXAMINER
DONABED, NINOS J

ART UNIT	PAPER NUMBER
2144	

MAIL DATE	DELIVERY MODE
05/14/2008	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Office Action Summary	Application No.	Applicant(s)	
	11/116,783	MAJOR ET AL.	
	Examiner	Art Unit	
	NINOS DONABED	2144	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *28 April 2005*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-34* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-34* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *28 April 2005* is/are: a)☒ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)

2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)

3) ☒ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date *06/29/2007, 04/10/2008*.

4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .

5) ☐ Notice of Informal Patent Application

6) ☐ Other: _____.

DETAILED ACTION

Claim Rejections - 35 USC § 101

1. 35 U.S.C. 101 reads as follows:

> Whoever invents or discovers any new and useful process, machine, manufacture, or composition of matter, or any new and useful improvement thereof, may obtain a patent therefor, subject to the conditions and requirements of this title.

Claims 1-10 and 34 are rejected under 35 U.S.C. 101 because they are geared towards software per se. The apparatus claimed is software implemented and thus it is rejected under 35 U.S.C 101.

Further regarding Claims 27-33 are rejected under 35 U.S.C. 101 because they are geared towards software per se. The computer readable medium is defined in the specification as a "signal bearing medium". A signal is not a tangible physical article or object (form of matter), thus it is not a machine or a composition of matter. The signal, which is a form of energy, does not fall within either of the two definition of a manufacture, and thus does not fall within one of the four statutory classes of ***35 USC § 101***.

Claim Rejections - 35 USC § 112

2. The following is a quotation of the second paragraph of 35 U.S.C. 112:

> The specification shall conclude with one or more claims particularly pointing out and distinctly claiming the subject matter which the applicant regards as his invention.

3. Claims 1-34 are rejected under 35 U.S.C. 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention.

Regarding claims 1, 11, 20, 27, and 34 the phrase “an agent controller module configured to simultaneously request at least portions of a plurality of streamlets” is vague and unclear. The reason for this is that the claim does not state what the request is being done simultaneously with.

Claims 2-10, 12-19, 21-26, and 28-33 are rejected for being dependent on claims 1, 11, 20, and 27 respectively.

Regarding Claims 7 and 16 the phrase “a minimum safety margin, and a current read ahead margin” is vague and unclear.

Regarding claim 9, the term “pre-request streamlets” is vague and unclear and it is unknown what a "pre-request" is.

4. Claim 17, 24, and 31 recite the limitation "second selected threshold” in the claim. There is insufficient antecedent basis for this limitation in the claim. There is no "first selected threshold”.

Claim Rejections - 35 USC § 102

5. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

A person shall be entitled to a patent unless –

(b) the invention was patented or described in a printed publication in this or a foreign country or in public use or on sale in this country, more than one year prior to the date of application for patent in the United States.

6. Claims 1, 3, 9-11, 13-20, 22-27 and 29-34 are rejected under 35 U.S.C. 102(b) as being anticipated by Radford (United States Patent Application Publication Number 2002/0144276.).

Regarding **Claim 1**,

Radford teaches apparatus for adaptive-rate content streaming, the apparatus comprising:

an agent controller module configured to simultaneously request at least portions of a plurality of streamlets; **(See figure 1 and paragraph [0018] – [0020], Radford teaches a request from a client.)**

the agent controller module further configured to continuously monitor the progress of subsequent responses to streamlet requests, and in accordance with the progress of the responses, request higher or lower quality streamlets; and **(See figure 2 and paragraphs [0029] – [0031], Radford teaches monitoring of progress of streamlets and requesting a change in the quality of the stream.)**

a staging module configured to stage the streamlets and arrange the streamlets for playback on a content player. **(See figure 2 and paragraphs [0029] – [0031], Radford teaches a media player which plays the stream.)**

Regarding **Claim 3**,

Radford teaches the apparatus of claim 1, wherein each streamlet further comprises a portion of a content file. **(See paragraphs [0008] – [0011], Radford.)**

Regarding **Claim 9**,

Radford teaches the apparatus of claim 1, wherein the agent controller module is further configured to anticipate streamlet requests and pre-request streamlets to enable fast-forward, skip randomly, and rewind functionality. **(See figure 2 and paragraphs [0030] - [0033], Radford.)**

Regarding **Claim 10**,

Radford teaches the apparatus of claim 1, wherein the agent controller module is configured to initially request low quality streamlets to enable instant playback of the content file, and subsequent upshifting according to the performance factor. **(See figure 1 and paragraphs [0008] – [0012], Radford.)**

Regarding **Claim 11**,

Radford teaches a system for adaptive-rate content streaming, the system comprising: a data communications network;

a content server coupled to the data communications network and having a content module configured to process content and generate a plurality of high and low quality streams; **(See figure 1 and paragraphs [0017] – [0020],**

Radford teaches a content sever able to process content and generate different quality streams.)

each of the high and low quality streams comprising a plurality of streamlets; **(See figure 1 and and paragraphs [0017] – [0020], Radford teaches high and low quality streams.)**

an agent controller module configured to simultaneously request at least portions of a plurality of streamlets; **(See figure 1 and paragraph [0018] – [0020], Radford teaches a request from a client.)**

the agent controller module further configured to continuously monitor the progress of subsequent responses to streamlet requests, and in accordance with the progress of the responses, request higher or lower quality streamlets; and **(See figure 2 and paragraphs [0029] – [0031], Radford teaches monitoring of progress of streamlets and requesting a change in the quality of the stream.)**

a staging module configured to stage the streamlets and arrange the streamlets for playback on a content player. **(See figure 2 and paragraphs [0029] – [0031], Radford teaches a media player which plays the stream.)**

Regarding **Claim 13**,

Radford teaches the system of claim 11, wherein each streamlet further comprises a portion of a content file. **(See paragraphs [0008] – [0011], Radford.)**

Regarding **Claim 14**,

Radford teaches the apparatus of claim 13, wherein the agent controller module is configured to generate a performance factor according to the responses to streamlet requests. **(See paragraphs [0008] – [0011], Radford.)**

Regarding **Claim 15**,

Radford teaches the system of claim 14, wherein the agent controller module is configured to upshift to a higher quality streamlet when the performance factor is greater than a selected threshold and the agent controller module determines the higher quality playback can be sustained according to a combination of factors. **(See figure 1 and paragraphs [0024] – [0027], Radford.)**

Regarding **Claim 16**,

Radford teaches the system of claim 15, wherein the combination of factors is selected from the group consisting of an amount of contiguously available streamlets stored in the staging module, a minimum safety margin, and a current read ahead margin. **(See figure 1 and paragraphs [0022] – [0026], Radford.)**

Regarding **Claim 17**,

Radford teaches the system of claim 14, wherein the agent controller module is configured to downshift to a lower quality streamlet when the

performance factor is less than a second selected threshold. **(See figure 1 and paragraphs [0024] – [0027], Radford.)**

Regarding **Claim 18**,

Radford teaches the system of claim 11, wherein the agent controller module is further configured to anticipate streamlet requests and pre-request streamlets to enable fast-forward, skip randomly, and rewind functionality. **(See figure 2 and paragraphs [0030] - [0033], Radford.)**

Regarding **Claim 19**,

Radford teaches the system of claim 11, wherein the agent controller module is configured to initially request low quality streamlets to enable instant playback of the content file, and subsequent upshifting according to the performance factor. **(See figure 1 and paragraphs [0008] – [0012], Radford.)**

Regarding **Claim 20**,

Radford teaches a method for presenting rate-adaptive streams, the method comprising:

simultaneously requesting at least portions of a plurality of streamlets; **(See figure 1 and paragraph [0018] – [0020], Radford teaches a request from a client.)**

continuously monitoring the progress of subsequent responses to streamlet requests, and in accordance with the progress of the responses,

requesting higher or lower quality streamlets; and **(See figure 2 and paragraphs [0029] – [0031], Radford teaches monitoring of progress of streamlets and requesting a change in the quality of the stream.)**

staging the streamlets and arranging the streamlets for playback on a content player. **(See figure 2 and paragraphs [0029] – [0031], Radford teaches a media player which plays the stream.)**

Regarding **Claim 22**,

Radford teaches the method of claim 20, further comprising generating a performance factor according to responses from streamlet requests. **(See paragraphs [0008] – [0011], Radford.)**

Regarding **Claim 23**,

Radford teaches the method of claim 20, further comprising upshifting to a higher quality streamlet when the performance factor is greater than a selected threshold, and determining if the higher quality playback can be sustained. **(See figure 1 and paragraphs [0024] – [0027], Radford.)**

Regarding **Claim 24**,

Radford teaches the method of claim 20, further comprising downshifting to a lower quality streamlet when the performance factor is less than a second selected threshold. **(See figure 1 and paragraphs [0024] – [0026], Radford.)**

Regarding **Claim 25**,

Radford teaches the method of claim 20, further comprising anticipating streamlet requests and pre-requesting streamlets to enable fast-forward, skip randomly, and rewind functionality. **(See figure 2 and paragraphs [0030] - [0033], Radford.)**

Regarding **Claim 26**,

Radford teaches the method of claim 20, further comprising initially requesting low quality streamlets to enable instant playback of a content file, and subsequent upshifting according to the performance factor. **(See figure 1 and paragraphs [0008] – [0012], Radford.)**

Regarding **Claim 27**,

Radford teaches a computer readable storage medium comprising computer readable code configured to carry out operations for presenting rate-adaptive content streams, the operations comprising:

simultaneously requesting at least portions of a plurality of streamlets; **(See figure 1 and paragraph [0018] – [0020], Radford teaches a request from a client.)**

continuously monitoring the progress of subsequent responses to streamlet requests, and in accordance with the progress of the responses, **(See figure 2 and paragraphs [0029] – [0031], Radford teaches monitoring of**

progress of streamlets and requesting a change in the quality of the stream.)

requesting higher or lower quality streamlets; and **(See paragraphs [0025] – [0028], Radford teaches requesting high or lower quality strem.)**

staging the streamlets and arranging the streamlets for playback on a content player. **(See figure 2 and paragraphs [0029] – [0031], Radford teaches a media player which plays the stream.)**

Regarding **Claim 29**,

Radford teaches the computer readable storage medium of claim 27, wherein the operations further comprise generating a performance factor according to responses from streamlet requests. **(See paragraphs [0008] – [0011], Radford.)**

Regarding **Claim 30**,

Radford teaches the computer readable storage medium of claim 27, wherein the operations further comprise upshifting to a higher quality streamlet when the performance factor is greater than a threshold, and determining if the higher quality playback can be sustained. **(See figure 1 and paragraphs [0024] – [0027], Radford.)**

Regarding **Claim 31**,

Radford teaches the computer readable storage medium of claim 27, wherein the operations further comprise downshifting to a lower quality streamlet when the performance factor is less than a second threshold. **(See figure 1 and paragraphs [0024] – [0026], Radford.)**

Regarding **Claim 32**,

Radford teaches the computer readable storage medium of claim 27, wherein the operations further comprise anticipating streamlet requests and pre-requesting streamlets to enable fast-forward, skip randomly, and rewind functionality. **(See figure 2 and paragraphs [0030] - [0033], Radford.)**

Regarding **Claim 33**,

Radford teaches the computer readable storage medium of claim 27, wherein the operations further comprise initially requesting low quality streamlets to enable instant playback of a content file, and subsequent upshifting according to the performance factor. **(See figure 1 and paragraphs [0008] – [0012], Radford.)**

Regarding **Claim 34**,

Radford teaches an apparatus to present rate-adaptive content streams, the apparatus comprising:

means for simultaneously requesting at least portions of a plurality of streamlets; **(See figure 1 and paragraph [0018] – [0020], Radford teaches a request from a client.)**

means for continuously monitoring the progress of subsequent responses to streamlet requests, and in accordance with the progress of the responses, requesting higher or lower quality streamlets; and **(See figure 2 and paragraphs [0029] – [0031], Radford teaches monitoring of progress of streamlets and requesting a change in the quality of the stream.)**

means for staging the streamlets and arranging the streamlets for playback on a content player. **(See figure 2 and paragraphs [0029] – [0031], Radford teaches a media player which plays the stream.)**

Claim Rejections - 35 USC § 103

7. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

8. Claims 2, 4-8, 12, 21, and 28 are rejected under 35 U.S.C. 103(a) as being unpatentable over Radford (United States Patent Application Publication Number 2002/0144276.).

Regarding **Claim 2**,

Radford teaches the apparatus of claim 1,

Radford further teaches connecting to the internet to download the stream to the media player on the client. **(See paragraphs [0018] - [0019], Radford.)**

Radford does not explicitly teach wherein the agent controller module is configured to establish multiple Transmission Control Protocol (TCP) connections with a content server, and request streamlets of varying bitrates.

However, it would have been obvious to one of ordinary skill in the art at the time the invention was made to have known that a client connecting to a server to request a stream of data would do so using the TCP protocol.

Regarding **Claim 4**,

Radford teaches the apparatus of claim 1,

Radford further teaches RTSP and RTP protocols. **(See paragraphs [0025] - [0026], Radford.)**

Radford does not explicitly teach wherein the streamlet comprises a Hyper Text Transfer Protocol file.

However, one of ordinary skill in the art at the time the invention was made would have known that the stream transfer could be done using any number of transfer protocols including HTTP, RTSP, and RTP.

Regarding **Claim 5**,

Radford teaches the apparatus of claim 3, wherein the agent controller module is configured to generate a performance factor according to the responses to streamlet requests. **(See paragraphs [0008] – [0011], Radford.)**

Regarding **Claim 6**,

Radford teaches the apparatus of claim 4, wherein the agent controller module is configured to upshift to a higher quality streamlet when the performance factor is greater than a selected threshold and the agent controller module determines the higher quality playback can be sustained according to a combination of factors. **(See figure 1 and paragraphs [0024] – [0027], Radford.)**

Regarding **Claim 7**,

Radford teaches the apparatus of claim 5, wherein the combination of factors is selected from the group consisting of an amount of contiguously available streamlets stored in the staging module, a minimum safety margin, and a current read ahead margin. **(See figure 1 and paragraphs [0022] – [0026], Radford.)**

Regarding **Claim 8**,

Radford teaches the apparatus of claim 4, wherein the agent controller module is configured to downshift to a lower quality streamlet when the

performance factor is less than a second selected threshold. **(See figure 1 and paragraphs [0024] – [0026], Radford.)**

Regarding **Claim 12**,

Radford further teaches connecting to the internet to download the stream to the media player on the client. **(See paragraphs [0018] - [0019], Radford.)**

Radford does not explicitly teach wherein the agent controller module is configured to establish multiple Transmission Control Protocol (TCP) connections with a content server, and request streamlets of varying bitrates.

However, it would have been obvious to one of ordinary skill in the art at the time the invention was made to have known that a client connecting to a server to request a stream of data would do so using the TCP protocol.

Regarding **Claim 21**,

Radford teaches the method of claim 20.

Radford further teaches connecting to the internet to download the stream to the media player on the client. **(See paragraphs [0018] - [0019], Radford.)**

Radford does not explicitly teach wherein the agent controller module is configured to establish multiple Transmission Control Protocol (TCP) connections with a content server, and request streamlets of varying bitrates.

However, it would have been obvious to one of ordinary skill in the art at the time the invention was made to have known that a client connecting to a server to request a stream of data would do so using the TCP protocol.

Regarding **Claim 28**,

Radford teaches the computer readable storage medium of claim 27.

Radford further teaches connecting to the internet to download the stream to the media player on the client. **(See paragraphs [0018] - [0019], Radford.)**

Radford does not explicitly teach wherein the agent controller module is configured to establish multiple Transmission Control Protocol (TCP) connections with a content server, and request streamlets of varying bitrates.

However, it would have been obvious to one of ordinary skill in the art at the time the invention was made to have known that a client connecting to a server to request a stream of data would do so using the TCP protocol.

Conclusion

Any response to this Office Action should be **faxed** to (571) 272-8300 or **mailed** to:

Commissioner for Patents
P.O. Box 1450
Alexandria, Virginia 22313-1450

Hand-delivered responses should be brought to
Customer Service Window
Randolph Building
401 Dulany Street
Alexandria, Virginia 22314

Any inquiry concerning this communication or earlier communications from the examiner should be directed to NINOS DONABED whose telephone number

is (571)270-3526. The examiner can normally be reached on Monday-Friday, 7:30 AM-5:00 PM EST.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, William Vaughn can be reached on (571) 272-3922. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

/Ninos Donabed/
Examiner, Art Unit 2144
/William C. Vaughn, Jr./
Supervisory Patent Examiner, Art Unit 2144

CERTIFICATE OF TRANSMISSION

I hereby certify that this correspondence is being transmitted by either submission using the EFS WEB submission system, fax to the U.S. Patent and Trademark office to fax number 571-273-8300, or is being deposited with the United States Postal Service as first class mail in an envelope addressed to: Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450 to on June ___29__, 2007.

/Brian C. Kunzler/
Attorney for Applicant

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicant(s) R. Drew Major Serial No.: 11/116,783 Filed: April 28, 2005 Title: APPARATUS, SYSTEM, AND METHOD FOR ADAPTIVE-RATE SHIFTING OF STREAMING CONTENT Attorney Docket No.: 2082.2.1	Group Art Unit: 2155 Examiner: Saleh Najjar

Commissioner for Patents
Washington, DC. 20231

INFORMATION DISCLOSURE STATEMENT

Dear Sir:

This Information Disclosure Statement is submitted:

_X_ under 37 CFR 1.97(b), or
(Within three months of filing national application; or date of entry of international application; or before mailing date of first office action on the merits; whichever occurs last)

____ under 37 CFR 1 .97(c) together with either a:
____ Statement under 37 CFR 1.97(e), or
____ a $180.00 fee under 37 CFR 1.17(p), or
(After the CFR 1.97(b) time period, but before final action or notice of allowance, whichever occurs first)

____ under 37 CFR 1.97(d) together with a:
_____ Statement under 37 CFR 1.97(e), and
_____ a $180.00 fee set forth in 37 CFR 1 .17 (p).
(Filed after final action or notice of allowance, whichever occurs first, but before payment of the issue fee)

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /N.D

____ under 37 CFR 1.97(e),

_____ each item of information contained herein was first cited in a communication from a foreign patent office in a counterpart foreign application not more than three months prior to the filing of the information disclosure statement.

_____ no item of information contained herein was cited in a communication from a foreign patent office in a counterpart foreign application nor was known to any individual designated in §1.56(c) more than three months prior to the filing of the information disclosure statement.

__X__ Applicant(s) submit herewith Form PTO 1449-Information Disclosure Citation together with copies, of patents, publications or other information of which applicant(s) are aware, which applicant(s) believe(s) may be material to the examination of this application and for which there may be a duty to disclose in accordance with 37 CFR 1.56.

The relevance of the attached references is that this is the closest art of which Applicant is aware. Applicant submits that the above references taken alone or in combination neither anticipate nor render obvious the present invention, Consideration of the foregoing in relation to this application is respectfully requested.

It is requested that the information disclosed herein he made of record in this application.

Respectfully submitted,

/Brian C. Kunzler/

Brian C. Kunzler
Attorney/Agent for Applicant(s)
Reg. No. 38,527

Date: *06/29/2007*

Telephone No.: 801-994-4646

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /N.D

FORM PTO-1449	SERIAL NO. 11/116,783	ATTORNEY DOCKET NO. 2082.2.1
LIST OF PATENTS AND PUBLICATIONS FOR APPLICANT'S INFORMATION DISCLOSURE STATEMENT	FILING DATE April 28, 2005	GROUP ART UNIT 2155
(use several sheets if necessary)	APPLICANT(S): R. Drew Major	

REFERENCE DESIGNATION **U.S. PATENT DOCUMENTS**

EXAMINER INITIAL		DOCUMENT NUMBER	DATE	NAME	CLASS/ SUBCLASS	FILING DATE
	A1	6,195,680	02/27/2001	Goldszmidt et al.	709/203	07/23/1998
	A2	6,366,614	04/02/2002	Pian et al.	375/240.02	10/11/1996
	A3	2003/0151753	08/14/2003	Li et al.	358/1.9	06/27/2002
	A4	2005/0108414	05/19/2005	Taylor et al.	709/231	11/15/2004

FOREIGN PATENT DOCUMENTS

EXAMINER INITIAL		DOCUMENT NUMBER	DATE	COUNTRY	CLASS/ SUBCLASS	TRANSLATION	
						YES	NO

NON-PATENT DOCUMENTS

EXAMINER INITIAL		DOCUMENT (Including Author, Title, Source, and Pertinent Pages

EXAMINER /Ninos Donabed/	**DATE CONSIDERED** 05/08/2008

EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609; Draw line through citation if not in conformance and not considered. Include copy of this form with next communication to applicant(s).

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /N.D

Search Notes	Application/Control No.	Applicant(s)/Patent Under Reexamination
	11116783	MAJOR ET AL.
	Examiner	**Art Unit**
	NINOS DONABED	2144

SEARCHED

Class	Subclass	Date	Examiner
709	246	05/07/2008	ND

SEARCH NOTES

Search Notes	Date	Examiner
See EAST Search History	05/07/2008	ND
Conducted double patenting search	05/07/2008	ND

INTERFERENCE SEARCH

Class	Subclass	Date	Examiner

Index of Claims	Application/Control No.	Applicant(s)/Patent Under Reexamination
	11116783	MAJOR ET AL.
	Examiner	**Art Unit**
	NINOS DONABED	2144

✓	Rejected	-	Cancelled	N	Non-Elected	A	Appeal
=	Allowed	÷	Restricted	I	Interference	O	Objected

☐ Claims renumbered in the same order as presented by applicant ☐ CPA ☐ T.D. ☐ R.1.47

CLAIM		DATE								
Final	Original	05/08/2008								
	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	✓								
	31	✓								
	32	✓								
	33	✓								
	34	✓								

EAST Search History

Ref #	Hits	Search Query	DBs	Default Operator	Plurals	Time Stamp
L1	0	709/246.pn.	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	ADJ	ON	2008/05/08 14:08
L2	2076	709/246.ccls.	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	ADJ	ON	2008/05/08 14:08
L3	15	2 and (dynamic near5 bandwidth)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	ADJ	ON	2008/05/08 14:08
S78	8110	(web page near5 (creat$4 or html near4 editor))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	ADJ	ON	2007/11/27 14:55
S79	148	S78 and browser control	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	ADJ	ON	2007/11/27 15:53
S80	0	S79 and web project	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	ADJ	ON	2007/11/27 15:53
S81	490	edit near5 web$1 page	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	ADJ	ON	2007/11/27 15:54
S82	14	(edit near5 web$1 page).ti.	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	ADJ	ON	2007/11/27 15:55

S83	0	S81 and (load%3 and stor$3 and perform$3)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	ADJ	ON	2007/11/27 15:56
S84	235	S81 and (load$3 and stor$3 and perform$3)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	ADJ	ON	2007/11/27 15:56
S85	1631	(edit$3 or modify$3) near5 web$1 page	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	ADJ	ON	2007/11/27 16:50
S87	2814	(edit$3 or modify$3) near5 (web page or webpage)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	ADJ	ON	2007/11/27 16:51
S88	2	S87 and template and web project	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	ADJ	ON	2007/11/27 16:51
S99	1	"10897816"	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	ADJ	ON	2007/11/28 15:20
S100	1	"10897816" and web project	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	ADJ	ON	2007/11/28 15:20
S101	1282	(creat$5 or edit$3 or updat $5) near5 ((web) near5 (page or site)) with ((real near5 time) or live or active)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/06 15:45
S102	17	((creat$5 or edit$3 or updat $5) near5 ((web) near5 (page or site)) with ((real near5 time) or live or active)).ti.	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/06 15:46

S103	116	((creat$5 or edit$3 or updat $5) near5 ((web) near5 (page or site)) with ((real near5 time) or live or active)).ti,ab.	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/06 15:49
S104	1282	((creat$5 or edit$3 or updat $5) near5 ((web) near5 (page or site)) with ((real near5 time) or live or active))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/06 15:50
S105	1067	((creat$5 or edit$3 or updat $5) near5 ((web) near5 (page or site)) near5 ((real near5 time) or live or active))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/06 15:50
S106	12	((creat$5 or edit$3 or updat $5) near5 ((web) near5 (page or site)) near5 ((real near5 time) or live or active)).ti.	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/06 15:51
S107	84	((creat$5 or edit$3 or updat $5) near5 ((web) near5 (page or site)) near5 ((real near5 time) or live or active)).ti,ab.	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/06 15:51
S108	1	((creat$5 or edit$3 or updat $5) near5 ((web) near5 (page or site)) near5 ((real near5 time) or live or active)).ti,ab. and wysiwyg	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/06 15:51
S109	1	((router or relay) near5 (soap and snmp))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/06 18:47
S110	5	((router or relay) with (soap and snmp))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/06 18:47
S111	17	((router or relay or server) near5 (soap and snmp))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/06 18:48

S112	17	S111 and (ram or storage or memory)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/06 18:51
S113	17	((router or relay or server or proxy) near5 (soap and snmp))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/06 18:51
S114	114469	((router or relay or server) near5 (storage or ram or memory))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/06 18:53
S115	11	S113 and ((router or relay or server) near5 (storage or ram or memory))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/06 18:53
S116	11	S113 and ((router or relay or server) with (storage or ram or memory))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/06 18:53
S117	0	20030103226+.pn.	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/06 18:55
S118	2	"20030103226".pn.	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/06 18:55
S119	1	S118 and (storage or memory or ram)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/06 18:55
S120	1	S118 and (storage or memory or ram or database or directory)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/06 18:55

S121	1	S118 and (storage or memory or ram or database or directory) near8 server	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/06 18:56
S122	1	"6446137".pn. and table	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/07 11:33
S123	2	"5815667".pn.	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/07 15:53
S124	0	"5815667".pn. and (less near5 time)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/07 15:54
S125	0	"5815667".pn. and (less with time)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/07 15:54
S126	1	"5815667".pn. and (retransmit$5)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/07 15:58
S127	7	"6057832" and pars$5	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/07 16:09
S128	0	"6057832.pn." and pars$5	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/07 16:09
S129	0	"6057832.pn."	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/07 16:09

S130	0	"6057832.pn"	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/07 16:09
S131	2	"6057832".pn.	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/07 16:09
S132	0	"6057832".pn. and pars$5	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/07 16:09
S133	0	"6057832".pn. and meta	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/07 16:11
S134	0	"6057832".pn. and tag	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/07 16:11
S135	0	"6057832".pn. and web	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/07 16:14
S136	0	"6057832".pn. and html	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/07 16:14
S138	53	((stream$3) near (vary$5 or chang$5 or updat$5)) with (quality near5 (packet or stream))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/07 19:09
S139	74	((stream$3) near (vary$5 or chang$5 or updat$5 or optimiz$5)) with (quality near5 (packet or stream))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/07 19:10

S140	10	((vary$5 or chang$5 or updat $5)) with ((quality) near5 (stream$5 or packet$5)) same (dynamic near5 bandwidth)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/07 19:15
S141	2	"6141053".pn.	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/07 19:16
S142	2	((adjust$5 or chang$5 or updat$5 or optimiz$4)) near5 ((quality) near5 (stream$5 or packet$5)) same (dynamic near5 bandwidth)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/07 19:18
S143	2	"6732183".pn.	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/07 19:24
S144	2	"5999979".pn.	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/07 19:26
S145	2	"5490168".pn.	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/07 19:27
S146	2	("2002/0144276").URPN.	USPAT	OR	ON	2008/05/07 19:27
S147	9	((automatic$5) near5 (chang $3 or alter$4 or incres$3 or decrease$3 or optimiz$4 or adjust$3)) near5 (quality near5 (stream))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/07 19:39
S148	4	"2004071209".pn.	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/07 19:41

S149	2	("20040071209").PN.	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2008/05/07 19:41
S150	0	("5490168.pn.").PN.	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2008/05/07 19:43
S151	2	("20040071209").PN.	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2008/05/07 19:52
S152	0	S151 and (play or forward)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/07 19:53
S153	14	((automatic$5) near5 (chang $3 or alter$4 or incres$3 or decrease$3 or optimiz$4 or adjust$3)) with (quality near5 (stream))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/07 20:13
S154	279451	((automatic$5) near5 (chang $3 or alter$4 or incres$3 or decrease$3 or optimiz$4 or adjust$3))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/07 20:14
S155	298	((automatic$5) near5 (chang $3 or alter$4 or incres$3 or decrease$3 or optimiz$4 or adjust$3)) with ((stream or packet) near5 (quality or resolution or content or rate or speed))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/07 20:15
S156	23	((automatic$5) near5 (chang $3 or alter$4 or incres$3 or decrease$3 or optimiz$4 or adjust$3)) with ((stream or packet) near5 (quality or resolution))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/07 20:15
S157	4	S156 and (play or rewind or forward)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/07 20:18

S158	0	"2006490627".pn.	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/07 20:19
S159	2	"6490627".pn.	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/07 20:19
S160	21	("5953506").PN. OR ("6490627").URPN.	US-PGPUB; USPAT; USOCR	OR	ON	2008/05/07 20:20
S161	374	((video or audio near5 (device)) same (chang$3 or alter$4 or incres$3 or decrease$3 or optimiz$4 or adjust$3)) with ((stream or packet) near5 (quality or resolution))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/07 20:21
S162	333	((video or audio near5 (device)) same (chang$3 or alter$4 or incres$3 or decrease$3 or optimiz$4 or adjust$3)) with ((stream) near5 (quality or resolution))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/07 20:21
S163	160	S162 and ((video or audio near5 (device))).ti.	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/07 20:22
S164	22	((video or audio near5 (device)) same (chang$3 or alter$4 or incres$3 or decrease$3 or optimiz$4 or adjust$3)) with (((stream or packet) near5 (quality or resolution)) near5 (bandwidth))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/07 20:23
S165	21	((video or audio near5 (device)) with (chang$3 or alter$4 or incres$3 or decrease$3 or optimiz$4 or adjust$3)) with (((stream or packet) near5 (quality or resolution)) near5 (bandwidth))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/07 20:26

S166	1	((request or ask) near5 (chang$3 or alter$4 or incres $3 or decrease$3 or optimiz $4 or adjust$3)) with (((stream or packet) near5 (quality or resolution)) near5 (bandwidth))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/07 20:27
S167	0	((audio or video) near5 (device or machine or player)) with (((request or ask) near5((chang$3 or alter $4 or incres$3 or decrease$3 or optimiz$4 or adjust$3))) with ((stream) near5 (quality or resolution)))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/07 20:29
S168	1	((audio or video) near5 (device or machine or player)).ti,ab. and (((request or ask) near5((chang$3 or alter$4 or incres$3 or decrease$3 or optimiz$4 or adjust$3))) with ((stream) near5 (quality or resolution)))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/07 20:30
S169	1	((audio or video) near5 (device or machine or player)).ti,ab. and (((request or ask) with ((chang$3 or alter$4 or incres$3 or decrease$3 or optimiz$4 or adjust$3))) with ((stream) near5 (quality or resolution)))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/07 20:31
S170	309655	((audio or video) near5 (device or machine or player))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/07 20:32
S171	15654	S170 and ((request or demand or ask) near5 (increas$3 or decreas$3 or chang$3 or updat$3 or optimiz$3 or alter$3 or adjust $3))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/07 20:33
S172	364	S170 and ((request or demand or ask) near5 ((increas$3 or decreas$3 or chang$3 or updat$3 or optimiz$3 or alter$3 or adjust $3)) near5 (quality or resolution))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/07 20:34

S173	7	S170 and ((request or demand or ask) near7 ((increas$3 or decreas$3 or chang$3 or updat$3 or optimiz$3 or alter$3 or adjust $3)) near7 ((quality or resolution)) near5 stream$3)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/07 20:34
S174	4	S170 and ((request or demand or ask) near7 ((increas$3 or decreas$3 or chang$3 or updat$3 or optimiz$3 or alter$3 or adjust $3)) near7 ((quality or resolution)) with (bandwidth and stream$3))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/07 20:37
S175	630523	((audio or video or media) near5 (device or machine or player))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/07 20:38
S176	8	S175 and ((request or demand or ask) near7 ((increas$3 or decreas$3 or chang$3 or updat$3 or optimiz$3 or alter$3 or adjust $3)) near7 ((quality or resolution)) with (bandwidth and stream$3))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/07 20:38
S177	4	S176 not S174	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/07 20:39
S178	38611	((media or video or audio) near (aparatus or player))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/07 20:40
S179	79396	((media or video or audio) near5 (aparatus or player))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/07 20:40
S180	35839	S170 and ((request or demand or ask) near7 ((increas$3 or decreas$3 or chang$3 or updat$3 or optimiz$3 or alter$3 or adjust $3)) near7 ((quality or resolution)) near5 stream$3 or packet)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/07 20:41

S181	10830	S179 and ((request or demand or ask) near7 ((increas$3 or decreas$3 or chang$3 or updat$3 or optimiz$3 or alter$3 or adjust $3)) near7 ((quality or resolution)) near5 stream$3 or packet)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/07 20:41
S182	5	S179 and ((request or demand or ask) near7 ((increas$3 or decreas$3 or chang$3 or updat$3 or optimiz$3 or alter$3 or adjust $3)) near7 ((quality or resolution)) near5 (stream$3 or packet))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/07 20:42
S183	3	S182 and (play or forward)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/07 20:47
S184	1	"6618776".pn. and media	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/07 20:49
S185	47	((media or video or audio) near5 (aparatus or playe or devicer))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/07 20:50
S186	521672	((media or video or audio) near5 (aparatus or playe or device))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/07 20:50
S187	2055	S186 and ((monitor$3 or check$3) near8 (bandwidth))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/07 20:51
S188	38	S186 with ((monitor$3 or check$3) near8 (bandwidth))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/07 20:51
S189	0	("2005/0262251").URPN.	USPAT	OR	ON	2008/05/07 20:53

S190	1404	709/230.ccls.	USPAT	OR	ON	2008/05/07 20:54
S191	41	S190 and ((media or video or audio) near5 player)	USPAT	OR	ON	2008/05/07 20:59
S192	193	S190 and ((media or video or audio) near5 (player or device)) and stream	USPAT	OR	ON	2008/05/07 20:59
S193	202	S190 and ((media or video or audio) near5 (player or device)) and stream$3	USPAT	OR	ON	2008/05/07 21:00
S194	191	S193 and @ad<="20040430"	USPAT	OR	ON	2008/05/07 21:00
S195	5	S194 and ((monitor$3 or check$3) near5 (bandwidth))	USPAT	OR	ON	2008/05/07 21:01
S196	160489	((media or video or audio) near5 (player or device))	USPAT	OR	ON	2008/05/07 21:02
S197	218221	((media or video or audio) near5 (apparatus or player or device))	USPAT	OR	ON	2008/05/07 21:03
S198	218221	((media or video or audio) near5 (apparatus or player or device)) ".ti.,ab."	USPAT	OR	ON	2008/05/07 21:03
S199	1866	S198 and ((adjust$3 or chang $3 or updat$3 or optimiz$4 or increas$4 or decrease$3)) with ((quality) near8 (stream $3 or packet$3 or data))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/07 21:05
S200	1363	S198 and ((adjust$3 or chang $3 or updat$3 or optimiz$4 or increas$4 or decrease$3)) with ((quality) near5 (stream $3 or packet$3 or data))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/07 21:05
S201	0	S198 and (((adjust$3 or chang$3 or updat$3 or optimiz$4 or increas$4 or decrease$3)) with ((quality) near5 (stream$3 or packet$3 or data))) same (bandwidht)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/07 21:05
S202	172	S198 and (((adjust$3 or chang$3 or updat$3 or optimiz$4 or increas$4 or decrease$3)) with ((quality) near5 (stream$3 or packet$3 or data))) same (bandwidth)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/07 21:06
S203	3	S202 and (play and forward and rewind)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/07 21:07

S204	2	"20020144276".pn.	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/07 21:21
S205	1	"20020144276".pn. and http	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/07 21:21
S206	0	"20020144276".pn. and (tcp or ip)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/07 21:22
S207	1	"20020144276".pn. and request	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/07 21:24
S208	0	"20020144276".pn. and monitor	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/05/07 21:25

5/8/2008 2:20:04 PM
C:\Documents and Settings\ndonabed\My Documents\EAST\Workspaces\11116783.pn..wsp

Notice of References Cited	Application/Control No. 11/116,783	Applicant(s)/Patent Under Reexamination MAJOR ET AL.	
	Examiner NINOS DONABED	Art Unit 2144	Page 1 of 1

U.S. PATENT DOCUMENTS

*		Document Number Country Code-Number-Kind Code	Date MM-YYYY	Name	Classification
*	A	US-2002/0144276	10-2002	Radford et al.	725/87
*	B	US-7,096,271	08-2006	Omoigui et al.	709/231
*	C	US-2004/0071209	04-2004	Burg et al.	375/240.01
*	D	US-6,490,627	12-2002	Kalra et al.	709/231
	E	US-			
	F	US-			
	G	US-			
	H	US-			
	I	US-			
	J	US-			
	K	US-			
	L	US-			
	M	US-			

FOREIGN PATENT DOCUMENTS

*		Document Number Country Code-Number-Kind Code	Date MM-YYYY	Country	Name	Classification
	N					
	O					
	P					
	Q					
	R					
	S					
	T					

NON-PATENT DOCUMENTS

*		Include as applicable: Author, Title Date, Publisher, Edition or Volume, Pertinent Pages)
	U	
	V	
	W	
	X	

*A copy of this reference is not being furnished with this Office action. (See MPEP § 707.05(a).)
Dates in MM-YYYY format are publication dates. Classifications may be US or foreign.

PTO/SB/08a (05-07)
Approved for use through 11/30/2007. OMB 0651-0031
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE
Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it contains a valid OMB control number.

INFORMATION DISCLOSURE STATEMENT BY APPLICANT

(Not for submission under 37 CFR 1.99)

Application Number	11116783
Filing Date	2005-04-28
First Named Inventor	R. Drew Major
Art Unit	2143
Examiner Name	
Attorney Docket Number	2082.2.1

U.S.PATENTS Remove

Examiner Initial*	Cite No	Patent Number	Kind Code[1]	Issue Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1	4535355		1985-08-13	Arn et al.	
	2	5953506		1999-09-14	Kalra et al.	
	3	6490627	B1	2002-12-03	Kalra et al.	
	4	6574591	B1	2003-06-03	Kleiman et al.	
	5	6604118	B2	2003-08-05	Kleiman et al.	

If you wish to add additional U.S. Patent citation information please click the Add button. Add

U.S.PATENT APPLICATION PUBLICATIONS Remove

Examiner Initial*	Cite No	Publication Number	Kind Code[1]	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1					

If you wish to add additional U.S. Published Application citation information please click the Add button. Add

FOREIGN PATENT DOCUMENTS Remove

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /N.D

INFORMATION DISCLOSURE STATEMENT BY APPLICANT

(Not for submission under 37 CFR 1.99)

Application Number	11116783
Filing Date	2005-04-28
First Named Inventor	R. Drew Major
Art Unit	2143
Examiner Name	
Attorney Docket Number	2082.2.1

Examiner Initial*	Cite No	Foreign Document Number[3]	Country Code[2] ¡	Kind Code[4]	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear	T[5]
	1							☐

If you wish to add additional Foreign Patent Document citation information please click the Add button Add

NON-PATENT LITERATURE DOCUMENTS Remove

Examiner Initials*	Cite No	Include name of the author (in CAPITAL LETTERS), title of the article (when appropriate), title of the item (book, magazine, journal, serial, symposium, catalog, etc), date, pages(s), volume-issue number(s), publisher, city and/or country where published.	T[5]
	1		☐

If you wish to add additional non-patent literature document citation information please click the Add button Add

EXAMINER SIGNATURE

Examiner Signature	/Ninos Donabed/	Date Considered	05/08/2008

*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609. Draw line through a citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

[1] See Kind Codes of USPTO Patent Documents at www.USPTO.GOV or MPEP 901.04. [2] Enter office that issued the document, by the two-letter code (WIPO Standard ST.3). [3] For Japanese patent documents, the indication of the year of the reign of the Emperor must precede the serial number of the patent document. [4] Kind of document by the appropriate symbols as indicated on the document under WIPO Standard ST.16 if possible. [5] Applicant is to place a check mark here if English language translation is attached.

EFS Web 2.0.2

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /N.D

REVOCATION OF POWER OF ATTORNEY WITH NEW POWER OF ATTORNEY AND CHANGE OF CORRESPONDENCE ADDRESS		
	Application Number	11/116,783
	Filing Date	April 28, 2005
	First Named Inventor:	R. Drew Major, et al.
	Art Unit	2144
	Examiner Name	Donabed, Nonos J.
	Attorney Docket Number	8532P001

I hereby revoke all previous powers of attorney given in the above-identified application.

☐ A Power of Attorney is submitted herewith

OR

☒ I hereby appoint the practitioners associated with the Customer Number: 08791

☒ Please change the correspondence address for the above-identified application to:

☒ The address associated with Customer Number: 08791

OR

☐ Firm or Individual Name	Daniel M. De Vos Blakeley, Sokoloff, Taylor & Zafman LLP				
Address	1279 Oakmead Parkway				
City	Sunnyvale	State	California	Zip	94085-4040
Country	U.S.A.				
Telephone	(408) 720-8300	Email	Dan_DeVos@bstz.com		

I am the:

☐ Applicant/Inventor.

☒ Assignee of record of the entire interest. See 37 C.F.R. 3.71. *Statement under 37 CFR 3.73(b) is enclosed. (Form PTO/SB/96)*

See Exhibit A

SIGNATURE of Applicant or Assignee of Record

Signature	RRS		
Name	Robin Stenerson		
Date	8/6/08	Telephone	(801) 216-8815

NOTE: Signatures of all the inventors or assignees of record of the entire interest or their representative(s) are required. Submit multiple forms if more than one signature is required. See below*.

☐ *Total of _____ forms are submitted

SEND TO: Commissioner for Patents, P.O. Box 1450, Alexandria, Virginia 22313-1450.

Based on Form PTO/SB/82 (01-06) as modified by BLAKELY, SOKOLOFF, TAYLOR & ZAFMAN LLP on 2/20/07.

Exhibit A

Statement Under 37 CFR § 3.73(b)

Applicant/Patent Owner: Move Networks, Inc.

Application No./Patent No.: 11/116,783 Filed/Issue Date: April 28, 2005

Entitled: Apparatus, System and Method for Adaptive-Rate Shifting of Streaming Content

Move Networks, Inc., a Corporation
(Name of Assignee) (Type of Assignee, e.g., corporation, partnership, university, government agency, etc.)

states that it is:

1. [X] the assignee of the entire right, title and interest; or

2. [ ] an assignee of less than the entire right, title and interest.
(The extent (by percentage) of its ownership interest is _____%)

in the patent application/patent identified above by virtue of either:

A. [X] An assignment from the inventor(s) of the patent application/patent identified above.
The assignment was recorded in the United States Patent and Trademark Office at
Reel 018020, Frame 0750 or for which a copy thereof is attached.

OR

B. [ ] A chain of title from the inventor(s), of the patent application/patent identified above, to the current assignee as shown below:

1. From: _____ To: _____
The document was recorded in the United States Patent and Trademark Office at
Reel _____, Frame _____, or for which a copy thereof is attached.

2. From: _____ To: _____
The document was recorded in the United States Patent and Trademark Office at
Reel _____, Frame _____, or for which a copy thereof is attached.

3. From: _____ To: _____
The document was recorded in the United States Patent and Trademark Office at
Reel _____, Frame _____, or for which a copy thereof is attached.

[ ] Additional documents in the chain of title are listed on a supplemental sheet.

[ ] As required by 37 CFR 3.73(b)(1)(i), the documentary evidence of the chain of title from the original owner to the assignee was, or concurrently is being, submitted for recordation pursuant to 37 CFR 3.11.
[NOTE: A separate copy (*i.e.*, a true copy of the original assignment document(s)) must be submitted to Assignment Division in accordance with 37 CFR Part 3, to record the assignment in the records of the USPTO. See MPEP 302.08]

The undersigned (whose title is supplied below) is authorized to act on behalf of the assignee.

RLSC 8/6/08
Signature Date

Robin Stenerson (801) 216-8815
Printed or Typed Name Telephone Number

General Counsel
Title

Based on Form PTO/SB/82 (06-06) as modified by BLAKELY, SOKOLOFF, TAYLOR & ZAFMAN LLP on 2/20/07.

Electronic Acknowledgement Receipt

EFS ID:	3791897
Application Number:	11116783
International Application Number:	
Confirmation Number:	5535
Title of Invention:	Apparatus, system, and method for adaptive-rate shifting of streaming content
First Named Inventor/Applicant Name:	R. Drew Major
Customer Number:	36491
Filer:	Daniel De Vos
Filer Authorized By:	
Attorney Docket Number:	2082.2.1
Receipt Date:	15-AUG-2008
Filing Date:	28-APR-2005
Time Stamp:	21:29:51
Application Type:	Utility under 35 USC 111(a)

Payment information:

Submitted with Payment	no

File Listing:

Document Number	Document Description	File Name	File Size(Bytes) /Message Digest	Multi Part /.zip	Pages (if appl.)
1	Power of Attorney	8532P001_Rev_POA.pdf	116306 90d7f57747781a59c8624dc0f0e4829e48237cb6	no	2

Warnings:

Information:

Total Files Size (in bytes):	116306

This Acknowledgement Receipt evidences receipt on the noted date by the USPTO of the indicated documents, characterized by the applicant, and including page counts, where applicable. It serves as evidence of receipt similar to a Post Card, as described in MPEP 503.

New Applications Under 35 U.S.C. 111
If a new application is being filed and the application includes the necessary components for a filing date (see 37 CFR 1.53(b)-(d) and MPEP 506), a Filing Receipt (37 CFR 1.54) will be issued in due course and the date shown on this Acknowledgement Receipt will establish the filing date of the application.

National Stage of an International Application under 35 U.S.C. 371
If a timely submission to enter the national stage of an international application is compliant with the conditions of 35 U.S.C. 371 and other applicable requirements a Form PCT/DO/EO/903 indicating acceptance of the application as a national stage submission under 35 U.S.C. 371 will be issued in addition to the Filing Receipt, in due course.

New International Application Filed with the USPTO as a Receiving Office
If a new international application is being filed and the international application includes the necessary components for an international filing date (see PCT Article 11 and MPEP 1810), a Notification of the International Application Number and of the International Filing Date (Form PCT/RO/105) will be issued in due course, subject to prescriptions concerning national security, and the date shown on this Acknowledgement Receipt will establish the international filing date of the application.

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NUMBER	FILING OR 371(C) DATE	FIRST NAMED APPLICANT	ATTY. DOCKET NO./TITLE
11/116,783	04/28/2005	R. Drew Major	2082.2.1

CONFIRMATION NO. 5535

POWER OF ATTORNEY NOTICE

36491
Kunzler & McKenzie
8 EAST BROADWAY
SUITE 600
SALT LAKE CITY, UT 84111

OC000000031745335

Date Mailed: 08/26/2008

NOTICE REGARDING CHANGE OF POWER OF ATTORNEY

This is in response to the Power of Attorney filed 08/15/2008.

- The Power of Attorney to you in this application has been revoked by the assignee who has intervened as provided by 37 CFR 3.71. Future correspondence will be mailed to the new address of record(37 CFR 1.33).

/sibrahim/

Office of Data Management, Application Assistance Unit (571) 272-4000, or (571) 272-4200, or 1-888-786-0101

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NUMBER	FILING OR 371(C) DATE	FIRST NAMED APPLICANT	ATTY. DOCKET NO./TITLE
11/116,783	04/28/2005	R. Drew Major	8532P001

CONFIRMATION NO. 5535

POA ACCEPTANCE LETTER

8791
BLAKELY SOKOLOFF TAYLOR & ZAFMAN LLP
1279 OAKMEAD PARKWAY
SUNNYVALE, CA 94085-4040

OC000000031745362

Date Mailed: 08/26/2008

NOTICE OF ACCEPTANCE OF POWER OF ATTORNEY

This is in response to the Power of Attorney filed 08/15/2008.

The Power of Attorney in this application is accepted. Correspondence in this application will be mailed to the above address as provided by 37 CFR 1.33.

/sibrahim/

Office of Data Management, Application Assistance Unit (571) 272-4000, or (571) 272-4200, or 1-888-786-0101

PTO/SB/06 (07-06)
Approved for use through 1/31/2007. OMB 0651-0032
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE
Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.

PATENT APPLICATION FEE DETERMINATION RECORD

Substitute for Form PTO-875

Application or Docket Number	Filing Date	
11/116,783	04/28/2005	☐ To be Mailed

APPLICATION AS FILED – PART I

FOR	(Column 1) NUMBER FILED	(Column 2) NUMBER EXTRA	SMALL ENTITY ☒ RATE ($)	FEE ($)	OR	OTHER THAN SMALL ENTITY RATE ($)	FEE ($)
☐ BASIC FEE (37 CFR 1.16(a), (b), or (c))	N/A	N/A	N/A			N/A	
☐ SEARCH FEE (37 CFR 1.16(k), (i), or (m))	N/A	N/A	N/A			N/A	
☐ EXAMINATION FEE (37 CFR 1.16(o), (p), or (q))	N/A	N/A	N/A			N/A	
TOTAL CLAIMS (37 CFR 1.16(i))	minus 20 =	*	X $ =		OR	X $ =	
INDEPENDENT CLAIMS (37 CFR 1.16(h))	minus 3 =	*	X $ =			X $ =	
☐ APPLICATION SIZE FEE (37 CFR 1.16(s))	If the specification and drawings exceed 100 sheets of paper, the application size fee due is $250 ($125 for small entity) for each additional 50 sheets or fraction thereof. See 35 U.S.C. 41(a)(1)(G) and 37 CFR 1.16(s).						
☐ MULTIPLE DEPENDENT CLAIM PRESENT (37 CFR 1.16(j))							
* If the difference in column 1 is less than zero, enter "0" in column 2.			TOTAL			TOTAL	

APPLICATION AS AMENDED – PART II

AMENDMENT	09/11/2008	(Column 1) CLAIMS REMAINING AFTER AMENDMENT		(Column 2) HIGHEST NUMBER PREVIOUSLY PAID FOR	(Column 3) PRESENT EXTRA	SMALL ENTITY RATE ($)	ADDITIONAL FEE ($)	OR	OTHER THAN SMALL ENTITY RATE ($)	ADDITIONAL FEE ($)
	Total (37 CFR 1.16(i))	* 18	Minus	** 34	= 0	X $25 =	0	OR	X $ =	
	Independent (37 CFR 1.16(h))	* 5	Minus	***5	= 0	X $105 =	0	OR	X $ =	
	☐ Application Size Fee (37 CFR 1.16(s))									
	☐ FIRST PRESENTATION OF MULTIPLE DEPENDENT CLAIM (37 CFR 1.16(j))							OR		
						TOTAL ADD'L FEE	0	OR	TOTAL ADD'L FEE	

AMENDMENT		(Column 1) CLAIMS REMAINING AFTER AMENDMENT		(Column 2) HIGHEST NUMBER PREVIOUSLY PAID FOR	(Column 3) PRESENT EXTRA	RATE ($)	ADDITIONAL FEE ($)		RATE ($)	ADDITIONAL FEE ($)
	Total (37 CFR 1.16(i))	*	Minus	**	=	X $ =		OR	X $ =	
	Independent (37 CFR 1.16(h))	*	Minus	***	=	X $ =		OR	X $ =	
	☐ Application Size Fee (37 CFR 1.16(s))									
	☐ FIRST PRESENTATION OF MULTIPLE DEPENDENT CLAIM (37 CFR 1.16(j))							OR		
						TOTAL ADD'L FEE		OR	TOTAL ADD'L FEE	

* If the entry in column 1 is less than the entry in column 2, write "0" in column 3.
** If the "Highest Number Previously Paid For" IN THIS SPACE is less than 20, enter "20".
*** If the "Highest Number Previously Paid For" IN THIS SPACE is less than 3, enter "3".
The "Highest Number Previously Paid For" (Total or Independent) is the highest number found in the appropriate box in column 1.

Legal Instrument Examiner:
/DEBRA R. WYATT/

This collection of information is required by 37 CFR 1.16. The information is required to obtain or retain a benefit by the public which is to file (and by the USPTO to process) an application. Confidentiality is governed by 35 U.S.C. 122 and 37 CFR 1.14. This collection is estimated to take 12 minutes to complete, including gathering, preparing, and submitting the completed application form to the USPTO. Time will vary depending upon the individual case. Any comments on the amount of time you require to complete this form and/or suggestions for reducing this burden, should be sent to the Chief Information Officer, U.S. Patent and Trademark Office, U.S. Department of Commerce, P.O. Box 1450, Alexandria, VA 22313-1450. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. **SEND TO: Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.**

If you need assistance in completing the form, call 1-800-PTO-9199 and select option 2.

Amendments to the Claims:

This listing of claims will replace all prior versions, and listings, of the claims in the application:

Listing of Claims:

1-19. (Cancelled)

20. (Currently amended) A method for presenting rate-adaptive streams, the method comprising:

streaming by a media player operating on an end user station a video from a server, wherein the video has a playback duration, wherein different copies of the video are stored on the server, wherein each of the different copies is encoded and compressed according to a different bit rate and includes the entire playback duration of the video, wherein the different copies are divided into portions whose playback is the same portion of the video for each of the different copies, wherein corresponding portions of the different copies have the same time index in relation to the beginning of the video, and wherein the streaming comprises:

~~simultaneously~~ requesting ~~at least portions of~~ a plurality of ~~streamlets~~ sequential portions of a first of the copies encoded and compressed at a first of the bit rates from the server over a plurality of Transmission Control Protocol (TCP) connections;

~~continuously monitoring the progress of subsequent responses to streamlet requests, and in accordance with the progress of the responses,~~ automatically and progressively requesting over the plurality of TCP connections subsequent portions of the video by requesting from different ones of the copies dependent upon successive determinations to shift the playback quality to a higher or lower

quality ~~streamlets~~one of the different copies, said automatically and progressively requesting including,

regularly generating performance factors indicative of the current ability to sustain the streaming of the video using different ones of the copies; and

making the successive determinations to shift the playback quality based on the performance factors to achieve continuous playback of the video using portions of the highest quality one of the copies determined sustainable at that time; and

~~staging~~presenting the ~~streamlets~~video to the user by playing back with ~~and arranging the streamlets for playback on a content~~the media player the requested portions from the different copies in order of ascending playback time.

21. (Original) The method of claim 20, further comprising establishing ~~multiple~~ the plurality of ~~Transmission Control Protocol (~~TCP~~)~~ connections with the server, ~~and requesting streamlets of varying bitrates~~, wherein said requesting the plurality of sequential portions of the first of the copies includes requesting sub-parts of the portions over different ones of the plurality of TCP connections, and wherein said method further includes reassembling the portions from the received sub-parts.

22. (Currently amended) The method of claim 20, ~~further comprising~~ wherein said generating ~~a~~the performance ~~factor~~factors ~~according to responses from streamlet requests~~comprises:

calculating a current performance ratio across a current window of samples of a playback length; and

calculating a current average to obtain a current one of the performance factors;

said making further comprises:

comparing the current performance factor with a set of one or more trigger thresholds; and

determining whether to upshift or downshift to a higher or lower quality one of the different copies based on said comparing.

23. (Currently amended) The method of claim 20, ~~further comprising~~wherein said determining whether to upshift or downshift comprises:

upshifting to a higher quality ~~streamlet~~one of the different copies when the performance factor is greater than a first selected threshold[[,]]; and

determining if the higher quality playback can be sustained.

24. (Currently amended) The method of claim [[20]] 23, ~~further comprising~~wherein said determining whether to upshift or downshift comprises

downshifting to a lower quality ~~streamlet~~ one of the different copies when the performance factor is less than a second selected threshold.

25. (Currently amended) The method of claim 20, further comprising:

~~anticipating streamlet requests; and~~

~~pre-requesting streamlets~~ receiving user input to enable fast-forward, skip randomly, and rewind functionality; and

requesting portions of the video at a specified time index that is not the next sequential time in the current playback.

26. (Currently amended) The method of claim [[20]] 22, further comprising:

initially requesting portions of video of a low quality streamletsone of the different copies to enable instant playback of [[a]] the video ~~content file~~[[,]] ; and

subsequent upshifting to a higher quality one of the different copies according to the performance factor after initially requesting portions of the video of the low quality.

27-34. (Cancelled)

35. (New) The method of claim 20, further comprising establishing the plurality of TCP connections with the server, wherein said requesting the plurality of sequential portions of the first of the copies includes requesting the plurality of sequential portions over different ones of the plurality of TCP connections.

36. (New) The method of claim 20, further comprising sequentially arranging the received portions from the different copies in the order of ascending playback time for playback on the media player.

37. (New) The method of claim 20, wherein said automatically and progressively requesting includes requesting sub-parts of the portions over different ones of the plurality of TCP connections, and wherein said method further includes reassembling the portions from the received sub-parts.

38. (New) The method of claim 20, wherein the performance factors are indicative of the available bandwidth of the plurality of TCP connections and a current read ahead margin, wherein the current read ahead margin is a current amount of contiguous, sequentially arranged, un-played portions of the video for playback on the media player.

39. (New) The method of claim 38, wherein the performance factors are indicative of a minimum safety margin, wherein the minimum safety margin is a desired

amount of contiguously available sequentially arranged portions of video for playback on the media player.

40. (New) The method of claim 20, wherein the performance factors are indicative of latency of the requested portions, wherein the latency is a time measure between when a request is issued and a time that response data of the requested portion begins to arrive at the end user station.

41. (New) The method of claim 20, wherein the performance factors are indicative of time intervals between successive receive times for each response to the requested portions.

42. (New) The method of claim 20, wherein the performance factors are indicative of delays or losses in one or more of the plurality of TCP connections.

43. (New) The method of claim 20, wherein the server is a web server, and wherein the divided portions are stored in separate files on the web server.

44. (New) The method of claim 20, wherein the separate files are requested from the web server using Hyper Text Transfer Protocol (HTTP), and wherein the web server is without server-side intelligence to retrieve the requested portions.

45. (New) The method of claim 20, wherein the server is a cache server of a network infrastructure operator, and wherein the divided portions are stored in separate files on the cache server.

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In Re Patent Application of:

R. Drew Major

Application No.: 11/116,783

Filed: April 28, 2005

Art unit: 2144

Examiner: Donabed, Ninos J

For: APPARATUS, SYSTEM AND METHOD FOR ADAPTIVE-RATE SHIFTING OF STREAMING CONTENT

Confirmation No.: 5535

CERTIFICATE OF TRANSMISSION
I hereby certify that this correspondence is being submitted electronically via EFS Web on the date shown below.

/Kevin O. Grange/ Sept. 11, 2008

Kevin O. Grange ***Date***

Mail Stop Amendment
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

AMENDMENT AND RESPONSE TO OFFICE ACTION

Examiner:

In response to the Office Action mailed on May 14, 2008, Applicant respectfully requests that the Examiner enter the following amendments and consider the following remarks.

Amendments to the Specification being on page 2 of this paper.

Amendments to the Claims are reflected in the listing of claims that begins on page 4 of this paper.

Remarks begin on page 9 of this paper.

REMARKS

Applicant respectfully requests reconsideration of this application in view of the following remarks. For the Examiner's convenience and reference, Applicant's remarks are presented in substantially the same order in which the corresponding issues were raised in the Office Action.

Interview Summary for Examiner Interview Conducted 9/11/2008

Applicant would like to thank the Examiner for the Examiner Interview. During the Interview, Applicant's representative presented a brief overview of the invention and the proposed amended claims. The art cited in the most recent Office Action was also discussed by Applicant's representative and the Examiner. Although there was no agreement that the proposed amendments would be allowable over the cited art, it was agreed that the amended claims appear to overcome the rejections of record, subject to additional review by the Examiner upon filing of the formal amendment.

Status of the Claims

Claims 20-26 and 35-45 are pending. Claims 1-19 and 27-34 are cancelled without prejudice. Claims 20-26 are currently amended. The claim amendments are supported by the specification. Claims 35-45 are added. The new claims are supported by the specification.

Summary of the Office Action

Claims 1-10 and 34 stand rejected under 35 U.S.C. § 101 as being directed to non-statutory subject matter. Further regarding claims 27-33 stand rejected under 35 U.S.C. § 101 as being directed to non-statutory subject matter.

Claims 1-34 stand rejected under 35 U.S.C. § 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which Applicant regards as his invention.

Claims 1, 3, 9-11, 13-20, 22-27, and 29-34 stand rejected under 35 U.S.C. § 102(b) as allegedly being anticipated by U.S. Patent Application Publication No. 2002/0144276 to Radford et al. (hereinafter "Radford")

Claims 2, 4-8, 12, 21, and 28 stand rejected under 35 U.S.C. § 103(a) as allegedly being unpatentable over Radford.

Response to Rejections of Cancelled Claims 1-19, 27-33, and 34

Applicant respectfully submits that claims 1-19, 27-33, and 34 have been cancelled. Applicant does not concede that the Applicant is not entitled to the cancelled claims over the rejections under 35 U.S.C. § 101, § 112, § 102(b), and § 103(a), and reserves the right to file the cancelled claims in a continuation application.

Since claims 1-10, 27-33, and 34 have been canceled, the rejection of claims 1-10, 27-33, and 34 under 35 U.S.C. § 101 are moot. Accordingly, Applicant respectfully request the rejection of the claims 1-10, 27-33, and 34 under 35 U.S.C. § 101 be withdrawn. Similarly, since claims 1-19, 27-33, and 34 have been cancelled, the rejections of claims 1-19, 27-33, and 34 under 35 U.S.C. § 112, second paragraph, claims 1, 3, 9-11, 13-19, 27, and 29-34 under 35 U.S.C. § 102(b), and claims 2, 4-8, 12, and 28 under 35 U.S.C. § 103(a) are moot. Accordingly, Applicant respectfully request that the rejections of claims 1-19, 27-33, and 34 be withdrawn.

Response to Rejections under 35 U.S.C. § 112, second paragraph

The Office Action rejected claims 20-26 under 35 U.S.C. § 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which Applicant regards as his invention. Applicant respectfully submits that the amendment to claim 20, removing "simultaneously," overcomes the rejection of claims 20-26, and requests that the rejections of claims 20-26 under 35 U.S.C. § 112, second paragraph be withdrawn.

Response to Rejections under 35 U.S.C. § 102(b)

The Office Action rejected claims 20 and 22-26 under 35 U.S.C. § 102(b) as allegedly being anticipated by Radford. Applicant respectfully submits that claims 20-26 are patentable over the cited reference because Radford does not disclose all of the limitations of the claim. In particular, Applicant respectfully submits that claim 20 is patentable over Radford because Radford fails to disclose the following reasons:

1) Automatic and Progressively Requesting Subsequent Portions

Applicant respectfully submits that Radford discloses manually controlling the quality of the video being delivered by *allowing the user to manually select* one of the multiple content files (e.g., high quality video, low quality video, slide show presentation, audio only) by clicking on one of the buttons on the user interface. Radford, page 4, paragraphs 29-31, Figure 2, elements 230-250. In contrast, claim 20 recites "automatically and progressively requesting … portions of the video by requesting from different ones of the copies". As such, Radford fails to disclose at least this limitation.

2) Anticipating a Poor User Experience for Continuous Playback

Applicant respectfully submits that Radford discloses manually selecting the quality of the streamed data *in response to a poor viewing experience* (e.g., the streamed data deteriorates or is marred by interruptions). Radford, page 4, paragraphs 29 and 31. In contrast, claim 20 recites, as part of "automatically and progressively requesting," regularly generating performance factors indicative of the current ability to sustain the streaming of the video using different ones of the copies, and making the successive determinations to shift the playback quality based on the performance factors **to achieve continuous playback of the video** using portions of the highest quality one of the copies determined sustainable at that time". As such, Radford fails to disclose at least this limitation.

3) Requesting Portions of Video

Applicant respectfully submits that Radford discloses requesting *entire* content files. For example, Radford discloses requesting an initial streamed content file, and allowing the user to subsequently select an entirely different content file (e.g., slide show file, audio-only file, low-quality video file). In response to the user action, client device generates a second request for the initiation of delivery of a new content file in its entirety to the client device. Radford, page 4, paragraphs 29-31. In contrast, claim 20 recites that "the different copies are divided into portions whose playback is the same portion of the video for each of the different copies, wherein corresponding portions of the different

copies have the same time index in relation to the beginning of the video", and the operation of "requesting a plurality of sequential portions of a first of the copies encoded and compressed at a first of the bit rates from the server", and "automatically and progressively requesting … portions of the video by requesting from different ones of the copies". As such, Radford fails to disclose at least these limitations.

4) Multiple TCP connections

Applicant respectfully submits that Radford discloses a *single* network connection. Radford, page 2, paragraph 11. In contrast, claim 20 recites requesting the portions "over a plurality of Transmission Control Protocol (TCP) connections". As such, Radford fails to disclose at least this limitation.

In sum, given that the cited reference fails to disclose all of the limitations of the claim, Applicant respectfully submits that claim 20 is patentable over the cited reference. Accordingly, Applicant requests that the rejection of claim 20 under 35 U.S.C. § 102(b) be withdrawn.

Given that claims 21-26 and 35-45 depend from independent claim 20, which is patentable over the cited reference, Applicant respectfully submits that dependent claims 21-26 and 35-45 are also patentable over the cited reference. Accordingly, Applicant requests that the rejection of claims 22-26 under 35 U.S.C. § 102(b) and the rejection of claim 21 under 35 U.S.C. § 103(a) be withdrawn.

CONCLUSION

It is respectfully submitted that in view of the amendments and remarks set forth herein, the rejections have been overcome. If the Examiner believes a telephone interview would expedite the prosecution of this application, the Examiner is invited to contact Kevin Grange at (408) 720-8300.

If there are any additional charges, please charge them to Deposit Account No. 02-2666.

Respectfully submitted,

BLAKELY, SOKOLOFF, TAYLOR & ZAFMAN LLP

Date: Sept. 11, 2008

/Kevin O. Grange/

Kevin O. Grange
Reg. No. 60,793

1279 Oakmead Parkway
Sunnyvale, CA 94085-4040
(408) 720-8300

AMENDMENTS

Amendments to the Specification:

Please amend the following paragraph as follows:

[0066] Figure 5 is a schematic flow chart diagram illustrating one embodiment of a method 500 for processing content in accordance with the present invention. In one embodiment the method 500 starts 502, and the content module 112 receives 504 content from the publisher 110. Receiving content 504 may comprise receiving 504 a digital copy of the content file 200, or digitizing a physical copy of the content file 200. Alternatively, receiving 504 content may comprise capturing a radio or television broadcast. Once received 504, the stream module 302 generates 506 a plurality of streams 202, each stream 202 having a different quality. The quality may be predefined, or automatically set according to end user bandwidth, or in response to pre-designated publisher guidelines.

Please amend the following paragraph as follows:

[0068] Figure 6 is a schematic flow chart diagram illustrating one embodiment of a method 600 for viewing a plurality of streamlets in accordance with the present invention. The method 600 starts and an agent ~~control~~controller module 402 is provided 604 and associated with a viewer 408 and provided with a staging module 409. The agent controller module 402 then requests 606 a streamlet from the streamlet cache module 404. Alternatively, the agent controller module 402 may simultaneously request 606 a plurality of streamlets from the streamlet cache module 404. If the streamlet is stored 608 locally in the streamlet cache 410, the streamlet cache module 404 retrieves 610 the streamlet and sends the streamlet to the agent controller module 402. Upon retrieving 610 or receiving a streamlet, the agent controller module 402 makes 611 a determination of whether or not to shift to a higher or lower quality stream 202. This determination will be described below in greater detail with reference to Figure 7.

Please amend the following paragraph as follows:

[0072] Due to multiple simultaneous streamlet processing, and in order to better judge the central tendency of the performance ratio *r*, the agent ~~control~~controller module 402 may calculate a geometric mean, or alternatively an equivalent averaging algorithm, across a window of size *m*, and obtain a performance factor φ:

$$\varphi_{current} = \left(\prod_{j=1}^{m} r_j \right)^{\frac{1}{m}}.$$

[0073] The policy determination about whether or not to upshift 710 playback quality begins by comparing $\varphi_{current}$ with a trigger threshold Θ_{up}. If $\varphi_{current} \geq \Theta_{up}$, then an up shift to the next higher quality stream may be considered 716. In one embodiment, the trigger threshold Θ_{up} is determined by a combination of factors relating to the current read ahead margin (i.e. the amount of contiguously available streamlets that have been sequentially arranged by the staging module 409 for presentation at the current playback time index), and a minimum safety margin. In one embodiment, the minimum safety margin may be 24 seconds. The smaller the read ahead margin, the larger Θ_{up} is to discourage upshifting until a larger read ahead margin may be established to withstand network disruptions. If the agent controller module 402 is able to sustain 716 upshift quality, then the agent controller module 402 will upshift 717 the quality and subsequently request higher quality streams. The determination of whether use of the higher quality stream is sustainable 716 is made by comparing an estimate of the higher quality stream's performance factor, φ_{higher}, with Θ_{up}. If $\varphi_{higher} \geq \Theta_{up}$ then use of the higher quality stream is considered sustainable. If the decision of whether or not the higher stream rate is sustainable 716 is "no," the agent ~~control~~controller module 402 will not attempt to upshift 717 stream quality. If the end of the stream has been reached 714, the method 618 ends 716.

Electronic Patent Application Fee Transmittal

Application Number:	11116783
Filing Date:	28-Apr-2005
Title of Invention:	Apparatus, system, and method for adaptive-rate shifting of streaming content
First Named Inventor/Applicant Name:	R. Drew Major
Filer:	Kevin Grange/Leslie Rogan
Attorney Docket Number:	8532P001

Filed as Small Entity

Utility under 35 USC 111(a) Filing Fees

Description	Fee Code	Quantity	Amount	Sub-Total in USD($)
Basic Filing:				
Pages:				
Claims:				
Miscellaneous-Filing:				
Petition:				
Patent-Appeals-and-Interference:				
Post-Allowance-and-Post-Issuance:				
Extension-of-Time:				
Extension - 1 month with $0 paid	2251	1	60	60

Description	Fee Code	Quantity	Amount	Sub-Total in USD($)
Miscellaneous:				
Submission- Information Disclosure Stmt	1806	1	180	180
		Total in USD ($)		**240**

Substitute for Form 1449/PTO

INFORMATION DISCLOSURE STATEMENT BY APPLICANT

(use as many sheets as necessary)

Complete if Known	
Application Number:	11/116,783
Filing Date:	April 28, 2005
First Named Inventor:	R. Drew Major
Art Unit:	2144
Examiner Name:	Donabed, Ninos J
Attorney Docket Number:	8532.P001

Sheet	**1**	**of**	**2**

U.S. PATENT DOCUMENTS

Examiner Initials*	Cite No.[1]	Document Number Number-Kind Code[2] (If known)		Publication Date MM-DD-YYYY	Name of Patentee or Applicant of Cited Document	Pages, Columns, Lines, Where Relevant Passages or Relevant Figures Appear
		US-	6,091,775	07-18-2000	Hibi et al.	
		US-	6,091,777	07-18-2000	Guetz et al.	
		US-	6,510,553 B1	01-21-2003	Hazra	
		US-	6,731,600 B1	05-04-2004	Patel et al.	
		US-	6,732,183 B1	05-04-2004	Graham	
		US-	6,968,387 B2	11-22-2005	Lanphear	
		US-	7,075,986 B2	07-11-2006	Girod et al.	
		US-	7,349,976 B1	03-25-2008	Glaser et al.	
		US-	2002/0073167 A1	06-13-2002	Powell et al.	
		US-	2005/0123058 A1	06-09-2005	Greenbaum et al.	
		US-	2007/0024705 A1	02-01-2007	Richter et al.	
		US-	2008/0028428 A1	01-31-2008	Jeong et al.	
		US-	2008/0056373 A1	03-06-2008	Newlin et al.	
		US-	2008/0195744 A1	08-14-2008	Bowra et al.	
		US-	2008/0195745 A1	08-14-2008	Bowra et al.	
		US-				
		US-				
		US-				
		US-				
		US-				
		US-				
		US-				

Examiner Signature		Date Considered	

*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609; Draw line through citation if not in conformance and not considered. Include copy of this form with next communication to applicant. [1]Applicant's unique citation designation number (optional). [2]See Kinds Codes of USPTO Patent Documents at www.uspto.gov or MPEP 901.04. [3]Enter Office that issued the document, by the two-letter code (WIPO Standard ST.3). [4]For Japanese patent documents, the indication of the year of reign of the Emperor must precede the serial number of the patent document. [5]Kind of document by the appropriate symbols as indicated on the document under WIPO Standard ST. 16 if possible. [6]Applicant is to place a check mark here if English language translation is attached. This collection of information is required by 37 CFR 1.97 and 1.98. The information is required to obtain or retain a benefit by the public which is to file (and by the USPTO to process) an application. Confidentiality is governed by 35 U.S.C. 122 and 37 CFR 1.14. This collection is estimated to take 2 hours to complete including gathering, preparing, and submitting the completed application form to the USPTO. Time will vary depending upon the individual case. Any comments on the amount of time you require to complete this form and/or suggestions for reducing this burden, should be sent to the Chief Information Officer, U.S. Patent and Trademark Office, P.O. Box 1450, Alexandria, VA 22313-1450. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. **SEND TO: Commissioner for Patents, P.O. Box 1450, Alexandria, Virginia 22313-1450**.

If you need assistance in completing the form, call 1-800-PTO-9199 (1-800-786-9199) and select option 2.

Substitute for Form 1449/PTO	Complete if Known	
INFORMATION DISCLOSURE STATEMENT BY APPLICANT (use as many sheets as necessary)	Application Number	11/116,783
	Filing Date	April 28, 2005
	First Named Inventor:	R. Drew Major
	Art Unit	2144
	Examiner Name	Donabed, Ninos J
Sheet 2 of 2	Attorney Docket Number	8532.P001

FOREIGN PATENT DOCUMENTS

Examiner Initials*	Cite No.[1]	Foreign Patent Document (Country Code[3] Number[4] Kind Code[5] (if known))	Publication Date MM-DD-YYYY	Name of Patentee or Applicant of Cited Document	Pages, Columns, Lines, Where Relevant Passages or Relevant Figures Appear	T[6]
		EP 0 711 077 A2	05-08-1996	Haskell et al. (AT&T Corp)		
		EP 0 919 952 A1	06-02-1999	Mattavelli et al. (Ecole Poly.)		
		EP 1 641 271 A2	03-29-2006	Klemets et al. (Microsoft Corp.)		
		EP 1 670 256 A2	06-14-2006	Klemets et al. (Microsoft Corp.)		
		EP 1 777 969 A1	04-25-2007	__ (British Telecommunications)		

NON PATENT LITERATURE DOCUMENTS

Examiner Initials*	Cite No.	Include name of the author (in CAPITAL LETTERS), title of the article (when appropriate), title of the item (book, magazine, journal, serial, symposium, catalog, etc.), date, page(s), volume-issue number(s), publisher, city and/or country where published	T[2]
		PATHAN, Al-Mukaddim, et al., "A Taxonomy and Survey of Content Delivery Networks", Australia, February 2007. Available at http://www.gridbus.org/reports/CDN-Taxonomy.pdf	
		On2 Technologies, Inc., "TrueMotion VP7 Video Codec", White Paper, Document Version 1.0, January 10, 2005, (13 pages).	

Examiner Signature		Date Considered	

*Examiner: Initial if reference considered, whether or not citation is in conformance with MPEP 609. Draw line through citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

[1]Applicant's unique citation designation number (optional). [2]Applicant is to place a check mark here if English Translation is attached.

This collection of information is required by 37 CFR 1.98. The information is required to obtain or retain a benefit by the public which is to file (and by the USPTO to process) an application. Confidentiality is governed by 35 U.S.C. 122 and 37 CFR 1.14. This collection is estimated to take 2 hours to complete including gathering, preparing, and submitting the completed application form to the USPTO. Time will vary depending upon the individual case. Any comments on the amount of time you require to complete this form and/or suggestions for reducing this burden, should be sent to the Chief Information Officer, U.S. Patent and Trademark Office, P.O. Box 1450, Alexandria, VA 22313-1450. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS.

SEND TO: Commissioner for Patents, P.O. Box 1450, Alexandria, Virginia 22313-1450.

If you need assistance in completing the form, call 1-800-PTO-9199 (1-800-786-9199) and select option 2.

(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) EP 0 919 952 A1

(12) **EUROPEAN PATENT APPLICATION**

(43) Date of publication:
02.06.1999 Bulletin 1999/22

(51) Int. Cl.[6]: **G06T 9/00**

(21) Application number: **98109423.8**

(22) Date of filing: **25.05.1998**

(84) Designated Contracting States:
AT BE CH CY DE DK ES FI FR GB GR IE IT LI LU MC NL PT SE
Designated Extension States:
AL LT LV MK RO SI

(30) Priority: **28.11.1997 US 66856 P**

(71) Applicant:
ECOLE POLYTECHNIQUE FEDERALE DE LAUSANNE
1015 Lausanne (CH)

(72) Inventors:
- **Mattavelli, Marco**
1196 Gland (CH)
- **Brunetton, Sylvain**
1009 Pully (CH)

(74) Representative:
Ravenel, Thierry Gérard Louis et al
I C B,
Ingénieurs Conseils en Brevets SA,
7, rue des Sors
2074 Marin (CH)

Remarks:
A request for correction of the claims numbering has been filed pursuant to Rule 88 EPC. A decision on the request will be taken during the proceedings before the Examining Division (Guidelines for Examination in the EPO, A-V, 3.).

(54) Method for coding/decoding of a digital signal

(57) The invention concerns a method for primary processing a digital signal comprising the steps of :

- providing a digital signal to a primary processing unit ;
- primary processing said signal with said primary processing unit according to a plurality of primary processing algorithms to provide primary processed output signals,
- determining statistics of use of at least one of said primary processing algorithms,
- providing a digital statistics signal representative of said statistics for each of said primary processed output signals
- associating to each of said primary processed output signals its own statistics signal ;
- determining a complexity primary processing information signal based on said statistic signal.

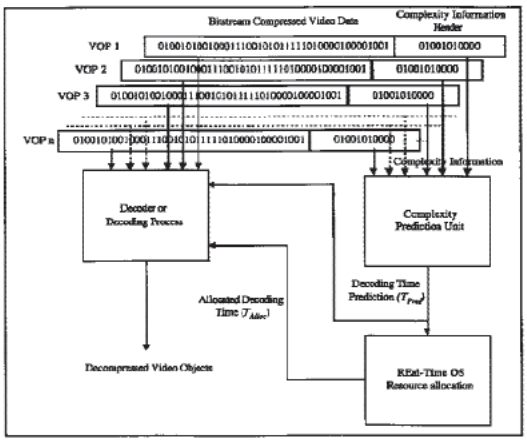


Figure 12

EP 0 919 952 A1

Ex. 1018-127

Description

DEFINITION OF THE INVENTION

[0001] The present invention is a processing apparatus which aims at controlling and guaranteeing the real-time performance of data processing. The invention is based on the principle of collecting statistics of the algorithms used to process the data, on transmitting such statistics when necessary, and of associating a complexity information to such statistics. The real-time processing control can be done generating a processing time prediction signal which is a function of the complexity information, and use such prediction signal to provide processing resources which guarantee real-time processing or enables the definition of an associated degraded processing that guarantees real-time processing. Although the described invention is of general application, this document describes its application to video coding and decoding and suggests the straightforward extensions to audio coding-decoding, and to composition and rendering of media objects.

1. FIELD OF THE INVENTION

[0002] The implementation of software based video/audio decoders is an advantageous solution over traditional dedicated hardware systems. The main reason of this fact is the possibility of decoding several different video/audio compression standards, at different levels and profiles, by just changing the software while using the same hardware platform, main memory, and interfaces. Another favorable feature is the flexibility of the decoder able to cope with new developments and new tools included into future versions of the standards by just updating the decoding software. However, simultaneously guaranteeing real-time performance, needed when processing video/audio bit-streams, and an efficient use of the processing power, remains a very difficult task.

[0003] The present invention reports the main reasons of such difficulties and presents a new technique able to predict the decoding time of compressed video/audio bit-streams without the need of the actual decoding. This can be obtained if appropriate statistical information about the actual encoding processing is included and transmitted with the compressed video/audio data. Moreover, the present invention reports how such results can be used to implement new efficient processing resource allocation strategies. New possible schemes of such intelligent interactions between the encoder and the real-time OS are also proposed. An example of implementation for the MPEG-4 VM6 video algorithm based on these developments is presented. This implementation enables, at the same time, to guarantee real time performances even when the available processing resources are lower than the one theoretically necessary, and it yields optimal (minimal) image quality degradation. Such results are obtained by using the prediction of the decoding-resource needs to implement Computational Graceful Degradation techniques. The invention only reports simulation results for the MPEG-4 VM6.1 video compression algorithm, but the techniques and concepts presented here are of general application. Indeed, these techniques can be easily applied to audio coding, synthetic audio reproduction and to any future or existing compression and composition method. More precisely, they can be used for any video compression standard that is capable of transmitting in an appropriate form the statistical coding information (i.e. MPEG-1 and MPEG-2 using an appropriate transport system), and they can be implemented in any processor-based platform.

[0004] The present description is organized as follows. Section 2 discusses the problems and utility of defining and using worst case and average case complexities, justifying the need of complexity prediction. Section 3 presents the modeling of decoding complexity and the general principles of the prediction techniques. Section 4 presents the background of the invention. Section 5 summarizes the main concepts of the invention while section 6 summarizes the content of the diagrams and drawings illustrating the details and main characteristics of the invention.

[0005] In section 7, experimental results of the MPEG-4 VM6 texture coding algorithm are reported. Section 8 presents the definition of the proposed intelligent interactions between real-time OS-schedulers and video decoders. Section 9 presents the basic principles and the definition of the developed Computational Graceful Degradation (CGD) techniques, and section 10 reports overall experimental results of the control of decoding complexity using complexity prediction joint with the application of CDG techniques. Finally, section 11 concludes the description of the invention, summarizing the achievements of the invention by declaring the corresponding claims.

2. WORST-CASE AND AVERAGE-CASE DECODING COMPLEXITY

[0006] A simple solution to the problem of the variability of decoding complexity is to implement software/hardware solutions that always guarantee worst case complexity decoding. Such solutions are, in fact, the ones implicitly implemented in most of the dedicated hardware decoders. Dedicated hardware platforms, in general, are designed to guarantee worst case complexity decoding. The various hardware modules implementing the different decoding procedures are synchronized with the input/output data rate. These dedicated hardware modules can always handle any input bitstream, not exceeding a maximum rate, and containing compressed sequences of appropriate maximum image reso-

Ex. 1018-128

lutions. This can be always guaranteed, irrespectively from the coding options selected to produce the compressed bitstream.

[0007] In contrast, in the case of programmable platforms it is necessary to define what is a worst case decoding complexity. Let us consider video coding even if the same concept can be extended to natural and synthetic audio and to other form of multimedia processing. In the video compressing field one can assume that the worst case decoding complexity is represented by the decoding of each bit/macroblock, using the coding options allowed by the considered standard that requires the highest decoding time. The coding algorithms/modes of H.263 can be approximately considered as a superset of the algorithms/modes of the various video compression standards (H.261, MPEG-1 and MPEG-2). The complete results and details of this worst-case/average-case study are omitted here for brevity. The conclusions resulting from this analysis can be summarized as follows :

1) Decoding functions can be divided into two classes. For one class a worst case can be easily defined. Let us denote this class as A. To this class belong function such as IDCT, inverse quantisation, motion compensated prediction in its various form (forward, backward, bi-directional, overlapped ... .). For the other class denoted as class B, complexity is extremely data dependent and cannot be easily characterized by the simple parameters of an hypothetical MPEG-2 like complexity profile. To this class belong operations such as VLD, VLC coding, Syntax-based Arithmetic Coding (SAC) and in general all parsing operations. The difficulty of defining a worst case is evident considering for instance SAC for which the difference between a theoretical worst-case and experimental cases is about two orders of magnitude.

2) A strict worst case complexity analysis, of both class A and B functions, leads to unrealistic results. In other words, the computational resources necessary to handle the strict worst-case complexity decoding is much higher than the resources used in average.

3) For class A functions, the actual decoding complexity depends on the actual coding options and reports a large range of variability.

4) For class B functions, the actual decoding complexity reports a dependence on coding options, but presents a reasonably narrow range of variability which is much lower than the range theoretical worst case-average case.

[0008] Worst case analysis, therefore, is not adequate to define a useful decoding complexity reference. This fact is true not only for operations which are strongly data dependent (class B), but in general for all functions even those that have a very well defined worst case such as all class A functions. From these results it is clear that guaranteeing a worst case decoding implies guaranteeing the decoding of pathologically complex sequences that in practice have an extremely low probability of occurrence. Moreover, such a guarantee would imply the allocation of a processing power up to two orders of magnitude higher than the one actually needed for the decoding of the standard test sequences. The processor in practice would work in normal conditions exploiting only a small fraction of the theoretically available processing power. Such solution to the problem of guaranteeing real-time performance is not economical and, obviously, not useful in practical applications.

[0009] Another possibility for the solution of the problem of guaranteeing real-time performance could be to consider as reference an average case decoding complexity. Before being able to successfully use such average complexity, a few questions need to be answered. Is it meaningful to define an average complexity? How can it be done? Is the average complexity sequence dependent? How can it be converted on different platforms? Can the average complexity accurately be described by parameters such as image resolution and bit-rate? Are average results characterized by sufficiently narrow variance bounds so as they can be used a priori without the need of knowing some information about the actual complexity of the incoming sequence?

[0010] One will here briefly summarize the main results and experiments that answer the mentioned questions. So as to try to define an average case for the complexity that have been considered in previously, the H.263 standard uses four typical test sequences over a wide bit-rate range from 16 kbit/sec up to 1024 kbit/sec. The sequences in CIF format are : Akiyo (300 frames), Coastguard (300 frames), Foreman (300 frames) and Mobile & Calendar (300 frames), and all these test sequences have been evaluated at fifteen different bit-rate values. Although a rigorous statistical analysis would have required a much higher number of sequences and specific statistical confidence tests, it is believed that the number of frames (1200) and the different contents of the test images are well representative of typical image sequence contents. One have considered separately the operations that have unrealistic worst case complexities (class B) from the one with a well-defined worst case behavior (class A). The experiments, for the class B operations, yield figures that are not strongly bit-rate dependent. Variance bounds for each frame are relatively narrow and these values do not report relevant variations depending on the test sequence. Such results confirm also the fact that it is not necessary to worry about strict worst case complexity. In the other case of class A, the complexity of the operations reports a relatively regular statistical behavior at very low bit-rates (< 16 kb/s), but complexity reports dependence on the bit-rate, on the image content and in particular on the used encoding options.

[0011] Figure 1 reports one example of H.263 complexity using different coding options for the same sequence (Akyio

Ex. 1018-129

CIF) and same bit-rate. On this figure, the decoding time or decoding complexity (expressed in μsec) is given versus the frame number for the sequence Akyio CIF encoded with H.263 at 128 Kb/sec using different coding options. Top curve options : advanced prediction modes (APM), syntax-based arithmetic coding and unrestricted motion vectors (UMV) ; middle curve options : APM and UMV ; bottom curve options : UMV. The same sequence presents, at the same bit-rate, average-decoding complexities that range up to a factor 4 (from about 0.05 to 0.2 sec per frame). Local fluctuations for a relatively static sequence such as Akyio are also considerably large. The decoder is the Telenor H.263 software release running on a Sun Ultrasparc platform.

[0012] Figure 2 reports the same experiment for the sequence Coastguard. The range of the complexity figures considering both average and local values is very large as can be easily noticed. On this figure, decoding time or decoding complexity (expressed in msec) is given versus frame number for the sequence Coastguard encoded with H.263 at 128 Kb/sec using different coding options. Top curve : advanced prediction modes (APM), syntactic arithmetic coding and unrestricted motion vectors (UMV) ; middle curve APM and UMV ; bottom curve : UMV. The same sequence at the same bit-rate has average decoding complexities that range within a factor 4 from about 0.05 to 0.2 sec per frame. The decoder is the Telenor H.263 software release on a Sun Ultrasparc platform.

[0013] Although far from the theoretical worst case complexities, the variation range is less than one order of magnitude, experimental results reports behaviors for which it is difficult to be able to predict the average complexity values only relying on resolution and bit-rate. Image content and coding options (i.e. intra or inter prediction modes, advanced prediction modes, half pixel vector prediction and so on) play a fundamental role. As it will be reported in the next sections, the situation of the complexity variations for MPEG-4 sequences is much higher due to the presence of different sized VOPs arbitrarily shaped objects, of static and dynamic sprites, and the large variety of coding modes.

[0014] From a pure mathematical point of view an average complexity value can always be defined considering a large data-base of encoded sequences and using the more probable occurrence of encoding options. Conversely, such a theoretical average value is not useful at all for the aim of the present invention that is the efficient scheduling of each decoding task and the efficient use of the available processing power. Even the local average values of the actual sequence often are not stationary, and their variations around the mathematical average one are so unpredictable that no useful scheduling policy can be realized.

[0015] Therefore, the consequences of this study are that straightforward solutions to the problem of real-time resource allocation policy are of two kinds. The first is a sustained over-sizing of the processing resources. The second is to accept interruptions of the audio/video output stream when the needed decoding resources exceed the ones available on the system. Obviously, both are not desirable being respectively not efficient and economical, or they provide a quality of the services which may easily results perceptually unacceptable.

[0016] The ideal solution would be to develop techniques that are able to accurately predict such unknown behaviors and, therefore, make feasible resource allocation policies that can guarantee real-time and minimize QoS degradation.

[0017] The description of such techniques and the presentation of the results are the subjects of the next sections.

[0018] As discussed above, the key issue for the implementation of any real-time application on a software-based platform is to be able to understand when the required decoding processing time exceeds the available processing time without the need of performing the actual decoding. This is particularly important, for instance, in MPEG-4 systems in which a number of bit-streams containing a variable number of VOPs and sprites need to be decoded respecting real-time constraints, but it is also important for existing compression standards (MPEG-1, MPEG-2). The available processing time is known, in general, because it is allocated by the OS-scheduler to each decoding process. The problem is how to estimate the necessary decoding processing requirements without the need of an actual decoding of each of the incoming VOP bit-streams. Decoding each bit-stream using a priori allocated processing could result in exceeding the time slot and, therefore, in missing real-time requirements. Real-time requirements are generally defined implicitly by the frame/sampling rate of video audio bit-streams or can be easily extracted by an appropriate relative time information associated to each video frame or audio object (also denoted as time stamp). Such extraction of the real-time constraints is trivial and is not discussed here. The consequence of missing real-time constraints is the introduction of sudden interruptions on the video content. A complete analysis of the conditions which lead to miss real time-constraints for the various OS scheduling algorithm (pre-emptive multi-threading, etc.) is out of the scope of this document and it is not discussed here. The reader can refer to the included reference S. Battista, F. Gasperoni, B. Klaiber, M. Mattavelli, L. Pautet, S. Tardieu, Concurrent over Sequential Modeling for MSDL, ISO/IEC JTC1/SC29/WG11, Doc. M1525, Maceio, Brazil, November 1996 incorporated herein by reference.

3. BACKGROUND OF THE INVENTION

[0019] Without any doubt, the recent trend in multimedia system implementation technology is towards moving from dedicated real-time hardware to software/processor based platforms. The processing power of the last on the market processors, and of the newly announced video/audio-processor families, enables the software implementation of video/audio decoding of various compression standards and different resolution levels. Unfortunately, simply relying on

Ex. 1018-130

the average processing power does not guarantee to respect real-time constraints of video/audio decoding ; furthermore it does not solve the problem of an efficient processing resource allocation. The fundamental reason of these facts is that the complexity of the video/audio decoding process is a fluctuating function of the time, as described in M. Mattavelli and S. Brunetton, A statistical study of MPEG-4 VM texture decoding complexity, ISO/IEC JTC1/SC29/WG11, Doc. M924, Tampere Finland, July 1996 incorporated herein by reference.

[0020] The term complexity here simply means the time needed by the decoding process. Other more abstract measures of complexity such as the amount and type of executed algorithmic elementary operations, the executed processor cycles, the amount of memory transfers to and from the external frame memories, and their combinations could be used as more precise and specific complexity measures. In the context of the present invention however, one is interested on the final results of these complexity factors. Without any loss of generality, one can consider, for any specific target software-hardware implementation, the actual processor decoding time. Indeed all the above mentioned measures of complexity can be converted for each specific platform into a single complexity measure given by the execution time.

[0021] Even by considering a specific processor platform or a specific compression standard with a relative level and profile, the complexity of video decoding is a variable and unpredictable quantity, as described in ISO/IEC MPEG2 incorporated herein by reference. The actual figures of decoding complexity depend in a complex way on many different factors. One can mention, for instance, the actual bit-rate, the image content, the options used for encoding, the Group Of Picture (GOP) structure, or the Group Of Video Object Planes (noted as VOP) structure (GOV) which can change on the fly, the state of the buffer that controls the quantization parameters and many others, as described in M. Mattavelli and S. Brunetton, Scalability of MPEG-4 VM based on the computational power of texture decoding, ISO/IEC JTC1/SC29/WG11, Doc. M925, Tampere, Finland, July 1996 incorporated herein by reference.

[0022] The complexity variability range increases with the richness of the standard. Richness means the number of implemented coding algorithms or coding modes and the number of possible coding options having different complexities. For instance, the richness of H.263 is much higher than H.261, but it is lower than the new MPEG-4 standard under development where multiple Video Object Planes (VOPs) of different sizes and characteristics need to be decoded.

[0023] In this description, Video Object (VO) means a temporal sequence of two-dimensional or three-dimensional arbitrarily shaped Video Object Planes (VOP). In case of rectangular or square shapes, these video object planes, also called hereinafter video planes, are denoted by the classical name of frames. An audio object means a stream of packets containing digital audio samples, or audio events in mono or multichannel format which can be associated to a VO. One or more audio packets can be associated to one VOP or video frame. By three-dimensional arbitrarily shaped video object planes we mean the projection or view of any three-dimensional natural or synthetic 3-dimensional object or 3-dimensional model.

In summary, the processor decoding time of each incoming frame or VOP for any given compression standard is highly variable and impossible to predict just relying on resolution and bit-rate information. Therefore, an efficient resource allocation policy (process scheduling) which aims at guaranteeing real-time performances is very difficult in these conditions. Scheduling policies that release the thread to the decoding process and receive it back after the task is completed are clearly not adequate. The same conclusion is true for a priori allocated processing time intervals. If more processing time than the one actually allocated is needed, real-time constraints are already violated by the video application. The implementation of emergency measures such as skipping frames or portions of images, on one hand might succeed in recovering the correct timing, on the other could have catastrophic effects on the Quality of Services (QoS) of the video application.

4. SUMMARY OF THE INVENTION

[0024] It is an object of the present invention to control and to guarantee real-time performance of data processing. It is another object of the present invention to provide a processing apparatus that associates a complexity information to the processed data.

[0025] It is another object of the present invention to provide a coding-decoding apparatus that further includes the associated complexity information to be transmitted to the decoder.

[0026] It is another object of the present invention to provide a system that, on the basis of said complexity information, provides a complexity prediction signal to be used to allocate processing resources to optimize real-time data processing.

[0027] It is another object of the present invention to provide a processing system that supervises the processing using an allocated resource signal, a predicted complexity signal, a complexity information signal and provide a processing that satisfy specified real-time constraints. According to the present invention, there is provided a processing apparatus comprising a method for processing a digital signal including the steps of : providing a digital signal, processing said signal using a plurality of processing algorithms, determining statistics of use of processing algorithms, providing a digital statistical signal representative of the statistics, associating to each of the processed signal its own

Ex. 1018-131

associated statistics and determining a complexity information signal based on said statistic signal.

[0028] According to the present invention, there is also provided a coding apparatus that further inserts the said complexity information signal into a transmission layer containing or not the associated data and transmits the said complexity information to a decoder apparatus.

[0029] According to the present invention, there is also provided a processing apparatus that generates from the complexity information signal a processing prediction signal which is used to allocate processing resources in order to optimize the real-time processing of the associated data.

[0030] According to the present invention, there is also provided a processing apparatus that receives the processing prediction signal, the allocated processing resources and the complexity information signal, and uses the said signals to generate processing directives so as the data is processed according to the system real-time constraints.

[0031] Although the described invention is of general application, the present description describes its application to the preferred embodiment of the video coding and decoding, and suggests the straightforward extensions to audio coding-decoding and to composition and rendering of media objects.

[0032] The technique provided by the present invention is capable of predicting with an excellent degree of approximation the complexity of video or audio decoding, without the need of the actual decoding of the incoming bit-stream. These results are possible when an appropriate coding information can be included in the incoming bit-streams. The experiments demonstrate that the degree of approximation achievable is excellent considering the range of variability of the actual decoding complexity of video or audio compression standards, and that such degree is more than adequate to enable an efficient processor resource allocation. One application of the proposed technique to the MPEG-4 VM6 video coding algorithm is also demonstrated. Real-time video performances are guaranteed using the predicted decoding complexity to implement Computational Graceful Degradation (CGD) features whenever the video decoding complexity exceeds the available processing resources. By means of the proposed technique, it can be guaranteed at the same time the satisfaction of the real-time constraints and the aim of minimizing the unavoidable degradation of the QoS due to temporary lack of processing resources or to peaks of the intrinsic video decoding complexity. The same approach can obviously be used for audio decoders.

[0033] It is clear that the approach is not limited to the coding/decoding process for compression purposes, but that it can be applied to the process of composing and rendering media objects (i.e. video, audio, text, graphics, two-dimensional and three-dimensional graphic primitives and models) to the display output. The term composition means the action of putting these objects together in the same representation space. The action of transforming the media objects from a common representation space to a rendering device (speakers and viewing window) is called rendering. An illustrative example of the terminal environment for the handling of multimedia object is reported in Figure 19. More specific possible examples of the decoding, composition and rendering processes are reported in Figure 20 and 21. Figure 19 reports the processing stages in an audiovisual terminal illustrating inputs outputs and processing of audiovisual objects. Figure 20 reports an example of a decoder which provides uncompressed audiovisual objects to the audio/video compositor. Figure 21 reports an illustrative example of composition of audiovisual objects.

[0034] Additional objectives and advantages of the invention will be set forth in the description that follows, and in parts will be obvious from the description, or may be learned by practice of the invention. The objectives and advantages of the invention may be realized and obtained by means of the instrumentalities and combinations particularly pointed out in the appended claims.

5. BRIEF DESCRIPTION OF THE DRAWINGS

[0035] The accompanying drawings constitute a part of the specification and illustrate the principles of the invention.

- Figures 1 and 2 report a graphical representation of the decoding complexity of H.263 video illustrating the needs and background of the invention for the video decoding embodiments ;
- Figure 3 reports the an example of the modeling of processing algorithms. It includes the association of processing statistic to each state associated to the execution of a specific algorithm ;
- Figure 4 reports the extension of the model of figure 3 for a more complete and accurate model of the algorithmic complexity ;
- Figure 5 reports one preferred embodiment of the invention for the MPEG-4 video decoding. It illustrates a complexity model with the associated statistics for MPEG-4 texture decoding ;
- Figure 6 extends the model of figure 5 including additional coding modes and coding parameters plus the associated statistics ;
- Figures 7 and 8 report simulation results of a preferred embodiment of the invention. By associating a complexity measure to the statistics of the data processing (video decoding in the preferred embodiment), it is possible to predict the processing complexity of data ; actual decoding time, predicted decoding time and prediction error are represented.

Ex. 1018-132

- Figure 9 reports the measured decoding time and Decoding Time Prediction (expressed in μsec) versus the sequence VOP number of the MPEG-4 VM6 for the sequence Coastguard (Background, Small boat, Big Boat and Water VOPs) at 64 Kb/s ; these results being obtained on a specific decoder.
- Figure 10 reports the measured decoding time and Decoding Prediction Time (expressed in μsec) versus the sequence VOP number of the MPEG-4 VM6 for the sequence Coastguard (Background, Small boat, Big Boat and Water VOPs) at 128 Kb/s ; these results being obtained on a specific decoder.
- Figure 11 reports the measured decoding time and Decoding Prediction Time (expressed in μsec) versus the sequence VOP number of the MPEG-4 VM6 for the sequence Coastguard (Background, Small boat, Big Boat and Water VOPs) at 128 Kb/s ; these results being obtained on a specific decoder
- Figure 12 reports a block diagram of part of the invention. The statistics carrying the complexity information and associated to each data to be processed are read by a prediction unit which generates a prediction signal sent to the OS scheduler which allocates resources as function of the prediction signal. The figure is reporting one example of preferred embodiment for the video decoding of multiple VOP ;
- Figure 13 reports a block diagram of part of the invention relative to the construction of the complexity information header at the encoder side. Statistics of the coding and/or decoding algorithms/modes/parameters are generated at the encoder side and then inserted in a bit-stream to be transmitted to the decoder ;
- Figure 14 reports a block diagram of part of the invention relative to the interaction between the decoder supervisor and the decoder using various signals in order to guarantee real-time decoding of the data ;
- Figure 15 reports the diagram of a complexity model including associated degradation algorithms ;
- Figures 16, 17, 18 report examples of results of a preferred embodiment of the invention. The video decoding for the MPEG-4 texture decoding is provided imposing a maximum constant decoding time (complexity) for each VOP. Degradation results are also reported ;
- Figure 19 reports an example of a possible preferred embodiment of the invention illustrating the various interactions and processing related to the handling of media objects (audio, video, text, graphic, 2-D and 3D models) in a multimedia terminal ;
- Figure 20 reports an example of a portion of multimedia terminal block diagram that is another possible embodiment of the invention, and
- Figure 21 reports a drawing illustrating the composition and rendering processes, both being possible preferred embodiments of the invention.

6. PREDICTING DECODING COMPLEXITY BY MEANS OF ENCODING STATISTIC INFORMATION

6.1 A model of decoding complexity for data dependent processing

[0036] To achieve the objectives of the present invention, one need a simple model of data processing complexity that preferably has the following properties :

1) it is only related to the processing of the incoming data,
2) it is as compact as possible so that its description requires the minimum amount of information (number of bits),
3) it is representative in terms of processing complexity,
4) it is not related to the any specific software or hardware implementation of the given processing,
5) it can be easily mapped on any software/hardware platform implementation,
6) it is sufficiently accurate to yield useful predictions of the processing execution time.

[0037] A way of describing the operations of processing incoming data can be to consider the sequence of ensembles of operations as appropriate distinct algorithms. If the sequence of algorithms is not dependent explicitly on the actual content of the data to be processed or by the intermediate results of the processing, the complexity of the processing can be easily calculated a priori and the complexity-modeling problem is trivial.

[0038] In general, if the sequence of operations is explicitly data dependent (i.e. the processing instructions are contained in the data themselves) and depends also on intermediate results of the processing, the problem of modeling algorithmic complexity is much more difficult.

[0039] It is well known that any processing can be modeled by a finite-state-machine for which next state depends on the input, current state and current output (the well-known machine of Mealy). This formulation is rigorous and general. In theory it could also be used to measure the processing complexity. If one assume to associate a complexity measure to each state transition, one could measure the global complexity of the processing by monitoring the state of the machine at each step, and adding up the complexity related to each state transition. In practice, such model is not useful for two main reasons. The state machine description of usual video decoding would result extremely complex. The global complexity associated to the sequence of state transition would result too abstract to make feasible the mapping

Ex. 1018-133

on different implementations than the canonical state machine. In practice, it would be extremely difficult to derive a complexity expressed as total execution time for different specific platforms.

[0040] A more useful model is to represent the decoding process by a flux diagram for which each data state corresponds to the execution of a specific algorithm f. Such representation is more naturally related to the usual representations of the execution of software programs. Let us associate to each data state a measure of the occurrence of such state, which is incremented each time the data processing falls in each state. Let us suppose to monitor the execution of the decoding process and, for the occurrence of each algorithm execution, to increment the associated measure. The result of this model is a representation of the data processing given by a sequence of measures associated to each algorithm composing the complexity model. In general, any encoder knows which algorithm sequence is needed for the decoding, since the encoding and the coding syntax define without ambiguities the decoding algorithms . In particular, video decoders have to implement a decoder, to build, reference images, therefore they are able indeed to generate the described complexity representation.

[0041] This modeling approach has the nice feature that admits different levels of complexity resolution. One can consider only a very limited number of main algorithms, or partitioning into a large set of sub-algorithms or any intermediate representation. The cost of the representation, obviously, depends on the number of algorithms that compose the model. The accuracy of the complexity representation depends on the number and type of algorithms, in other words on their relevance in terms of complexity. A simple example of a complexity model is reported in figure 3 which reports a flow graph for the modeling of a processing composed by three algorithms f and associated occurrence complexity measures c. In this model, the processing is represented by tree algorithms f and by the associated counters c. Figure 4 reports a flow graph of the model of Figure 3 for which the complexity modeling of algorithms f1 and f3 have been designed by partitioning f1 and f3 in two new algorithms and their associated complexity models, the model is more accurate since the algorithms 1 and 3 are partitioned into other algorithms associated to new algorithm occurrence measures. It must be noted that what are considered as coding modes associated to specific coding options in the usual coding language, are from the point of view of the invention distinct algorithms, since different complexities correspond to different coding modes. It must be also noted that the executions of the same algorithm using different parameters can be considered as the executions of distinct algorithms associated to the corresponding parameters.

[0042] The mapping of the complexity into any software/hardware implementation is straightforward. By knowing the complexity (execution time) of each algorithm on a specific platform and by using the information of the actual occurrence measure of each algorithms in the decoding process, one can easily calculate the total execution time (what is denoted as complexity) for the processing of an incoming bit-stream.

[0043] The key issue of this approach is to be able to find the representation in terms of minimal set of algorithms and sub-algorithms (tools) that being representative of the processing complexity, yields the best approximation of the execution time. This is particularly evident for very complex processing task such as video decoding. The importance of a minimal set is to keep as compact as possible the cost of the complexity representation and, therefore, make possible its separate transmission at a low cost.

[0044] Another feature of the proposed model is that it can easily take into account parallel software or hardware implementations. No hypothesis has been done on the internal implementation of each algorithms. In fact, they can be implemented at any level of hardware or software parallelism. The case for which different algorithms of the model have parallel implementations can also be treated by simplifying the model and just ignoring some counters associated to algorithms implemented in parallel. Simple practical examples of these cases are given below after the complete definition of the modeling-prediction complexity problem.

6.2 <u>A technique to predict video decoding complexity : Decoding Complexity Prediction by Coding Statistics (DCPCS)</u>

[0045] Let us consider a specific software/hardware implementation of a video decoder, running on a specific hardware platform. Since no hypothesis is made on the architecture of the software or on the architecture of the hardware, the approach doesn't loose any generality.

[0046] Let us consider that the true complexity C* (execution time) for the decoding of a given input data (i.e. a single frame or a MPEG-4 VOP) is a function P*, specific for the software/hardware implementation at hand, of the sequence of all the various processing operations p_l 1=1,...,L needed to decode the incoming data :

$$C^* = P^*(p_1, p_2, \ldots, p_l, \ldots p_L). \tag{1}$$

[0047] The p_l describe the sequence of operations that define the decoding process. They are, for instance, the parsing of the bit-stream, the sequence of operations for decoding the in various form Intra, Inter, Not-Coded, Shape, Padding, the inverse quantisation, the IDCT, the motion compensation in its various form OBMC, simple prediction, 4 vector prediction , the parameter that characterize the processing of each prediction mode (integer pel vector prediction, half pel vector prediction), the number of bitwise parsing operations, the number of non-zero DCT coefficients and so on.

[0048] The actual expression of the function P* is certainly extremely difficult to obtain. On the other hand, the objective of the present invention is to avoid the necessity to know all the parameters that define the complexity, but just to be able to predict the true complexity C* with a good degree of approximation. The cost of the description of the exact complexity function P* in terms of all coding parameters would be, indeed, prohibitive for the purposes of the present invention.

[0049] Therefore, let us leave the problem of finding the exact expression of the complexity function P* and try to find a sufficiently accurate approximation F of the function P* which provides a prediction C, using the model proposed in the previous section. One is therefore limited to consider a number I of algorithms fi, with i=1,...,I with I << L, and the associated model counters c_i :

$$C = F(c_1 \cdot f_1, c_2 \cdot f_2, \ldots c_i \cdot f_i, \ldots c_1 \cdot f_1) = c_1 \cdot w_1 + c_2 \cdot w_2 + \ldots + c_i \cdot w_i + \ldots + c_1 \cdot w_1 \approx C^* \qquad (2)$$

where the coefficients w_i related to each c_i are complexity weight corresponding to the model algorithm f_i which accounts for the specific software and hardware architecture at hand.

[0050] Let us suppose now, that the values of the vector c for the incoming data (frame or VOP) are known. If the vector w of the w_i weights is known, one can predict the complexity of the incoming VOP bit-stream by a simple linear combination.

6.3 <u>A method to estimate the algorithms weight coefficients w</u>

[0051] The values of the vector w can be obtained by direct measurements on the platform, if the complexity values (execution time) of each function are accessible. In general, they can be also indirectly estimated by means of a simple linear regression applied on a set of measures of the true complexities C^*_k, with k =1 ... K, obtained by K samples of data processing (i.e. K frames or K VOPs taken from different sequences) on the selected platform :

$$\begin{cases} C^*_1 = W_1 \cdot C_{11} + W_2 \cdot C_{12} + \ldots + W_i \cdot C_{1i} + \ldots + W_I \cdot C_{1I} \\ C^*_2 = W_1 \cdot C_{21} + W_2 \cdot C_{22} + \ldots + W_i \cdot C_{2i} + \ldots + W_I \cdot C_{2I} \\ \ldots\ldots\ldots \\ C^*_k = \ldots \\ \ldots\ldots\ldots \\ C^*_K = W_1 \cdot C_{K1} + W_2 \cdot C_{K2} + \ldots + W_i \cdot C_{Ki} + \ldots + W_I \cdot C_{KI} \end{cases} \qquad (3)$$

[0052] In matrix form the estimated vector of weights w can be expressed by the classical least square minimization expression :

$$\hat{w} = (c^T c)^{-1} c^T C, \qquad (4)$$

where T indicates the transpose matrix,
and -1 indicates the inverse matrix.

[0053] Obviously, the linear regression and the measures of the C_k complexities can be made off-line since they have as objective the estimation of the vector w which is characteristic of a specific software/hardware decoder implementation and do not depend on the incoming data.

[0054] Once the vector of the w parameters is estimated, the evaluation of the actual complexity C of the incoming bit-stream can be simply done by calculating the linear combination of the complexity function. In practice, this can be done by reading the actual c_i counter values written in an appropriate form in the incoming data, and calculating the value of F(c). This fact implies, for instance, that video compression standards should be able to insert complexity information in appropriate headers of frames or VOPs, or transmit them as accompanying information. Figure 7 reports the measured decoding time and Decoding Time Prediction (expressed in msec) versus the sequence VOP number of the texture decoding of MPEG-4 VM6 for the sequence Akyio (Background and Akyio VOPs) at different bit-rates. The decoder is the Momusys VM6 on a Sun Ultrasparc. The absolute value of the Decoding Time Prediction error is

Ex. 1018-135

reported in the bottom curve near the 0 value.

6.4 Definition of the complexity models for the texture decoding of MPEG-4 VM6 VOPs

[0055] In the previous section, a model for decoding complexity has been developed, aiming at being at the same time related only to the processing and apt to be mapped on any platform implementation. These ideal requirements have been expressed in six points. A last requirement needs to be verified. Does the model and the associated complexity representation provide complexity (time) predictions sufficiently compact but accurate enough for the aims of the present invention? There is described in this paragraph an example of application of the modeling to the texture decoding of MPEG-4 VM6 VOPsm, excluding the processing of shape coding and the padding of macroblocks. This exclusion is just motivated by the fact that at the time of the modeling and simulations, one have investigated only the stable coding modules of the MPEG-4 VM. The same modeling principles can be applied to those and to all new MPEG-4 coding algorithms and modes.

[0056] Let us partition the decoding of each MPEG-4 VOP in the following seven algorithms which represent all the texture coding process of any VOP including luminance and chrominance (Intra, Inter, Skip, Advanced Prediction Mode (ADM), Normal Prediction Mode (NPM), DCT coefficient processing (DCT_coeff), and DCT line processing (DCT_line)). Let us associate to each coding-decoding algorithm a statistic measure defined as follows. For Intra, Inter and Skip, it counts each time a macroblock is coded by using the Intra, Inter or Non-Coded coding mode. For ADM and NPM the statistic measure indicates the number of blocks 8x8 predicted using advanced prediction modes (OBMC or four vectors for macroblock) or normal prediction modes. For DCT_coeff, the measure counts the total number of DCT coefficients per VOP, and DCT_line the total number of executed DCT 8x1 per VOP. All these counters can be easily inserted in any MPEG-4 compliant or any other video/audio encoder. Figure 13 reports a schematic diagram of the process of counting coding algorithms/modes/parameters and provides a statistic description of the decoding process according to any complexity model. This diagram represents the generation of the Complexity Information Header at the encoder side. The encoder increments the counters of the coding algorithms/modes/parameters that are used to encode the video frame or video object to generate the associated statistics signal (known as complexity information). At the end of the encoding of each video frame or video object the statistics of the used coding algorithms/modes/parameters is inserted in the Complexity Information Header. Figure 1 reports a simplified flow graph of the MPEG-4 VM6 texture decoding algorithms. Note that, in the proposed technique, it is not necessary to explicitly model the actual decoder flow graph. The above graph represents the schematic flow diagram of a possible simple VOP texture decoding model of MPEG-4. It has to be noticed that by estimating the complexity weights using the total VOP decoding measures as explained in the previous paragraph, it is not even necessary to explicitly specify all transitions in the decoding flow diagram model.

[0057] More accurate complexity models could be designed by introducing new algorithms and the associated measures of occurrences. An example of a more accurate model including the partitioning of APM and NPM into predicted or bi-directional modes and into half-pel motion compensation modes is reported in Figure 6 which represents an example of extension of the MPEG-4 texture decoding complexity model including the different mode prediction and half-pel motion compensation functions.

[0058] A reduction to simpler decoding models when many counters of different algorithms are available is straightforward. Some counters can be ignored if, in a specific software/hardware implementation, they do not implies any or any relevant complexity difference. For instance, if the implementation does not yields different processing time if a macroblock is predicted by a full pel or half pel vector, or if there is no difference between APM or NPM, all these counters can be ignored or can be combined. The specific action depends on the architecture and on the complexity features of the software/hardware decoding platform used.

[0059] As reported in these model examples, a meaningful partitioning of the algorithms in terms of complexity can be easily done only referring to the processing of the data and without the need of specifying any software or hardware implementation. Only in the phase of estimating the weights w_i and to perform the actual prediction it is necessary to refer to a specific hardware/software implementation.

[0060] The two simple models presented here are also capable of taking into account parallel implementations which are implemented in each algorithm such as any packed arithmetic implementation or higher levels of parallelism. Parallel implementations at levels of macroblock decoding are not likely to occur in MPEG-4 systems. Decoders need to respect a given sequential order in decoding the macroblocks that is intrinsic in the syntax of the video data flow. Moreover, for multimedia processor implementations, the ones that are particularly considered in this description, parallelism is likely to be implemented at much lower levels than macroblock decoding. Higher levels are indeed very unlikely to appear.

[0061] Platforms that include also some hardware module accelerators, can be taken into account for the model. If one have, for instance, the hardware implementation of a 8x8 DCT module, one can simply discard the DCT_coeff and DCT_line counters and simply use the combination of Intra and Inter counters to predict the DCT decoding complexity.

Similar measures can be taken for other hardware accelerator modules.

[0062] The present complexity modeling approach can be applied to the audio/video decoding, composition and rendering of media objects or to any real-time processing of data.

7. EXPERIMENTAL RESULTS OF VIDEO DECODING COMPLEXITY PREDICTION

[0063] This section reports simulation results of the prediction of VOP decoding complexity for the MPEG-4 VM6 compression standard. Counters corresponding to the algorithms defined by the model of Figure 5 have been inserted in the Momusys VM6 encoder implementation to generate the mentioned statistic measures. These integer figures have been converted into percentage values referred to the actual number of VOP macroblocks, normalized to 8 bit values and have been supposed to be transmitted as a VOP complexity header for each VOP. The cost of this information transmission results of a few tenths of bits and can be considered negligible if compared to the total video bit-rate. Obviously, any other representation of the statistical coding information which has a sufficient precision can be employed. A schematic diagram of this decoding time (complexity) prediction signal generation process is reported in figure 14 which represents a schematic diagram of VOP video decompression interactions between the Video Decoder and the Video Decoder Supervisor when the Allocated Decoding Time (T_{Alloc})is lower than the Decoding Time Prediction (T_{Pred}). The Target Decoding Time and Decoder Time Reference are used to verify that the Decompressed Video elements are made available at the desired time. The Target Decoding Time is derived implicitly by the frame or sampling rate or by the information associated to each VOP or video frame (associated time stamps).

[0064] Measures of the texture decoding time of each VOP for different video sequence have been performed on a Sun UltraSparc platform. The decoder is the Momusys VM6 decoder.

[0065] The estimation of the weight w vector has been obtained by a linear regression applied on the measured decoding time of each VOP using the expression of equation (4). To demonstrate the robustness of the proposed technique, the complexity prediction results have been generated using a vector w, estimated using the complexity measurements of all the test sequences, except the one for which the prediction is calculated. For instance, to predict a sequence A, the weights estimated on sequences B and C have been used, and so on respectively for the other test sequences. Table 1 reports the results of the different estimates of the vector w for the sequences Akyio, Coastguard and Bream. As can be easily noticed, complexity (time) estimates from different sequences do not differ relevantly from one to the other. Disposing of a relatively large set of sequences with the associated measurements, one can easily perform off-line estimates of decoder weights w characteristic of the decoder at hand.

[0066] Figure 7 reports the measured VOP decoding time, the VOP decoding time prediction and the time prediction errors (in absolute value) for the two VOPs of the sequence Akyio. Figure 8 represents the measured decoding time and Decoding Time Prediction (expressed in msec) versus the sequence VOP number of the MPEG-4 VM6 for the sequence Bream (Background and Fish VOP) at different bit-rates. The decoder is the Momusys VM6 on a Sun Ultrasparc. The absolute value of the Decoding Time Prediction error is reported in the bottom curve near the 0 value. The results have been obtained at different bit-rates and all measures are expressed in msec. The same experiment results are reported for the sequences Bream and Coastguard. In figures 9, 10 and 11, as can be easily noticed, while the measured actual decoding time has strong variations for each VOP, the decoding time prediction is very accurate and the two curves for most of the samples are difficult to distinguish one from the other. In these experiments, the error of the prediction is lower than 5% for each VOP and in average is around 1-2%.

Ex. 1018-137

W (msec/f)	Akiyo	Bream	Coastguard
Intra	38.68	38.84	38.15
Inter	30.95	29.60	29.56
Skip	5.55	5.53	5.51
DCT_line	2.15	0.37	1.24
DCT_coeff	1.47	6.58	6.18
APM	226.47	230.23	228.70
NPM	36.30	37.60	36.01

Table 1. Estimated complexity weights vector w for the texture decoding complexity model of Figure 5 for different sets of measures. The complexity relevance for each weight is given by the product of the algorithm count and by the weight itself.

[0067] The excellent results demonstrate that a relatively simple complexity model is able to predict the decoding time (complexity) with a very good degree of approximation, when appropriate coding statistics are available. The degree of approximation resulting from the proposed method, indeed, is more than adequate to enable an intelligent and efficient resource allocation strategy.

[0068] The description and discussion of the new possible schemes of interaction between the real-time OS and the video decoder are the subject of next sections.

[0069] Another important conclusion is the fact that a very complex process such as MPEG-4 texture video decoding can be described by a suitable high level coding statistic information which is not directly related to any software or hardware implementation. The vector of coding mode counter values is, in fact, a compact coding parameter and coding mode information of the compressed bit-stream. Moreover, the obtained results reports that it is not necessary to define a very detailed model of the software/hardware decoder platform. Very complex decoder features such as cash strategy or memory accesses which could have a non-strictly deterministic behavior, can be taken into account in a statistical sense when one estimate the value of the weight vector characteristic of the decoder platform used.

8. NEW SCHEMES OF DECODER-OS SCHEDULER INTERACTIONS

[0070] New schemes of interaction between the decoder and the OS scheduler of the multimedia processor are possible when decoding time prediction is feasible. These schemes of interaction can be designed in order to provide the best resource allocation policy able to guarantee real-time performances.

[0071] In Figure 12, an example of decoding scheme employing decoding time prediction is reported. This figure represents a decoding scheme example in which the Complexity Information of each VOP is processed by the Complexity Prediction Unit which provides the Decoding Time Prediction signal to the decoder and to the real-time OS/scheduler that allocates the necessary resources (Allocated Decoding Time) to the decoder or decoding processes. The Decoding Time Prediction unit reads the Complexity Information Header carried in each VOP/frame bit-stream header or transmitted as side information in any format, produces the decoding time prediction for each frame/VOP, and provides this information to the real-time OS scheduler. In this way, the OS scheduler can allocate the necessary resources (i.e. Allocated Decoding Time) for the decoding of each or all the VOPs respecting the associated real-time constraints. In case of non real-time encoding, the complexity information can be compressed and transmitted before the real-time transmission and real-time decoding of the video/audio bit-streams. The usefulness of such scheme is evident considering that many VOPs of unknown complexity (decoding time) need to be decoded within an associated time interval

Ex. 1018-138

T_c. This scheme is able to guarantee real-time performances if the available video processing resources T_{Alloc} (i.e. Allocated Decoding Time) are sufficient for the decoding of all N incoming VOPs within the given time interval. Formally, the following relation has to be satisfied :

$$T_c \geq T_{Alloc} \geq \sum_{n=1}^{N} T_n(t) \tag{5}$$

where T_n is the decoding time of the nth VOP, with n=1,..., N, where N being the VO and VOP index and t being the VOP sampling time.

[0072] This simple procedure scheme is applicable in two cases. The first is when all the VOPs belongs to the same sampling time t (implicitly defined by the sampling rate or also carried by time stamps associated to each VOP). In this case, there is no additional delay introduced by the decoder. The second case occurs when the VOPs can also belong to sequential sampling periods, i.e. M sampling intervals of the reference time t denoted as Dt (also referred as different time stamps). In this case, an additional delay has to be considered and an output buffer needs to be added to the decoder :

$$T_c \geq T_{Alloc} \geq \sum_{n=1}^{N} \sum_{m=1}^{M} T_n(t+m\Delta t) \tag{6}$$

[0073] This second scheme enables to smooth out computational peaks on a larger number of VOPs and on a larger time interval than a single image sampling period Dt. In other words, expression (6) might be satisfied even when expression (5) is not for some values of the sampling time t. Obviously, this result is obtained at the cost of an additional delay and an additional output buffer.

[0074] When the available processing resources (i.e. time) are not sufficient to decode all VOPs, this simple strategy cannot guarantee the respect of the real-time constraints. In such cases, other form of interactions and new decoding techniques are needed.

[0075] The main idea is to reduce the complexity of the decoding in order to satisfy relation (5) and (6) accepting some degradation to the resulting image quality. These techniques are called here Computational Graceful Degradation (CGD).

[0076] A schematic diagram of the construction of the complexity information header at the encoder side is reported in Figure 13. The encoder, during the encoding process, increments the registers (counters) corresponding to a set of encoding algorithms/modes/parameters used for the actual VOP or frame compression. At the end of the process, the counters contains a compact encoding/decoding statistical information that is written in the Complexity Information Header.

9. COMPUTATIONAL GRACEFUL DEGRADATION (CGD)

[0077] The main idea of CGD is to make available results of poorer quality when the results of the desirable quality cannot be obtained in real time. When the complexity of the task exceeds the available processing power, real-time services, possibly of degraded quality, are nevertheless provided respecting all real-time constraints. This approach makes the guarantee of the correct scheduling of tasks possible, even when the computational system load fluctuates and relation (5) and (6) cannot be satisfied.

[0078] The concept of graceful system degradation using imprecise results has been previously introduced in the field of real-time computing with applications to control system and automation. In these contexts, imprecise results are defined as intermediate results with the property that the more recent and intermediate result is the closer one that approximates the final result (i.e. the idea only works in monotonically improving computations). For more details, the reader can refer to W.A. Halang, A.D. Stoyenko, Constructing predictable real time systems, Kluwer Academic Publisher, 1991 ; K. Lin, S. Natarjan, J. Liu, Imprecise results : Utilizing partial computations in real-time systems, Proceedings of IEEE Real-Time System Symposium, December 1987 ; J. Liu, K. Lin, S. Natarjan, Scheduling real time periodic jobs using imprecise results, Proceedings of the IEEE Real-Time System Symposium, December 1987 incorporated herein by reference.

[0079] In these cases the problems to be solved are more concerned with fault tolerance, I/O overloads, stability and reactivity to unexpected inputs, and the corresponding techniques are not suitable for purposes of real-time decoding of compressed video/audio bit-streams.

[0080] In contrast, the aim of the present invention is to be able to provide the guarantee of real-time video decoding

Ex. 1018-139

with possible degradations of the image quality, i.e. reductions of the peak signal to noise ratio (PSNR), without loosing parts of the image or parts of the audio content. Such alternative is certainly much more desirable than other straightforward options such as cutting image portions or simply skipping frames. These latter options, as can be easily understood, can have catastrophic consequences on prediction based coding systems. In H.263 systems, for instance, skipping the decoding of a single frame or a portion of it implies the skipping of all future incoming frames (or portion of them) before the next incoming Intra frame. The time interval of such skipping operation might cover many seconds, and the information content loss might be very relevant. Such abrupt interruptions, which are perceptually very annoying, might he caused not only by the simple fluctuation of the decoding complexity load as discussed in the previous sections, but also by the fluctuations of the load of other non-video/audio processing tasks which reduce the available processing resources (time). Moreover, such events are in general out of the control of the video/audio decoder system and cannot be easily forecasted. Considering these facts, the importance of techniques able to filter out complexity decoding peak loads or able to reduce the average decoding computation load without causing perceptually annoying consequences is evident. Skipping B-VOP could be, in some cases, a viable option for video accepting the introduction of motion judder in the sequence reproduction. Unfortunately, the presence of B-frames or B-VOP cannot be guaranteed in any bit-stream and just at the right time for which a reduction of the complexity is needed to respect the real time constraints. For audio applications, any skip of part of the content has perceptually a very annoying result.

[0081] In contrast with what could be commonly thought, also the classical multi-resolution coding features such as spatial or temporal scalabilities are not really useful for the aims of the present invention. In general, they are used to optimize the transmission bandwidth usage, enabling the compatibility of services at different bit-rates and spatial resolutions, but they must be embedded in the original bit-stream at the encoder side. If they are not embedded, the changes of resolution at the decoder size can results more computationally expansive than the simple decoding at the given resolution.

[0082] Another important reason for the introduction of CGD is given by the definition of profiles, levels and conformance tests for video compression standards based on software/processor platforms. For the MPEG-4 standard, it is expected to have a range of applications that will be much broader than the one of MPEG-1 and MPEG-2. Two possible scenarios can be identified. The first is the definition of a limited number of profiles and levels based on computational power. The second is the possibility of a continuum of decoders of different computational power. The first scenario faces very difficult tasks such as the definition of decoder conformance testing, and at the end it may result in a jungle of too many different levels and profiles.

[0083] The second scenario is completely open to platforms of different computational power. It requires pure functional decoder conformance testing and is able to provide the maximum quality of the service that is achievable by the capabilities of each platform.

[0084] The second scenario, although much more attractive than the first, poses new questions and problems have not yet been studied in video and audio coding. It is mandatory that all decoders can decode all MPEG-4 services but at different levels of quality. It means that a scalability-based on computational power is needed. This new type of scalability should enable a graceful degradation of the video quality when less than theoretically necessary computational power is available at the decoding side. In contrast with MPEG-2 scalabilities that are discrete, embedded in the bitstream and designed for specific applications, this computational power based scalability should be continuous and implemented in all MPEG-4 bit-streams.

[0085] From these arguments, the usefulness of CGD and its important features and applications are clear. The last key point that has left CGD as a good idea but of difficult usage and implementation for video coding was that it was not clear how to implement it efficiently. Now that the present invention has developed a technique able to predict with a good degree of approximation the required decoding time of each decoding process, one have the key for useful and efficient implementation of CGD techniques.

9.1 Principles of decoding using CGD techniques

[0086] Let us consider a complexity model such as the one of Figure 3 composed by a set of I algorithms fi, with i = I,...,I and suppose to be able to design for all, or only for a subset S^{CGD} of the functions f_i composing the model, new functions f_i^{CGD} which have the following three properties :

1) the execution of a function f_i^{CGD} yields results that are an approximation of the ones produced by the original function f_i,
2) the execution of f_i^{CGD} does not prevent the exact execution of any subsequent decoding functions,
3) the weight w_i^{CGD} associated to f_i^{CGD} in the considered decoder implementation is lower than the corresponding w_i associated at f_i (i.e. $w_i^{CGD} < w_i$).

[0087] For a schematic representation of a complexity model which includes new degraded functions, one can refer

Ex. 1018-140

to Figure 15 which represents a schematic diagram of a complexity model including CGD algorithms, CGD counters (b_i) and new state transitions generated by the new CGD functions. In this example $f_1, f_3 \hat{I} S^{CGD}$, and $f_2 \hat{I} S^{orig}$. In other words, for video decoding for instance, the execution of any f_i^{CGD} might produces PSNR degradations in the reconstruction of some image parts, but not prevents from correctly decoding the rest of the frame or VOP.

[0088] Let us partition the functions forming the complexity model into two disjoint sets, S^{Orig} and S^{CGD} respectively, for the ones that do not admit a CGD version and for the ones for which it is possible to design a CGD version.

[0089] Let us consider for simplicity an image sequence composed by a sequence of single frames or single VOPs (i.e. N=1) and suppose that expression (5) is not satisfied for some values of the decoder reference time t :

$$T(t) > T_{Alloc} \tag{7}$$

where T(t) is the decoding time of the VOP at hand at the sampling time t, and to define as processing deficit T_{Def} the time for which the decoding complexity exceeds the Allocated Decoding Time :

$$T(t) - T_{Alloc} = T_{Def} \tag{8}$$

[0090] Since the decoding time can be approximated by its prediction (Decoding Time Prediction) it can be expressed by :

$$T(t) \approx T_{Pred}(t) = \sum_{i=1}^{I} c_i w_i \tag{9}$$

where the c_i are the counters of the complexity model and the w_i are the corresponding complexity weights.

[0091] The lower limit of the decoding time executing all CGD algorithms f_i^{CGD} at the place of f_i for any $i \hat{I} S^{CGD}$, is given by :

$$T_{CGD} = \sum_{i \in S_{CGD}} c_i w_i^{CGD} + \sum_{i \in S_{Orig}} c_i w_i \tag{10}$$

[0092] If $T_{CGD} < T_{Alloc} < T_{Pred}$ it is therefore possible to guarantee the real-time constraints by executing the f_i^{CGD} algorithms and accepting approximate decoding results. The difference $T_{Pred} - T_{CGD}$ can also be expressed as :

$$T_{Pred} - T_{CGD} = \sum_{i \in S_{CGD}} c_i (w_i - w_i^{CGD}) \tag{11}$$

[0093] If $T_{CGD} < T_{Alloc} < T_{Pred}$, it is therefore possible to find for each c_i with $i \hat{I} S^{CGD}$, a couple of integer coefficients a_i, b_i, for which :

$$a_i + b_i = c_i \tag{12}$$

and denoting as $T_{Process}$ any decoding time interval which respects the real time constraints (obtained by the implicit sampling rate or expressed using appropriate time stamps associated to each VOP or video frames, or audio packet), one can express the desired process decoding time as any decoding time value that satisfy the expression :

$$T_{process} = \sum_{i \in S_{CGD}} (a_i w_i^{CGD} + b_i w_i) + \sum_{i \in S_{Orig}} c_i w_i \leq T_{Alloc} \tag{13}$$

Ex. 1018-141

[0094] Obviously for any Allocated Decoding Time the aim is to respect the real-time constraints with the lowest degradation in the video/audio content. Associating a degradation measure to each w_i^{CGD} it is possible to minimize the degradation for a given TAlloc. Therefore, the aim is to find the set $S_{a,b}$ of couple of coefficients a_i, b_i, with i Î S^{CGD}, that satisfy expression (12) and minimize the video or audio degradations expressed as difference with the non degraded output :

$$S_{a,b}=\text{argmin}(a,b)_{i\in S_{CGD}}:(PSNR-PSNR(a_i,b_i)) \tag{14}$$

[0095] The determination of the set $S_{a,b}$ can be done by means of any suitable optimization method or any suitable optimization heuristic by the Video Decoder Supervisor, schematically represented in Figure 14. In summary, the Video Decoder Supervisor receives from the OS the Allocated Decoding Time signal (T_{Alloc}), the decoding time prediction signal (TPred) from the Complexity Prediction unit, and the Complexity Information signal represented by the actual values of the coding/decoding statistic (set of ci coefficients) for the VO. Since the Video Decoder Supervisor knows the values of the set of w_i and w_i^{CGD} it can determine, by a suitable standard optimization technique, the corresponding set of coefficients a_i,b_i such that expression (12) is satisfied. Having defined the set of a_i,b_i coefficients the CGD decoding is partially defined. In other words, the set of coefficients defines for each i Î S^{CGD} how many degraded algorithms (f_j^{CGD}) at the place of original algorithms (f_j) need to be executed.

[0096] Since the determination of the degradation coefficients is done on the basis of the decoding time prediction, the video decoder can generate a Decoding Status signal reporting the status of the decoding (i.e. the fraction of the bit-stream or the fraction of the VO already decoded). The signal representing the Decoding Status of the decoding process reports to the Video Decoder Supervisor an information which can be used to correct the Computational Graceful Degradation Directive signal (i.e. the setting of the a_j,b_j coefficients in this example) if the Current Decoding Time and the Current Decoding Time Prediction signal are not conformant. Current Predicted Decoding Time means the Decoding Time Prediction calculated by using elation (13), only counting the occurrences of the decoding algorithms actually used to decode the fraction of the decoded bit-stream at the current reference time (time for which the decoding status signal is generated). The term not conformant means that the actual Decoding Time is greater than the Decoding Time Prediction of an amount of time which overcomes statistical variations which are functions of the complexity information, thus reporting possible errors in the Decoding Time Prediction signal.

[0097] The described interactions between OS, the Decoder and Video Decoder Supervisor can be easily extended for the case of sequences containing multiple VO at the same sampling time (associated time stamps), or belonging to a defined time interval containing several sampling periods (several associated time stamps). Expressions (10,11,13,14) can be easily modified including the appropriate summation over all VOP. Therefore, in the case of N VOP at the same sampling time t, one have :

$$T_N(t)=\sum_{n=1}^{N} T_n(t)>T_{Alloc} \tag{15}$$

where T_N (t) is the total decoding time of the N VOP.

$$T_N(t)\cong T_{Pred,N}=\sum_{n=1}^{N} T_{Predn}(t)=\sum_{n=1}^{N}\sum_{i=1}^{I} c_{i,n}w_i \tag{16}$$

$$T_{CGD,N}=\sum_{n=1}^{N}\Big(\sum_{i\in S_{CGD}} c_{i,n}w_i^{CGD}+\sum_{i\in S_{Orig}} c_{i,n}w_i\Big) \tag{17}$$

$$T_{Pred,N}-T_{CGD,N}=\sum_{n=1}^{N}\sum_{i\in S_{CGD}} c_{i,n}(w_i-w_i^{CGD}) \tag{18}$$

$$a_{i,n}+b_{i,n}=c_{i,n} \tag{19}$$

$$T_{Process,N}=\sum_{n=1}^{N}\left(\sum_{i\in S_{CGD}}(a_{i,n}w_i^{CGD}+b_{i,n}w_i)+\sum_{i\in S_{Orig}}c_{i,n}w_i\right)\leq T_{Alloc} \tag{20}$$

$$S_{a,b,N}=\operatorname{argmin}_{(a,b)\in S_{CGD}}\left(\sum_{n=1}^{N}(PSNR-PSNR(a_{i,n},b_{i,n}))\right) \tag{21}$$

or :

$$S_{a,b,N}=\operatorname{argmin}_{(a,b)\in S_{CGD}}\left(\sum_{n=1}^{N}\alpha_n(PSNR-PSNR(a_{i,n},b_{i,n}))\right) \tag{22}$$

where a_n is a semantic degradation coefficient which might be available in the coded bitstreams.

[0098] For N Video Objects and M sampling time intervals Dt (denoted also as time stamps) one have :

$$T_{N,M}=\sum_{m=1}^{M}T_N(t+m\Delta t)=\sum_{m=1}^{M}\sum_{n=1}^{N}T_n(t+m\Delta t)>T_{Alloc} \tag{23}$$

$$T_{N,M}\cong T_{Pred,N,M}=\sum_{m=1}^{M}\sum_{n=1}^{N}T_{Predn}(t+m\Delta t)=\sum_{m=1}^{M}\sum_{n=1}^{N}\sum_{i=1}^{I}c_{i,n,m}w_i \tag{24}$$

$$T_{CGD,N,M}=\sum_{m=1}^{M}\sum_{n=1}^{N}\left(\sum_{i\in S_{CGD}}c_{i,n,m}w_i^{CGD}+\sum_{i\in S_{Orig}}c_{i,n,m}w_i\right) \tag{25}$$

$$T_{Pred,N,M}-T_{CGD,N,M}=\sum_{m=1}^{M}\sum_{n=1}^{N}\sum_{i\in S_{CGD}}c_{i,n,m}(w_i-w_i^{CGD}) \tag{26}$$

$$a_{i,n,m}+b_{i,n,m}=c_{i,n,m} \tag{27}$$

$$T_{Process,N,M}=\sum_{m=1}^{M}\sum_{n=1}^{N}\left(\sum_{i\in S_{CGD}}(a_{i,n,m}w_i^{CGD}+b_{i,n,m}w_i)+\sum_{i\in S_{Orig}}c_{i,n,m}w_i\right)\leq T_{Alloc} \tag{28}$$

Ex. 1018-143

$$S_{a,b,N,M} = \mathrm{argmin}_{(a,b)_i \in S_{CGD}} \left(\sum_{m=1}^{M} \sum_{n=1}^{N} \left(PSNR - PSNR(a_{i,n,m}, b_{i,n,m}) \right) \right) \quad (29)$$

or

$$S_{a,b,N,M} = \mathrm{argmin}_{(a,b)_i \in S_{CGD}} \left(\sum_{m=1}^{M} \sum_{n=1}^{N} \alpha_n \left(PSNR - PSNR(a_{i,n,m}, b_{i,n,m}) \right) \right) \quad (30)$$

9.2 Extension of the techniques to video/audio composition

[0099] The methods aiming at predicting the decoding time (i.e. generating a Decoding Time prediction Signal) of video frames or video objects, so as to respect the real-time constraints specified implicitly by the audio/video rates or by the timing information carried by the associated time stamps, can be easily extended to the video/audio composition processes. Composition process means the action of putting decompressed video or audio objects together in the same representation space. In a similar way of the decoding process, one can generate a Composition Time Prediction for each composition process according to the interactions described in Figure 12. Such Composition Time Prediction is a function of the occurrence of composition algorithms, composition modes and composition parameters (associated composition statistics). Differently from the decoding process, composition algorithms, modes and parameters can be also provided by an interactive user interface. In a similar way the Composition Allocated Time can be generated as function of the Composition Time Prediction. The interactions between the compositor and the OS can be described as in Figure 12 where the Decoder is substituted by the Compositor.

[0100] The technique of associating Decoding Degradation Algorithms and decoding by using degradation algorithms described in section 6.1 and Figure 14 can be applied by substituting respectively the Allocated Decoding Time by the Decoding Time Prediction, the Target Decoding Time and the Decoder Reference Time by the Allocated Composition Time, the Composition Time Prediction by the Target Composition Time and the Compositor Reference Time. All relations and equations reported in section 6.1 referring to the decoding process can be referred to the composition process.

9.3 Examples of possible degradation modes

[0101] The decoding operations of a video compression algorithm can be divided into two main categories :

1) operations that cannot be simplified since they may prevent the extraction and usage of relevant coding information,
2) operations that can be simplified, possibly leading to imprecise or degraded video results.

[0102] To the first category typically belong all bit-stream parsing operations, entropy decoding (i.e. VLD and arithmetic coding), and all operations related to synchronization and to the syntax of the decoding process. For these operations, no degradation mode definition is possible, otherwise it would imply skipping image portions or entire frames since either the data or the coding modes and coding parameters are lost.

[0103] To the second category, for instance, belong operations such as the IDCT, the prediction with and without motion compensation in its various forms (PB/B-frame prediction, overlapped block motion compensation (OBMC), half pixel interpolation and so on, padding, shape coding. Simplifications of such operations applied to pixels or transform coefficients are always possible but at the expenses of imprecise results. The challenge can be expressed as follows : given a specific compression standard, software and processor architecture, implement it and find the f_i^{CGD} which provides the maximum complexity reduction (w_i - w_i^{CGD}) versus the minimum image/audio quality degradation (usually measured in PSNR). Many are the algorithms that can be simplified. We report here only two examples that illustrate the idea.

9.3.1 IDCT simplified modes

[0104] IDCT on a 8x8 block can be decomposed into 16 IDCT operation on a 8x1 block, or on IDCT considering only

Ex. 1018-144

the lowest 4x4 or 2x2 coefficients. Skipping higher order 8x1 IDCT or enabling processing only 2x2 or 4x4 coefficients constitute some alternatives which are less complex than the standard 8x8 IDCT. Obviously, these less demanding alternatives yield a filtering of the high frequency possibly contained in the higher order coefficients.

9.3.2 Simplified prediction modes

[0105] Many prediction modes can be simplified at the expenses of some PSNR degradations. In particular, the most complex such as the OBMC with or without half-pel motion compensation can yield relevant complexity decreases of both computational complexity and memory bandwidth by implementing simpler algorithms.

10. EXAMPLES OF RESULTS FOR THE MPEG-4 VIDEO COMPRESSION STANDARD

[0106] This section reports simulation results of the decoding of VOP using the complexity model and decoding time prediction results described in Section 4. The MPEG-4 VM6 algorithm in the Momusys VM6 implementation modified adding degradation functions to the advanced prediction modes has been used for the simulations. A fixed amount of processing time is allocated for each VOP composing the sequences. In case the decoding time prediction (T_{Pred}) for each VOP exceeds the fixed allocated time (T_{Alloc}), an appropriate number of advanced prediction modes are executed using degraded functions in order to satisfy relation (13).

[0107] Figures 16, 17, 18 report the decoding results at 64, 128 and 256 kb/s for the four VO (Small Boat, Big Boat, background and Water) of the sequence Coastguard. On the left, the diagrams report the decoding time of each VO expressed in msec versus the VOP number. Two curves are reported : original decoding and decoding using degradation function in order to satisfy the decoding time limits. The PSNR image degradations corresponding to the execution of degraded functions are reported on the right.

[0108] Figure 16 reports on the left standard decoding time and CGD decoding time (in msec) for the four VO of the sequence Coastguard at 64 kb/s. Each VO has a upper bound for the CGD decoding time. On the right the PSNR loss of the CGD decoding with reference to the standard decoding.

[0109] Figure 17 reports on the left standard decoding time and CGD decoding time (in msec) for the four VO of the sequence Coastguard at 128 kb/s. Each VOP has a upper bound for the CGD decoding time. On the right the PSNR loss of the CGD decoding with reference to the standard decoding.

[0110] Figure 18 reports on the left standard decoding time and CGD decoding time (in msec) for the four VO of the sequence Coastguard at 256 kb/s. Each VO has a upper bound for the CGD decoding time. On the right the PSNR loss of the CGD decoding with reference to the standard decoding.

[0111] The implementation of CGD base on the decoding time prediction generation technique enables, within the discussed limits, the a priori setting of the decoding time guaranteeing the possibility of a scheduling of each decoding task that satisfy real time constraints, even if the available resources (Allocated Decoding Time) are inferior to the theoretically necessary ones. Moreover, the degradations introduced are visually acceptable and they can be minimized according to expression (14) by suitable techniques. The results obtained with a PSNR degradated are certainly much more desirable than introducing sudden interruptions and image content losses to the video sequence, and enables an efficient resource allocation policy.

Claims

1. Method for primary processing a digital signal comprising the steps of :

   - providing a digital signal to a primary processing unit ;
   - primary processing said signal with said primary processing unit according to a plurality of primary processing algorithms to provide primary processed output signals,
   - determining statistics of use of at least one of said primary processing algorithms,
   - providing a digital statistics signal representative of said statistics for each of said primary processed output signals
   - associating to each of said primary processed output signals its own statistics signal ;
   - determining a complexity primary processing information signal based on said statistic signal.

2. Digital primary processed output signal, generated by a primary processing unit using primary processing algorithms and being intended to be transmitted to a receiving device wherein said primary processed output signal comprises data bit streams, each of said data bit streams comprising a main bit string representative of said primary processed output signal and a secondary bit string representative of the statistics of use of said primary processing algorithms associated to said main bit string.

Ex. 1018-145

3. Method for secondary processing a secondary digital output signal according to claim 2 comprising the steps of :

(a) reading said secondary bit string ;
(b) determining, from the secondary bit string, primary processing algorithms to be used to secondary process said primary output signal associated to said main bit string, for generating a complexity primary processed information signal for each signal ;
(c) allocating a secondary processing time and/or operating system resources as a function of said complexity primary processed information signal for secondary processing of said primary output signal ;
(d) sending main bit string to secondary processing means ;
(e) secondary processing said main bit string using said allocated secondary processing time and/or operating system resources, according to a secondary process and
(f) generating a secondary processed signal.

4. Method for coding a digital signal comprising the steps of :

- providing an uncompressed digital signal including audio packets, video frames, video planes, or audio/video objects to an encoder ;
- compressing said signal with said encoder according to a compression syntax, said encoder using a plurality of coding algorithms to provide a compressed signal including coded audio packets, video frames, video planes or audio/video objects,
- determining statistics of use of at least one of decoding algorithms necessary to decode each of said coded audio packets, video frames, video planes or audio/video objects from said associated coding algorithms, and/or determining statistic of use of at least one of coding algorithms,
- providing a digital statistics signal representative of said statistics for each of said coded audio packets, video frames, video planes or audio/video objects,
- associating to each of said coded audio packets, video frames, video planes or audio/video objects its own statistics signal.

5. Coding method according to claim 3, wherein the step of determining statistics consists in determining the statistics of use of some of decoding algorithms, wherein to each decoding algorithm is associated at least one decoding mode and/or decoding parameter and wherein the statistics signal is based on the decoding modes and/or decoding parameters.

6. Coding method according to claim 3 or 4, wherein the step of determining statistics further consists in determining the statistics of use of some of coding algorithms when said coding algorithms and the compression syntax allow the determination of corresponding decoding algorithms necessary to decode each of said coded audio packets, video frames, video planes or audio/video objects wherein to each coding algorithm is associated at least one coding mode and/or coding parameter and wherein the statistics signal is based on the coding modes and/or coding parameters necessary to decode when using said coding algorithms.

7. Coding method according to claim 4, wherein the statistic signal is generated by juxtaposing digital words each representing the occurrences of use of a decoding mode and/or decoding parameter necessary to decode said associated audio packet, video frame, video plane or audio/video object.

8. Coding method according to claim 5 and/or 6, wherein the statistic signal is generated by juxtaposing digital words each representing the occurrences of use of a coding mode and/or coding parameter necessary to decode said associated audio packet, video frame, video plane or audio/video object.

9. Coding method according to any of claims 3 to 7, wherein the step of associating consists in inserting said statistics signal into said compressed signal for each associated coded audio packet, video frame, video plane or audio/video object.

10. Coding method according to any of claims 3 to 8, wherein the step of determining statistics consists in computing the occurrence of use of said coding and/or decoding algorithms for determining a complexity information for each coded audio packet, video frame, video plane or audio/video object.

11. Digital compressed signal including coded audio packets, video frames, video planes or audio/video objects, said signal being generated by an encoder using a compression syntax and coding algorithms and being intended to be

Ex. 1018-146

transmitted to a receiving device wherein said signal is decoded according to decoding algorithms, and wherein said signal comprises at least a layer of coded audio packets, video frames, video planes or audio/video objects bit streams, each of said bit streams comprising a main bit string representative of the coded audio packets, video frames, video planes or audio/video objects and a secondary bit string representative of the statistics of use of said coding and/or decoding algorithms associated to said main bit string.

12. Digital compressed signal, according to claim 10, wherein said secondary bit string is inserted in said layer of said bit streams.

13. Digital compressed signal, according to claim 10, further comprising a plurality of parallel layers of coded audio packets, video frames, video planes, or audio/video objects bit streams, wherein said main bit string is inserted or transmitted in a layer different to said second bit string.

14. Digital compressed signal, according to claim 11 or 12, wherein said secondary bit string is inserted or transmitted prior to said main bit string.

15. Digital compressed signal, according to any of claims 10 to 13, wherein to each decoding algorithm is associated at least one decoding mode and/or decoding parameter and wherein said secondary bit string consists of words each representing the occurrences of use of the decoding modes and/or decoding parameters.

16. Digital compressed signal, according to claim 10 or 14, wherein to each coding algorithm is further associated at least one coding mode and/or coding parameter and wherein said secondary bit string consists of words each representing the occurrences of use of the coding modes and/or coding parameters, said coding algorithms and the compression syntax allowing the determination of corresponding decoding algorithms necessary to decode each of said coded audio packets, video frames, video planes or audio/video objects.

17. Method for decoding a digital compressed signal according to any of claims 10 to 15 comprising the steps of :

(a) reading said secondary bit string ;
(b) determining, from the secondary bit string, decoding algorithms to be used to decode said coded audio packets, video frames, video planes or audio/video objects, associated to said main bit string, for generating a complexity information signal for each coded audio packet, video frame, video plane or audio/video object ;
(c) allocating a decoding time and/or operating system resources as a function of said complexity information signal for the decoding process of said compressed audio packet, video frames, video planes or audio/video objects ;
(d) sending main bit string to decoding means ;
(e) decoding said main bit string using said allocated decoding time and/or operating system resources, according to a decoding process and
(f) generating a decompressed signal including decoded audio packets, video frames, video planes or audio/video objects.

18. Method for decoding a digital compressed signal according to claim 16 wherein the step determining decoding algorithms is also based on the occurrence of use of the decoding modes and/or decoding parameters necessary to decode the said audio packets, video frames, video planes or audio/video objects.

19. Method for decoding a digital compressed signal according to claim 16 or 17 wherein the step determining decoding algorithms is further based on the occurrence of use of the coding modes and/or coding parameters used by said coding algorithms, wherein said coding algorithms and the compression syntax allow the determination of corresponding decoding algorithms necessary to decode each of said coded audio packets, video frames, video planes or audio/video objects

20. Decoding method according to any of claims 16 to 18 further comprising the step of determining a decoding time prediction signal as a function of said complexity information signal ; said decoding time prediction signal being used for the step of allocating said decoding time and/or operating system resources for optimizing said decoding process and or decoding processes.

21. Decoding method according to claim 19 wherein the step of decoding the main bit string consists in

Ex. 1018-147

- generating an allocated decoding time signal,
- sending said decoding time prediction signal and allocated decoding time signal to decoder supervisor means ;
- generating and sending a degradation directive signal to said decoding means, as a function of said decoding time prediction signal, allocated decoding time, so as to define the use of number and type of decoding algorithms and degradation algorithms to be used in the decoding process of said main bit string.

22. Decoding method according to claim 20 wherein the step of decoding the main bit string further consists in :

- generating a decoder time reference signal and a target decoding time signal,
- sending said decoder time reference signal and said target decoding time signal to decoder supervisor means ;
- generating and sending said degradation directive signal to said decoding means while also taking into account said decoder time reference signal and said target decoding time signal.

23. Decoding method according to claim 20 or 21 wherein the step of decoding the main bit string further consists in :

- generating a decoding status signal from said decoding means
- sending said decoding status signal to said decoder supervisor means from the said decoding means ;
- generating and sending said degradation directive signal to said decoding means while also taking into account said decoding status signal.

24. Decoding method according to claim 16 wherein steps (a) to (c) are carried out prior to step (e).

25. Decoding method according to any of claims 20 to 23, further comprising the step of sending said complexity information signal to said decoder supervisor means.

26. Decoding method according to any of claims 20 to 24 wherein the step of generating said degradation directive signal consists in associating at least one degradation algorithm to at least one decoding algorithm, said decoding algorithm and said degradation algorithm being associated to a complexity coefficient/weight representative of the execution time of said decoding algorithm and said degradation algorithm.

27. Decoding method according to claim 25 wherein said complexity coefficients/weights are dependent on the decoder used for implementing said decoding method.

28. Decoding method according to claim 25 or 26, wherein said decoding time prediction is a function of the occurrence of use of said decoding algorithm and said complexity coefficients/weights according to the following equation :

$$T(t) \cong T_{Pred}(t) = \sum_{i=1}^{I} c_i w_i \qquad (9),$$

where T is the time necessary to decode a coded audio packet, video frame, video plane or audio/video object, T_{Pred} is the decoding time prediction of said audio packets, video frames, video planes or audio/video objects, c_i is the occurrence of said decoding algorithm, and w_i is the complexity coefficient/weight.

29. Decoding method according to claim 27, wherein when the decoding time prediction is a function of the occurrence of use of said decoding algorithms and degradation algorithms, said degradation algorithms being associated to a complexity coefficient/weight.

30. Decoding method according to claim 27 or 28, wherein when the decoding time prediction is greater than the allocated decoding time, one uses a degradation method according to which one selects the degradation algorithm instead of its associated decoding algorithm according to equation :

$$T_{process} = \sum_{i \in S_{CGD}} (a_i w_i^{CGD} + b_i w_i) + \sum_{i \in S_{Orig}} c_i w_i \leq T_{Alloc} \qquad (13)$$

where, $T_{Process}$ is the prediction time necessary to decode a audio packets, video frames, video planes or audio/video objects, said decoding time prediction taking account the degradation, c_i is the occurrence of decoding algorithm, a_i is the number the selected degradation algorithms, w_i^{CGD} is the complexity coefficient/weight of the degradation algorithm, b_i is the number of the non selected degradation algorithms, and w_i is the complexity coefficient/weight, where $a_i + b_i = c_i$, so as to reach a $T_{Process}$ lesser than or equal to said allocated decoding time.

31. **30.** Decoding method according to claim 28 or 29, wherein said decoding status signal is also used for monitoring the current decoding time of a coded audio packets, video frames, video planes or audio/video objects and make sure that said current decoding time is in conformity with the associated decoding time prediction.

32. Decoding method according to claim 30, wherein if said current decoding time is not in conformity with the associated predicted decoding time, then said decoder supervisor means uses at least one further degradation algorithm defined in claim 29.

33. Decoding method according to claim 31, wherein if said current decoding time is not in conformity with the associated decoding time prediction, then said decoder supervisor means further uses said complexity information signal.

34. Decoding method according to any of claims 29 to 32 for a plurality of audio packets, video frames, or video planes, some of them belonging to different audio/video objects, wherein when the decoding time prediction is the sum of the decoding time prediction of each of said audio packets, video frames or video planes of said plurality, said decoding time prediction being defined in claim 28.

35. Coding system for an uncompressed digital signal including audio packets, video frames, video planes or audio/video objects, comprising :

- an audio/video encoder receiving said audio packets, video frames, video planes or audio/video objects on one input terminal, said encoder being intended to code said signal according to a compression syntax by using a plurality of coding algorithms, and to provide, on one output terminal, a compressed signal including coded audio packets, video frames, video planes or audio/video objects intended to be decompressed according to associated decoding algorithms ;
- means for determining statistic of use of at least one of said coding algorithms, and/or means for determining statistic of use of at least one of said decoding algorithms ;
- and means for generating a digital statistic signal representative of said statistics for each of said coded audio packets, video frames, video planes or audio/video objects.

36. Coding system according to claim 27, wherein said statistics determining means comprise computing means for counting the occurrence of use of coding modes and/or coding parameters used by the encoder for encoding said signal, when using said coding algorithms.

37. Coding system according to claim 34 or 35, wherein said statistics determining means further comprise computing means for counting the occurrence of use of decoding modes and/or decoding parameters from said coding algorithms used by the encoder for encoding said signal.

38. Coding system according to any of claim 34 to 36, further comprising means for generating a compressed output signal including coded audio packets, video frames, video planes or audio/video objects wherein each of said coded audio packet, video frame, video plane or audio/video object is associated with its statistics signal.

39. Decoding system for a digital signal including coded audio packets, video frames, video planes or audio/video objects according to any of claims 5 to 15, comprising :

- a decoder receiving said main bit string, representative of a coded audio packets, video frames, video planes or audio/video objects, on a first input terminal and providing, on an output terminal, a decompressed signal including decoded audio packets, video frames, video planes or audio/video objects,
- resource allocation means ;
- means for complexity prediction receiving, on one input terminal, said secondary bit string, and providing to resource allocation means a decoding time prediction signal which is determined as a function of the information contained in said secondary bit string, said resource allocation means providing to the decoder information representative of the allocated decoding time.

Ex. 1018-149

40. Decoding system according to claim 38 wherein said resource allocation means comprises an operating system or a layer of an operating system.

41. Decoding system according to claim 38 or 39 wherein said decoding time prediction signal is also provided to a second input terminal of said decoder.

42. Decoding system according to any of claims 38 to 40, wherein

- said decoder comprises decoder supervisor means and audio/video decoding means,
- said decoder supervisor means receiving, on a first input terminal, said allocated decoding time signal and, on a second input terminal, said predicted decoding time signal and providing on one output terminal a degradation directive signal,
- said decoding means receiving on a first input terminal said main bit string and on a second input terminal said degradation directive signal and providing on an output terminal said decompressed signal including decoded audio packets, video frames, video planes or audio/video objects.

43. Decoding system according to claim 41 wherein said decoder further receive a complexity information signal on third input terminal.

44. Decoding system according to claim 41 wherein said decoder further receive a decoder time reference signal and a target decoding time signal respectively on fourth and fifth input.

45. Decoding system according to any of claims 41 at 43 wherein said decoding means are arranged to provide a decoding status signal to a sixth input of the decoder supervisor means.

46. Decoding system according to claim 44 wherein said complexity information signal is applied to said decoder supervisor means.

47. Method for composing a digital signal comprising the steps of :

- providing an uncompressed video element including video frames and/or video planes and/or video objects, textures, texts, 2 and 3 dimensional graphic primitives, 2 and 3 dimensional models to a composing device ;
- composing said video element with said composing device according to a composition syntax an/or a interactivity information, said composing device using a plurality of composing algorithms necessary to compose each of said video elements and provide a representation in a common visual representation space of the said multiple elements,
- determining statistics of use of at least one of said composing algorithms for generating a complexity information
- associating to each of said video elements its own complexity information.

48. Composition method according to claim 46 further consisting in :

- providing an uncompressed digital audio element including audio packets, audio objects, synthetic audio objects
- composing said audio element with said composing device according to a composition syntax, said audio composing device using a plurality of composing algorithms necessary to compose each of said audio elements and provide a representation in a common audio representation space
- determining statistics of use of at least one of said algorithms for generating a complexity information.
- associating to each of said audio element its own complexity information.

49. Composition method according to claim 46 or 47, wherein to each composition algorithm is associated at least one composition mode and/or composition parameter and wherein the statistics is based on the occurrence of use of composition modes and/or composition parameters.

50. Method for composing a digital signal according to any of claims 46 to 48 comprising the steps of :

(a) reading the complexity information associated to at least one of said video element and/or audio element
(c) allocating a composition time and/or operating system resources as a function of said complexity informa-

Ex. 1018-150

tion signal for the composition process of said associated video element and/or audio element ;

(d) sending said video element and/or audio element to composition means ;

(e) compose in a common representation space said video element and/or audio element using said allocated composition time and/or operating system resources, according to a composition process, and

(f) generating a common representation space including said video element and/or audio element according to the composition syntax.

51. Method for composing digital video/audio elements according to claim 49 wherein the step determining composition algorithms is also based on the occurrence of use of the composition modes and/or composition parameters necessary to compose the said composition elements in a common composition space.

52. Composition method according to any of claims 48 to 50 further comprising the step of determining a composition time prediction signal as a function of said complexity information ; said composition time prediction signal being used for the step of allocating said composition time and/or operating system resources for optimizing said composition process and or composition processes.

53. Composition method according to claim 50 wherein the step of composing the audio/video elements consists in

- generating an allocated composition time signal,
- sending said composition time prediction signal and allocated composition time signal to composition supervisor means ;
- generating and sending a degradation directive signal to said composition means, as a function of said composition time prediction signal, allocated composition time, so as to define the use of number and type of composition algorithms and degradation algorithms to be used in the composition process of said video/audio composition elements.

54. Composition method according to claim 52 wherein the step of composing the video/audio elements further consists in :

- generating a compositor time reference signal and a target composition time signal,
- sending said compositor time reference signal and said target composition time signal to composition supervisor means ;
- generating and sending said degradation directive signal to said composition means while also taking into account said compositor time reference signal and said target composition time signal.

55. Composition method according to claim 52 or 53 wherein the step of composing the audio/video elements further consists in :

- generating a compositor status signal from said composition means
- sending said compositor status signal to said composition supervisor means from the said composition means ;
- generating and sending said degradation directive signal to said means while also taking into account said compositor status signal.

56. Composition method according to claim 48 wherein steps (a) to (c) are carried out prior to step (e).

57. Composition method according to any of claims 51 to 55, further comprising the step of sending said complexity information signal to said compositor supervisor means.

58. Decoding method according to any of claims 53 to 56 wherein the step of generating said degradation directive signal consists in associating at least a degradation algorithm to at least one composition algorithm, said composition algorithm and said degradation algorithm being associated to a complexity coefficient/weight representative of the execution time of said composition algorithm and said degradation algorithm.

59. Composition method according to claim 57 wherein said complexity coefficients/weights are dependent on the compositor used for implementing said composition method.

60. Composition method according to claim 57 or 58, wherein said composition time prediction signal is a function of

Ex. 1018-151

the occurrence of use of said composition algorithms and said complexity coefficients/weights according to the following equation :

$$T(t) \simeq T_{Pred}(t) = \sum_{i-1}^{I} c_i w_i \qquad (9),$$

where T is the time necessary to compose a audio/video element, T_{Pred} is the composition time prediction of said audio/video element, c_i is the occurrence of said composition algorithm, and w_i is the complexity coefficient/weight.

61. Composition method according to claim 59, wherein when the composition time prediction is a function of the occurrence of use of said composition algorithms and degradation algorithms, said degradation algorithms being associated to a complexity coefficient/weight.

62. Composition method according to claim 59 or 60, wherein when the composition time prediction is greater than the allocated composition time, one uses a degradation method according to which one selects the degradation algorithm instead of its associated composition algorithm according to equation :

$$T_{process} = \sum_{i \in S_{CGD}} (a_i w_i^{CGD} + b_i w_i) + \sum_{i \in S_{Orig}} c_i w_i \leq T_{Alloc} \qquad (13)$$

where, $T_{Process}$ is the prediction time necessary to compose a audio/video element, said composition time prediction taking account the degradation, c_i is the occurrence of composition algorithm, a_i is the number the selected degradation algorithms, w_i^{CGD} is the complexity coefficient/weight of the degradation algorithm, b_i is the number of the non selected degradation algorithms, and w_i is the complexity coefficient/weight, where $a_i + b_i = c_i$, so as to reach a $T_{Process}$ lesser than or equal to said allocated composition time.

63. Composition method according to claim 60 or 61, wherein said composition status signal is also used for monitoring the current composition time of a composition video/audio element and make sure that said current composition time is in conformity with the associated composition time prediction.

64. Composition method according to claim 62, wherein if said current composition time is not in conformity with the associated composition time prediction, then said composition supervisor means uses at least one further degradation algorithm defined in claim 61.

65. Composition method according to claim 63 wherein if said current composition time is not in conformity with the associated composition time prediction, then said decoder supervisor means further uses said complexity information signal.

66. Composition method according to any of claims 51 to 63 for a plurality of audio/video composition elements wherein when the composition time prediction is the sum of the composition prediction time of each of said audio/video composition elements, said composition time prediction being defined in claim 50.

67. Composition system for an uncompressed digital video element including video frames and/or video planes and/or video objects, textures, texts, 2 and 3 dimensional graphic primitives, 2 and 3 dimensional models, comprising :

- a video compositor receiving said video element on one input terminal, said compositor being intended to compose said video element using a composition syntax using a plurality of composing algorithms, and to provide, on one output terminal, a representation in a common video representation space said video element ;
- means for determining statistic of use of at least one of said composition algorithms,
- and means for generating a digital statistic signal representative of said statistics for each of said composed video element.

68. Composition system for an uncompressed digital audio element including audio packets, audio objects, synthetic audio objects, comprising :

Ex. 1018-152

- a audio compositor receiving said audio element on one input terminal, said compositor being intended to compose said audio element using a composition syntax using a plurality of composing algorithms, and to provide, on one output terminal, a representation in a common audio representation space said audio element ;
- means for determining statistic of use of at least one of said composition algorithms,
- and means for generating a digital statistic signal representative of said statistics for each of said composed audio element.

69. Composition system for an audio-visual object comprising a composition system according to claim 66 combined with a composition system according to claim 67.

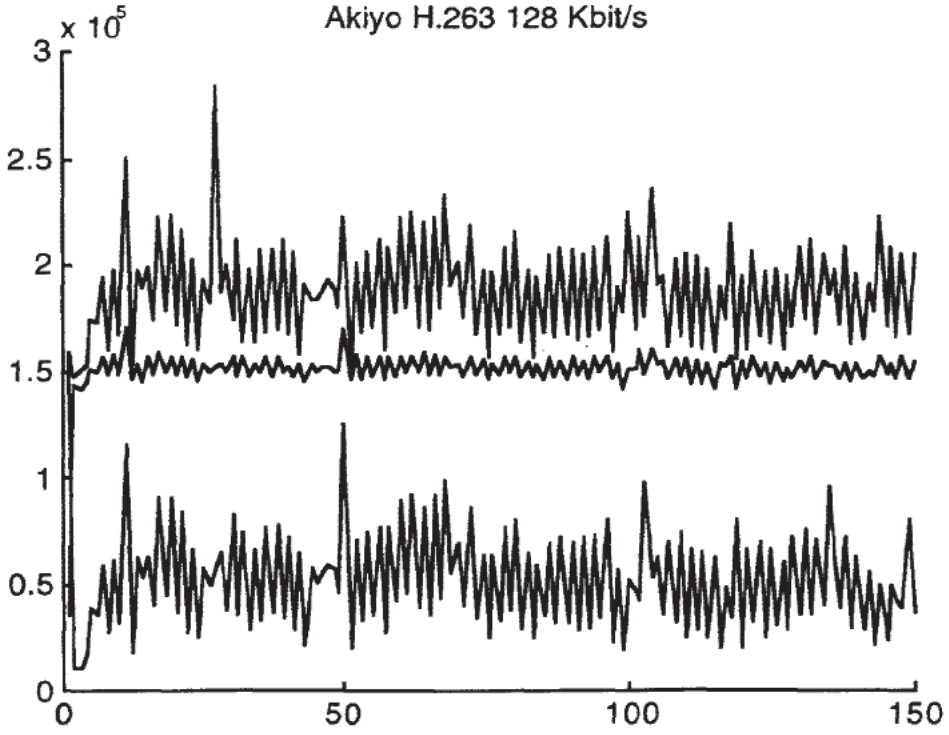


Figure 1

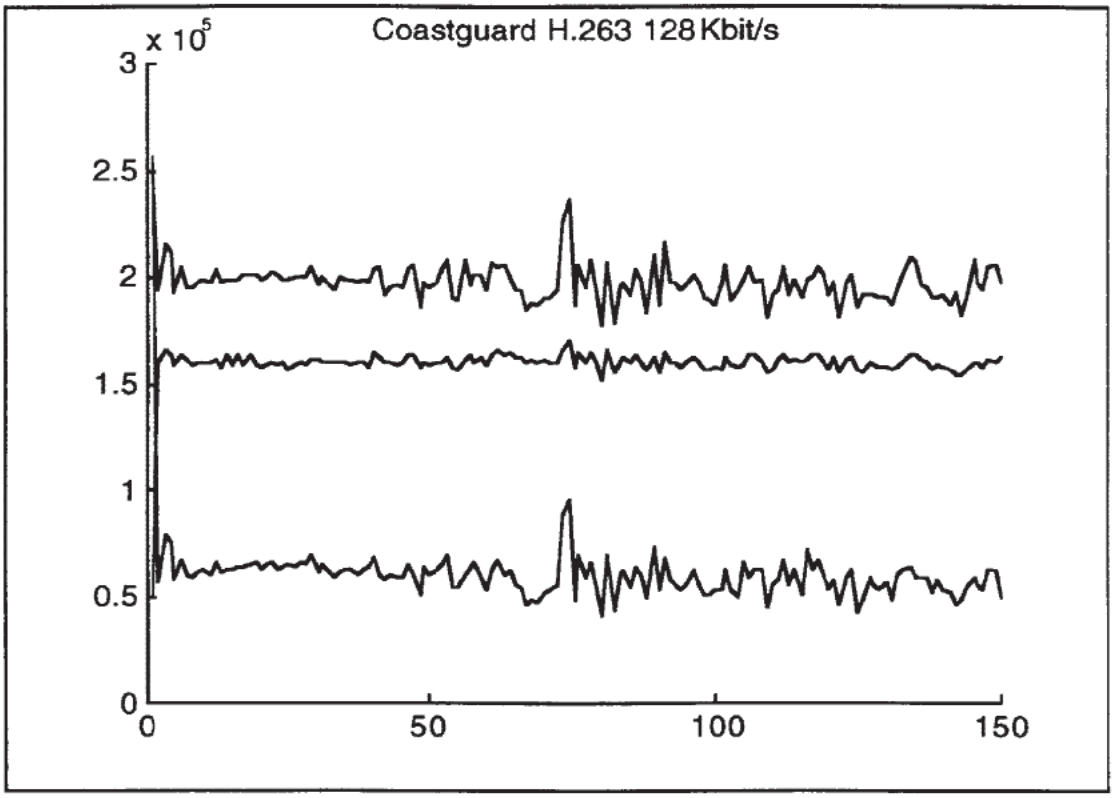


Figure 2

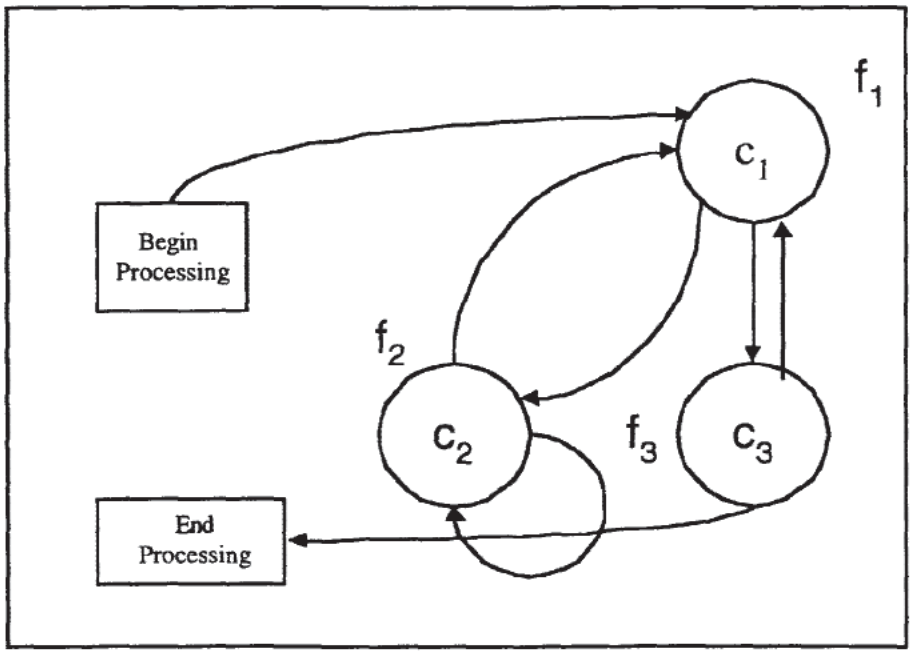


Figure 3

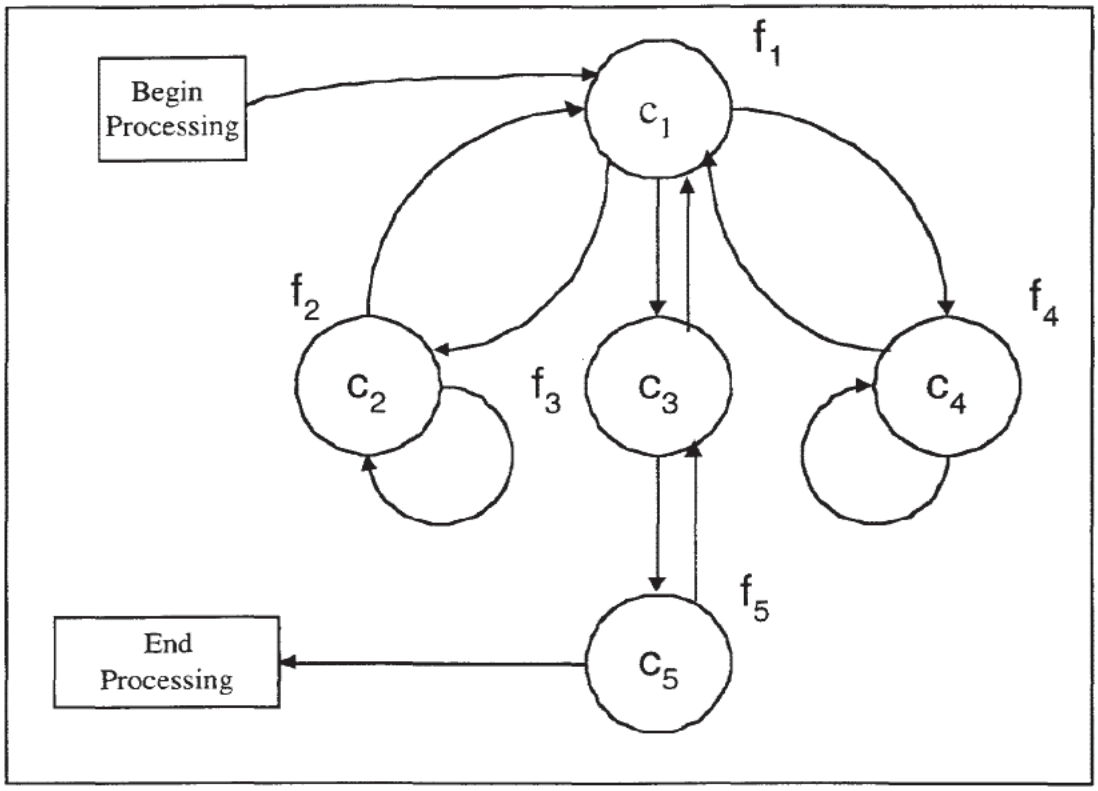


Figure 4

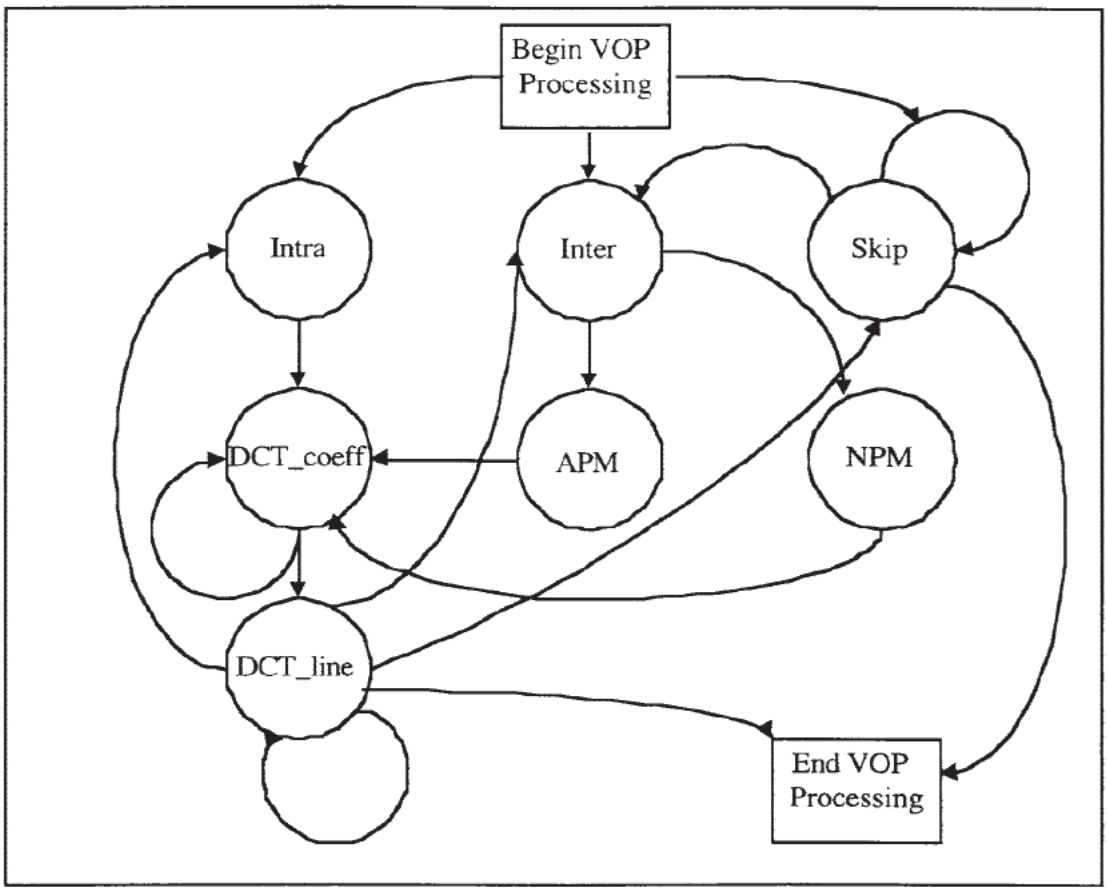

Begin VOP Processing
Intra
Inter
Skip
DCT_coeff
APM
NPM
DCT_line
End VOP Processing

Figure 5

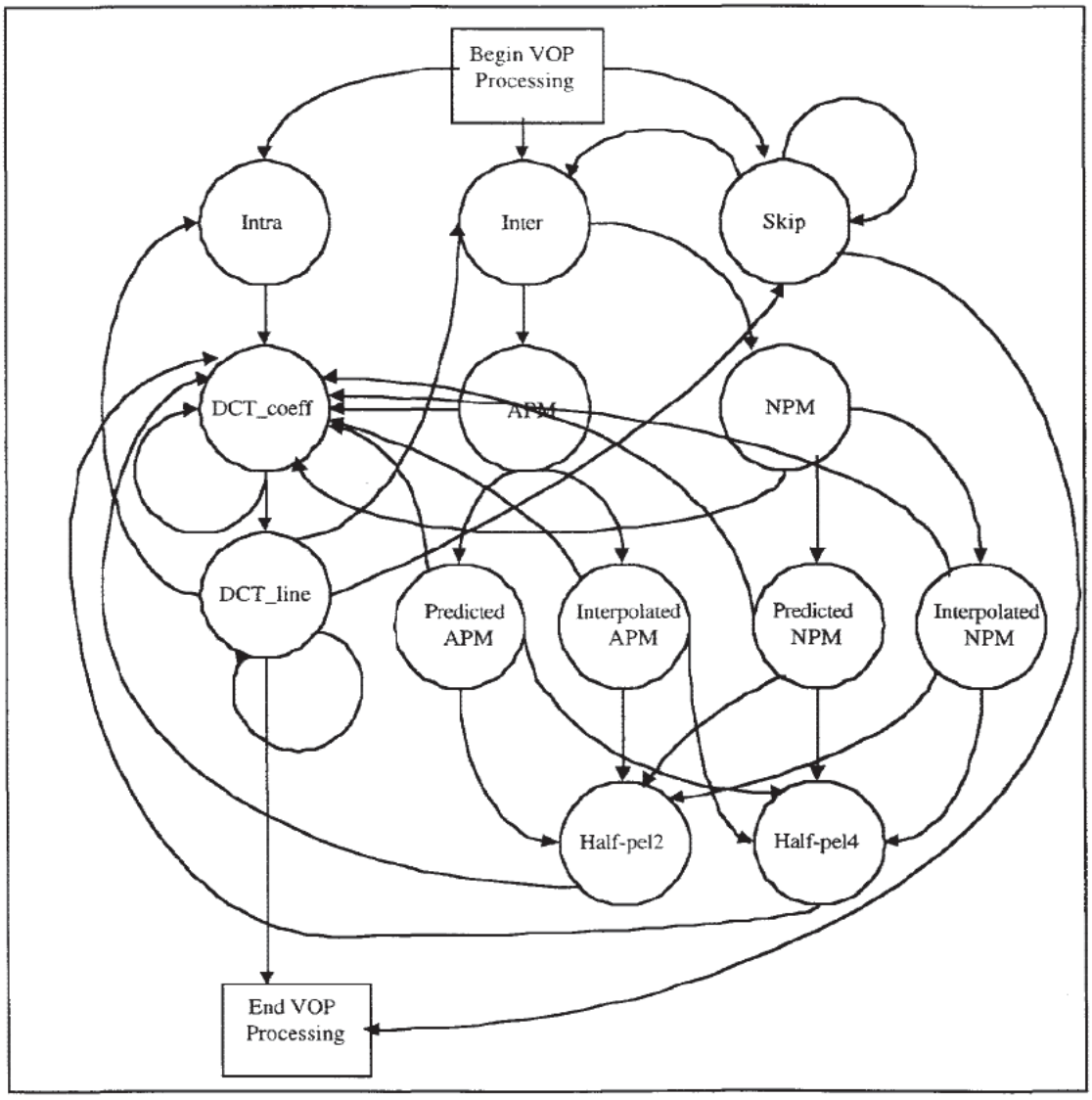

Begin VOP Processing
Intra
Inter
Skip
DCT_coeff
APM
NPM
DCT_line
Predicted APM
Interpolated APM
Predicted NPM
Interpolated NPM
Half-pel2
Half-pel4
End VOP Processing

Figure 6

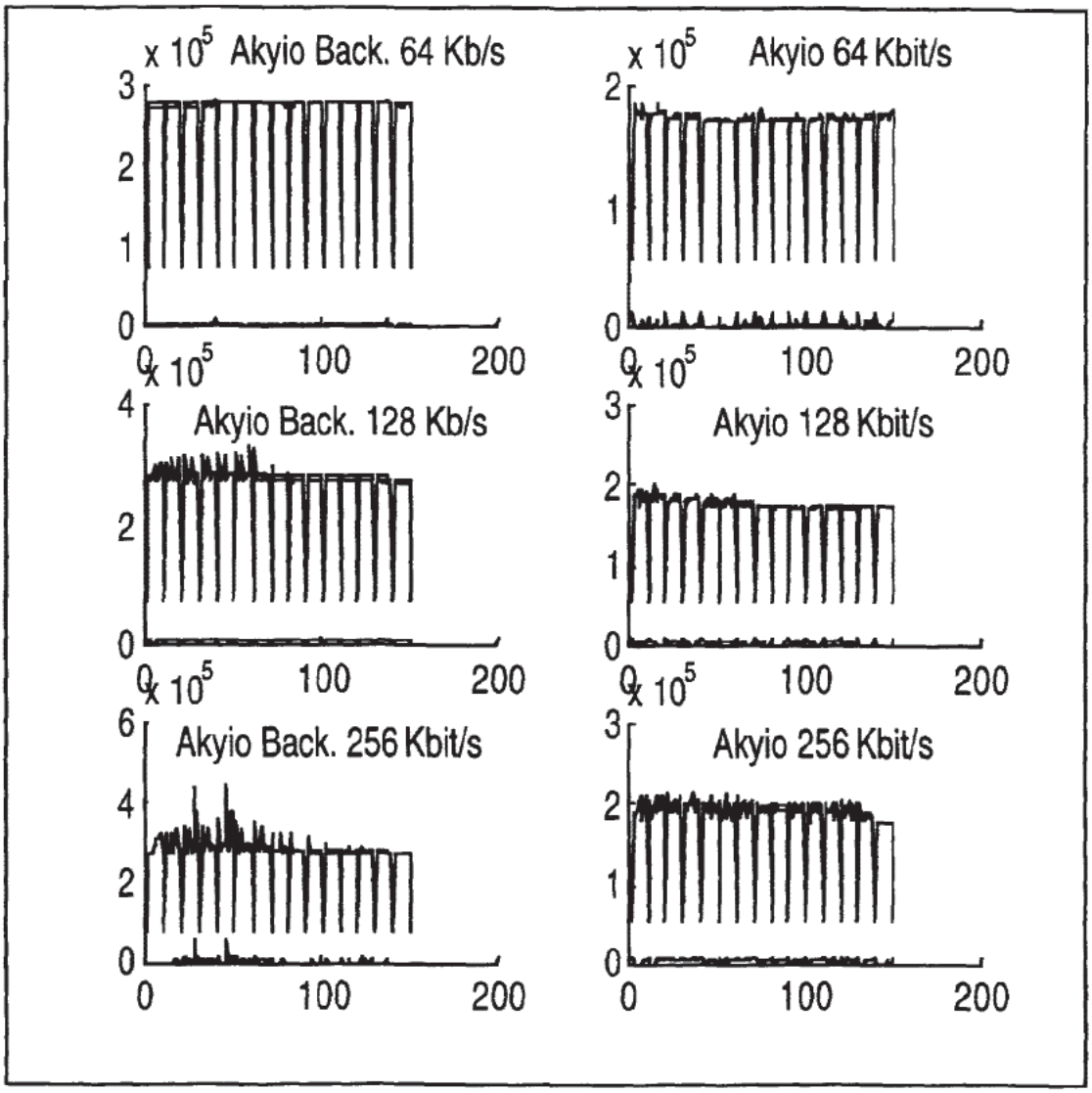


Figure 7

Ex. 1018-160

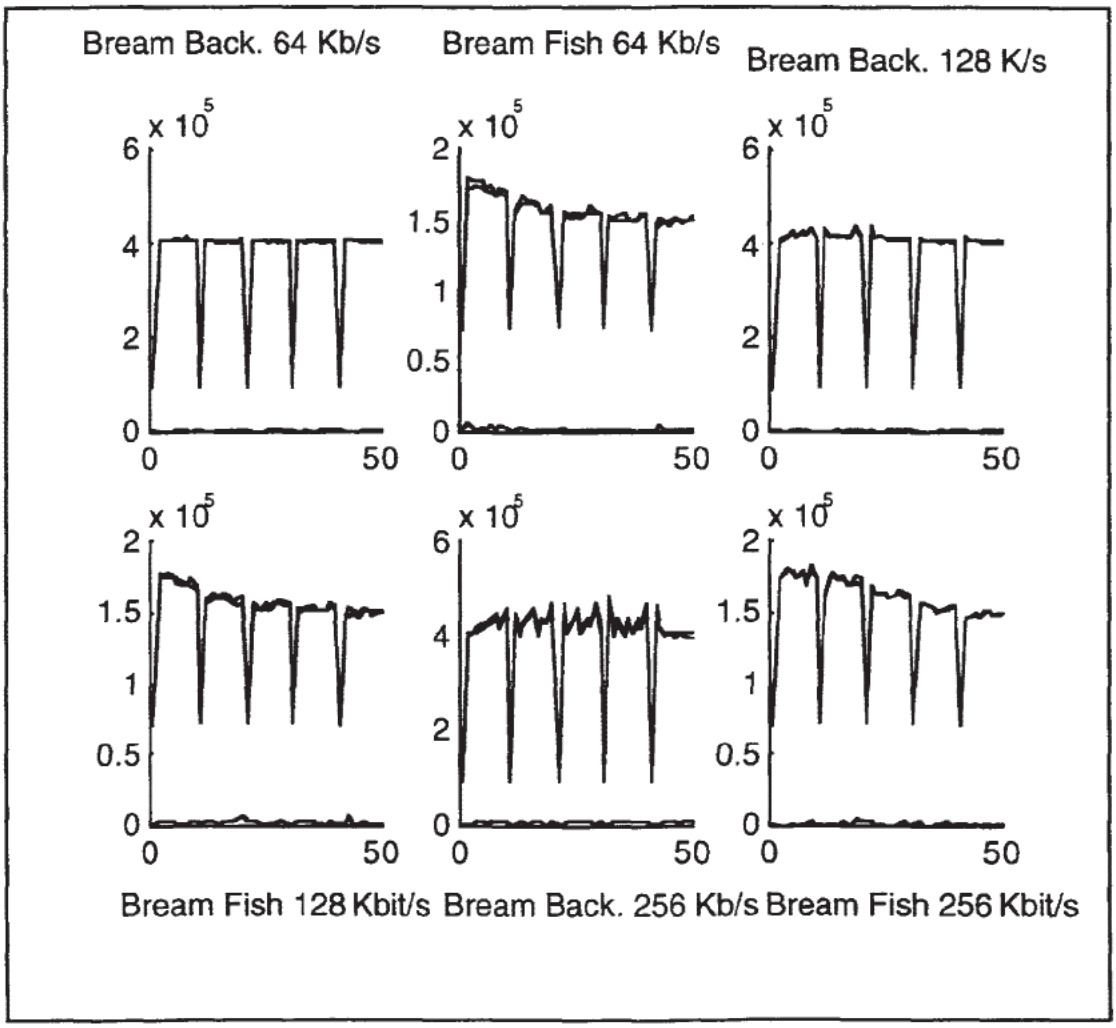

Bream Back. 64 Kb/s
Bream Fish 64 Kb/s
Bream Back. 128 K/s
x 10^5
6
4
2
0
0
50
2
1.5
1
0.5
Bream Fish 128 Kbit/s
Bream Back. 256 Kb/s
Bream Fish 256 Kbit/s

Figure 8

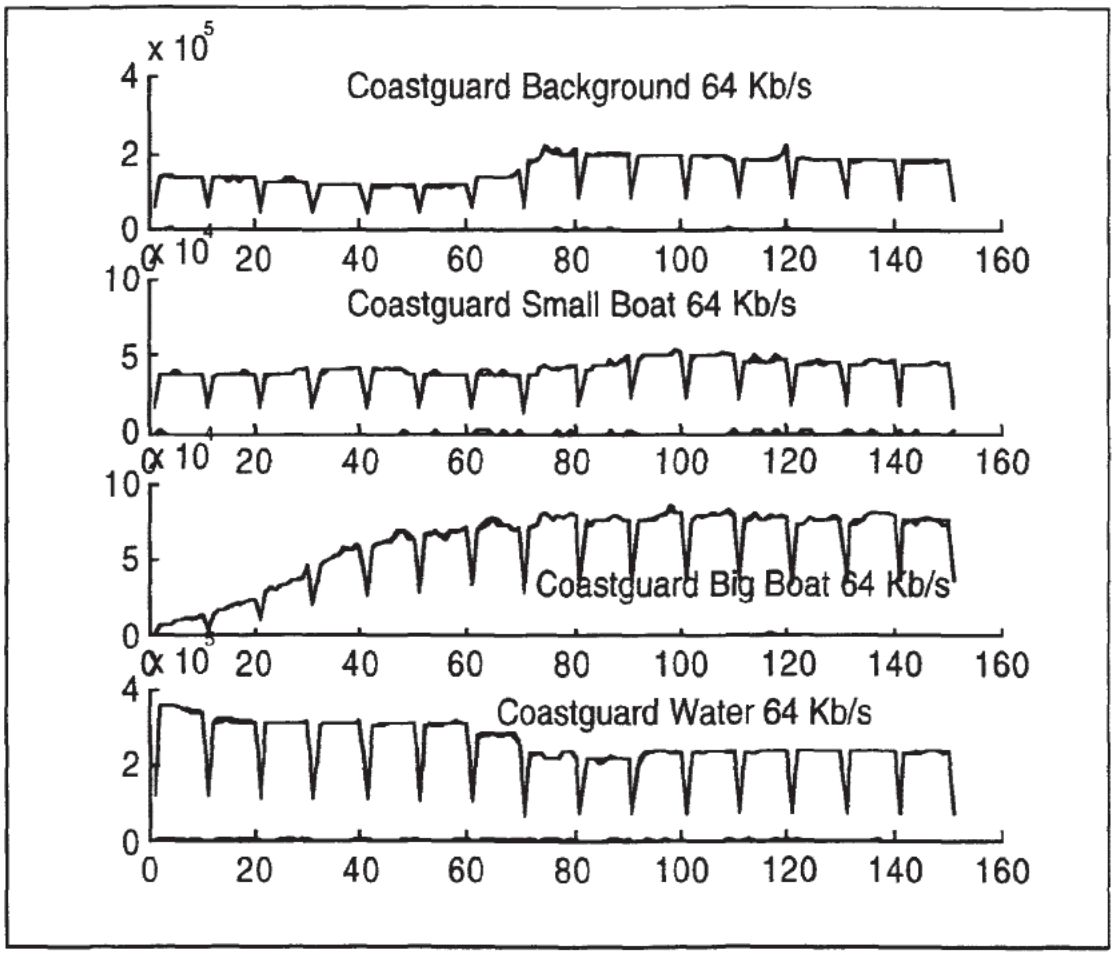

x 10^5
Coastguard Background 64 Kb/s
Coastguard Small Boat 64 Kb/s
Coastguard Big Boat 64 Kb/s
Coastguard Water 64 Kb/s
0
20
40
60
80
100
120
140
160

Figure 9

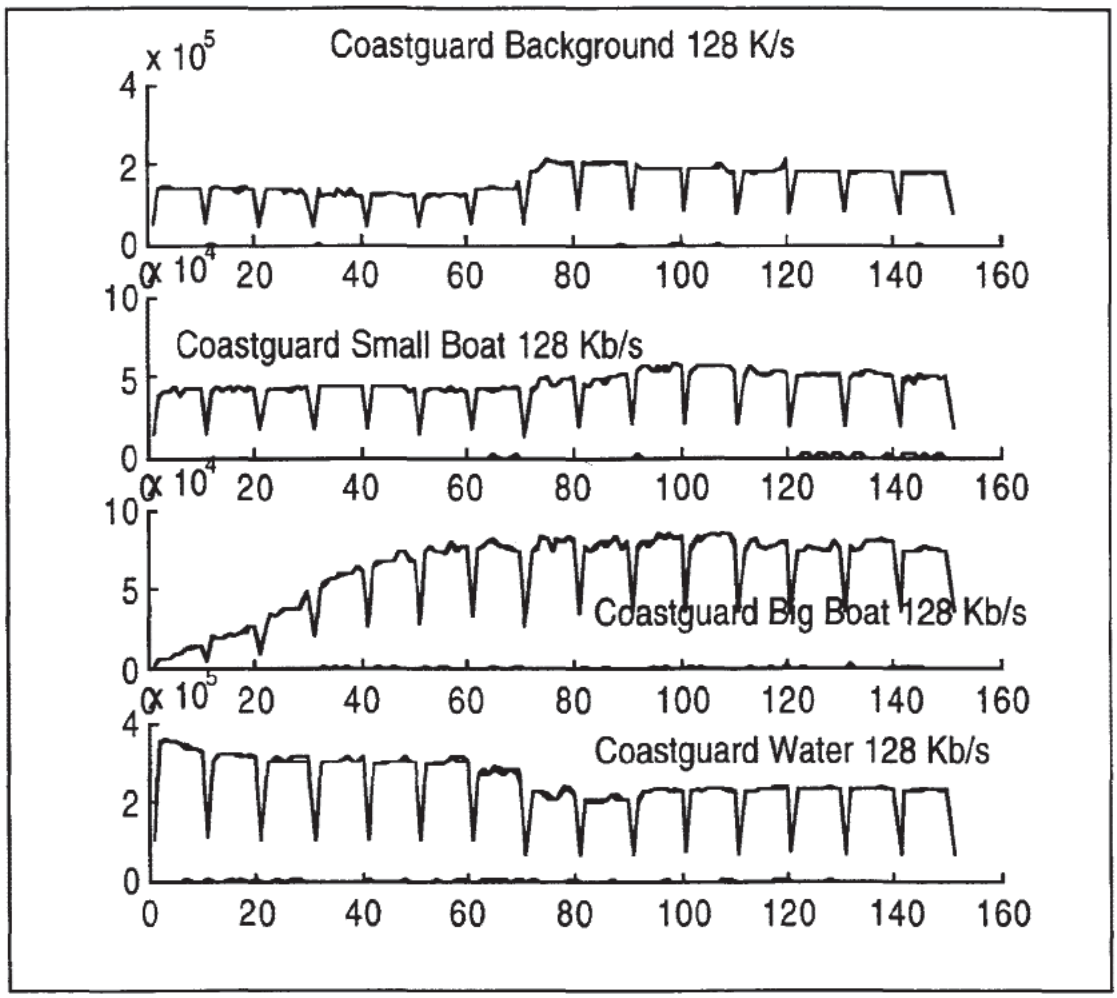

Coastguard Background 128 K/s
x 10^5
Coastguard Small Boat 128 Kb/s
x 10^4
Coastguard Big Boat 128 Kb/s
x 10^4
Coastguard Water 128 Kb/s
x 10^5
0 20 40 60 80 100 120 140 160

Figure 10

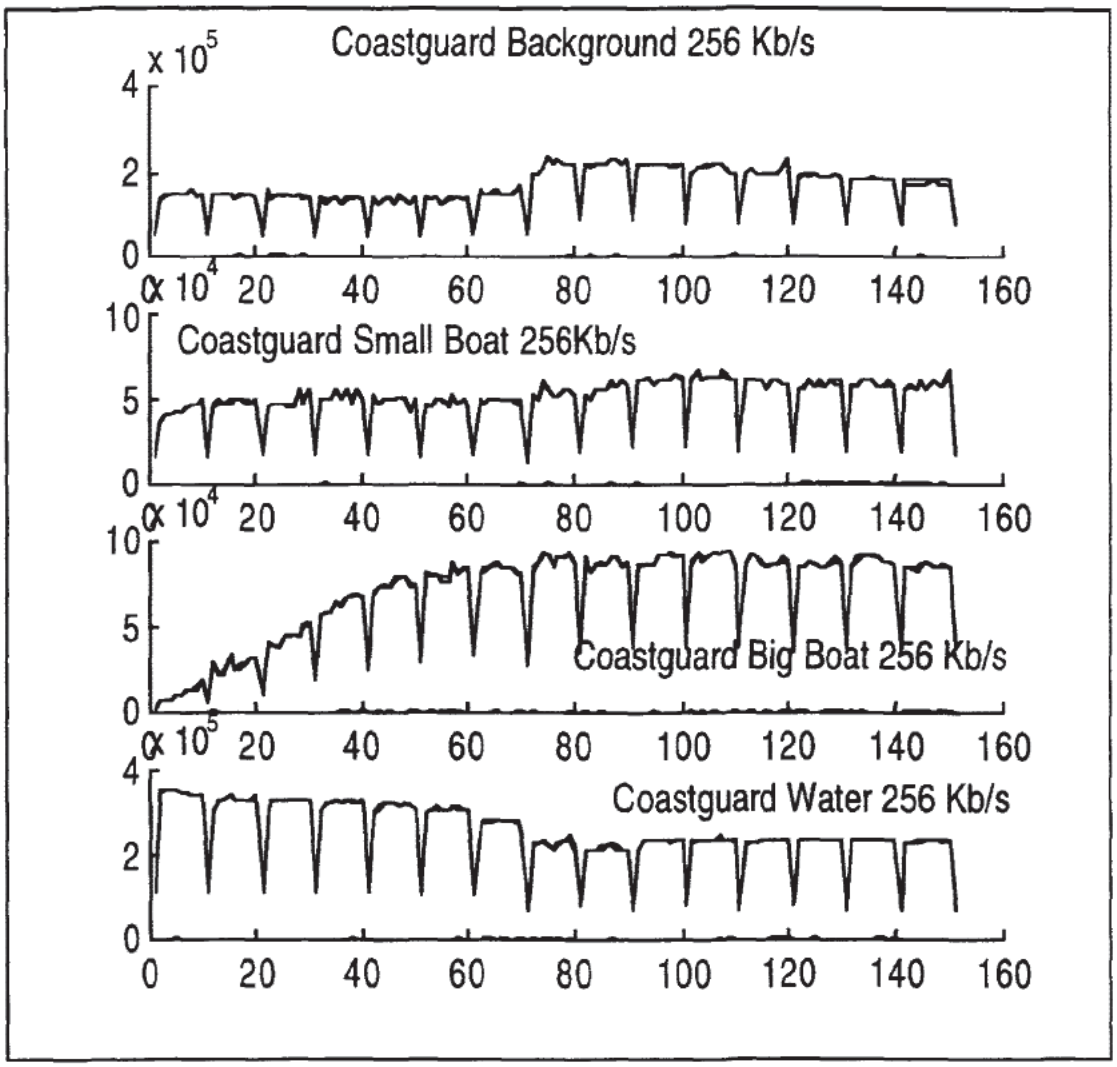

Coastguard Background 256 Kb/s
x 10^5
4
2
0
0 20 40 60 80 100 120 140 160
x 10^4
10
5
0
Coastguard Small Boat 256Kb/s
0 20 40 60 80 100 120 140 160
x 10^4
10
5
0
Coastguard Big Boat 256 Kb/s
0 20 40 60 80 100 120 140 160
x 10^5
4
2
0
Coastguard Water 256 Kb/s
0 20 40 60 80 100 120 140 160

Figure 11

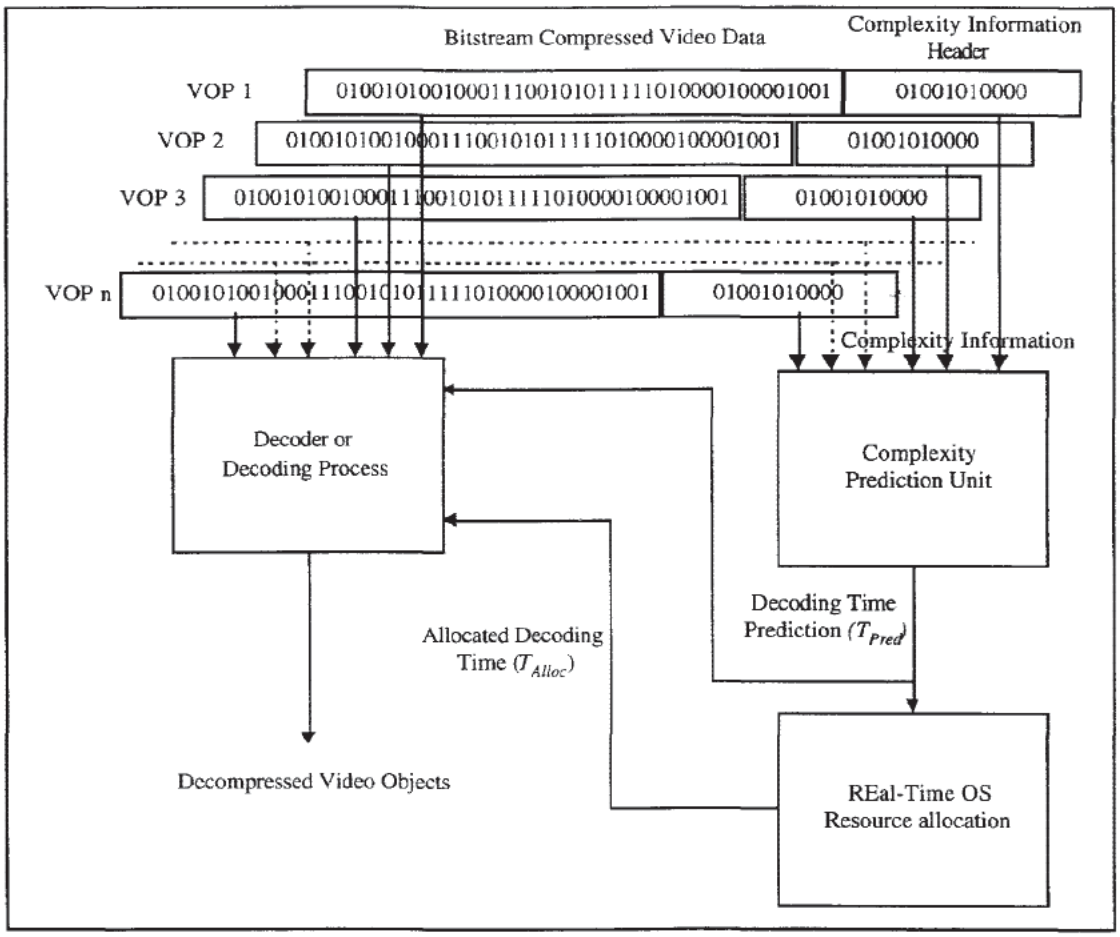

Bitstream Compressed Video Data
Complexity Information Header
VOP 1
01001010010001110010101111101000010000 1001
01001010000
VOP 2
01001010010001110010101111101000010000 1001
01001010000
VOP 3
01001010010001110010101111101000010000 1001
01001010000
VOP n
01001010010001110010101111101000010000 1001
01001010000
Complexity Information
Decoder or Decoding Process
Complexity Prediction Unit
Decoding Time Prediction (T_{Pred})
Allocated Decoding Time (T_{Alloc})
Decompressed Video Objects
REal-Time OS Resource allocation

Figure 12

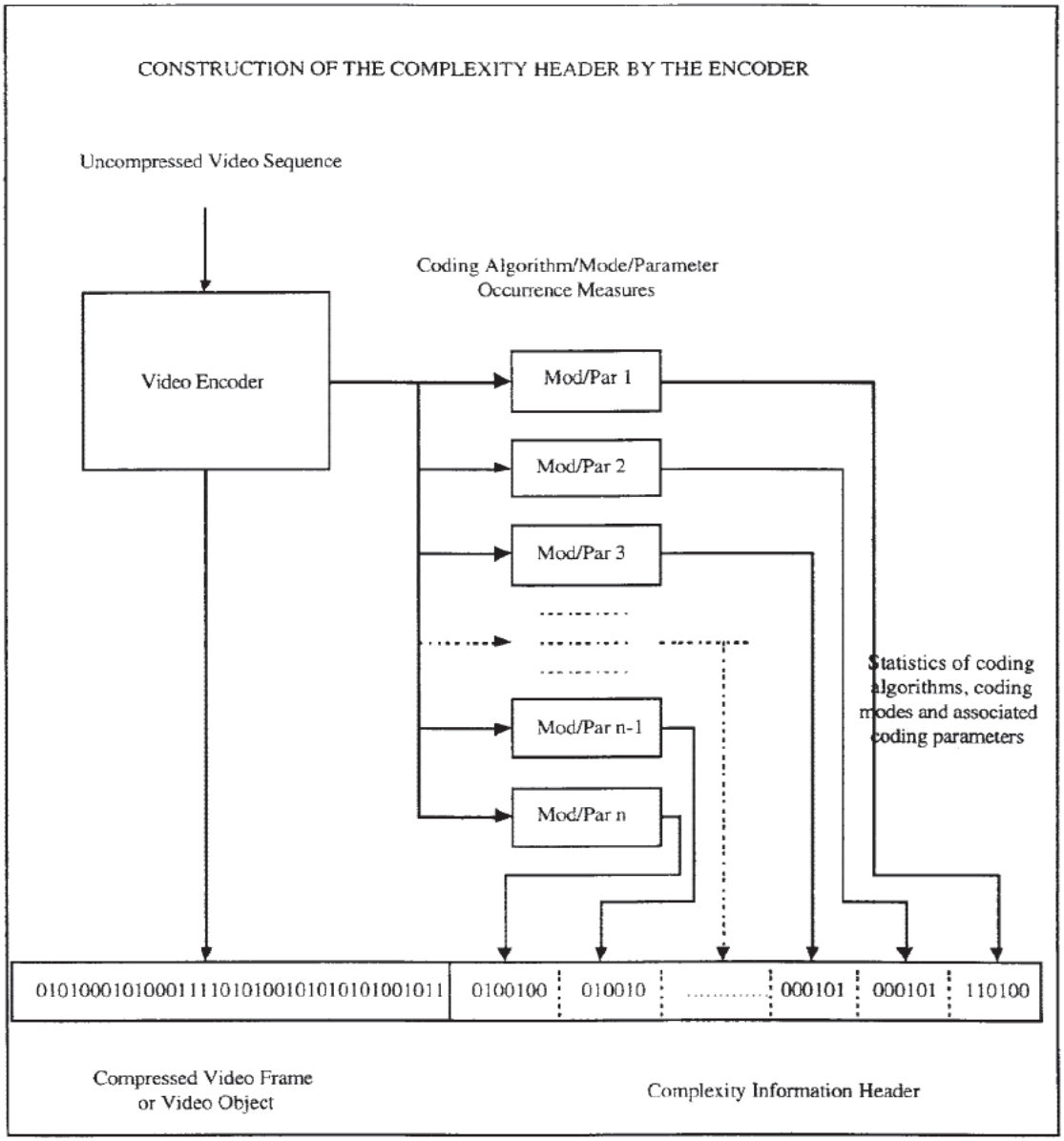

CONSTRUCTION OF THE COMPLEXITY HEADER BY THE ENCODER
Uncompressed Video Sequence
Coding Algorithm/Mode/Parameter Occurrence Measures
Video Encoder
Mod/Par 1
Mod/Par 2
Mod/Par 3
Mod/Par n-1
Mod/Par n
Statistics of coding algorithms, coding modes and associated coding parameters
0101000101000111101010010101010100101 1
0100100
010010
000101
000101
110100
Compressed Video Frame or Video Object
Complexity Information Header

Figure 13

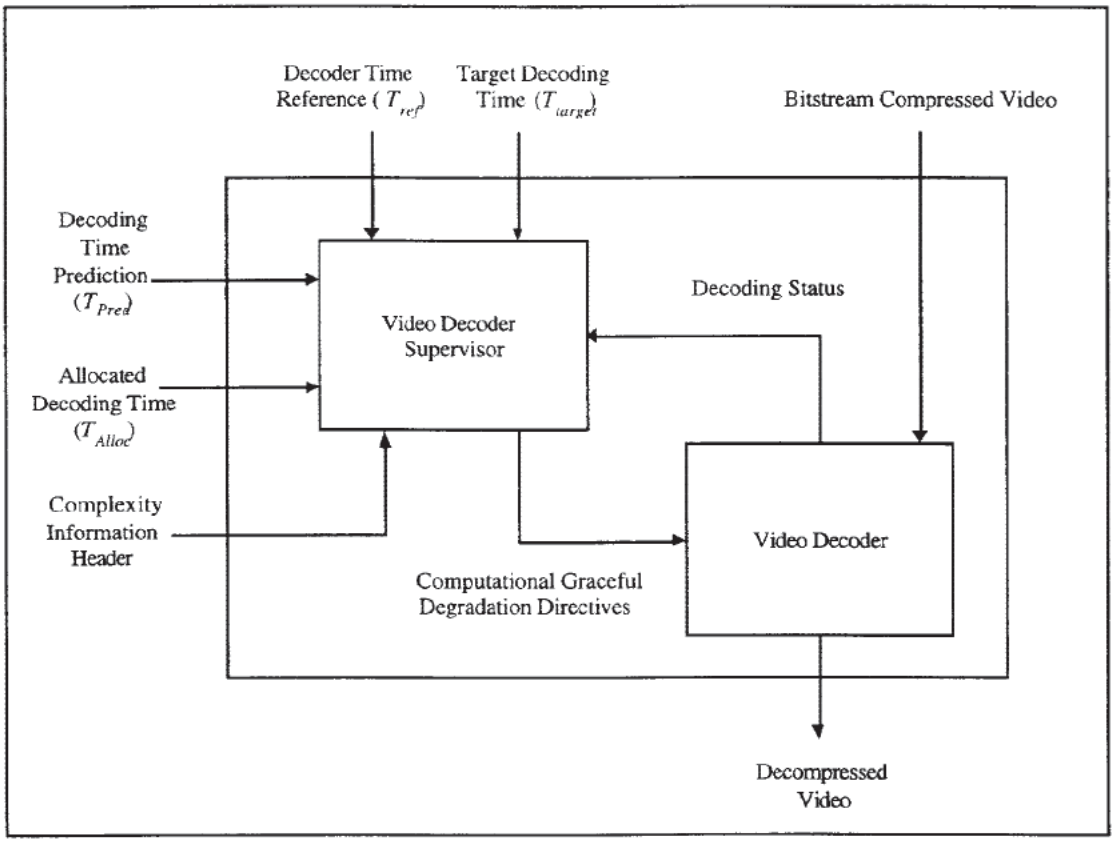

Decoder Time Reference (T_{ref})
Target Decoding Time (T_{target})
Bitstream Compressed Video
Decoding Time Prediction (T_{Pred})
Allocated Decoding Time (T_{Alloc})
Complexity Information Header
Video Decoder Supervisor
Decoding Status
Video Decoder
Computational Graceful Degradation Directives
Decompressed Video

Figure 14

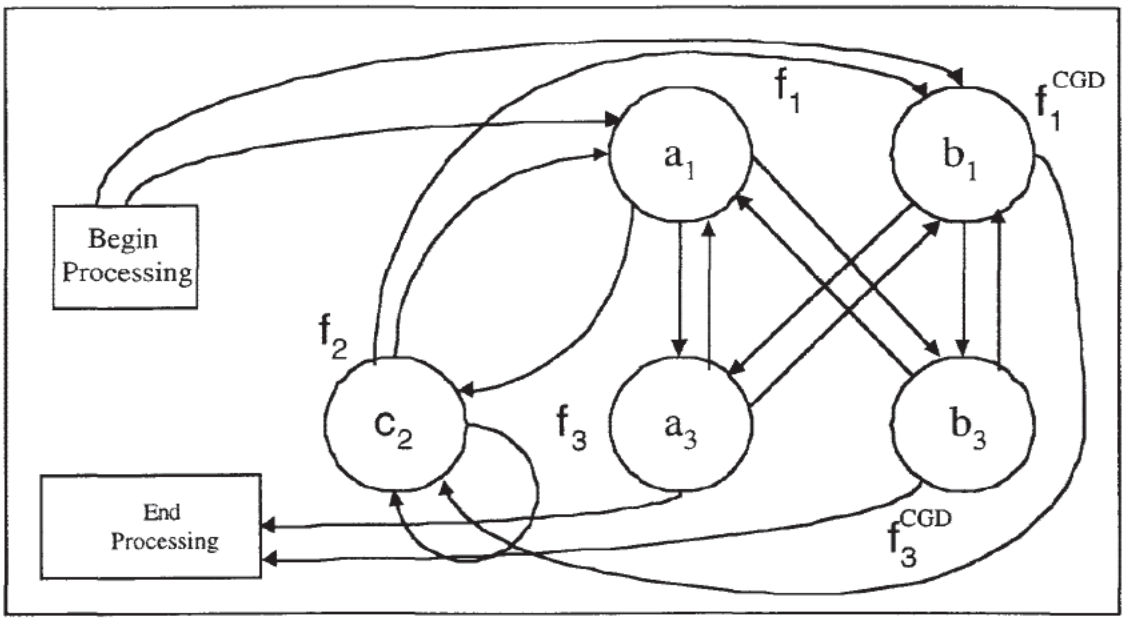

Begin Processing
End Processing
a_1
b_1
c_2
a_3
b_3
f_1
f_1^{CGD}
f_2
f_3
f_3^{CGD}

Figure 15

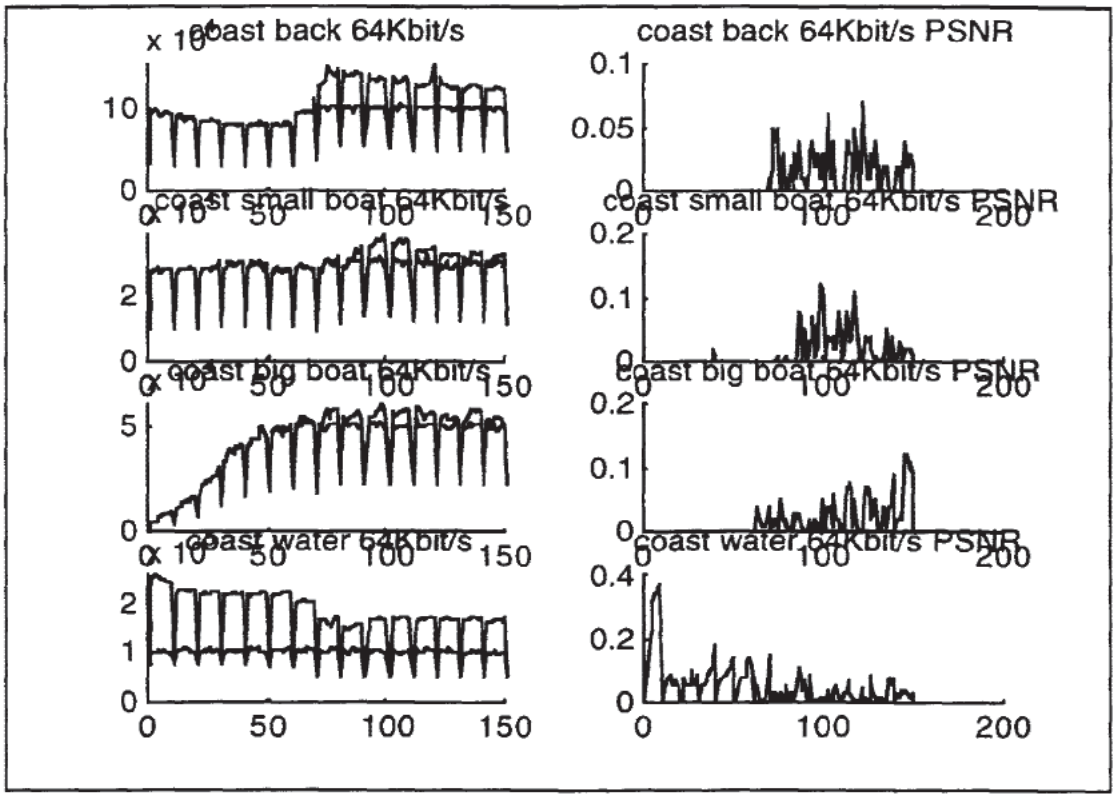


Figure 16

Ex. 1018-169

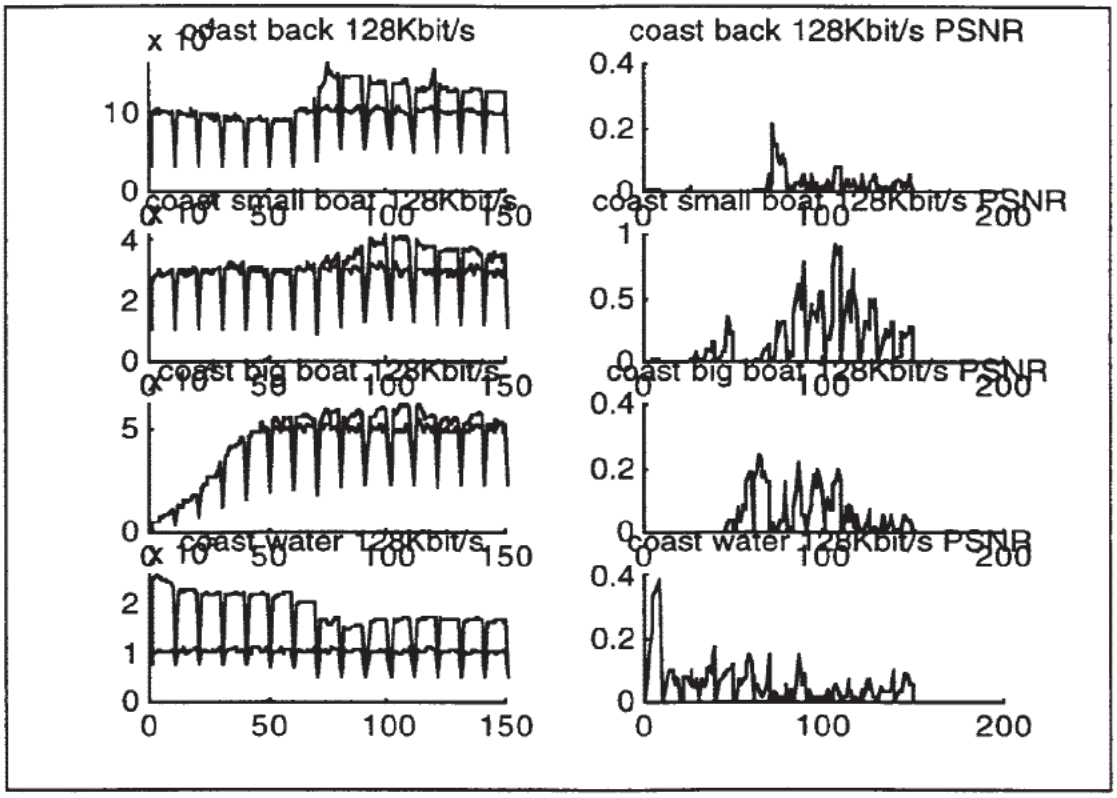

coast back 128Kbit/s
coast back 128Kbit/s PSNR
coast small boat 128Kbit/s
coast small boat 128Kbit/s PSNR
coast big boat 128Kbit/s
coast big boat 128Kbit/s PSNR
coast water 128Kbit/s
coast water 128Kbit/s PSNR

Figure 17

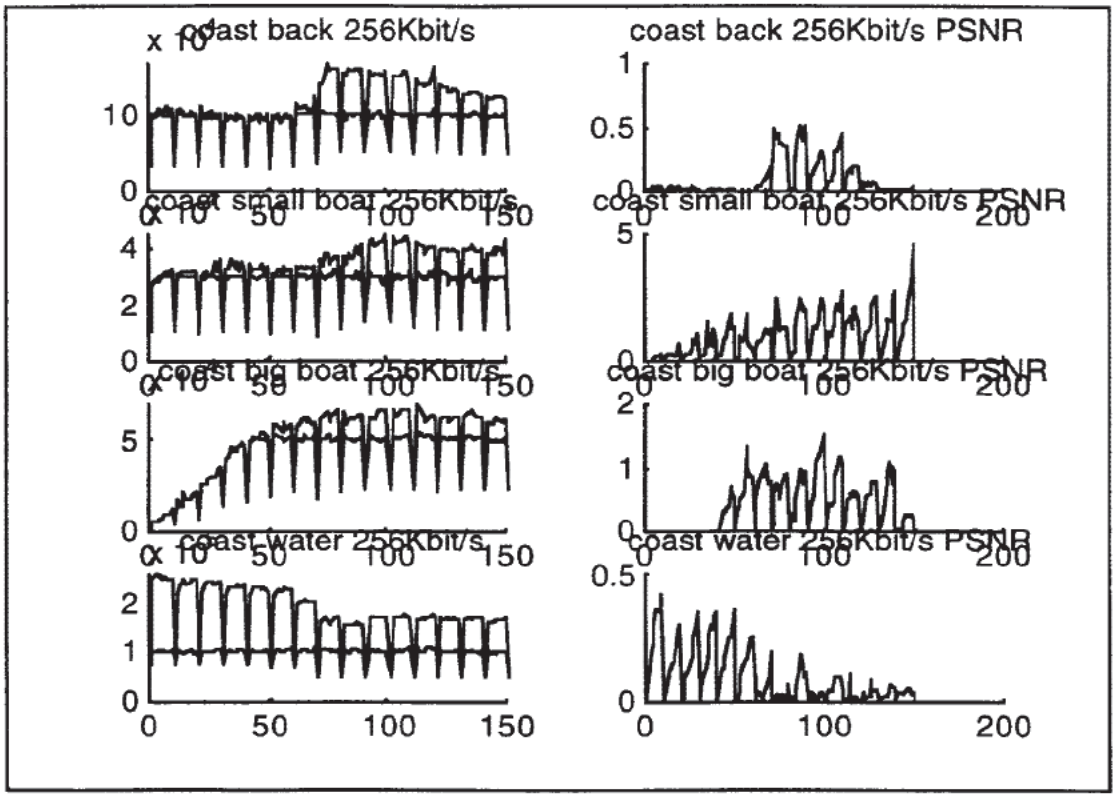

coast back 256Kbit/s
coast back 256Kbit/s PSNR
coast small boat 256Kbit/s
coast small boat 256Kbit/s PSNR
coast big boat 256Kbit/s
coast big boat 256Kbit/s PSNR
coast water 256Kbit/s
coast water 256Kbit/s PSNR

Figure 18

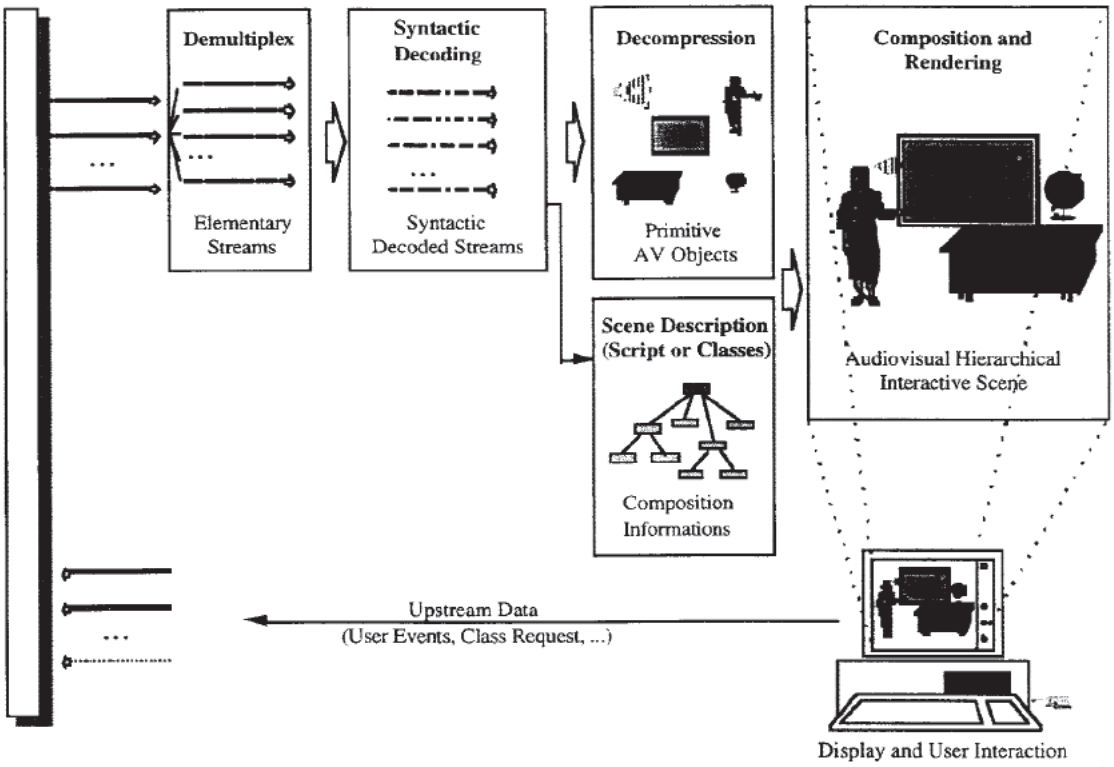

Demultiplex
Elementary Streams
Syntactic Decoding
Syntactic Decoded Streams
Decompression
Primitive AV Objects
Scene Description (Script or Classes)
Composition Informations
Composition and Rendering
Audiovisual Hierarchical Interactive Scene
Upstream Data
(User Events, Class Request, ...)
Display and User Interaction

Figure 19

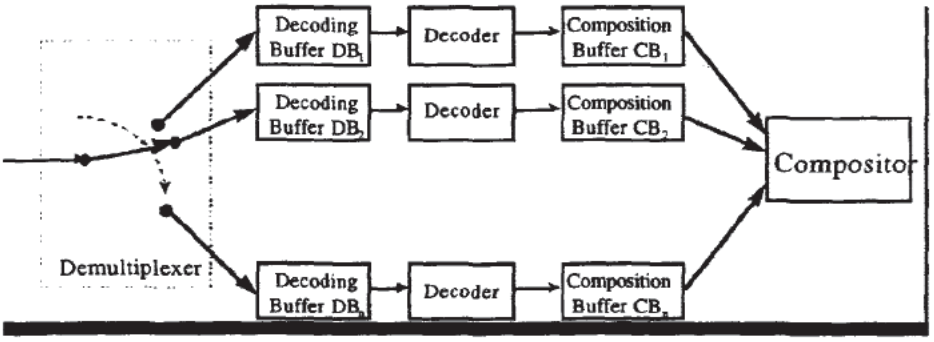

Decoding Buffer DB1
Decoder
Composition Buffer CB1
Decoding Buffer DB2
Decoder
Composition Buffer CB2
Compositor
Demultiplexer
Decoding Buffer DBn
Decoder
Composition Buffer CBn

Figure 20

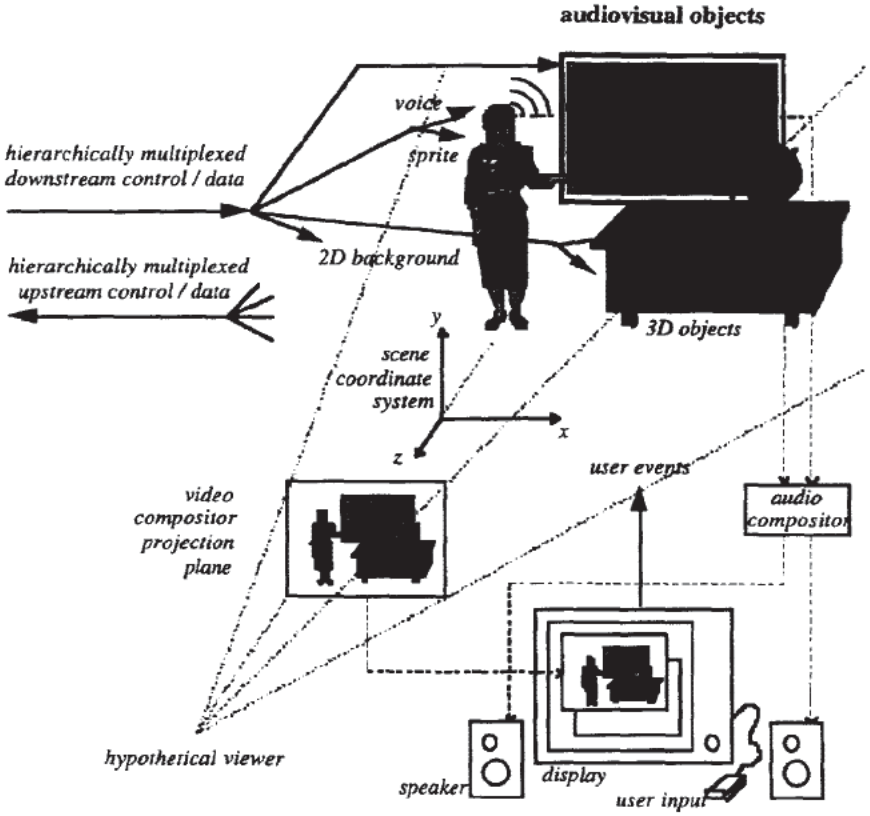

audiovisual objects
voice
sprite
hierarchically multiplexed
downstream control / data
2D background
hierarchically multiplexed
upstream control / data
3D objects
y
scene
coordinate
system
x
z
user events
video
compositor
projection
plane
audio
compositor
hypothetical viewer
speaker
display
user input

Figure 21

European Patent Office

EUROPEAN SEARCH REPORT

Application Number
EP 98 10 9423

DOCUMENTS CONSIDERED TO BE RELEVANT

Category	Citation of document with indication, where appropriate, of relevant passages	Relevant to claim	CLASSIFICATION OF THE APPLICATION (Int.Cl.6)
E	EP 0 866 426 A (ALSTHOM CGE ALCATEL) 23 September 1998 * abstract; claims 1,5,6 * ---	1-68	G06T9/00
A	EP 0 512 174 A (SEMAPHORE INC) 11 November 1992 * page 2, line 40 - line 43 * * page 3, line 32 - line 36 * ---	1-68	
A	MATTAVELLI M ET AL: "Computational graceful degradation for video sequence decoding" PROCEEDINGS. INTERNATIONAL CONFERENCE ON IMAGE PROCESSING (CAT. NO.97CB36144), PROCEEDINGS OF INTERNATIONAL CONFERENCE ON IMAGE PROCESSING, SANTA BARBARA, CA, USA, 26-29 OCT. 1997, pages 330-333 vol.1, XP002080307 ISBN 0-8186-8183-7, 1997, Los Alamitos, CA, USA, IEEE Comput. Soc, USA -----		

TECHNICAL FIELDS SEARCHED (Int.Cl.6)

G06T

The present search report has been drawn up for all claims

Place of search	Date of completion of the search	Examiner
THE HAGUE	12 October 1998	Pierfederici, A

CATEGORY OF CITED DOCUMENTS

X : particularly relevant if taken alone
Y : particularly relevant if combined with another document of the same category
A : technological background
O : non-written disclosure
P : intermediate document

T : theory or principle underlying the invention
E : earlier patent document, but published on, or after the filing date
D : document cited in the application
L : document cited for other reasons

& : member of the same patent family, corresponding document

EPO FORM 1503 03.82 (P04C01)

(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) EP 1 777 969 A1

(12) **EUROPEAN PATENT APPLICATION**

(43) Date of publication:
25.04.2007 Bulletin 2007/17

(51) Int Cl.:
H04N 7/46 *(2006.01)*

(21) Application number: **05256302.0**

(22) Date of filing: **10.10.2005**

(84) Designated Contracting States:
AT BE BG CH CY CZ DE DK EE ES FI FR GB GR HU IE IS IT LI LT LU LV MC NL PL PT RO SE SI SK TR
Designated Extension States:
AL BA HR MK YU

(71) Applicant: **BRITISH TELECOMMUNICATIONS public limited company**
London EC1A 7AJ (GB)

(72) Inventor: **The designation of the inventor has not yet been filed**

(74) Representative: **Lloyd, Barry George William et al**
BT Group Legal
Intellectual Property Department
PP C5A
BT Centre
81 Newgate Street
London EC1A 7AJ (GB)

(54) **Adaptive video transmission with variable frame rate**

(57) Video signals are sent via a network, such as an internet protocol network, the bandwidth of which varies with time. The amount of data to be sent is controlled by varying the frame rate. This is achieved by (a) measuring the transmission capacity to the receiver; (b) determining the rate of change of transmission capacity; and (c) adjusting the transmitted video frame rate as a function of the said rate of change.

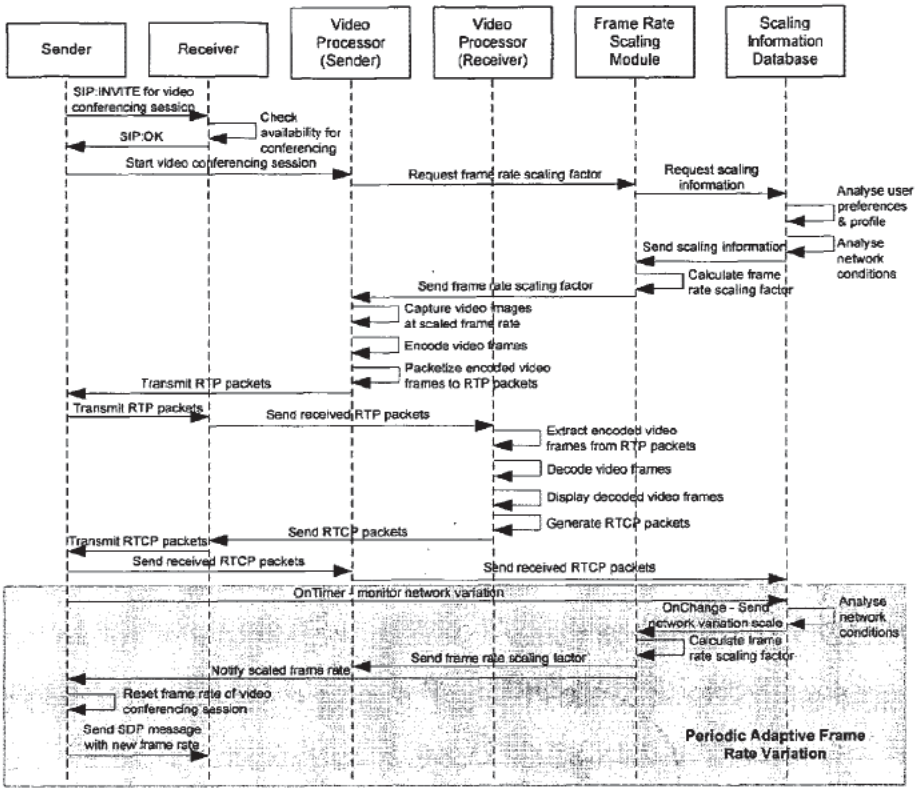


Figure 3 Sequence Diagram of Adaptive Video Frame Rate Application

EP 1 777 969 A1

Printed by Jouve, 75001 PARIS (FR)

Description

[0001] The present invention is concerned with the transmission of video signals via a network.

[0002] In the future, the next generation communication environment is envisaged as allowing users to roam freely and to communicate with each other at any time and any place [1]. Client devices with different capabilities, which range from low power mobile phones to high performance workstations, may be used for communications. Users will be interconnected by networks with various capacities, with packet loss rates that range from almost lossless transmission to quite error prone environments. The rapid development of the client device industry and the proliferation of network access technologies is expected to result in the extensive growth of bandwidth-intensive real-time multimedia internetworking applications (e.g. voice over IP, video streaming, video conferencing and so on).

[0003] Real-time multimedia transmission over a variable bandwidth network environment requires a strict guarantee of network bandwidth and of several other quality of service (QoS) parameters, such as end-to-end delay, delay variation and packet loss. Unfortunately, most of the current QoS mechanisms in existing networks (such as wireless and mobile networks) do not provide such guarantees. In dynamic and unpredictable network environments, due to diverse forms of mobility and interference at the air interface, it gives rise to varying bandwidth and dynamically changing error rates [2]. This further intensifies the challenge for consistent provision of QoS. Traditional QoS mechanisms (e.g. Integrated Services and Differentiated Services), which are typically designed for wired networks, assume a fixed network topology and a fixed amount of available resources. These are inadequate to satisfy users' QoS requirements or to utilize the system resources efficiently, especially in bandwidth-limited and error prone network environments.

[0004] In order to solve this, adaptive multimedia applications are proposed. By implementing adaptive applications, media quality can, in the case of bandwidth variation of network, be adapted appropriately. The bandwidth of the media stream can be scaled down to provide a degraded quality at the destination. In this invention, we are particularly interested in the implementation of video applications. Adaptive video applications use protocols such as Real-Time Streaming Protocol (RTSP) and Real-Time Protocol (RTP), to adapt to changing network conditions using a number of techniques. These techniques include hierarchical encoding, efficient compression, bandwidth smoothing, rate shaping, error control, and adaptive synchronization. Feedback from lower networking layers can be used by the application layer and this helps to match the transmission to the available bandwidth as well as to increase the robustness of the video transmission. However, the perceived quality of video applications is being neglected. Variations of frame rates are proposed in some of these adaptive mechanisms [3] [4]. However, abrupt changes of frame rates can annoy users and hence have a severe impact on the perceived quality of such video applications.

[0005] QoS provisioning is an important issue in real-time internetworking video applications. The dramatic fluctuation of network conditions further intensifies the problems of QoS provisioning. Under variable bandwidth network environments (such as users are moving from one network to another with different bandwidth capacity or network is being congested by heavy traffic), characteristics of video applications need to be adjusted in order to fit the requirements of the current operating network and also maintain the perceived QoS.

[0006] An IBM document on QoS policies [5] clearly states that "the priority for smooth video is higher than the priority for frame quality". It points out that the importance of smooth video is even more significant when users are watching high motion videos (such as sports programmes). Traditional adaptive frame rate variation mechanisms in video applications, which adjust the video requirements according to the rate supportable by the network, attempt to keep the quality of video constant by varying the frame rate. In these mechanisms, the perceived quality of video application is often degraded as frame rates are reduced to a much lower rate or being increased to a much higher rate in order to fit network conditions. The abrupt changes of frame rates (from higher rate to lower rate and vice versa), which cause jerkiness and unsteady video displays, annoy users who are expecting a smooth and continuous flow of video frames in real-time video applications. The users are able to notice the sudden change of frame rate in video transmission. For example, a video application may be sending video at 25 frames per second (fps) during normal network condition and frame rate is reduced to 8fps when network congestion is detected; the frame rate is then increased to 15fps again when network conditions improve. This drastic fluctuation of frame rates will affect the smoothness of the video display and hence have a severe impact on perceived quality.

[0007] References:

1. T. Takebayashi et al, Approaches to User-Centric Computing System for Ubiquitous Solutions, Fujitsu Sci. Tech. J., 39, 2, December 2003, p.261-269.

2. M. Naghshineh et al, End-to-End QoS Provisioning in Multimedia Wireless/Mobile Networks Using an Adaptive Framework, IEEE Communication Magazine November 1997.

3. J. Kim et al, TCP-Friendly Internet Video Streaming Employing Variable Frame-Rate Encoding and Interpolation, IEEE Transaction on Circuits and Systems for Video Technology, Vol.10, NO.7, October 2000.

Ex. 1018-177

[4] W. Guetz et al., U.S. Patent 6,091,777.

[5] IBM. Functions of mobile multimedia QoS control. http://www.trl.ibm.com/projects/mmqos/system_e.htm.

[0008] According to the present invention there is provided a method of transmitting video over a network to a receiver, comprising repeatedly

(a) measuring the transmission capacity to the receiver;
(b) determining the rate of change of transmission capacity;
(c) adjusting the transmitted video frame rate as a function of the said rate of change.

[0009] Other aspects of the invention are set out in the claims.

[0010] Some embodiments of the invention will now be described, with reference to the accompanying drawings.

[0011] The basic system operational view of our implementation is depicted in Figure 1. The figure shows one-way communication (sender-to-receiver), however, it can be implemented as a two-way communication application (client-to-client). Session Initiation Protocol (SIP) can be used to start a communication session between two or more clients. Real-Time Protocol (RTP) and

[0012] Real-Time Control Protocol (RTCP) can be used as underlying transport layer to send data packets and also control packets. In this implementation, Session Description Protocol (SDP) signalling can be used to reset the parameters of a video conferencing session and send them to the receiving end.

[0013] The video processor, which involves capture/display, encoder/decoder and transmitter/receiver, is mainly for media processing. The adaptation mechanism appears as a standalone module in the application. A scaling information database extracts and stores network statistical information (monitored periodically using a timer) and user profiles (set by users). The frame rate scaling module is the core mechanism which calculates a frame rate scaling factor based on information obtained from information database. A scaled frame rate will then be fed to the video processor for setting new video parameters.

[0014] Figure 2 illustrates how the different parameters influence the frame rate in the case of fluctuation in the network . To calculate the frame rate scaling factor, several parameters need to be taken into consideration. These parameters may include network statistics, RTCP feedback statistics and the user profile. The main contributor to the scaling factor is the network statistics, where the changes with time (Δbandwidth/Δt) are tracked.

[0015] The sequence diagram (Figure 3) shows the activity flows of our implementation. The periodic adaptive frame rate variation mechanism (shaded portion) is applied to avoid the situation that frame rate variation is only incurred when network congestion is detected. A timer is initiated for periodic network condition monitoring at the client end. The frame rate will be scaled based on the scaling factor extracted from scaling information as discussed previously. This will improve the perceived quality by giving a gradual change of frame rate and smooth playbacks of video throughout entire communication.

[0016] The control mechanism essentially comprises the following steps:

(a) measurement, at intervals, of the network bandwidth $B(t_i)$;
(b) determination of the gradient $\{B(t_i) - B(t_{i-1})\}/(t_i - t_{i-1})$;
(c) determination of a scaling factor *M*, as a function of the gradient *m*;
(d) determination of a new frame rate, multiplying the existing rate by the scaling factor - subject however to the rate remaining within a range determined by preset maximum and minimum values..

[0017] In more detail:

(a) Bandwidth could be measured by any one of a number of conventional methods. If the communication session is peer to peer within a similar subnet, say, in an enterprise scenario, the backhaul link of the connecting equipment could expose its bandwidth availability through its SNMP agent residing in that particular equipment. This could be queried via SNMP calls. On the other hand, in an environment when there are several nodes in the path of its communication, an application level implementation of measuring its delay and thus bottleneck within that particular path could be used. This would suggest bandwidth availability.
Clearly it is important, in terms of reducing the risk of large changes in frame rate, to measure the bandwidth (and adjust the frame rate) at frequent intervals - that is to say, frequent in relation to the rate of change of network conditions. A typical rate might be every 10 seconds. It would be possible to measure the bandwidth once per frame, but this is not preferred, in that it involves additional overheads. In theory the bandwidth could be measured more often than once per frame, but naturally the frame rate adjustment cannot occur more often than once per frame.
(b) The gradient is determined by the formula given above. The use of the gradient rather than simply the absolute value of the bandwidth is significant because it introduces an element of prediction into the assessment of bandwidth availability.
(c) The determination of the scaling factor *M* as a function of the gradient *m* should be such that

Ex. 1018-178

If *m* is large & positive, multiplier is >1;
If *m* is small & positive, multiplier is > 1 (but smaller than when *m* is large);
If *m* is 0, multiplier is 1;
If *m* is small & negative, multiplier is <1;
If m is large & negative, multiplier is <1 (but smaller than when *m* is small and negative.
For example, a possible relationship for the situation where the bandwidth gradient is expected to vary over the range -50 to 50 kbit/s/s is: that M varies between 0.4 and 2.5. A simple, piecewise-linear implementation would be to specify that for m>0: M = 1 + 0.03 m; for m=0: M = 1; and for m<0: M = 1 + 0.012 m. However we prefer a nonlinear relationship in which frame rate changes are more extreme at the ends of the scale. Such a relationship would best be implemented by the use of a look-up table.
(d) A typical limit range for frame rate might be from 10 to 25 frames per second. Thus if the existing frame rate is f_i then the new frame rate will be

$$f_{i+1} = \mathrm{Max}\{\mathrm{Min}\{Mf_i, 25\}, 10\}.$$

[0018] The user preferences referred to earlier could mean constraints on the level of adaptation that the user is prepared to live with in a congested network situation, i.e reaction time, image resolution resizing, colour depth. This could influence the extent for the need to switch to a lower frame rate.

[0019] In order to achieve better perceived quality of real-time internetworking video application in variable bandwidth network environments, degradation of video quality needs to be occur gracefully. In the method we have described, an adaptive mechanism, which resides in the video application, allows smooth variation of frame rates under variable bandwidth network environments. It varies frame rates gradually and hence minimises the jerkiness and flickering of video displays. A gradual degradation or upgrade of video quality gives an illusion to users that there is only little change in video quality throughout the communication period. The variation incurred is less aggressive and significant as compared with traditional frame rate variation mechanisms.

[0020] The method improves the perceived QoS of real-time internetworking video applications by implementing smooth variation frame rate mechanism of adaptive video application. An adaptation mechanism resides in a real-time internetworking multimedia application to adapt video quality dynamically to changes in network conditions. Network conditions are absorbed and analysed from time to time and network statistical analysis is then fed back to the video processor of the application. The application is then triggered to set the frame rate of the video to a requested rate based on the analysis of network conditions. The frame rate changes dynamically when network conditions vary - but not only when network congestion is detected. The degree of variation in frame rate depends on the degree of variation in network conditions (such as network utilization) and the variations of frame rates occur throughout the communication session. Hence, it prevents sudden changes in frame rate of the video transmission and helps to minimise the jerkiness of video displays under variable bandwidth conditions. User preferences and profiles may also be taken into account when frame rate is being adjusted. This further improves user perceived QoS of the application.

Claims

1. A method oftransmitting video over a network to a receiver, comprising repeatedly

   (a) measuring the transmission capacity to the receiver;
   (b) determining the rate of change of transmission capacity;
   (c) adjusting the transmitted video frame rate as a function of the said rate of change.

2. A method according to claim 1 in which the frame rate is adjusted to equal the existing frame rate multiplied by a factor which is a monotonically increasing function of rate of change of transmission capacity.

3. A method according to claim 2 in which the factor is a nonlinear function of rate of change of transmission capacity.

4. A method according to claim 3 in which the factor is obtained by means of a look-up table.

5. A method according to any one of the preceding claims in which the rate of change of transmission capacity is evaluated to determine whether an adjustment of frame rate is necessary at regular intervals.

Ex. 1018-179

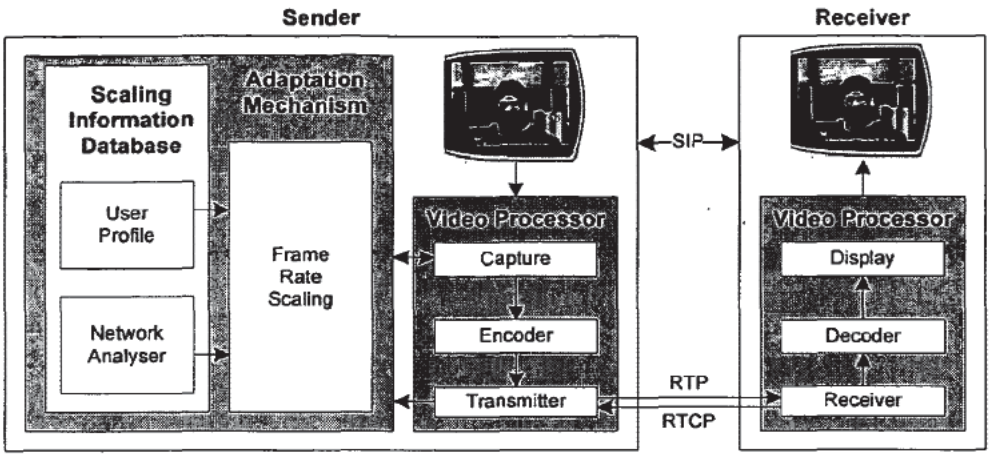


Figure 1 Basic System Operational View

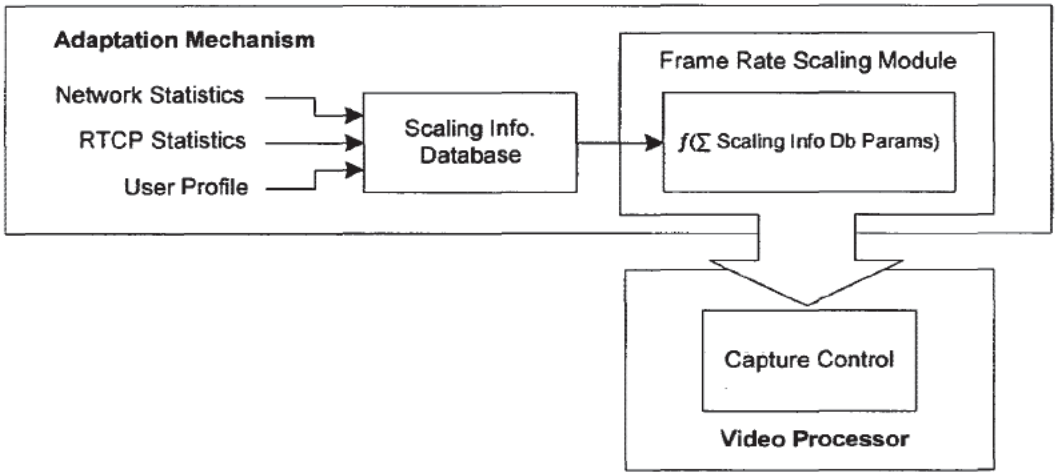


Figure 2

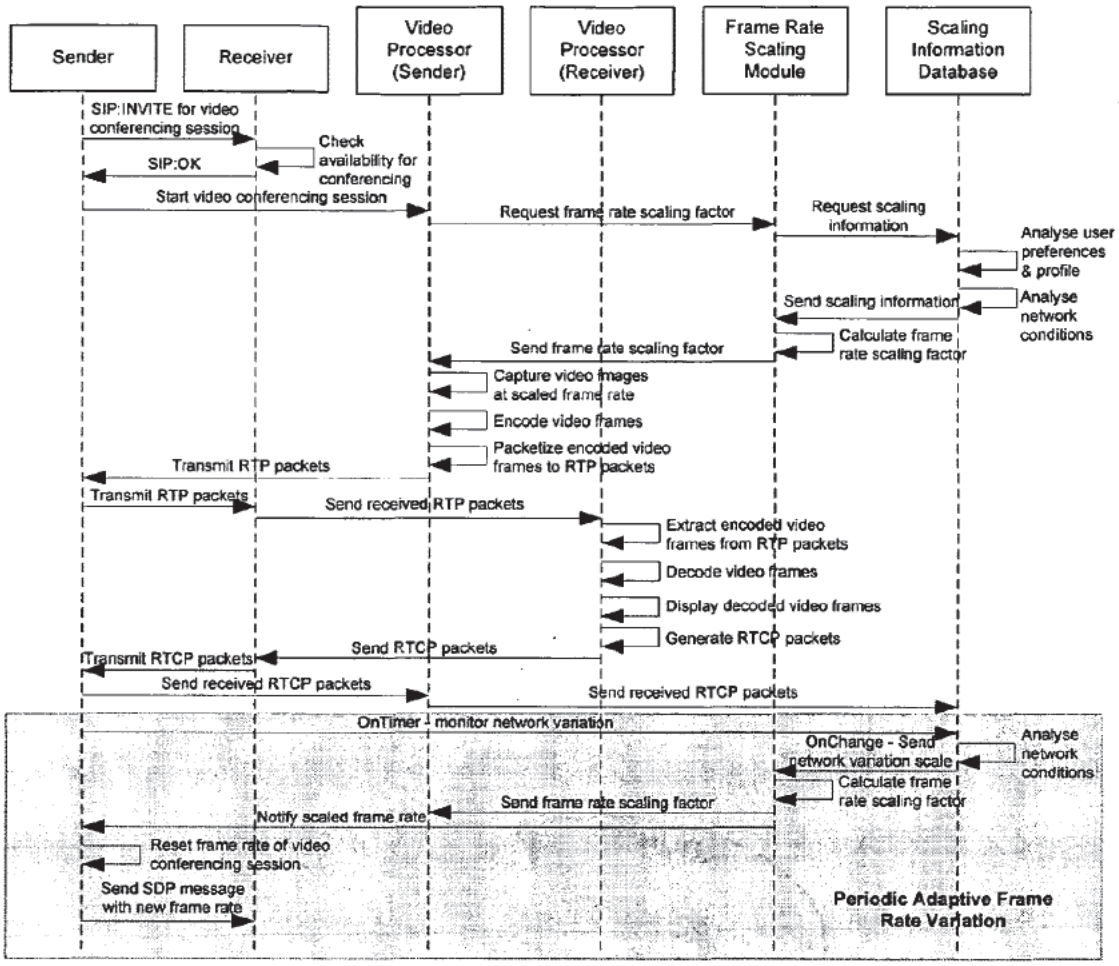


Figure 3 Sequence Diagram of Adaptive Video Frame Rate Application

European Patent Office

EUROPEAN SEARCH REPORT

Application Number
EP 05 25 6302

DOCUMENTS CONSIDERED TO BE RELEVANT

Category	Citation of document with indication, where appropriate, of relevant passages	Relevant to claim	CLASSIFICATION OF THE APPLICATION (IPC)
X,D	KIM JONGWON ET AL: "TCP-Friendly Internet Video Streaming Employing Variable Frame-Rate Encoding and Interpolation" IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY, IEEE SERVICE CENTER, PISCATAWAY, NJ, US, vol. 10, no. 7, 1 October 2000 (2000-10-01), pages 1164-1177, XP002369021 ISSN: 1051-8215 * page 1168, left-hand column, paragraph B * -----	1-5	H04N7/46
A	REED E C ET AL: "CONSTRAINED BIT-RATE CONTROL FOR VERY LOW BIT-RATE STREAMING-VIDEO APPLICATIONS" IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY, IEEE SERVICE CENTER, PISCATAWAY, NJ, US, vol. 11, no. 7, July 2001 (2001-07), pages 882-889, XP001083363 ISSN: 1051-8215 * page 885, right-hand column - page 886, left-hand column * -----	1-5	TECHNICAL FIELDS SEARCHED (IPC) H04N
A	N. SEELAM, P. SETHI, W.FENG: "A hysteresis based approach for quality, frame rate, and buffer management for video streaming using TCP" SPRINGER VERLAG, INTERNATIONAL CONFERENCE ON MANAGEMENT OF MULTIMEDIA NETWORKS AND SERVICES, vol. 2216, 29 October 2001 (2001-10-29), pages 1-15, XP002374403 * paragraph [3.1.3] * ----- -/--	1-5	

The present search report has been drawn up for all claims

2

Place of search	Date of completion of the search	Examiner
Berlin	28 March 2006	Raeymaekers, P

CATEGORY OF CITED DOCUMENTS

X : particularly relevant if taken alone
Y : particularly relevant if combined with another document of the same category
A : technological background
O : non-written disclosure
P : intermediate document

T : theory or principle underlying the invention
E : earlier patent document, but published on, or after the filing date
D : document cited in the application
L : document cited for other reasons

& : member of the same patent family, corresponding document

EPO FORM 1503 03.82 (P04C01)

European Patent Office

EUROPEAN SEARCH REPORT

Application Number
EP 05 25 6302

DOCUMENTS CONSIDERED TO BE RELEVANT

Category	Citation of document with indication, where appropriate, of relevant passages	Relevant to claim	CLASSIFICATION OF THE APPLICATION (IPC)
A	NEE P ET AL: "The performance of two-dimensional media scaling for Internet videoconferencing" NETWORK AND OPERATING SYSTEM SUPPORT FOR DIGITAL AUDIO AND VIDEO, 1997., PROCEEDINGS OF THE IEEE 7TH INTERNATIONAL WORKSHOP ON ST. LOUIS, MO, USA 19-21 MAY 1997, NEW YORK, NY, USA,IEEE, US, 19 May 1997 (1997-05-19), pages 223-234, XP010251701 ISBN: 0-7803-3799-9 * page 226, right-hand column * -----	1-5	
			TECHNICAL FIELDS SEARCHED (IPC)

The present search report has been drawn up for all claims

Place of search	Date of completion of the search	Examiner
Berlin	28 March 2006	Raeymaekers, P

CATEGORY OF CITED DOCUMENTS

X : particularly relevant if taken alone
Y : particularly relevant if combined with another document of the same category
A : technological background
O : non-written disclosure
P : intermediate document

T : theory or principle underlying the invention
E : earlier patent document, but published on, or after the filing date
D : document cited in the application
L : document cited for other reasons

& : member of the same patent family, corresponding document

2

EPO FORM 1503 03.82 (P04C01)

REFERENCES CITED IN THE DESCRIPTION

This list of references cited by the applicant is for the reader's convenience only. It does not form part of the European patent document. Even though great care has been taken in compiling the references, errors or omissions cannot be excluded and the EPO disclaims all liability in this regard.

Patent documents cited in the description

- US 6091777 A, W. Guetz **[0007]**

Non-patent literature cited in the description

- **T. TAKEBAYASHI et al.** Approaches to User-Centric Computing System for Ubiquitous Solutions. *Fujitsu Sci. Tech. J.,* 02 December 2003, vol. 39, 261-269 **[0007]**
- **M. NAGHSHINEH et al.** End-to-End QoS Provisioning in Multimedia Wireless/Mobile Networks Using an Adaptive Framework. *IEEE Communication Magazine,* November 1997 **[0007]**
- **J. KIM et al.** TCP-Friendly Internet Video Streaming Employing Variable Frame-Rate Encoding and Interpolation. *IEEE Transaction on Circuits and Systems for Video Technology,* October 2000, vol. 10 (7 **[0007]**
- *IBM. Functions of mobile multimedia QoS control, http://www.trl.ibm.com/projects/mmqos/system_e.htm* **[0007]**

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In Re Application of:	Examiner: Donabed, Ninos J
R Drew Major	Art Unit: 2144
Application No.: 11/116,783	Confirmation No.: 5535
Filed: April 28, 2005	
For: APPARATUS, SYSTEM AND METHOD FOR ADAPTIVE-RATE SHIFTING OF STREAMING CONTENT	

Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

PETITION FOR EXTENSION OF TIME PURSUANT TO 37 C.F.R. § 1.136 (a)

Applicant respectfully petitions pursuant to 37 CFR 1.136(a) for a one month extension of time to file this response to the Office Action mailed May 14, 2008. The extended period is set to expire on September 14, 2008. Authorization is given to charge the Applicant's Deposit Account No. 02-2666 in the amount of $60.00, to cover the fee for a one month extension of time.

Please charge any shortages and credit any overages to our Deposit Account No. 02-2666.

Respectfully submitted,

BLAKELY, SOKOLOFF, TAYLOR & ZAFMAN

Dated: September 11 , 2008 /Kevin O. Grange/
Kevin O. Grange
Reg. No.: 60,793

1279 Oakmead Parkway
Sunnyvale, CA 94085-4040
(408) 720-8300

CERTIFICATE OF ELECTRONIC FILING

I hereby certify that this correspondence is being submitted electronically via EFS on the date shown below.

/Kevin O. Grange/ September 11, 2008
Kevin O. Grange (Date)

Attorney's Docket No. 8532P001 PATENT
Confirmation No.: 5535

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In Re Application of:	
R. Drew Major	Examiner: Donabed, Ninos J.
Application No.: 11/116,783	Art Unit: 2144
Filed: April 28, 2005	
For: APPARATUS SYSTEM AND METHOD FOR ADAPTIVE-RATE SHIFTING OF STREAMING CONTENT	

Mail Stop Amendment
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

INFORMATION DISCLOSURE STATEMENT

Sir:

Enclosed is a copy of Information Disclosure Citation Form PTO-1449 or PTO/SB/08 together with copies of the documents cited on that form. It is respectfully requested that the cited documents be considered and that the enclosed copy of Information Disclosure Citation Form PTO-1449 or PTO/SB/08 be initialed by the Examiner to indicate such consideration and a copy thereof returned to applicant(s).

CERTIFICATE OF ELECTRONIC FILING

I hereby certify that this correspondence is being submitted electronically via EFS on the date shown below.

/Kevin O. Grange/ September 11, 2008
Kevin O. Grange (Date)

Pursuant to 37 C.F.R. § 1.97, the submission of this Information Disclosure Statement is not to be construed as a representation that a search has been made and is not to be construed as an admission that the information cited in this statement is material to patentability.

Pursuant to 37 C.F.R. § 1.97, this Information Disclosure Statement is being submitted under one of the following (as indicated by an "X" to the left of the appropriate paragraph):

_____ 37 C.F.R. §1.97(b).

__X__ 37 C.F.R. §1.97(c). If so, then enclosed with this Information Disclosure Statement is one of the following:

____ A statement pursuant to 37 C.F.R. §1.97(e) or

__X__ Deposit Account authorization for $180.00 for the fee under 37 C.F.R. § 1.17(p).

_____ 37 C.F.R. §1.97(d). If so, then enclosed with this Information Disclosure Statement are the following:

(1) A statement pursuant to 37 C.F.R. §1.97(e); and

(2) A check for $180.00 for the fee under 37 C.F.R. §1.17(p) for submission of the Information Disclosure Statement.

If there are any additional charges, please charge Deposit Account No. 02-2666.

Respectfully submitted,

BLAKELY, SOKOLOFF, TAYLOR & ZAFMAN LLP

Dated: September 11, 2008

/Kevin O. Grange/
Kevin O. Grange
Reg. No. 60,793

1279 Oakmead Parkway
Sunnyvale, CA 94085-4040
(408) 720-8300

Electronic Acknowledgement Receipt

EFS ID:	3930627
Application Number:	11116783
International Application Number:	
Confirmation Number:	5535
Title of Invention:	Apparatus, system, and method for adaptive-rate shifting of streaming content
First Named Inventor/Applicant Name:	R. Drew Major
Customer Number:	08791
Filer:	Kevin Grange/Leslie Rogan
Filer Authorized By:	Kevin Grange
Attorney Docket Number:	8532P001
Receipt Date:	11-SEP-2008
Filing Date:	28-APR-2005
Time Stamp:	21:16:04
Application Type:	Utility under 35 USC 111(a)

Payment information:

Submitted with Payment	yes
Payment Type	Deposit Account
Payment was successfully received in RAM	$240
RAM confirmation Number	6478
Deposit Account	022666
Authorized User	

The Director of the USPTO is hereby authorized to charge indicated fees and credit any overpayment as follows:

Charge any Additional Fees required under 37 C.F.R. Section 1.21 (Miscellaneous fees and charges)

File Listing:

Document Number	Document Description	File Name	File Size(Bytes)/ Message Digest	Multi Part /.zip	Pages (if appl.)
1		8532P001_ROA.pdf	78788 f10f5e7513bb78433f718844451923681a970fe1	yes	13
	Multipart Description/PDF files in .zip description				
	Document Description		**Start**	**End**	
	Amendment/Req. Reconsideration-After Non-Final Reject		1	1	
	Specification		2	3	
	Claims		4	8	
	Applicant Arguments/Remarks Made in an Amendment		9	13	
Warnings:					
Information:					
2	Information Disclosure Statement (IDS) Filed (SB/08)	8532P001_IDS_and_SB08.pdf	49137 74561b0fb83df1ddcc262594296b8159ba231ed4	no	4
Warnings:					
Information:					
This is not an USPTO supplied IDS fillable form					
3	Foreign Reference	EP0711077_Haskell.pdf	1510432 1c2693cd661c5d1d8421b0f3e910202f7b31274f	no	22
Warnings:					
Information:					
4	Foreign Reference	EP0919952_Mattavelli.pdf	2294545 878657d6e239971c161897028553db6d70c5fbaf	no	49
Warnings:					
Information:					
5	Foreign Reference	EP1641271_Klemets.pdf	1024812 ff83c3c62915214300d6a17a67c5afcd6bb31cc4	no	17
Warnings:					
Information:					
6	Foreign Reference	EP1670256_Klemets.pdf	2765717 02643fb68d918c659f73bf1d4627a6e1f90c4eaa	no	43
Warnings:					

Information:					
7	Foreign Reference	EP1777969.pdf	553655 155d901602ad05711f5215e490f3548078625be3	no	9
Warnings:					
Information:					
8	NPL Documents	CDN_Taxonomy.pdf	1264085 8fb6fc17c8aa4fde372b3e29a25326f4b3cbc9f4	no	44
Warnings:					
Information:					
9	NPL Documents	vp7_whitepaper.pdf	180190 29a758bc0f48c121e4ab78af643d2e696f979c1b	no	13
Warnings:					
Information:					
10	Extension of Time	8532P001_Petition_For_Extension_of_Time.pdf	14968 c38a99d40b1b793d33792716ae38e17bd376675c	no	1
Warnings:					
Information:					
11	Fee Worksheet (PTO-06)	fee-info.pdf	31954 67b9f1d83584a7b5166920520d57ec70211f6658	no	2
Warnings:					
Information:					
		Total Files Size (in bytes):	9768283		

This Acknowledgement Receipt evidences receipt on the noted date by the USPTO of the indicated documents, characterized by the applicant, and including page counts, where applicable. It serves as evidence of receipt similar to a Post Card, as described in MPEP 503.

New Applications Under 35 U.S.C. 111
If a new application is being filed and the application includes the necessary components for a filing date (see 37 CFR 1.53(b)-(d) and MPEP 506), a Filing Receipt (37 CFR 1.54) will be issued in due course and the date shown on this Acknowledgement Receipt will establish the filing date of the application.

National Stage of an International Application under 35 U.S.C. 371
If a timely submission to enter the national stage of an international application is compliant with the conditions of 35 U.S.C. 371 and other applicable requirements a Form PCT/DO/EO/903 indicating acceptance of the application as a national stage submission under 35 U.S.C. 371 will be issued in addition to the Filing Receipt, in due course.

New International Application Filed with the USPTO as a Receiving Office
If a new international application is being filed and the international application includes the necessary components for an international filing date (see PCT Article 11 and MPEP 1810), a Notification of the International Application Number and of the International Filing Date (Form PCT/RO/105) will be issued in due course, subject to prescriptions concerning national security, and the date shown on this Acknowledgement Receipt will establish the international filing date of the application.

(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) EP 1 641 271 A2

(12) **EUROPEAN PATENT APPLICATION**

(43) Date of publication:
29.03.2006 Bulletin 2006/13

(51) Int Cl.:
H04N 7/24 (2006.01) *H04N 5/00* (2006.01)

(21) Application number: **05018936.4**

(22) Date of filing: **31.08.2005**

(84) Designated Contracting States:
AT BE BG CH CY CZ DE DK EE ES FI FR GB GR HU IE IS IT LI LT LU LV MC NL PL PT RO SE SI SK TR
Designated Extension States:
AL BA HR MK YU

(30) Priority: **24.09.2004 US 948999**

(71) Applicant: **Microsoft Corporation**
Redmond WA 98052 (US)

(72) Inventors:
- **Klemets, Anders E.**
  Redmond,
  WA 98052 (US)
- **Stewart, James C.**
  Redmond,
  WA 98052 (US)
- **Zhang, Geqiang**
  Redmond,
  WA 98052 (US)

(74) Representative: **Goddar, Heinz J. et al**
FORRESTER & BOEHMERT
Pettenkoferstrasse 20-22
80336 München (DE)

(54) **Methods and systems for presentation of media obtained from a media stream**

(57) An implementation, as described herein, facilitates fast start-up of a new media stream while mitigating and in some implementations, avoiding temporal interruption (i.e., "stutter") of the presentation of that new media stream. At least one implementation, described herein, measures a rate at which a multimedia stream is received at a downstream component configured to present the multimedia to a user; and, ascertains a duration of time to buffer the received multimedia before presentation thereof, at least in part utilizing the measured rate.

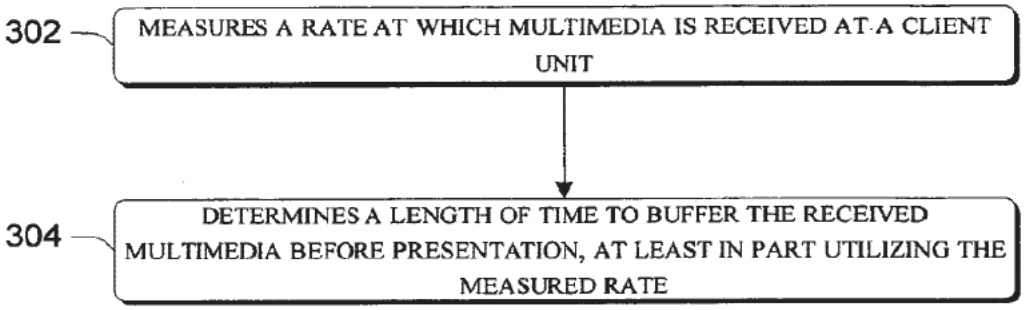


Fig. 3

EP 1 641 271 A2

Printed by Jouve, 75001 PARIS (FR)

Description

TECHNICAL FIELD

[0001] This invention generally relates to a data signal transmission technology.

BACKGROUND

Introduction

[0002] Digital media streaming technology can allow a user to see and/or hear digital media received from a remote location such as media sending unit. A user at a client or receiving unit may select a file for viewing. A user may alternatively select data from a real time encoder for viewing. Media or data from the selected file/encoder can be sent as a media stream to the receiving unit where it is played or otherwise presented for the user. Selecting a file can include various scenarios. For example, a user may click on a movie trailer at a web-site or change channels when watching TV through an internet provider. Since this media often includes data to create both audio and visual sensations it may be referred to as "multi-media" data or content.

[0003] Due to the amount of data required to accurately represent such multimedia, it may be compressed and encoded at an encoder before being sent to the client unit. To reproduce the original media for presentation, the media is typically decompressed and decoded before it is presented. Both the encoder and the decoder may utilize buffers to compensate for any variability in the bit rate of the content. This process is sometimes analogized to a leaky bucket model, where the encoder is encoding according to a leaky bucket and the client unit has a similarly sized bucket to avoid underflow. This comes about because the bit rate of the content may be somewhat variable, and the bucket size is basically a measure of the maximum variability of the bit rate.

Transmission of Media Streams

[0004] For practical purposes, media streams carrying audio and/or video from a media-stream encoder are typically not continuous, but are instead broken into multiple packets for transmission. In some formats such packets comprise one or more intra frames or I frames sometimes referred to as key frames. An I frame is encoded as a single image, with no reference to past or future frames.

Underflow

[0005] When a receiving unit runs out of data to decode (or present), it is called "underflow." Underflow occurs when the receiving unit is ready to decode (or present) the next frame, but the receiving unit has not yet received (or decoded) all of the data of that frame.

[0006] The practical and noticeable manifestation of an underflow is a temporal interruption (i.e., "hiccup" or "stutter") in the motion video presentation rather than the desired effect, which is a smoothly playing motion video. For example, instead of showing a motion video at a fixed frequency (e.g., 15 frames per second), a receiving unit experiencing underflow would show a frame of the video stream followed by a noticeable delay before the next or a later frame would display. This may continue for several seconds or minutes.

[0007] The conditions for an underflow are particularly ripe when a new file or encoded content is selected, i.e. changing channels. If the receiving unit immediately presents frames as soon as it receives and decodes the incoming media stream of the new channel, an underflow condition is likely to arise.

[0008] Instead, it is common for presentation of the multimedia data to be delayed for a time period sufficient to fill the client side buffer. As the multimedia data is decoded and presented, the receiving unit empties the data stored in its buffer. However, while the receiving unit is presenting decoded data, more encoded data is downloaded to re-fill the buffer. As long as the data is downloaded at least as fast as it is being played back in such a way that the buffer is never completely empty, the file should play smoothly.

[0009] The conventional techniques, described above, that are designed to prevent underflow, can produce an annoying delay between a user selection and actual presentation of the selection to the user.

SUMMARY

[0010] Various implementations described herein provide methods and systems for presenting media obtained from a media stream. At least one implementation, described herein, measures a rate (measured rate) at which a multimedia stream is received at a downstream component configured to present the multimedia to a user; and, ascertains a duration

(measured duration) of time to buffer the received multimedia before presentation thereof, at least in part utilizing the measured rate.

BRIEF DESCRIPTION OF THE DRAWINGS

[0011] The same numbers are used throughout the drawings to reference like elements and features.

[0012] Fig. 1 represents a schematic diagram illustrating an exemplary media-stream network topology capable of (wholly or partially) implementing at least one embodiment described herein.

[0013] Fig. 1A represents a schematic diagram illustrating an exemplary media-stream network topology capable of (wholly or partially) implementing at least one embodiment described herein.

[0014] Fig. 2 represents a flow diagram illustrating a methodological implementation described herein.

[0015] Fig. 3 represents a flow diagram illustrating a methodological implementation described herein.

DETAILED DESCRIPTION

Overview

[0016] The following systems and methods reduce buffering delay while mitigating stuttering of a multimedia presentation. Some implementations may reduce buffering delay while altogether avoiding stuttering of the presentation. A sending unit streams media from a selected file or live encoded session to a client unit (hereinafter the term "source" is utilized to refer to either a file or a session or equivalents). The term "streaming rate" is used to identify the rate at which the media is streamed from the sending unit to the client unit. Upon receipt, the client unit can immediately or subsequently present the streamed media to a client or user. A rate at which the data is utilized for the presentation is.known as the presentation rate. The average presentation rate may be approximately equal to the average streaming rate; however, at instantaneous points in time, the streaming rate and the presentation rate may be different.

[0017] Prior to streaming the media, the sending unit sends advisory data relating to the media to the client unit. The advisory data is described in more detail below and can comprise various components such an acceleration rate at which the sending unit will stream the media, an acceleration interval, various buffer values, a start-up profile, and/or a recommended or estimated buffer duration, among others. The acceleration rate is the streaming rate at which the sending unit will attempt to send data, expressed relative to the presentation rate, or it could be expressed as an absolute value in terms of bits per second or bytes per second. The sending unit may start streaming at an acceleration rate, such as two times (2x) the presentation rate, for a duration of time known as the acceleration interval. Typical acceleration rates could be in the range of 1.0x, 1.2x, 1.5x, 2.0x, and 3.0 x, though these are simply examples and any other value can occur depending on the characteristics of the sending unit and the intervening network between the sending unit and the client unit.

[0018] The client unit can buffer data in order to have sufficient data to avoid an underflow condition during the presentation. The client unit can add data to the buffer in at least two ways. First, the client unit can buffer initially before beginning presentation of the multimedia. In such an instance all of the received data is buffered. However, such buffering results in start-up delay where start-up delay is the period between, for example, a user pressing play and actually seeing or hearing the multimedia presentation. Second, the client unit can also buffer during the acceleration interval while presenting the media since the client unit may be receiving data at a greater rate than it is presenting the data. In one example of the later instance, a presentation rate may be at 1 megabit per second (Mbps). The sending unit may stream the media during the acceleration interval at 2x or 2 Mbps. The client unit can buffer this excess data for the duration of the acceleration interval while simultaneously utilizing data for the presentation.

[0019] The advisory data can, in some implementations, comprise a start-up profile(s). Generally, the start-up profile can relate to the amount of buffer required or recommended at one or more points in the presentation to avoid underflow. The sending unit can compile an estimated start-up profile before sending the media stream. In various implementations an estimated or first start-up time is calculated which satisfies the start-up profile at an estimated acceleration rate. The start-up time is a measure of the amount of buffer that the client should accumulate from when the client unit begins to receive the media stream until beginning to present the media. This amount of buffer can be measured either in units of bits, or in units of time, where units of time are simply the bit values divided by the average bit rate.

[0020] Once the client unit begins to receive the media stream the start-up time can be verified, or recalculated. Some implementations can recalculate the start-up time or buffer duration utilizing a measured acceleration rate at which the client unit is receiving the media stream. Utilizing the start-up profile, along with the actual measure of the amount of data buffered during the acceleration interval, can allow a more accurate second or measured start-up duration to be calculated. Other implementations may simply utilize the measured acceleration rate to verify the accuracy of the estimated start-up duration before beginning to present the multimedia.

[0021] In some implementations, the method can compare the measured acceleration rate to the earlier estimated

acceleration rate and act accordingly. If the comparison shows that the measured acceleration rate is slower than the estimated acceleration rate then the method can recalculate the buffer time and buffer longer. If the data is received at a rate which exceeds the estimated rate then the method may be able to shorten the buffer time. The amount of pre-start-up buffering calculated in this way may be substantially reduced compared to existing technologies. This recalculation of the startup profile can occur at any point after the client begins to receive data. For example it could occur at regular time intervals of say 100ms or it could occur at fixed fractions of the initial estimated startup time -for example at 50% of the initial calculated buffer time.

[0022] The client unit is the device closest to presentation of the file to the user, or stated another way, the client unit is the device farthest downstream. Various delays, bottlenecks and/or other intervening conditions along the media path result in the client unit being in a better position to make the start-up calculations than upstream components.

[0023] Among other advantages, allowing the client unit to determine how long to buffer data prior to start-up can reduce inaccuracies in the buffer time which might otherwise result from intervening events. For example, an upstream sending unit may calculate a buffer duration for an estimated transmission rate that it intends to stream the media to the client unit. The client unit may in fact receive the media stream at a different, e.g. higher or lower rate. As such a calculation made by the client unit can be more accurate than one made at the sending unit. The client unit can have the most accurate view of the rate at which it is receiving data and can most accurately calculate the amount of time it should buffer in order to avoid underflowing.

Exemplary Implementations

[0024] The following description sets forth one or more exemplary implementations of methods and systems for presentation of media obtained from a media stream that incorporate elements recited in the appended claims. These implementations are described with specificity in order to meet statutory written description, enablement, and best-mode requirements. However, the description itself is not intended to limit the scope of this patent.

Media-Stream Network Topology

[0025] Figs. 1 and 1A illustrate high level representations of exemplary media-stream network topology 100 in which the techniques, systems, and other aspects described herein may be implemented (partially or wholly).

[0026] The exemplary topology 100 includes one or more media-stream sending units 110 such as a server, which is communicatively coupled to a receiving or client unit 120 via a transmission means 130. Receiving unit 120 can comprise, among others, a personal computer (PC), cell-phone or set-top box. In this embodiment, receiving unit 120 further comprises a display or presentation device 140 which comprises a display screen and one or more speakers. The functionality of the sending unit can be achieved by a single component or multiple components, similarly the functionality of the client unit can be achieved by a single component or multiple components.

[0027] Sending unit 110 can store sources such as files for transmission to receiving unit 120 and/or can access the sources via network 150. In various embodiments, network 150 may be a limited access network or can comprise the internet, among other configurations. Network 150 may be coupled to an encoder 160 for encoding and compressing multimedia data or content which is accessed by sending unit 110. Alternatively, sending unit 110 can encode the media prior to streaming it over transmission means 130 to client unit 120 which can decode the media prior to presentation.

[0028] Topology 100 can allow a user engaging client unit 120 to select a source, such as a file, for presentation. Client unit 120 can request that sending unit 110 stream media from the file to the client unit. Transmission means 130 can comprise any suitable transmission means such as Ethernet, broadband, dial-up and wireless. Client unit 120 can present the media to the user via presentation device 140. Although client unit 120 and presentation device 140 are described here as distinct components, other implementations may integrate their functions into a single unit.

First Exemplary Method and Implementation

[0029] Fig. 2 represents a flow diagram of acts in accordance with one method for presentation of media obtained from a media stream.

[0030] Act 202 requests to receive a multimedia stream. Such a request can be initiated by a user as a selection on a client unit. For example, the user may click on content in a web-browser, or may switch channels while watching TV over an internet provider. The request may simply be to stream a selected source such as a file or may be more specific. For example, the request may be to stream content corresponding to 10 seconds of presentation time starting at 20 seconds into the selected file. The request can cause the file corresponding to the play request to be accessed, such as by the sending unit.

[0031] At least in part by accessing the source, advisory data associated with presentation of the source's data can be obtained. Advisory data can comprise information stored in a header of the file by a program which originally created

the file. This program examines the file to determine a maximum or recommended amount of buffer for any point in the file. The buffer data can be written into a file header where it can be accessed by the sending unit.

[0032] The sending unit may simply gather the advisory data and send it to the client unit. In other implementations the sending unit may process various components of the advisory data, such as the buffer data, prior to sending it to the client unit. In some examples, the sending unit calculates one or more estimated start-up profiles which it sends to the client unit. The start-up profile can relate to the amount of data which the client unit should acquire at various points in the presentation. For example, the start-up profile can comprise a set of values indicating the maximum amount of underflow, the time at which the underflow occurs, and the acceleration rate used to calculate the underflow. In some implementations described below, data relating to multiple acceleration rates can be contained in a single start-up profile. Other implementations may generate separate start-up profiles for individual acceleration rates.

[0033] In one implementation, the start-up profile is calculated by adding the packet sizes at each presentation time to determine presentation requirements at various times during presentation of the file. Based upon various specific acceleration rates, the method can calculate an estimate of how much data the client should have received due to accelerated streaming at each presentation time. The method can provide an estimated buffer duration prior to beginning the presentation in order to avoid underflowing the client buffers (running out of data).

[0034] The start-up profile can comprise various forms. In but one example, the start-up profile comprises a data array. Array elements can be in relation to specific acceleration rates. For example, for each acceleration rate the array comprises an underflow byte value and the time at which this underflow occurs.

[0035] In some implementations, the start-up profile relates to two buffering conditions for a given acceleration rate. The first buffering condition can relate to a recommended or estimated buffer size prior to starting presentation of the file. The second buffering condition can relate to a recommended buffer size at the end of the acceleration interval. These two conditions will be described in more detail below.

[0036] The first buffer condition can be utilized to calculate how long the client unit should buffer initially so that by each underflow time it has at least an adequate number of bytes to reduce and/or avoid stuttering. The second buffer condition relates to calculating a worst case buffer scenario to ensure that when the client unit fmishes the acceleration period that it has a sufficient amount of buffer for any point in the presentation. This worst case buffer scenario can be the header value that was written by the encoding process that indicates the recommended amount of buffer that the client should have to avoid underflowing for the worst case or most "action" portion of the content. This second calculation is based on the premise that the client unit should only rely on increasing its buffer amount during the acceleration interval. In some implementations, the sending device sends, in the advisory data, the header values relating to the recommended amount of buffer. In other implementations, the sending device, as part of the advisory data may expressly provide a recommended full buffer value for the client device to achieve by the end of the acceleration interval.

[0037] In such an implementation, the method determines an amount of buffered data sufficient for the entire presentation which the client unit should achieve prior to leaving the acceleration interval. New data is of course added to the buffer during the presentation as the client unit continues to receive streaming media, but the new data is added at a rate which is closer to the rate that data is decoded for presentation. As such the total buffer size should not be depended upon to increase significantly. This implementation of the method calculates a buffer duration before presentation to satisfy both of these two buffer values at the estimated acceleration rate that the sending unit intends to stream the media.

[0038] In still another implementation, the sending unit may perform the buffer calculation, and simply pass the client unit a single value for the amount of time that the client unit should buffer. Alternatively, the sending unit can pass an array of buffer times to the client unit, for different acceleration rates, and the client unit can simply pick the time closest its own measured acceleration rate, or interpolate a more accurate value. The skilled artisan should recognize that these are but examples of which components execute the algorithm(s). For example, in some implementations more or all of the calculations may be executed by the sending unit. In still other implementations, more or all of the calculations may be executed by the client unit.

Exemplary Sending unit Algorithm

[0039] This section describes a more detailed example of an exemplary algorithm utilized by the sending unit in one implementation. This algorithm is configured to function with various client requests. In one such example, the client requests to join an already running broadcast publishing point such as a TV program. In another such example, the client requests to play a file from an on-demand publishing point that might be referenced on a web-site.

[0040] For the broadcast publishing point example, the method finds the I frame that will be the starting point of the media it streams to the client unit. The algorithm below operates on that set of data which will be sent to the client unit, namely from that I frame onward. In the on-demand example, the calculation simply starts at the first packet in the presentation. Alternatively, the user can seek to any point in the file. In that case the method finds an I frame close to that seek point and begins sending data from that point. The calculation begins from that I frame.

[0041] In the case of real time encoded content, there is no actual file that is streamed to the client unit. However, the

sending unit may buffer some amount of the real time content, and send data from that buffer to the client unit. That buffer is logically equivalent to a file in many ways for the purposes of startup profile. That buffer could be arranged in a circular manner where as new data is added on, old data is deleted from the end. In order to provide acceleration from the sending unit to the client unit for real time encoded content, the sending unit simply sends part of this buffered amount more quickly than the actual playback rate to the client.

[0042] In this particular implementation, the sending unit generally performs a profiling process that is used to fmd peaks in client unit buffer requirements over a particular period of the media stream. This particular implementation calculates the data for each presentation time, by summing the previous packet sizes. The data for any particular presentation time is the sum of all packets with lesser presentation times. This value is a measure of the amount of data that the client unit should have, in order to avoid under-flowing.

[0043] The implementation further calculates the average bit rate of the stream by taking the number of bytes sent to the client unit up to that point, i.e. presentation time, and dividing it by the difference in send time. This value is the local actual average bit rate, and it could be different from the average bit rate value stored in the file header, over portions of the stream. This particular implementation utilizes this value rather than the average value from the file header because it may be more accurate over the portion of the file which is being examined.

[0044] The implementation calculates an amount of streamed data which will be sent to the client unit at each of the same presentation time values, using the above calculated average. The method can do this calculation for several acceleration rates.

[0045] The implementation can calculate the estimated excess amount of client buffer at each interval by subtracting the second number from the first (needed data- streamed data), for each acceleration rate. As long as this number is negative, the client will be streamed more data than it needs and it will not underflow. However if this value is positive at any point in the presentation then an underflow condition will exist at the client unit.

[0046] The method also keeps track of the largest underflow value of the client buffer and the presentation time at which the largest underflow occurs for each acceleration rate. The point in the presentation time where the client will have the greatest buffer requirements may change as the acceleration rate changes. In such circumstances the method calculates the underflow values at several acceleration rates.

[0047] This will produce one underflow value, such as an amount of data, and presentation time of that underflow value for each acceleration rate. The underflow value will often be different for each acceleration rate, because the rate at which data is delivered to the client will be different due to the differing acceleration rates. However, the presentation time value for each underflow could be the same for several acceleration rates, or it could be different. Depending on the shape of the estimated data delivery versus time curve different acceleration rates could yield underflows at different times.

[0048] This implementation then can send the client unit these calculated underflow buffer values and the times at which they occur, the acceleration rate used to calculate them and the average bit rate to that point in the presentation. In some instances the advisory data compiled above can be sent to the client unit as the start-up profile for the client unit, where the start-up profile contains an estimated buffer duration as calculated by the sending unit. For example, the sending unit may send the amount of buffer utilized for each of one or more acceleration rates, one of which may be the estimated acceleration rate at which the sending unit intends to stream the data. Based at least in part upon this buffer data, the client unit can subsequently interpolate or otherwise calculate the best value to use knowing more accurately what its acceleration rate is as will be described in more detail below. Other implementations may have even more of the calculations done by the sending unit. For example, the sending unit could monitor the amount of data that it has sent to the client unit and then send a signal to the client to exit buffering and begin presentation.

[0049] One example of such a start-up profile is provided below.

```
X-StartupProfile:
Rate=10,12,15,20,30;MaxBytes=123424,113482,103245,94030,85125;Tim
e=520,280,160,120,40;StartTime=5500;LastTime=1640;MaxDiffTime=0;
MaxDiffSndTime=0;ByteRate=135626,126514,119730,116204,106546
```

[0050] Act 204 receives advisory data relating to the multimedia stream. As described above, the advisory data can relate to the start-up profile at one or more acceleration rates which can be employed by the sending unit and supported by the intervening network or transmission means since the network itself may be the limiting factor in the achievable bit rate between the sending unit and the client unit.

[0051] Below is but one example of a header utilized by the sending unit and client unit to convey which features are supported by the respective devices. In this particular example the sending unit and client unit can support advisory data

in the form of a start-up profile(s).

Supported: com.microsoft.wm.startupprofile

[0052] Some implementations may allow the client unit to have some interactive ability or algorithmic ability to control the sending unit based upon the advisory data. For example, the client unit may choose an acceleration rate, and an acceleration interval based upon the advisory data, and request the sending unit to stream the media accordingly. In another example, the sending unit may determine the acceleration rate, but the client unit may be allowed to request the acceleration interval. As will be described in more detail below, at any acceleration rate the client unit can calculate how long the sending unit should accelerate, to avoid additional start-up delay due to the full buffer value at the end of the acceleration interval. Therefore, upon receiving the advisory data, specifically the estimated acceleration rate from the sending unit, the client unit may calculate and request a corresponding acceleration interval. Alternatively, the client unit may calculate and request a corresponding acceleration interval before it has actually received the advisory data. It may pass this requested acceleration interval to the sending unit during the initial request to receive the media stream.

[0053] In other implementations, the sending unit makes decisions regarding the media stream without deference to the client unit. In one such implementation, the sending unit simply informs the client, in the advisory data, of the estimated acceleration rate and the acceleration interval and streams the media. Note that in some network topologies the acceleration interval is accurately controllable by the sending unit whereas the acceleration rate depends on the characteristics of the intervening network as well as the rate at which the sending unit is sending.

[0054] If the advisory data contains an estimated acceleration rate and resultant estimated buffer duration the method can proceed to act 206. Alternatively or additionally, the client unit may determine an estimated acceleration rate and/or associated estimated buffer duration before proceeding to act 206. For example, the client unit may estimate the acceleration rate of the future stream utilizing packet pair or run time packet pair. In another example the client determines the estimated rate via a URL modifier such as "?wmbitrate=xxx".

[0055] One example for calculating the estimated buffer duration can be seen in the following scenario. In this instance the acceleration interval is 1.0 seconds, the average presentation rate is 1.0 Mbps and the worst case buffer demand is 2.5 megabits of data. The estimated acceleration rate is 3 Mbps. As such 2.0 megabits of data can be buffered during the acceleration interval if playback occurs as soon as streaming commences leaving a person to believe that 0.5 megabits should be buffered prior to presentation. However, since there must be some buffering prior to playback, the client will not experience acceleration for a full 1 second after playback starts because part of that acceleration time will be spent in buffering before playback starts. In this particular case if the client buffers for ½ second it will acquire 3 / 2 = 1.5 Mbits of data. Then it plays for ½ second during which time the sending unit is still accelerating at 3 Mbps, but 1 Mbps is consumed by playback, therefore 2 Mbps is available to be buffered. Since this goes on for ½ second then ½ $\times$ 2 = 1 Mbit of data is buffered during playback. This results in 1.5 + 1 = 2.5 megabits of data in the buffer at the end of the acceleration interval, which is sufficient to avoid stuttering. The 2.5 megabits that should be buffered is divided into two pieces: the 1.5 Mbits buffered before the presentation starts and the 1.0 Mbits buffered after the presentation starts.

[0056] Act 206 begins to receive and buffer the multimedia stream. The media stream can be received by the client unit from the sending unit. Buffering can be started based upon the estimated buffer duration described above.

[0057] Act 208 ascertains a measured rate at which the multimedia stream is being received. The client unit may measure the rate itself or may ascertain the receiving rate from a third party component. The skilled artisan should recognize suitable methodology for measuring the rate at which the media stream is being received at the client unit.

[0058] Act 210 calculates a duration to buffer the media stream based, at least in part, on the measured rate. Hereinafter this duration will be referred to as the measured buffer duration or measured duration to distinguish it from the estimated buffer duration described above and is not intended to imply that the duration is itself measured but rather that the measured buffer duration is calculated utilizing the measured acceleration rate. The measured buffer duration can be more accurate than the estimated duration, based on the worst case buffer requirements of the entire multimedia content file, and can allow the method to reduce the likelihood of stutter while reducing start-up time.

[0059] In this example, it may be that the client unit is not actually receiving the streaming media at the estimated rate. As a result, the estimated buffer duration may actually be too short or longer than necessary. As the method approaches the estimated buffer duration, the measured acceleration rate is utilized by the client unit to recalculate the startup profile calculation to see, if in fact, it has buffered long enough at the measured acceleration rate to buffer adequate data.

[0060] If the calculation reveals that the client unit has not buffered long enough, then the client unit will buffer an incremental amount. For example the client unit may choose to recheck the buffer calculation when the amount is half the difference between the estimated buffer duration and the measured buffer duration. The client may wish to put a minimum bound on this difference, such as 100ms to avoid performing the calculation repeatedly in a small time interval. When this extra buffer duration is achieved, the client again performs the startup profile calculation with the current measured acceleration. Since the acceleration may vary somewhat, it is possible that the value is now higher and

potentially the client is able to exit the buffering state at this point. One reason for choosing a new buffering time that is half the difference between the old and new, is to give the client unit the opportunity to exit buffering as quickly as possible.

Exemplary Client Algorithm

[0061] This particular implementation utilizes the data array of the start-up profile supplied by the sending unit as described above. The startup profile data also includes a measure of the total amount of presentation time profiled on the sending unit as well as the maximum difference between the audio and video send times, if the audio is sent after the video. The audio and video send times are discussed in more detail below.

[0062] In this particular instance individual elements of the array for each acceleration rate comprise an underflow byte value, the time at which this underflow occurs, the acceleration rate used in this calculation, and the average bit/byte rate of the file up to that point. The sending unit's start-up profile indicates a number of extra bytes out in the future (above the average streaming rate multiplied by acceleration) that the client should have in its buffer at a particular time to avoid under-flowing. As stated above, the client unit acquires those extra bytes two ways: by initial buffering and by acceleration during the streaming time. The client unit calculates how long it has to buffer initially so that by each underflow time T, it has at least the right number of bytes to avoid an underflow.

[0063] The data array provides underflow values which are tied to a specific acceleration. Thus, the sending unit is telling the client unit the amount of underflow that will occur for specific values of acceleration at some point T in time. It is very unlikely that the measured acceleration rate at which the client unit receives the media stream exactly matches the sending units' estimated acceleration. As such, initial buffer durations of the data array may result in the client unit buffering too much or too little data. The client recalculates the amount of buffer; using its own measured acceleration. In this particular implementation the client unit then interpolates its own value for underflow at the measured acceleration rate that it achieves.

This exemplary client calculation proceeds as follows:

[0064] The client estimates the acceleration rate or relies on an estimate from the sending unit to determine the estimated buffer duration.

[0065] Once the client unit begins to receive the media stream and ascertains the measured rate at which it is receiving the media stream it can utilize the measured rate to calculate the measured buffer duration for each of the underflow/time pairs passed to it by the sending unit, using the sending unit's average bit rate of the content.

[0066] In this implementation, the sending unit may have previously sent in the advisory data a maximum acceleration parameter which tells the client that it will not send data any faster than a particular amount. The client unit utilizes this amount as an upper bound on the acceleration parameter in the calculation. Knowing the measured acceleration rate the client unit can calculate the number of excess bytes it will have accrued (due to acceleration) by any particular time after it starts presenting.

[0067] The client unit then subtracts the calculated bytes from the underflow byte value passed down by the sending unit. If there is a positive result then the client unit should buffer long enough to accumulate that extra number of bytes. The client unit keeps track of the maximum value of the buffer bytes using each of the underflow/time value pairs from the sending unit in its calculation. The client interpolates each underflow value from the sending unit to its own measured acceleration rate, before using that value in the calculation. Rather than keeping track of buffer bytes, the client can equally well perform the calculation in the time space by dividing the byte values by the average bit rate value passed down from the sending unit. Using time rather than bytes might be more convenient for some client units that use presentation or send time (rather than acquired bytes) as the measure of how much data they have received, and when to exit buffering and start playback.

[0068] In an alternative implementation, the client unit calculates the measured buffer duration only for the underflow/time pairs passed to it that most closely matches its measured acceleration rather than performing the calculation for all underflow/time pairs. The client unit may also use some other subset of the underflow/time pairs, such as the two pairs closest to its own measured acceleration.

Additional Factors

Send Chunk Size

[0069] As noted above, streaming media commonly comprises grouped packets that are sent at intervals rather than continuously. In recognition of this process, some implementations may add an estimate of sending unit bandwidth variation, which is the interval between sends from the sending unit. For example, with some sending unit configurations, the sending unit will send data in 1 second chunks. On a link that is high speed relative to the content, sending data in

large chunks could cause the client to come out of buffering too early if no account is taken of this chunk size.

[0070] The reasoning for such occurrences is that following each large chunk send there may be a significant period (almost 1 second) where no data will arrive from the sending unit. However, in some configurations, the client calculation assumes that data is arriving linearly at the accelerated rate. If it just barely gets enough buffered data to satisfy its buffer requirements from the 1 second send, then it will likely run out of data before the next 1 second send occurs, because it is assuming that it received data linearly at the average acceleration rate.

[0071] To avoid this problem the sending unit can send an indication of its send chunk size to the client unit, and the client unit adds this amount to its buffer time. In one such example where the sending unit sends in 150ms chunks (for 1.2Mbit content). Responsively, the client can add 150-200ms to its startup calculation to avoid this chunk effect.

Audio Video separation

[0072] In some implementations the advisory data received at act 204 may contain information relating to any separation of corresponding portions of the audio and video in the media stream. In some file formats corresponding audio and video data may not be sent together. For example, if the video for presenting at time 1 second is 1000 bytes into the file and the audio for the same 1 second presentation time is 2000 bytes into the file then the client unit has to wait for the audio before presenting this portion of the file. This factor may outweigh other fast start-up buffer considerations since it would be undesirable to begin presentation until the client unit has both the corresponding audio and video. For example, if the audio is three seconds later than the video, the method may wait for that 3 second duration even if it otherwise has enough buffer to start playback and avoid stuttering. In one such implementation, the sending unit sends this information to the client unit as a header. One header for the audio video relationship and one for the startup profile buffer times for given acceleration rates.

[0073] In one such implementation, the sending unit attempts to quantify the lateness of the audio with respect to the video by looking at the difference in the presentation times of the audio and video packets in the file.

[0074] In this particular implementation, each time an audio packet is encountered, the method calculates the difference between the presentation time of the last video packet and the audio packet. When this value is positive (video presentation time is greater than the audio presentation time) it means that the audio data corresponding to this video sample is at some point further along in the file - at some point in the future and the client unit will have to wait for that audio to be able to synchronize audio and video. The method can save the maximum of the values calculated in the first step, as a worst case measure of how far behind the audio is relative to the video. The method can then send the maximum value of the audio lateness to the client unit.

[0075] This particular implementation can repeat this calculation each time it identifies an audio sample, it can then add the difference in send time to the calculated value, since this sample will be sent that amount of time later, contributing further to the lateness of the audio.

[0076] In another implementation a video sample is spread across several packets with an audio sample interspersed. The calculation can find the last video packet at a given presentation time and compare that against the audio presentation time.

[0077] Act 212 initiates presentation of the multimedia based on the measured buffer duration. The presentation can be initiated at a reduced time while mitigating and/or avoiding stutter during the presentation.

Second Exemplary Method and Implementation

[0078] Fig. 3 represents a flow diagram of acts in accordance with one method for presentation of media obtained from a media stream.

[0079] Act 302 measures a rate (measured rate) at which multimedia is received at a downstream component configured to present the multimedia. Measuring the rate at which the multimedia is being received can be achieved utilizing any suitable means. Measuring the receive rate at this relative position can reduce discrepancies, such as those caused by intervening events or conditions, which may reduce the accuracy of attempting to estimate such a rate farther upstream.

[0080] Act 304 determines a duration of time to buffer the received media before presentation, at least in part utilizing the measured rate. In but one implementation, the method begins buffering the streaming media and calculates a duration to buffer prior to starting playback based upon the measured receive rate. Utilizing the measured rate in the buffer duration calculation can result in a reduced start-up time while still avoiding stuttering of the presentation. Acts 302 and 304 may be repeated multiple times prior to beginning presentation of the media to even more precisely determine the buffer duration.

Computing Environment

[0081] The one or more embodiments, as described herein, may be implemented with numerous other general purpose

or special purpose computing system environments or configurations. Examples of well known computing systems, environments, and/or configurations that may be suitable for use include, but are not limited to, personal computers, server computers, thin clients, thick clients, hand-held or laptop devices, cell phones, multiprocessor systems, microprocessor-based systems, set-top boxes, programmable consumer electronics, network PCs, minicomputers, mainframe computers, distributed computing environments that include any of the above systems or devices, and the like.

[0082] The one or more implementations, as described herein, may also be practiced in distributed computing environments where tasks are performed by remote processing devices that are linked through a communications network. In a distributed computing environment, program modules may be located in both local and remote computer storage media including memory storage devices.

[0083] The one or more implementations, as described herein, may be described in the general context of processor-executable instructions, such as program modules, being executed by a processor or a computer. Generally, program modules include routines, programs, objects, components, data structures, etc. that perform particular tasks or implement particular abstract data types and functions. Typically, the functionality of the program modules may be combined or distributed as desired in various embodiments.

Conclusion

[0084] The above described implementations allow a unit configured to actually present media content to a user to make the final determination regarding a duration to buffer a media stream before presenting the media to the user. Such a configuration can reduce buffer duration while mitigating and/or avoiding stuttering during the presentation.

[0085] Although the inventive concepts are described in language specific to structural features and/or methodological acts, it is to be understood that the inventive concepts described in the appended claims are not necessarily limited to the specific features or acts described. Rather, the specific features and acts are disclosed as exemplary forms of implementing the claimed inventive concepts.

Claims

1. A computer-readable medium having processor-executable instructions that, when executed by a processor, performs a method comprising:

   measuring a rate (measured rate) at which a multimedia stream is received at a downstream component configured to present the multimedia to a user; and,
   ascertaining a duration (measured duration) of time to buffer the received multimedia before presentation thereof, at least in part utilizing the measured rate.

2. A medium as recited in claim 1 further comprising prior to said measuring, receiving advisory data that provides a cumulative buffer amount for the presentation at an estimated acceleration rate.

3. A medium as recited in claim 2 further comprising, determining an estimated duration to buffer the received multimedia before presentation based, at least in part, on the estimated acceleration rate and the cumulative buffer amount.

4. A medium as recited in claim 1 further comprising prior to said measuring, receiving advisory data that provides a data array for one or more acceleration rates, and wherein for individual acceleration rates the data array contains a corresponding underflow byte value, a time in the presentation when the underflow byte value occurs, and an average data rate to a given point in the stream.

5. A medium as recited in claim 4 further comprising, determining an estimated duration to buffer the received multimedia before presentation for each of the acceleration rates of the data array.

6. A medium as recited in claim 5, wherein determining the measured duration comprises interpolating the measured rate between a higher data array acceleration rate and a lower data array acceleration rate.

7. A computing device comprising a medium as recited in claim 1.

8. A method facilitating fast start-up of a new multimedia stream associated with a source, the method comprising:

   requesting to receive a multimedia stream;

receiving advisory data relating to the multimedia stream;
beginning to receive and buffer the multimedia stream;
ascertaining a rate (measured rate) at which the multimedia stream is being received;
calculating a duration (measured duration) to buffer said multimedia stream based, at least in part, on said measured rate; and,
initiating presentation of the multimedia based on the measured duration.

9. A method as recited in claim 8, wherein said receiving comprises receiving advisory data that comprises an estimated buffer duration prior to beginning presentation of the multimedia for an estimated acceleration rate at which an upstream component intends to stream the multimedia.

10. A method as recited in claim 8, wherein said receiving comprises receiving advisory data that comprises a data array of acceleration rates, and wherein individual acceleration rates are associated with cumulative buffer amounts for the presentation, and wherein the data array further comprises an estimated acceleration rate at which an upstream component intends to stream the multimedia and an acceleration interval for which the upstream component intends to maintain the estimated acceleration rate.

11. A method as recited in claim 10, further comprising prior to the act of beginning, determining, at least in part from the advisory data, an estimated buffer duration prior to beginning to present the multimedia for the acceleration rates of the data array.

12. A method as recited in claim 8, wherein said act of receiving comprises receiving advisory data comprising a start-up profile relating to multiple acceleration rates and wherein said calculating comprises interpolating the measured rate between a first of the multiple acceleration rates having a higher value than the measured rate and a second of the multiple acceleration rates having a lower value than the measured rate and selecting the measured duration to satisfy both of the first and second acceleration rates.

13. A method as recited in claim 8, wherein said calculating comprises calculating a first measured duration and prior to reaching the first measured duration ascertaining a second subsequent measured rate and calculating a second measured duration, and wherein said initiating is based on said second measured duration.

14. A computing device configured to implement a method as recited in claim 8.

15. A computer implemented method comprising:

determining a duration to buffer a multimedia stream before beginning to present the multimedia; and,
subsequent to beginning to receive the multimedia stream, but prior to beginning to present the multimedia, verifying the duration.

16. A method as recited in claim 15, wherein the act of verifying comprises measuring a rate (measured rate) at which the multimedia is being received and comparing the measured rate to an estimated rate that the multimedia would be received.

17. A method as recited in claim 15, wherein the acts of determining and verifying are achieved by a single component.

18. A system comprising:

a sending unit configured to transmit multimedia; and,
a receiving unit configured to receive the multimedia and present the multimedia to a user, wherein the receiving unit is further configured to ascertain a measured rate at which the receiving unit is receiving the multimedia and to calculate, based at least in part on the measured rate, how long it should buffer the multimedia before starting to present the multimedia.

19. A system as recited in claim 18, wherein the sending unit comprises a server.

20. A system as recited in claim 18, wherein the sending unit is configured to access encoded multimedia and to transmit the encoded multimedia to the receiving unit.

21. A system as recited in claim 18, wherein the sending unit is configured to encode the multimedia prior to transmitting the encoded multimedia to the receiving unit.

22. A system as recited in claim 18, wherein the sending unit is configured to access the multimedia from a live encoding session.

23. A system as recited in claim 18, wherein the sending unit is configured to access the multimedia from a file.

24. A system as recited in claim 18, wherein the receiving unit comprises a personal computer.

25. A system as recited in claim 18, wherein the receiving unit comprises a set-top box.

26. A system as recited in claim 18, wherein the receiving unit comprises a cell phone.

27. A system as recited in claim 18, wherein the receiving unit is configured to decode the multimedia received from the sending unit.

28. A system as recited in claim 18, wherein the sending unit is further configured to send information regarding buffer values associated with presentation of the multimedia.

29. A system as recited in claim 18, wherein the sending unit is further configured to access header information associated with a presentation of the multimedia contained in a file and to send a summary of the information to the receiving unit.

30. A system as recited in claim 29, wherein the summary comprises at least one data array comprising at least one acceleration rate and a summarized buffer value associated with the presentation of the multimedia.

31. A system as recited in claim 29, wherein the summary comprises at least one data array comprising one or more of an acceleration rate, an associated buffer value associated with the presentation of the multimedia, or a time in the presentation associated with the buffer value.

32. A system as recited in claim 18, wherein the sending unit is further configured to send an estimated acceleration rate at which it intends to transmit the multimedia and a duration which it plans to transmit at the estimated acceleration rate.

33. A system as recited in claim 18 further comprising a transmission means coupling the sending unit and the receiving unit and wherein the transmission means may cause a transmission rate from the sending unit to be greater than the measured rate at the receiving unit.

34. A system comprising:

a downstream component configured to receive and buffer a multimedia stream and after buffering an amount of units of the multimedia to begin a presentation of the multimedia for a user; and,
an upstream component configured to access the multimedia and stream the multimedia to the downstream component, and wherein the upstream component is further configured to obtain at least one start-up profile relating to a recommended amount of buffer at one or more points in the presentation to mitigate an underflow condition.

35. A system as recited in claim 34, wherein the at least one start-up profile comprises multiple start-up profiles which relate to potential streaming rates of the multimedia stream.

36. A system as recited in claim 34, wherein the upstream component compiles the at least one start-up profile.

37. A system as recited in claim 34, wherein the upstream component calculates an estimated buffer time to buffer the amount of units, based at least in part on the at least one start-up profile.

38. A system as recited in claim 34, wherein the upstream component calculates an estimated buffer time to buffer the amount of units, based at least in part on the at least one start-up profile and wherein the upstream component passes the estimated buffer time to the downstream component which buffers according to the estimated buffer

time without further calculation or verification of the estimated buffer time.

39. A system as recited in claim 34, wherein the upstream component calculates an estimated buffer time to buffer the amount of units, based at least in part on the at least one start-up profile and wherein the upstream component passes the estimated buffer time to the downstream component which begins to buffer according to the estimated buffer time and prior to beginning the presentation the downstream component verifies the estimated buffer time.

40. A system as recited in claim 34, wherein the downstream component calculates an estimated buffer time to buffer the amount of units based at least in part on the at least one start-up profile.

41. A system as recited in claim 40, wherein the downstream component is configured to verify the estimated buffer time by calculating a second buffer time to buffer the amount of units based at least in part on a measured rate at which the downstream component is receiving the multimedia stream.

42. A system as recited in claim 41, wherein the downstream component is configured to delay beginning the presentation beyond the second buffer time due to properties of the multimedia stream.

43. A system as recited in claim 42, wherein the properties comprise one or more of send chunk size or audio/video separation.

Ex. 1018-203

Ex. 1018-204

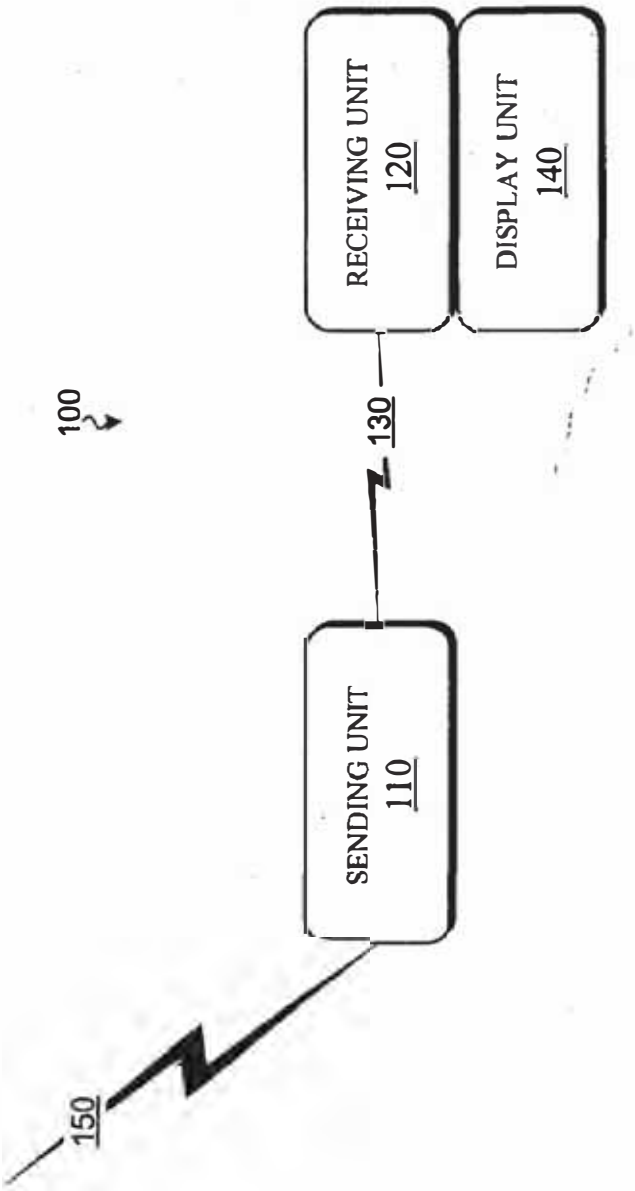


Fig. 1

Ex. 1018-205

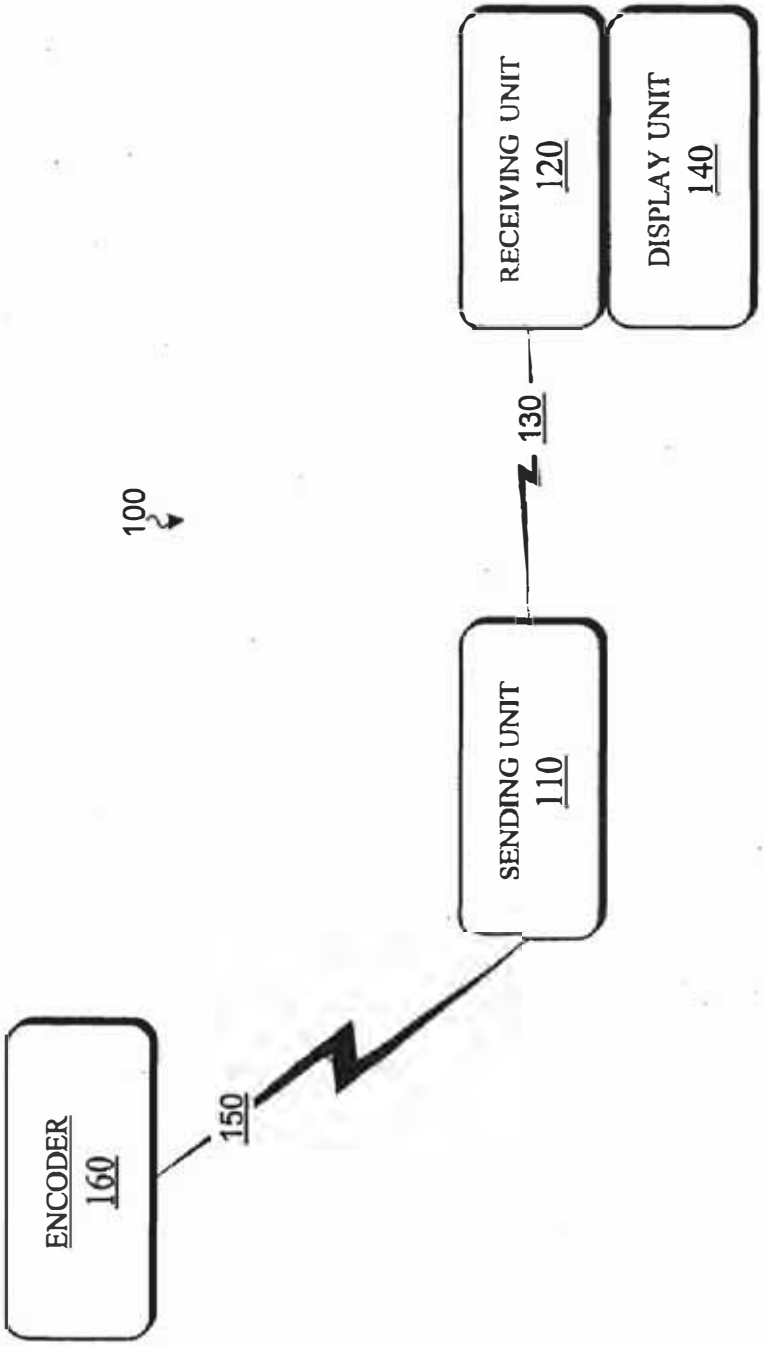


Fig. 1a

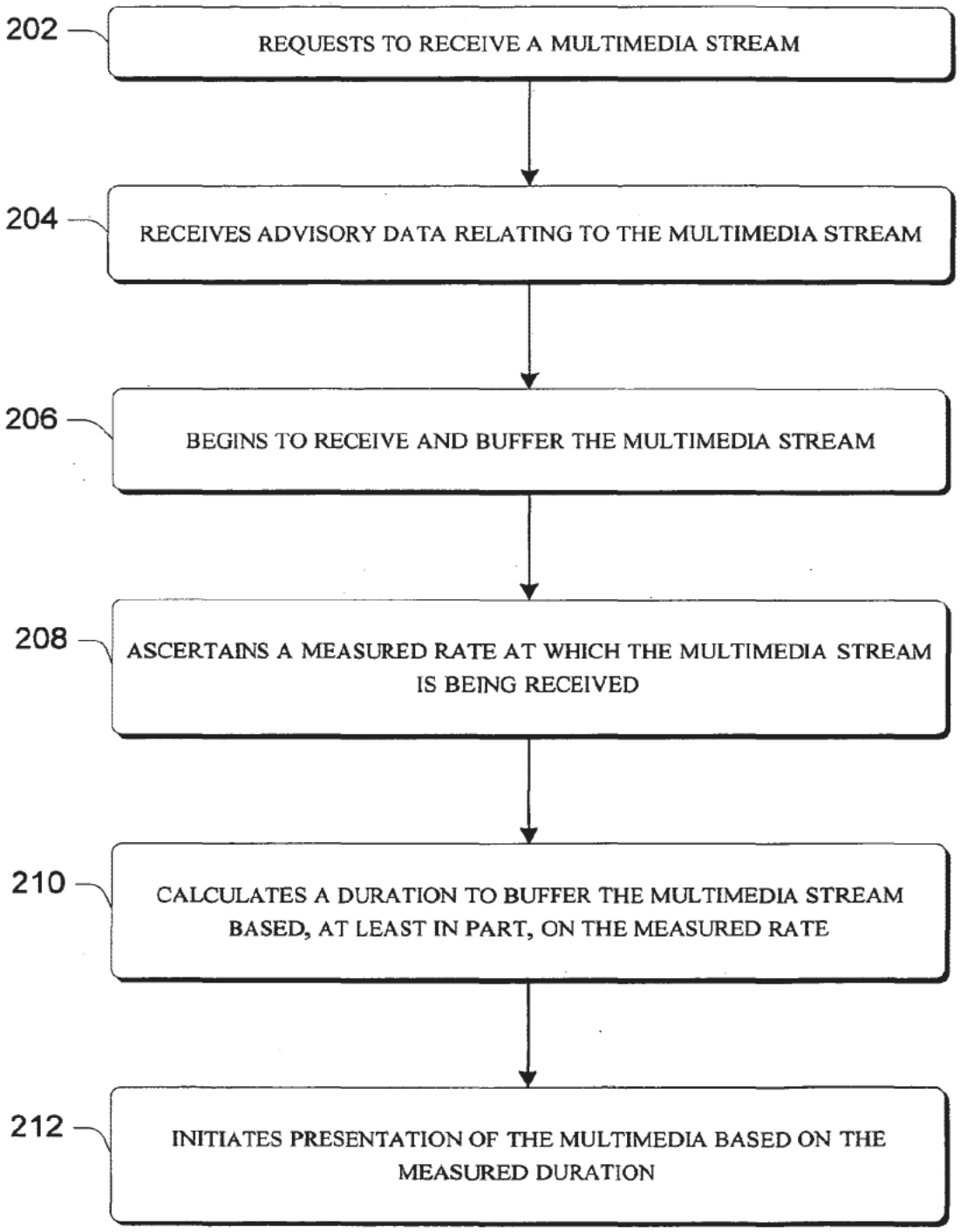

202
REQUESTS TO RECEIVE A MULTIMEDIA STREAM
204
RECEIVES ADVISORY DATA RELATING TO THE MULTIMEDIA STREAM
206
BEGINS TO RECEIVE AND BUFFER THE MULTIMEDIA STREAM
208
ASCERTAINS A MEASURED RATE AT WHICH THE MULTIMEDIA STREAM IS BEING RECEIVED
210
CALCULATES A DURATION TO BUFFER THE MULTIMEDIA STREAM BASED, AT LEAST IN PART, ON THE MEASURED RATE
212
INITIATES PRESENTATION OF THE MULTIMEDIA BASED ON THE MEASURED DURATION

Fig. 2

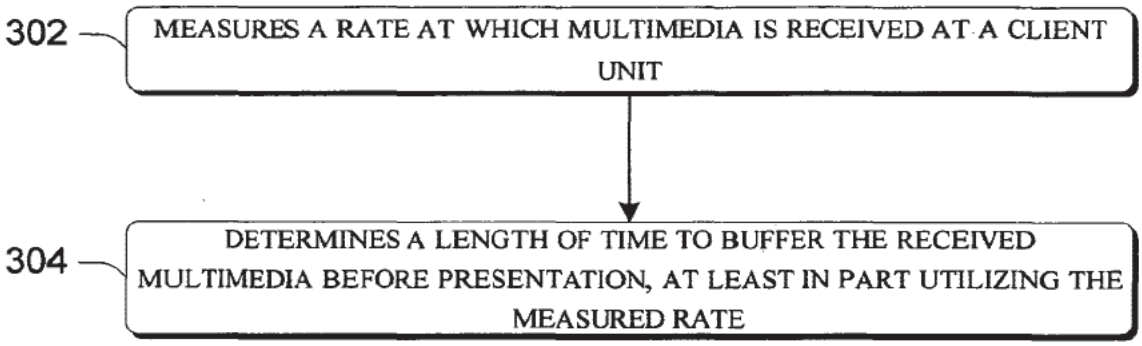


Fig. 3

Ex. 1018-207

(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) EP 0 711 077 A2

(12) EUROPEAN PATENT APPLICATION

(43) Date of publication:
08.05.1996 Bulletin 1996/19

(51) Int Cl.[6]: H04N 7/24

(21) Application number: 95307532.2

(22) Date of filing: 24.10.1995

(84) Designated Contracting States:
DE FR GB

(30) Priority: 01.11.1994 US 332984

(71) Applicant: AT&T Corp.
New York, NY 10013-2412 (US)

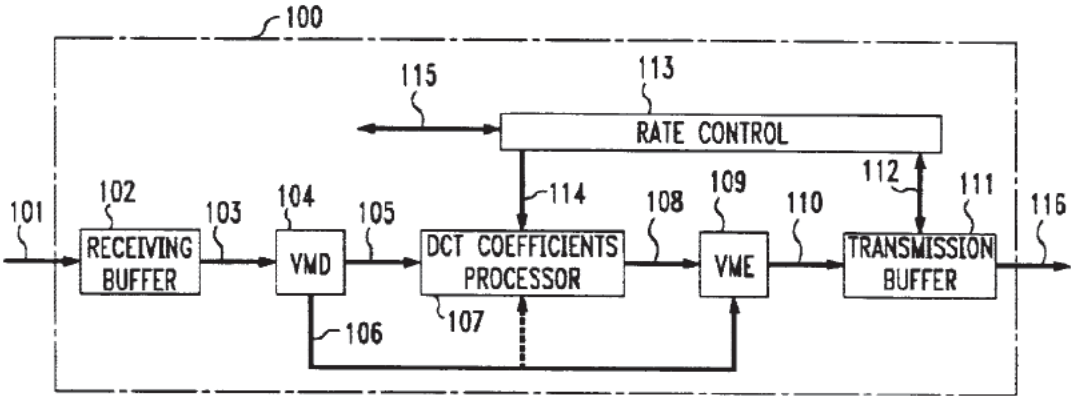
(72) Inventors:
- Haskell, Barin Geoffry
Tinton Falls, New Jersey 07724 (US)
- Yan, Li
Holmdel, New Jersey 07733 (US)

(74) Representative:
Buckley, Christopher Simon Thirsk et al
AT&T (UK) LTD.,
AT&T Intellectual Property Division,
5 Mornington Road
Woodford Green, Essex IG8 0TU (GB)

(54) Video transmission rate matching for multimedia communication systems

(57) Digital video transmission rate matching techniques are disclosed. A bit stream rate matching apparatus includes a bit rate increasing device and a bit rate reducing device (107). The bit rate increasing device converts a video bit stream having a first transmission rate to a video bit stream having a second transmission rate characterized in that the first transmission rate is less than the second transmission rate. The bit rate reducing device (107) converts a video bit stream (101) having the second transmission rate to a video bit stream having the first transmission rate. The bit stream rate matching apparatus is useful in the context of a multimedia conference where a first endpoint device employs the first transmission rate and a second endpoint device employs the second transmission rate.

FIG. 1



EP 0 711 077 A2

Printed by Jouve, 75001 PARIS (FR)

Description

Background of the Invention

1. Field of the Invention

The invention relates generally to multimedia communication systems which utilize endpoint devices, and more specifically to techniques for matching the video transmission rates of a plurality of endpoint devices.

2. Descriptionof thePriorArt

Video transmission rate matching is a technique which has been used to convert the bit rate of a first video bit stream to match that of a second video bit stream. This conversion may require bit rate reduction and/or bit rate increases. Among these two conversion processes, bit rate reduction is more critical, due to the fact that this process involves the removal of bits. Since these bits represent video information, the quality of a video signal having a reduced bit rate may be degraded. The amount of degradation is related to the manner in which the bit rate reduction is achieved.

With respect to bit rate increases, no information is removed from the video bit stream. Instead, such a rate increase requires the incorporation of additional bits into the bit stream. Although it would be desirable to add bits to the bit stream for the purpose of enhancing the video signal, such enhancements are often in the pel domain. For example, if an incoming video signal has been subjected to a process of data compression, the information discarded in the compression process is not stored. The information is lost, and cannot be recovered and added to the bit stream during the process of increasing the bit rate. Therefore, bit rate increases are implemented by adding "extra" bits to the bit stream, wherein these "extra" bits are not the same bits that may have been removed during a previous data compression step. Bits are added using conventional bit- stuffing patterns to occupy the additional bandwidth of a rate-increased video signal.

Video transmission rate matching has been applied to a wide variety of applications in the field of multimedia communication, especially in environments involving multipoint communication topologies. For example, in multimedia conferencing applications, a "bridge" or "multipoint control unit" (MCU) is typically employed. This MCU is a computer-controlled device which includes a multiplicity of communication ports. The ports may be selectively interconnected in a variety of ways to provide communication among a group of endpoint devices. MCUs are often used to establish multi-party conferences among a plurality of endpoint devices, and to establish multi-party conference calls. A state-of-the-art MCU is described in ITU Document H.243, "Procedures for Establishing Communication Between Three or More Audiovisual Terminals Using Digital Channels up to 2 Mbps", March 1993, and in ITU Document H. 231, "Multipoint Control Units for Audiovisual systems Using Digital Channels up to 2 Mbps", March 1993.

Existing MCUs require all endpoint devices participating in a given multimedia conference to use the same video transmission rate. Typically, during the initial setting up of a multimedia conference, or at the time that it is desired to add an additional endpoint device to an existing conference, the MCU polls all of the endpoint devices to ascertain the video transmission rate or rates each endpoint device is equipped to implement. When the MCU polls a given endpoint device, the endpoint device transmits a capability code back to the MCU. The capability code includes information specifying the video bit rate transmission capabilities, audio bit rate transmission capabilities, etc., of a given endpoint device. These capability codes may also specify the compression algorithm or algorithms used by a given endpoint device.

Based upon the capability codes received by the MCU from the various endpoint devices, the MCU selects a minimum video bit rate to be used as the video transfer rate for the multimedia conference. The MCU sends a video rate signal to all endpoint devices which specifies use of this video transfer rate. Upon receipt of the video rate signal, the endpoint devices all operate using the same video transfer rate.

Use of the same video transfer rate for all endpoint devices participating in a given conference presents significant shortcomings. For example, in many cases, a conference participant may desire to receive high-quality video by using a relatively high video transfer rate, whereas other participants may want to use less expensive equipment offering a lower video transfer rate. State-of-the-art MCUs cannot implement a multimedia conference using a plurality of different video transfer rates. Therefore, it would be desirable to equip existing MCUs with a video transmission rate matching device. Through the use of video transmission rate matching techniques, video communication among a plurality of endpoint devices will be provided, even when the endpoint devices use different video transmission rates.

One video transmission rate matching method well-known to those skilled in the art is termed the transcoding method. Pursuant to the transcoding method, a compressed video bit stream having a first bit rate is fully decoded into a video space known as the pel domain. This fully-decoded bit stream, which may be conceptualized as a completely reconstructed video sequence, is then encoded into a video bit stream having a second bit rate.

The existing transcoding method is disadvantageous. Since decoding as well as encoding processes are required, transcoding is very time-consuming. As a practical matter, the time delay is at least twice that of the end-to-end encoding delay. Such a delay is not tolerable for applications requiring real-time communication, such as multimedia conferencing. A faster method

Ex. 1018-209

of performing video bit rate matching is needed.

Summary of the Invention

Digital video transmission rate matching techniques are disclosed. A bit stream rate matching apparatus includes a bit rate increasing device, a bit rate reducing device, and first and second switching devices. The first switching device switches each of a plurality of incoming video bit streams to any one of the bit rate increasing device, the bit rate reducing device, and the second switching device. Each of these incoming video bit streams originates from a particular endpoint device. The second switching device switches video bit streams from the bit rate reducing device, the bit rate increasing device, and the first switching device to each of a plurality of outgoing video bit streams. The first and second switching devices are controlled by a processor. The bit stream rate matching apparatus is useful in the context of a video communication where a first endpoint device employs the first transmission rate and a second endpoint device employs the second transmission rate.

Brief Description of the Drawings

FIG. 1 is a hardware block diagram of a video transmission rate reduction apparatus according to a preferred embodiment disclosed herein;

FIGs. 2 and 3 are data structure diagrams setting forth illustrative coding formats for representing video information;

FIG. 4 is a data structure diagram illustrating the arrangement of DCT coefficients within one of the blocks shown in FIG. 3;

FIG. 5 is a data structure diagram setting forth an illustrative example of DCT coefficient partitioning;

FIG. 6 is a software flowchart setting forth a procedure for performing video bit rate matching according to a preferred embodiment disclosed herein;

FIG. 7 is a hardware block diagram setting forth an illustrative structure for the discrete cosine transformation (DCT) processor of FIG. 1;

FIG. 8 is a hardware block diagram setting forth the system architecture of a multimedia conferencing system which includes an MCU having a video processor equipped to perform video transmission rate matching methods according to various preferred embodiments disclosed herein;

FIG. 9 is a block diagram showing an illustrative hardware configuration for the video processor of FIG. 8; and

FIG. 10 is a hardware block diagram setting forth a bit rate matching system for use with five endpoint devices.

Detailed Description

For illustrative purposes, video transmission rate matching will be described in the operational context of an H.261 environment. However, it is to be understood that the video transmission rate matching techniques described herein are generally applicable to any video compression algorithm which uses transformation and quantization processes. For example, the techniques disclosed herein may be employed in conjunction with video telephones of the type described in CCITT recommendation COST 211. However, for purposes of the present disclosure, video data to and/or from the video transmission rate matching system will be compressed in an H.261-like format.

An "H.261-like" format is any coding format which is similar to the coding format currently being established by the International Telecommunications Union (ITU-T). The format is described in the ITU-T document "Recommendation H.261, Video Codec for Audiovisual Services at px64 kbits/s", May 1992, and the ITU-T document "Description of Reference Model 8", June 9, 1989.

FIG. 1 is a hardware block diagram showing a transmission rate reduction system 100 for a video signal which has been compressed in an H.261-like format. The transmission rate reduction system 100 includes an input port 101 for accepting a compressed video signal having a first bit transmission rate and an output port 116 for providing a compressed video signal having a second bit transmission rate. A compressed video signal is defined as the binary representation of a video signal which has been compressed by a coding algorithm substantially similar to that described in the H.261 standard, and then coded according to a syntax substantially similar to that described in the H.261 standard.

A compressed video signal in the form of a digital, coded bit stream is presented to input port 101. This compressed video signal includes coded digitized video information along with a header. The header may include the types of information specified in the H.261 standard, and/or other types of information such as the source, the destination, the content, and/or the organization of the video information.

Input port 101 is connected to a receiving buffer 102, which is a conventional digital buffer. This receiving buffer 102 provides electrical isolation between the source of the compressed video signal and the various circuit elements shown in FIG. 1.

The output of the receiving buffer 102 is coupled to a video multiplex decoder (VMD) 104. VMD 104 includes the combination of a decoder and a demultiplexer. The demultiplexer is equipped to demultiplex the coded bit stream. The decoder decodes header information which has been coded into the aforementioned coded bit stream. The demultiplexer and decoder perform the function of recovering compressed video data from the coded bit stream.

The VMD 104 includes a first output port 105 and a second output port 106. The VMD 104 provides the first output port 105 with quantized DCT coefficients along

Ex. 1018-210

with quantization information. The nature of these DCT coefficients is described in greater detail in the H.261 standard. The second output port 106 is provided with motion vectors. First output port 105 is coupled to DCT coefficients processor 107, and second output port 106 is coupled to a video multiplex encoder (VME) 109. The operation and structure of VME 109 will be described in greater detail below. The motion vectors are sent directly from VMD 104 to VME 109 because no motion estimation is employed in the present example.

The DCT coefficients received from the first output 105 of VMD 104 are sent to DCT coefficients processor 107. The DCT coefficients processor 107 processes the DCT coefficients in such a manner that the output 108 of DCT processor 107 is a signal which requires fewer bits to encode than the number of bits which were received from the first output 105 of the VMD 104. The DCT coefficients processor 107 reduces the number of bits such that the resulting video quality is not substantially degraded as perceived by human visual processes. In other words, the video quality is degraded "gracefully". The amount of information reduced by processing coefficients at DCT coefficients processor 107 is controlled by the rate-control signal 114 sent by rate control unit 113. Various techniques for programming DCT coefficients processor 107 to degrade video quality gracefully are well known to those skilled in the art. Three such techniques for gracefully degrading video quality will be discussed in greater detail hereinafter.

The processed DCT coefficients are produced at the output 108 of DCT coefficients processor 107. These processed DCT coefficients, together with the motion vectors 106 produced at the second output of VMD 104, are sent to the video multiplexing encoder (VME) 109 to form a new video bit stream substantially conforming to the H.261 standard. The VME 109 encodes the processed DCT coefficients and motion vectors, and multiplexes them into the layered data structures shown in FIG. 3, to be described in greater detail hereinafter. The new coded bit stream, produced at the output 110 of VME 109, is sent to a transmission buffer 111.

As it is well-known that compressed video data signals may include a plurality of components, wherein each component may be represented by a different number of bits, transmission buffer 111 performs a variable-rate to constant-rate translation for the compressed video. The transmission buffer 111 includes circuitry to ascertain and to indicate the status of the transmission buffer 111, which is defined as the occupancy ratio of the memory locations within the transmission buffer 111. The occupancy ratio refers to the ratio between the number of occupied memory locations within a given buffer and the total number of memory locations within this buffer. The buffer status is produced at a first input/output 112 of the transmission buffer 111. This first output 112 is coupled to a rate control 113 circuit. This rate control 113 circuit adjusts the average data rate provided by the DCT coefficients processor 107.

Rate control 113 circuit and DCT coefficients processor 107 are coupled together via signal line 114. The rate control 113 circuit includes a first communications line 115 adapted for receiving a signal, (for example, from a control processor) which specifies a desired output bit rate for transmission buffer 111. The communications line 115 is also adapted to transmit signals to control processor 840 (FIG. 9). The desired output rate signal is processed by rate control circuit 113 in conjunction with the buffer status signal received at the first input/output 112, to generate a rate control output signal which is downloaded via signal line 114 to DCT coefficients processor 107. Based upon the desired output rate signal and the buffer status signal, the rate control 113 circuitry computes the total number of bits for each frame, as well as the bits targeted for each macro block. The targeted bits per macro block or bits per frame are used as reference to produce a proper control signal, which is applied to a second input 114 of the DCT coefficients processor 107. The function of the second input 114 is to force the DCT coefficients processor 107 to operate in such a manner that the bits produced for each macro block are as close to the targeted bits per macro block as possible. The type of signal applied to second input 114 is dependent upon the specific type of DCT coefficients processor 107 which is utilized.

DCT coefficients processor 107 is the processing unit where the DCT coefficients recovered from the VMD 104 are further processed in order to match a desired (i.e., target) output video rate. Three methods can be used to process the DCT coefficients to reduce the total number of bits. Each of these methods provides for the graceful degradation of video quality, as was discussed above. The first method is termed DCT coefficients zeroing, the second method is called the requantization of the DCT coefficients, and the third method consists of the combination of the first and second methods. In the first method, DCT coefficients are partitioned into groups based upon the relative importance of the various coefficents. Due to the fact that DCT coefficients are generally organized into two-dimensional arrays wherein the array entries which are relatively close to the upper left-hand corner of the array include relatively low-frequency components, as compared with array entries which are relatively close to the lower right-hand corner of the array, the relative importance of various DCT coefficients is known. The lower frequency components are more important and the higher frequency components are less important. Based upon the output produced by rate control 113 circuit on signal line 114, the coefficients of the least important group are set to zeroes. Here, the control signal on signal line 114 consists of a digital representation of the indices of a plurality of specific importance groups, or simply indices of the DCT coefficients within a macro block, whose coefficients will subsequently be set to zeroes. By forcing some DCT coefficients to zero, the amount of data pro-

Ex. 1018-211

duced by the DCT coefficients processor 107 can be properly controlled by rate control 113 circuit.

FIGs. 2 and 3 are data structure diagrams setting forth illustrative coding formats for representing video information in accordance with the H.261 standard. Referring now to FIG. 2, video information consists of a plurality of frames 201, 203, 205, 207, 209, 211, 213, 215, 217, 219, 221. Each of these frames contains a a representation of a two-dimensional video image in the form of a pixel array. Since a given frame may represent a video image at a specific moment in time, a plurality of frames may be employed to represent a moving image. Each frame represents a picture. A plurality of frames represents a coded video sequence.

Each of the frames is compressed according to any one of two types of compression algorithms, termed intra-frame coding (I) and predictive coding (P). For example, frames 201 and 211 are compressed using intra-frame coding (I), and frames 203, 205, 207, 209, 213, 215, 217, 219, and 221 are compressed using predictive coding. The sequence of frames shown in FIG. 2 establish a data structure for representing a video image in the form of an encoded video sequence having a plurality of levels arranged in a two-dimensional array, wherein each level represents the value of a pixel element. This encoded video sequence may be termed a coded video bit stream.

If intra-frame coding (I) is to be applied to a given frame, such as frame 201, the frame is termed an I-designated frame, and if predictive coding (P) is to be applied to a given frame, such as frame 205, the frame is termed a P-designated frame.

Pursuant to intra-frame coding (I) compression processes, the I-designated frame 201 is divided into a plurality of pixel blocks, wherein each block consists of an array of 8 x 8 pixels. Next, a discrete cosine transform (hereinafter, DCT), is performed on each of the pixels in the 8x8 pixel block, in accordance with procedures well-known to those skilled in the art, to generate a plurality of DCT coefficients. Thereafter, quantization is performed on the DCT coefficients, in accordance with well-known quantization procedures. These quantized DCT coefficients constitute compressed video image information for the I-encoded frame 201.

Predictive coding (P) is implemented on a P-designated frame, such as frame 205, by: 1) partitioning the P- designated frame into a plurality of macro blocks. For example, if the frame includes a plurality of pixel arrays, each having 16 x 16 pixels (FIG. 2, 251, 252, 253, 254, 257, 258), each 16 x 16 pixel array may be partitioned into 4 contiguous blocks, wherein each block is an 8 x 8 pixel array; a 16 x 16 pixel array (luminance) together with an 8 x 8 pixel block (chrominance) and an 8 x 8 pixel block (chrominance), comprises a macro block 247; 2) for each of the macro blocks (16 x 16 pixels) created in step (1), searching the most recent previously occurring frame (which could be either a P- or an I-designated frame, but in the present example is frame 203) for the macro block which contains image information that is most similar to the image information in the macro block created in step (1); 3) generating motion vectors to spatially translate the macro block found in the prior I or P frame in step (2) to the location of the similar macro block in the P frame presently being compressed; 4) generating a predicted frame from the most recent previously occurring frame using the motion vectors; 5) on a macro-block-by-macro-block basis, subtracting the predicted frame from the P-frame being compressed, to generate blocks of residues; 6) performing DCT's on the blocks of residues; 7) quantizing the coefficients of the blocks of transformed residues; and 8) concatenating the quantized residue coefficients and the motion vectors to form a compressed video signal.

In an intra-frame coded (I) picture, every macro block is intra-coded. That is, each macro block is coded without referring to any macro block in the previous I-or P-frame. In the predictive-coded (P) picture, the macro block can be either intra-coded or inter-coded.

To form the coded video bit stream for transmission, the compressed image information, as well as other information such as motion vectors, are coded using specified code words. The code words are then multiplexed into a layered data structure to form the final bit stream. In an H.261-like environment, the coded bit stream is organized into a hierarchical format, the structure of which is illustrated in FIG. 3.

Referring to FIG. 2, the sequence of frames 201, 203, 205, 207, 209, 211, 213, 215, 217, 219, 221 forms a coded video bit stream. This bit stream may be conceptualized as a serial representation of coded frames which can be processed to form a moving video image (i.e., a moving picture). A typical sequence of frames is IPPP..PIPPP, where I indicates an intra-coded frame, and P designates a predictive-coded frame. For each frame 221, the coded bit stream representing the frame includes a header 263 and coded data 265. Each header 263 includes a start code and data related to the respective frame (i.e., picture). In an H.261 system environment, much of the header information is required for synchronization purposes. For example, at the frame (picture) layer for frame 221, header 263 includes a picture start code (PCS) field 267, a picture number (TR) field 269, a picture type (PTYPE) field 271, a PEI field 273, and a PSPARE field 274. The PEI field 273 and the PSPARE field 274 are adapted to accommodate extra information which may be required for future applications.

Picture data is segmented into Groups of Blocks (GOB) 223, 225, 227, 229, 231, 233, 235, 237, 239, 241, 243, and 245. A GOB (for example, GOB 229) comprises one-twelfth of the coded frame (CIF) 221. Therefore, GOB 229 may be conceptualized as including one-third of one quarter of a coded picture area. The area represented by one-quarter of a coded picture is abbreviated as QCIF. Accordingly, there are 12 GOBs 223, 225, 227, 229, 231, 233, 235, 237, 239, 241, 243, 245 in a CIF

Ex. 1018-212

frame 221, and three GOBS in a QCIF frame. The arrangements of GOBs in a CIF/QCIF picture are depicted in FIGs. 2 and 3.

Each GOB 229 includes a header field 291, followed by a macro block data field 298. The header field 291 includes a GOB start code (GBSC) field 292, a group number (GN) field 293, a group type (GTYPE) field 294, a GOB quantizer (GQUANT) field 295, and spare information fields in the form of GEI field 296 and GSPARE field 297. Each GOB 229 consists of 33 macro blocks, such as "macro block 24" (reference numeral 247) and "macro block 25" (reference numeral 249). The arrangement of macro blocks within a GOB is depicted in FIG. 2.

Each macro block includes a header field 275 followed by a block data field 277. The header field 275 includes a macro block address (MBA) field 279, a block type information (MTYPE) field 281, a quantizer type (MQUANT) field 283, a motion vector (MVD) field 285, and a coded block pattern (CBP) field 287. The block data field 277 of each macro block 247 consists of 6 blocks, including four luminance blocks Y1 (reference numeral 251), Y2 (reference numeral 252), Y3 (reference numeral 253), Y4 (reference numeral 254), one chrominance block U (reference numeral 257), and one chrominance block V (reference numeral 259). An illustrative example of the contents of luminance block U (reference numeral 257) is set forth in FIG. 2. Note that this block includes an 8 x 8 pixel array wherein all pixels have a luminance value of black.

A block represents a matrix (array) of pixels, e. g., 8 x 8, over which a discrete cosine transform (DCT) is performed. The array of pixels is represented by a matrix of pixel array coefficients, including a DC coefficient and a plurality of AC coefficients. The transformed coefficients (TCOEFF) 301 (FIG. 3) consists of a DCT DC coefficient occurring first, followed by respective pixel array coefficients (AC), in the order of their relative importance. The arrangement of DC and AC coefficients in an illustrative block data field 277 (FIG. 3) is shown in FIG. 4. The block data field 277 (FIG. 3) consists of the transformed coefficients (TCOEFF) 301 and an end of block code (EOB) 303 which is appended at the end of each successively occurring block of data.

A typical partitioning of DCT coefficients is illustrated in FIG. 5. The DCT coefficients are arranged in a two-dimensional array 500 stored in block data field 277 (FIG. 3). The two-dimensional array 500 (FIG. 5) includes eight rows and eight columns. Each entry in the array corresponds to a specific entry group, such as Group 506, Group 507, or Group 508. The groups are based upon the relative importance of the entries contained therein. Each group includes entries conforming to a specific range of importance levels. These importance levels relate to the relative extent to which the elimination of a particular entry would degrade the quality of the overall video image in a given frame. In the example of FIG. 5. Group 506 is the most important group, and includes entries having a relatively high level of importance. Group 507 includes entries having an intermediate level of importance, and Group 508 includes entries having the least importance to the overall quality of the video image.

FIG. 6 is a software flowchart setting forth a procedure for performing video bit rate matching according to a preferred embodiment disclosed herein. The operations set forth in the flowchart may be implemented using the hardware previously described in connection with FIG. 1, wherein the functioning of rate control 113 circuit was disclosed. As shown in FIG. 6, the video bit rate matching procedure consists of six steps. The first step is initialization 600, followed by macro block processing 601, macro block parameter updating 602, and macro block counting 603. At block 603, a test is performed to ascertain whether or not the number of macro blocks that have already been processed are greater than a maximum number of macro blocks. If not, the program loops back to block 601. If so, the program progresses to block 604, where frame parameters are updated. Next, at block 605, frames are counted, and a test is performed to ascertain whether or not there are additional frames to consider. If so, the program loops back to block 601. If not, the program ends.

Referring to block 600, in the initialization stage, the rate control unit 113 (FIG. 1) performs the following series of operations:

1. Obtaining a value for the desired video output bit rate. This value may be received, for example, from one or more endpoint devices;
2. Specifying the maximum output frame rate based upon the desired video output bit rate;
3. Sending the maximum output frame rate to one or more endpoint devices; i.e., sender endpoint devices--which are to be used to send video information to other endpoint devices. Once the sender endpoint device receives the maximum output frame rate, this endpoint device is forced to encode video signals with the specified maximum frame rate.
4. Computing the average number of bits per frame and stores this number in a memory location designated as average_bits_per_frame. If the desired video output bit rate is R_{out} and the maximum frame rate is F_{out}, then the average_bits_per_frame= $\frac{R_{out}}{F_{out}}$.
5. Initializing the transmission buffer with an initial buffer fullness specified as B_0.
6. Specifying the targeted_bits_per_frame for the 1st frame. targeted_bits_per_frame = $K*\frac{R_{out}}{F_{out}}$ where K is a constant which is chosen based on the maximum frame rate and the initial buffer fullness B_0. Then the targeted_bits_per_mblk is targeted_

Ex. 1018-213

$$\text{bits_per_mblk} = \frac{\textit{targeted_bit_per_frame}}{\textit{total_number_of_mblk}}$$

7. According to the targeted_bits_per_mblk, specifying the particular group indices which are to be set to zero.

At the macro block processing step (FIG. 6, block 601), the DCT coefficients processor 107 (FIG. 1) performs the following steps:

1. Receiving a macro block from the VMD 104 (FIG. 1);
2. Receiving a control signal over signal line 114 from rate control 113 circuit in the control signal specifies one or more DCT coefficients to be set to zero; and
3. Setting the DCT coefficients specified by the control signal to zero.

After processing one macro block, the rate control 113 circuit receives the new buffer status from transmission buffer 111 via first input/output 112. The rate control 113 circuit uses the buffer status, defined above, to update the control signal. Since the transmission buffer 111 is used as a temporary storage facility for video information, the buffer may operate in a first state, where the buffer is in the process of receiving new video information from input buffer 110, or the buffer may operate in a second state, where the buffer has already stored incoming video information and is adapted to output this information to buffer output 116. Therefore, the status of the buffer refers to the state in which the buffer is operating at a given moment.

The steps implemented by rate control circuit 113 for updating the control signal include:

1. Obtaining the total number of bits used for the mblk, bits_per_mblk;
2. Computing the difference between the targeted_bits_per_mblk and the actual bits_per_mblk.
bits_difference += targeted_bits_per_mblk-bits_per_mblk.
3. Updating the control signal on signal line 114 based on the following:
If difference > 0 , reduce the number of indices to be set to zero.
else if difference < 0, increase the number of indices to be set to zero.
else no change.

At the end of processing each macro block, the macro block counter is checked against the total number of mblk to ascertain whether or not a frame is finished. If a frame is finished, rate control 113 circuit starts updating the frame parameters. At block 604, the rate control 113 circuit performs the following series of operations:

1. Obtaining the transmission buffer 111 status;
2. Obtaining the total number of bits used in the frame;
3 Based on the targeted memory location occupancy rate for transmission buffer 111 (i.e., buffer fullness), computing the targeted bits for the next frame and the targeted bits for each macro block;
4 Based on the targeted bits for each macro block, providing an appropriate control signal for the 1st macro block of the next frame.

After frame parameter updating, the new frame is checked. If there are no more bits, then the procedure ends. Otherwise, the procedure reverts back to processing macro blocks.

A second method of video bit rate matching is the requantization of DCT coefficients. The output signal at the first output 105 of VMD 104 includes two components: quantized DCT coefficients, and a quantization parameter. In order to determine values for the DCT coefficients, an inverse quantization operation is performed on the quantized DCT coefficients as follows. Let $\{x_i, i=0,1,2,..63\}$ be the quantized DCT coefficients and $\{y_i, i=0, 1,...63\}$ be the reconstructed DCT coefficients, with Qp representing the quantization parameter. Then, with respect to an H.261-like environment, in the I-coding mode, the reconstructed DC coefficient y_0 is calculated using the relationship

$$y_0 = x_0 * 8,$$

and the remaining coefficients are calculated using the formula

$$y_i = [x_i * 2 + \sin(x_i)] * Qp.$$

where $\{i= 1, 2, ...63\}$ in I mode and $\{i=0, 1, ...63\}$ in P mode, and the sign(w) function is defined as follows:

$$\text{sign}(w) = \begin{cases} 1 & \text{if } w \geq 0 \\ -1 & \text{if } w < 0 \end{cases}$$

To control the amount of data produced by the DCT coefficients processor 107 (FIG. 1), the rate-control unit computes the proper quantization parameter Qp_{new} based on the targeted bits per macro block and sends it to the DCT coefficients processor 107 to requantize the DCT coefficients. Let $\{z_i, i=0,1,..63\}$ be the new quantized DCT coefficients, and Qp_{new} be the new quantization parameter obtained from the rate control 113 circuit. Then, the new quantized DCT coefficients are determined by

$$z_0 = (y_0+4)/8,$$

where z_0 is the DC coefficient of the I-coded macro block. The rest of the coefficients are obtained by

$$z_i = y_i(2*Qp_{new})$$

where $\{i=1, ...63\}$ for the intra-coded macro block, and $\{i=0,1,...,63\}$ for inter-coded macro blocks.

With respect to the second method of matching vid-

Ex. 1018-214

eo signal bit rates, the sequence of operations performed by the hardware configuration of FIG. 1 is virtually identical to the process set forth in FIG. 6, with the following exceptions. Referring back to FIG. 6, at the initialization stage (block 600), the rate control 113 circuit performs the following steps:

1. Obtaining the new (desired and/or target) video output bit rate;
2. According to the new video output bit rate, specifying the maximum output frame rate;
3. Sending the maximum frame rate via communications line 115 to the sender endpoint device (defined above) to force this endpoint to encode the video with the maximum frame rate.
4. Computing the average number of bits per frame as average _bits_per _frame. Let new video bit rate be R_{out} and the maximum frame rate be F_{out}, then the average_bits_per_frame$=\frac{R_{out}}{F_{out}}$.
5. Initializing the transmission buffer with an initial buffer memory occupancy rate (fullness) of B_0.
6. Specifying the targeted_bits_per_frame for the 1st frame.

$$\text{targeted_bits_per_frame} = K^* \frac{R_{out}}{F_{out}}$$

where K is a constant which is chosen based on the maximum frame rate and the initial buffer fullness B_0. Then the targeted_bits_per_mblk is

$$\text{targeted_bits_per_mblk} = \frac{\textit{targeted_bits_per_frame}}{\textit{total_number_of_mblk}}$$

7. According to the targeted_bits_per_mblk, specify the new quantization parameter Qp_{new}.

At the macro block processing step (block 601), the DCT coefficients processor 107 performs the following steps:

1. Obtaining an mblk from the VMD 104;
2. Performing inverse quantization based on the Qp and recovering the DCT coefficients;
3. Obtaining the control signal on signal line 114 from rate control 113 circuit;
4. Using the control signal on signal line 114, requantizing the DCT coefficients.

After the processing of one macro block has been completed, the rate control 113 circuit 113 obtains the new (current) transmission buffer 111 status and updates the control signal on signal line 114. The steps implemented by rate control 113 circuit include:

1. Obtaining the total number of bits used for the macro block, bits_per_mblk;
2. Computing the difference between the targeted_bits_per_mblk and the actual bits_per_mblk:

bits_difference+=targeted_bits_per_mblk-bits_per_mblk.

3. Updating the control signal 114 based on the following:

If difference > 0 , reduce the size of the quantization parameter; else if difference < 0, increase the size of the quantization parameter; else no change.

At the end of processing each macro block, a macro block counter which counts the number of macro blocks which have been processed, is checked against the total number of macro blocks to ascertain whether or not a frame has been completed. If a frame has been completed, rate control 113 circuit commences updating the frame parameters. At block 604, the rate control 113 circuit performs the following steps:

1. Obtaining the transmission buffer 111 status;
2. Obtaining the total bits used by the frame;
3. Based upon the targeted buffer fullness (memory location occupancy rate), computing the targeted bits for the next frame and the targeted bits for each macro block;
4. Based on the targeted bits for each macro block, generating an appropriate control signal for the first macro block of the next frame.

After frame parameter updating, the new frame is checked. If there are no more bits, then the procedure ends. Otherwise, the procedure reverts back to the macro block processing step at block 601.

A third method of video bit rate matching may be employed in conjunction with a preferred embodiment disclosed herein. This third method includes all methods which represent combinations of various features of the first and second methods. The manner in which the first and second methods are combined is determined by the specific applications of a given system. One illustrative combination of the first and second methods is the process of using DCT coefficient partitioning to process intra-coded macro blocks, and then employing requantization to process the inter-coded macro blocks.

Although system 100 (FIG. 1) with the DCT processor equipped for implementing the three different processing schemes described above is satisfactory for lower rate reduction and intra-coded frames, there is a mismatch, "drift" between an endpoint device that transmits video information at a fast rate relative to other endpoint devices which decode this video information at a slower rate. This mismatch is brought about because the video encoder is required to operate at a faster bit rate than the video decoder. This mismatch exists for all the inter-coded frames and is likely to accumulate with time, unless an intra-coded frame is periodically inserted into the video bit stream. To control the accumulation of the mismatch, an improved DCT processor with the

Ex. 1018-215

mismatch correction elements is shown in FIG. 7.

FIG. 7 is a hardware block diagram setting forth an illustrative structure for the discrete cosine transformation (DCT) processor of FIG. 1. The hardware configuration of FIG. 7 represents an improvement over the DCT coefficient processor disclosed before in connection with FIG. 1, as well as other existing state of the art systems, such as the systems described in an ITU-T document entitled, "Low Bitrate Coding (LBC) for Videophone", document no. LBC-94-166. This document describes methods for reducing the bit rate of compressed video information with a minimal amount of associated processing delays. One implementation described in the ITU document utilizes one motion-compensated prediction storage device and two transform operations: a forward transform operation, and an inverse transform operation. The main purpose of this implementation is to correct the "drift", i.e., the mismatch, between a video encoder and a video decoder.

According to a preferred embodiment disclosed herein, the two transform operations described in the preceding paragraph are no longer required. Rather, motion compensation is performed in the transform domain, as the terms "motion compensation" and "transfer domain" are generally understood by those skilled in the art. With reference to FIG. 7, one feature of this embodiment is that the drift error signal stored in a picture memory of a prediction frame storage device 703 need not be stored with full accuracy. In particular, only a small number of the lower-frequency components of the transform coefficients need to be retained in the picture memory. Since only a relatively small number of coefficients are now involved in the motion compensation process, and the transform operations are no longer needed, implementation of the embodiments disclosed herein is simplified considerably over the system described in the above-referenced ITU-T document identified as no. LBC-94-166.

The simplified system disclosed herein is described below with reference to FIG. 7. An improved DCT (discrete cosine transformation) processor 107 is shown, which includes an inverse quantizer 701, a quantizer 702, and a prediction frame storage device 703. The inverse quantizer 701 accepts DCT coefficients and quantization parameters of an input bit stream from the first output 105 of VMD 104 (FIG. 1). The output of inverse quantizer 701 which is the reconstructed DCT coefficient, is coupled to a first input of a summer 704, and this output is also coupled to a first input of a subtractor 706. The output of summer 704 is fed to a first input of quantizer 702. A second input of quantizer 702 is under control of the signal line 114 which is coupled to rate control 113 circuit (FIG. 1).

The output of quantizer 702 (FIG. 7) which is the re-quantized DCT coefficient 108. is fed to a second input of subtractor 706. The output of subtractor 706, which is the difference between output 707 and output 108, representing the DCT coefficients of the error signals; i.e., "drift" signal, is connected to a first input of summer 705. The output of summer 705 is coupled to a first input of prediction frame memory storage device 703, and a second input of prediction frame memory storage device 703 is connected to the second output of VMD 104 (FIG. 1). The output of prediction frame storage device 703 is fed to a second input of summer 704 and this output is also fed to a second input of summer 705.

Inverse quantizer 701, quantizer 702, summers 704, 705, and subtractor 706 are system components which are well-known to those skilled in the art. Conventional components may be used for these items. With respect to the prediction frame storage device 703, this device includes a video buffer for storing information corresponding to one or more video frames, a random-access memory device, and a microprocessor for controlling the operation of the buffer and the random-access memory. The microprocessor is equipped to execute a software program adapted to perform the steps outlined below in connection with the prediction frame storage device 703.

The hardware configuration of FIG. 7 operates as follows. Assume that the quantized DCT coefficients of an input video bit stream having a bit rate of R1 passes from the first output 105 of VMD 104 (FIG. 1) to the input of inverse quantizer 701 (FIG. 7). At inverse quantizer 701, the quantified DCT coefficients are re-constructed to produce the DCT coefficients. The DCT coefficients plus the DCT coefficients of the "drift" error are sent to DCT coefficient processor 107. One purpose of DCT coefficients processor 107 (FIGs. 1 and 7) is to generate an output signal representing the processed coefficients. When the DCT coefficients processor 107 is initially started up, there is no drift error between output 707 and output 108 (FIG. 7). Therefore, upon initial startup, inverse quantizer 701 provides an output signal including reconstructed DCT coefficients, and this signal passes unchanged through summer 704, to quantizer 702.

The operation of quantizer 702 is controlled by a signal on signal line 114 from the rate control 113 circuit (FIG. 1), so as to provide a signal having the desired output bit rate at the output of the bit rate matching system of FIG. 1. The output of this bit rate matching system may be provided from the transmission buffer 111 (FIG. 1). Note that the output of quantizer 702 (FIG. 7) represents the DCT coefficients processor output 108. This output 108 is then recoded and multiplexed with motion vectors and quantization information by VME 109 (FIG. 1). The VME 109 may then send the recoded, multiplexed signal to transmission buffer 111. The signal is stored in the transmission buffer 111 prior to transmission at the desired output bit rate. The fullness, or buffer memory location occupancy ratio, of transmission buffer 111 is used to control the quantization levels for quantizer 702.

Next, assume that the output of inverse quantizer 701 does not equal the output of quantizer 702. The out-

Ex. 1018-216

put 707 of inverse quantizer 701 will be denoted as "A", and the output 108 of quantizer 702 will be denoted as "B". Thus, an error of B-A is added to the picture data. This error, denoted as Ed, is subtracted from the picture data by the system of FIG. 7. At initial startup. Ed is zero, and the data pass unchanged through summer 705 to the prediction frame storage device 703. Typically, only a small number of low-frequency coefficients are fed to subtractor 706, and thus, Ed is only an approximation of the actual drift error due to requantization. During recoding of the next video frame, Ed is approximately equal to the drift error of the previous frame.

During motion-compensated prediction, prediction frame storage device 703 uses motion vectors on the second output 106 of VMD 104 (FIG. 1) to output a displaced drift error signal. As the prediction is performed directly in the DCT domain, the output of the prediction frame storage device 703 representing the displaced drift error signal, is represented by its DCT coefficients.

To compute the displaced drift error directly in the DCT domain, the following operations are performed by the prediction frame storage device 703. Since the motion vector 106 represents an arbitrary number of pels, and the DCT representation of the reference frame stored in prediction frame storage device 703 are grouped into a block-based format, the motion compensated optimal block may overlay with four neighboring blocks in the DCT block structure.

Let D_1, D_2, D_3, D_4 be the four neighboring (adjoining) blocks in the reference frame, and PMC error be the DCT representation of the displaced drift error block. Then

$$D_{mc-error} = \sum_{i=1}^{4} G_{h`z} D_{`z} G_{`wz}$$

where G_h, G_w are the DCT representations of the spatial sparse H_n and H_w matrices of the form:

$$H_n = \begin{bmatrix} O & O \\ I_n & O \end{bmatrix} \quad H_w = \begin{bmatrix} O & I_w \\ O & O \end{bmatrix}$$

where h and w represent overlay width.

The G_{hz} and G_w can be precomputed and stored in prediction frame storage device 703. As the drift error signal contains a lot of zeroes, the required computation is greatly reduced. Further, if motion vectors are zero, or integer multiples of the block width, the above block adjustment procedure can be avoided. This drift error signal will be seen at the DCT coefficients processor output 108, and at transmission buffer 111 (FIG. 1) which receives bits at the desired output bit rate. Without correction, this drift error will accumulate over time and eventually result in unacceptable system performance. In order to ameliorate the problem of drift error accumulation, the previous frame motion compensated drift error Ed is added to the present frame signal (output 707) prior to requantization by quantizer 702. If quantizer 702 introduced very little error, this would completely correct the drift error accumulation problem. However, since quantizer 702 introduces a finite amount of error, the drift can only be partially corrected, and the output of subtractor 706 will not, in general, be zero. Thus, summer 705 adds the drift error from the current frame to the approximate accumulated drift error from previous frames to produce an approximate accumulated drift error Ed for the current frame.

The prediction frame storage device 703 only has to compute a small number (i.e., N) of compensated coefficients. Note that, for intra-blocks of video data, the prediction frame storage device 703 is programmed to set Ed to zero. The relatively small number of computations required to implement the methods disclosed herein is vastly reduced as contrasted with the relatively large number of computations required to perform existing processes using pel domain motion compensation. An additional advantage of the disclosed methods is that these methods require much less memory space than existing prior art methods.

The video transmission rate reduction system shown in FIG. 1 can be implemented, for example, by using a general-purpose microprocessor, a digital signal processor (such as an AT&T DSP 3210 or an AT&T DSP 1610), and/or a programmable video processing chip (such as an integrated circuit known to those skilled in the art as the ITT VCP chip).

Multimedia System Using Video Processing of the Present Invention

To illustrate various typical applications for the present invention in the context of multimedia conferencing, FIG. 8 shows a multimedia system using a video processor embodying the bit stream rate matching techniques disclosed herein. Referring now to FIG. 8, a block diagram setting forth the system architecture of a multimedia conferencing system 800 is shown. The conferencing system includes an MCU 810, an ISDN network 804, and a plurality of endpoint devices such as first endpoint device 801, second endpoint device 802, and third endpoint device 803.

Endpoint devices 801, 802, and 803 are coupled to MCU 810 via ISDN network 804. These endpoint devices 801, 802, and 803 may include one or more user interface devices. Each interface device includes either an input means, an output means, or an input means combined with an output means. Output means are adapted to convert multimedia electronic signals representing audio, video, or data into actual audio, video, or data. Input means are adapted to accept audio, video, and/or data inputs, and to convert these inputs into electronic signals representing audio, video, and/or data. Examples of user interface devices include video display, keyboards, microphones, speakers, and video cameras, or the like.

Endpoint devices 801, 802, and 803 are adapted to

Ex. 1018-217

communicate using existing multimedia communication protocols such as ISDN. The endpoint device multimedia communication protocol controls the presentation of media streams (electronic signals representing audio, video, and/or data information) to the endpoint device user. Endpoint devices may function bi-directionally, both sending and receiving multimedia information, or, alternatively, endpoint devices may function uni-directional, receiving but not sending multimedia information, or sending but not receiving multimedia information.

An example of a suitable endpoint device is an ITU-T H.320 audiovisual terminal, but any device capable of terminating a digital multimedia stream and presenting it to the user constitutes an endpoint device. A particular product example of an H.320-compatible endpoint is the AT&T-GIS Vistium.

MCU 810 is a computer-controlled device which includes a multiplicity of communications ports, such as first communications port 870 and second communications port 872, which may be selectively interconnected in a variety of ways to provide communication among a group of endpoint devices 801, 802, 803. Although the system of FIG. 8 shows two communications ports, this is done for illustrative purposes, as any convenient number of communications ports may be employed. MCU 810 also includes a control processor 840, an audio processor 841, a video processor 842, a data processor 843, and a common internal switch 819. Each communications port includes a network interface, a demultiplexer, and a multiplexer. For example, first communications port 870 includes network interface 811, demultiplexer 813, and multiplexer 822.

Although MCU 810 is shown with two communications ports 870, 872 for purposes of illustration, MCU 810 may, in fact, include any convenient number of communications ports. For an MCU 810 having N ports, there are N network interfaces, one control processor, one audio processor, one video processor, and one data processor. For each processor, there are N input signals coming from N demultiplexers and N output signals going to the N multiplexers. Therefore, MCU 810 may be conceptualized as an N-port MCU where only two communications ports 870, 872 are explicitly shown.

As shown in FIG. 8, first communications port 870 includes network interface 811, demultiplexer 813, and multiplexer 812. Network interface 811 is a circuit which provides the conversion function between the standard line signal coding used by ISDN network 804 and the Px64 kbps H.221 signal used by MCU 810. Network interface 811 includes output port 812, which is adapted to provide an output in the form of an H.221 signal. The H.221 signal is actually a multiplex of several different types of information (audio, video, data, control); therefore, network interface 811 must send the incoming MCU H.221 signal to a demultiplexing device such as demultiplexer 813. Likewise, network interface 811 has an input port 823 adapted to receive an H.221 signal from multiplexer 822. Multiplexer 822 combines a plurality of individually-processed signals which are to be transmitted to a particular endpoint device.

Demultiplexer 813 separates an incoming multimedia signal stream received from network interface 811 into four components: a first component 814, comprising electronic signals representing control; a second component 815, comprising electronic signals representing audio; a third component 816, comprising electronic signals representing video; and a fourth component 817, representing data. The first, second, third, and fourth components 814, 815, 816, 817 represent outputs of demultiplexer 813 which are coupled to common internal switch 819.

Multiplexer 822 accepts a plurality of incoming multimedia signal components from common internal switch 819, such as a first component 818 representing control, a second component 891 representing audio, a third component 820 representing video, and a fourth component 821 representing data. The multiplexer 822 integrates the first, second, third, and fourth components 818, 891, 820, 821 onto a single multimedia signal stream which is coupled to network interface 811. This single multimedia signal stream may be conceptualized as the output of multiplexer 822. The network interface 811 routes this multimedia signal stream to a specific endpoint device 801, 802, 803. For second communications port 872, the four output components are first component 824, representing control, second component 825, representing audio, third component 826, representing video, and fourth component 827, representing data. The four input components to multiplexer 834 are first component 828, representing control, second component 829, representing audio, third component 830, representing video, and fourth component 831, representing data.

Common internal switch 819 contains a plurality of electronic switches, buffers, and/or amplifiers under the control of control processor 840. Common internal switch 819 is coupled to audio processor 841 for mixing and switching electronic signals representing audio; common internal switch 819 is also coupled to video processor 842 and data processor 843 for mixing and switching electronic signals representing video and data, respectively. Therefore, common internal switch 819 effectively receives four output components from each communications port 870, 872 and routes these output components to selected ones of respective processors (control processor 840, audio processor 841, video processor 842, and/or data processor 843) within MCU 810. Likewise, common internal switch 819 receives the output components of each processor in MCU 810 and routes these outputs to the multiplexer 822 of each communications port 870.

Common internal switch 819 receives output control signals from control processor 840 over signal line 851, and provides input control signals to control processor 840 over signal line 850. Common internal switch 819 receives output audio signals from audio processor

Ex. 1018-218

841 over signal line 853, and provides input audio signals to audio processor 841 over signal line 852. Common internal switch 819 receives output video signals from video processor 842 over signal line 855, and provides input video signals to video processor 842 over signal line 854. Common internal switch 819 receives output data signals from data processor 843 over signal line 857, and provides input data signals to data processor 843 over signal line 856. Control processor 840 provides control signals to the audio processor 841, video processor 842, and data processor 843 over signal line 844.

ISDN network 804 is connected to MCU 810 over signal line 805. Within MCU 810, signal line 805 is parallel-connected to first and second communications ports 870, 872. For example, in the case of first communications port 870, signal line 805 is connected to network interface 811. Network interface 811 is coupled to demultiplexer 813 over signal line 812, and this network interface 811 is also coupled to multiplexer 822 over signal line 823. Signal line 812 is coupled to the input terminal of demultiplexer 813, and signal line 823 is coupled to the output terminal of multiplexer 822.

Audio processor 841 includes software and hardware for processing audio signals. The processing may take the form of switching the audio, mixing the audio, or both. In the case of audio mixing, the input signal to audio processor 841 is an aggregate audio signal consisting of each of the audio output signals from all of the communications ports 870, 872 of MCU 810. For an N-port MCU 810, this signal includes the N audio signals from the demultiplexers within each communications port 870, 872.

To mix the audio, audio processor 841 decodes each of the audio inputs, linearly adds the signals obtained by decoding, and then re-encodes the linear sum. For each endpoint device, this linear sum may be subjected to additional processing steps, so as to provide each endpoint device with audio information specific to that endpoint device. These additional processing steps may include, for example, any of the following: the output sum for a given endpoint device may exclude that endpoint's input; the sum may include inputs whose present or recent past values exceed a certain threshold; or the sum may be controlled from a specially-designated endpoint device used by a person termed the "chair", thereby providing a feature generally known as chair-control. Therefore, the output of the audio processor 841 is in the form ofN processed audio signals.

In the case of audio switching, the input signal to audio processor 841 is a single audio signal which is selected from a given communications port 870 or 872, based upon control signals received from control processor 840. No audio processing is implemented in the present example which involves only audio switching. The audio input is broadcast to all other audio processor 841 outputs, either automatically or under manual control.

Data processor 843 includes hardware and software means for implementing one or both of the functions generally known to those skilled in the art as "broadcast" or "MLP". For each type of broadcast data, data input is accepted from only one endpoint device at any one time. Therefore, the input signal to data processor 843 is the data output from one of the communications ports 870, 872. This data output is broadcast to the other endpoint devices as determined by control processor 840, according to the capabilities of specific endpoint devices to receive such data, as set forth in the capability codes stored in memory units (RAM or ROM) of respective endpoint devices.

Control processor 840 is responsible for determining the correct routing, mixing, switching, format and timing of the audio, video, data and control signals throughout a multimedia conference. The control processor 840 retrieves one or more capability codes from each endpoint device. Capability codes, which, are stored in endpoint device RAM and/or ROM, specify the audio, video, data, and/or control capabilities for this endpoint device. Control processor 840 retrieves the capability codes from all N endpoint devices participating in a multimedia conference. These capability codes are stored in a memory unit (RAM) of MCU 810 so that control processor 840 can correctly manage the conference for all endpoint devices. This storage may occur, for example, in a random-access memory (RAM) device associated with control processor 840. In turn, MCU 810 sends the capability codes to each of the N communications ports 870, 872 so that each of the endpoint devices 801, 802, 803 are enabled to communicate with MCU 810 at a bit rate determined by MCU 810 and appropriate for that specific endpoint device 801, 802, 803. To properly control the operations in the video processing unit 842, MCU 810 sends back different capability codes for different video operations. If the endpoints are coupled to MCU 810 via communications links with different transmission rates for switching operation, the minimum transfer rate is adapted in the capability code. The new code is sent to all the endpoint devices participating in the conference to force the endpoint devices to operate with the minimum transfer rate. For performing a transmission rate matching operation, instead of adapting the minimum transfer rate, a new maximum frame rate is specified in the capability code based on all the different transfer rates. The new capability codes, which include the specified maximum frame rates, are sent back to all the endpoint devices. In this case, the video bitstreams generated by the endpoint devices will have different transfer rates but the same frame rate.

Control processor 840 receives inputs which are entered by conference participants into the user interface of an endpoint device 801, 802, 803. These inputs are in the form of chair-control commands and commands embedded in bit streams conforming to the H. 221 standard. Commands from endpoint devices are routed to the control processor 840 to ensure the correct

Ex. 1018-219

distribution of bit streams to the audio, video, and data processors 841, 842, 843, respectively, to ensure that the correct audio decoding algorithm is used at the inputs to an audio mixer within audio processor 841, and to ensure that any incoming data is sent to a data broadcast unit or MLP processor within data processor 843.

The control processor 840 also directs the switching of the bit streams from the audio, video, and data processors 841, 842, 843, respectively, to each multiplexer 822, 834, and specifies the audio encoding algorithm used in the audio mixer of audio processor 841, and the algorithm used at each output from the audio mixer. The bit streams are routed to and from the various processors 841, 842, 843 by the common internal switch 819, which is under control of the control processor 840.

Video processor 842 processes the video signals received from the common internal switch 819. The processing may take the form of switching the video, or matching the video bit rate. In video switching, the video processor 842 receives one selected video signal from the switch 819, and transmits the video signal to some or all other endpoint devices participating in a given multimedia conference. Video selection may be automatic or under manual control. For instance, the audio processor 841 and the video processor 842 may be automatically controlled by control processor 840, such that an endpoint device with currently active audio (i.e., an endpoint device used by the "present speaker" which provides an audio signal to MCU 810 above a predetermined audio amplitude threshold) receives the picture of the endpoint device which previously had active audio (i.e., an endpoint device used by the "previous speaker"), while all other endpoint devices receive the picture of the present speaker.

A time delay may be incorporated into the video switching implemented by video processor 842 to avoid excessively frequent video image changes caused by spurious sounds. As in the case of audio switching, video switching may be controlled directly from a specially-designated endpoint device used by a person termed the "chair". If the delay in the video processor 842 and the delay in the audio processor 841 differ by a significant (humanly perceptible) amount, a compensating delay may be inserted into the appropriate bit stream to retain lip synchronization.

To match video bit rates, video processor 842 incorporates the video transmission rate matching techniques of the present invention.

With reference to FIG. 9, the hardware configuration of an illustrative stand-alone video processor 842 is shown. This video processor 842 may be employed with the MCU 810 of FIG. 8 or, alternatively, the video processor 842 may exist as a stand-alone unit. Such a stand-alone unit may be used to match bit rates for all types of digital video information. The video processor 842 consists of a video transmission rate reduction unit 100, a bit stuffing unit 200, an input switcher 905, and an output switcher 906. If desired, the input switcher 905 and the output switcher 906 could be combined into a single integrated switcher unit, such as common internal switch 819 (FIG. 8) and, hence, would not be incorporated into video processor 842. The inputs to the video processor 842, are for an N-port MCU, an the N-coded video bit stream obtained from N demultiplexers. The outputs of the system are the N-coded video bit streams which are processed to have different transfer rates. The outputs are the inputs to the N multiplexers. The number of required transmission rate reduction units and bit stuffing units is proportional to the difference in the transmission rates among the various endpoint devices. If there are M different video transfer rates involved in a conference, the video processor 842 needs to incorporate (M-1) bit rate reduction systems and (M-1) bit rate increasing units. In the configuration of FIG. 9, M=2 for illustrative purposes.

Signals produced by video processor 842 are coupled to signal line 844 which is used to convey the control signals which control input switcher 905, output switcher 906, transmission rate reduction unit 100 and bit stuffing unit 200. Input switcher 905 provides N-to-1, N-to-2, and N-to-3 switching capabilities. Similarly, output switcher 906 provides 1-to-N, 2-to-N, and 3-to-N switching capabilities. The switching is controlled by control signals on signal line 844. These control signals are generated by the control processor 840. Control processor 840 has control capabilities for handling different transmission rates among a plurality of endpoint devices.

If the transmission rates among all the endpoint devices are the same, the control signals sent out by control processor 840 over signal line 844 serves to place video processor 842 into a switching mode. In such a case, video processor 842 works as a buffer and switcher, such that the input switcher provides an N-to-1 switch and the output provides an 1-to-(N-1) switch. A direct signal path between input switcher 905 and output switcher 906 is utilized. whereas video transmission rate reduction unit 100 and bit stuffing unit 200 are not used.

If the transmission rates among the endpoint devices are different, and if there are more than two different transmission rates among the endpoints, extra rate reduction units 100 and extra bit stuffing units 200 are needed in the video processor 842. As an example, FIG. 10 shows a video transmission bit rate matching system which uses five endpoint devices connected to an ISDN network 804 via a five-point connection wherein the switching functions of video processor 842 are integrated into common internal switch 819. The ISDN network 804 is connected to a five-port MCU 810, including first communications port 840 and additional communications ports 871. A first endpoint device, endpoint A 901, is connected to the ISDN network 804 via a 384 kbits/s communications link. A second endpoint device, endpoint B 902, is connected via a 256 kbits/s link. Third and fourth endpoint devices, endpoint C 903 and endpoint D 904, respectively, are each connected to ISDN

Ex. 1018-220

network 804 via 128 kbits/s links, and a fifth endpoint device, endpoint E 905, is connected via a 64 kbits/s link. Therefore, the MCU 910 must utilize a video processor 842 having five inputs and five outputs. To produce five outputs with three different transfer rates, the video processor needs three rate reduction units 920, 921, 922, three bit stuffing units 923, 924, 925, and an optional video buffer. The three rate reduction units 920, 921, 922 provide respective rate reductions of 348-to-256 kbits/sec, 384-to-128 kbits/sec, and 384-to-64 kbits/sec. The three bit stuffing units 923, 924, 925 provide respective bit stuffing conversions of 64-to-128 kbits/sec, 64-to-256 kbits/sec, and 64-to-384 kbits/sec.

To provide the proper switching for the inputs and the outputs, the control processor 841 has to provide proper control signals. If endpoint A 901 is used by the current conference speaker and endpoint B 902 is used by the conference speaker who spoke immediately prior to the speaker using endpoint A 901, then the input switcher 305 provides an input switching function wherein 5 inputs are switched to 4 outputs, such that the coded bit stream from endpoint A 901 is switched to any one of the rate reduction units 920, 921, 922 which provide respective bit rate reductions of 384-to-256 kbits/sec, 384-to-128 kbits/sec, and 384-to-64 kbits/sec. To send the picture of the previous speaker to endpoint A 901, the coded bit stream from endpoint B 902 is switched to bit stuffing unit 925, which performs a bit stuffing from a rate of 256 kbits/sec to a rate of 384 kbits/sec. Via the output switcher 306 (FIG. 8), the output of rate reduction unit 920, which performs a bit rate reduction of 384 kbits/sec to 256 kbits/sec, is routed to endpoint B 902, the output of rate reduction unit 921, which performs a rate reduction of 384 kbits/sec to 128 kbits/sec, is routed to endpoints C and D, 903, 904, respectively, and the output of rate reduction unit 922, performing a reduction of 384-to-64 kbits/sec, is routed to endpoint E 905. The output of bit stuffing unit 925 (64-to-384 kbits/sec), is routed to endpoint A 901.

Claims

1. A bit stream rate matching apparatus characterized by:

   (a) bit rate increasing means for converting a video bit stream having a first transmission rate to a video bit stream having a second transmission rate, wherein the first transmission rate is less than the second transmission rate;
   (b) bit rate decreasing means for converting a video bit stream having a third transmission rate to a video bit stream having a fourth transmission rate wherein the third transmission rate is greater than the fourth transmission rate;
   (c) first switching means for switching each of a plurality of incoming video bit streams to any one of the bit rate increasing means the bit rate decreasing means and a second switching means; and
   (d) second switching means for switching video bit streams from the bit rate increasing means, the bit rate decreasing means, and the first switching device to each of a plurality of outgoing video bit streams.

2. A bit stream rate matching apparatus as set forth in Claim 1 characterized in that each of the plurality of incoming video bit streams is from a corresponding endpoint device, and each of the plurality of outgoing video bit streams is directed to a corresponding endpoint device, specific ones of the incoming video bit streams being associated with specific ones of the outgoing video bit streams to form an associated incoming-outgoing video bit stream, each associated incoming-outgoing video bit stream corresponding to a particular endpoint device.

3. A bit stream rate matching apparatus as set forth in Claim 1 further comprising processor means 840, coupled to the first and second switching means, bit rate increasing means, and bit rate reduction means, for controlling the first and second switching means.

4. A bit stream rate matching apparatus as set forth in Claim 2 further characterized by processor means, coupled to the first and second switching means for controlling the first and second switching means.

5. A bit stream rate matching device characterized by

   (a) bit rate increasing means for converting a video bit stream having a first transmission rate to a video bit stream having a second transmission rate, wherein the first transmission rate is less than the second transmission rate, and;
   (b) bit rate reducing means for converting a video bit stream having the second transmission rate to a video bit stream having the first transmission rate.

6. A video bit stream rate matching device for use in a system characterized by:

   a first endpoint device characterized by video transceiver means for transmitting and receiving a video bit stream having a first transmission rate, and
   a second endpoint device characterized by video transceiver means for transmitting and receiving a video bit stream having a second transmission rate, wherein the second transmission rate differs from the first transmission

Ex. 1018-221

rate;
the video bit stream rate matching device being coupled to the first and second endpoint devices;

CHARACTERIZED IN THAT
the video bit stream rate matching device includes rate conversion means for converting a video bit stream having a first transmission rate to a video bit stream having a second transmission rate, and for converting a video bit stream having a second transmission rate to a video bit stream having a first transmission rate.

7. The video bit stream rate matching device of Claim 1, further characterized by:

(a) video multiplex decoder means for decoding and demultiplexing an input video bit stream having a first transmission rate; the video multiplex decoder having a first output adapted to generate quantized information in the form of quantized discretet cosine transformation coefficients in response to the input video bit stream; the video multiplex decoder means having a second output adapted to generate motion vectors in response to the input video bit stream;
(b) video multiplex encoder means for encoding and multiplexing quantized discrete cosine transformation coefficients with motion vectors to generate an output video bit stream; the video multiplex encoder means having a first input adapted to receive quantized direct cosine transformation coefficients, a second input adapted to receive motion vectors, and an output adapted to generate the output video bit stream;
(c) discrete cosine transformation coefficients processor means for processing a plurality of quantized discrete cosine transformation coefficients generated from a decoded, demultiplexed video bit stream having N bits, the processor processing the quantized discrete cosine transformation coefficient such that, after processing, the quantized discrete cosine transformation coefficients provide an encoded. multiplexed video bit stream having M bits, wherein M≠N; the discrete cosine transformation coefficients processor means having a rate control input for specifying a desired bit rate for the encoded, multiplexed video bit stream; and
(d) rate control means for controlling the bit rate of the encoded, multiplexed video bit stream, the rate control means having:

(i) a first communications port for receiving a desired bit rate, and adapted for coupling to an endpoint device;
(ii) a second communications port for receiving a video bit stream having a transmission rate R; and
(iii) an output port for generating a video bit stream transmission rate S^1 in response to the receipt of transmission rates at the first and second communications ports;

characterized in that the first output of the video multiplex decoder means is coupled to the direct cosine transformation coefficients processor means, the second output of the video multiplex decoder means is coupled to the second input of the video multiplex encoder means, the output of the video multiplex encoder means is coupled to the second communications port of the rate control means, and the output port of the rate control means is coupled to the rate control input of the direct cosine transformation coefficients processor.

8. A digital video transmission rate matching system characterized by:

a first switching means having N input ports, first, second, and third output ports, and a control signal input port; the control signal port being adapted to receive a plurality of control signals, each control signal specifying a particular set of connections between the N input ports and any of the first, second and third output ports; the first switching means, in response to the receipt of any one of the plurality of control signals, connecting the N input ports to any of the first, second, and third output ports as specified by the any one of the plurality of control signals;
a second switching means having first, second, and third input ports, N output ports and a control signal input port; the control signal input port being adapted to receive a plurality of control signals, each control signal specifying a particular set of connections between any of the first, second, and third input ports and the N output ports; the second switching means, in response to the receipt of any one of the plurality of control signals, connecting the N output ports to any of the first, second and third input ports as specified by the any one of the plurality of control signals;
a transmission rate reduction means having an input adapted to receive a video bit stream of bit rate R bits per second, an output adapted to transmit a video bit stream of bit rate S^1 bits per second, wherein R is greater than S^1, and video bit stream compression means for transforming a video bit stream of bit rate R bits per second

Ex. 1018-222

to a video bit stream of bit rate S bits per second;
a transmission rate increasing means having an input adapted to receive a video bit stream of bit rate S bits per second, an output adapted to transmit a video bit stream of bit rate R bits per second, wherein R is greater than S, and video bit stream expansion means for transforming a video bit stream of bit rate S bits per second to a video bit stream of bit rate R bits per second;
characterized in that the first output port of the first switching means is coupled to the input of the transmission rate reduction means, the second output port of the first switching means is coupled to the input of the transmission rate increasing means, the third output port of the first switching means is coupled to the third input port of the second switching means;
the first input port of the second switching means is coupled to the output of the transmission rate reduction means; the second input port of the second switching means is coupled to the output of the transmission rate increasing means;
the N input ports of the first switching means are each adapted to receive an incoming video bit stream and the N output ports of the second switching means are each adapted to transmit an outgoing video bit stream.

9. The digital video transmission rate matching system of Claim 4 characterized in that the transmission rate reduction means is the video bit stream rate matching device of Claim 1, 2 or 3.

10. The digital video transmission rate matching system of Claim 4, 5, 6 or 7 characterized in that the transmission rate increasing means is a bit stuffing device.

11. A method of providing video bit stream rate matching, characterized by the following steps:

a) demultiplexing an incoming video bit stream to generate a plurality of demultiplexed video bit streams;
b) decoding each of a plurality of the demultiplexed video bit streams;
c) performing a discrete cosine transformation (DCT) on each of the demultiplexed, decoded video bit streams to generate a plurality of DCT coefficients;
d) discarding a subset of the DCT coefficients;
e) encoding the non-discarded DCT coefficients to generate a plurality of encoded video bit streams; and
f) multiplexing the plurality of encoded video bit streams to generate a multiplexed, encoded, outgoing video bit stream.

12. A method of providing video bit stream rate matching as set forth in Claim 11, characterized in that the DCT coefficients include a first subset of DCT coefficients corresponding to low-frequency video information and a second subset of DCT coefficients corresponding to high-frequency video information, the step of discarding being performed by discarding the second subset of DCT coefficients.

13. A method of providing video bit stream rate matching as set forth in Claim 12 characterized in that the high-frequency video information corresponds to video information above approximately 4.5 mHz.

Ex. 1018-223

FIG. 1

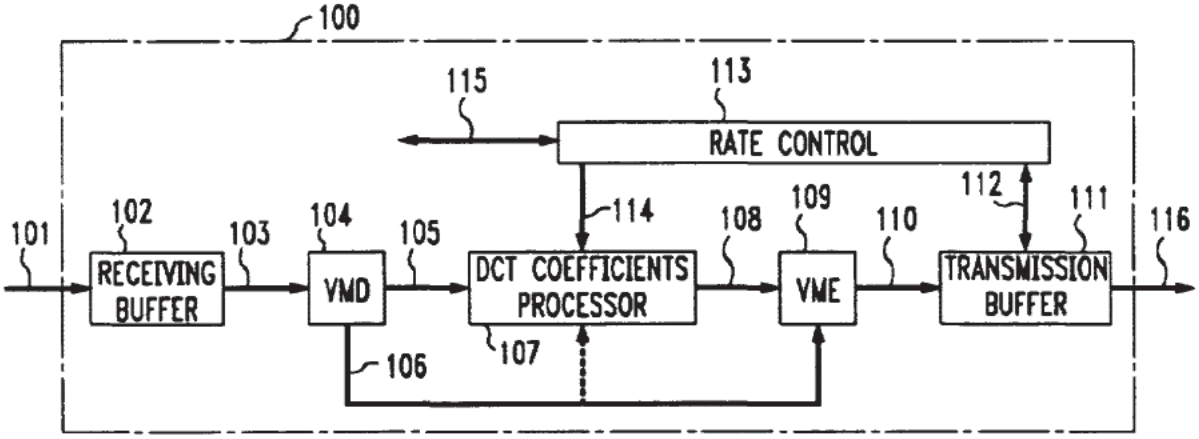
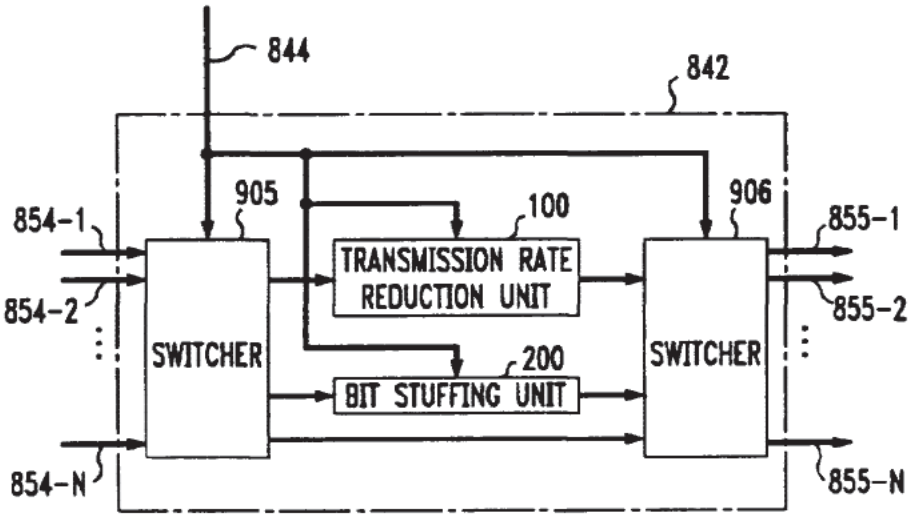


FIG. 9



Ex. 1018-224

Ex. 1018-225

FIG. 2

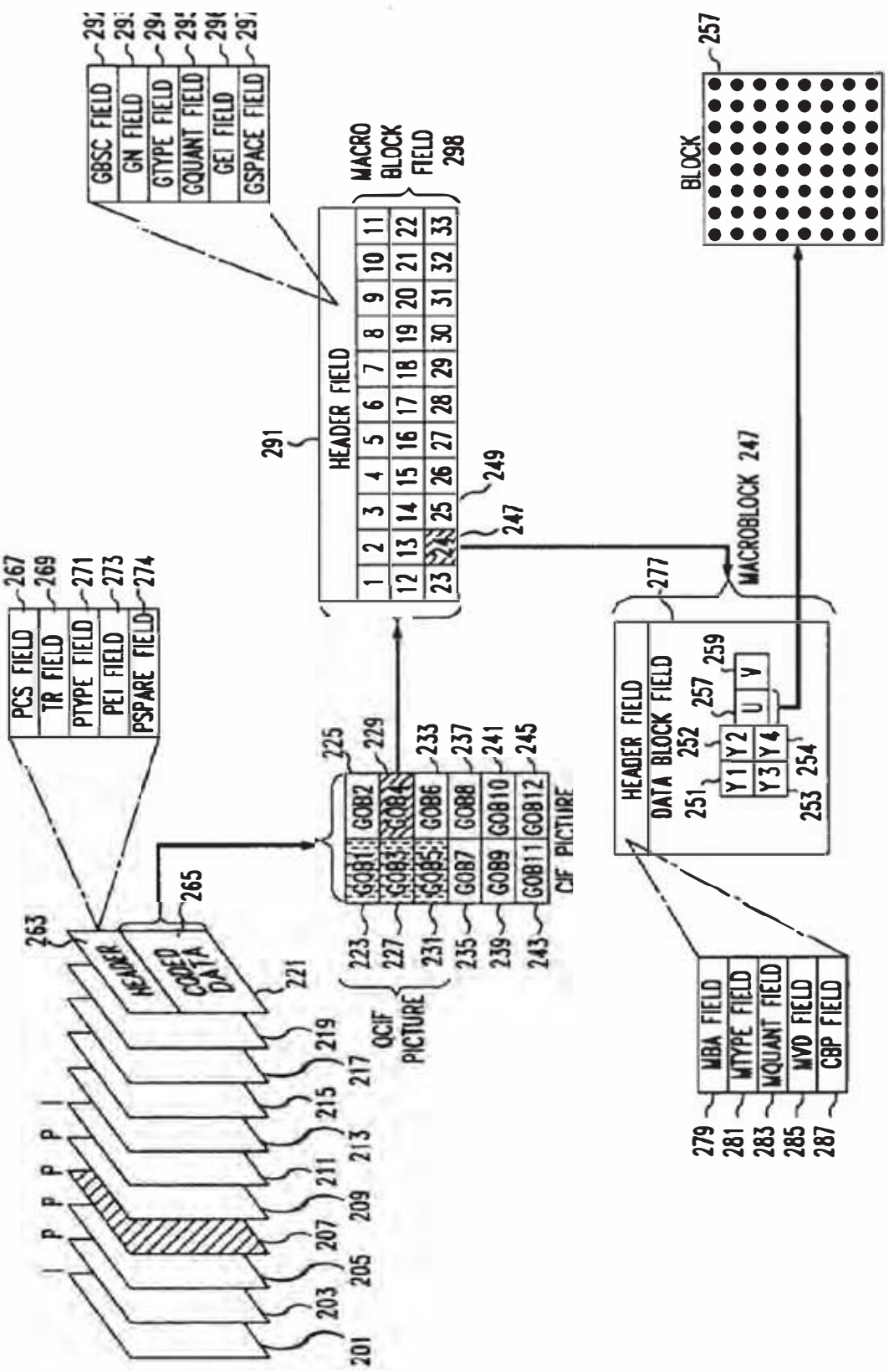


FIG. 3

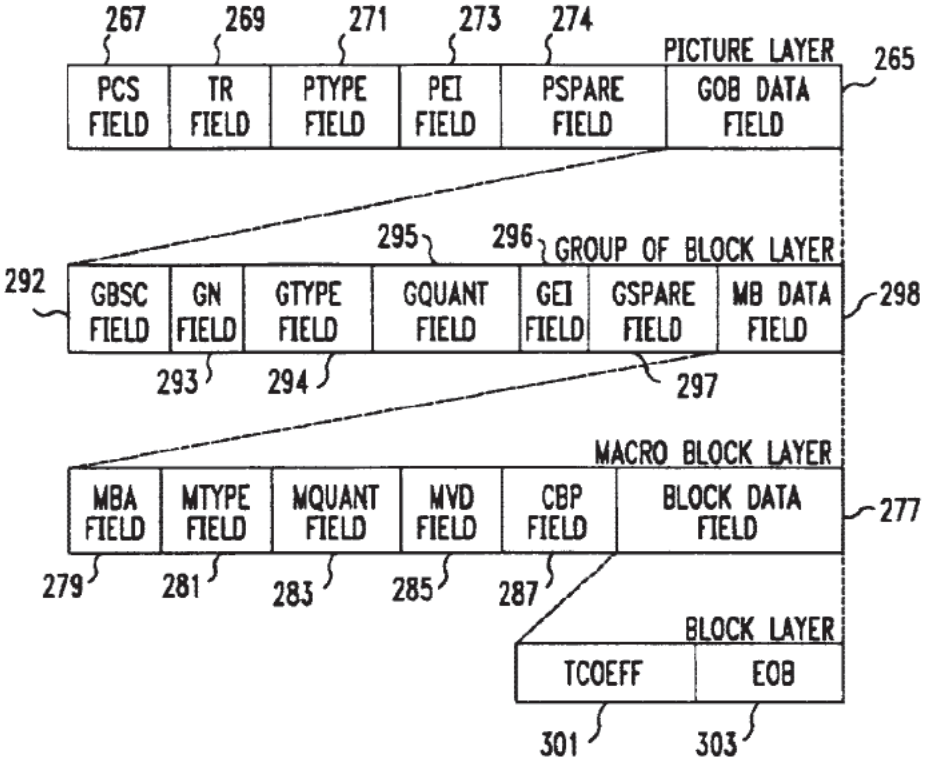
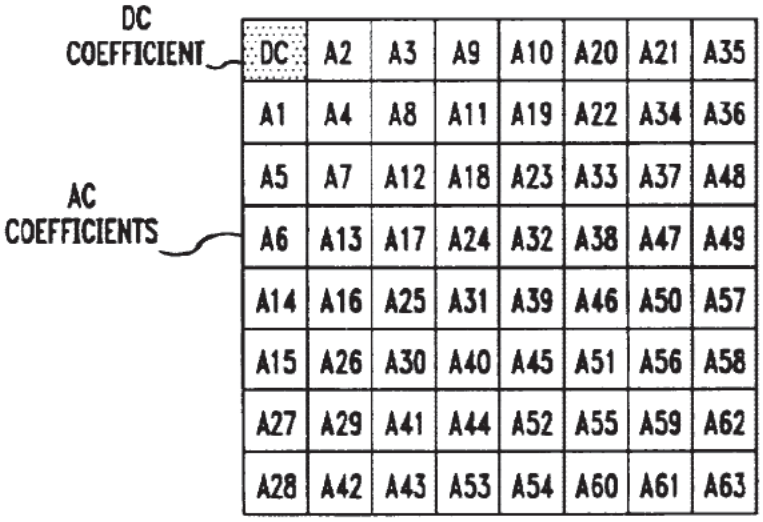


FIG. 4



Ex. 1018-226

FIG. 5

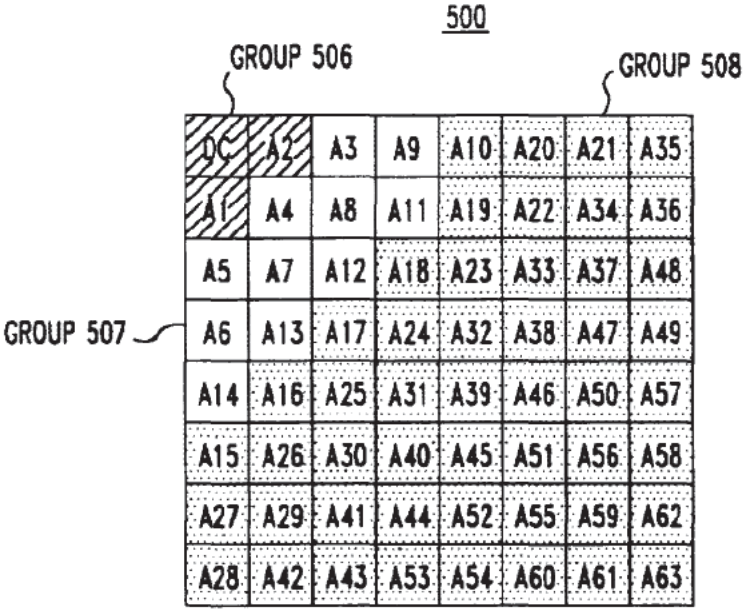

500
GROUP 506
GROUP 508
GROUP 507
DC A2 A3 A9 A10 A20 A21 A35
A1 A4 A8 A11 A19 A22 A34 A36
A5 A7 A12 A18 A23 A33 A37 A48
A6 A13 A17 A24 A32 A38 A47 A49
A14 A16 A25 A31 A39 A46 A50 A57
A15 A26 A30 A40 A45 A51 A56 A58
A27 A29 A41 A44 A52 A55 A59 A62
A28 A42 A43 A53 A54 A60 A61 A63

FIG. 6

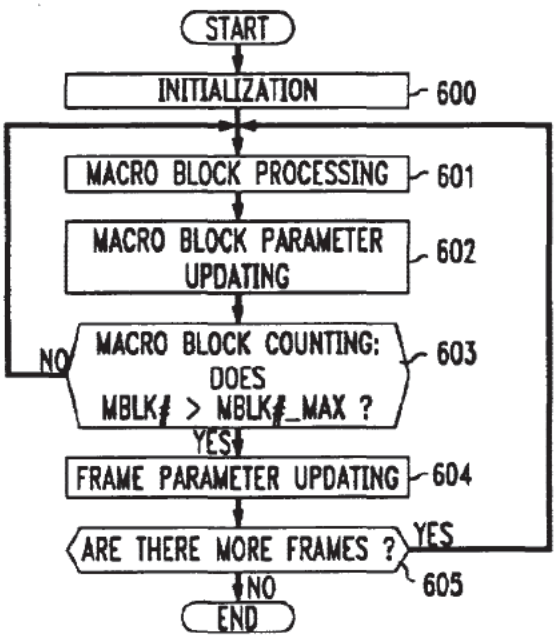

START
INITIALIZATION
600
MACRO BLOCK PROCESSING
601
MACRO BLOCK PARAMETER UPDATING
602
MACRO BLOCK COUNTING: DOES MBLK# > MBLK#_MAX ?
603
NO
YES
FRAME PARAMETER UPDATING
604
ARE THERE MORE FRAMES ?
YES
605
NO
END

FIG. 7

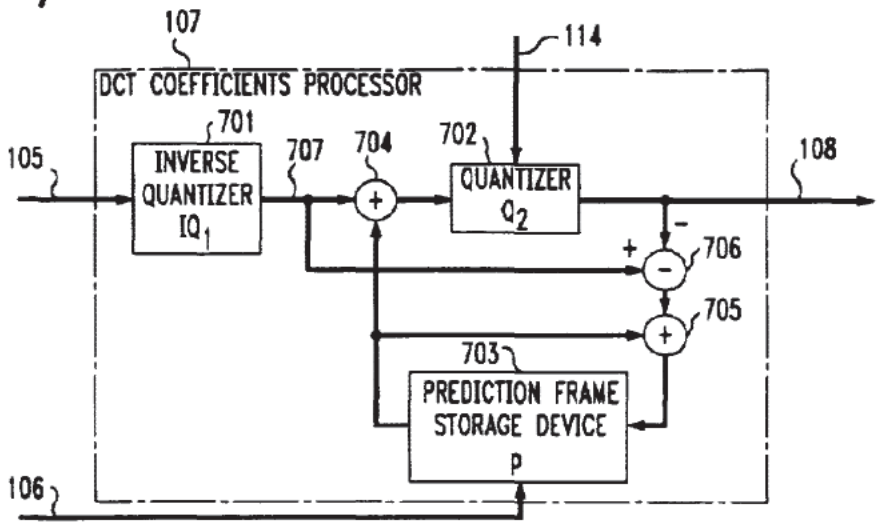
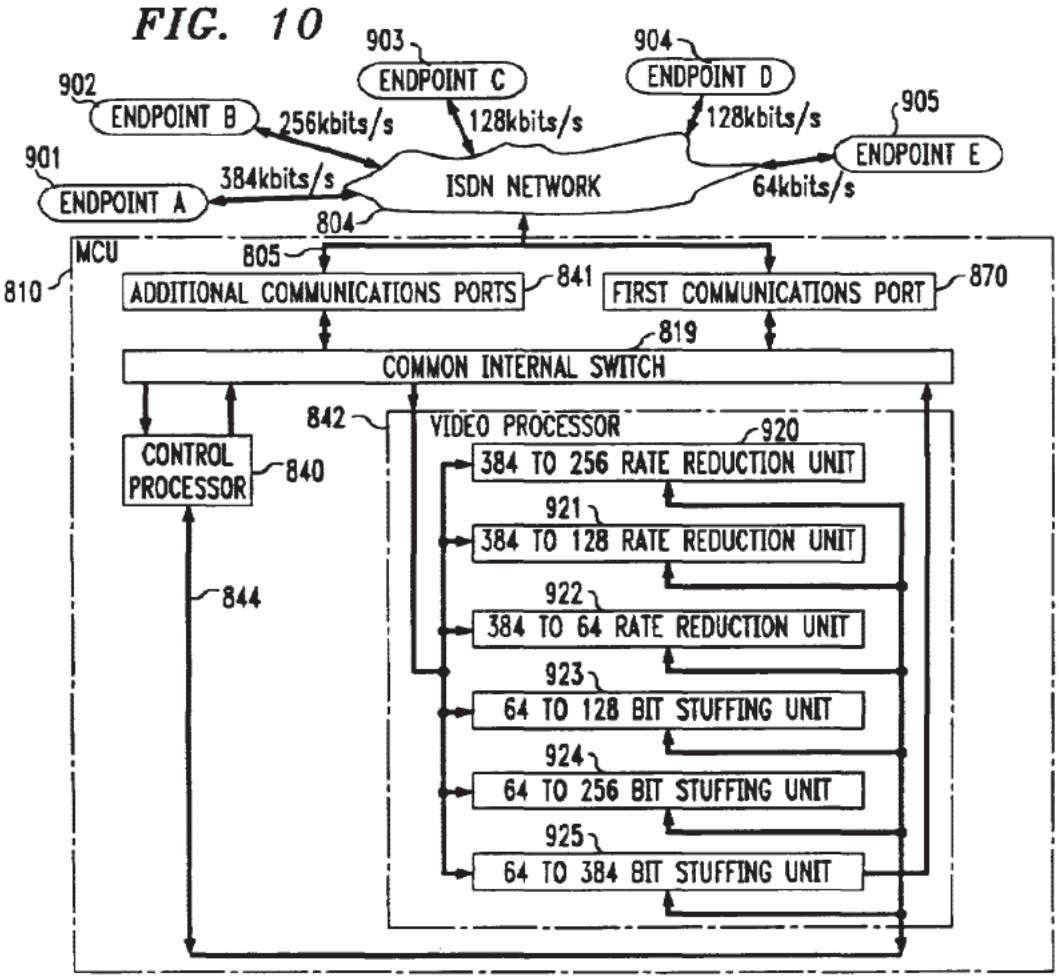


FIG. 10



Ex. 1018-228

Ex. 1018-229

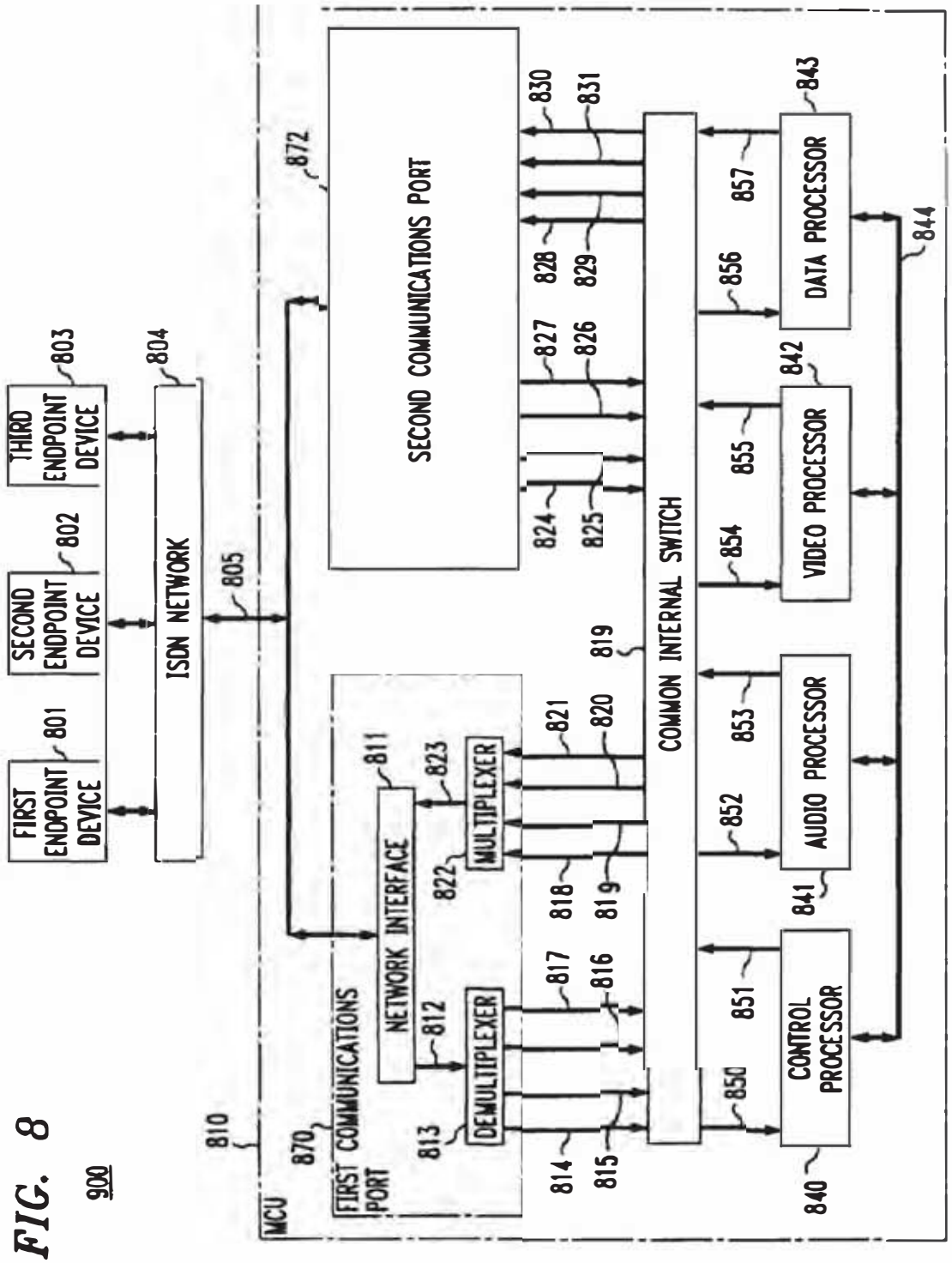


FIG. 8

900

(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) EP 1 670 256 A2

(12) **EUROPEAN PATENT APPLICATION**

(43) Date of publication:
14.06.2006 Bulletin 2006/24

(51) Int Cl.:
H04N 7/26 *(2006.01)*

(21) Application number: **05111156.5**

(22) Date of filing: **23.11.2005**

(84) Designated Contracting States:
AT BE BG CH CY CZ DE DK EE ES FI FR GB GR HU IE IS IT LI LT LU LV MC NL PL PT RO SE SI SK TR
Designated Extension States:
AL BA HR MK YU

(30) Priority: **10.12.2004 US 10114**
10.12.2004 US 10113
10.12.2004 US 10040

(71) Applicant: **Microsoft Corporation**
Redmond WA 98052 (US)

(72) Inventors:
- **Klemets, Anders E.,**
c/o Microsoft Corporation
Redmond, WA 98052 (US)
- **Huang, Cheng,**
c/o Microsoft Corporation
Redmond, WA 98052 (US)
- **Chou, Philip A.,**
c/o Microsoft Corporation
Redmond, WA 98052 (US)

(74) Representative: **Grünecker, Kinkeldey,**
Stockmair & Schwanhäusser
Anwaltssozietät
Maximilianstrasse 58
80538 München (DE)

(54) **A system and process for controlling the coding bit rate of streaming media data**

(57) A system and process for controlling the coding bit rate of streaming media data is presented. This coding bit rate control involves dynamically adjusting the coding bit rate to control client buffer duration to prevent the buffer from underflowing, while keeping the average coding bit rate close to the average transmission bit rate of the network (an thus maximizing the quality of the data playback). Using the theory of optimal linear quadratic control, the client buffer duration is kept as close as possible to a target level while still keeping the coding bit rate (and hence the quality) as constant as possible. In addition, a leaky bucket model is incorporated into the control loop so that the changes in buffer duration due to natural variation in the instantaneous coding bit rate are not mistaken for changes in buffer duration due to network congestion.

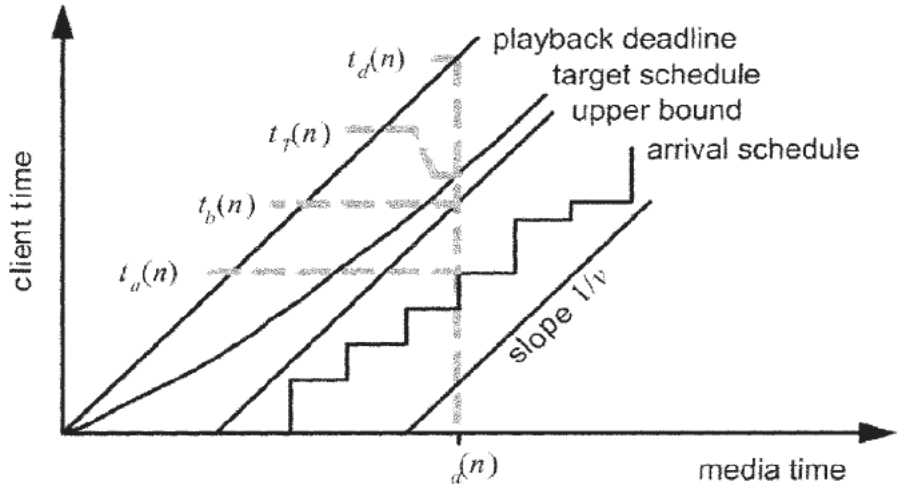


FIG. 5

EP 1 670 256 A2

Printed by Jouve, 75001 PARIS (FR)

Description

BACKGROUND

Technical Field:

[0001] The invention is related to controlling the coding bit rate of streaming media, and more particularly to a system and process for controlling the coding bit rate of streaming media data that provides fast startup, continuous playback, and maximal quality and smoothness over the entire streaming session.

Background Art:

[0002] Perhaps the major technical problem in streaming media on demand over the Internet is the need to adapt to changing network conditions. As competing communication processes begin and end, the available bandwidth, packet loss and packet delay all fluctuate. Network outages lasting many seconds can and do occur. Resource reservation and quality of service support can help, but even they cannot guarantee that network resources will be stable. If the network path contains a wireless link, for example, its capacity may be occasionally reduced by interference. Thus it is necessary for commercial-grade streaming media systems to be robust to hostile network conditions. Moreover, such robustness cannot be achieved solely by aggressive (nonreactive) transmission. Even constant bit rate transmission with re-transmissions for every packet loss cannot achieve a throughput higher than the channel capacity. Some degree of adaptivity to the network is therefore required.

[0003] End users expect that a good streaming media system will exhibit the following behavior: content played back on demand will start with low delay; once started, it will play back continuously (without stalling) unless interrupted by the user; and it will play back with the highest possible quality given the average communication bandwidth available. To meet these expectations in the face of changing network conditions, buffering of the content at the client before decoding and playback is required.

[0004] Buffering at the client serves several distinct but simultaneous purposes. First, it allows the client to compensate for short-term variations in packet transmission delay (i.e., "jitter"). Second, it gives the client time to perform packet loss recovery if needed. Third, it allows the client to continue playing back the content during lapses in network bandwidth. And finally, it allows the content to be coded with variable bit rate, which can dramatically improve overall quality. Note that even so-called constant bit rate (CBR) coded content is actually coded with variable bit rate within the constraints of a decoding buffer of a given size. The larger the decoding buffer size, the better the quality. The required decoding buffering is part of the larger client buffer.

[0005] The size of the client buffer can be expressed as the number of seconds of content in the buffer, called the buffer *duration.* The buffer duration tends to increase as content enters the buffer and tends to decrease as content leaves the buffer. Content leaves the buffer when it is played out, at a rate of v seconds of content per second of real time, where v is the *playback speed* (typically 1 for normal playback, but possibly more than 1 for high speed playback or less than 1 for low speed playback). Content enters the buffer when it arrives at the client over the network, at a rate of r_a/r_c seconds of content per second of real time, where r_a is the *arrival rate,* or average number of bits that arrive at the client per second of real time, and r_c is the *coding bit rate,* or the average number of bits needed to encode one second of content. Thus the buffer duration can be increased by increasing r_a, decreasing r_c, and/or decreasing v (and vice versa for decreasing the buffer duration). Although the buffer duration can be momentarily controlled by changing r_a or changing v, these quantities are generally not possible to control freely for long periods of time. The arrival rate r_a on average is determined by the network capacity, while the playback speed v on average is determined by user preference. Thus if the network capacity drops dramatically for a sustained period, reducing the coding bit rate r_c is the only appropriate way to prevent a *rebuffering event* in which playback stops (v=0) while the buffer refills.

[0006] Thus, adaptivity to changing network conditions requires not only a buffer, but also some means to adjust the coding bit rate r_c of the content. This can be done by stream switching in combination with multi bit rate (MBR) coding or coarse grained or fine grained scalable coding. Today's commercial streaming media systems [1] rely on MBR coding as well as *thinning,* which is a form of coarse grained scalability. In MBR coding, semantically identical content is encoded into alternative bit streams at different coding bit rates and stored in the same media file at the server, allowing the content to be streamed at different levels of quality corresponding to the coding bit rates r_c, possibly using bit stream switching [2]. In coarse grained scalable coding (such as MPEG-2/4 temporal or SNR scalability) the content is encoded into several sub-streams or *layers,* so that the coding bit rate r_c can be changed in large deltas by adding or dropping (at possibly restricted times) one layer of content at a time. Thinning is a special case of coarse grained scalability in which dependent video frames (P and B frames) are dropped before independent video frames (I frames), which are in turn are dropped before audio frames. Future commercial systems may support fine grained scalability (FGS) as well. Fine grained scalable coding (such as 3D SPIHT [6], MPEG-4 FGS [7], or EAC [8]) allows the coding bit rate r_c to change

Ex. 1018-231

at any time in deltas sometimes as small as one byte per presentation. FGS coding offers great flexibility in adapting to variable network conditions, and can demonstrably improve quality under such conditions.

[0007] Some examples of existing technology that adjusts the coding bit rate r_c of the content in an attempt to adapt to changing network conditions includes de Cuetos and Ross [9], which decouples the transmission rate and the coding bit rate. They assume that the transmission rate is determined by the network transport protocol (TCP or TFRC). Based on this, they develop a heuristic real time algorithm for adaptive coding bit rate control and compare its performance to an optimal offline coding bit rate control policy if the transmission rate is given prior to streaming. The work of Rejaie, Handley and Estrin [4] proposes a scheme for transmitting layered video in the context of unicast congestion control, which basically includes two mechanisms. One mechanism is a coarse-grained mechanism for adding and dropping layers (changing the overall coding bit rate and quality). The other is a fine-grained interlayer bandwidth allocation mechanism to manage the receiver buffer (not changing the overall coding bit rate or quality). A potential issue with this approach is that it changes the coding bit rate by adding or dropping one (presumably coarse) layer at a time. If the layers are fine-grained, as in the case of FGS coded media, then adding or dropping one (fine-grained) layer at a time typically cannot provide a prompt enough change in coding bit rate. Moreover, since the adding and dropping mechanism is rather empirical, the mechanism may simply not be suitable for FGS media. The work of Q. Zhang, Zhu and Y-Q. Zhang [5] proposes a resource allocation scheme to adapt the coding bit rate to estimated network bandwidth. The novelty of their approach is that they consider minimizing the distortion (or equivalently maximizing the quality) of all applications, such as file-transfers and web browsing in addition to audio/video streaming. However, their optimization process does not include the smoothness of individual streams and might lead to potential quality fluctuations.

[0008] However, even with buffering and the ability to adjust the coding bit rate, existing technologies for streaming media on demand over the Internet suffer from two problems:

1. Playback often stalls during network congestion. That is, during playback of high bit rate content, if the network bit rate drops below the content bit rate, the client buffer runs out of content and playback stops while the client rebuffers (known as a "rebuffering" event).
2. Start-up delay is often too long (about 5 seconds).

There are existing solutions to both of these problems, but they do not always work well. One solution to the first problem is to stream the content encoded at a coding bit rate that is low relative to the average bit rate transmitted over the network (the *transmission bit rate*). This will enable the buffer to build up over time. With such a large reserve of unplayed information on the client, temporary network congestion will not affect playback. However, this solution has two problems. First, the coding bit rate of the content is not as high as the average transmission bit rate of the network and hence the quality is lower than it could be. Second, the buffer can grow nearly as large as the streamed file itself. This may demand too many resources on the client device.

[0009] Another solution to the first problem is to try to maintain the client buffer at a constant level (typically about 10 seconds), while switching between different coding bit rates for the same content, trying to match the transmission bit rate of the network. However, rebuffering events are still commonly observed in practice, because choosing the right time to switch streams is difficult. One reason that it is difficult is that there are natural variations in the instantaneous coding bit rate of the content, even in so-called constant bit rate encodings, which can confuse the client buffer management algorithm.

[0010] The second problem above (long start-up delay) also has multiple solutions. One solution is to fill up the client buffer quickly, with a quick initial transmission rate burst. With the client buffer full, playback can safely begin. However, this solution has several problems. First, it is only applicable when there is sufficient "headroom" in the network to increase the transmission bit rate for a few seconds. Thus it is usually not applicable for modem connections, for example. Second, it stresses the network, causing other applications in the network to back off. It has been shown that during the burst period, there can be as much as 80% packet loss, causing all TCP connections sharing the same bottleneck to back off. Third, by implication, if there is headroom in the network for bursting, then the streaming application may not be using the full bandwidth available to it during the remainder of the file, meaning that quality is lower than it should be.

[0011] Another solution to the second problem is to play back the content slower than real time, allowing playback to begin while the client buffer builds up. This is an innovative solution, but has the obvious temporal distortion.

[0012] A final solution to the second problem is to lower temporarily the coding bit rate of the content below the transmission bit rate of the network, allowing playback to begin while the client buffer builds up. This is a solution proposed by Chou et al. in [13].

[0013] The system and process of the present invention resolve the problems of the existing techniques and provide fast startup, continuous playback, and maximal quality and smoothness over the entire streaming session.

[0014] It is noted that in the preceding paragraphs, as well as in the remainder of this specification, the description refers to various individual publications identified by a numeric designator contained within a pair of brackets. For example, such a reference may be identified by reciting, "reference [1]" or simply "[1]". A listing of references including

Ex. 1018-232

the publications corresponding to each designator can be found at the end of the Detailed Description section.

SUMMARY

[0015] The present invention is directed toward a system and process for controlling a coding bit rate of streaming media data being transmitted to a client from a server over a computer network. In general, this coding bit rate control involves dynamically adjusting the coding bit rate of the streaming media data to control the client buffer duration. The purpose of this is to prevent the client buffer from underflowing, while keeping the average coding bit rate close to the average transmission bit rate of the network (an thus maximizing the quality of the playback of the data). The problem of coding bit rate control is formulated as a standard problem in linear quadratic optimal control, in which the client buffer duration is controlled as closely as possible to a target level. The smoothness of the average coding bit rate over consecutive frames is also considered when deciding whether to change the coding bit rate as part of the optimal control process. This yields a higher and more stable quality as network conditions change. In addition, the natural variation in the instantaneous coding bit rate that occurs for a given average coding bit rate is explicitly take into consideration. This is accomplished by incorporating the leaky bucket model into the control loop so that the changes in buffer duration due to natural variation in the instantaneous coding bit rate are not mistaken for changes in buffer duration due to network congestion. It is noted that in the present system and process, it is not the actual fullness of the client buffer that is controlled, but an upper bound on the time of arrival of bits into the client buffer, to a target level. The upper bound is based on the leaky bucket model of the coding bit rate.

[0016] Preventing the client buffer from underflowing, while keeping the average coding bit rate close to the average transmission bit rate of the network, meets two of the aforementioned user expectations. Namely, if the buffer never underflows, it allows for continuous playback. In addition, keeping the average coding bit rate close to the average transmission bit rate of the network means the data is played back with the highest possible quality given the average communication bandwidth available. This leaves the remaining issue of startup delay. This issue is resolved in the present system and process by controlling the size of the client buffer over time. More particularly, the aforementioned client buffer target level is adjusted to start small, and then grow slowly over time. If the buffer is initially small, it allows for shorter startup delay. In addition, as the buffer is eventually allowed to grow large, it enhances the robustness of the system as well as creating high, nearly constant quality. Thus, client buffer management is a key element affecting the performance of streaming media systems.

[0017] More particularly, the present system and process involves a server generating a streaming media data stream that exhibits one of a number of coding bit rates supported by the server. Initially, the server chooses the coding bit rate during a startup period. However, after the startup period, the client provides coding bit rate requests to the server. In response, the server transmits the streaming media data at the most appropriate supported coding bit rate closest to the rate requested by the client. The coding bit rates requested by the client are those estimated to provide a high quality playback of the streaming media data while still keeping a decoder buffer of the client used to receive streaming media data from the server filled to a desired duration level.

[0018] The client computes the coding bit rate that will provide the desired results on an ongoing basis using the aforementioned linear quadratic optimal control technique. This coding bit rate computation involves determining on a frame-by-frame basis a coding bit rate that reduces a difference between an estimated latest anticipated arrival time of the frame under consideration and a prescribed target arrival time, while at the same time reducing the change in the coding bit rate to a prescribed degree. In one embodiment involving the aforementioned leaky bucket model, the coding bit rate computations are based on parameters indicative of the state of the encoder buffer of the server. These parameters are computed by the server and provided to the client along with the streaming media data.

[0019] More particularly, the server first computes a set of parameters defining an initial state of the encoder buffer as it would exist if a data stream corresponding to a supported coding bit rate where streamed therefrom. A separate set of parameters is computed for each of the supported coding bit rates, and are provided to the client in a preamble to the streaming media data. These parameters include the coding bit rate associated with the data stream, the size of the encoder buffer employed with the coding bit rate of the data stream, and a value indicative of the initial encoder buffer fullness exhibited at the coding bit rate of the data stream. It is noted that the size of the encoder buffer employed with each supported coding bit rate varies and is chosen so as to be the minimum size buffer that will still contain the data stream at any point in the streaming process given the coding bit rate and the initial encoder buffer fullness.

[0020] In addition to the initial encoder buffer parameters, the server also computes an upper bound gap for each frame of the streaming media data generated for each coding bit rate supported by the server. This upper bound gap is defined as the number of data bits that the server's encoder buffer can contain over the bits currently contained therein after a just-generated frame is fully input into the buffer. More particularly, the upper bound gap for each frame is computed as the difference between the encoder buffer size and the last computed value for an encoder buffer fullness after insertion of the just-generated frame. The encoder buffer fullness value after insertion of the just-generated frame is computed as the sum of the last computed value of the encoder buffer fullness value prior to insertion of the just-

Ex. 1018-233

generated frame and the size of that just-generated frame. The encoder buffer fullness value prior to insertion of the just-generated frame is computed as either zero, or the difference between the value computed for the encoder buffer fullness after the insertion of the frame generated immediately before the just-generated frame and the coding bit rate associated with that prior frame divided by the instantaneous frame rate associated with said prior frame, whichever is larger. The instantaneous frame rate is equal to the reciprocal of, the time the next frame is scheduled to be encoded less the time the just-generated frame was encoded. In one embodiment, the server provides the upper bound gap computed for the first frame of a sequence of frames generated after a change in the coding rate to the client along with an indication of the coding bit rate associated with the sequence of frames as part of the data associated with the first frame of the sequence. In another embodiment, the server provides the gap value with every frame generated.

[0021] The client employs the encoder buffer parameters and the upper bound gap values to determine on a frame-by-frame basis the coding bit rate that will reduce a difference between the estimated latest anticipated arrival time of a frame under consideration and a prescribed target arrival time of that frame, in order to keep the client buffer at approximately the desired duration level. In cases where the upper bound gap is only provided with the first frame after a change in the coding bit rate, the client estimates the upper bound gap for the frames that do not include a gap value. It is also noted that given the aforementioned initial encoder buffer conditions, it is possible for the client to estimate the upper bound gap values for every frame received. Thus, an alternate embodiment where the server is not configured to compute and provide the gap values on an ongoing basis, the client computer can still compute them on its own.

[0022] When a frame with a new average coding bit rate arrives at the client there is a shift in the upper bound gap. This shift can be on the order of seconds and hence, rather than being negligible, can be confusing to the controller. One solution to this is to introduce a simultaneous shift in the control target schedule. To this end, the server also computes a shift value for each frame representing the first frame after a coding bit rate change. This shift value represents the difference between the upper bound gap that would be associated with the frame generated immediately before this first frame had it been encoded at the new coding bit rate and the upper bound gap actually associated with the frame as encoded at the previous coding bit rate. The shift value is provided to the client along with the first frame after a change in the coding bit rate. The client shifts the currently scheduled target arrival time associated with the just received "first" frame by the shift value provided, and the currently scheduled target arrival times for future frames are shifted such that they collectively approach, over a period of time, and eventually coincide with, the previous target arrival times for those frames. In this way the adverse effects of an upper bound gap shift when the coding bit rate is changed are mitigated and the target arrival time values are eventually brought back in line.

[0023] In regard to the target arrival time for each frame, these are chosen so as to make the amount of time between the target time of a frame and its playback time large enough, after the startup period, that network jitter, delays and throughput changes which may cause the actual arrival time of the frame to be after its target arrival time do not result in the frame arriving after its scheduled playback time. During the startup period, the target arrival times are chosen so as to make the target arrival time closer to the playback time to assist in reducing startup delay. In one embodiment this is accomplished by setting the target arrival time for each frame using a logarithmic target schedule. In another embodiment this is accomplished by setting the target arrival time for each frame using a two-piece linear target schedule where the difference between the target arrival time and the playback time for a frame arriving during the startup period increases linearly to a prescribed amount of time that is large enough to account for the network jitter, delays and throughput changes and after which the difference remains substantially constant.

[0024] It is noted that the startup delay can be further minimized by beginning playback of the first frame of the streaming media data when the client's decoder buffer has the minimum amount of data therein that is needed to ensure an underflow condition does not occur at the initial coding bit rate. Still further, the initial coding bit rate can be set to a level that is less than the anticipated arrival rate of the streaming media data at the client since the minimum amount of data needed to ensure an underflow condition does not occur, decreases as the arrival rate increases relative to the coding bit rate. Thus, in the startup period before the client request changes in the coding bit rate, the server provides the streaming media data at the initial coding bit rate. In one embodiment, the initial coding bit rate is set to approximately one-half the anticipated arrival rate, where the anticipated arrival rate is estimated to be the anticipated transmission rate from the server at the current bandwidth available on the network. It is noted that the startup period can be defined as the period of time prior to the first instance of an estimated latest arrival time of a frame under consideration computed for the initial coding bit rate being earlier than a target arrival time for that frame.

[0025] In regard to the client identifying the coding bit rate for each frame that also reduces the change in the coding bit rate to a prescribed degree, it is noted that the prescribed degree can vary depending on whether the coding bit rate increased. If it did increase, then the prescribed degree to which any future change in the coding bit rate is minimized is made greater in comparison to the prescribed degree in cases where the last change in the coding bit rate decreased the rate. Still further, it is also desirable to minimize quality variations due to large or frequent changes to the coding bit rate. In order to stabilize the code bit rate changes, the following actions can be taken. First, in the case where the client identifies a new coding bit rate that represents an increase, a new coding bit rate would be requested from the server only if the new rate does not exceed the current moving average arrival rate of the frames. In another embodiment, the

Ex. 1018-234

new coding bit rate representing an increase would only be requested from the server, even if it exceeds the current moving average arrival rate of the frames, if the current client buffer duration exceeds the desired duration level by an amount that it is estimated will not be expended at the higher coding bit rate prior to the passing of a prescribed period of time (e.g., 60 seconds). However, in a case where the client identifies a new coding bit rate that represents a decrease, a new coding bit rate would be requested from the server even if it represents a significant departure from the immediately previous coding bit rate. This is because there is little risk in the client buffer underflowing when the coding rate is decreased.

[0026] In regard to the aforementioned moving average arrival rate of the frames, a new procedure for computing this rate is employed with the present system and process. More particularly, the moving average arrival rate is computed on a packet-by-packet basis, by calculating the product of the moving average arrival rate computed for the immediately preceding packet to a currently received packet and a fractional weighting factor, added to the product of the instantaneous arrival rate of the currently received packet and one minus the fractional weighting factor. In this computation, the fractional weighting factor is not a constant as in the past, but instead is based on the inter-arrival gaps between received packets. More particularly, the fractional weighting factor $\beta(k)$ for a currently received packet k is computed as $\frac{e^{-\alpha[t(k)-t(k-1)]}-e^{-\alpha[t(k)-t(0)]}}{1-e^{-\alpha[t(k)-t(0)]}}$ where α is the reciprocal of a prescribed time constant, $t(k)$ is the actual arrival time of the current packet, $t(k\text{-}1)$ is the actual arrival time of the packet received immediately prior to the current packet, and $t(0)$ is the arrival time of the first packet of the streaming media data. In this case, the instantaneous arrival rate $r_a(k)$ of the current packet k is computed as $\frac{b(k)}{t_a(k)-t_a(k-1)}$, where $b(k)$ is the size of the current packet. If $b(k)$ is expressed in bits and $t_a(k) - t_a(k - 1)$ is expressed in seconds, then the moving average arrival rate represents the arrival rate of the streaming media data bit bits per second. It is noted that a similar procedure can be employed for any arbitrary units of rate of the streaming media.

[0027] As mentioned previously, the server generates data streams that exhibit one of a number of coding bit rates supported by the server, and transmits data streams at the most appropriate supported coding bit rate closest to the rate requested by the client. The number of available coding bit rates will depend on how the data is encoded. For example, if a fine grain scalable coding scheme is used theoretically there could be a large number of rates available (although for practical reasons the number is more likely to be lower, e.g., 50). However, if a coarse grain scalable coding scheme or a multiple bit rate coding scheme is employed, there could be a more limited number of coding bit rates available from the server. Thus, in some cases an optimum coding bit rate identified by the client may not be available from the server. In addition, even if there is a matching coding bit rate available, the upper bound gap may be such that switching to that rate would risk a client buffer underflow. Given this, the present system and process includes techniques for determining the most appropriate coding bit rate, from the available rates, in view of the optimum coding bit rate identified by the client. It is noted that the available rates could be provided to the client from the server in a preamble to the streaming media data. In such a case, the client itself would perform the analysis and request the resulting supported coding bit rate from the server. However, in an alternate embodiment, the client would request the optimum coding bit rate it identified, and the server would perform the analysis to determine which supported rate is most appropriate. The server would then provide a data stream exhibiting the selected coding bit rate. In either case, the analysis involves finding a supported coding bit rate that is equal to, or if none are equal, the closest smaller rate to, the optimum coding bit rate identified by the client. Whenever the supported coding bit rate found is lower than the coding bit rate associated with the last generated frame of the streaming media data, all future frames (or those beginning with a frame specified by the client) are generated at that supported rate. However, when the supported coding bit rate found is higher than the coding bit rate associated with the last generated frame, it is determined if a difference between the upper bound gap associated with the last generated frame had it been encoded at the supported coding bit rate found and the upper bound gap associated with that frame encoded at the current coding bit rate, is less than or equal to a maximum allowable difference value. If the difference is less than the maximum allowable difference value, the affected future frames are generated at the supported rate found. But, if the difference is not less than the maximum allowable difference value, the next lower supported coding bit rate is found and the foregoing actions are repeated to ultimately identify the appropriate rate.

[0028] The maximum allowable difference value is computed by the client, and if the server is doing the analysis, the client provides the value along with the request for data at the new coding bit rate. The client chooses the maximum allowable difference value such that the latest anticipated arrival time associated with a frame provided by the server immediately prior to a frame under consideration by the client, had it been coded at the requested coding bit rate, is no more than a prescribed fraction of the way from that frame's target arrival time to its playback deadline (e.g., 1/3 of the way).

[0029] In addition to the just described benefits, other advantages of the present invention will become apparent from

Ex. 1018-235

the detailed description which follows hereinafter when taken in conjunction with the drawing Fig.s which accompany it.

DESCRIPTION OF THE DRAWINGS

[0030] The specific features, aspects, and advantages of the present invention will become better understood with regard to the following description, appended claims, and accompanying drawings where:

FIG. 1 is a diagram depicting a general purpose computing device constituting an exemplary system for implementing the present invention.

FIG. 2 is block diagram of a simplified streaming media communication pipeline.

FIG. 3 is a graph showing schedules at which bits in the coded bit stream pass points A, B, C and D in the communication pipeline of Fig. 2 in terms of bits vs. media time.

FIG. 4 is a graph showing a buffer tube containing a coding schedule in terms of bits vs. media time.

FIG. 5 is a graph showing an arrival schedule and its upper bound in client time vs. media time, where the upper bound is controlled to a target schedule that is increasingly in advance of the playback deadline to provide greater robustness over time.

FIG. 6 is a graph showing a target arrival schedule design.

FIGS. 7(a)-(b) are graphs showing two different target arrival schedules. Fig. 7(a) shows a logarithmic schedule and Fig. 7(b) shows a two-piece linear schedule.

FIG. 8 is a graph showing buffer tubes for various transmission rates.

FIG. 9 is a graph illustrating exponential averaging.

FIG. 10 is a graph showing buffer tube changes and control target adjustments.

FIGS. 11A-C show a flow chart diagramming an embodiment of the coding bit rate control process according to the present invention.

FIG. 12 is a timeline illustrating the conservative limit used in the conservative coding bit rate up-switching procedure.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

[0031] In the following description of the preferred embodiments of the present invention, reference is made to the accompanying drawings which form a part hereof, and in which is shown by way of illustration specific embodiments in which the invention may be practiced. It is understood that other embodiments may be utilized and structural changes may be made without departing from the scope of the present invention.

I. The Computing Environment

[0032] Before providing a description of the preferred embodiments of the present invention, a brief, general description of a suitable computing environment in which portions of the invention may be implemented will be described. Fig. 1 illustrates an example of a suitable computing system environment 100. The computing system environment 100 is only one example of a suitable computing environment and is not intended to suggest any limitation as to the scope of use or functionality of the invention. Neither should the computing environment 100 be interpreted as having any dependency or requirement relating to any one or combination of components illustrated in the exemplary operating environment 100.

[0033] The invention is operational with numerous other general purpose or special purpose computing system environments or configurations. Examples of well known computing systems, environments, and/or configurations that may be suitable for use with the invention include, but are not limited to, personal computers, server computers, hand-held or laptop devices, multiprocessor systems, microprocessor-based systems, set top boxes, programmable consumer electronics, network PCs, minicomputers, mainframe computers, distributed computing environments that include any of the above systems or devices, and the like.

[0034] The invention may be described in the general context of computer-executable instructions, such as program modules, being executed by a computer. Generally, program modules include routines, programs, objects, components, data structures, etc. that perform particular tasks or implement particular abstract data types. The invention may also be practiced in distributed computing environments where tasks are performed by remote processing devices that are linked through a communications network. In a distributed computing environment, program modules may be located in both local and remote computer storage media including memory storage devices.

[0035] With reference to Fig. 1, an exemplary system for implementing the invention includes a general purpose computing device in the form of a computer 110. Components of computer 110 may include, but are not limited to, a processing unit 120, a system memory 130, and a system bus 121 that couples various system components including the system memory to the processing unit 120. The system bus 121 may be any of several types of bus structures including a memory bus or memory controller, a peripheral bus, and a local bus using any of a variety of bus architectures. By way of example, and not limitation, such architectures include Industry Standard Architecture (ISA) bus, Micro Channel Architecture (MCA) bus, Enhanced ISA (EISA) bus, Video Electronics Standards Association (VESA) local bus, and Peripheral Component Interconnect (PCI) bus also known as Mezzanine bus.

[0036] Computer 110 typically includes a variety of computer readable media. Computer readable media can be any available media that can be accessed by computer 110 and includes both volatile and nonvolatile media, removable and non-removable media. By way of example, and not limitation, computer readable media may comprise computer storage media and communication media. Computer storage media includes both volatile and nonvolatile, removable and non-removable media implemented in any method or technology for storage of information such as computer readable instructions, data structures, program modules or other data. Computer storage media includes, but is not limited to, RAM, ROM, EEPROM, flash memory or other memory technology, CD-ROM, digital versatile disks (DVD) or other optical disk storage, magnetic cassettes, magnetic tape, magnetic disk storage or other magnetic storage devices, or any other medium which can be used to store the desired information and which can be accessed by computer 110. Communication media typically embodies computer readable instructions, data structures, program modules or other data in a modulated data signal such as a carrier wave or other transport mechanism and includes any information delivery media. The term "modulated data signal" means a signal that has one or more of its characteristics set or changed in such a manner as to encode information in the signal. By way of example, and not limitation, communication media includes wired media such as a wired network or direct-wired connection, and wireless media such as acoustic, RF, infrared and other wireless media. Combinations of the any of the above should also be included within the scope of computer readable media.

[0037] The system memory 130 includes computer storage media in the form of volatile and/or nonvolatile memory such as read only memory (ROM) 131 and random access memory (RAM) 132. A basic input/output system 133 (BIOS), containing the basic routines that help to transfer information between elements within computer 110, such as during start-up, is typically stored in ROM 131. RAM 132 typically contains data and/or program modules that are immediately accessible to and/or presently being operated on by processing unit 120. By way of example, and not limitation, Fig. 1 illustrates operating system 134, application programs 135, other program modules 136, and program data 137.

[0038] The computer 110 may also include other removable/non-removable, volatile/nonvolatile computer storage media. By way of example only, Fig. 1 illustrates a hard disk drive 141 that reads from or writes to non-removable, nonvolatile magnetic media, a magnetic disk drive 151 that reads from or writes to a removable, nonvolatile magnetic disk 152, and an optical disk drive 155 that reads from or writes to a removable, nonvolatile optical disk 156 such as a CD ROM or other optical media. Other removable/non-removable, volatile/nonvolatile computer storage media that can be used in the exemplary operating environment include, but are not limited to, magnetic tape cassettes, flash memory cards, digital versatile disks, digital video tape, solid state RAM, solid state ROM, and the like. The hard disk drive 141 is typically connected to the system bus 121 through a non-removable memory interface such as interface 140, and magnetic disk drive 151 and optical disk drive 155 are typically connected to the system bus 121 by a removable memory interface, such as interface 150.

[0039] The drives and their associated computer storage media discussed above and illustrated in Fig. 1, provide storage of computer readable instructions, data structures, program modules and other data for the computer 110. In Fig. 1, for example, hard disk drive 141 is illustrated as storing operating system 144, application programs 145, other program modules 146, and program data 147. Note that these components can either be the same as or different from operating system 134, application programs 135, other program modules 136, and program data 137. Operating system 144, application programs 145, other program modules 146, and program data 147 are given different numbers here to illustrate that, at a minimum, they are different copies. A user may enter commands and information into the computer 110 through input devices such as a keyboard 162 and pointing device 161, commonly referred to as a mouse, trackball or touch pad. Other input devices (not shown) may include a microphone, joystick, game pad, satellite dish, scanner, or the like. These and other input devices are often connected to the processing unit 120 through a user input interface 160 that is coupled to the system bus 121, but may be connected by other interface and bus structures, such as a parallel port, game port or a universal serial bus (USB). A monitor 191 or other type of display device is also connected to the

system bus 121 via an interface, such as a video interface 190. In addition to the monitor, computers may also include other peripheral output devices such as speakers 197 and printer 196, which may be connected through an output peripheral interface 195. A camera 192 (such as a digital/electronic still or video camera, or film/photographic scanner) capable of capturing a sequence of images 193 can also be included as an input device to the personal computer 110. Further, while just one camera is depicted, multiple cameras could be included as input devices to the personal computer 110. The images 193 from the one or more cameras are input into the computer 110 via an appropriate camera interface 194. This interface 194 is connected to the system bus 121, thereby allowing the images to be routed to and stored in the RAM 132, or one of the other data storage devices associated with the computer 110. However, it is noted that image data can be input into the computer 110 from any of the aforementioned computer-readable media as well, without requiring the use of the camera 192.

[0040] The computer 110 may operate in a networked environment using logical connections to one or more remote computers, such as a remote computer 180. The remote computer 180 may be a personal computer, a server, a router, a network PC, a peer device or other common network node, and typically includes many or all of the elements described above relative to the computer 110, although only a memory storage device 181 has been illustrated in Fig. 1. The logical connections depicted in Fig. 1 include a local area network (LAN) 171 and a wide area network (WAN) 173, but may also include other networks. Such networking environments are commonplace in offices, enterprise-wide computer networks, intranets and the Internet.

[0041] When used in a LAN networking environment, the computer 110 is connected to the LAN 171 through a network interface or adapter 170. When used in a WAN networking environment, the computer 110 typically includes a modem 172 or other means for establishing communications over the WAN 173, such as the Internet. The modem 172, which may be internal or external, may be connected to the system bus 121 via the user input interface 160, or other appropriate mechanism. In a networked environment, program modules depicted relative to the computer 110, or portions thereof, may be stored in the remote memory storage device. By way of example, and not limitation, Fig. 1 illustrates remote application programs 185 as residing on memory device 181. It will be appreciated that the network connections shown are exemplary and other means of establishing a communications link between the computers may be used.

[0042] The exemplary operating environment having now been discussed, the remaining parts of this description section will be devoted to a description of the program modules embodying the invention.

II. Problem Formulation

A. Temporal Coordinate Systems

[0043] It will pay to distinguish between the temporal coordinate systems, or clocks, used to express time. Herein, *media time* refers to the clock running on the device used to capture and timestamp the original content, while *client time* refers to the clock running on the client used to play back the content. It is assumed that media time is real time (i.e., one second of media time elapses in one second of real time) at the time of media capture, while client time is real time at the time of media playback. The symbol τ is used to express media time and the symbol t to express client time, with subscripts and other arguments to indicate corresponding events. For example, $\tau_d(0),\tau_d(1),\tau_d(2),\ldots$ is used to express the playback deadlines of frames 0,1,2,... in media time, while $t_d(0), t_d(1), t_d(2),..$ is used to express the playback deadlines of frames 0,1,2,... at the client. Content may be played back at a rate v times real time. Thus the conversion from media time to client time can be expressed as

$$t = t_0 + \frac{\tau - \tau_0}{v}, \qquad (1)$$

where t_0 and τ_0 represent the time of a common initial event, such as the playback of frame 0 (or the playback of the first frame after a seek or rebuffering event) in media and client coordinate systems, respectively.

B. Leaky Bucket Model

[0044] For the moment imagine a scenario in which both the encoder 200 and the decoder 202 run in real time over an isochronous communication channel of a network 204. In this case, to match the instantaneous coding bit rate to the instantaneous channel rate, an *encoder buffer* 206 is required between the encoder 200 and the network 204 and a *decoder buffer* 208 is required between the network 204 and the decoder 202 , as illustrated in Fig. 2. A *schedule* is the sequence of times at which successive bits in the coded bit stream pass a given point in the communication pipeline.

The graph in Fig. 3 illustrates the schedules of bits passing the points A, B, C, and D in Fig. 2. Schedule A is the schedule at which captured frames are instantaneously encoded and put into the encoder buffer. This schedule is a staircase in which the nth step rises by $b(n)$ bits at time $\tau(n)$, where $\tau(n)$ is the time at which frame n is encoded, and $b(n)$ is the number of bits in the resulting encoding. Schedules B and C are the schedules at which bits respectively enter and leave the communication channel. The slope of these schedules is R bits per second, where R is the communication rate of the channel. Schedule D is the schedule at which frames are removed from the decoder buffer and instantaneously decoded for presentation. Note that Schedule D is simply a shift of Schedule A. Note also that Schedule B is a lower bound to Schedule A, while Schedule C is an upper bound to Schedule D. Indeed, the gap between Schedules A and B represents, at any point in time, the size in bits of the encoder buffer, while the gap between Schedules C and D likewise represents the size of the decoder buffer. The encoder and decoder buffer sizes are complementary. Thus the coding schedule (either A or D) can be contained within a *buffer tube,* as illustrated in the graph of Fig. 4, having slope R , height B , and initial offset F^d from the top of the tube (or equivalently initial offset $F^e = B - F^d$ from the bottom of the tube). It can be seen that $D = F^d/R$ is the *startup delay* between the time that the first bit arrives at the receiver and the first frame is decoded. Thus it is of interest to minimize F^d for a given R.

[0045] A *leaky bucket* is a metaphor for the encoder buffer. The encoder dumps $b(n)$ bits into the leaky bucket at time $\tau(n)$, and the bits leak out at rate R. In general it is possible for the leak rate R to be high enough so that the bucket occasionally empties. Thus the encoder buffer fullness $F^e(n)$ immediately before frame n is added to the bucket and the encoder buffer fullness $B^e(n)$ immediately after frame n is added to the bucket evolve from an initial encoder buffer fullness $F^e(0) = F^e$ according to the dynamical system

$$B^e(n) = F^e(n) + b(n), \tag{2}$$

$$F^e(n+1) = \max\{0, B^e(n) - R/f(n)\}, \tag{3}$$

where

$$f(n) = \frac{1}{\tau(n+1) - \tau(n)} \tag{4}$$

is the instantaneous frame rate, for n=0,1,2,... If R is sufficiently low, then the bucket will never run dry (underflow), but if R is too low the bucket will eventually overflow. The largest R such that the buffer will never run dry is taken to be the average coding bit rate r_c of the bit stream. This is made more precise in the following two paragraphs.

[0046] A leaky bucket with size B, rate R, and initial fullness F^e is said to *contain* a stream having a schedule characterized by the steps $\{(b(n),\tau(n))\}$ if $B^e(n) \le B$ for all n. The minimum bucket size needed to contain the stream given leak rate R and initial fullness F^e is defined as

$$B^e_{\min}(R, F^e) = \min_n B^e(n), \tag{5}$$

while the corresponding initial decoder buffer fullness is defined as

$$F^d_{\min}(R, F^e) = B^e_{\min}(R, F^e) - F^e. \tag{6}$$

The minimum of each of these over F^e is denoted as

$$B^e_{\min}(R) = \min_{F^e} B^e_{\min}(R, F^e), \tag{7}$$

$$F^d_{\min}(R) = \min_{F^e} F^d_{\min}(R, F^e). \tag{8}$$

It is shown in [10, Proposition 2] that remarkably; these are each minimized by the same value of F^e, which is hence equal to

$$F^e_{\min}(R) = B^e_{\min}(R) - F^d_{\min}(R). \tag{9}$$

Thus the bit stream with schedule $\{(b(n),\tau(n))\}$, for each bit rate R there is a unique leaky bucket that contains the stream and that has the minimum buffer size B as well as the minimum startup delay $D = F^d/R$. These parameters can be computed with the above equations.

[0047] For sufficiently low leak rates R, the leaky bucket does not underflow, when beginning with initial fullness $F^e = F^e_{\min}(R)$. Tthe maximum such rate R may be used as the average coding bit rate r_c of a bit stream with coding schedule $\{(b(n),\tau(n))\}$.

[0048] Leak rates R greater than r_c can also be employed. It is shown in [10] that both $B^e_{\min}(R)$ and $F^d_{\min}(R)$ are decreasing, piecewise linear, and convex in R. Hence if the transmission rate R is greater than the average coding bit rate r_c, the startup delay $D = F^d_{\min}(R)/R$ can be reduced compared to $D = F^d_{\min}(r_c)/R$. This fact will be used in Section IV-A.

[0049] A leaky bucket with leak rate $R=r_c$, size $B = B^e_{\min}(r_c)$ and initial decoder buffer fullness $F^d = F^d_{\min}(r_c)$ thus corresponds to a straight buffer tube bounding the coding schedule as in Fig. 4. Each stream in the media file has a coding schedule; thus each stream corresponds to a straight buffer tube with slope equal to the average coding bit rate r_c of the stream. The size B of the buffer tube and its offset F^e (or F^d) relative to the coding schedule can be either computed by the above formula for a variable bit rate (VBR) stream (such as a constant-quality sub-stream of a scalable stream), or obtained from the size B and initial state F^e of the actual encoder buffer used to encode the stream if it is a constant bit rate (CBR) stream.

[0050] In the following the gap $g(n)$ at frame n between the buffer tube *upper bound* and the coding schedule is considered, as depicted in Fig. 4. Note that the decoder buffer fullness $F^d(n) = B - F^e(n)$ can also be expressed

$$F^d(n) = b(n) + g(n) = g(n-1) + \frac{r_c(n)}{f(n)}, \tag{10}$$

where $r_c(n)$ is the coding bit rate of the buffer tube, now taking into account that different frames may lie in different buffer tubes with different coding bit rates as coding bit rate control is applied and streams are switched.

C. Rate Control Model

[0051] Assume for the moment that bits arrive at the client at a constant rate r_a. Then frame n (having size $b(n)$) arrives at the client $b(n)/r_a$ seconds after frame n-1. Indeed, the index of a bit is proportional to its arrival time. Dividing the vertical scale of the schedules in Fig. 4 by r_a, the schedules are obtained in terms of client time, rather than bits, as shown in the graph of Fig. 5. The coding schedule divided by r_a becomes the *arrival schedule,* which provides for each n the time $t_a(n)$ of arrival of frame n at the client. The buffer tube upper bound (in bits) divided by r_a becomes the buffer tube upper bound (in time), which provides for each n the time $t_b(n)$ by which frame n is guaranteed to arrive. In the

Ex. 1018-240

same plot the *playback deadline* is shown, which is the time $t_d(n)$ at which frame n is scheduled to be played (after instantaneous decoding). Thus the gap between a frame's arrival time and its playback deadline is the client buffer duration at the time of the frame arrival. This must be non-negative to allow continuous playback.

[0052] In reality the arrival rate is not constant. If $t_a(n\text{-}1)$ and $t_a(n)$ are the arrival times of frames n and n-1 respectively, then

$$r_a(n) = \frac{b(n)}{t_a(n) - t_a(n-1)} \tag{11}$$

can be defined as the *instantaneous arrival rate* at frame n. In practice the average arrival rate at frame n is estimated by a moving average $\tilde{r}_a(n)$ of previous values of $r_a(n)$, as detailed in Section IV-C. Hence using Eq. (11) the arrival time of frame n can be expressed in terms of the arrival time of frame n-1 as

$$t_a(n) = t_a(n-1) + \frac{b(n)}{r_a(n)} \tag{12}$$

$$= t_a(n-1) + \frac{b(n)}{\tilde{r}_a(n)} + v(n), \tag{13}$$

where the $v(n)$ term is an error term that captures the effect of using the slowly moving average $\tilde{r}_a(n)$. From Eq. (10), however, it can be seen that

$$b(n) = \frac{r_c(n)}{f(n)} + g(n-1) - g(n), \tag{14}$$

whence (substituting Eq. (14) into Eq. (13)) yields

$$t_a(n) = t_a(n-1) + \frac{r_c(n)}{f(n)\tilde{r}_a(n)} + \frac{g(n-1)}{\tilde{r}_a(n)} - \frac{g(n)}{\tilde{r}_a(n)} + v(n). \tag{15}$$

Now defining the buffer tube upper bound (in time) of frame n as

$$t_b(n) = t_a(n) + \frac{g(n)}{\tilde{r}_a(n)}, \tag{16}$$

so that

$$t_b(n) - t_b(n-1) = t_a(n) - t_a(n-1) + \frac{g(n)}{\tilde{r}_a(n)} - \frac{g(n-1)}{\tilde{r}_a(n-1)}, \qquad (17)$$

the following update equation is obtained:

$$t_b(n) = t_b(n-1) + \frac{r_c(n)}{f(n)\tilde{r}_a(n)} + w(n-1), \qquad (18)$$

where

$$w(n-1) = \frac{g(n-1)}{\tilde{r}_a(n)} - \frac{g(n-1)}{\tilde{r}_a(n-1)} + v(n). \qquad (19)$$

is again an error term that captures variations around a locally constant arrival rate.

[0053] Using Eq. (16), the client can compute $t_b(n\text{-}1)$ from the measured arrival time $t_a(n - 1)$, the estimated arrival rate $\tilde{r}_a(n - 1)$, and $g(n - 1)$ (which can be transmitted to the client along with the data in frame n-1 or computed at the client as described in Section V-E). Then using Eq. (18), the client can control the coding bit rate $r_c(n)$ so that $t_b(n)$ reaches a desired value, assuming the frame rate and arrival rate remain roughly constant. From this perspective, Eq. (18) can be regarded as the state transition equation of a feedback control system and it is thus possible to use a control-theoretic approach to regulate the coding bit rate.

D. Control Objective

[0054] With the state transition equation defined in Eq. (18), uninterrupted playback can be achieved by regulating the coding bit rate so that the client buffer does not underflow. To introduce a margin of safety that increases over time, a *target schedule* is introduced, as illustrated in Fig. 5, whose distance from the playback deadline grows slowly over time. By regulating the coding bit rate, it is attempted to control the buffer tube upper bound so that it tracks the target schedule. If the buffer tube upper bound is close to the target schedule, then the arrival times of all frames will certainly be earlier than their playback deadlines and thus uninterrupted playback will be ensured. Note that controlling the actual arrival times (rather than their upper bounds) to the target would result in an approximately constant number of bits per frame, which would in turn result in very poor quality overall. By taking the leaky bucket model into account, it is possible to establish a control that allows the instantaneous coding bit rate to fluctuate naturally according to the encoding complexity of the content, within previously established bounds for a given average coding bit rate.

[0055] Although controlling the upper bound to the target schedule is our primary goal, it is also desirable to minimize quality variations due to large or frequent changes to the coding bit rate. This can be achieved by introducing into the cost function a penalty for relative coding bit rate differences.

[0056] Letting $t_T(n)$ denote the target for frame n, the following cost function is used to reflect both of concerns:

$$I = \sum_{n=0}^{N} \left((t_b(n) - t_T(n))^2 + \sigma \left(\frac{r_c(n+1) - r_c(n)}{\tilde{r}_a(n)} \right)^2 \right), \qquad (20)$$

where the first term penalizes the deviation of the buffer tube upper bound from the target schedule and the second term penalizes the relative coding bit rate difference between successive frames. N is the control window size and σ is a Lagrange multiplier or weighting parameter to balance the two terms.

III. Optimal Control Solution

[0057] Before presenting the optimal control solution, the design rational of the target schedule will be described.

A. Target Schedule Design

[0058] Fig. 6 is a graph showing an illustrative target schedule. The gap between the playback deadline and the target schedule is the desired client buffer duration (in client time). If the gap is small at the beginning of streaming, then it allows a small startup delay, while if the gap grows slowly over time, it gradually increases the receiver's ability to counter jitter, delays, and throughput changes.

[0059] The slope of the target schedule relates the average coding bit rate to the average arrival rate. Let $t_T(n)$ be the target for frame n. As illustrated in Fig. 6, the slope of the target schedule at frame n is

$$s(n) = \frac{t_T(n+1) - t_T(n)}{\tau(n+1) - \tau(n)}. \qquad (21)$$

If the upper bound $t_b(n)$ aligns perfectly with the target schedule (i.e., $t_b(n) = t_T(n)$) and the arrival rate r_a is constant (i.e., the $w(n\text{-}1)$ term vanishes), from Eq. (18),

$$s(n) = \frac{t_b(n+1) - t_b(n)}{\tau(n+1) - \tau(n)} = \frac{r_c(n)}{r_a}. \qquad (22)$$

Thus initially, when the slope was low, i.e., less than $1/v$, r_a/r_c is greater than v and more than v seconds of content are received per second of client time, causing the client buffer (which is playing out only v seconds of content per second of client time) to grow. Over time, as the slope approaches $1/v$, r_a/r_c approaches v and the buffer remains relatively constant (except for changes due to variations in the instantaneous coding bit rate), since content is received and played back at the same speed v. Next two target schedule functions are presented that illustrate the general design idea.

1) Logarithmic Target Schedule

[0060] One way to choose the target schedule t_T is to have the client buffer duration grow logarithmically over time. Specifically, if t_d is the playback deadline, then for each t_d greater than some start time t_{d0},

$$t_T = t_d - \frac{b}{a}\ln(a(t_d - t_{d0}) + 1). \qquad (23)$$

Since by Eq. (1), $t_d = t_{d0} + (\tau_d - \tau_{d0})/v$ this yields

$$s = \frac{dt_T}{d\tau_d} = \frac{dt_T dt_d}{dt_d d\tau_d} = \frac{1}{v} - \frac{b}{a(\tau_d - \tau_{d0}) + v}, \qquad (24)$$

and hence the initial slope at frame 0 (when $t_d = t_{d0}$) is $s(0) = (1\text{-}b)/v$. Setting b=0.5 implies that initially $r_c/r_a = 0.5/v$, causing the client buffer to grow initially at two times real time. Further setting a=0.15 implies that the client buffer duration will be 7.68 seconds after 1 minute, 15.04 seconds after 10 minutes, and 22.68 seconds after 100 minutes, regardless of v.

2) Two-piece Linear Target Schedule

[0061] Another way to choose the target schedule t_T is to have the client buffer duration grow linearly at rate b seconds of media time per second of client time until the buffer duration reaches a seconds of media time, after which it remains constant. Specifically, for each t_d greater than some start time t_{d0},

$$t_T = \begin{cases} t_d - b(t_d - t_{d0}) & t_d \le t_{d0} + a/b \\ t_d - a & t_d \ge t_{d0} + a/b \end{cases} \qquad (25)$$

The initial slope is again $s(0)=(1-b)/v$. Setting b=0.5 implies that initially r_c/r_a = 05/v, causing the client buffer to grow initially at two times real time. Further setting a = 10 implies that the client buffer duration will reach 10 seconds of media time after 20 seconds of client time, regardless of v.

[0062] Figs. 7(a) and (b) show graphs of the above two target schedules. As one can see, if a client buffer duration of 10 seconds is considered to be a safe level against jitter, delay and network fluctuations, then the two-piece linear target schedule reaches the safe level in 20 seconds, much faster than the logarithmic target schedule. On the other hand, the slope of the two-piece linear target schedule remains lower for longer (hence the coding bit rate and quality are lower for longer) and furthermore experiences an abrupt change at 20 seconds when its slope changes from 0.5/v to 1/v. Consequently, the coding bit rate will not change as smoothly as with the logarithmic target schedule, although it will not be as abrupt as the schedule itself because of the smoothness objective in the controller design.

B. Optimal Controller Design

[0063] Recall from Eq. (18) the fundamental state transition equation, which describes the evolution of the buffer tube upper bound $t_b(n)$ in terms of the coding bit rate $r_c(n)$:

$$t_b(n+1) = t_b(n) + \frac{r_c(n+1)}{f\tilde{r}_a} + w(n). \qquad (26)$$

Here it is now assumed that the frame rate f and the average arrival rate $\tilde{r}_a$ are relatively constant. Deviations from this assumption are captured by $w(n)$.

[0064] It is desired to control the upper bound by adjusting the coding bit rate. As each frame arrives at the client, a feedback loop can send a message to the server to adjust the coding bit rate. Note, however, that by the time frame n arrives completely at the client, frame $n + 1$ has already started streaming from the server. Thus the coding bit rate $r_c(n + 1)$ for frame $n + 1$ must already be determined by time $t_a(n)$. Indeed, at time $t_a(n)$, frame n+2 is the earliest frame for which the controller can determine the coding bit rate. Hence at time $t_a(n)$, the controller's job must be to choose $r_c(n + 2)$. This one-frame delay in the feedback loop must be explicitly accounted for.

[0065] For simplicity, the target schedule is linearized around the time that frame n arrives. The linearization is equivalent to using a line tangent to the original target schedule at a particular point as an approximate target schedule. Thus,

$$t_T(n+1) - 2t_T(n) + t_T(n-1) = 0. \qquad (27)$$

Rather than directly control the evolution of the upper bound, which grows without bound, for the purposes of stability an error space formulation is used. Defining the error as

$$e(n) = t_b(n) - t_T(n), \qquad (28)$$

Ex. 1018-244

results in

$$\begin{aligned} & e(n+1)-e(n) \\ & = (t_b(n+1)-t_T(n+1))-(t_b(n)-t_T(n)) \end{aligned} \tag{29}$$

$$= (t_b(n+1)-t_b(n))-(t_T(n+1)-t_T(n)) \tag{30}$$

$$= \frac{r_c(n+1)}{f\tilde{r}_a}-(t_T(n+1)-t_T(n))+w(n), \tag{31}$$

from which in turn

$$\begin{aligned} & (e(n+1) - e(n)) - (e(n) - e(n-1)) \\ & = [r_c(n+1)-r_c(n)]/f\tilde{r}_a \\ & \quad -(t_T(n+1)-2t_T(n)+t_T(n-1)) \\ & \quad +(w(n)-w(n-1)) \end{aligned} \tag{32}$$

$$= \frac{r_c(n+1)-r_c(n)}{f\tilde{r}_a}+(w(n)-w(n-1)) \tag{33}$$

The control input is then defined as

$$u(n) = \frac{r_c(n+2)-\hat{r}_c(n+1)}{\tilde{r}_a}, \tag{34}$$

where $\hat{r}_c(n+1)$ is a possibly quantized version of $r_c(n+1)$ (as defined in Section IV-D) and the disturbance is defined as

$$d(n) = \frac{\hat{r}_c(n)-r_c(n)}{f\tilde{r}_a}+w(n)-w(n-1). \tag{35}$$

Then Eq. (33) can be rewritten

$$e(n+1) = 2e(n)-e(n-1)+\frac{u(n-1)}{f}+d(n). \tag{36}$$

Therefore , defining the error vector

$$\mathbf{e}(n)=\begin{bmatrix} e(n) \\ e(n-1) \\ u(n-1) \end{bmatrix}=\begin{bmatrix} t_b(n) \\ t_b(n-1) \\ \dfrac{r_c(n+1)}{\tilde{r}_a} \end{bmatrix}-\begin{bmatrix} t_T(n) \\ t_T(n-1) \\ \dfrac{\hat{r}_c(n)}{\tilde{r}_a} \end{bmatrix} \qquad (37)$$

the error space representation of the system can be expressed as

$$\mathbf{e}(n+1)=\begin{bmatrix} 2 & -1 & \dfrac{1}{f} \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}\mathbf{e}(n)+\begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}u(n)+\begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}d(n), \qquad (38)$$

or $e(n+1) = \Phi e(n) + \Gamma u(n) + \Gamma_d d(n)$ for appropriate matrices Φ, Γ and Γ_d.

[0066] Assuming the disturbance *d(n)* is a pure white noise, and assuming *perfect state measurement* (i.e., all components of $e(n)$ can be measured without using an estimator), the disturbance $d(n)$ does *not* affect the controller design. Thus a linear controller represented by

$$u(n)=-G\mathbf{e}(n) \qquad (39)$$

can be used where G is a *feedback gain.* By the time frame n is completely received, all elements of $\mathbf{e}(n)$ are available at the client and $u(n)$ can thus be computed. The ideal coding bit rate for frame $n+2$ can then be computed as

$$r_c(n+2)=\hat{r}_c(n+1)-G\mathbf{e}(n)\tilde{r}_a. \qquad (40)$$

Finding the optimal linear controller amounts to finding the feedback gain G^* that minimizes the quadratic cost function defined in Section II-D. Before continuing with the design, the system *controllability matrix C,*

$$C=\begin{bmatrix} \Gamma & \Phi\Gamma & \Phi^2\Gamma \end{bmatrix}=\begin{bmatrix} 0 & \dfrac{1}{f} & \dfrac{2}{f} \\ 0 & 0 & \dfrac{1}{f} \\ 1 & 0 & 0 \end{bmatrix} \qquad (41)$$

will be checked first which has full rank for any frame rate f. Thus, the system is *completely controllable* and the state $e(n)$ can be regulated to any desirable value. Now recall that the cost function defined in Section II-D is

$$I = \sum_{n=0}^{N} \left\{ (t_b(n) - t_T(n))^2 + \sigma \left(\frac{r_c(n+1) - r_c(n)}{\tilde{r}_a} \right)^2 \right\} \tag{42}$$

$$= \sum_{n=0}^{N} \{ e(n)^T Q e(n) + u(n-1)^T R u(n-1) \}, \tag{43}$$

where $Q = C^T C$ (with C = [1 0 0]) and R = σ. Then, the original control problem of tracking the target schedule while smoothing the coding bit rate fluctuations (i.e., minimizing the cost function *I*) is converted to a standard regulator problem in the error space. Letting $N \to \infty$, the infinite horizon optimal control problem can be solved by applying the results in [11, Section 3.3] to obtain an optimal regulator in two steps: 1) solving, to get S, the *discrete algebraic Riccati equation* (DARE)

$$S = \Phi^T \{ S - S\Gamma[\Gamma^T S \Gamma + R]^{-1} \Gamma S \} \Phi + Q, \tag{44}$$

and 2) computing the optimal feedback gain

$$G^* = [\Gamma^T S \Gamma + R]^{-1} \Gamma^T S \Phi. \tag{45}$$

The existence and uniqueness of S (and in turn of G^*) is guaranteed when *Q* is nonnegative definite and R is positive definite, which is straightforward to verify in this case.

C. Frame Rate

[0067] In the previous section, it was assumed that the frame rate is constant. This assumption is reasonable when streaming a single medium, such as video without audio. Variable frame rate video is usually achieved by skipping frames, which can be accommodated by setting $b(n) = 0$. However, usually video and audio are streamed together, and their merged coding schedule may have no fixed frame rate. Even if there is a fixed frame rate *f*, it may be desirable to operate the controller at a rate lower than *f*, to reduce the feedback rate, for example.

[0068] To address these issues, in practice the notion of a *virtual frame rate* is used. A virtual frame rate *f* is chosen, for example *f* =1 frame per second (fps); media time is partitioned into intervals of size 1/*f*; and all of the (audio and video) frames arriving within each interval are modeled as a *virtual frame* whose decoding and playback deadline is the end of the interval.

[0069] This approach has several advantages. First, it allows offline design of a universal feedback gain, which is independent of the actual frame rate of the stream or streams. Second, it allows the rate of feedback from the client to the server to be reduced. And finally, since the interval between virtual frames is typically safely larger than a round trip time (RTT), a one-frame delay in the error space model (as described in the previous section) is sufficient to model the feedback delay. Otherwise it would be necessary to model the feedback delay with approximately *RTT/f* additional state variables to represent the network delay using a shift register of length *RTT/f*.

[0070] In the remainder of this description a virtual frame rate *f* = 1 fps is employed, and is referred to simply as the frame rate. Likewise, a virtual frame is referred to simply as a frame.

[0071] It is noted that with the present controller design rationale, the deviation of the buffer tube upper bound from the control target can be reduced by decreasing the σ value. A smaller value of σ value implies a relative larger penalty on the deviation term in the cost function and thus forces the upper bound to track the target more closely. This, however, happens at the cost of sacrificing coding bit rate smoothness, since the corresponding term in the cost function will be weighted less. It has been found with σ = 500 that while the buffer tube upper bound deviates only slightly from the control target, the coding bit rate has undesirable oscillations. On the other hand, a large σ value will certainly yield smoother coding bit rates, but might also incur client buffer underflow since the buffer tube upper bound is allowed to

deviate significantly away from the control target. Therefore, a good choice of σ should take into account this trade-off. In tested embodiments of the present system and process, $\sigma = 4000$ was chosen when the coding bit rate switches up and $\sigma = 2000$ when it switches down. Note that a slightly more aggressive strategy is allowed in the latter case to further reduce the chance of client buffer underflow.

IV. Practical Issues with Streaming

A. Fast Startup

[0072] As discussed in previous sections, the startup delay is the length of the period from the time that content first begins to arrive at the client to the time that playback begins. During this period, content accumulates in the receiver buffer to counter packet jitter, retransmission delay, variations in network bandwidth, and variations in instantaneous coding bit rate. It is conceivable that a longer startup delay would increase the chances of being able to maintain continuous playback in a dynamic network environment. On the other hand, users expect the startup delay to be as small as possible. Thus, it is desirable to investigate techniques that can reduce the startup delay while retaining robustness. One possible approach is to transmit the content at a faster than normal rate at the beginning of streaming. This *bursting* technique will certainly build up the buffer duration in a small amount of time. It, however, puts extra pressure on the network by demanding a higher than normal initial bandwidth, which may not even be available.

[0073] The present system and process employs an alternative *fast startup* technique, which takes advantage of the properties of adaptive media. As discussed in previous sections, by choosing an initial coding bit rate r_c equal to half the arrival rate r_a (divided if necessary by the playback speed v), the client buffer duration can grow at two times real time during playback. Growing the client buffer during playback enables the startup delay to be low, because playback can begin while the buffer duration is still low. Beginning playback while the buffer duration is low is not particularly risky over the short term, because the probability of deep congestion occurring in any short interval is low. However, the probability of deep congestion occurring in a long interval is high, so it is important for the buffer duration to be high over the long term. Without the ability to grow the buffer duration during playback, startup would have to be delayed until the buffer duration was sufficiently high to guarantee continuous playback over the long term.

[0074] Moreover, if the transmission rate is twice the coding bit rate, the startup delay can be further reduced by taking advantage of properties of the leaky bucket model [10]. As detailed in Section II-B, the startup delay for a given bit stream is $D = F^d_{\min}(R)/R$ when the stream is transmitted at rate R. This is ordinarily equal to $F^d_{\min}(r_c)/r_c$ when transmitting the stream at its coding bit rate. However, when transmitting the stream at a rate $r_a > r_c$ ($r_c = 0.5r_a/v$), then the startup delay drops to $F^d_{\min}(r_a)/r_a$. Thus the startup delay D decreases both because the numerator decreases and because the denominator increases.

[0075] The graph in Fig. 8 illustrates the decrease in the initial decoder buffer fullness $F^d_{\min}(R)$ as R changes from r_c to r_a. In particular, it depicts the coding schedule for a given bit stream, as well as upper and lower bounds, denoted Tube I and Tube II, corresponding to two leaky buckets with leak rates r_c and r_a respectively, both containing the coding schedule. Tube II is smaller than Tube I, since the minimum size $B_{\min}(R)$ of a leaky bucket containing a given stream is decreasing in the leak rate R [10]. Likewise, the initial decoder buffer fullness $F_{\min}(R)$ is decreasing in R [10]. Hence the playback deadline for frame 0 can begin as early as client time $t_{0_II} = F^d_{\min}(r_a)/r_a$, instead of $t_{0_I} = F^d_{\min}(r_c)/r_a$. From there, the playback deadline advances at $1/v$ seconds of client time per second of media time.

B. Controller Initialization

[0076] As illustrated in Fig. 8, the target schedule starts at the same time as the playback deadline and grows according to a predefined function. The controller attempts to control the upper bound of Tube I to the target schedule. Initially the upper bound of Tube I is above the target schedule (and is indeed above the playback deadline, though it is known that this is safe). Hence, when the playback starts, the controller would try to close the gap by decreasing the coding bit rate. This, however, would not be desirable because the current coding bit rate is already lower than the arrival rate to allow the client buffer to grow. Further reduction of the coding bit rate would not be proper. To avoid this effect, the controller is initialized when the upper bound of Tube I exceeds the target schedule i.e., at point B in Fig. 8. Point B can be found analytically, but in practice there is no need to explicitly solve for it. The controller can be initialized as soon as the upper bound of Tube I exceeds the target.

C. Exponential Averaging of the Arrival Rate

[0077] Using the average arrival rate (instead of the instantaneous arrival rate) helps to reduce coding bit rate oscillations. This section details a new exponential averaging algorithm for the arrival rate.

[0078] Let $\tilde{r}_a(k)$ and $r(k)$ be the average arrival rate and the instantaneous arrival rate, respectively, when packet k is received. Note that unlike the controlling operation, the rate averaging operation may be performed after the arrival of every *packet,* rather than after the arrival of every *frame.* Hence the discrete packet index k is used rather than the frame index n . Instead of using the widely adopted exponentially weighted moving average (EWMA)

$$\tilde{r}_a(k) = \beta(k)\tilde{r}_a(k-1) + (1-\beta(k))r_a(k) \qquad (46)$$

with constant $\beta(k) = \beta$, the exponential averaging is performed more carefully. In the present algorithm, the factor $\beta(k)$ is not constant, but varies according to the packets' inter-arrival gaps. This new algorithm has several advantages over the EWMA algorithm with constant $\beta(k)$. First, the estimate of the average arrival rate $\tilde{r}_a(k)$ goes to zero naturally as the gap since the last packet goes to infinity, rather than being bounded below by $\beta\tilde{r}_a(k\text{-}1)$. Second, the estimate of the average arrival rate $\tilde{r}_a(k)$ does not go to infinity as the gap since the last packet goes to zero. This is especially important, since packets often arrive in bursts, causing extremely high instantaneous arrival rates. And finally, the estimate of the average arrival rate $\tilde{r}_a(k)$ does not over-weight the initial condition, as if it represented the infinite past. This is especially important in the early stages of estimation.

[0079] As in Eq. (11), the instantaneous arrival rate after packet k is defined as

$$r_a(k) = \frac{b(k)}{t_a(k) - t_a(k-1)}, \qquad (47)$$

where here $b(k)$ denotes the size of packet k and $t_a(k)$ denotes the arrival time of packet k. The discrete time function $r_a(k)$ is extended to the piecewise constant continuous time function $r_a(t)$ by

$$r_a(t) = r_a(k) \quad \text{for all } t \in (t_a(k-1), t_a(k)), \qquad (48)$$

as illustrated in the graph in Fig. 9. Then the function $r_a(t)$ is filtered by the exponential impulse response $\alpha e^{-\alpha t}$, $t \geq 0$, for some time constant $1/\alpha$:

$$\tilde{r}_a(k) = \frac{\int_{t(0)}^{t(k)} r_a(t')\alpha e^{-\alpha(t(k)-t')}dt'}{\int_{t(0)}^{t(k)} \alpha e^{-\alpha(t(k)-t')}dt'}. \qquad (49)$$

(Here and in the remainder of this sub-section the subscript is suppressed from the arrival time $t_a(k)$.) Note that $\int_t^{\infty} \alpha e^{-\alpha t'}dt' = e^{-\alpha t}$, the denominator integral can be expressed as $1 - e^{-\alpha(t(k)-t(0))}$. Now, the range of the numerator integral is split into ranges (t(0), t(k-1)] and (t(k-1),t(k)] to obtain a recursive expression for $\tilde{r}_a(k)$ in terms of $\tilde{r}_a(k\text{-}1)$ and $r_a(k)$,

$$\tilde{r}_a(k) = \frac{1-e^{-\alpha[t(k-1)-t(0)]}}{1-e^{-\alpha[t(k)-t(0)]}} e^{-\alpha[t(k)-t(k-1)]} \tilde{r}_a(k-1) + \frac{1-e^{-\alpha[t(k)-t(k-1)]}}{1-e^{-\alpha[t(k)-t(0)]}} r_a(k) \tag{50}$$

$$= \beta(k)\tilde{r}_a(k-1) + (1-\beta(k))r_a(k), \tag{51}$$

where

$$\beta(k) = \frac{e^{-\alpha[t(k)-t(k-1)]} - e^{-\alpha[t(k)-t(0)]}}{1-e^{-\alpha[t(k)-t(0)]}}. \tag{52}$$

Note that $\beta(k)$ is numerically stable as k goes to infinity. However, as the gap $\delta = t(k) - t(k\text{-}1)$ goes to zero, $1 - \beta(k)$ goes to zero while $r_a(k)$ goes to infinity. Their product, however, is well behaved. Indeed,

$$\tilde{r}_a(k) = \frac{1-e^{-\alpha[t(k-1)-t(0)]}}{1-e^{-\alpha[\delta+t(k-1)-t(0)]}} e^{-\alpha\delta} \tilde{r}_a(k-1) + \frac{1-e^{-\alpha\delta}}{1-e^{-\alpha[t(k)-t(0)]}} \frac{b(k)}{\delta} \tag{53}$$

$$\rightarrow \tilde{r}_a(k-1) + \frac{\alpha b(k)}{1-e^{-\alpha[t(k)-t(0)]}} \tag{54}$$

as $\delta \rightarrow 0$, using l'Hôpital's rule. Thus Eq. (54) is the update rule in the case when $t(k) = t(k\text{-}1)$.

D. Choosing a Stream Given a Coding bit rate

[0080] When the client requests a coding bit rate $r_c(n)$, the server complies by choosing a stream (or sub-stream of a scalable stream) having coding bit rate $\hat{r}_c(n)$ approximately equal to $r_c(n)$. There are several reasons that $\hat{r}_c(n)$ may differ from $r_c(n)$. The first reason is that there are only a finite number of streams (or sub-streams) in the media file, even if fine grain scalable coding is used. Thus there may be no stream in the media file with average coding bit rate exactly equal to $r_c(n)$. The second reason is that, even if there is a stream in the media file with average coding bit rate exactly equal to $r_c(n)$, the buffer tube for the stream may be too large to allow switching to the stream without risk of client buffer underflow. In fact, whenever the stream switches, there is generally a discontinuity in the upper bound, which may be either positive or negative. A positive shift in the upper bound is illustrated in the graph in Fig. 10, which, if large, could cause the client buffer to underflow either immediately or eventually.

[0081] Thus the server must choose a stream that causes the upper bound to shift up no more than some amount $\Delta^{max}g(n\text{-}1)$ supplied to it by the client. The client supplies $\Delta^{max}g(n\text{-}1)$ to the server in its feedback along with $r_c(n)$, shortly after client time $t_a(n\text{-}2)$ (after frame n-1 has already begun streaming). Upon receiving the feedback, the server selects a stream with coding bit rate $\hat{r}_c(n)$ as high as possible such that $\hat{r}_c(n) \le r_c(n)$ and, if $\hat{r}_c(n) > \hat{r}_c(n\text{-}1)$ (i.e., if it is a switch up in rate), then $g^{new}(n\text{-}1)\text{-}g^{old}(n\text{-}1) \le \Delta^{max}g(n\text{-}1)$, where $g^{new}(n\text{-}1)$ and $g^{old}(n\text{-}1)$ are illustrated in Fig. 10. The constraint given by $\Delta^{max}g(n\text{-}1)$ is not applied if it is a switch down in rate.

[0082] The client chooses $\Delta^{max}g(n - 1)$ to limit (its prediction of) what the upper bound would be at time $t_a(n - 1)$ if the new coding bit rate were in effect, namely,

$$t_b^{new}(n-1) \approx \frac{t_b(n-2)+\hat{r}_c(n-1)}{f\tilde{r}_a} + \frac{\Delta g(n-1)}{\tilde{r}_a} \qquad (55)$$

$$\leq t_T(n-1) + p[t_d(n-1) - t_T(n-1)]. \qquad (56)$$

That is, the client chooses $\Delta^{max}g(n\text{-}1)$ to limit $t_b^{new}(n-1)$ so that it would be no more than fraction p of the way from the target $t_T(n\text{-}1)$ to the playback deadline $t_d(n\text{-}1)$. In tested embodiments of the present system and process, p=1/3 was chosen.

[0083] It is noted that in an alternate embodiment, the information as to what coding bit rates the server supports can be provided to the client ahead of time and the client can be responsible for performing the foregoing analysis. In this case, the client would only request coding bit rates that are supported by the server, and there would be no need to provide the $\Delta^{max}g(n\text{-}1)$ to the server with the request.

E. Control Target Adjustment

[0084] When a frame with a new average coding bit rate $\hat{r}_c(n)$ arrives at the client at time $t_a(n)$, there is a shift in the upper bound. This shift can be on the order of seconds and hence, rather than being negligible, can be confusing to the controller. If the shift is upward, for example, the controller will immediately try to reduce the coding bit rate $r_c(n+2)$. If the shift is downward, on the other hand, the controller will immediately try to increase the coding bit rate $r_c(n+2)$. Either way is probably not good; the intention is that $\hat{r}_c(n)$ will be maintained unless there is a disturbance in the arrival rate. One solution is to introduce a simultaneous shift in the control target schedule equal to $\Delta g(n-1)/\tilde{r}_a$; where $\Delta g(n-1) = g^{new}(n-1) - g^{old}(n\text{-}1)$ is the actual shift in the upper bound (in bits) at frame n-1 computed at the server, as illustrated in Fig. 10. The server can send this value to the client along with frame n. If there is no stream change, this value is simply zero.

[0085] If the control target schedule is adjusted whenever the coding bit rate changes, it will no longer follow the designed target schedule. The adjusted target schedule will be referred to as the *control target* schedule to distinguish it from the *designed target* schedule (or simply the *target schedule).*

[0086] The control target schedule, of course, should have a tendency to approach the designed target schedule. The basic idea is to decrease the slope of the control target schedule when it is above the designed target schedule and to increase the slope when it is below.

[0087] For the logarithmic target schedule $t_T = t_d - \frac{b}{a}\ln(at_d+1)$ (where $t_d = t_{d0} + (\tau_d - \tau_{d0})/v$), according to Eq. (24) the slope at media time τ_d is

$$s = \frac{dt_T}{d\tau_d} = \frac{1}{v} - \frac{b}{a(\tau_d - \tau_{d0}) + v}. \qquad (57)$$

If d is defined as the distance between the playback deadline and the target schedule, namely

$$d = \frac{b}{a}\ln\left(a\left(\frac{\tau_d - \tau_{d0}}{v}\right)+1\right), \qquad (58)$$

then the slope may be expressed as a function of d

$$s = \frac{1}{\nu} - \frac{b}{\nu e^{(a/b)d}}. \qquad (59)$$

Hence whenever d is the distance between the playback deadline and the control target, the slope of the control target is set to s in Eq. (59). Specifically, if $t_T^{\Delta}(n)$ is the control target at frame n after the shift, then $t_T^{\Delta}(n\text{-}1)$ is reset to be $t_T^{\Delta}(n)\text{-}s/f$. Then $t_T^{\Delta}(n)$ and $t_T^{\Delta}(n\text{-}1)$ is used in place of $t_T(n)$ and $t_T(n\text{-}1)$ to compute the error vector $e(n)$ in Eq. (37). The resulting error vector is then used to compute the ideal coding bit rate in Eq. (40).

[0088] For the two-piece linear target schedule, the slope is easy to compute by using a predefined time period over which the control target schedule is expected to return to the target schedule. The slope of the control target schedule can then be computed from the distance d and the period. In tested embodiments of the present system and process the period was set to 50 seconds.

V. Implementation Details

[0089] This section highlights implementation details on both the sender and the receiver side.

A. Generation of Virtual Streams

[0090] In tested embodiments of the present a system and process, a fine grained scalable (FGS) stream comprises a set of data units, each tagged by a Lagrange multiplier λ representing the per-bit decrease in distortion if the data unit is received by the client. If the λ for the data unit is above a threshold, then the data unit is included in a virtual stream corresponding to that threshold. Each threshold corresponds to an overall number of bits and hence an average coding bit rate for the virtual stream. In tested embodiments, $N = 50$ virtual streams are generated. A threshold is chosen for each stream such that the resulting streams have coding bit rates that are uniformly spaced in the log domain between lower and upper bounds.

[0091] During streaming, when the server reads a data unit from the media file, it includes the data unit in the virtual stream currently being transmitted if its Lagrange multiplier λ is above the threshold for the stream.

B. Leaky Bucket Computations at the Sender

[0092] For each virtual stream, leaky bucket parameters $(R, B_{\min}(R), F_{\min}^{d}(R))$ are pre-computed offline for $R = R_{avg}$ and $R = R_{\max}$, where $R_{avg} = r_c$ is the average coding bit rate of the stream, and $R_{max} = 2r_c$. These leaky bucket parameters are sent to the client in a preamble.

[0093] In addition, during streaming the server performs on-line leaky bucket simulations for each stream. Specifically, whenever the server reads a data unit from the media file, it determines the virtual streams to which the data unit belongs, using the Lagrange multiplier of the data unit and the list of thresholds for each stream. The sender then updates, for the determined streams, the states of those leaky buckets having leak rates equal to an average coding bit rate R_{avg}, using Eqs. (2) and (3). Once all the data units in a frame are read from the media file, the sender computes $g(n) = B_{\min}(R_{avg}) - B^{e}(n)$ for each of the virtual streams. On a stream switch (i.e., $\hat{r}_c(n) \neq \hat{r}_c(n-1)$), the gap $g^{new}(n)$ for the new stream is transmitted to the client along with $\Delta g(n\text{-}1)=g^{new}(n\text{-}1)\text{-}g^{old}(n\text{-}1)$ as described below. It is easy to see that the cost of updating the leaky bucket states is quite low. However, it is also possible to pre-compute these values and store them with each data unit in the media file.

C. Initial Coding bit rate Selection

[0094] At the beginning of a streaming session, the sender needs to have some knowledge of the available network bandwidth so that it can choose an initial coding bit rate (usually half of the bandwidth). The bandwidth estimate can be drawn from proactive measurements, using approaches such as packet pair [12], path chirp [3], etc., or reactive approximations based on history values. The exact form of the initial bandwidth estimation is beyond the scope of this work.

D. Coding bit rate Switching

[0095] The rate control feedback from the client contains the frame number at which feedback is generated (s.g., n -

2 in the previous section) and the maximum allowable shift of the upper bound in bits (e.g., $\Delta^{max}g(n\text{-}1)$ in the previous section). If the sender finds a suitable coding bit rate and makes a switch at frame n, it will transmit three values to the client along with the frame: the new coding bit rate $\hat{r}_c^{new}(n)$, the current gap to the upper bound $g^{new}(n)$, and the shift $\Delta g(n\text{-}1) = g^{new}(n\text{-}1)\text{-}g^{old}(n\text{-}1)$. With this information, the client can properly adjust its control target schedule as well as its upper bound. Note that coding bit rate switching always happens at the beginning of a new frame, never inside a frame.

E. Optimal Rate Control at the Client

[0096] Whenever a new coding bit rate starts, the client receives the value $g(n)$ along with the new frame. The values of $g(n)$ for successive frames can be then inferred by the client itself based on the coding bit rate $\hat{r}_c(n)$ and the frame size $b(n)$. The client records the arrival frame time $t_a(n)$, calculates the buffer tube upper bound $t_b(n)$ and then computes the deviation $e(n)$. If there is a coding bit rate switch, it will also compute the buffer tube shift and adjust the control target schedule accordingly. Then, $e(n)$ is fed to the optimal rate controller, which outputs a desired new coding bit rate. The latest new coding bit rate is fed back to the sender whenever there is a feedback opportunity, which could be generated at regular intervals or on-demand.

F. Exemplary Process

[0097] An exemplary process for the implementation of the foregoing system will now be provided as applicable to one embodiment of the present invention.

[0098] Referring to Figs. 11A-C, the server first computes a set of parameters defining an initial state of the encoder buffer as it would exist if a data stream corresponding to a supported coding bit rate were streamed therefrom, for each of the supported coding bit rates (process action 1100). These parameters, which include the coding bit rate associated with the data stream, the size of the encoder buffer employed with the coding bit rate of the data stream, and a value indicative of the initial encoder buffer fullness exhibited at the coding bit rate of the data stream, are provided to the client in a preamble to the streaming media data (process action 1102). The server then streams the media data to the client at a prescribed initial coding bit rate, during a startup period (process action 1104). As the data is streamed, the server computes the upper bound gap for each frame generated for each coding bit rate supported by the server (process action 1106). Initially, the client employs the encoder buffer parameters to determine on a frame-by-frame basis a new, optimal coding bit rate for a future frame (process action 1108). In addition, the client computes the aforementioned maximum allowable difference value for the frame generated by the server just prior to the future frame, i.e., the maximum shift value (process action 1110). The client also begins playback of the frames it has received as soon as the client's decoder buffer has the minimum amount of data therein that is needed to ensure an underflow condition does not occur at the initial coding bit rate and continues to playback each frame received thereafter (process action 1112). In addition, once the aforementioned startup period is over, the client requests that the streaming media data be supplied at a new coding bit rate corresponding to the determined optimal coding rate, starting at the future frame, and provides the maximum shift value along with the request (process action 1114).

[0099] The server finds a supported coding bit rate that is equal to, or if none are equal, the closest smaller rate to, the optimum coding bit rate requested by the client (process action 1116). It is next determined by the server if the supported coding bit rate found is lower or higher than the coding bit rate associated with the frame generated immediately prior to the frame specified in the client request (process action 1118). If it is lower, then server computes the aforementioned shift value for the frame generated immediately prior to the frame specified in the client request (process action 1120), and generates all future frames beginning with the frame specified by the client at the supported rate found and streams them to the client, along with the computed upper bound gap and shift value, and an indication of the new coding bit rate (process action 1122). However, if the supported coding bit rate found is higher, then server computes the shift value for the frame generated immediately prior to the frame specified in the client request (process action 1124) and then determines if the shift value is less than or equal to the maximum shift value provided in the client request (process action 1126). If the shift value is less than or equal to the maximum shift value, then the server generates all future frames beginning with the frame specified by the client at the supported coding bit rate found and streams them to the client, along with the computed upper bound gap and shift value, and an indication of the new coding bit rate (process action 1122). But, if the shift value is greater than the maximum shift value, then the next lower supported coding bit rate is found (process action 1128) and the process actions 1118 through 1128 are repeated.

[0100] The client shifts the currently scheduled target arrival time associated with a just received first frame at the new coding bit rate by the shift value provided, and shifts the currently scheduled target arrival times for future frames such that they collectively approach, over a period of time, and eventually coincide with, the previous target arrival times for those frames (process action 1130). The client then employs the encoder buffer parameters and the upper bound gap value provided by the server to determine a new, optimal coding bit rate for a future frame and to compute the

maximum shift value for the frame generated by the server just prior to the future frame (process action 1132). The client requests that the streaming media data be supplied at a new coding bit rate corresponding to the determined optimal coding rate starting at the future frame specified by the client, and includes the computed maximum shift value (process action 1134). Process actions 1116 through 1134 are then repeated for the duration of the streaming media data transmission.

VI. Multiple Bit Rate Streaming

[0101] Multiple bit rate (MBR) streaming is a network adaptive technique that is widely used in commercial streaming media systems. In MBR streaming, in contrast to scalable streaming, the content is encoded into several (typically 5-7) independent streams at different coding bit rates. Often, each stream is optimized for a common type of network connection (e.g., dial-up, DSL, cable). During an MBR streaming session, the proper coding bit rate is dynamically selected based on the available network bandwidth, with the goal of achieving the maximum possible quality under the condition of uninterrupted playback. It is easy to see that MBR streaming is analogous to scalable streaming. Indeed MBR streaming can be viewed as a special case of scalable streaming with a limited number of coding bit rates available. Hence, the foregoing optimal control approach should be applicable to this case.

[0102] There are, however, several differences that complicate MBR streaming, which need to be carefully addressed. First, as just mentioned, in MBR streaming there are only a limited number of coding bit rates available. This coarse quantization of the desired coding bit rate introduces a significant nonlinearity into the closed loop system. In fact, the large gaps between the available coding bit rates introduce oscillations. For example, if two neighboring coding bit rates straddle a constant arrival rate, the controller will oscillate between the two coding bit rates in an attempt to keep the client buffer at a target level.

[0103] Second, in MBR streaming the coding bit rate cannot be switched at an arbitrary time. In fact, before the server can switch to a new stream, it may have to wait for the next clean point (e.g., *I* frame) in the new stream, which could be five or ten seconds away. Thus, the old coding bit rate may continue for quite a while before it changes to the new coding bit rate. From the controller's perspective, this long random extra delay tends to destabilize the closed-loop system.

[0104] Third and finally, in MBR streaming, server performance issues are critical. The commercial-grade streaming media systems that use MBR streaming do so because of the minimal computational load that it imposes on the server compared to scalable streaming. Thus, for MBR streaming it is important to keep almost all computation and state maintenance on the client side. In particular, the server will not be able to update the leaky bucket information for each stream. Instead, the client must use some mechanism for estimating and maintaining this information.

A. Conservative Up-Switching

[0105] In this subsection, a technique to help stabilize the control system and reduce steady state oscillations to a period of at least a minute is described. With this technique, rapid down-switching is permitted. In fact, the value of σ is reduced from 2000 to 500, changing the balance between responsiveness and smoothness of the coding bit rate in favor of rapid switching response. However, only conservative up-switching is permitted. Conservative up-switching ensures that spurious changes in coding bit rate do not occur, and that oscillations in the coding bit rate have a low frequency. In particular, conservative up-switching reduces the oscillations between two adjacent but widely spaced MBR coding bit rates, one above the arrival rate and one below the arrival rate.

[0106] In one embodiment, conservative up-switching establishes a conservative limit that limits the coding bit rate to no more than the arrival rate. However, in another embodiment, the method behind conservative up-switching is to establish a conservative limit on how high the coding bit rate can be raised above the arrival rate. If the current coding bit rate is below the arrival rate, and the client buffer duration begins to increase above its target level, then the coding bit rate can be switched up to a new coding bit rate above the arrival rate only if the new coding bit rate is below the conservative limit. When the client buffer duration begins at the target level, the conservative limit is equal to the arrival rate. However, as the client buffer duration increases, the conservative limit increases as well. Thus, if the current coding bit rate is below the arrival rate, and the next higher coding bit rate is above the arrival rate, then it will be possible to switch up to the next higher coding bit rate only after the client buffer duration has increased sufficiently so that the conservative limit rises above the higher coding bit rate. Once the coding bit rate is switched up to the higher coding bit rate, the client buffer begins to drain since the coding bit rate is then above the arrival rate. Eventually, when the buffer drains back below its target level, the controller will rapidly switch the coding bit rate back down to the coding bit rate below the arrival rate.

[0107] Given the current client buffer duration, the conservative limit is set to a value such that if the coding bit rate is switched up to a new coding bit rate at this value, the client buffer would take at least Δt seconds of client time to drain back to the target level. Thus, the mechanism ensures that the period of oscillation will be at least Δt seconds. In tested embodiments of the present system and process, Δt is set to 60 seconds.

[0108] Fig. 12 shows how the conservative limit is computed. Let $\Delta\tau_1$ be the client buffer duration (in media time) at the moment that the coding bit rate is switched up from r_c^{old} to r_c^{new}. Thus $\Delta\tau_1$ is the number of seconds of content that will be consumed at the old coding bit rate r_c^{old} before content at the new coding bit rate begins to be consumed. (For simplicity it is assumed that all of the content in the client buffer at the time of the switch is coded at rate r_c^{old}.) Let $\Delta\tau_2$ be the number of seconds of content that is consumed at the new coding bit rate r_c^{new} before the client buffer duration drops to some level $\Delta\tau_3$ seconds (in media time), greater than the target level $\Delta\tau_T$. The duration of this phase is determined such that the total time since the switch is exactly $\Delta t = (\Delta\tau_1 + \Delta\tau_2)/v$ seconds (in client time). Now, the number of bits that arrive in this time is $r_a\Delta t = r_c^{new}(\Delta\tau_2 + \Delta\tau_3) \geq r_c^{new}(\Delta\tau_2 + \Delta\tau_T) = r_c^{new}(v\Delta t - \Delta\tau_1 + \Delta\tau_T)$, or

$$r_c^{new} \leq \frac{r_a\Delta t}{v\Delta t - \Delta\tau_1 + v\Delta t_T}, \qquad (60)$$

where Δt_T is the target buffer duration in client time. The parameter Δt can be tuned to yield the desired behavior. A large Δt means that up-switching will be more conservative, while a smaller Δt means that up-switching will be more prompt. In tested embodiments, Δt is set to 60 seconds while the target Δt_T is typically about 10 seconds.

B. Buffer Tube Upper Bound Estimation

[0109] In Section V-D it was specified that the server sends three values to the client at the beginning of each change in coding bit rate: the new coding bit rate $\hat{r}_c^{new}$, the current gap to the upper bound $g^{new}(n)$, and the control target shift $\Delta g(n - 1) = g^{new}(n - 1) - g^{old}(n - 1)$. The server computes the latter two values by running a leaky bucket simulator for each coding bit rate. The client continues to update $g(n)$ for the new coding bit rate by running its own leaky bucket simulator for the new coding bit rate. That is, beginning with the initial condition $F^e(n) = B - b(n) - g^{new}(n)$, for each successive frame the client computes

$$B^e(n) = F^e(n) + b(n) \qquad (61)$$

$$F^e(n+1) = \max\{0, B^e(n) - \hat{r}_c/f(n)\}, \qquad (62)$$

where

$$f(n) = \frac{1}{\tau(n+1) - \tau(n)} \qquad (63)$$

is the instantaneous frame rate, as in Eqs. (2), (3) and (4). From this, the client can compute

$$g(n) = B - B^e(n) \qquad (64)$$

for each frame.

[0110] However, if the server is unable to simulate the leaky buckets and cannot send $g^{new}(n)$ to the client, then the client must estimate this information for itself. In this case it is recommended that the client estimates $g^{new}(n)$ as an upper bound such as $\hat{g}^{new}(n)=B-b(n)\geq g^{new}(n)$. Then, beginning with initial condition $\hat{F}^{e}(n)=B-b(n)-g^{new}(n)$ (which equals 0 in this case), for each successive frame the client computes

$$\hat{B}^{e}(n) = \hat{F}^{e}(n) + b(n) \qquad (65)$$

$$\hat{F}^{e}(n+1) = \max\{0, \hat{B}^{e}(n) - \hat{r}_c/f(n)\}, \qquad (66)$$

as well as

$$\hat{g}(n) = B - \hat{B}^{e}(n). \qquad (67)$$

It is easy to see by induction that $\hat{F}^{e}(n) \leq F^{e}(n)$, $\hat{B}^{e}(n) \leq B^{e}(n)$, and $\hat{g}(n) \geq g(n)$. Moreover, these bounds each become tighter by $\delta(n) = \hat{r}_c/f(n) - B^{e}(n)$ whenever $\delta(n) > 0$, i.e., whenever $F^{e}(n+1)$ is clipped to 0 in Eq. (66). In fact, given enough time they may eventually become tight.

[0111] Note that whenever the bounds tighten by $\delta(n) > 0$, the control target must be shifted by $\Delta g(n)/\tilde{r}_a$, where $\Delta g(n) = -\delta(n)$. Furthermore, whenever n is the first frame of a new coding bit rate, the control target should be shifted by $\Delta g(n)/\tilde{r}_a$, where $\Delta g(n) = \hat{g}^{new}(n)-\hat{g}^{old}(n)$. Here, $\hat{g}^{old}(n)$ can be determined by running Eqs. (65), (66) and (67) for one extra step, namely if n is the first frame of the new coding bit rate,

$$\hat{F}^{e}(n) = \max\{0, \hat{B}^{e}(n-1) - \hat{r}_c^{old}/f(n-1)\} \qquad (68)$$

$$\hat{B}^{e}(n) = \hat{F}^{e}(n) + b(n) \qquad (69)$$

$$\hat{g}^{old}(n) = B - \hat{B}^{e}(n). \qquad (70)$$

It is easy to see that if $\hat{g}^{new}(n) = B - b(n)$, then $\Delta g(n)=\hat{F}^{e}(n)$ as computed in Eq. (68).

VII. Sender-Driven Streaming

[0112] The preceding discussions have assumed that the control process for adapting the stream or its bit rate to the prevailing conditions is located within the client. This is called *receiver-driven* streaming because the client (i.e., the receiver) makes the decisions and informs the server (i.e., the sender). However, it should be clear that sender-driven streaming is also possible, in which the control process operates at the sending side. For example, using TCP for transmission of the streaming media data, the control process can operate at the server, measuring the transmission times of each frame and estimating the transmission rate, instead of the control process operating at the client, measuring the arrival times of each frame and estimating the arrival rate. Using TCP, the transmission time of a frame is substantially the same as its arrival time, and likewise the transmission rate is substantially the same as its arrival rate; hence these can be used interchangeably. Generally, the transmission time or the arrival time of a packet may be called a time stamp. In sender-driven streaming, the server can infer the state of the client buffer by knowing what it has already transmitted and by knowing the playout times for the transmitted frames. Sender-driven streaming reduces the overhead of communicating control information between the client and the server, but it also increases the computational burden on the server. In some situations, this may be a desirable trade-off. Those skilled in the art will appreciate that it is possible to appropriately modify the protocols described herein to cases where the control algorithm resides in the server or in

Ex. 1018-256

another location, rather than in the client.

VIII. References

[0113]

[1] G.J. Conklin, G.S. Greenbaum, K.O. Lillevold, A.F. Lippman, and Y.A. Reznik. Video coding for streaming media delivery on the Internet. *IEEE Trans. Circuits and Systems for Video Technology,* 11 (3):269-281, March 2001. special issue on Streaming Video.

[2] T. Wiegand and G. Sullivan. Joint video specification rec. h.264 & 14496-10 avc. Non-Final Draft of Final Draft International Standard (FDIS) JVT-G050, ITU-T & ISO/IEC, Pattaya, Thailand, March 2003.

[3] V. J. Ribeiro, R. H. Riedi, J. Navratil, L. Cottrell, and R. G. Baraniuk. pathchirp: efficient available bandwidth estimation for network paths. In *Proc. Passive and Active Measurement Workshop (PAM)*, La Jolla, CA, April 2003.

[4] R. Rejaie, M. Handley, and D. Estrin. Layered quality adap-tation for Internet streaming video. *IEEE J. Selected Areas in Communications,* 18(12):2530-2543, December 2000.

[5] Q. Zhang, Y.-Q. Zhang, and W. Zhu. Resource allocation for multimedia streaming over the Internet. *IEEE Trans. Multimedia*, 3(3):339-355, September 2001.

[6] B.-J. Kim, Z. Xiong, and W. A. Pearlman. Low bit-rate scalable video coding with 3D set partitioning in hierarchical trees (3-D SPIHT). *IEEE Trans. Circuits and Systems for Video Technology,* 10(8):1374-1387, December 2000.

[7] F. Wu, S. Li, and Y.-Q. Zhang. A framework for efficient progressive fine granularity scalable video coding. *IEEE Trans. Circuits and Systems for Video Technology,* 11(3):301-317, March 2001.

[8] J. Li. Embedded audio coding (eac) with implicit psychoacoustic masking. In *Proc. Int'l Conf. Multimedia,* pages 592-601, Nice, France, December 2002. ACM.

[9] P. de Cuetos and K. W. Ross. Adaptive rate control for streaming stored fine-grained scalable video. In *Proc. Int'l Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDA V)*, Miami Beach, FL, May 2002.

[10] J. Ribas-Corbera, P. A. Chou, and S. Regunathan. A generalized hypothetical reference decoder for H.264/AVC. *IEEE Trans. Circuits and Systems for Video Technology*, 13(7), July 2003.

[11] B. D. O. Anderson and J. B. Moore. *Optimal Control: Linear Quadratic Methods*. Prentice Hall, 1990.

[12] S. Keshav. Packet-pair flow control. http://www.cs.cornell.edu/-skeshav/papers.html.

[13] P. A. Chou and Z. Miao, "Rate-distortion optimized streaming of packetized media," *Microsoft Research Technical Report MSR-TR-2001-35*, February 2001

Claims

1. A computer-implemented process for controlling a coding bit rate of streaming media data being transmitted to a client from a server over a computer network, comprising performing the following process actions:

   employing a linear quadratic control technique to establish, on an ongoing basis, a current coding bit rate for the streaming media data, which is estimated to provide a high quality playback of the streaming media data while still keeping a decoder buffer of the client used to receive streaming media data from the server filled to a desired duration level so as to reduce the chance of an underflow condition which would result in an interruption of the playback of the streaming media data by the client;
   using each coding bit rate established to control said rate for the streaming media data.

2. The process of Claim 1, wherein the process action of establishing the current coding bit rate comprises the actions of:

determining on a frame-by-frame basis the coding bit rate which is estimated to keep the client buffer level approximately at the desired target level; and
establishing a new coding bit rate only when it is determined the coding bit rate estimate will not change in relation to the last established rate by more than a desired degree.

3. The process of Claim 2, wherein the client buffer level is determined by a leaky bucket model of the streaming media data at the estimated coding bit rate.

4. The process of Claim 3, wherein the process action of determining on a frame-by-frame basis the coding bit rate which is estimated will keep the client buffer approximately at the desired level, comprises an action of identifying a coding bit rate that reduces a difference between an estimated latest anticipated arrival time of the frame under consideration and a prescribed target arrival time.

5. The process of Claim 4, wherein the process action of determining on a frame-by-frame basis the coding bit rate which is estimated will keep the client buffer approximately at the desired level, comprises an action of reducing the change in the coding bit rate to a prescribed degree, using a quadratic cost function.

6. The process of Claim 5, wherein the process action of estimating the latest anticipated arrival time of the frame under consideration, comprises the actions of:

re-estimating the latest anticipated arrival time associated with the frame immediately preceding the frame under consideration if it were to exhibit the current coding bit rate; and
adding said re-estimated time to the current coding bit rate divided by the product of a current instantaneous frame rate and an estimated arrival rate of the frames.

7. The process of Claim 6, wherein the process action of re-estimating the latest anticipated arrival time associated with the frame immediately preceding the frame under consideration comprises adding the actual arrival time of said immediately preceding frame as measured by the client, to an upper bound gap of an encoder buffer of the server for the immediately preceding frame divided by the estimated arrival rate associated with said immediately preceding frame, wherein the upper bound gap of a frame is defined as the number of data bits that the server's encoder buffer can contain over the bits currently contained therein after the frame is fully input into the encoder buffer.

8. The process of Claim 7, wherein the process action of establishing the current coding bit rate is accomplished by the client, and wherein the client estimates the latest anticipated arrival time of the frame under consideration based on information received from the server concerning the state of the encoder buffer.

9. The process of Claim 8, wherein the information received from the server comprises the current coding bit rate of the streaming media data being received by the client which is received as part of the first frame of the sequence of frames exhibiting that coding bit rate.

10. The process of Claim 9, further comprising, an action of estimating the upper bound gap for each frame received by the client, said estimating comprising:

computing what the encoder buffer fullness value would have been after the last received frame was placed therein as the sum of the last computed value of the encoder buffer fullness prior to insertion of the last received frame and the size of that last received frame,

wherein the encoder buffer fullness value prior to insertion of the last received frame associated with the first frame received by the client at the current coding bit rate of the streaming media is assumed to be zero regardless of its actual value,
and wherein the encoder buffer fullness value prior to insertion of the last received frame associated with each subsequent frame received by the client at the current coding bit rate is deemed to be either zero, or the difference between the last computed value for the encoder buffer fullness after the insertion of the last received frame and the coding bit rate divided by the instantaneous frame rate, whichever is larger, wherein the instantaneous frame rate is equal to the reciprocal of a scheduled playback time of the next frame to be received less a scheduled playback time of the last received frame; and

computing the upper bound gap associated with the last received frame as the encoder buffer size less the last computed value for the encoder buffer fullness after insertion of the last received frame.

11. The process of Claim 10, wherein the encoder buffer size varies depending on the coding bit rate and is received by the client from the server as a preamble to the streaming media data for all coding bit rates available from the server.

12. The process of Claim 10, further comprising, whenever a frame is received by the client which has a different coding bit rate than the immediately preceding frame, the actions of:

shifting the currently scheduled target arrival time associated with the just received frame by a shift value, wherein said shift value represents a difference between the upper bound gap of the frame and the upper bound gap that would be associated with the frame had it been coded at the immediately preceding coding bit rate;
shifting the currently scheduled target arrival times for future frames such that they, collectively approach over a period of time, and eventually coincide with, the previous target arrival times for those frames; and
shifting each shifted frame by the difference between the encoder buffer fullness value after insertion of the last received frame and the current coding bit rate, divided by the frame rate, and then divided by the estimated arrival rate associated with the last received frame.

13. The process of Claim 12, wherein the upper bound gap that would be associated with the frame had it been coded at the immediately preceding coding bit rate is computed as the encoder buffer size less the last computed value for the encoder buffer fullness value after insertion of the last received frame, wherein the encoder buffer fullness value after insertion of the last received frame is computed as the sum of the last computed value of the encoder buffer fullness value prior to insertion of the last received frame and the size of that last received frame, and wherein the encoder buffer fullness value prior to insertion of the last received frame is computed as either zero, or the difference between the value computed for the encoder buffer fullness after the insertion of the frame received prior to the last received frame and the immediately preceding coding bit rate divided by the instantaneous frame rate associated with the frame received prior to the last received frame, whichever is larger.

14. The process of Claim 8, wherein the information received from the server comprises:

the current coding bit rate of the streaming media data being received by the client which is received as part of the first frame of the sequence of frames exhibiting that coding bit rate; and
the current upper bound gap which is received as part of every frame.

15. The process of Claim 8, wherein the information received from the server comprises:

the current coding bit rate of the streaming media data being received by the client which is received as part of the first frame of the sequence of frames exhibiting that coding bit rate; and
the current upper bound gap associated with the first frame of the sequence of frames exhibiting the current coding bit rate, said current upper bound gap also being received as part of the first frame of said sequence.

16. The process of Claim 15, wherein the information received from the server further comprises a shift value which is received as part of the first frame of the sequence of frames exhibiting a coding bit rate that is different from the immediately preceding frame, said shift value representing a difference between the upper bound gap that would be associated with the immediately preceding frame received by the client had it been coded at the current coding bit rate and the upper bound gap actually associated with the immediately preceding frame coded at the previous coding bit rate.

17. The process of Claim 16, further comprising an action of whenever a frame is received by the client which has a different coding bit rate than the immediately preceding frame, shifting the currently scheduled target arrival time associated with the just received frame by the shift value and shifting the currently scheduled target arrival times for future frames such that they collectively approach over a period of time and eventually coincide with the previous target arrival times for those frames, said shifting of the target values being accomplished so as to prevent the client from attempting to establish new coding bit rates based on the shift in the upper bound gap between the old and new coding bit rates rather than a change in the arrival rate of the frames.

18. The process of Claim 15, further comprising, for each frame received by the client that is not the first frame of the sequence of frames exhibiting the current coding bit rate, an action of estimating the upper bound gap.

19. The process of Claim 6, wherein computing the estimated arrival rate of the frames of the streaming media data comprises:

computing, on a packet-by-packet basis, the product of the estimated arrival rate computed for the immediately preceding packet to the currently received packet and a first fractional weighting factor, added to the product of the instantaneous arrival rate of the currently received packet and a second fractional weighting factor, wherein at least one of the fractional weighting factors is not a constant but instead based on the time between the packets.

20. The process of Claim 5, wherein the process action of identifying the coding bit rate that reduces the difference between the estimated latest anticipated arrival time of the frame under consideration and the prescribed target arrival time, while at the same time reducing the change in the coding bit rate to a prescribed degree, comprises an action of choosing the prescribed target arrival time for the frame under consideration so as to be a prescribed amount of time earlier than a scheduled playback time for that frame.

21. The process of Claim 20, wherein the process action of choosing the prescribed target arrival time for the frame under consideration comprises an action of making said prescribed amount of time large enough after a startup period that network jitter, delays and throughput changes which may cause the actual arrival time of the frame to be after its target arrival time do not result in the frame arriving after its scheduled playback time.

22. The process of Claim 21, wherein the process action of choosing the prescribed target arrival time for the frame under consideration further comprises making the target arrival time closer to the playback time during the startup period than afterwards for those frames arriving during that period to assist in reducing a startup delay between when the streaming media data begins to be received by the client and when the first frame is played back.

23. The process of Claim 22, wherein the process action of choosing the prescribed target arrival time for the frame under consideration further comprises setting the target arrival time for each frame of the streaming media using a logarithmic target schedule.

24. The process of Claim 22, wherein the process action of choosing the prescribed target arrival time for the frame under consideration further comprises setting the target arrival time for each frame of the streaming media using a two-piece linear target schedule wherein the difference between the target arrival time and the playback time for a frame arriving during the startup period increases linearly to a prescribed amount of time that is large enough to account for said network jitter, delays and throughput changes after which the difference remains substantially constant.

25. The process of Claim 22, further comprising a process action of reducing the startup delay, said reducing action comprising beginning playback of the first frame of the streaming media data as early as when a decoder buffer of the client has a minimum amount of data therein that is needed to ensure an underflow condition does not occur at an initial coding bit rate.

26. The process of Claim 25, wherein the process action of beginning playback of the first frame of the streaming media data as early as when the decoder buffer of the client has a minimum amount of data therein, comprises the actions of:

obtaining an initial encoder buffer fullness value associated with said first frame, wherein the initial encoding buffer fullness value is defined as the size of the encoder buffer less the size of the first frame and less the upper bound gap associated with the first frame;
establishing said minimum amount of data as the encoder buffer size associated with the initial coding bit rate less an initial encoder buffer fullness.

27. The process of Claim 26, wherein the initial coding bit rate, initial encoder buffer fullness, and encoder buffer size are received from the server as a preamble to the streaming media data.

28. The process of Claim 26, wherein the process action of reducing the startup delay further comprises setting the initial coding bit rate to a level that is less than the anticipated arrival rate of the streaming media data at the client since said minimum amount of data decreases as the arrival rate increases relative to the coding bit rate and the startup delay is substantially said minimum amount of data divided by the arrival rate so if the coding bit rate is less than the arrival rate, said minimum amount of data required may be smaller and the startup delay may be less.

29. The process of Claim 5 wherein the process action of establishing the current coding bit rate comprises an action of initiating the establishment of coding bit rates only after a startup period, wherein said startup period is defined as the period of time prior to the first instance of an estimated latest arrival time of a frame under consideration computed for the initial coding bit rate being earlier than a target arrival time for that frame.

30. The process of Claim 5, wherein the prescribed degree to which the change in the coding bit rate is reduced varies depending on whether the coding bit rate increased, in which case the prescribed degree to which any future change in the coding bit rate is reduced is made greater in comparison to the prescribed degree in cases where the last change in the coding bit rate decreased the rate.

31. The process of Claim 3, wherein the process action of establishing a current coding bit rate for the streaming media data, comprises, in the case where the determined coding bit rate represents an increase from the immediately previous coding bit rate, establishing a new coding bit rate only if the new rate does not exceed the current estimated arrival rate of the frames.

32. The process of Claim 3, wherein the process action of establishing a current coding bit rate for the streaming media data, comprises, in the case where the determined coding bit rate represents an increase from the immediately previous coding bit rate which exceeds the current estimated arrival rate of the frames, establishing a new coding bit rate only if the current client buffer exceeds the desired level by an amount that it is estimated by the client will not be expended at the higher coding bit rate prior to the passing of a prescribed period of time.

33. The process of Claim 3, wherein the process action of establishing a current coding bit rate for the streaming media data, comprises, in the case where the determined coding bit rate represents a decrease from the immediately previous coding bit rate, establishing a new coding bit rate even if it represents a significant departure from the immediately previous coding bit rate.

34. A computer-implemented process for controlling a coding bit rate of streaming media data being transmitted to a client from a server over a computer network, comprising using the server to perform the following process actions:

generating a streaming media data stream which exhibits a coding bit rate supported by the server, and which after an initial startup period exhibits a coding bit rate related to a rate requested by the client;
computing parameters indicative of the state of an encoder buffer of the server, said encoder buffer being used to stage portions of the streaming media data prior to transmission to the client; and
providing the encoder buffer state parameters and the streaming media data stream to the client, whereby said parameters are usable by the client to establish, on an ongoing basis, a current coding bit rate for the streaming media data, which is estimated will provide a high quality playback of the streaming media data while still keeping a decoder buffer of the client used to receive streaming media data from the server filled to a desired duration level so as to reduce the chance of an underflow condition which would result in an interruption of the playback of the streaming media data by the client.

35. The process of Claim 34, wherein the process action of computing parameters indicative of the state of an encoder buffer, comprises an action of, for each coding bit rate supported by the server, computing a set of parameters defining an initial state of the encoder buffer as it would exist if a data stream corresponding to the coding bit rate where streamed therefrom, wherein said encoder buffer parameters comprise the coding bit rate associated with the data stream, the size of the encoder buffer employed with the coding bit rate of the data stream, and a value indicative of the initial encoder buffer fullness exhibited at the coding bit rate of the data stream.

36. The process of Claim 35, wherein the process action of providing the encoder buffer state parameters and the streaming media data stream to the client, comprises an action of providing the encoder buffer parameters computed for each data stream exhibiting one of the supported coding bit rates to the client as part of a preamble sent to the client prior to sending the streaming media data.

37. The process of Claim 35, wherein the size of the encoder buffer employed with the coding bit rate of the data stream is chosen so as to be the minimum size buffer that will still contain the data stream at any point in the streaming process given the coding bit rate and the initial encoder buffer fullness.

38. The process of Claim 35, further comprising a process action of the server computing an upper bound gap for each frame of the streaming media data generated for each coding bit rate supported by the server, wherein the upper

bound gap is defined as the number of data bits that the server's encoder buffer can contain over the bits currently contained therein after the just-generated frame is fully input into the encoder buffer.

39. The process of Claim 38, further comprising a process action of providing the upper bound gap computed for the first frame of a sequence of frames exhibiting a coding bit rate that is different from that associated with the frame generated immediately before the first frame of said sequence of frames, to the client along with an indication of the coding bit rate associated with the sequence of frames, said gap value and coding bit rate being provided with said first frame of the sequence.

40. The process of Claim 39, wherein the client estimates the upper bound gap for frames provided by the server that do not include the associated gap value.

41. The process of Claim 40, wherein the client employs the encoder buffer parameters and the upper bound gap values to determine on a frame-by-frame basis the coding bit rate that will keep the client buffer approximately at the desired level, said determining comprising identifying a coding bit rate that reduces a difference between an estimated latest anticipated arrival time of the frame under consideration and a prescribed target arrival time.

42. The process of Claim 39, further comprising the process actions of:

the server computing a shift value for each frame representing the first frame of the sequence of frames exhibiting a coding bit rate that is different from the frame generated immediately before this first frame of the sequence, said shift value representing a difference between the upper bound gap that would be associated with the frame generated immediately before the first frame of the sequence had it been encoded at the current coding bit rate and the upper bound gap actually associated with the frame generated immediately before the first frame of the sequence encoded at the previous coding bit rate; and
providing the shift value computed for the first frame of each sequence of frames exhibiting a coding bit rate that is different from the frame generated immediately before the first frame of the sequence, to the client with the data comprising this first frame.

43. The process of Claim 42, wherein whenever a frame is provided to the client which has a different coding bit rate than the immediately preceding frame, the currently scheduled target arrival time associated with the just received frame is shifted by the shift value provided with the just-received frame and the currently scheduled target arrival times for future frames are shifted such that they collectively approach over a period of time and eventually coincide with the previous target arrival times for those frames.

44. The process of Claim 38, wherein a limited number of coding bit rates are supported by the server, and wherein the process action of generating a streaming media data stream which exhibits a coding bit rate supported by the server, and which after an initial startup period exhibits a coding bit rate related to a rate requested by the client, comprises the actions of:

(a) receiving a coding bit rate request from the client;
(b) finding a supported coding bit rate that is equal to, or if none are equal, the closest smaller rate to, the requested coding bit rate;
(c) whenever a supported coding bit rate is found that is a lower rate than the coding bit rate associated with the last generated frame of the streaming media data, generating all future frames at that supported rate; and
(d) whenever a supported coding bit rate is found that is a higher rate than the coding bit rate associated with the last generated frame of the streaming media data, determining if a difference, between the upper bound gap associated with the last generated frame had it been encoded at the supported coding bit rate found and the upper bound gap associated with that frame encoded at the current coding bit rate, is less than or equal to a maximum allowable difference value,

whenever the difference is less than the maximum allowable difference value, generating all future frames at the found supported rate, and
whenever the difference is not less than the maximum allowable difference value, finding the next lower supported coding bit rate and repeat actions (c)-(d).

45. The process of Claim 44, wherein the maximum allowable difference value is received from the client along with the coding bit rate request.

46. The process of Claim 45, wherein the client chooses the maximum allowable difference value such that a latest anticipated arrival time associated with a frame provided by the server immediately prior to a frame under consideration by the client had it been coded at the requested coding bit rate is no more than a prescribed fraction of the way from that frame's target arrival time to its playback deadline.

Ex. 1018-264

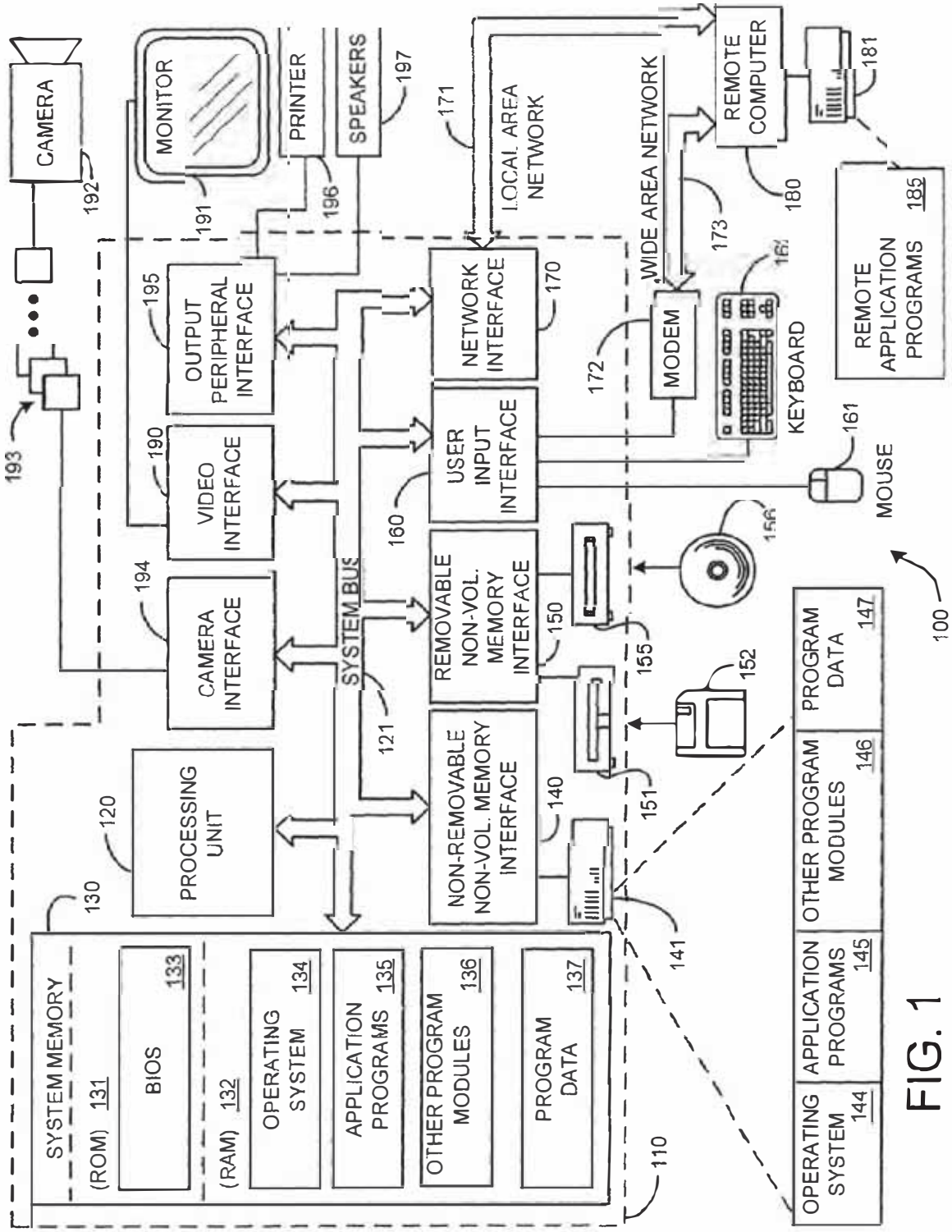


FIG. 1

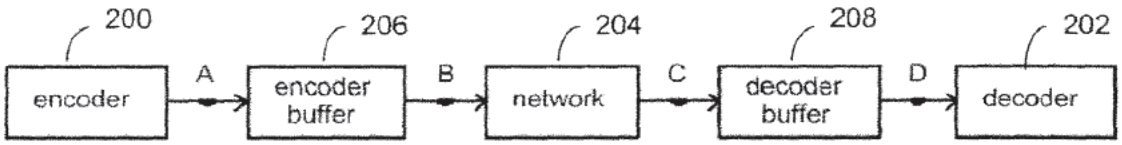


FIG. 2

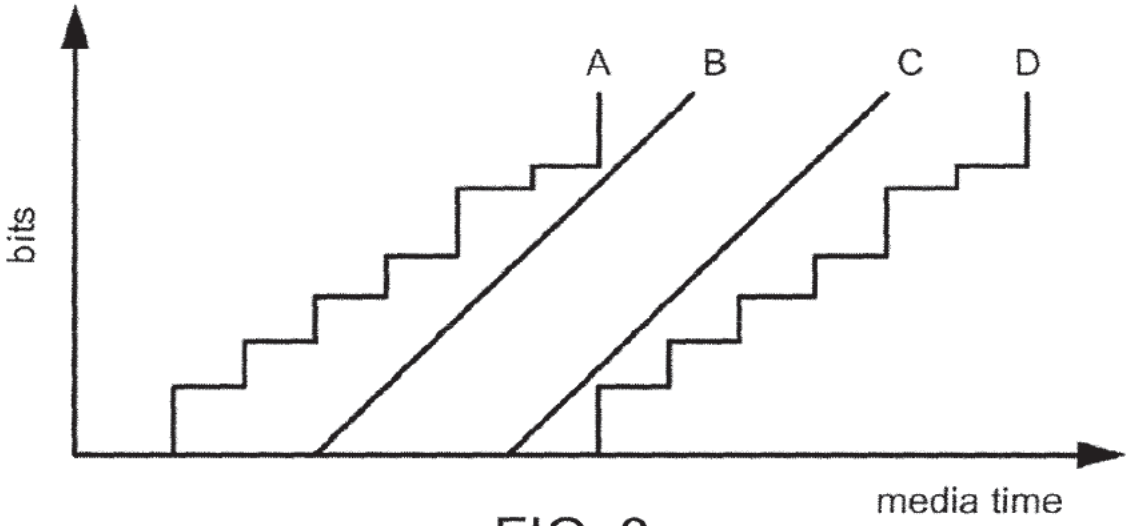


FIG. 3

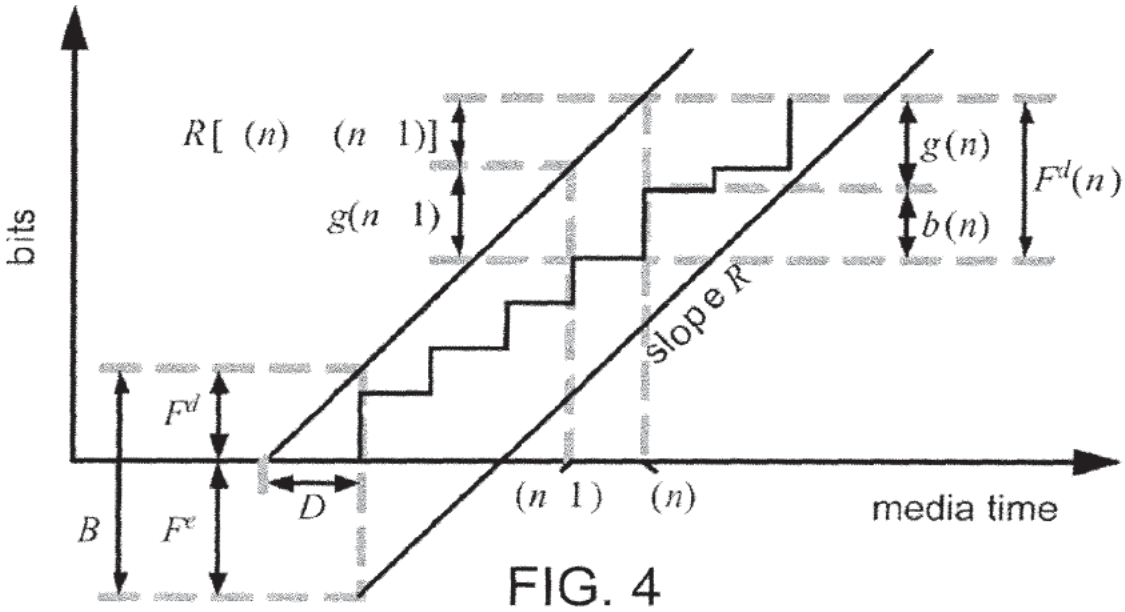


FIG. 4

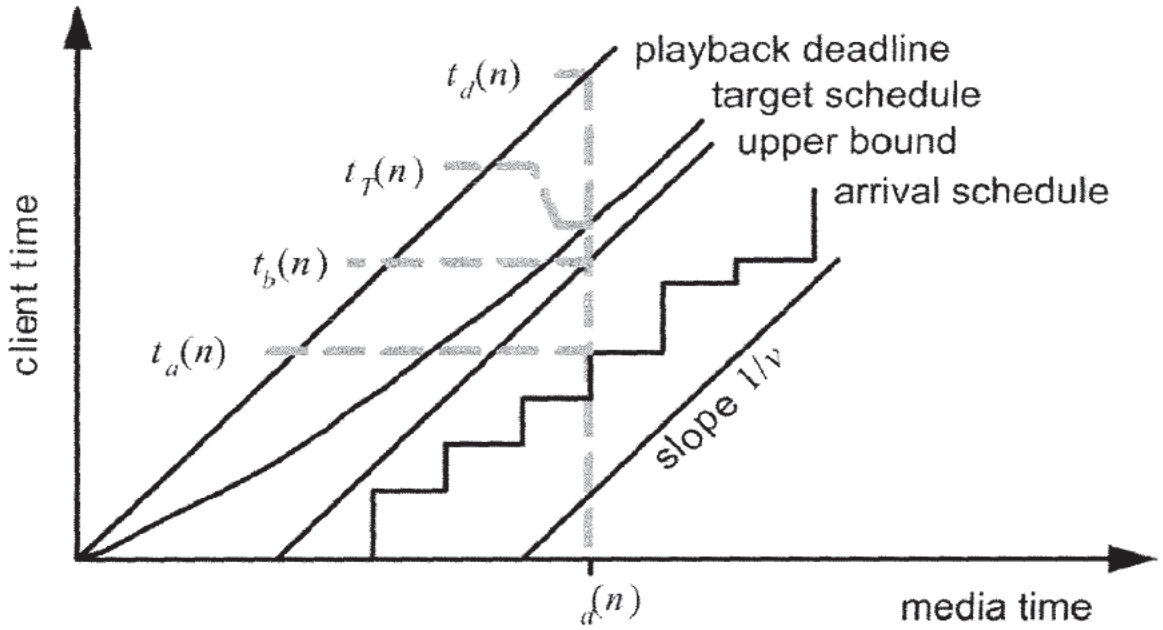

playback deadline
target schedule
upper bound
arrival schedule
$t_d(n)$
$t_T(n)$
$t_b(n)$
$t_a(n)$
client time
slope 1/ν
$_d(n)$
media time

FIG. 5

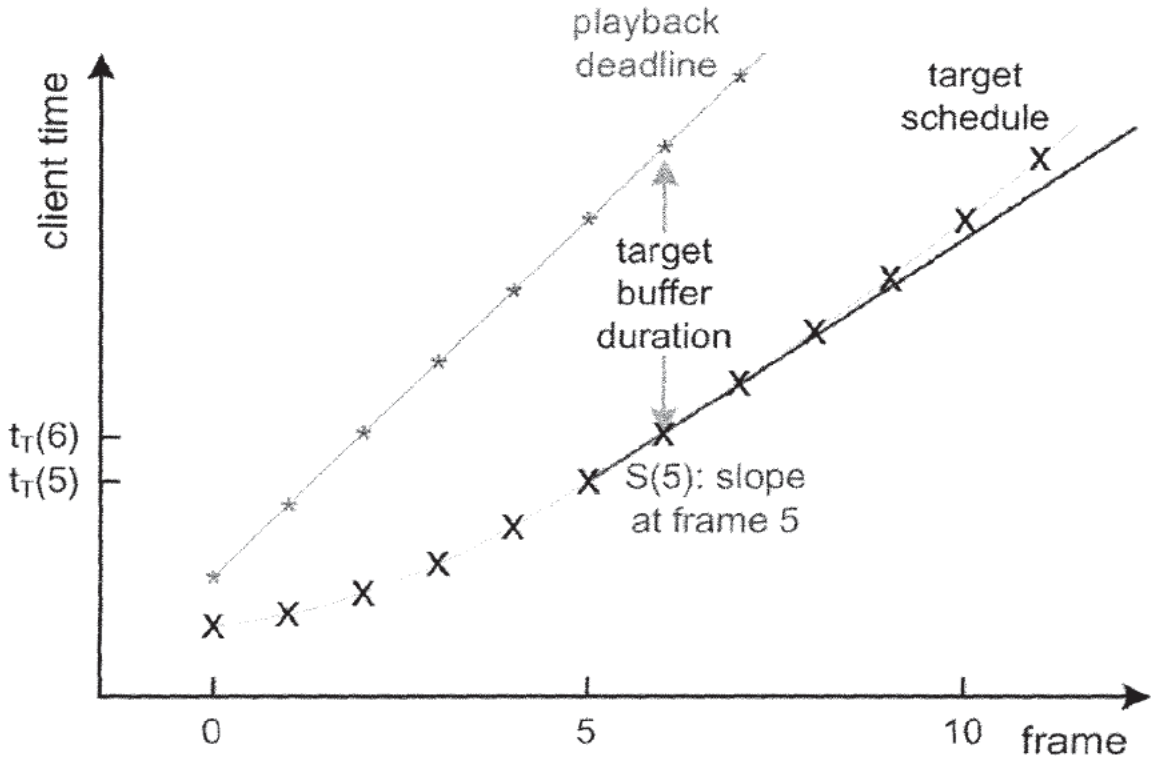

playback deadline
target schedule
client time
target buffer duration
$t_T(6)$
$t_T(5)$
S(5): slope at frame 5
0
5
10
frame

FIG. 6

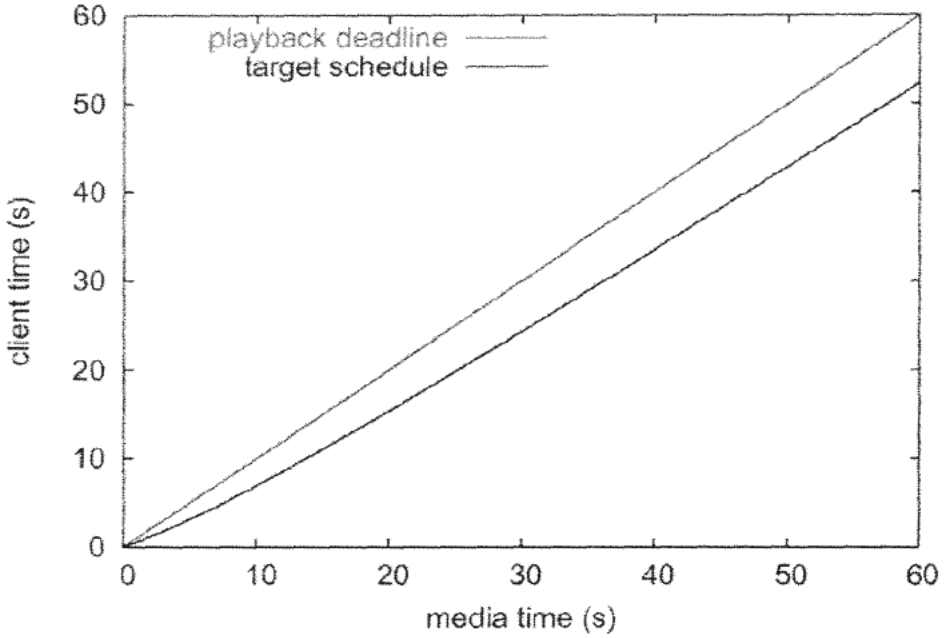

playback deadline
target schedule
client time (s)
media time (s)
0
10
20
30
40
50
60

FIG. 7(a)

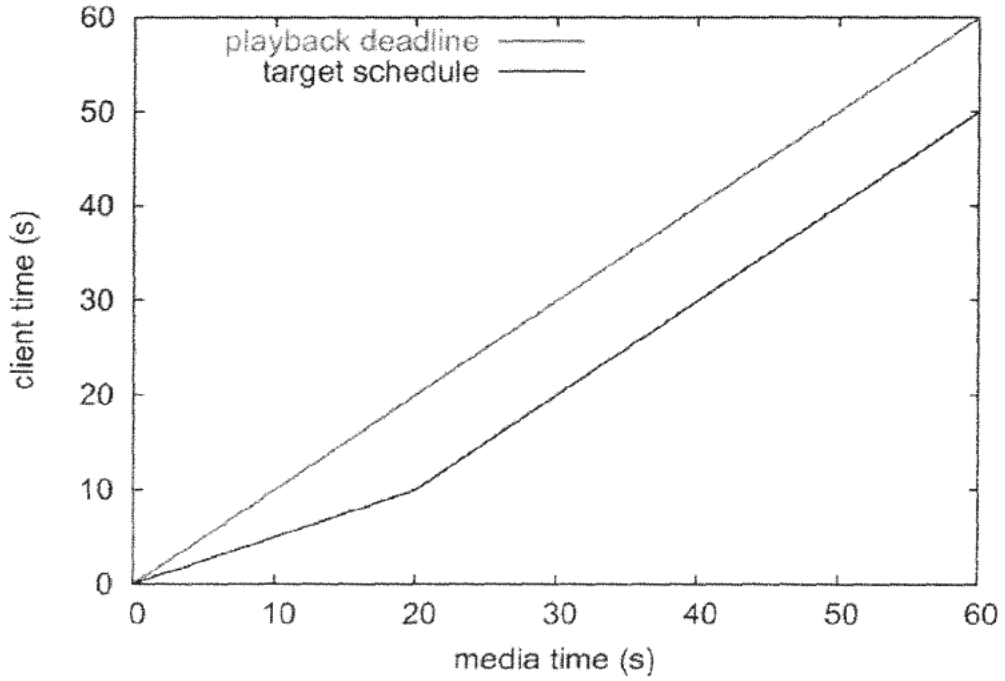

playback deadline
target schedule
client time (s)
media time (s)
0
10
20
30
40
50
60

FIG. 7(b)

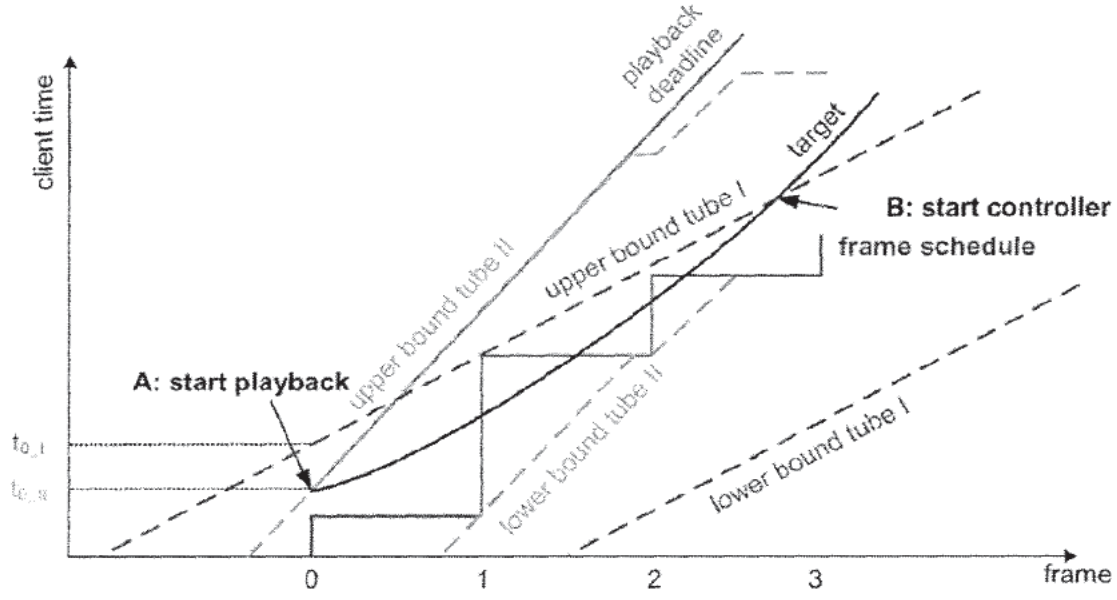

client time
playback deadline
target
B: start controller
frame schedule
upper bound tube I
upper bound tube II
A: start playback
lower bound tube II
lower bound tube I
0
1
2
3
frame

FIG. 8

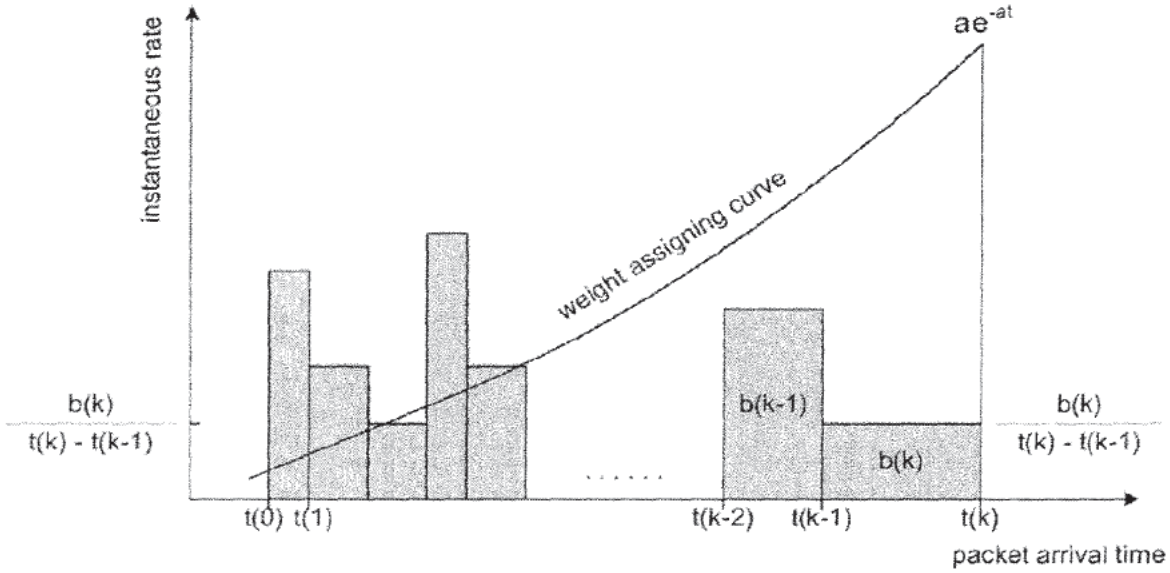

instantaneous rate
ae^{-at}
weight assigning curve
b(k)
t(k) - t(k-1)
b(k-1)
b(k)
b(k)
t(k) - t(k-1)
t(0) t(1)
t(k-2) t(k-1)
t(k)
packet arrival time

FIG. 9

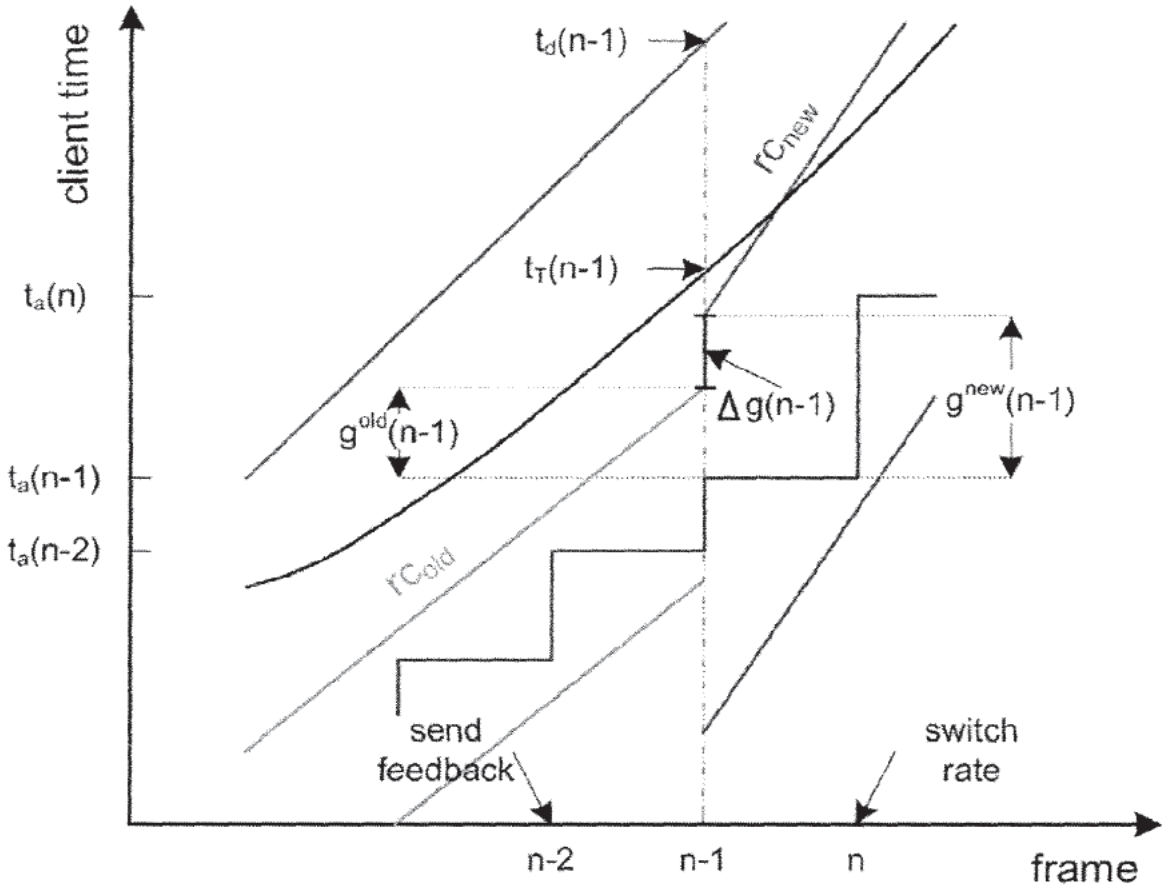

client time
$t_d(n-1)$
rC_{new}
$t_T(n-1)$
$t_a(n)$
$\Delta g(n-1)$
$g^{old}(n-1)$
$g^{new}(n-1)$
$t_a(n-1)$
$t_a(n-2)$
rC_{old}
send feedback
switch rate
n-2
n-1
n
frame

FIG. 10

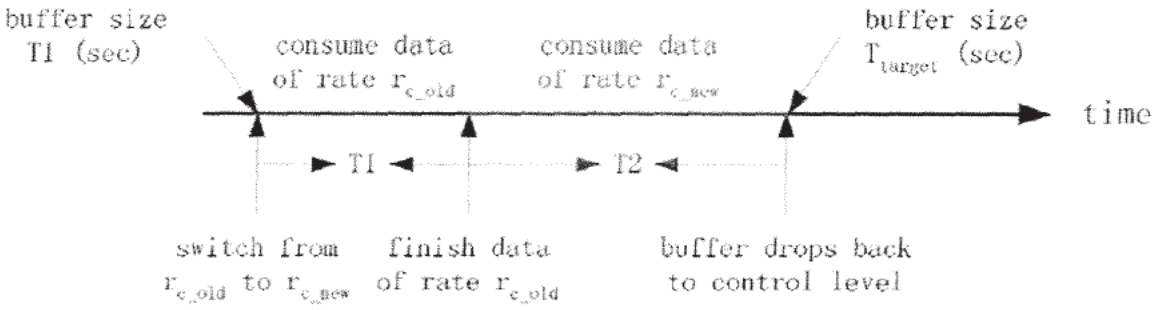

buffer size
T1 (sec)
consume data of rate r_{c_old}
consume data of rate r_{c_new}
buffer size
T_{target} (sec)
time
T1
T2
switch from r_{c_old} to r_{c_new}
finish data of rate r_{c_old}
buffer drops back to control level

FIG. 12

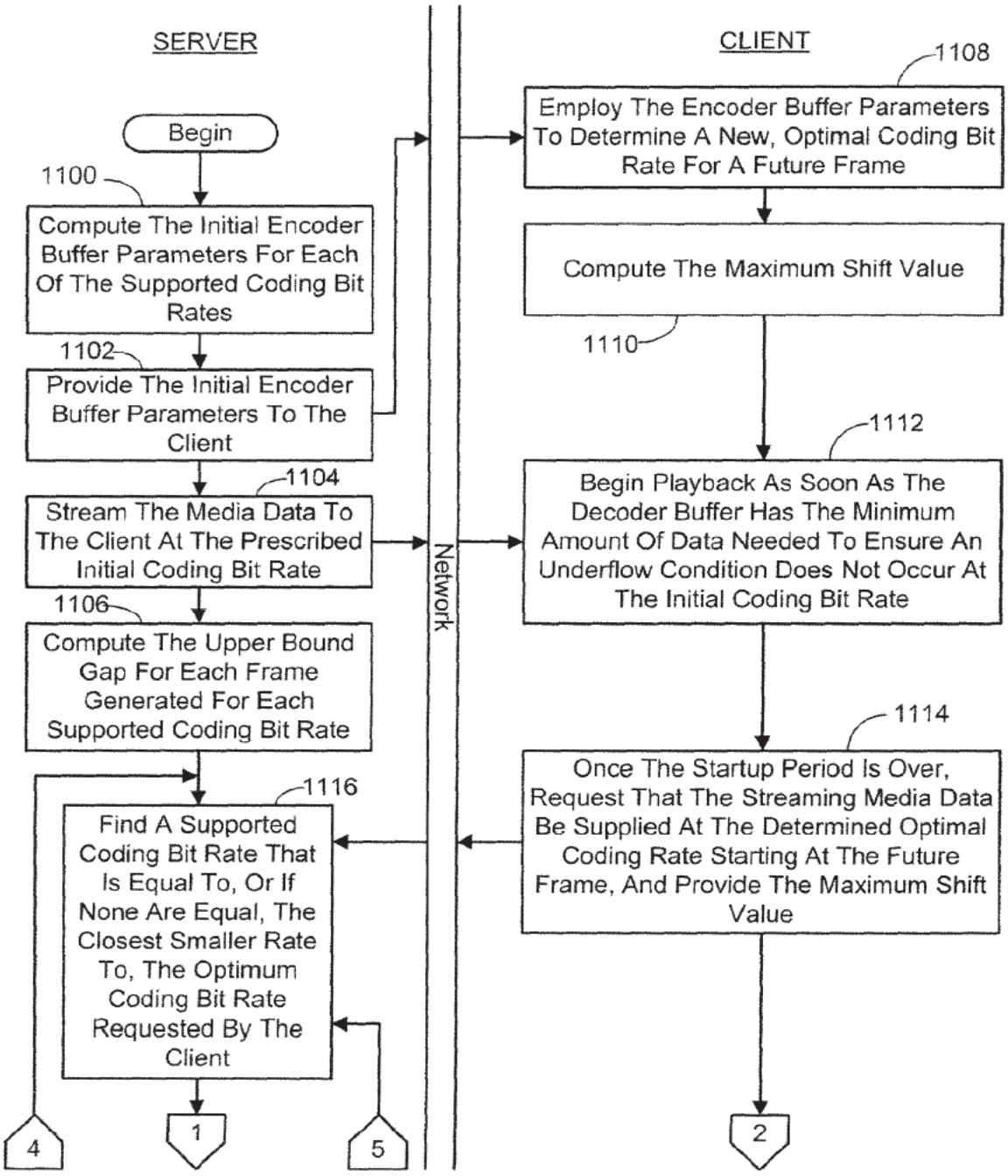

SERVER
CLIENT
Begin
1100
Compute The Initial Encoder Buffer Parameters For Each Of The Supported Coding Bit Rates
1102
Provide The Initial Encoder Buffer Parameters To The Client
1104
Stream The Media Data To The Client At The Prescribed Initial Coding Bit Rate
1106
Compute The Upper Bound Gap For Each Frame Generated For Each Supported Coding Bit Rate
1116
Find A Supported Coding Bit Rate That Is Equal To, Or If None Are Equal, The Closest Smaller Rate To, The Optimum Coding Bit Rate Requested By The Client
Network
1108
Employ The Encoder Buffer Parameters To Determine A New, Optimal Coding Bit Rate For A Future Frame
Compute The Maximum Shift Value
1110
1112
Begin Playback As Soon As The Decoder Buffer Has The Minimum Amount Of Data Needed To Ensure An Underflow Condition Does Not Occur At The Initial Coding Bit Rate
1114
Once The Startup Period Is Over, Request That The Streaming Media Data Be Supplied At The Determined Optimal Coding Rate Starting At The Future Frame, And Provide The Maximum Shift Value
4
1
5
2

FIG. 11A

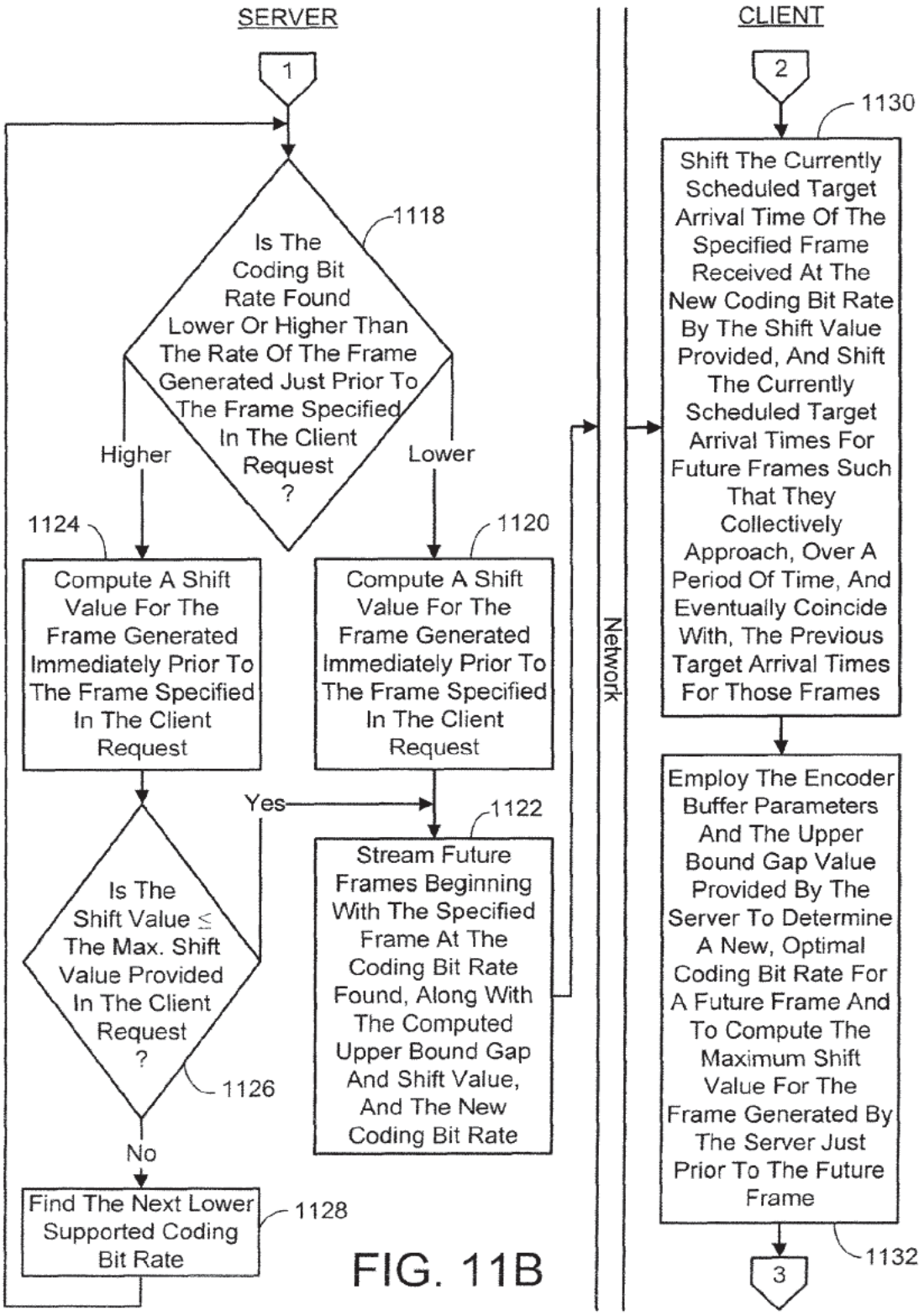

SERVER
CLIENT
1
2
1118
Is The Coding Bit Rate Found Lower Or Higher Than The Rate Of The Frame Generated Just Prior To The Frame Specified In The Client Request ?
Higher
Lower
1124
Compute A Shift Value For The Frame Generated Immediately Prior To The Frame Specified In The Client Request
1120
Compute A Shift Value For The Frame Generated Immediately Prior To The Frame Specified In The Client Request
Yes
1122
Stream Future Frames Beginning With The Specified Frame At The Coding Bit Rate Found, Along With The Computed Upper Bound Gap And Shift Value, And The New Coding Bit Rate
Is The Shift Value ≤ The Max. Shift Value Provided In The Client Request ?
1126
No
Find The Next Lower Supported Coding Bit Rate
1128
Network
1130
Shift The Currently Scheduled Target Arrival Time Of The Specified Frame Received At The New Coding Bit Rate By The Shift Value Provided, And Shift The Currently Scheduled Target Arrival Times For Future Frames Such That They Collectively Approach, Over A Period Of Time, And Eventually Coincide With, The Previous Target Arrival Times For Those Frames
Employ The Encoder Buffer Parameters And The Upper Bound Gap Value Provided By The Server To Determine A New, Optimal Coding Bit Rate For A Future Frame And To Compute The Maximum Shift Value For The Frame Generated By The Server Just Prior To The Future Frame
1132
3

FIG. 11B

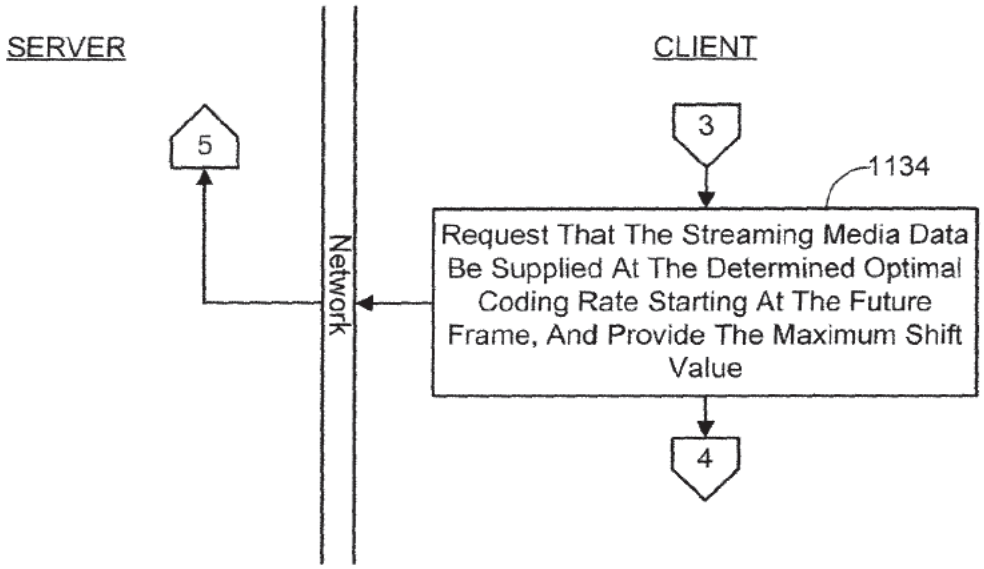

SERVER
CLIENT
5
3
1134
Network
Request That The Streaming Media Data Be Supplied At The Determined Optimal Coding Rate Starting At The Future Frame, And Provide The Maximum Shift Value
4

FIG. 11C

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
11/116,783	04/28/2005	R. Drew Major	8532P001	5535

8791 7590 09/11/2008
BLAKELY SOKOLOFF TAYLOR & ZAFMAN LLP
1279 OAKMEAD PARKWAY
SUNNYVALE, CA 94085-4040

EXAMINER
DONABED, NINOS J

ART UNIT	PAPER NUMBER
2144	

MAIL DATE	DELIVERY MODE
09/11/2008	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Interview Summary	Application No.	Applicant(s)	
	11/116,783	MAJOR ET AL.	
	Examiner	Art Unit	
	NINOS DONABED	2144	

All participants (applicant, applicant's representative, PTO personnel):

(1) *NINOS DONABED*.

(2) *Peling Shaw*.

(3)*Kevin Grange (reg. no. 60,793)*.

(4)_____.

Date of Interview: *09 September 2008*.

Type: a)☐ Telephonic b)☐ Video Conference
c)☒ Personal [copy given to: 1)☐ applicant 2)☒ applicant's representative]

Exhibit shown or demonstration conducted: d)☐ Yes e)☒ No.
If Yes, brief description: _____.

Claim(s) discussed: *1,11 and 35*.

Identification of prior art discussed: *Radford, U.S. Patent Publication 20020144276*.

Agreement with respect to the claims f)☐ was reached. g)☒ was not reached. h)☐ N/A.

Substance of Interview including description of the general nature of what was agreed to if an agreement was reached, or any other comments: *Applicant's Attorney presented proposed amendments to independent claims 1 and 11. He explained the differences between the invention and the prior art, i.e. Radford.*.

(A fuller description, if necessary, and a copy of the amendments which the examiner agreed would render the claims allowable, if available, must be attached. Also, where no copy of the amendments that would render the claims allowable is available, a summary thereof must be attached.)

THE FORMAL WRITTEN REPLY TO THE LAST OFFICE ACTION MUST INCLUDE THE SUBSTANCE OF THE INTERVIEW. (See MPEP Section 713.04). If a reply to the last Office action has already been filed, APPLICANT IS GIVEN A NON-EXTENDABLE PERIOD OF THE LONGER OF ONE MONTH OR THIRTY DAYS FROM THIS INTERVIEW DATE, OR THE MAILING DATE OF THIS INTERVIEW SUMMARY FORM, WHICHEVER IS LATER, TO FILE A STATEMENT OF THE SUBSTANCE OF THE INTERVIEW. See Summary of Record of Interview requirements on reverse side or on attached sheet.

/Ninos Donabed/ Examiner, Art Unit 2144	

Application Number	Application/Control No.	Applicant(s)/Patent under Reexamination
	11/116,783	MAJOR ET AL.
	Examiner	Art Unit
	PELING A. SHAW	2144

Attorney's Docket No. 8532P001 PATENT
Confirmation N.: 5535

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In Re Application of:

R. Drew Major

Application No.: 11/116,783

Filed: April 28, 2005

For: APPARATUS, SYSTEM AND METHOD FOR ADAPTIVE-RATE SHIFTING OF STREAMING CONTENT

Examiner: Donabed, Ninos J

Art Unit: 2144

Commissioner for Patents
P.O. Box 1450
Alexandria, Virginia 22313-1450

STATEMENT UNDER 37 C.F.R. §1.97(e)

Sir:

I hereby certify (as indicated by an "X" to the left of the appropriate paragraph):

X That each item of information contained in the accompanying Information Disclosure Statement was first cited in any communication from a foreign patent office in a counterpart foreign application not more than three months prior to the filing of the Information Disclosure Statement.

CERTIFICATE OF ELECTRONIC FILING

I hereby certify that this correspondence is being submitted electronically via EFS on the date shown below.

/Kevin O. Grange/ November 7, 2008
Kevin O. Grange (Date)

_____ That no item of information contained in the accompanying Information Disclosure Statement was cited in a communication from a foreign patent office in a counterpart foreign application, and, to the knowledge of the undersigned attorney after making reasonable inquiry, no item of information contained in the Information Disclosure Statement was known to any individual designated in §1.56(c) more than three months prior to the filing the Information Disclosure Statement.

Respectfully submitted,

BLAKELY, SOKOLOFF, TAYLOR & ZAFMAN LLP

Dated: November 7, 2008

/Kevin O. Grange/
Kevin O. Grange
Reg. No. 60,793

Customer No.: 8791
1279 Oakmead Parkway
Sunnyvale, CA 94085-4040
(408) 720-8300

Substitute for Form 1449/PTO

INFORMATION DISCLOSURE STATEMENT BY APPLICANT

(use as many sheets as necessary)

Complete if Known	
Application Number	11/116,783
Filing Date	April 28, 2005
First Named Inventor:	R. Drew Major
Art Unit	2144
Examiner Name	Donabed, Ninos J
Attorney Docket Number	8532P001

Sheet	**1**	**Of**	**1**

U.S. PATENT DOCUMENTS

Examiner Initials*	Cite No.[1]	Document Number Number-Kind Code[2] (If known)		Publication Date MM-DD-YYYY	Name of Patentee or Applicant of Cited Document	Pages, Columns, Lines, Where Relevant Passages or Relevant Figures Appear
		US-	2002/0174434 A1	11-21-2002	Lee et al.	
		US-	2003/0067872 A1	04-10-2003	Harrell et al.	
		US-	2004/0030797 A1	02-12-2004	Akinlar et al.	

FOREIGN PATENT DOCUMENTS

Examiner Initials*	Cite No.[1]	Foreign Patent Document Country Code[3] Number[4] Kind Code[5] (if known)	Publication Date MM-DD-YYYY	Name of Patentee or Applicant of Cited Document	Pages, Columns, Lines, Where Relevant Passages or Relevant Figures Appear	T[6]
		WO 00/67469	11-09-2000	Hannuksela (Nokia Mobil Phones Ltd.)		
		CA 2466482	05-22-2003	Doron et al. (Musicmatch, Inc.)		

NON-PATENT LITERATURE

Examiner Initials*			T[2]
		Supplemental European Search Report, September 30, 2008, (3 pages)	

Examiner Signature		Date Considered	

*Examiner: Initial if reference considered, whether or not citation is in conformance with MPEP 609. Draw line through citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

[1]Applicant's unique citation designation number (optional). [2]Applicant is to place a check mark here if English Translation is attached.

This collection of information is required by 37 CFR 1.98. The information is required to obtain or retain a benefit by the public which is to file (and by the USPTO to process) an application. Confidentiality is governed by 35 U.S.C. 122 and 37 CFR 1.14. This collection is estimated to take 2 hours to complete including gathering, preparing, and submitting the completed application form to the USPTO. Time will vary depending upon the individual case. Any comments on the amount of time you require to complete this form and/or suggestions for reducing this burden, should be sent to the Chief Information Officer, U.S. Patent and Trademark Office, P.O. Box 1450, Alexandria, VA 22313-1450. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS.

SEND TO: Commissioner for Patents, P.O. Box 1450, Alexandria, Virginia 22313-1450.

If you need assistance in completing the form, call 1-800-PTO-9199 (1-800-786-9199) and select option 2.

Electronic Acknowledgement Receipt

EFS ID:	4254762
Application Number:	11116783
International Application Number:	
Confirmation Number:	5535
Title of Invention:	Apparatus, system, and method for adaptive-rate shifting of streaming content
First Named Inventor/Applicant Name:	R. Drew Major
Customer Number:	08791
Filer:	Kevin Grange/Leslie Rogan
Filer Authorized By:	Kevin Grange
Attorney Docket Number:	8532P001
Receipt Date:	07-NOV-2008
Filing Date:	28-APR-2005
Time Stamp:	19:38:03
Application Type:	Utility under 35 USC 111(a)

Payment information:

Submitted with Payment	no

File Listing:

Document Number	Document Description	File Name	File Size(Bytes)/ Message Digest	Multi Part /.zip	Pages (if appl.)
1	Information Disclosure Statement (IDS) Filed (SB/08)	8532P001_IDS_and_SB08.pdf	39898 c9a53f39396816284a26e0409d40efdf7d57d666	no	3

Warnings:

Information:

This is not an USPTO supplied IDS fillable form					
2	Miscellaneous Incoming Letter	8532P001_1_97_e_Statement.pdf	15242 543a9acbbe9f7b3bac17bd003648088f1fff3e30	no	2
Warnings:					
Information:					
3	NPL Documents	CA_2466482.pdf	1402976 7b62bd54226f2e023d5148c062b3a73574da5375	no	42
Warnings:					
Information:					
4	NPL Documents	Supplemental_EP_Srch_Rpt_09-30-08.pdf	133357 61d4a680ee6482775b043ae19fa76f945956ea74	no	3
Warnings:					
Information:					
5	NPL Documents	WO_0067469.pdf	2369910 f480b1a1a30e55385b0d8dfd1638ac1e06614524	no	50
Warnings:					
Information:					
		Total Files Size (in bytes):	3961383		

This Acknowledgement Receipt evidences receipt on the noted date by the USPTO of the indicated documents, characterized by the applicant, and including page counts, where applicable. It serves as evidence of receipt similar to a Post Card, as described in MPEP 503.

New Applications Under 35 U.S.C. 111
If a new application is being filed and the application includes the necessary components for a filing date (see 37 CFR 1.53(b)-(d) and MPEP 506), a Filing Receipt (37 CFR 1.54) will be issued in due course and the date shown on this Acknowledgement Receipt will establish the filing date of the application.

National Stage of an International Application under 35 U.S.C. 371
If a timely submission to enter the national stage of an international application is compliant with the conditions of 35 U.S.C. 371 and other applicable requirements a Form PCT/DO/EO/903 indicating acceptance of the application as a national stage submission under 35 U.S.C. 371 will be issued in addition to the Filing Receipt, in due course.

New International Application Filed with the USPTO as a Receiving Office
If a new international application is being filed and the international application includes the necessary components for an international filing date (see PCT Article 11 and MPEP 1810), a Notification of the International Application Number and of the International Filing Date (Form PCT/RO/105) will be issued in due course, subject to prescriptions concerning national security, and the date shown on this Acknowledgement Receipt will establish the international filing date of the application.

(19) Canadian Intellectual Property Office
An Agency of Industry Canada

Office de la Propri,t, Intellectuelle du Canada
Un organisme d'industrie Canada

(11) CA 2 466 482 (13) A1

(40) 22.05.2003
(43) 22.05.2003

(12)

(21) 2 466 482

(22) 09.11.2002

(51) Int. Cl.[7]: H04J 3/16, H03M 7/00, G06F 12/02, H04N 5/92

(85) 07.05.2004

(86) PCT/US02/035903

(87) WO03/042783

(30) 60/337,939 US 09.11.2001
10/291,210 US 08.11.2002

(71) MUSICMATCH, INC.,
270-16935 West Bernardino Drive, SAN DIEGO , XX (US).

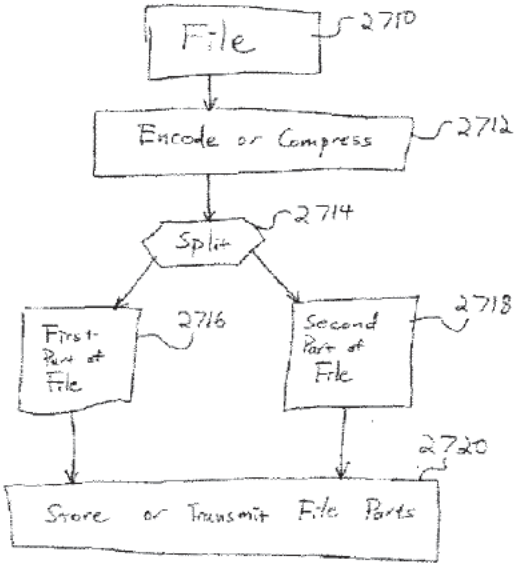
(72) DORON, AMIR (US).
DUNNING, TED E. (US).
GRUHNE, MATTHIAS (US).

(74) SIM & MCBURNEY

(54) DECOUPAGE DE FICHIERS, CODAGE ECHELONNABLE, ET TRANSMISSION ASYNCHRONE DANS UN TRANSFERT CONTINU DE DONNEES

(54) FILE SPLITTING SCALADE CODING AND ASYNCHRONOUS TRANSMISSION IN STREAMED DATA TRANSFER

(57) Files (2750) are divided into parts (2716, 2718) and at least some of the parts are transmitted to a client using a communication channel. At least some of the transmitted parts are cached locally. This allows subsequent streaming playback of the file while using less bandwidth by transmitting the part of the file that hasn't been cached, and combining the cached parts with the tramitted parts. In some embodiments, files may be represented at a low quality level by a first data set (2804), and at higher quality levels with additional data sets (2806). Data sets are cached locally, so that during subsequent streaming playback of the file, the quality level of the playback may be improved by sending additional data sets using bandwidth that would otherwise be dedicated to transmitting the cached data sets.



Office de la Propriété Intellectuelle du Canada
Un organisme d'Industrie Canada

Canadian Intellectual Property Office
An agency of Industry Canada

CA 2466482 A1 2003/05/22

(21) 2 466 482

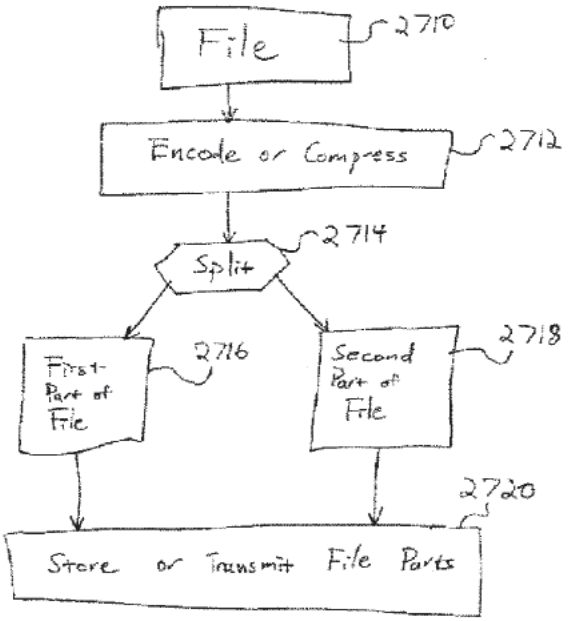
(12) DEMANDE DE BREVET CANADIEN
CANADIAN PATENT APPLICATION

(13) A1

(86) Date de dépôt PCT/PCT Filing Date: 2002/11/09
(87) Date publication PCT/PCT Publication Date: 2003/05/22
(85) Entrée phase nationale/National Entry: 2004/05/07
(86) N° demande PCT/PCT Application No.: US 2002/035903
(87) N° publication PCT/PCT Publication No.: 2003/042783
(30) Priorités/Priorities: 2001/11/09 (60/337,939) US; 2002/11/08 (10/291,210) US

(51) Cl.Int.[7]/Int.Cl.[7] H04J 3/16, G06F 12/02, H04N 5/92, H03M 7/00
(71) Demandeur/Applicant:
MUSICMATCH, INC., US
(72) Inventeurs/Inventors:
DUNNING, TED E., US;
DORON, AMIR, US;
GRUHNE, MATTHIAS, US
(74) Agent: SIM & MCBURNEY

(54) Titre : DECOUPAGE DE FICHIERS, CODAGE ECHELONNABLE, ET TRANSMISSION ASYNCHRONE DANS UN TRANSFERT CONTINU DE DONNEES
(54) Title: FILE SPLITTING SCALADE CODING AND ASYNCHRONOUS TRANSMISSION IN STREAMED DATA TRANSFER



(57) Abrégé/Abstract:
Files (2750) are divided into parts (2716, 2718) and at least some of the parts are transmitted to a client using a communication channel. At least some of the transmitted parts are cached locally. This allows subsequent streaming playback of the file while using less bandwidth by transmitting the part of the file that hasn't been cached, and combining the cached parts with the

(57) Abrégé(suite)/Abstract(continued):
tramitted parts. In some embodiments, files may be represented at a low quality level by a first data set (2804), and at higher quality levels with additional data sets (2806). Data sets are cached locally, so that during subsequent streaming playback of the file, the quality level of the playback may be improved by sending additional data sets using bandwidth that would otherwise be dedicated to transmitting the cached data sets.

BNSDOCID: <CA ..... 2466482A1 I_>

(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
22 May 2003 (22.05.2003)

PCT

(10) International Publication Number
WO 03/042783 A3

(51) International Patent Classification[7]: H04J 3/16, G06F 12/02, H04N 5/92, H03M 7/00

(21) International Application Number: PCT/US02/35903

(22) International Filing Date: 9 November 2002 (09.11.2002)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
60/337,939 9 November 2001 (09.11.2001) US
Not furnished 8 November 2002 (08.11.2002) US

(71) Applicant: MUSICMATCH, INC. [US/US]; 1635 W. Bernardino Drive, Suite 270, San Diego, CA 92127 (US).

(72) Inventors: DUNNING, Ted, E.; 11797 Rives Rim Road, San Diego, CA 92126 (US). DORON, Amir; 8455 Lake Ben Avenue, San Diego, CA 92119 (US). GRUHNE, Matthias; Schulstrasse 2a, 99718 Westgreussen (US).

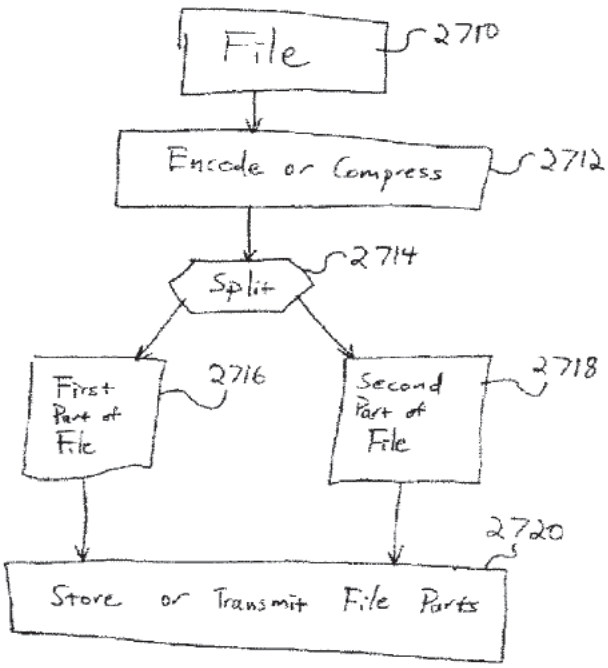
(74) Agents: PLIMIER, Michael, D. et al.; Fenwick & West LLP, Two Palo Alto Square, Palo alto, CA 94306 (US).

(81) Designated States (national): AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EC, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NO, NZ, OM, PH, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TN, TR, TT, TZ, UA, UG, UZ, VN, YU, ZA, ZM, ZW.

(84) Designated States (regional): ARIPO patent (GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE, SK,

[Continued on next page]

(54) Title: FILE SPLITTING SCALABLE CODING AND ASYNCHRONOUS TRANSMISSION IN STREAMED DATA TRANSFER



(57) Abstract: Files (2750) are divided into parts (2716, 2718) and at least some of the parts are transmitted to a client using a communication channel. At least some of the transmitted parts are cached locally. This allows subsequent streaming playback of the file while using less bandwidth by transmitting the part of the file that hasn't been cached, and combining the cached parts with the transmitted parts. In some embodiments, files may be represented at a low quality level by a first data set (2804), and at higher quality levels with additional data sets (2806). Data sets are cached locally, so that during subsequent streaming playback of the file, the quality level of the playback may be improved by sending additional data sets using bandwidth that would otherwise be dedicated to transmitting the cached data sets.

WO 03/042783 A3

WO 03/042783 A3

TR), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

(88) Date of publication of the international search report:
10 July 2003

Published:

— *with international search report*

— *before the expiration of the time limit for amending the claims and to be republished in the event of receipt of amendments*

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

FILE SPLITTING, SCALABLE CODING, AND ASYNCHRONOUS TRANSMISSION IN STREAMED DATA TRANSFER

Cross-Reference to Related Application

[0001] This application is a continuation-in-part of U.S. Patent Application Serial Number 09/846,823, filed on April 30, 2001, Patent No. X,XXX,XXX, the disclosure of which is incorporated by reference, and which claimed priority from U.S. Provisional Application Serial Number 60/201,622, filed May 3, 2000, the disclosure of which is incorporated by reference. The present application also claims priority from provisional U.S. Patent Application Serial No. 60/337,939, for "File Splitting, Scalable Coding, and Asynchronous Transmission in Streamed Data Transfer," filed November 9, 2001, the disclosure of which is incorporated by reference.

Background of the Invention

Field of the Invention

[0002] The present invention is related to delivery of streamed programs such as audio and video, and more particularly to systems and methods providing improved control, efficiency, and quality of such streamed transmissions using file splitting, scalable coding, and asynchronous transmission.

Description of the Background Art

[0003] Delivery of audio and video programs over the Internet provides several advantages over conventional broadcast media such as radio and television. Unlike broadcast media, which require users to receive programs at particular times, or to record them for later use, Internet delivery allows users to select and receive programs upon demand, at a time that is convenient to them. For example, a user may browse to a news story, presented in text format, and click on a link that initiates playback of an audio report on the news item. Users may also click on links to hear songs or song samples, as is common in e-commerce sites, such as amazon.com or cdnow.com, in order to permit users to sample songs before purchasing compact discs (CDs). Internet delivery of audio programming may also be used for implementing personalized radio stations, which deliver music tracks selected in response to the tastes of particular users. Similar functionality is available for the delivery of video programs on demand, allowing users to view sports highlights, news reports, music videos, and even films and television shows, over the Internet.

[0004] Software for playing back such audio and video files is known in the art, including for example the Windows Media Player from Microsoft Corporation, and the RealPlayer from Real Networks, Inc. Functionality for audio and video delivery may also be built into browsers such as Netscape Navigator and Microsoft Internet Explorer.

[0005] One disadvantage of Internet delivery of audio and video programs is the relatively limited bandwidth that is available for any particular program. Compared with broadcast media, such as radio, television (over-the-air or cable), and the like, typical Internet connections offer far less bandwidth and thus limit the amount of information that may be transmitted to the end user in a given period of time. Thus, even with the application of aggressive compression algorithms, sound quality and picture quality of streamed programs received over the Internet tend to be substandard. For example, due to the limited number of bits available, existing Internet delivery schemes fail to provide received audio programs at a quality approaching that of CDs.

BNSDOCID: <CA _____2466482A1_I_>

[0006] As an alternative to streamed programs, and in order to improve quality of the received programs, users may download programs and then, subsequent to downloading them, play the received program via their computers. Since the program is not being viewed or listened to in "real time," more time is available to transmit and receive the program, so that larger files may be provided, and better quality achieved. However, it may take many minutes, or even hours, to download a short program at a reasonable level of quality. Thus, such a scheme does not provide the user with an immediate listening or viewing experience; rather, the user must wait until the program is downloaded before it can be viewed or listened to.

Summary of the Invention

[0007] The present invention provides improved quality, security, and efficiency for streamed programs, by combining received streamed data with previously-stored (cached) data. In one embodiment, a program file containing program data such as audio, video, or the like, is split into two or more sections. In one embodiment, each section is not usable on its own to provide program output (e.g., audio signals where the program file is an audio file), but can only be used when recombined with the remaining sections. In one embodiment, one section of the split data is considerably smaller than the other section(s). This splitting allows the second section to be cached by a client machine without a danger that the client has a persistent usable copy of the original file. This allows the benefits of caching to be realized without providing users with unauthorized persistent copies of files.

[0008] For example, the second section may be transmitted in advance of the actual playback of a program file, thus avoiding bottlenecks that may result when the entire program file is transmitted upon user demand. Until the first section is transmitted, the user will not be able to play the program file. The smaller first section may be streamed to allow playback of the program file. The first section is discarded after playback so that no persistent usable copy of the program file remains on the user's machine. The second section remains cached, allowing future streaming playback to be performed efficiently. The ability to not provide a persistent usable

BNSDOCID: <CA_____2466482A1_I_>

copy of the program file can be advantageous because copyright holders may not wish to authorize such persistent copies, since the persistent copy could cannibalize revenues from sales of the program file in a tangible medium (such as audio CDs), or paid future downloads, streamed or otherwise, of the program file.

[0009] Alternatively, the program file is split into sections in order to provide scalability in the playback quality level. The first section is playable at a relatively low quality level. Additional sections of data may be added to the first section to increase the quality level. Thus, the program file may be downloaded and played at a quality level determined by the bandwidth of the connection. The sections may be cached so that when it is desired to play the program file again, additional sections may be downloaded and added for higher quality playback.

[0010] In some embodiments, a persistent copy of a playable version of the program file is not stored on the client. The additional sections are not playable without the first level. Thus, the additional sections may be cached, so that the file can be played back by combining a downloaded first section with cached additional sections.

[0011] The file splitting and scalable coding embodiments may also both be used on the same program file. This allows most of the file to be cached while still providing the server control over when the file may be played back. It also allows subsequent playbacks of the file after the first to be progressively higher quality as more of the scalable sections are transmitted with every playback.

Brief Description of the Drawings

[0012] Fig. 1A is a block diagram of a functional architecture.

[0013] Fig. 2 is a block diagram of a conceptual architecture for one embodiment.

[0014] Figs. 3A and 3B illustrate a method for splitting a program file according to one embodiment of the present invention.

[0015] Figs. 3C and 3D illustrate a method of receiving and playing a previously split program file according to one embodiment of the present invention.

BNSDOCID: <CA_____2466482A1_I_>

[0016] Fig. 4A illustrates an example of a scalably-coded program file.

[0017] Fig. 4B is a block diagram illustrating the use and combination of primary data and secondary data to provide different playback quality levels.

[0018] Figs. 4C and 4D depict examples of scalable coding according to one embodiment of the present invention.

[0019] Figs. 5A and 5B depict techniques for combining file splitting and scalable coding according to one embodiment of the present invention .

Detailed Description of the Preferred Embodiments

[0020] The following description of preferred embodiments of the present invention is presented in the context of an Internet-based jukebox or personalized radio station where it is desirable to stream music files. One skilled in the art will recognize that the present invention may be implemented in many other domains and environments, both within the context of musical files, and in other contexts. Accordingly, the following description, while intended to be illustrative of a particular implementation, is not intended to limit the scope of the present invention or its applicability to other domains and environments. Rather, the scope of the present invention is limited and defined solely by the claims.

Architecture Overview

[0021] Referring now to Fig. 2, there is shown a conceptual architecture of one embodiment of the present invention. In the architecture of Fig. 2, the invention is implemented in connection with a web-based "jukebox" 103, or personalized radio station.

[0022] Stream delivery system 150 interacts with jukebox 103 to specify a sequence of audio files to deliver to jukebox 103. In some embodiments, this jukebox 103 may run on a client such as a personal computer, while the stream deliver server system is part of a remote server, connected to the client via a network. Jukebox 103 transmits requests to stream delivery system 150, and stream delivery system 150 delivers the audio files, as tracks, to jukebox 103. Stream delivery system 150 also communicates with real-time subscription authorization module 157, which includes real-

BNSDOCID: <CA 2466482A1 I_>

time server 154 and database 156 that keep track of which user accounts are active and enforces global business rules about which accounts can listen to the radio at a given time. Within stream delivery system 150, there are a number of distinct software entities. Radio sequence generator 1613 receives requests from jukebox 103, receives format definitions 1611 and general constraints 1616, and receives recommendations from recommendation engine 107, to generate track selections to be transmitted to jukebox 103. The track selections generated by radio sequence generator 1613 specify which files to play according to estimated listener preferences as well as predetermined station formats. Authorization and content server 1614 keeps a record of the files that are selected by radio sequence generator 1613; server 1614 is consulted by radio sequence generator 1613 when files are requested. If generator 1613 does not provide the necessary security information, server 1614 flags this anomaly and declines to provide the data.

[0023] Referring now to Fig. 1A, there is shown a block diagram of a functional architecture for one embodiment of the present invention. Content index 110 provides a concise index of content stored in database 102, and is generated by conventional index generation means, to enable more efficient searching and updating of database 102.

Stream Delivery

[0024] In one embodiment, program files, such as audio files, are delivered to users as streamed data. The relationship discovery engine described in U.S. Patent Application Serial Number 09/846,823, filed on April 30, 2001, Patent No. X,XXX,XXX, describes one way to implement a personalized radio station, which may be used in conjunction with the present invention. The radio sequence transmitter 121 may deliver units of data to jukebox 103 in a format wherein each unit encodes a period of music. Since radio stations typically repeat their programming several times, it is beneficial to cache the data units in order to reduce the amount of transmitted data. In addition, if a sufficiently large time scale is used, different channels of the radio station may have considerable overlap among currently playing selections that are being delivered to various users. By identifying these common units, trans-

BNSDOCID: <CA_____2466482A1_I_>

mitter 121 can take advantage of further economies of transmission, so as to provide more efficient delivery of audio data.

[0025] In one embodiment, the transmitter 121 employs file splitting to improve the streaming performance, and provide streaming performance even in situations where channel capacity would otherwise be insufficient to provide the desired level of quality. In some embodiments, files are split prior to being stored in the content database 102 or with the compressed signal files 1615. In other embodiments, the files 101 are stored in the content database 102 or with the compressed signal files 1615 without first being split; the files are then split prior to being transmitted to a user for playback.

[0026] Fig. 3A is a flow diagram illustrating a method of splitting and storing a program file (such as a bitstream) containing program data such as audio, video, or the like, according to one embodiment. The process begins with the unsplit file 2710. Prior to being split, the file 2710 may be encoded and/or compressed 2712, although such operations are not essential to the present invention. Next the file 2710 is split 2714 into two or more parts, resulting in at least a first part 2716 of the file and a second part 2718 of the file. Particular techniques for splitting 2714 file 2710 are described in more detail in connection with Fig. 3B below. In one embodiment, the second part 2718 is substantially larger than the first part 2716. The file may be split 2714 into more than two parts. The parts 2716 and 2718 of the file 2710 are then stored 2720 in the content database 102 or with the compressed signal files 1615, or in some other location for later transmission. Alternatively, in an embodiment wherein files are split 2714 immediately prior to being transmitted, parts 2716 and 2718 are transmitted 2720 to an end user or remote computer for playback.

[0027] Fig. 3B is a block diagram illustrating an example of a method of splitting 2714 file 2710 into parts 2716, 2718 according to one embodiment. In the illustrated example, the file 2710 has M bits 2722-1 through 2722-M. The system generates first part 2716 of the file 2710 by taking N out of every P bits. The system generates second part 2718 of the file by taking the remaining P-N out of every P bits. In Fig. 3B, N is equal to one, and P is equal to four. Thus, the system assigns first bit 2722-1 to first part 2716, and assigns the next three bits 2722-2 through 2722-4 to sec-

BNSDOCID: <CA 2466482A1_I_>

ond part 2718. The system then assigns fifth bit 2722-5 to first part 2716 and the next three bits 2722-6 through 2722-8 to second part 2718. The system repeats this pattern until all the bits of the file 2710 have been assigned to one of first and second parts 2716, 2718. In short, then, the system divides file 2710 into sections, with the first bit (or some other bit) of each section being sent to the first part 2716, and the remaining bits of each section being sent to the second part 2718. In general, the splitting technique of the present invention results in two or more sections 2716, 2718 each of which cannot be played back on its own, but can be played only when recombined with the other section or sections.

[0028] In one embodiment, N is equal to one, and P is equal to twenty, so that the first part 2716 is approximately 5% the size of the original file 2710, and the second part 2718 is approximately 95% the size of the original file 2710.

[0029] One skilled in the art will recognize that the system of the invention is not limited to file-splitting by bits, but may split files according to other units, such as bytes, or simply an arbitrary unit size. Alternatively, other file-splitting techniques may be used. For example, the system can include all headers, compression table selectors and scale factors into the first section 2716, and include all encoded program data (such as audio or video) into the second part 2718. Other methods of splitting the file 2710 into two or more parts may also be used.

[0030] In general, it is desirable in most embodiments that each part of the file 2710 is not decodable or able to be played back on its own, but can only be decoded and/or played back when recombined with the remaining part(s). This allows a part to be handled, transmitted, cached, or otherwise processed, without concerns that a user may permanently store and reuse an unauthorized copy of the program. Since a part cannot be used independently to recreate a recognizable facsimile of the signal in the original file 2710, the part can be cached by a client machine without any danger that the client has been given an unauthorized, useable persistent copy of the program file 2710. The ability to not provide a persistent usable copy of the program file can be advantageous because copyright holders may not wish to authorize such persistent copies, since the persistent copy could cannibalize revenues from sales of

BNSDOCID: <CA 2466482A1 I >

the program file in a tangible medium (such as audio CDs), or paid future downloads, streamed or otherwise, of the program file.

[0031] Fig. 3C is a flow diagram illustrating a method of receiving and playing a previously split program file according to one embodiment of the present invention. When a file is to be transmitted from stream delivery system 150 to the jukebox 103, the jukebox 103 or server 1614 determines 2724 whether the second part 2718 of the file 2710 has previously been stored, or cached, locally and is available for retrieval. If the second part 2718 is not available locally for retrieval, jukebox 103requests transmission of both parts 2716, 2718 of the file 2710. In response, server 1614 retrieves the requested parts 2716, 2718 from the files 1615 or the content database 102 (depending on where the parts 2716, 2718 have been stored), and transmits both the parts 2716, 2718 of the file 2710 to the jukebox 103, which receives 2726 them. Alternatively, if the files 2710 are stored unsplit, server 1614 retrieves the file 2710 to be sent, splits the requested file according to techniques described above, and then transmits the parts 2716, 2718 to the jukebox 103, which receives 2726 them.

[0032] Upon receiving 2726 parts 2716, 2718, in one embodiment the jukebox 103 stores 2732 the second part 2718 locally, so as to avoid the need to retransmit the second part 2718 if it is needed in the future. The jukebox 103 combines 2728 both parts 2716, 2718, and plays back 2730 the recombined program file 2710. This reception 2726, combination 2728, and playback 2730 may all take place concurrently, for example according to a streamed transmission methodology. In one embodiment, as the bits arrive at the jukebox 103, they are combined 2728 and played back 2730; the jukebox 103 need not wait until the entirety of the parts 2716, 2718 are received 2726 before combining 2728 and playing them back 2730. In one embodiment, the recombined file 2710 and the first part 2716 are discarded 2731 after or during playback. Since the cached second part 2718 represents only a portion of the program file 2710 and in one embodiment cannot be decoded or played back on its own, the invention prevents users from retaining an unauthorized playable copy of the program file 2710. Ultimate control over the user's playback of the program file 2710 thus remains with the server 1614, since the stored second part 2718 cannot be used to play back of the file 2710 until the server 1614 retransmits the first part 2716.

BNSDOCID: <CA_____2466482A1_I_>

[0033] If the second part 2718 has been received and stored locally in a prior session, jukebox 103 requests transmission of only the first part 2716 of the file 2710. In response, server 1614 retrieves the requested first part 2716 from the files 1615 or the content database 102, and transmits the first part 2716 to the jukebox 103, which receives 2734 it. Alternatively, if the files 2710 are stored unsplit, server 1614 retrieves the file 2710 to be sent, splits the requested file according to techniques described above, and then transmits the first part 2716 of the file 2710 to the jukebox 103, which receives 2734 it. The jukebox 103 retrieves 2736 the second part 2718 from local storage and combines 2738 the first and second parts 2716, 2718 to reconstruct the file 2710. The jukebox 103 plays back 2740 the reconstructed program file 2710. In one embodiment, as the bits arrive at the jukebox 103, they are combined 2738 with the stored second part 2718 to form a representation of the program file and played back 2740; the jukebox 103 need not wait until the entirety of the first part 2716 is received 2734 before combining and playing back the program. In other words, playback 2740 may take place while the first part 2716 is being streamed. As described above, in one embodiment, the recombined file 2710 and the first part 2716 are discarded 2731 after or during playback.

[0034] As can be seen from the description above, it is not necessary to transmit the second part 2718 more than once. The second part 2718 can be stored locally after the first transmission, so that, in one embodiment, only the first part 2716 need be transmitted for subsequent streaming playback of the program file 2710. Since in some embodiments the stored second part 2718 is substantially larger than the first part 2716, the present invention greatly reduces bandwidth requirements for streaming playback of the program file 2710. After the second part has been stored locally 2732, even clients with slow connections may effectively stream the program file 2710. Thus, from the user's perspective, the program file 2710 appears to be streamed in a manner equivalent to conventional streaming.

[0035] In addition, the described file splitting may provide advantages in licensing distribution of programs for playback by users. The file splitting benefits the licensor because the user does not receive a persistent usable copy of the program file, and the server retains control of playback of the program file 2710. This is advan-

BNSDOCID: <CA 2466482A1 I_>

tageous because in contexts where blanket licensing or compulsory licensing is available for providing streamed data, in audio programming for example, a persistent copy would cannibalize revenues from future licensed streamed data downloads, and lack of control would make collecting licensing revenues difficult. The file splitting benefits users who do not want to wait while a large file is transmitted and received before accessing the program file. By splitting the file and storing part locally, transmission of large and unwieldy amounts of data in real-time is unnecessary. The user has quick access to the program file, even if he or she is using a slow communication channel.

[0036] The present invention thus improves efficiency in the transmission of program files such as audio or video files, and helps avoids bottlenecks associated with conventional streaming techniques, while still providing control and security.

[0037] Fig. 3D is a flow diagram illustrating a method of receiving and playing a previously split program file according to one embodiment of the present invention, so as to provide improved streaming performance from a user's perspective, even if adequate bandwidth to stream the entire file 2710 is not available, through advance downloading of the second section 2718. For example, the second section 2718 may be transmitted in advance of the actual playback of an audio or video program in a first session, thus avoiding potential bottlenecks that may result when the entire program file 2710 is transmitted upon user demand according to streamed techniques. Advance transmission can be performed at off-peak times, thus avoiding bandwidth bottlenecks. This advance transmission can occur when the user is not actually listening to music, so as to facilitate improved usage of an otherwise idle network connection.

[0038] The jukebox 103 determines 2742 which program files are likely to be requested by a user, so that at idle times it can transfer data that is likely to be useful in the future. Such determination may be made, for example, by using the learned artist relationships described in U.S. Patent Application Serial Number 09/846,823, filed on April 30, 2001, Patent No. X,XXX,XXX, in order to "guess" which tracks the user is most likely to request in the future. Alternative methods of determination may also be used. In some embodiments, the files the user will likely want may be

BNSDOCID: <CA_____2466482A1_I_>

known, such as when the user listens to a personalized radio station, or tends to listen to a particular online radio station. Such radio stations often have a list of the files that will be streamed later. Thus, this list can be used to determine 2742 what files will be required.

[0039] After the jukebox 103 determines 2742 what files will likely be required, the jukebox 103 requests the second parts 2718 of the files at a time when the connection is idle, or when sufficient bandwidth exists to transmit files. The authorization and content server 1614 then transmits 2744 the requested second parts 2718 to the jukebox 103. The jukebox 103 locally stores 2746 the second parts 2718 to the files. At a later time, when a user requests that a program file 2710 be streamed, the jukebox 103 requests the first part 2716 of the program file 2710, as described above. In response, authorization and content server 1614 transmits 2748 the first part 2716. The jukebox 103 combines 2750 the received first part 2716 with the stored second part 2718 to form a representation of the file 2710 and plays back 2752 the representation of the file 2710. Such a technique permits higher-quality playback than would otherwise be available, by pre-transmitting a significant portion of the data needed for playing back the media program. As described above, in one embodiment, the recombined file 2710 and the first part 2716 are discarded 2754 after or during playback.

[0040] In one alternative embodiment, instead of transmitting 2744 the second parts 2718 over a network, the second parts 2718 may be stored on physical media (such as a compact disc or floppy disk). That physical media is then physically sent to the location of the jukebox 103, where it can act as a local storage device, or the second parts 2718 may be transferred from the physical media to desired local storage. Thus, even for slow network connections, or for very large files, the user may enjoy a streaming experience, while the server retains control of the program file.

[0041] In another embodiment, the file is further split according to circumstances surrounding its transmission. The stream delivery system 150 begins to transmit all or part of the program file, such as an audio file, to the jukebox 103. When the program file, or part of the program file is partially transmitted, transmission is stopped. This stop can occur because of a disconnection between the stream

BNSDOCID: <CA_____2466482A1_I_>

delivery system 150 and the jukebox 103, because a user wishes to stop transmission and use the bandwidth for other purposes (such as when skipping to the next streamed song in a personalized radio station), or other reasons. The partially transmitted section of the program file or part of the program file is stored locally. Then, at a later time, the rest of the program file or part of the program file is transmitted and combined with the stored part. Such a process is more efficient than starting to transmit the entire program file or part of the program file from the beginning if it was not completely transmitted in a prior session.

[0042] In one embodiment, the size of the transmitted information is used to determine whether the entire file or section has been transmitted, and if not, what further information must be transmitted. At the beginning of transmission, the stream delivery system 150 sends information to the jukebox 103 indicating the size of the file and parts to be transmitted. This information may be sent to the jukebox 103 in other ways as well, as part of a sequence listing, for example. The jukebox 103 can then use the received size of the file in conjunction with the stored file size to determine what remaining information is needed from the stream delivery system 150. Other methods to identify what further information is needed to complete a partially received file may also be used.

[0043] For example, one embodiment splits a 1000 byte audio file into two parts, a first section 2716 of 50 bytes, and a second section 2718 of 950 bytes. The stream delivery system 150 transmits 800 bytes of the second section 2718 to the jukebox 103. At this point, a user chooses to start transmission of another audio file, and end transmission of the first audio file. The transmitted 800 bytes of the second section 2718 are stored locally. Thus, to allow the user to listen to the first audio file in its entirety, only the remaining 150 bytes of the second part 2718 need be transmitted to the jukebox. This subsequent transmission of the remaining 150 bytes may be done as needed, such as when a user begins to listen to the audio file again and the first 800 bytes are taken from local storage, then the last 150 bytes are received from the stream delivery system. This transmission can also be performed in advance of the time the user wishes to listen to the audio file, such as a scheme where the jukebox 103 periodically checks for partial file sections, such as partial second parts 2718, and auto-

BNSDOCID: <CA_____2466482A1_I_>

matically request transmission of the rest of the second part 2718. The jukebox 103 may identify partial file sections by comparing the total file size to the stored size, and then requesting only the missing information, or through other methods.

[0044] In another alternative embodiment, instead of discarding the first part 2716, the jukebox 103 encrypts the first part 2716 and stores the first part 2716 locally. This further reduces the amount of data that must be transmitted for program file 2710 playback. The server 1614 may still retain control over playback of the program file 2710 by controlling when the jukebox 103 may decrypt the first part 2716, for recombination with the second part 2718 and playback of the program file 2710. The second part 2718 could also be encrypted in place of, or in addition to, the first part 2716.

[0045] In yet another embodiment, transmitter 121 employs scalable coding to increase the quality of audio output despite limitations in channel capacity. Fig. 4A is a block diagram illustrating an example of a scalably-coded program file, including data stored in two sections 2804, 2806 that together form a program file 2802. Primary set of data 2804 is sufficient to play back the program file 2802 at a low quality level. Secondary set of data 2806 may be added to the primary set 2804 for higher-quality playback. The sets of data 2804, 2806 may be stored separately in the content database 102 or with the signal files 1615, or may be stored in a combined state, to be separated and transmitted as needed.

[0046] Fig. 4B is a block diagram illustrating the use and combination of primary data 2804 and secondary data 2806 to provide different playback quality levels. As mentioned above, the primary data 2804 is sufficient to play back the program file 2802 at a low quality level 2808. In one embodiment, the secondary data 2806 alone does not provide information sufficient for playback 2810 of the program file 2802. However, when the secondary data 2806 is combined 2812 with the primary data 2804, the combined data 2812 is sufficient to play back the program file 2802 at a high quality level 2814. For example, if the program file 2802 is an audio file, low-quality audio can be produced using the primary data 2804. Audio cannot be produced with secondary information 2806 alone. If secondary information 2806 is combined with the primary information 2804, output quality is enhanced. In one em-

BNSDOCID: <CA_____2466482A1_I_>

bodiment, additional levels of information may also be provided, each of which can be combined with the lower levels to further enhance output quality. This allows caching of lower-quality audio received in a first session and later receiving secondary information 2806 in a second session, which enables the jukebox 103 to combine the primary and secondary information to output audio of increased quality.

[0047] Since different sets of data are available that successively improve program file 2802 quality level, the quality of the output by the jukebox 103 depends on how much data can be transmitted in the time available. The first time an audio track is transmitted, transmitter 121 provides jukebox 103 with the primary information 2804 first, which allows the jukebox 103 to playback a representation of the program file 2802. Secondary 2806 (and additional) information is transmitted to allow improved quality levels as time permits. Jukebox 103 outputs the audio track with whatever level of information it has received at the time output is to commence. If only primary information 2804 has been received, jukebox 103 outputs lower-quality audio. If secondary information 2806 has also been received, it is combined with the primary information 2804 and jukebox 103 outputs higher-quality audio.

[0048] In addition, jukebox 103, in one embodiment, caches the received information. If the same audio track is requested at a later time, transmitter 121 provides jukebox 103 with the next level of information. Therefore, even if jukebox 103 was unable to provide higher-quality audio during the first listening, it may be able to provide higher-quality audio during subsequent listenings, by combining secondary (and/or additional) information with the previously cached primary information to generate the higher-quality audio output. Such a technique facilitates the output of high quality audio even when network transmission capacities are limited.

[0049] Referring now to Fig. 4C, there is shown an example of a transfer sequence for a channel with moderate bandwidth. Initially, tracks A and B are requested. Primary information for track A 2820 is downloaded. As primary information 2820 is downloaded, a low-quality representation of track A 2828 is played, according to conventional streaming audio techniques. Downloaded primary information 2820 is cached.

BNSDOCID: <CA_____2466482A1_I_>

[0050] Once the download of primary information for track A 2820 is complete, jukebox 103 begins to download primary information for track B 2822. This download may begin even though track A is still playing 2828. In the example shown in Fig. 4C, the download of primary information for track B 2822 is completed while track A is still playing 2828. Therefore, jukebox 103 begins to download secondary information for track B 2824. Then, when playback 2828 of track A is finished, jukebox 103 is able to output a high quality representation of track B 2830, by combining secondary information 2824 with previously downloaded primary information 2822. The output of the high quality version 2830 may take place while secondary information 2824 is still being downloaded, again using streaming techniques.

[0051] In the example of Fig. 4C, a request to play track A a second time is received. Therefore, once secondary information 2824 has been downloaded, jukebox 103 begins to download secondary information for track A 2826. Once the high quality version of track B 2830 is finished playing, jukebox 103 outputs a high quality representation of track A 2832, by combining secondary information 2826 with previously downloaded primary information 2820.

[0052] Referring now to Fig. 4D, there is shown another example of a transfer sequence for a channel with a lower bandwidth than that of Fig. 4C. Here, the secondary information for track B 2824 is not downloaded, because it would not arrive in time to improve the output of track B. Accordingly, a lower quality version of track B 2834 is output in lieu of the higher quality version 2830 of Fig. 4C. However, the higher quality version of track A 2832 can still be presented, since there is sufficient time to download secondary information for track A 2826 before the second playback of track A commences.

[0053] One skilled in the art will recognize that the tracks depicted in Figs. 4C and 4D may refer to individual songs, or song segments, or any other unit of information. One skilled in the art will further recognize that the scalable coding techniques described herein may be applied to video data, or to any other type of data, and are not limited to audio data.

[0054] The scalable coding techniques of the present invention thus facilitate the trading off of quality in bandwidth-limited situations, without requiring

BNSDOCID: <CA 2466482A1 I >

Ex. 1018-301

complex bandwidth estimation and determination. If insufficient bandwidth exists for the delivery of higher-quality versions, the system simply continues playing lower quality versions of tracks. No skipping, pausing, or other interruption of the audio stream is necessary. Jukebox 103 can determine whether to continue any particular transfer to improve the available quality or to download the next requested track, based on upcoming track selections. At any given moment, the next data segment to request can be determined by requesting the highest priority data segment from the next few audio segments. In one embodiment, priorities are defined to either play audio at a maximum short-term quality level or at a consistent quality level.

[0055] In one embodiment, jukebox 103 requests data for downloading according to the following order of priorities:

Priority	Type of value
1	Primary information, next track
2	Secondary information, next track
3	Primary information, track after next
4	Secondary information, track after next
5	Tertiary information, next track
6	Tertiary information, track after next
7	Data for subsequent tracks

[0056] One skilled in the art will recognize that any desired priority list may be provided. For example, if item 5 in this table is moved up to the third rank, the system will give more priority to high quality presentation at the possible expense of inconsistent quality on lower bandwidth connections.

[0057] In one embodiment, locally-cached downloaded data is stored in an encrypted or otherwise protected form, so as to prevent its abuse and to inhibit copyright infringement. In another embodiment, primary information is stored in an encrypted or otherwise protected form, but secondary and subsequent information is not, since the secondary and subsequent information is unusable without access to the primary information.

BNSDOCID: <CA 2466482A1_I_>

[0058] In one embodiment, jukebox 103 downloads data sets when the user is not actually listening to music, so as to facilitate improved usage of an otherwise idle network connection. Jukebox 103 determines which items are likely to be requested by a user, so that at idle times it can transfer data that is likely to be useful for rendering audio segments in the future. Such determination may be made in the same way as described above with respect to the split files of Fig. 3D. In one embodiment, secondary information 2806 for such "predicted" audio segments is downloaded first, so that encryption is not required unless and until the user actually requests the tracks and the primary information 2804 is to be downloaded. Scalable coding may also be used to process a signal of a conventional broadcast radio station that plays music. An audio recognition device, as is conventional, pre-processes the signal in order to identify individual songs. Those portions of audio information that are not music are compressed and stored, and a transfer sequence is sent to jukebox 103 that references these recently encoded non-music segments as well as previously known and cached musical segments. The recently encoded segments can be encoded at a lower quality level in order to allow a jukebox 103 connected by a low speed line to transfer the recently encoded segments in real-time while still playing the cached musical segments at a higher quality level.

[0059] As with file splitting, portions of scalably coded files may also be sent on physical media rather than transmitted over a network. Similarly, if sections of scalably coded information are only partially transmitted, they may be stored locally, and then only the information not previously received is transmitted in a subsequent session.

[0060] File splitting and scalable coding may also be used in combination. Fig. 5A is a block diagram illustrating how a program file 2902 may be divided into primary data 2904 and secondary data 2906, and the primary data 2904 file split into a first part 2908 and a second part 2910. The primary data 2904 is split as described above with respect to Figs. 3A and 3B. This allows almost all of the primary data 2904 and secondary data 2906 to be cached locally without requiring encryption. Since, in one embodiment, the second part 2910 of the primary data 2904 is not playable without the first part 2908, and the secondary data 2906 is not playable without a playable

BNSDOCID: <CA_____2466482A1_I_>

set of primary data 2904, all of the program file 2902 except the first part 2908 of the primary data 2904 may be cached locally without providing the user with a usable copy of the program file 2902 that may be played back. This allows the server 1614 to retain control of playback of the program file 2902, as long as the first part 2908 of the primary data 2904 is not stored locally in a usable form.

[0061] Fig. 5B is a block diagram that illustrates how such a scheme provides advantages for streaming program files 2902. First, the first and second parts 2908, 2910 of the primary data 2904 are transmitted 2912 to the client by the server 1614. The first and second parts 2908, 2910 may be transmitted at roughly the same time in response to the jukebox 103 requesting the program file 2902. Alternatively, the second part 2910 may be transmitted in advance, after determination of what files 2902 are likely to be desired, as described above with respect to Fig. 3D. In an embodiment wherein the second part 2910 is transmitted in advance, the first part 2908 is transmitted in response to the jukebox 103 requesting the program file 2902. The first and second parts 2908, 2910 are sufficient to provide low quality playback 2914. The second part 2910 is then stored 2916 locally, and the first part 2908 is discarded. At a later time, such as when the jukebox 103 requests the program file 2902 again, the server 1614 transmits 2918 the first part of the primary data 2908 and the secondary data 2906. This allows sufficient information for the program file 2902 to be played back at a high quality level 2920. The secondary data 2906 is stored locally 2922, and the first part 2908 of the primary data 2904 is discarded. For subsequent playback of the program file 2902, only the first part 2908 of the primary data 2904 need be transmitted 2924 in order to enable high quality level playback 2920. Again, after such playback, the first part 2908 of the primary data 2904 is discarded, so that the server may retain control of program file 2902 playback. As an alternative to discarding the first part 2908, the first part 2908 may be encrypted, with decryption only occurring under the server's 1614 control.

[0062] Thus, a significant portion of the program file 2902 may be stored locally, with only the relatively small first part 2908 being transmitted from the server 1614 to provide the user with streaming file performance. This allows high quality streaming performance even over a relatively narrow bandwidth connection.

BNSDOCID: <CA_____2466482A1_I_>

[0063] As described above, when scalable coding or file splitting is used, the jukebox 103 plays back a representation of the program file. This representation may be an exact digital replica of the program file, such as when the program file is split, all parts are transmitted to the jukebox 103, and the entire file is combined for playback. The representation may also be a less than exact replica, such as when only some portions of a scalably coded program file are combined and played back.

Recommendation Engine

[0064] One method of determining in advance what program files a user may want streamed is through use of a recommendation engine, described more fully in U.S. Patent Application Serial Number 09/846,823, filed on April 30, 2001, Patent No. X,XXX,XXX. Note that there are also many other ways to determine what program files a user may want streamed in advance. Referring now to Fig. 2, the jukebox 103 or personalized radio station accepts a user's selections of music tracks and makes additional recommendations as to music tracks the user is likely to enjoy. The user is able to search for particular tracks and/or artists, and to control the playback of selected tracks. The system monitors the user's behavior with regard to searching, listening, and playback control, and generates and analyzes logs of such behavior in order to refine recommendations. Advertising, offers, and other information may be selected and presented to the user based on observations of user behavior and analysis as to which material may be of interest to the user.

[0065] Compressed signal files 1615 contain descriptions of music tracks, and in one embodiment contains digitized representations of the music tracks themselves. Compressed signal files 1615 are stored, for example, using conventional database storage means or in a conventional file system, and in one embodiment include several fields providing descriptive information regarding music tracks, such as title, album, artist, type of music, track length, year, record label, and the like.

[0066] Selected tracks are played via jukebox 103, which is implemented in one embodiment as a standalone application, or as a plug-in or bundled feature in browser 105. Jukebox 103 receives digitized representations of music tracks and plays the tracks over a speaker or headphones at the user's computer. In one embodiment,

BNSDOCID: <CA_____2466482A1_I_>

jukebox 103 can download and save music tracks in local storage (such as a hard disk drive, memory, or other local storage) in a compressed format, such as MP3, for playback on the user's computer or on a portable digital music listening device.

[0067] Recommendation engine 107 generates track preferences based on user information. Radio sequence generator 1613 uses track preferences, along with general constraints 1616 and format definitions 1611, to generate a sequence of tracks to be played. General constraints 1616 include particular rules and restrictions on the sequence of tracks, as may be required by law or as may be determined to be desirable for marketing or aesthetic purposes or for other reasons. Examples of constraints 1616 include: "no more than one song per hour from a particular album," or "do not play a fast song immediately after a slow song." Radio sequence generator 1613 may also incorporate a randomization element, if desired, and may be configurable by a website operator.

[0068] The track list is sent to jukebox 103 to be played to the user. A user activates jukebox 103 and selects music tracks for playback and/or purchase, via a user interface including controls and selectors. Authorization and content server 1614 checks that the appropriate security measures are in place (in order to prevent the user from "hacking" jukebox 103 to request unauthorized tracks from content server 1614), obtains the actual music tracks from files 1615, and provides them to jukebox 103 for output.

[0069] In one embodiment, the connections among the various elements of Figs. 1A and 2 are implemented over the Internet, using known protocols such as HTTP and TCP/IP. Secure sockets layer (SSL) or other encryption techniques may be employed for added security.

[0070] The recommendation engine may be employed in connection with conventional radio station programming techniques, to implement an improved personalized radio station. As is known in the art, conventional radio stations typically divide a programming block into a number of segments. Each segment is assigned a programming category, such as "power hit," "new release," "recurrent hit," and the like. For a particular programming block, music tracks are assigned to each of the segments based on the particular programming format of the radio station. Music

BNSDOCID: <CA_____2466482A1_I_>

scheduling software, such as Selector® by RCS Sound Software, applies heuristic rules for repetition limits and classes of songs, to automatically generate track lists for use by radio stations. The recommendation engine may be combined with such existing radio station programming techniques, to populate the defined segments with music tracks that are likely to appeal to a particular listener. Additional rules may be applied in generating track lists, so as to limit undesired repetition and to comply with limiting legislation (such as the Digital Millennium Copyright Act) and other restrictions.

[0071] From the above description, it will be apparent that the invention disclosed herein provides a novel and advantageous system and method for providing a user with a streaming program file experience. The foregoing discussion discloses and describes merely exemplary methods and embodiments of the present invention. As will be understood by those familiar with the art, the invention may be embodied in other specific forms without departing from the spirit or essential characteristics thereof. For example, the invention may be applied to other domains and environments, and may be employed in connection with additional applications where personalized recommendations are desirable. Accordingly, the disclosure of the present invention is intended to be illustrative, but not limiting, of the scope of the invention, which is set forth in the following claims.

BNSDOCID: <CA_____2466482A1_I_>