a decoder for decompressing the encoded database on the client computer into a sequence of frames for real time display.

There is also provided in accordance with another preferred embodiment of the present invention a system for transmitting model based data representations of three dimensional images over plural transmission links having limited bandwidth, said system including:

a digital data source storing model based data representations of three dimensional images;

an image processor for rendering views of said model based data representations into raster bitmap format;

a digital data receiver receiving said digital data in said raster bitmap format over a one of the plural transmission links having limited bandwidth; and

a digital data transmitter operative to transmit the digital data in said raster bitmap format to said receiver over a transmission link having a limited bandwidth in plural blocks which are sequentially transmitted at a rate determined by the limited bandwidth, each block being an incomplete collection of data which includes parts of multiple frames, each frame being viewable in a selectable order by said receiver even when less than all of the plural blocks have been received, receipt of subsequent blocks by the receiver being used to cumulatively improve the quality of the digital data viewed by the receiver.

Further in accordance with a preferred embodiment of the present invention the model based data representations comprise VRML representations.

Still further in accordance with a preferred embodiment of the present invention the model based data representations comprise CAD-CAM representations.

Additionally in accordance with a preferred embodiment of the present invention the image processor is operative to render only views which are selected by a user.

There is also provided in accordance with another preferred embodiment of the present invention a method for transmitting model based data representations of three dimensional images over plural transmission links having limited bandwidth, said system including:

storing model based data representations of three dimensional images;

rendering views of the model based data representations into raster bitmap format;

receiving the digital data in said raster bitmap format over a one of said plural transmission links having limited bandwidth; and

transmit the digital data in said raster bitmap format to the receiver over a transmission link having a limited bandwidth in plural blocks which are sequentially transmitted at a rate determined by the limited bandwidth, each block being an incomplete collection of data which includes parts of multiple frames, each frame being viewable in a selectable order by the receiver even when less than all of the plural blocks have been received, receipt of subsequent blocks by the receiver being used to cumulatively improve the quality of the digital data viewed by the receiver.

Further in accordance with a preferred embodiment of the present invention the model based data representations comprise VRML representations, and CAD-CAM representations.

Still further in accordance with a preferred embodiment of the present invention the image processor is operative to render only views which are selected by a user.

The following definitions are employed throughout the specification and claims:

RESOLUTION--The relationship between the number of digital samples per unit of an original and the number of digital samples per unit in a rendered version thereof. Specifically, when dealing with images, resolution refers to the relationship between the number of pixels per unit area of an original image or scene and the number of pixels per unit area in a displayed image. Specifically, when dealing with audio, resolution refers to the relationship between the number of samples per unit time of an original sound and the number of samples per unit time in a played sound.

QUALITY--The degree to which a rendered version of an original is faithful to the original. Specifically, when dealing with images, quality refers to the degree to which the displayed image is faithful to the original image or scene. Normally this is expressed as the degree to which the approximation of pixel values in the displayed image approaches the correct pixel values in the original image or scene. Specifically, when dealing with audio, quality refers to the degree to which a played sound is faithful to the original sound.

FRAME--A portion of an original which can be independently and interactively manipulated. Specifically, when dealing with images, frame refers to a portion of an image or of a collection of images which can be independently and interactively manipulated. Specifically, when dealing with audio, frame refers to a portion of a sound which is delimited in time and can be independently and interactively manipulated.

BLOCK--A sequentially transmitted collection of partial data which is used to build multiple frames. The frames are built up of one or more sequentially transmitted blocks, whose contents are accumulated. Specifically, when dealing with images, the block contains image data. Specifically when dealing with audio, the block contains audio data.

PARTIAL FRAME--The part of a frame which is contained in a given block.

TILE--A window sized pixel array of a predetermined given size forming part of an image. For example, tiles partition an image into a plurality of arrays, each of which contains an identical number of pixels.

BRIEF DESCRIPTION OF THE DRAWINGS

The present invention will be understood and appreciated more fully from the following detailed description, taken in conjunction with the drawings in which:

Fig. 1 and Fig. 2 are simplified block diagrams illustrating a system for scalable representation of multimedia data for progressive asynchronous transmission, constructed and operative in accordance with a preferred embodiment of the present invention;

Fig. 3A is a simplified schematic diagram of the database structure of the present invention which includes three databases, embodied within the client-server system of the present invention;

Fig. 3B is a simplified diagram of a database structure particularly useful in the client database of Fig. 3A, illustrating its two-dimensional nature;

Fig. 4 is an illustration of the operation of a preferred embodiment of the present invention;

Fig. 5 is a simplified schematic diagram of a production tool for converting a digital multimedia file into a progressive scalable database representation for storage on a server computer in accordance with a preferred embodiment of the present invention;

Fig. 6 is a simplified schematic diagram of the structure of a block within the server database, partitioned into frames which can be accessed randomly in accordance with a preferred embodiment of the present invention;

Fig. 7 is a simplified schematic diagram of a decoder for receiving and integrating data blocks from a scalable database, to form a version of a digital multimedia object for playback in accordance with a preferred embodiment of the present invention;

Fig. 8 is a simplified schematic diagram of a scalable progressive database for a video clip in which the first data blocks are used for previewing the video in accordance with a preferred embodiment of the present invention;

Fig. 9 and Fig. 10 are simplified schematic diagrams of a system for incorporating a scalable progressive database into a time-based video sequence of frames indexed by two time scales: a macro and micro scale, in accordance with a preferred embodiment of the present invention;

Fig. 11 is a simplified block diagram of a proxy system used to cache in a central hub multimedia data which is transmitted from servers to clients in accordance with a preferred embodiment of the present invention;

Fig. 12 is a simplified block diagram of a system for generating a scalable database from digital media data, by running a compressor in a feedback loop in accordance with a preferred embodiment of the present invention;

Fig. 13 is a simplified block diagram of a decoder for the database generated by the system of Fig. 12 in accordance with a preferred embodiment of the present invention;

Fig. 14 is a simplified diagram illustrating a scalable progressive database useful for a large still image in accordance with a preferred embodiment of the present invention;

Fig. 15 is a simplified diagram illustrating a virtual reality system constructed and operative in accordance with a preferred embodiment of the present invention;

Fig. 16 is a simplified flowchart illustrating operation of the system of Fig. 15; and

Fig. 17 is an illustration of the operation of a preferred embodiment of the present invention illustrated in Figs. 15 and 16, permission to reproduce Fig. 17 was granted by Tecnomatix, Ltd.

DETAILED DESCRIPTION OF PREFERRED EMBODIMENTS

The preferred embodiments of the present invention may be better understood given the following technical and theoretical explanation.

The present invention provides a novel method for representing multimedia data. The invention provides a scalable representation, so that the data can be asynchronously transmitted to clients having different bandwidth connections, played on-line almost immediately after the transmission begins, interactively controlled, and also progressively upgraded as it is replayed.

Although the present invention is described hereinbelow with particular reference to image data, it is to be appreciated that it is applicable also to non-image data, such as audio data.

When addressing bandwidth limitations, it is natural to think in terms of data rates, or velocities. Video players, for example, play at standard rates such as thirty frames per second (fps), and require the images for display to be available at this rate. If the images are already stored on a local hard disk, then all that is necessary is disk access, which is very fast. On the other hand, if the images are streamed in from a server, then in order for on-line playback to be possible before a full download is finished, the rate of transmission must be great enough to supply the frames at thirty fps. This does not mean that the network link has to transmit the data equivalent to thirty full frames every second. Due to compression, it suffices if the network transmits thirty *compressed* frames every second.

For example, if the compression achieved is 10:1, then it suffices to transmit at a rate of three fps, provided that the client CPU can decompress thirty compressed frames into full frames every second. In fact, compression is the mediator between the video player and the bandwidth. The player does not slow down when bandwidth is low; rather, the compression ratio has to be greater. Should a bottleneck arise, and a frame is not available when the player needs it, then the player simply skips that frame, but continues to expect frames at the thirty fps rate. The video can be preset at the outset for lower rates than thirty fps, but not much lower, since slow video playback breaks the continuity between frames, and thus loses the effect of motion.

Thus, realizing that higher compression means lower quality, it can be appreciated that users having high bandwidth connections can receive high quality video for on-line playback, and users having low bandwidth connections receive low quality video. In a non-scalable setting, it is thus necessary to prepare different compressed media files for each playback rate and bandwidth combination. For example, four different versions of the media could exist for (a) 24 fps playback, 14.4 Kbs bandwidth, (b) 24 fps playback, 28.8 Kbs bandwidth, (c) 30 fps playback, 14.4 Kbs bandwidth, (d) 30 fps, 28.8 Kbs bandwidth. As described below, using the present invention a single media file can be used to accommodate all four of these combinations.

In the present invention, the media data is comprised of m frames $F_1, F_2, \ldots, F_m$. A frame can be, for example, an individual frame of a movie sequence, a piece of a panoramic view, an individual segment of an audio signal, or even a sub-sampled version of a large still image. It can also be a group of such frames, such as for example, a group of inter-frames between key frames in a video segment, in a case where an H.263 codec is being used. In broad terms, frames are units of interactivity. For example, in object movies where interactivity means frame advance, a frame unit is an individual still image, whereas in gazing applications where interactivity means zooming in and out, the frame units are multi-resolution tiles.

The representation encodes the media data into n data blocks $B_1, B_2, \ldots, B_n$ preferably of roughly equal size. Each encoded data block B_j contains m compressed frame units $F_1^j, F_2^j, \ldots, F_m^j$. Thus it is appreciated that the database is arranged in two dimensions, corresponding to blocks and frames. The dimension used for blocks is for achieving progressiveness, and the dimension used for frames is for achieving interactivity. The frame data can be transmitted in a selective order, but the blocks must be transmitted in sequence, since they build cumulatively. This is an essential feature of the subject invention.

Data block B_1 is used to deliver the media at the lowest bandwidth, say f_1 Kbs; data blocks B_1 and B_2 when integrated together, are used to deliver the media at bandwidth $f_2 = 2f_1$ Kbs; and in general, for $1 \leq k \leq n$, data blocks $B_1, B_2, \ldots, B_m$, when integrated together, are used to deliver the media at bandwidth $f_k = kf_1$ Kbs. Each higher bandwidth version delivers a higher quality rendition of the media. In this way, the representation can accommodate multiple clients connecting to the network with different bandwidths. Moreover, a client connecting with bandwidth f_1 who has downloaded data block B_1 and played the media, can continue downloading block B_2 in background, since the bandwidth has been freed, and thereby achieve the same quality as bandwidth f_2 the next time the media is replayed.

Within each block the frames can be accessed randomly and delivered selectively, so that the user can vary the quality level among the frames. For example, a viewer who wants to gaze at frame #3 may instruct the database to send frame #3 data from the first ten blocks, but only one block of data for all of the other frames. The viewer selection is carried out interactively, through the use of keyboard presses and mouse clicks, as the media is being played. Whereas for some applications it may be most natural to transmit the entire blocks in sequence, for other applications it may be more effective to first deliver as much data as possible for specific frames at the expense of lowering the quality of other frames. The two dimensionality of the database, together with its interactive accessibility, gives the user complete control over the transmission sequence.

The scalable representation that is the subject of the present invention is embodied in a production tool which enables the producer to control the bandwidth parameters f_k, or equivalently, the qualities of the media versions obtained by integrating blocks B_1, B_2, ..., B_k. In general, it is not necessary that the blocks be of equal size, nor that the frequencies f_k be given by kf_1, although this is the preferred embodiment. The production tool also enables the producer to control the final quality of the highest bandwidth version, or equivalently, the total number, n, of data blocks in the representation.

In contrast, if the media data representation is not scalable, but is encoded instead for a specific bandwidth f, then only clients with bandwidth f or greater can play the media on-line as it is being downloaded. A client with a lower bandwidth than f would have to download the entire data stream to memory in order to begin playback, which can take a great deal of time on account of the large file sizes typically used in multimedia production. A client with a higher bandwidth connection than f would not be able to take advantage of it to receive higher quality media. Moreover, there would be no means of upgrading media quality, even for clients with high bandwidth connections, other than to transmit an entirely new data stream from the server side, and discard the previously downloaded data.

Applications of the invention include, inter alia, scalable audio and video transmission, video previewing, progressively rendered object movies and panoramas, large still images, efficient proxy or multi-casting unit (MCU) management for web and other hubs, and VRML transmission, as described hereinafter in greater detail.

Scaleable audio transmission: Digital audio data can be progressively encoded into a scalable database for asynchronous transmission at different bandwidths. A client connected with a low bandwidth line can receive a low quality version of the audio, which can be played back on-line at the low bandwidth as the data streams in. After the audio is played, additional data blocks can continue to be received and integrated with the previous blocks, so that the audio is upgraded to higher quality for replay.

Scaleable video transmission: Similar to the description above for the audio transmission, digital video data can be encoded into a scalable database for asynchronous delivery and progressive quality upgrade.

For certain time-based video systems, there is disclosed a novel way to incorporate the scalable database so that progressiveness and immediate playback can be achieved, even in a single-play mode. Specifically, this applies to video systems with two time scales, such as is present in the Apple QUICKTIME® movie player. The first time scale (hereinafter referred to as the "major scale") is used to advance from one frame to the next, based on major units of time. The second time scale (hereinafter referred to as the "minor scale") is a sub-division of the major scale into smaller time units, and is used to incorporate small changes or fluctuations into the frame being displayed. For example, the major scale can be advancing through a movie of a bird flying and the minor scale can be adding fluttering to the bird's wings. The advantage of such a two-scale player is that the decoder, which does the intensive processing to supply the frames, need only run at the slower rate, e.g. 3 frames per second (fps), governed by the major scale, whereas the viewer, doing the less intensive processing, is playing at the faster rate, e.g. 30 fps, governed by the minor scale.

The subject invention can be incorporated into a system having two time scales as described hereinabove by using the minor scale in a way different from the way that was originally intended. Instead of being used to introduce fluctuations, it is used to display progressively rendered versions of a frame. At each minor time unit, the player displays the latest version available of the frame indexed by the major scale. For example, suppose there are ten minor time units within the major time unit during which frame #4 is to be displayed. At the first minor time unit, the player initially displays the version of frame #4 which it has available from the first data blocks already processed. As additional blocks of data are accumulated and higher quality versions of frame #4 become available, the player displays those frames at successive minor time units. This continues for ten minor time units, until the next major time unit, at which time frame #5 is to be displayed. The cycle then repeats, and the version of frame #5 which is already available is initially displayed at the first minor time unit. Thus it can be seen that progressiveness can be achieved by interjecting into the minor scale, the versions of the frames obtained by accumulating successive blocks.

For the user to be able to view the video immediately, without waiting for the entire file to download, the production tool must store the encoded video in the order of successive blocks. Each partial frame must be handled as if it were an entire frame. That is, the production tool must treat the movie as if there were a total of $m \cdot n$ distinct frames being encoded. On the other hand, each frame is sent only once to the codec for encoding, and is returned as a series of encoded partial frames. Thus it is necessary to post-process the encoded data file, to rearrange the data items from a frame dominated order to a block dominated order. This rearrangement process is referred to as "flattening" in the art.

The player in turn, however, must know that although it is receiving what appears to be $m \cdot n$ data items, there are really only a total of m frames. It must decode and accumulate every successive sequence of m data items with the previous ones, to update the frames. The combined effect of the flattening on the production side and the player's interpretation on the client side enables seamless integration of the scalable progressive database within a non-progressive video interface. That is, the incorporation of progressive blocks does not require any modifications to the existing interface.

<u>Video previewing</u>: When encoding digital video data into a scalable database, the first data blocks can be used to generate a preview of the video, restricted to selected

frames. The preview can be played back by the client almost immediately after the streaming begins. Moreover, additional data blocks received are integrated with the data blocks from the preview, to form full view versions of the video.

Object movies: Advertising agencies are using object movies to produce interactive 3-D virtual reality presentations of merchandise on the Internet. The user can rotate and zoom the 3-D object, and examine it from different viewing angles. Using the methodology of the current invention, object movies can be progressively encoded so that the viewer can download and begin playing them almost immediately after the streaming begins. Initially the movie will scale to a quality commensurate with the bandwidth of the user's network connection, but as the data blocks are received and the user interacts with the movie, additional data blocks are delivered and integrated with the previous blocks, resulting in a higher and higher quality movie. An important feature of the invention is that, regardless of bandwidth, the user can begin playback and interaction almost immediately, and does not need to wait for the complete download, as the first version of the movie delivered scales itself to the native bandwidth. As playback continues additional data streams in the background and the movie version is upgraded to higher and higher quality.

Panoramas: Panoramas are very large images which the user cannot view in their entirety, but rather sees within a restricted viewing window. By panning in various directions, and zooming in and out, the user navigates through the panorama. The continuous change in viewing window gives the effect of movement within a scene. Similar to the description above for object movies, panoramas can be progressively encoded so that the viewer can download and begin navigating through them almost immediately after the streaming begins. Initially the panorama scales to the client bandwidth, and after the first data blocks are received, additional data blocks are streamed in background while the panorama is playing, to provide higher and higher image quality.

Large still images: Although large high quality still images are not composed of frames in the conventional sense, the large sizes of the image data files makes the present invention an effective means for interactive on-line viewing. The frames can be small image tiles within the full image at different resolutions, the smaller tiles having higher resolution than the larger ones. The locations of the tiles can be marked as hot spots. When the viewer clicks on a hot spot within a specific tile, the database delivers that tile at a higher resolution, giving the effect of a zoom in. Within the higher resolution tile there can be more hot spots, and the zooming can continue through the database.

As a simple example, the first frame within a data block may contain the full image sub-sampled by 4:1, for example, in each dimension. The next set of frames within the data block may contain (some subset of) the four quadrants of the full image sub-sampled by 2:1 in each dimension. The next set of frames within the data block may contain (some subset of) the sixteen quadrants within the above four quadrants at the original resolution. A viewer could see the 4:1 reduction of the original image (the first block), click on one of the quadrants and then see that quadrant at a 2:1 reduction (a frame from the second set), and click further on one of its quadrants and then see it at full scale (a frame from the third set).

Efficient proxy/MCU management: Proxies are large storage devices, located as hubs within networks, used as large caches for data being delivered from servers to clients. Similarly MCUs are large storage devices used as caches for data being delivered from broadcasting stations to viewers, such as cable TV. As data is streamed from servers to clients or from broadcasting stations to viewers upon request, the proxy or MCU stores the data in a central hub so that it is available for delivery at a high bandwidth if requested again by any of the clients connected to the hub. It plays a similar role to paging files on a local computer disk, but on a much larger scale and for a much larger clientele.

The scalable representation of the subject invention is particularly well suited for proxies and MCUs which operate in asynchronous environments. Server/client connections and broadcast transmissions can be of many different bandwidths, and so the proxy or MCU can be accumulating versions of the same multimedia data corresponding to different qualities. Without scalability these versions are all independent of one another, and cannot be combined to achieve quality levels other than those originally preset or combined to save space. Using the scalable representation of the subject invention, the proxy or MCU can be optimized to cache the various progressive building blocks. This affords great flexibility in being able to create versions of different quality levels, and reduces the space requirements.

As a simple example, a first user with a low bandwidth connection f_1 to the server, who demands a multimedia file, downloads data block B_1, which is then also saved to cache on the proxy, and the user receives a low quality version (quality level 1) of the media. A second user with a higher bandwidth connection f_2 to the server, who demands the same multimedia file, can download data blocks B_2 and B_3 from the server, and can access block B_1 directly from the proxy. The three data blocks are integrated and the second user receives a very high quality version (quality level 3) of the media. Data blocks B_2 and B_3 would then also be stored on the proxy. A third user with a direct connection to the proxy of bandwidth f_2 who now demands the same multimedia file, can receive the high quality version (quality level 2) comprising of blocks B_1 and B_2 directly from the proxy. Without a scalable representation it would not be possible for the second user, with a bandwidth connection of f_2, to receive a quality level 3 version corresponding to the higher bandwidth f_3, nor for the proxy to deliver to the third user a version at a quality level different from those directly available in its cache. Moreover, without a scalable representation the proxy memory required to cache the f_1, f_2 and f_3 versions would equal the size of six data blocks, rather than three. It can thus be seen that the proxy or MCU inherits the scalability from the servers, giving it a great deal more flexibility in its media delivery to the clients than would be possible in a non-scalable environment.

<u>VRML transmission</u>: Virtual reality modeling language (VRML) is a descriptive language for representing and rendering three-dimensional objects. The objects are modeled as collections of polygonal elements, the description of which forms a VRML database. Depending on the viewing parameters, the individual elements are processed and the desired view of the object is rendered into a raster bitmap for display. The VRML representation is rich enough to encapsulate all possible views of the object. In fact, there is an infinity of possible variations in viewing parameters. A user interacts with the VRML object by adjusting viewing parameters, through mouse clicks and keyboard presses. VRML was first popularized by Silicon Graphics. Their top Iris workstations, for example, can render on the order of a million polygonal elements per second. VRML images are characterized by their sharp photo-realistic attributes.

The present invention can be applied to efficiently deliver VRML imagery over a server/client network, for on-line interaction. Within the VRML environment itself, two problems arise when transmitting VRML databases over a network for on-line interaction. First, there is the bandwidth limitation, which inhibits the rate of transmission. Second, there is the intense processing on the client side, necessary to render the many polygonal elements into a bitmap for viewing. It would be preferable to have the server (typically a more powerful computer) do the rendering, but then it would be necessary to store rendered bitmaps of every possible set of viewing conditions -- a feat that would require on the order of terabytes of disk space.

The present invention can be used to mitigate the problem by allowing the rendering to be done on the server computer without requiring enormous memory, and yet enable the client to freely interact with the VRML object in an on-line interactive setting. This is one of many examples involving real-time encoding. The invention operates by receiving the viewing parameters from the user, rendering the corresponding image on the server into a raster bitmap image, encoding the bitmap into progressive partial frames and inserting them into a two-dimensional server database. The encoded data within the server database is continually streamed from server to client, enabling the client to begin viewing a low quality image as soon as the first partial frame data arrives. As the user navigates through the VRML, additional bitmaps are rendered, encoded and inserted into the server database. Whenever the user re-traces steps, so that the viewing parameters are the same as those selected at some previous stage, the server does not need to render the same bitmaps again. Rather, the streaming simply continues in background, and the quality of the image on the client side is enhanced as additional partial frames are integrated. Similarly, if the user stays focused on a single view, then the bitmap being displayed is enhanced as additional partial frames stream in. Once all of the partial frames are integrated, the image has the same sharp photo-realistic quality as is characteristic of VRML images. On the other hand, the user does not have to wait for all of the data to arrive in order to interact with the object, nor does the client computer have to do the intensive processing to render the VRML database into bitmaps.

Reference is now made to Fig. 1 which shows a block diagram of a system for providing on-line virtual reality (VR) movies. The system includes a production workstation 3 for receiving input images and processing same, as will be described hereinafter. Such input images may be constituted by photographs which are scanned into the workstation. The output from production workstation 3, being a raw VR movie, is fed into an encoder 5 for preparing the movie for transmission and in turn, applied to a server 7 essentially used for storage and transmission of the movie to clients, namely, subscribers or user units 9.

Reference is now made to Fig. 2, which shows user unit 9 of Fig. 1 in greater detail. Seen is a transceiver 34, an asynchronous memory/database 35, a decoder 36 and a user's workstation 40.

Typical operation of a preferred embodiment of the present invention is now described with reference to Figs. 1 and 2. Selected images are introduced in production workstation 3 in which the VR movie is produced in accordance with a certain script. The producer at the workstation determines the number and size (for example, in bytes) of the partial frames and also defines the various available interactions between the frames by defining hot spots and objects at 21, using auxiliary standard devices for producing movies, such as a keyboard, a mouse, speakers and a CPU all designated by 23. The product obtained is a raw VR movie, which is a complete VR movie that has not been reformatted for transmission. The preparation of the movie for transmission is effected in encoder 5 where partial frames are generated through an iterative process.

The partial frames are generated by encoder 5 as controlled by the controller 25, as follows A partial resolution frame or a partial resolution slice of each frame of the VR movie sequence is generated. One example of such a partial resolution frame is sub-sampled scan lines, e.g., the removal of every 10th line of a 150 line frame or a compression encoded frame, which partial resolution results in a blurry display. The partial resolution frame is then subtracted from the original frame by a partial frame subtractor 27, yielding a residual frame or a remainder frame. This process can be repeated on the residual frame, generating a second partial resolution frame. The procedure is also repeated time and time again, until the number (which is determined by the producer) of partial resolution frames is generated. The net result is a set of partial resolution frames that can be recombined into the original full resolution frame.

When this process is completed on each frame of the sequence, the partial resolution frames are transmitted. The order of transmission follows the script given by the producer, and commonly, the first partial resolution frame of each frame is transmitted, followed by the second partial resolution frame, and so on. This sequence of transmission allows for the whole sequence to be viewed in a partial resolution format that progressively comes into focus.

Partial resolution frames may be optionally compression encoded, possibly taking into account similarities between various frames. This is effected in the compressor 29.

Encoded and compressed VR movie parts are passed to a server 7 where the movie parts are stored in a database 31 and transmitted to a user's unit 9 part by part, by means of a transceiver 33.

As seen in Fig. 2, a user's transceiver 34 receives movie parts and transmits requests for additional information. A user's database 35 is progressively updated with requested images or, alternatively, may be progressively updated by the server 7. User's database 35 functions asynchronously, supplying the frames to the user via a decoder 36 by request independent of data transmission. Upon receiving the frames, the decoder initially decompresses the frames as indicated at 37 (if compression took place) and then decodes and recombines them by means of a partial frame integrator 38. Following this, the partial frames are stored in the user's database where the frames may be stored in a compressed format, effected by a compressor 39. A user's workstation 40 enables the user to view and interact with the VR movie. The user utilizes the workstation for sending requests for images to the decoder which retrieves (and decompresses, if necessary), the images from the user's database and sends requests for particular images which may not yet have been transmitted to the server database. Furthermore, the user's workstation actuates any script produced in the production workstation 3. Hence the user's workstation also includes the standard devices included in the workstation 3 and designated by the number 23 (Fig. 1).

As a preface to the explanation of the remaining figures, it is important to understand that the progressive scalable database which is a subject of the current invention preferably is stored in three databases within the server/client system.

Reference is now made to Fig. 3A which shows a preferred database structure in accordance with a preferred embodiment of the present invention. It is a particular feature of the present invention that three databases are employed, a server database 41, which is arranged in a serial form, containing multiple data blocks, each including multiple partial frame data, a client database 42 which is arranged in a two dimensional structure, conceptually illustrated in Fig. 3B and an interactive database 43 which contains a single data block including multiple frames, which is dynamically updated from the client database 42. In an alternative embodiment of the invention, wherein extremely high processing speeds are available at the client, the client database could be eliminated.

It is a particular feature of the present invention that interactive data streaming is provided. The use of three databases as described above enables interactive data streaming to be achieved in an efficient and cost effective manner. The use of databases having a two-dimensional structure greatly simplifies the data processing.

Server database 41, which is archived on a server constitutes a first database of the progressive scalable database. The server database 41 includes a plurality of data blocks in encoded form. As seen particularly in Fig. 3B, the progressive scalable database is two-dimensional in nature. It has a progressive dimension indexed by block number, and an interactive dimension indexed by frame number but it is serialized for streaming and can only be accessed sequentially. Server database 41 is streamed from server to client via the transmission and buffering protocol of the Internet browser.

Client database 42, the second database, is built up on the client side as the information streams in, to mirror the server database 41. Client database 42 is truly two-dimensional, with random access capability within the data blocks. The data blocks within it are also in encoded form.

Interactive database 43, the third database, is created by decoding the data from the client database 42. This interactive database 43 is one-dimensional, and contains only one sequence of frames, but it is dynamically updated. As additional block data is integrated, these frames are updated, with the previous versions over-written.

When a frame has been updated, the encoded frame used to update it is deleted from client database 42. Thus while interactive database 43 is being created and updated, client database 42 is being deleted. Since the updating of the frames is asynchronous, however, client database 42 typically contains frames from many different blocks simultaneously at any point in time. In essence, then, the progressive dimension of the database is being realized through a time dimension in interactive database 43.

Interactive database 43 is controlled by the user interface through keyboard presses and mouse clicks. The creation and update of interactive database 43 from client database 42 is done in background time slices, while the client CPU is idle. Interactive database 43 may store the frames in either raw bitmap form or in an intermediate compressed form, as long as the intermediate compression is such that the frames can be decompressed in real time for display. An advantage of using an intermediate compression is to confine interactive database 43 to internal RAM, which has fast access time, rather than swap to hard disk memory, which has slow access time. The swapping in itself is a drain on processing speed.

When the user requests a frame to be displayed, interactive database 43 displays that frame immediately, if it is available. In case the frame is not available, interactive database 43 passes a message back to client database 42 requesting that frame. Client database 42 accesses the specific frame requested from its first encoded data block, if it is available, and sends it to the decoder for decompression and integration, and subsequent incorporation into interactive database 43. Once a frame is incorporated, interactive database 43 displays the frame at once. If client database 42 has not yet received the requested frame from the server stream, then it must wait until the encoded frame arrives, since the streaming is sequential. If the streaming were instead random access, client database 42 would be able to directly request the specific frame it needs from server database 41.

In summary, the server database is two-dimensional but serialized for sequential streaming; the client database is two-dimensional with random access within blocks; and the interactive database is one-dimensional but dynamically updated. In the interactive database, the progressive database dimension is actually being represented as time rather than space. This "three database strategy," using three different databases: (*i*) two-dimensional serialized, (*ii*) two-dimensional, (*iii*) one space and one time dimension, is a key to the present invention, and to the discussion of the figures in detail which follow.

For each of these three databases, the progressive dimension manifests itself in a different way. Within the server database, *progressiveness means quality*. The encoder builds the blocks of the database based on achieving the best quality at given bit rates.

Within the client database, *progressiveness means cumulative integration*. Progressiveness within the client database is a computational property. The accumulator on the client computer integrates frames from successive blocks with those from previous blocks. Finally, within the interactive database, *progressiveness means time*. As time progresses, the frames are dynamically updated as more blocks have been accumulated. The transmission from the server database to the client database is streamed serially, and this is where the progressive dimension is effectively converted from "space" to "time."

The progressiveness manifests itself in bandwidth during the streaming. The transmission from the client database to the interactive database is asynchronous. The client database is being created in the background while the interactive database is being played, and the former acts as a buffer for the latter. Moreover, the client can interact with the media almost immediately after the streaming begins, and does not have to wait for the client database to be constructed entirely.

The interactive dimension of the database corresponds to whatever functionality the user interface allows. For example, it can manifest itself as frame advance for videos and object movies, navigating for panoramas, and gazing for large still images.

Reference is now made to Fig. 4 which illustrates one application of the three database structure described hereinabove in Figs. 3A and 3B. For simplicity, referring additionally to Fig. 3B, each image in Fig. 4 is built up of corresponding partial frames in successive data blocks which are cumulatively received. Thus, for simplicity, one may consider the five images in a first horizontal row, to correspond to five interactively viewable frames in a first block of data, each successive frame typically illustrating a successive position of an imaged model.

Considering each successive horizontal row in Fig. 4, it can be seen that the quality of the images improves successively from the top row to the bottom row. This corresponds to the quality improvement sensed by a viewer as successive blocks of data are incorporated in the information made available to the viewer.

It is appreciated that in accordance with a preferred embodiment of the present invention, a limited bandwidth user first receives the first row of images and is immediately able to interact therewith. Over time, depending on the bandwidth available to that user, successive data blocks are received, each cumulatively enhancing the quality. It is a particular feature of the invention that during receipt of successive blocks of data, the user is able to fully interact with the images.

Reference is now made to Fig. 5 showing a production tool 71 which accepts as an input a sequence of original digital frame units 72, integrated into a digital multimedia file 73. The production tool 71 includes an encoder unit 74 which operates by partitioning and compressing the digital multimedia file 73 into a scalable progressive database 75 comprised of data blocks 76. Successive blocks combine together to form higher bandwidth versions *a* to *n* of the media. Database 75 is stored on server 77. The production tool 71 enables the producer to control the bandwidth or quality granularity through control parameters 78. These parameters are used to calculate the data block sizes and compression settings within encoder unit 74.

Reference is now made to Fig. 6 showing the structure of data block 76 in scalable progressive database 75. Where a random access server is available, selective encoded frames 72 from block 76 are accessed at 79 on the server 77 database, based on interactive requests coming from the client. The encoded frames are transmitted from server 77 to a client computer 80 and integrated within client database 81, to mirror server database 75. It is appreciated that both sequential and random access servers may be advantageously employed in the present invention, although random access servers are preferred.

Reference is now made to Fig. 7 showing the decoder on a client computer 80. The client computer 80 receives from server 77 (Fig. 5) into a buffer 82 a series of data blocks 76 from scalable database 75. As the blocks are received, a client database, which mirrors the server database, is built up. A decoder unit 83 decompresses the blocks and an accumulator unit 84 integrates them to form a suitable low quality version 85 of the multimedia file 73 (Fig. 5), which is stored in a buffer 86, thus building up an interactive database.

The operations of the decoder 83 and accumulator 84 are governed by a CPU 87. They may operate in either order; i.e., the decoding may be carried out before the accumulation, or the accumulation may be carried out before the decoding. The multimedia file 73 is played on a player unit 88 in response to interactive user commands.

As the user interactively requests specific frames to be played, the buffer 86 supplies the highest quality version which it possesses. If the desired frame is not available, the buffer 86 sends back a request to buffer 82 to decode and accumulate that frame. If the frame is also not available in buffer 82, then that buffer 82 sends back a request to the server database 75 to transmit the frame. As playback continues and the bandwidth frees, additional data blocks 76 are received and integrated with the previously received blocks into higher quality versions 89 of the multimedia file.

The description of Figs. 1 - 7 has been directed towards the overall system and method provided by the present invention. The description which follows is directly principally to particular applications of the system and method described hereinabove.

Reference is now made to Fig. 8 showing a progressive database 75 for a digital video file which corresponds to digital multimedia file 73 (Fig. 5). The progressive database 75 comprises data blocks, where the first ones of data blocks 76 are used to create a preview 90 of the video, and the second ones of data blocks 76 are accumulated with the first blocks to create a full view 91 of the video. The views are stored in the interactive database buffer 86 and played in player unit 88, in response to interactive user commands.

Reference is now made to Fig. 9 showing a system for incorporating a progressive scalable database into a time-based video frame sequence with a macro and micro time scale, such as the one used in Apple's QUICKTIME® movies in accordance with a preferred embodiment of the present invention. Individual frames 72 are arranged according to a macro time scale, denoted by major axis markings 92 in Fig. 9. Each frame is displayed at the respective macro times indicated by markings 92.

Between successive frames 72, small fluctuations can be introduced. For example, the major time scale can be displaying a bird flying, and the minor time scale can be used to add fluttering to the bird's wings. The fluctuations typically involve only a small portion of the image area, and are displayed in rapid succession, according to minor axis markings 93 in Fig. 9. Such a time based sequence allows the decoder, which does the intensive processing to supply the frames 72, to run at a slow rate; e.g. 3 fps, whereas the viewer, doing the less intensive processing, can be playing at a fast rate; e.g. 30 fps. The fluctuations must be simple enough, though, that they can be rendered at the full 30 fps rate.

The present invention includes a novel use of such a two-scale time based system, to enable progressive streaming. In order to utilize the progressive database of the present invention in viewing such time-based video sequences, the minor time scale 93 is used to display progressive versions of a frame, rather than to display fluctuations, as originally conceived in the prior art. At each time corresponding to a minor axis marking, the player displays the latest version of the frame which is available. For example, in Fig. 9 the low quality version 94 of frame #4, corresponding to the first block 76, is displayed over a duration of three minor axis marks, by which time the second block 76 has been accumulated to form the medium quality version 95. This medium quality version is then displayed over a duration of two minor axis marks, by which time the third block 76 has been accumulated to form the high quality version 96 of the media. This high quality version is then displayed for a duration of three further minor axis marks, following which the frame advances to frame #5.

The cycle then repeats for progressive display of frame #5. If additional blocks arrive and are accumulated for frame #4, they are displayed when the video sequence is replayed. Thus it is seen that the progressive dimension of the database can be incorporated within the interactive dimension, through the use of a minor time scale, which is situated within the major time scale used for advancing the frames. The major time scale is the interactive axis, and the minor time scale becomes the progressive axis. The overall accomplishment is to enable viewing of the video before the full media stream has been downloaded. Initially, the low quality frames are displayed, and while additional blocks are downloaded and accumulated in background, the quality of the frames steadily improves.

In accordance with a preferred embodiment of the present invention, for the user to be able to view the video immediately, without waiting for the entire file to download, a production tool must store the encoded video in the order of successive blocks. Once the first block is downloaded, the video can already be viewed. The order of the encoded data items must therefore be:

Frame 1/Block 1, Frame 2/Block 1, ..., Frame *m* /Block 1,...,
Frame 1/Block *n*, Frame 2/Block *n*, ..., Frame *m* /Block *n*

Each row in this order comprises one entire block. In order to integrate this file format within the two-scale time-based system, each partial frame must be handled as if it were an entire frame. That is, the production tool must treat the movie as if there were a total of $m \cdot n$ distinct frames being encoded. The player in turn, however, must know that although it is receiving what appears to be $m \cdot n$ encoded frames, there are really only a total of m frames. It must decode and accumulate every successive sequence of m data items (i.e., each row of the above sequence) with the previous ones, to update the frames.

On the other hand, the natural order in which the production tool produces the encoded data items is:

Frame 1/Block 1, Frame 1/Block 2, ..., Frame 1/Block n, ...,
Frame m /Block 1, Frame m /Block 2, ..., Frame m /Block n

This is because each frame is sent only once to the codec for encoding, and returned as a series of encoded partial frames. Thus it is necessary to post-process the encoded data file, to rearrange the data items from this latter frame dominated order to the former block dominated order. This rearrangement process is referred to as "flattening" in the art, and is illustrated in Fig. 10.

As seen in Fig. 10, the frames 72 are supplied in sequence to the production tool, which produces a series of partial blocks 76 for each frame. The partial blocks are reordered, as indicated by the mapping in Fig. 10, into a single file stream 97. As indicated, the first partial blocks of each frame form the first m data units in the file 97, the second set of partial blocks of each frame form the next m data units, etc.

Reference is now made to Fig. 11 which shows a proxy system 98 for caching in a central hub 99 multimedia data which streams from servers 77 to clients 80. The servers 77 store their multimedia data in the progressive scalable representation described above in the server database 75, with each media encoded into data blocks 76 B_1, B_2, ..., B_n. Multiple servers 77 are connected through the central hub 99 to clients 80 via data communication channels. These channels are of various bandwidths. Whenever a client 80 issues a request for data from a specific server, a proxy computational unit 100 first determines, based on the server bandwidth, which data blocks are to be transmitted to the client. If those data blocks are not already cached in the hub, then the proxy 98 retrieves the required blocks 76 from the server along low bandwidth communication channels 101, at the appropriate bandwidth, and delivers them to the client 80 along high bandwidth communication channels 102. The blocks 76 are also cached in the hub 99.

There is provided an inventory flag database 103 on the hub which keeps a record of which data blocks are available. If some of the required data blocks 76 are already cached, then the proxy computational unit 100 computes which data blocks must be delivered from the server 77 in order to transmit to the client the highest quality version of the media possible, within the bandwidth constraint. The proxy 98 then retrieves the required blocks 76 from the server 77 along low bandwidth communication channels 101, stores them in its cache and delivers them to the client 80 along high bandwidth communication channels 102. A decoder unit 83 on the client computer decodes the data blocks received, and an accumulator unit 84 integrates them. The proxy 98 may also have its own decoder unit 83 and accumulator unit 84, which converts the data from its original compressed form on the server database 77 to an intermediate compressed form on the hub database 104, one which is faster to decompress than that of the client database. The proxy accumulation unit 84 may perform the necessary data block accumulation to store on the hub all possible versions of the multimedia in its intermediate compressed form, in which case the client accumulator 84 unit is unnecessary.

An update communication line 105 links servers 77 to the proxy 98, through which servers 77 can notify proxy 98 if any of the multimedia files have been updated. If proxy 98 receives such notification, then it clears its cache of any data blocks associated with those updated files, and resets its inventory flags 103, so that upon future client requests it will know that it has to retrieve the updated files from the server again.

There are several ways to generate the data blocks $B_1, B_2, \ldots, B_n$ for the progressive scalable database, so as to satisfy the user-selected constraints that

(*i*) for each $1 \leq k \leq n$, the first consecutive compressed data blocks $B_1, B_2, \ldots, B_k$ when integrated together produce a version of the media at a quality level commensurate with bandwidth f_k;

(*ii*) the sizes of the compressed data blocks are such that the first consecutive compressed data blocks $B_1, B_2, \ldots, B_k$ when transmitted at bandwidth f_k suffice to enable on-line playback of the media version.

Reference is now made to Fig. 12 which shows a system for generating a progressive scalable database from digital media data by cascading compressors in tandem. The user selects bit-rate and quality control parameters 106 for the encoding. The original multimedia data file 73 is input to a compressor 107 along with the user-selected control parameters 106, resulting in compressed data 108 adapted to a user-selected bandwidth. The compressed data 108 is transmitted to the scalable database 75 as the first data block 76. It is also transmitted to decompressor 109, which reconstructs the media as it would be generated on the client side.

The reconstructed data 110 is subtracted from the original data 73 to arrive at a residual 111. The residual 111 is fed back to the compressor 107 in a feedback loop 112, and compressed by compressor 107 with bit-rate control so that the compressed data 108 is adapted to the difference between the first and second user-selected bandwidths. The compressed data is transmitted to the progressive scalable database 75 as the second data block 76, and the loop continues repeatedly until the user-selected final quality is achieved.

It is not necessary for the compressor 107 to use the same compression method each time it operates. Rather, it can use a block identifier 113 as a parameter for switching between methods. For example, with a video encoder, the first block could be encoded using a low quality version of H.263 and successive blocks could be encoded using spatial vector quantization and temporal wavelets.

One possible approach to compression is the use of fractal technology, such as that described and claimed in applicant's U.S. Patent No. 5,497,435, the disclosure of which is hereby incorporated by reference.

Another approach to generating a scalable database is through the use of progressive JPEG. The progressive JPEG standard allows the encoder to segment the compression into spectral selection and successive approximation scans. In spectral selection the DCT coefficients are grouped into spectral bands, and in successive approximation the bits used to represent them are divided into lower and higher precision information. Progressive JPEG is described by Pennebaker, W. B. and Mitchell, J. L. in *JPEG: Still Image Data Compression*, Van-Nostrand Reinhold, New York, 1993, the disclosure of which is hereby incorporated by reference.

Reference is now made to Fig. 13 which shows a decoder for the scalable database 75 on the client side, corresponding to the encoder from Fig. 12. Blocks 76 are successively transmitted from the server to the client and are decompressed in decoder 83. The decoded data is integrated in an accumulator 84, which converts it into a form compatible with the player 88, and stores it in the interactive buffer 86. While the player is showing the media, additional blocks 76 are received in the background, decoded and accumulated, so that the media quality is upgraded when it is replayed. The player 88 requests frames from buffer 86, and if they are not available then buffer 86 requests them from database 75. A block identifier 113 provides an input to the decoder as each block is decoded, so that the decoder can apply the appropriate decompression method for that data block, corresponding to the compression method which was performed in compressor 107.

Reference is now made to Fig. 14 which shows a progressive database for a large still image. The first encoded frame unit of a data block consists of the full image sub-sampled 4:1 in each dimension. The second set of frames consist of four encoded frame units corresponding to each of the quadrants of the full image, sub-sampled at 2:1. Although this second set of frames appears to be four times the size of the first frame, it can be stored using only three times as much data since the decoder can also accumulate data from the previously displayed frame. The third set of frames includes sixteen encoded frame units, each corresponding to a quadrant of a quadrant of the full image, but at the original scale. Again, this third set of frames stores three times as much data as the second set. These three sets of frames comprise the entire encoded data block, and their sum total is, of course, the same amount of data as the full image at the original scale. (The effect of the compression is being ignored here.) The frames are all arranged sequentially in the encoded data block 76. The mapping from multi-resolution image tiles to sequential frames 72 is shown in Fig. 14.

When the frame #1 is downloaded, decoded and displayed, the viewer sees a low resolution version of the full image. In the example shown in Fig. 14, the viewer clicks on a hot spot in the northwest quadrant and link 114 then transfers over to the northwest frame in the second set of frames, which corresponds to frame #2 in Fig. 14. Displaying this new frame gives the appearance of having zoomed in on the quadrant. The viewer next clicks on a hot spot in the southeast quadrant of the frame being displayed from the second set, and link 114 then transfers over to the indicated frame in the third set of frames, which corresponds to frame #11 in the Fig. 14, giving the effect of yet another zoom in.

While the viewer is looking at the first frame, the data from the second set of frames is being downloaded in the background. Once received, this data is decoded and stored in the client interactive database. When the hot spot in the northwest quadrant is clicked, the client CPU looks for the desired frame #2 in its interactive database. If the frame is already present, it is delivered to the viewer for immediate display; otherwise the interactive buffer sends the request back to the progressive database. The progressive database can access and send the specific encoded frame required from the second set of frames, without sending all of the frames. Finally, as successive blocks are integrated into the interactive database, the quality of the individual frames is enhanced, and the zoomed in tile looks progressively better and better, having cumulatively enhanced quality.

It is evident to those skilled in the art that the above discussion applies to animations and panoramas of large images which contain areas of interest such as sprites. The frames comprising the interactivity dimension of the database would correspond to multi-resolution tiles from each of the individual images in the animation or panorama.

Fig. 15 illustrates a system for transmitting VRML images over a server/client network. A VRML database 121 is stored on a server. A client 80 interactively controls the VRML viewing parameters 122 through use of a mouse and keyboard 23. The viewing parameters 122 are used in conjunction with the VRML database 121 to render a raster bitmap image 123 of the VRML object on the server computer. If the bitmap image corresponding to the viewing parameters was already rendered previously, then the frame data for the bitmap in the client database 42 is used for display by the player 88. The raster bitmap is encoded into partial frames by encoder 74, and the partial frames are inserted into a server two-dimensional database 41. The server database is continually streamed to the client, building up the client database 42. The client database 42 is used to provide the frames for display.

Fig. 16 depicts a flowchart for the VRML application shown in Fig. 15. At step 131, the user interacts with the mouse and keyboard, and the client computer updates the viewing parameters. At step 132 the client computer checks whether or not those viewing parameters have already been processed. Although the viewing parameters can vary continuously, a preferred embodiment of the present invention discretizes them to a finite number of settings; for example, 10° resolution for angles. This makes it likely that the user will navigate back to the same settings used earlier.

If the viewing parameters are new, then they are sent to the server computer, which renders the VRML database into a raster bitmap corresponding to the specific viewing parameters selected, at step 133. At step 134 the bitmap is encoded into partial frames, and incorporated into the server database. Step 135 is continually operative to transmit additional encoded data from the server to the client. As data is received on the client, the client database is built up at step 136. At step 137 the client database generates the latest version of the bitmap on demand, and displays it. This step is also carried out whenever step 132 results in confirmation that the viewing parameters have already been processed.

It is evident to those skilled in the art that the above discussion applies to CAD/CAM models for three-dimensional objects, as well as VRML models. Specifically, Fig. 17 illustrates a typical two-dimensional array of CAD/CAM images, indexed vertically according to progressive coordinate and horizontally according to interactive coordinate. It is appreciated that the images in a given horizontal row may be viewed interactively. Each successive horizontal row of images is built up over time at a rate determined by bandwidth availability and has increased quality inasmuch as it is based on an increasing number of data blocks.

It will be appreciated by persons skilled in the art that the present invention is not limited by what has been particularly shown and described hereinabove but extends also to embodiments which would naturally occur to persons reading the above description and to combinations and subcombinations of embodiments described hereinabove.

CLAIMS

1. A system for transmitting digital data representing an original over plural transmission links, at least some of which have limited bandwidth, comprising:

a digital data source storing digital data representing the original;

a digital data receiver receiving said digital data representing the original via one of said plural transmission links having limited bandwidth; and

a digital data transmitter operative to transmit said digital data representing the original to said receiver over a transmission link having a limited bandwidth in plural blocks which are sequentially transmitted at a rate determined by said limited bandwidth, each block being an incomplete collection of data which includes parts of multiple frames, each frame being viewable in a selectable order by said receiver even when less than all of said plural blocks have been received, receipt of subsequent blocks by the receiver being used to cumulatively improve the quality of the digital data viewed by the receiver.

2. A system according to claim 1 and wherein a first one of said plural blocks contains digital data which represents a first approximation to said original.

3. A system according to claim 2 and wherein additional ones of said plural blocks, when combined with said first one of said plural blocks provide additionally accurate approximations to said original.

4. A system according to claim 1 and wherein each of said multiple frames comprises a portion of data which can be independently and interactively manipulated.

5. A system according to claim 1 and also comprising a block generator operative to receive digital data representing said original and to provide said plural blocks.

6. A system according to claim 5 and wherein said block generator is operative to provide plural blocks which are distinguished from each other by their respective frequency bands.

7. A system according to claim 5 and wherein said block generator comprises a fractal compression engine.

8. A system according to claim 7 and wherein said block generator is operative to decompose said digital data representing the original in relatively high frequency and relatively low frequency digital data portions, and wherein a first plurality of blocks containing said relatively low frequency portion is transmitted by said data transmitter prior to transmission of a second plurality of blocks containing said relatively high frequency portion.

9. A system according to claim 5 and wherein said block generator is operative to provide plural blocks by sampling said received digital data.

10. A system according to claim 9 and wherein the sampling rate of a plurality of blocks is equal to the sum of the sampling rates of individual ones of said plurality of blocks.

11. A system according to claim 6 and wherein said block generator comprises a wavelet encoder.

12. A system according to claim 5 and wherein said block generator comprises a quantizer which produces blocks each of which contain quantized data of a different order, such that accumulation of multiple blocks provides combined data of greater precision than that contained in any single block.

13. A block generator comprising:

a producer interface; and

a digital data compressor, operative in response to producer control parameters received via said producer interface for receiving digital data representing the original and providing plural blocks, each block being an incomplete collection of data which includes parts of multiple frames.

14. A block generator according to claim 13 and being operative to provide plural blocks which are distinguished from each other by their respective frequency bands.

15. A block generator according to claim 13 and comprising a fractal compression engine.

16. A block generator according to claim 15 and being operative to decompose said digital data representing the original in relatively high frequency and relatively low frequency digital data portions, and wherein a first plurality of blocks containing said relatively low frequency portion is transmitted by said data transmitter prior to transmission of a second plurality of blocks containing said relatively high frequency portion.

17. A block generator according to claim 13 and being operative to provide plural blocks by sampling said received digital data.

18. A block generator according to claim 17 and wherein the sampling rate of a plurality of blocks is equal to the sum of the sampling rates of individual ones of said plurality of blocks.

19. A block generator according to claim 14 and comprising a wavelet encoder.

20. A block generator according to claim 13 and comprising a quantizer which produces blocks each of which contain quantized data of a different order, such that accumulation of multiple blocks provides combined data of greater precision than that contained in any single block.

21. A block generator according to claim 15 and comprising a quantizer which produces blocks each of which contain quantized data of a different order, such that accumulation of multiple blocks provides combined data of greater precision than that contained in any single block.

22. A block generator according to claim 17 and comprising a quantizer which produces blocks each of which contain quantized data of a different order, such that accumulation of multiple blocks provides combined data of greater precision than that contained in any single block.

23. A block generator according to claim 20 and comprising a quantizer which produces blocks each of which contain quantized data of a different order, such that accumulation of multiple blocks provides combined data of greater precision than that contained in any single block.

24. A digital data receiver comprising:

a data receipt interface receiving digital data representing an original in a plurality of sequential blocks, each block being an incomplete collection of data which includes parts of multiple frames;

a block accumulator for combining plural blocks as they are received for viewing by the recipient; and

a viewer including a recipient interface which permits each frame to be viewed in an order selected by the recipient, even when less than all of said plural blocks have been received, combining of plural blocks by said block accumulator being used to improve the quality of the digital data viewed by said recipient.

25. A digital data receiver according to claim 24 and wherein said block accumulator is operative to combine plural blocks which are distinguished from each other by their respective frequency bands.

26. A digital data receiver according to claim 24 and comprising a fractal decompression engine.

27. A digital data receiver according to claim 26 and wherein said data receipt interface is operative to initially receive a first plurality of blocks containing relatively low frequency data and thereafter receive a second plurality of blocks containing relatively high frequency data and wherein said block accumulator is operative to reconstitute said digital data representing the original from said blocks representing relatively high frequency and relatively low frequency data.

28. A digital data receiver according to claim 24 and wherein said block accumulator is operative to combine plural blocks having different sampling.

29. A digital data receiver according to claim 28 and wherein the sampling rate of a combined plurality of blocks is equal to the sum of the sampling rates of individual ones of said plurality of blocks.

30. A digital data receiver according to claim 25 and comprising a wavelet decoder.

31. A digital data receiver according to claim 24 and wherein said block accumulator comprises a dequantizer which combines blocks each of which contain quantized data of a different order, such that accumulation of multiple blocks provides combined data of greater precision than that contained in any single block.

32. A digital data receiver according to claim 26 and wherein said block accumulator comprises a dequantizer which combines blocks each of which contain quantized data of a different order, such that accumulation of multiple blocks provides combined data of greater precision than that contained in any single block.

33. A digital data receiver according to claim 30 and wherein said block accumulator comprises a dequantizer which combines blocks each of which contain quantized data of

a different order, such that accumulation of multiple blocks provides combined data of greater precision than that contained in any single block.

34. A digital data transmitter actuator comprising:

an organizer operative, when actuated, to access digital data representing the original which is organized in plural blocks for subsequent transmission, each block being an incomplete collection of data which includes parts of multiple frames, each frame being viewable in a selectable order by said a receiver even when less than all of said plural blocks have been received; and

a receiver instruction interface responsive to interactive inputs from a receiver for actuating said organizer to select a given block and at least one given partial frame within said given block for transmission.

35. A method for transmitting digital data representing an original over plural transmission links at least some of which have limited bandwidth comprising:

storing digital data representing the original;

receiving at a receiver said digital data representing the original via one of said plural transmission links having limited bandwidth; and

transmitting said digital data representing the original to said receiver over a transmission link having a limited bandwidth in plural blocks which are sequentially transmitted at a rate determined by said limited bandwidth, each block being an incomplete collection of data which includes parts of multiple frames, each frame being viewable in a selectable order by said receiver even when less than all of said plural blocks have been received, receipt of subsequent blocks by the receiver being used to cumulatively improve the quality of the digital data viewed by the receiver.

36. A method according to claim 35 and wherein a first one of said plural blocks contains digital data which represents a first approximation to said original.

37. A method according to claim 36 and wherein additional ones of said plural blocks, when combined with said first one of said plural blocks provide additionally accurate approximations to said original.

38. A method according to claim 35 and wherein each of said multiple frames comprises a portion of data which can be independently and interactively manipulated.

39. A method according to claim 35 and also comprising the step of block generation operative to receive digital data representing said original and to provide said plural blocks.

40. A method according to claim 39 and wherein said block generation is operative to provide plural blocks which are distinguished from each other by their respective frequency bands.

41. A method according to claim 39 and wherein said block generation employs fractal compression.

42. A method according to claim 41 and wherein said block generation includes decomposing said digital data representing the original in relatively high frequency and relatively low frequency digital data portions, and wherein a first plurality of blocks containing said relatively low frequency portion is transmitted prior to transmission of a second plurality of blocks containing said relatively high frequency portion.

43. A method according to claim 39 and wherein said block generation provides plural blocks by sampling said received digital data.

44. A method according to claim 43 and wherein the sampling rate of a plurality of blocks is equal to the sum of the sampling rates of individual ones of said plurality of blocks.

45. A method according to claim 40 and wherein said block generation comprises wavelet encoding.

46. A method according to claim 39 and wherein said block generation comprises quantizing into blocks each of which contain quantized data of a different order, such that accumulation of multiple blocks provides combined data of greater precision than that contained in any single block.

47. A method for digital data transmission comprising:

organizing digital data representing the original into plural blocks for subsequent transmission, each block being an incomplete collection of data which includes parts of multiple frames, each frame being viewable in a selectable order by said a receiver even when less than all of said plural blocks have been received;

responsive to interactive inputs from a receiver for actuating said organizer, selecting a given block and at least one given partial frame within said given block for transmission; and

transmitting the selected given block and at least one given partial frame to a user.

48. A method for providing on-line virtual reality movies, comprising:

inputting a cyclic movie sequence into an encoder; determining the number of portions that each frame of said movie is divided into, and forming partial frames;

specifying hot-spots and independent objects for interaction within a partial frame;

transmitting the partial frames part by part to a user's asynchronous database, and

displaying said frames on a user's interface.

49. The method as claimed in claim 48, wherein said partial frames are generated by forming a partial resolution frame and subtracting same from the original frame from which it was formed.

50. The method as claimed in claim 48, wherein said partial frames are generated by repeating the process several times.

51. The method as claimed in claim 48, wherein said partial frames are compression encoded prior to being transmitted and compression decoded prior to being displayed on the user's interface.

52. A system for providing on-line virtual reality movies, comprising:

a production workstation for producing VR movies;

an encoder for preparing the VR movie for transmission; and

a server including a repository for the VR movie and a transceiver for transmitting the movie part by part to a user, upon request.

53. The system as claimed in claim 52, wherein said encoder comprises a partial frame generator and a controller.

54. The system as claimed in claim 52, wherein the encoder further comprises a compressor for compression encoding image parts produced in said workstation.

55. The system as claimed in claim 52, further comprising

a user's transceiver for receiving VR movie parts transmitted by said server;

an asynchronous database for supplying the user with images upon request, independent of transmission, and a decoder for integrating partial frames and facilitating a progressively updated display of images; and

a user's workstation for interaction with the VR movie.

56. The system as claimed in claim 55, wherein the decoder further comprises a decompressor and a partial frame integrator.

57. A method for providing on-line virtual reality movies, according to claim 48, substantially as hereinbefore described and with reference to the accompanying drawings.

58. A system for providing on-line virtual reality movies, according to claim 52, substantially as hereinbefore described and with reference to the accompanying drawings.

59. A method for encoding original digital video data to be stored on a server computer for on-line delivery to client computers, comprising the steps of:

encoding the digital video into a database comprising a series of encoded data blocks, each block comprising a sequence of encoded frames, with the property that successive blocks when decoded and integrated together provide successively higher bandwidth versions of the video for on-line playback;

storing the database on a server computer;

processing a request by a client computer for on-line delivery of the video in order to determine which data blocks to transmit, so as to accommodate the client bandwidth;

transmitting the necessary data blocks to the client;

decoding the data blocks on the client computer;

integrating the data blocks together on the client computer to reconstruct an appropriate version of the original digital video; and

playing the reconstructed video on the client computer.

60. A method according to Claim 59 wherein said step of encoding includes a bit-rate control device enabling the producer to pre-select the sequence of bandwidths or quality levels for the database.

61. A method according to Claim 59 wherein said step of encoding is performed in such a way that the first blocks of the database correspond to previews of the video.

62. A method according to Claim 59 wherein said steps of transmitting, decoding, integrating and playing are repeated in succession a number of times in order to transmit additional data blocks to the client, thereby upgrading the quality of the video while it is replayed.

63. A method for encoding original digital audio data to be stored on a server computer for on-line delivery to client computers, comprising the steps of:

encoding the digital audio into a database comprising a series of encoded data blocks, each block comprising a sequence of encoded frames, with the property that successive blocks when decoded and integrated together provide successively higher bandwidth versions of the audio for on-line playback;

storing the database on a server computer;

processing a request by a client computer for on-line delivery of the audio in order to determine which data blocks to transmit, so as to accommodate the client bandwidth;

transmitting the necessary data blocks to the client;

decoding the data blocks on the client computer;

integrating the data blocks together on the client computer to reconstruct an appropriate version of the original digital audio; and

playing the reconstructed audio on the client computer.

64. A method according to Claim 63 wherein said step of encoding includes a bit-rate control device enabling the producer to pre-select the sequence of bandwidths or quality levels for the database.

65. A method according to Claim 63 wherein said steps of transmitting, decoding, integrating and playing are repeated in succession a number of times in order to transmit additional data blocks to the client, thereby upgrading the quality of the audio while it is replayed.

66. A method for encoding original digital object movie data to be stored on a server computer for on-line delivery to client computers, comprising the steps of:

encoding the digital object movie into a database comprising a series of encoded data blocks, each block comprising a sequence of encoded frames, with the property that successive blocks when decoded and integrated together provide successively higher bandwidth versions of the object movie for on-line playback;

storing the database on a server computer;

processing a request by a client computer for on-line delivery of the object movie in order to determine which data blocks to transmit, so as to accommodate the client bandwidth;

transmitting the necessary data blocks to the client;

decoding the data blocks on the client computer;

integrating the data blocks together on the client computer to reconstruct an appropriate version of the original digital object movie; and

playing the reconstructed object movie on the client computer.

67. A method according to Claim 66 wherein said step of encoding includes a bit-rate control device enabling the producer to pre-select the sequence of bandwidths or quality levels for the database.

68. A method according to Claim 66 wherein said steps of transmitting, decoding, integrating and playing are repeated in succession a number of times in order to transmit additional data blocks to the client, thereby upgrading the quality of the object movie while it is replayed.

69. A method for encoding an original digital panorama to be stored on a server computer for on-line delivery to client computers, comprising the steps of:

encoding the digital panorama into a database comprising a series of encoded data blocks, each block comprising a sequence of encoded frames, with the property that successive blocks when decoded and integrated together provide successively higher bandwidth versions of the panorama for on-line playback;

storing the database on a server computer;

processing a request by a client computer for on-line delivery of the panorama in order to determine which data blocks to transmit, so as to accommodate the client bandwidth;

transmitting the necessary data blocks to the client;

decoding the data blocks on the client computer;

integrating the data blocks together on the client computer to reconstruct an appropriate version of the original digital panorama; and

playing the reconstructed panorama on the client computer.

70. A method according to Claim 69 wherein said step of encoding includes a bit-rate control device enabling the producer to pre-select the sequence of bandwidths or quality levels for the database.

71. A method according to Claim 69 wherein said steps of transmitting, decoding, integrating and playing are repeated in succession a number of times in order to transmit additional data blocks to the client, thereby upgrading the quality of the panorama while it is replayed.

72. A method for encoding original digital large still image data to be stored on a server computer for on-line delivery to client computers, comprising the steps of:

encoding the large digital image into a database comprising a series of encoded data blocks, each block comprising a sequence of encoded multi-resolution tiles of the image, with the property that successive blocks when decoded and integrated together provide successively higher quality versions of the tiles for display;

storing the database on a server computer;

processing a request by a client computer for on-line delivery of the image in order to determine which data blocks to transmit;

transmitting the necessary data blocks to the client;

decoding the data blocks on the client computer;

integrating the data blocks together on the client computer to reconstruct an appropriate version of the original multi-resolution image tiles; and

interactively displaying the reconstructed tiles on the client computer.

73. A method according to Claim 72 wherein said step of encoding includes a compression control device enabling the producer to pre-select the sequence of quality levels for the database.

74. A method according to Claim 73 wherein said step of encoding operates on a plurality of images forming an animation, and each encoded data block is comprised of multi-resolution tiles from the plurality of images.

75. A video processing system operative on digital video data for encoding the digital video, storing it on a server computer and delivering it to client computers on-line upon request comprising:

an encoder for compressing the digital video into a database comprising a series of encoded data blocks, each block comprising a sequence of encoded frames, with the property that successive blocks when decoded and integrated together provide successively higher bandwidth versions of the video for on-line playback;

a storage device for archiving the database on a server computer;

a processing unit for accepting a request by a client computer for on-line delivery of the video and determining which data blocks to transmit, so as to accommodate the client bandwidth;

a transmitter for delivering the necessary data blocks to the client;

a decoder for decompressing the data blocks back into video data on the client computer;

an accumulator for integrating the data blocks together on the client computer to reconstruct an appropriate version of the original digital video; and

a player on the client computer for playing the reconstructed digital video.

76. The system as claimed in Claim 75 wherein said encoder includes a bit-rate controller enabling the user to pre-select the sequence of bandwidths or quality levels for the database.

77. The system as claimed in Claim 75 wherein said encoder compresses the digital video in such a way that the first blocks of the database correspond to previews of the video.

78. The system as claimed in Claim 75 wherein the transmitter, decoder, accumulator and player repeatedly operate in succession a number of times in order to transmit additional data blocks to the client, thereby upgrading the quality of the video while it is being replayed.

79. An audio processing system operative on digital audio data for encoding the digital audio, storing it on a server computer and delivering it to client computers on-line upon request comprising:

an encoder for compressing the digital audio into a database comprising a series of encoded data blocks, each block comprising a sequence of encoded frames, with the property that successive blocks when decoded and integrated together provide successively higher bandwidth versions of the audio for on-line playback;

a storage device for archiving the database on a server computer;

a processing unit for accepting a request by a client computer for on-line delivery of the audio and determining which data blocks to transmit, so as to accommodate the client bandwidth;

a transmitter for delivering the necessary data blocks to the client;

a decoder for decompressing the data blocks back into audio data on the client computer;

an accumulator for integrating the data blocks together on the client computer to reconstruct an appropriate version of the original digital audio; and

a player on the client computer for playing the reconstructed digital audio.

80. The system as claimed in Claim 79 wherein said encoder includes a bit-rate controller enabling the user to pre-select the sequence of bandwidths or quality levels for the database.

81. The system as claimed in Claim 79 wherein the transmitter, decoder, accumulator and player repeatedly operate in succession a number of times in order to transmit additional data blocks to the client, thereby upgrading the quality of the audio while it is being replayed.

82. An object movie processing system operative on digital object movie data for encoding the digital object movie, storing it on a server computer and delivering it to client computers on-line upon request comprising:

an encoder for compressing the digital object movie into a database comprising a series of encoded data blocks, each block comprising a sequence of encoded frames, with the property that successive blocks when decoded and integrated together provide successively higher bandwidth versions of the object movie for on-line playback;

a storage device for archiving the database on a server computer;

a processing unit for accepting a request by a client computer for on-line delivery of the object movie and determining which data blocks to transmit, so as to accommodate the client bandwidth;

a transmitter for delivering the necessary data blocks to the client;

a decoder for decompressing the data blocks back into object movie data on the client computer;

an accumulator for integrating the data blocks together on the client computer to reconstruct an appropriate version of the original digital object movie; and

a player on the client computer for playing the reconstructed digital object movie.

83. The system as claimed in Claim 82 wherein said encoder includes a bit-rate controller enabling the user to pre-select the sequence of bandwidths or quality levels for the database.

84. The system as claimed in Claim 83 wherein the transmitter, decoder, accumulator and player repeatedly operate in succession a number of times in order to transmit additional data blocks to the client, thereby upgrading the quality of the object movie while it is being replayed.

85. A panorama processing system operative on digital panorama data for encoding the digital panorama, storing it on a server computer and delivering it to client computers on-line upon request comprising:

an encoder for compressing the digital panorama into a database comprising a series of encoded data blocks, each block comprising a sequence of encoded frames, with the property that successive blocks when decoded and integrated together provide successively higher bandwidth versions of the panorama for on-line playback;

a storage device for archiving the database on a server computer;

a processing unit for accepting a request by a client computer for on-line delivery of the panorama and determining which data blocks to transmit, so as to accommodate the client bandwidth;

a transmitter for delivering the necessary data blocks to the client;

a decoder for decompressing the data blocks back into panorama data on the client computer;

an accumulator for integrating the data blocks together on the client computer to reconstruct an appropriate version of the original digital panorama; and

a player on the client computer for playing the reconstructed digital panorama.

86. The system as claimed in Claim 85 wherein said encoder includes a bit-rate controller enabling the user to pre-select the sequence of bandwidths or quality levels for the database.

87. The system as claimed in Claim 85 wherein the transmitter, decoder, accumulator and player repeatedly operate in succession a number of times in order to transmit additional data blocks to the client, thereby upgrading the quality of the panorama while it is being replayed.

88. An image processing system operative on large digital image data for encoding the digital image, storing it on a server computer and delivering it to client computers on-line upon request comprising:

an encoder for compressing the large digital image into a database comprising a series of encoded data blocks, each block comprising a sequence of encoded multi-resolution tiles of the image, with the property that successive blocks when decoded and integrated together provide successively higher quality versions of the image tiles;

a storage device for archiving the database on a server computer;

a processing unit for accepting a request by a client computer for on-line delivery of the image and determining which data blocks to transmit;

a transmitter for delivering the necessary data blocks to the client;

a decoder for decompressing the data blocks back into image tile data on the client computer;

an accumulator for integrating the data blocks together on the client computer to reconstruct an appropriate version of the original multi-resolution image tiles; and

an interactive viewer on the client computer for displaying the reconstructed image tiles.

89. The system as claimed in Claim 88 wherein said encoder includes a compression controller enabling the user to pre-select the sequence of quality levels for the database.

90. The system as claimed in Claim 88 wherein said encoder operates on a plurality of images forming an animation, and each encoded data block is comprised of multi-resolution tiles from the plurality of images.

91. A method for caching of data which gets transmitted from servers to clients on a central hub within a network, comprising the steps of:

encoding digital multimedia data into databases comprising a series of encoded data blocks, each block comprising a sequence of encoded frames, with the property that successive blocks when decoded and integrated together provide successively higher bandwidth versions of the media for on-line playback;

storing the databases on a multitude of server computers;

managing within the hub requests by client computers for on-line delivery of media stored on server computers in order to determine which data blocks to transmit, so as to accommodate the client bandwidth;

transmitting the necessary data blocks from the server and from the hub to the client;

storing the data blocks delivered by the server in the cache residing in the central hub;

processing within the hub the data blocks it receives;

decoding the data blocks on the client computer;

integrating the data blocks together on the client computer to reconstruct an appropriate version of the original digital media; and

playing the reconstructed media on the client computer.

92. A method according to Claim 91 wherein said step of managing is performed by:

setting inventory flags to indicate which data blocks are currently stored in the hub.

93. A method according to Claim 92 wherein said step of managing further includes the steps of:

communicating with the servers to monitor which media data is outdated;

removing from cache the blocks corresponding to the media data which is outdated; and

resetting the inventory flags to indicate that the above blocks are no longer stored in the cache.

94. A method according to Claim 91 wherein said step of processing comprises the steps of:

decoding the data blocks received;

integrating the data blocks together to reconstruct appropriate versions of the original digital media; and

encoding the reconstructed media versions into an intermediate database for future transmission to the clients.

95. A proxy system operative on a server/client network for caching of data which gets transmitted from servers to clients on a central hub, comprising:

an encoder for compressing digital multimedia data into databases comprising a series of encoded data blocks, each block comprising a sequence of encoded frames, with the property that successive blocks when decoded and integrated together provide successively higher bandwidth versions of the media for on-line playback;

server communication lines from the servers to the hub for sending data blocks;

client communication lines from the hub to the clients for sending digital data;

storage devices for archiving the databases on a multitude of server computers;

a management unit within the hub to process requests by client computers for on-line delivery of media stored on server computers in order to determine which data blocks to transmit, so as to accommodate the client bandwidth;

a transmitter for delivering the necessary data blocks on the server communication lines from the server to the hub, and on the client communication lines from the hub to the client;

a storage device for saving the data blocks delivered by the server communication lines in the cache residing in the central hub;

a processing unit within the hub for processing the data blocks which the hub receives;

a decoder for decompressing the data blocks on the client computer;

an accumulator for integrating the data blocks together on the client computer to reconstruct an appropriate version of the original digital media; and

a player for playing the reconstructed media on the client computer.

96. The system as claimed in Claim 95 wherein said management unit operates by setting inventory flags to indicate which data blocks are currently stored in the hub.

97. The system as claimed in Claim 96 wherein said management unit operates by monitoring from the servers which media data is outdated, removing from cache the blocks corresponding to the media data which is outdated, and resetting the inventory flags to indicate that the above blocks are no longer stored in the cache.

98. The system as claimed in Claim 95 wherein said processing unit comprising:

a decoder for decompressing the data blocks received;

an accumulator for integrating the data blocks together to reconstruct appropriate versions of the original digital media; and

an encoder for compressing the reconstructed media versions into an intermediate database for future transmission to the clients.

99. A multi-casting unit (MCU) system operative on a broadcasting network for caching of data which gets transmitted from stations to viewers, comprising:

an encoder for compressing digital multimedia data into databases comprising a series of encoded data blocks, each block comprising a sequence of encoded frames, with the property that successive blocks when decoded and integrated together provide successively higher bandwidth versions of the media for on-line playback;

station communication lines from the stations to the MCU for sending data blocks;

viewer communication lines from the MCU to the viewers for sending data;

viewer receiver units for receiving the data sent by the MCU;

storage devices for archiving the databases on a multitude of station computers;

a management unit within the MCU to process requests by viewers for on-line delivery of media stored on station computers in order to determine which data blocks to transmit, so as to accommodate the viewer bandwidth;

a transmitter for delivering the necessary data blocks on the station communication lines from the station to the MCU, and on the viewer communication lines from the MCU to the viewer receiver units;

a storage device for saving the data blocks delivered by the station communication lines in the cache residing in the MCU;

a processing unit within the MCU for processing the data blocks which the MCU receives;

a decoder for decompressing the data blocks on the viewer receiver;

an accumulator for integrating the data blocks together on the viewer receiver unit to reconstruct an appropriate version of the original digital media; and

a player for playing the reconstructed media from the viewer receiver unit.

100. The system as claimed in Claim 99 wherein said management unit operates by setting inventory flags to indicate which data blocks are currently stored in the MCU.

101. The system as claimed in Claim 100 wherein said management unit operates by monitoring from the stations which media data is outdated, removing from cache the blocks corresponding to the media data which is outdated, and resetting the inventory flags to indicate that the above blocks are no longer stored in the cache.

102. The system as claimed in Claim 99 wherein said processing unit within the MCU comprising:

a decoder for decompressing the data blocks received;

an accumulator for integrating the data blocks together to reconstruct appropriate versions of the original digital media; and

an encoder for compressing the reconstructed media versions into an intermediate database for future transmission to the viewers.

103. A method for streaming multimedia data over a network, comprising the steps of:

encoding the media into a progressive database indexed according to frame and progressive block numbers;

serializing the encoded database;

storing the serialized database on a server;

streaming the serialized database to a client upon request;

creating a mirror copy of the encoded database on the client computer from the data which streams in; and

decoding the encoded database on the client computer into a sequence of frames for real time display.

104. A multimedia network streaming system, comprising:

an encoder for compressing the media into a progressive database indexed according to frame and progressive block numbers;

a sequencer for serializing the encoded database;

a storage device for archiving the serialized database on a server;

a transmitter for streaming the serialized database to a client upon request;

a processor for creating a mirror copy of the encoded database on the client computer from the data which streams in; and

a decoder for decompressing the encoded database on the client computer into a sequence of frames for real time display.

105. A system for transmitting model based data representations of three dimensional images over plural transmission links having limited bandwidth, said system comprising:

a digital data source storing model based data representations of three dimensional images;

an image processor for rendering views of said model based data representations into raster bitmap format;

a digital data receiver receiving said digital data in said raster bitmap format over a one of said plural transmission links having limited bandwidth; and

a digital data transmitter operative to transmit said digital data in said raster bitmap format to said receiver over a transmission link having a limited bandwidth in plural blocks which are sequentially transmitted at a rate determined by said limited bandwidth, each block being an incomplete collection of data which includes parts of multiple frames, each frame being viewable in a selectable order by said receiver even when less than all of said plural blocks have been received, receipt of subsequent blocks by the receiver being used to cumulatively improve the quality of the digital data viewed by the receiver.

106. A system according to claim 105 and wherein said model based data representations comprise VRML representations.

107. A system according to claim 105 and wherein said model based data representations comprise CAD-CAM representations.

108. A system according to claim 105 and wherein said image processor is operative to render only views which are selected by a user.

109. A method for transmitting model based data representations of three dimensional images over plural transmission links having limited bandwidth, said system comprising:

storing model based data representations of three dimensional images;

rendering views of said model based data representations into raster bitmap format;

receiving said digital data in said raster bitmap format over a one of said plural transmission links having limited bandwidth; and

transmit said digital data in said raster bitmap format to said receiver over a transmission link having a limited bandwidth in plural blocks which are sequentially transmitted at a rate determined by said limited bandwidth, each block being an incomplete collection of data which includes parts of multiple frames, each frame being viewable in a selectable order by said receiver even when less than all of said plural blocks have been received, receipt of subsequent blocks by the receiver being used to cumulatively improve the quality of the digital data viewed by the receiver.

110. A method according to claim 109 and wherein said model based data representations comprise VRML representations.

111. A method according to claim 109 and wherein said model based data representations comprise CAD-CAM representations.

112. A method according to claim 109 and wherein said image processor is operative to render only views which are selected by a user.

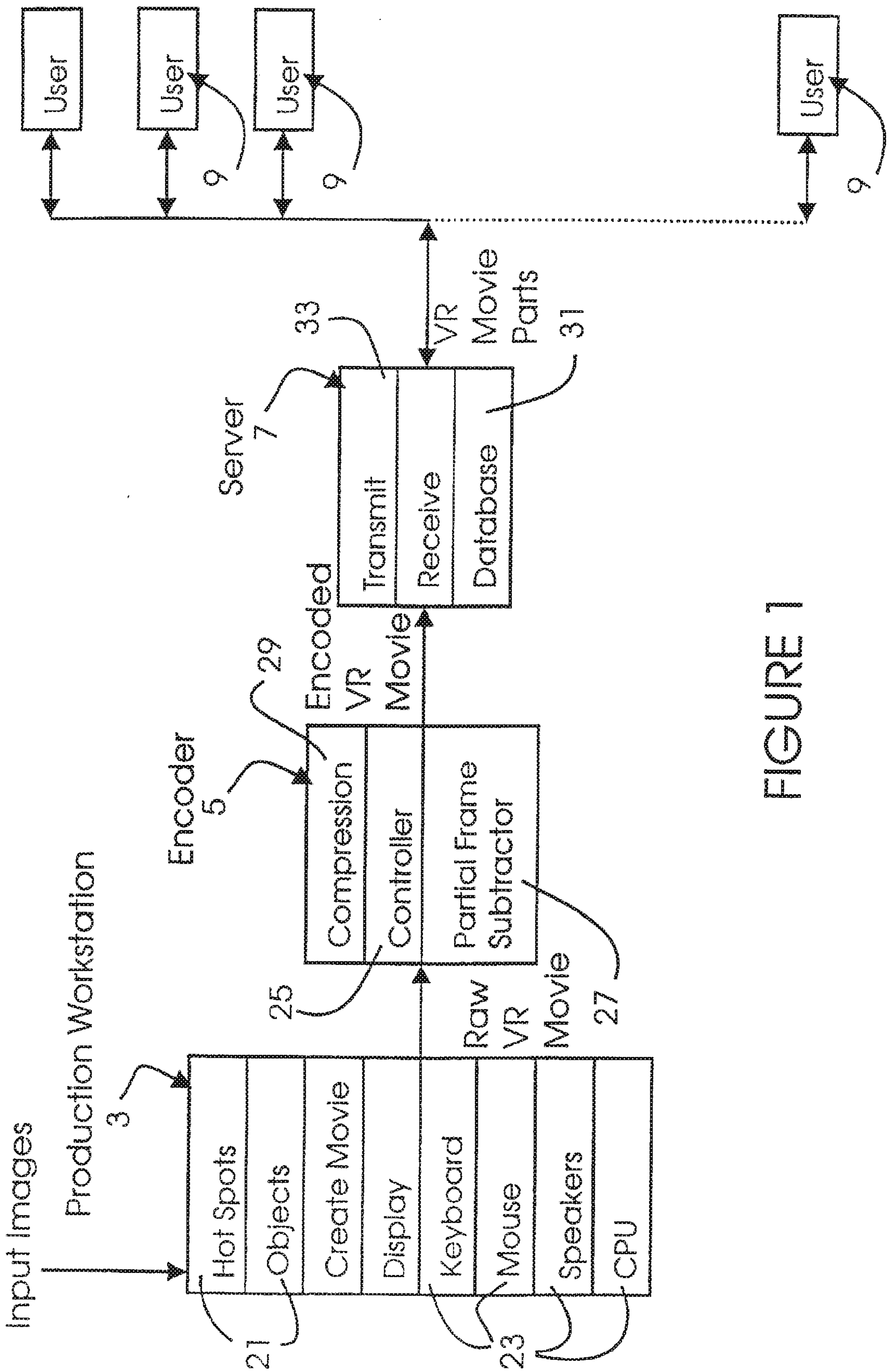
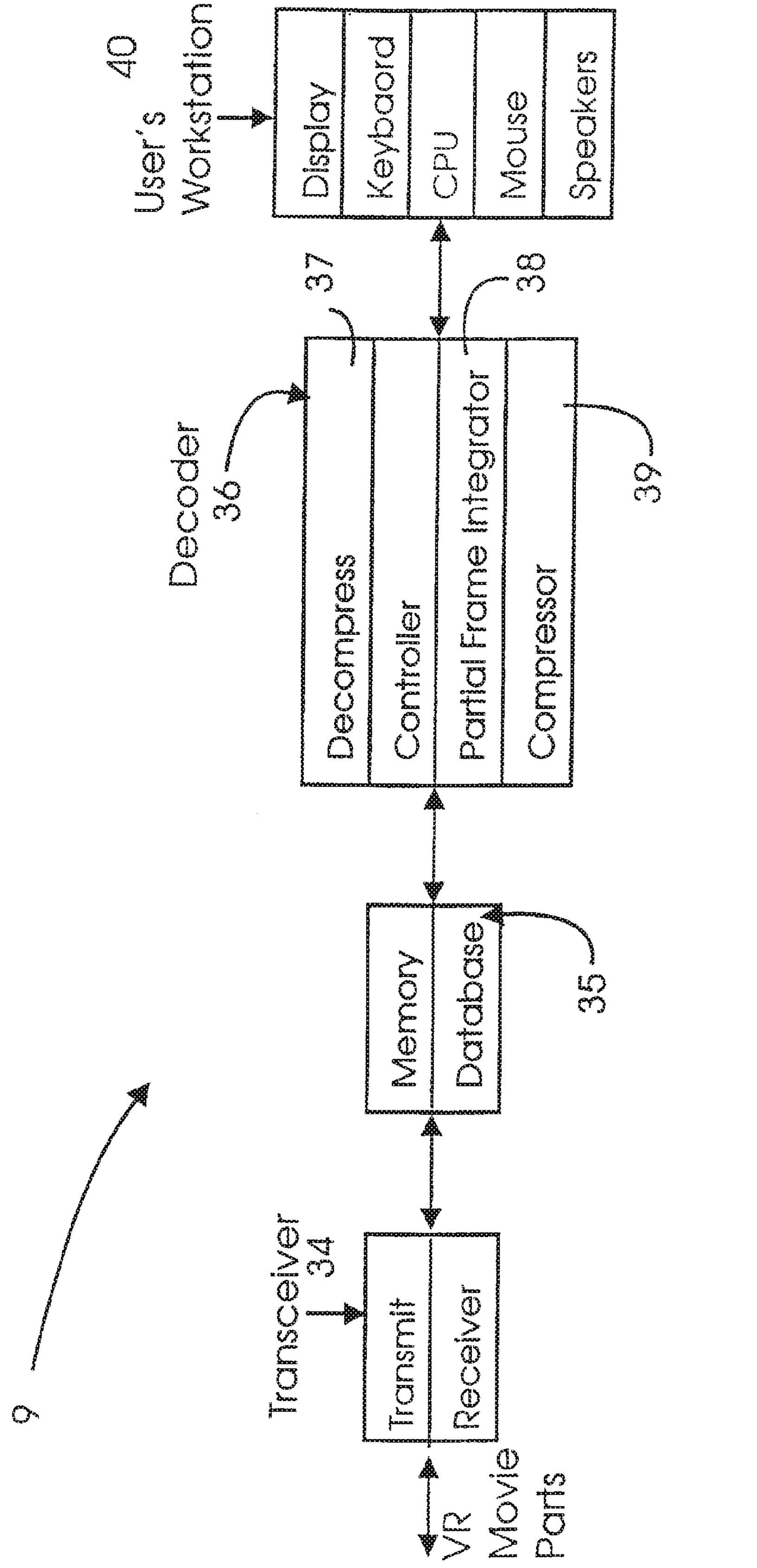

Input Images
Production Workstation
3
Hot Spots
Objects
Create Movie
Display
Keyboard
Mouse
Speakers
CPU
21
23
Raw VR Movie
Encoder
5
Compression
Controller
Partial Frame Subtractor
25
27
29
Encoded VR Movie
Server
7
Transmit
Receive
Database
33
31
VR Movie Parts
User
User
User
User
9
9
9

FIGURE 1



9
VR Movie Parts
Transceiver 34
Transmit
Receiver
Memory
Database
35
Decoder 36
Decompress
Controller
Partial Frame Integrator
Compressor
37
38
39
User's Workstation 40
Display
Keybaord
CPU
Mouse
Speakers

FIGURE 2

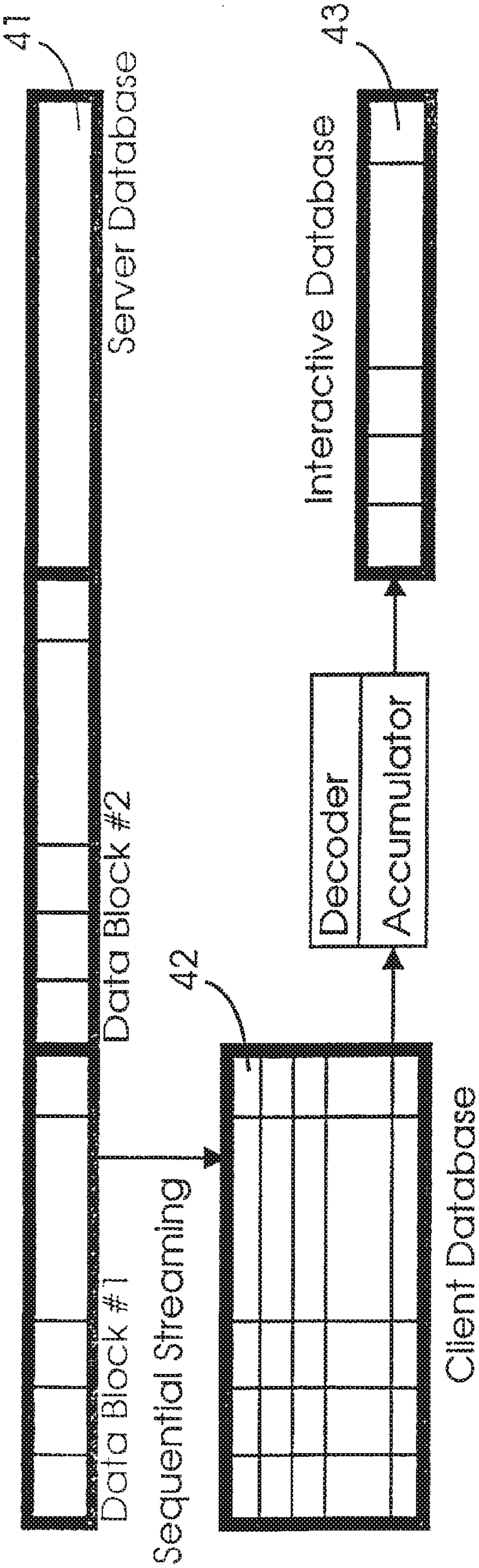
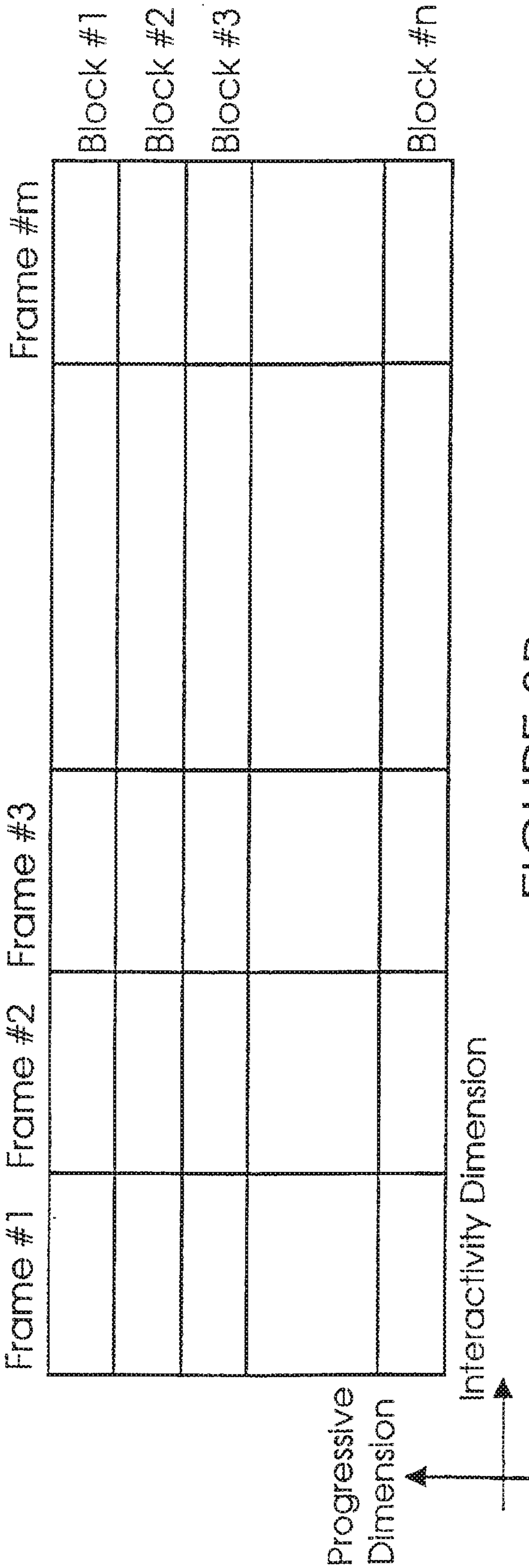

41
Server Database
Data Block #2
Data Block #1
Sequential Streaming
42
Client Database
Decoder
Accumulator
43
Interactive Database

FIGURE 3A



Frame #1
Frame #2
Frame #3
Frame #m
Block #1
Block #2
Block #3
Block #n
Progressive Dimension
Interactivity Dimension

FIGURE 3B

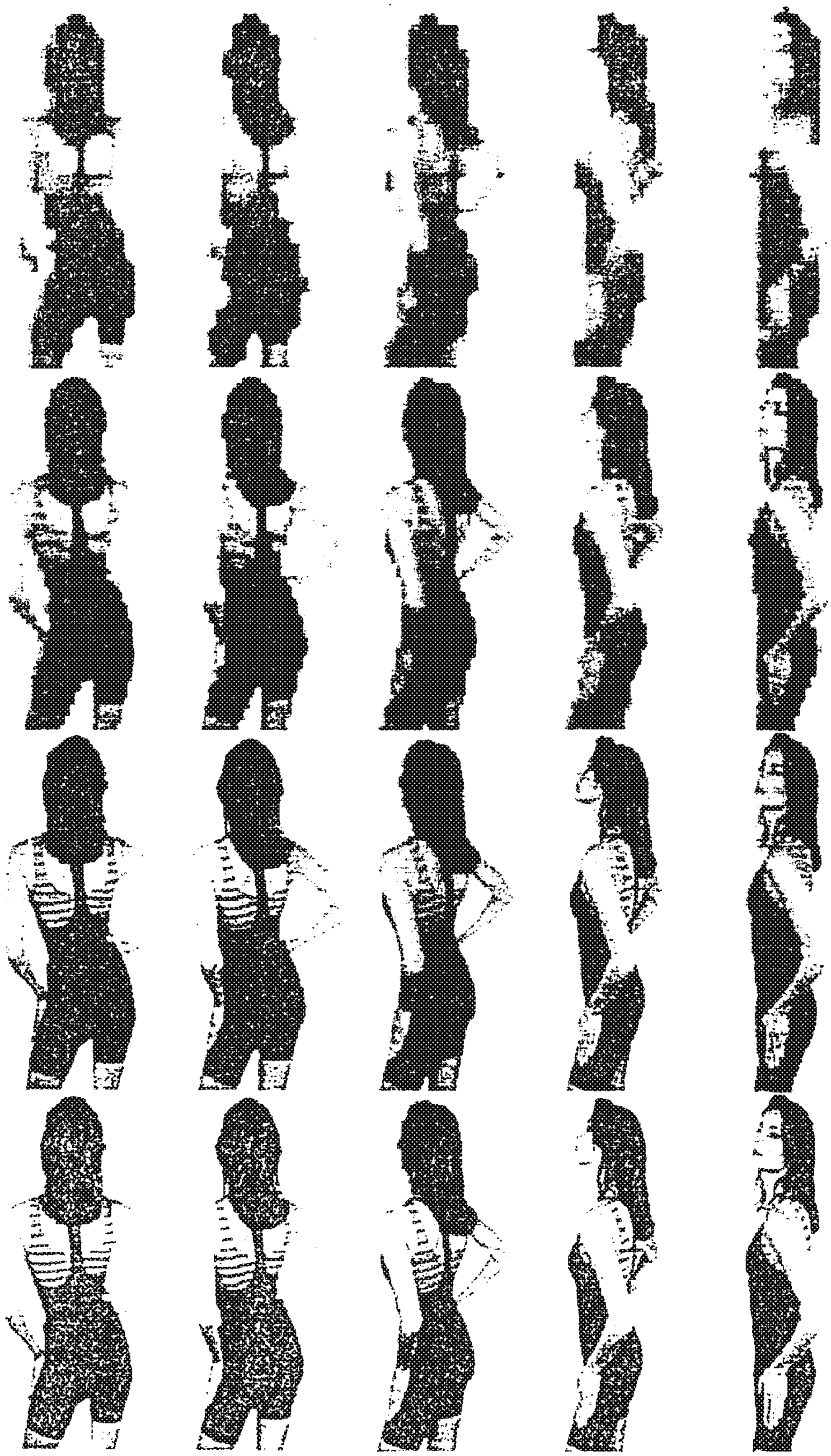


FIGURE 4

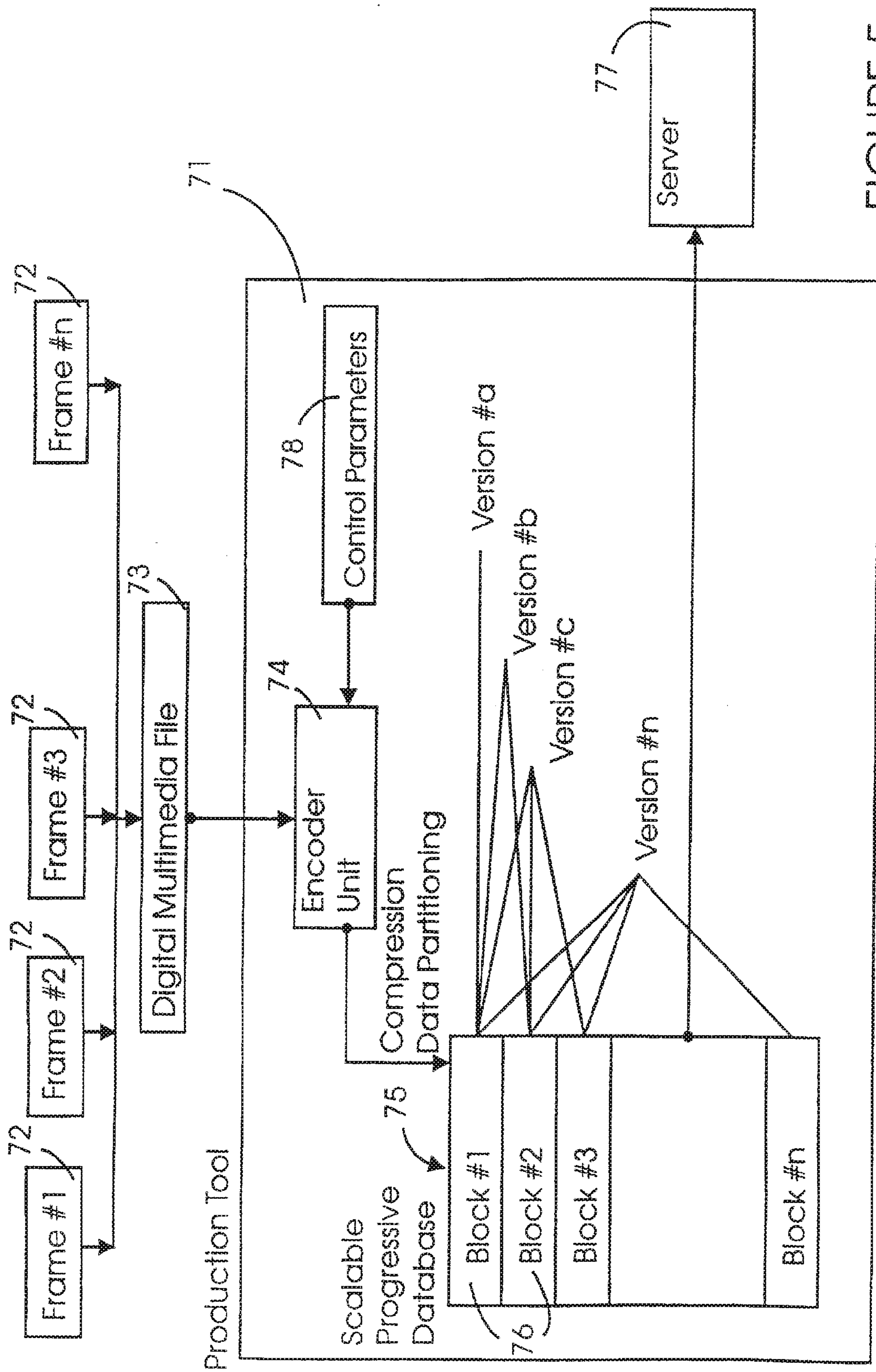
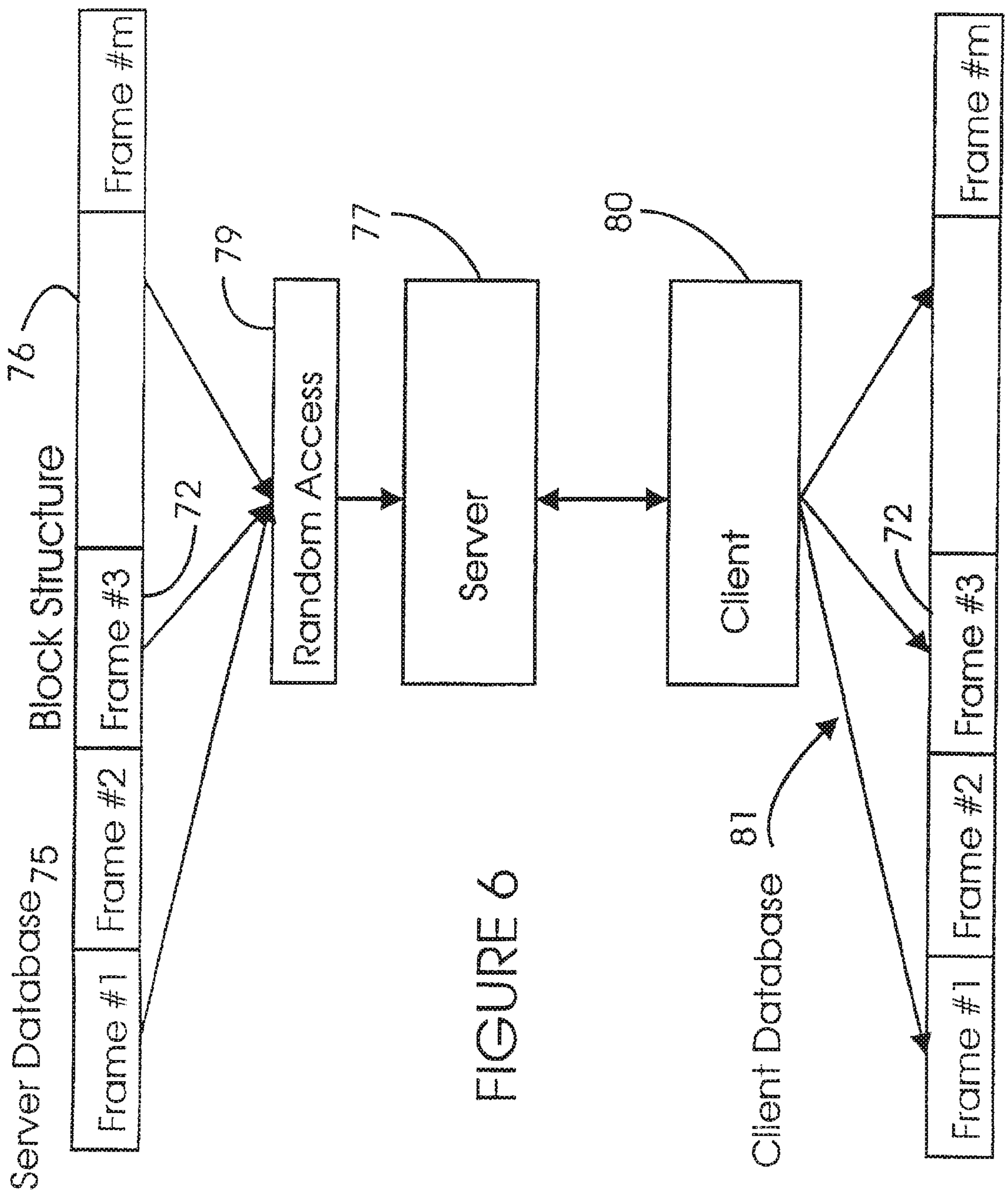

Frame #1
Frame #2
Frame #3
Frame #n
72
Digital Multimedia File
73
Production Tool
71
Encoder Unit
74
Control Parameters
78
Compression Data Partitioning
Scalable Progressive Database
75
Block #1
Block #2
Block #3
Block #n
76
Version #a
Version #b
Version #c
Version #n
Server
77

FIGURE 5



Server Database
75
Frame #1
Frame #2
Frame #3
Frame #m
Block Structure
76
72
Random Access
79
Server
77
Client
80
Client Database
81
Frame #1
Frame #2
Frame #3
Frame #m
72

FIGURE 6

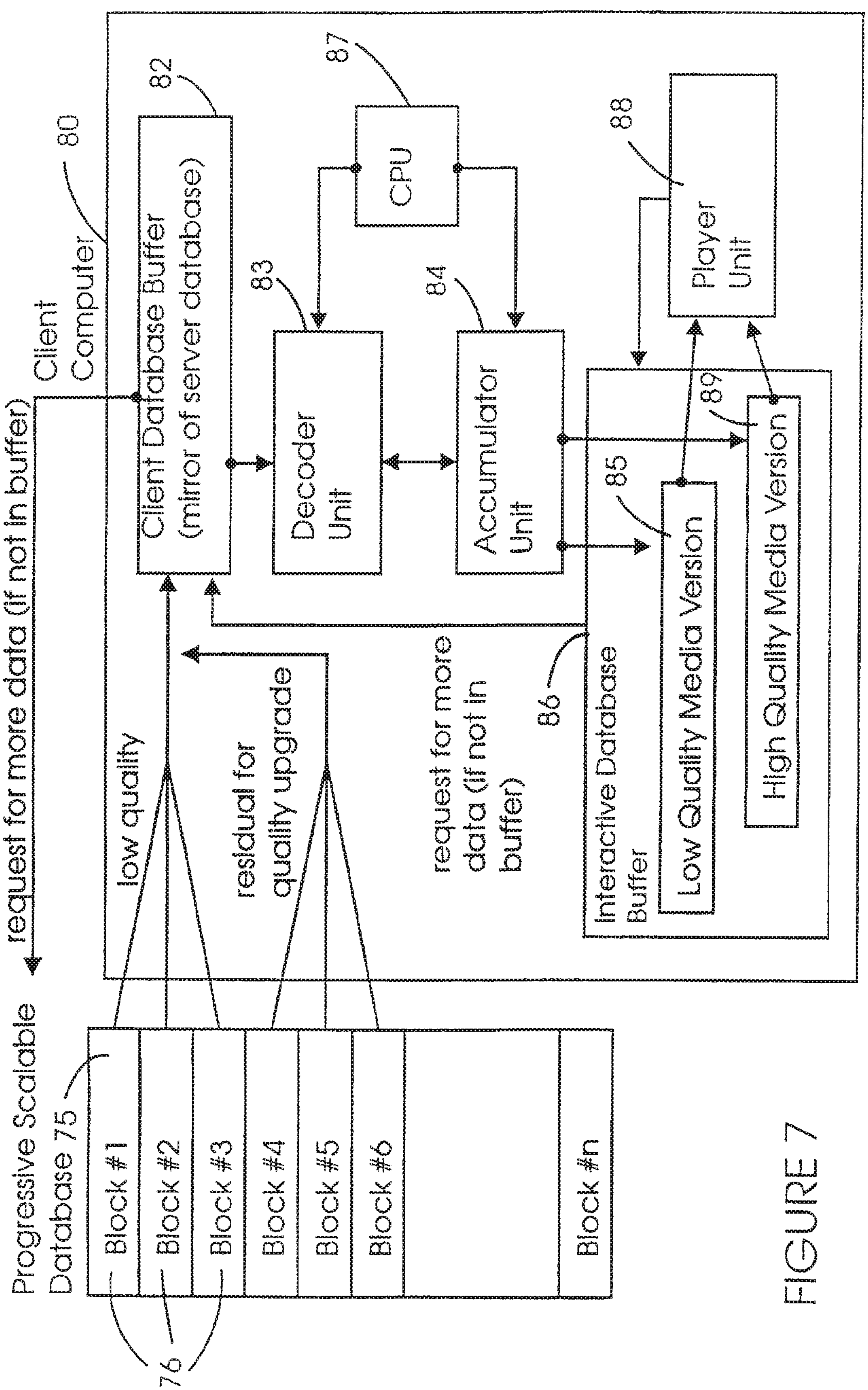
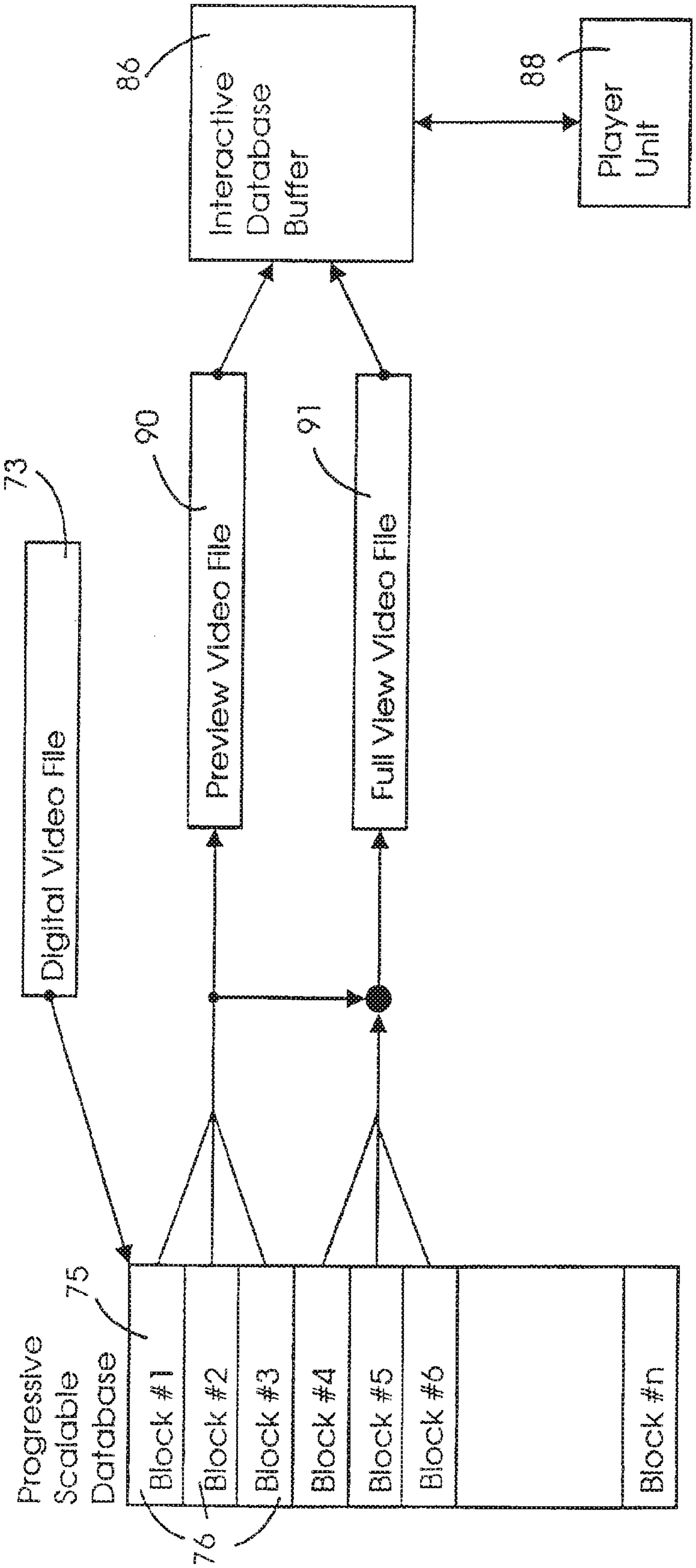

Client Computer
80
request for more data (if not in buffer)
Client Database Buffer (mirror of server database)
82
CPU
87
Decoder Unit
83
Accumulator Unit
84
Player Unit
88
Interactive Database Buffer
86
Low Quality Media Version
85
High Quality Media Version
89
low quality
residual for quality upgrade
request for more data (if not in buffer)
Progressive Scalable Database 75
Block #1
Block #2
Block #3
Block #4
Block #5
Block #6
Block #n
76

FIGURE 7

9/18



86
Interactive Database Buffer
88
Player Unit
73
Digital Video File
90
Preview Video File
91
Full View Video File
75
Progressive Scalable Database
Block #1
Block #2
Block #3
Block #4
Block #5
Block #6
Block #n
76

FIGURE 8

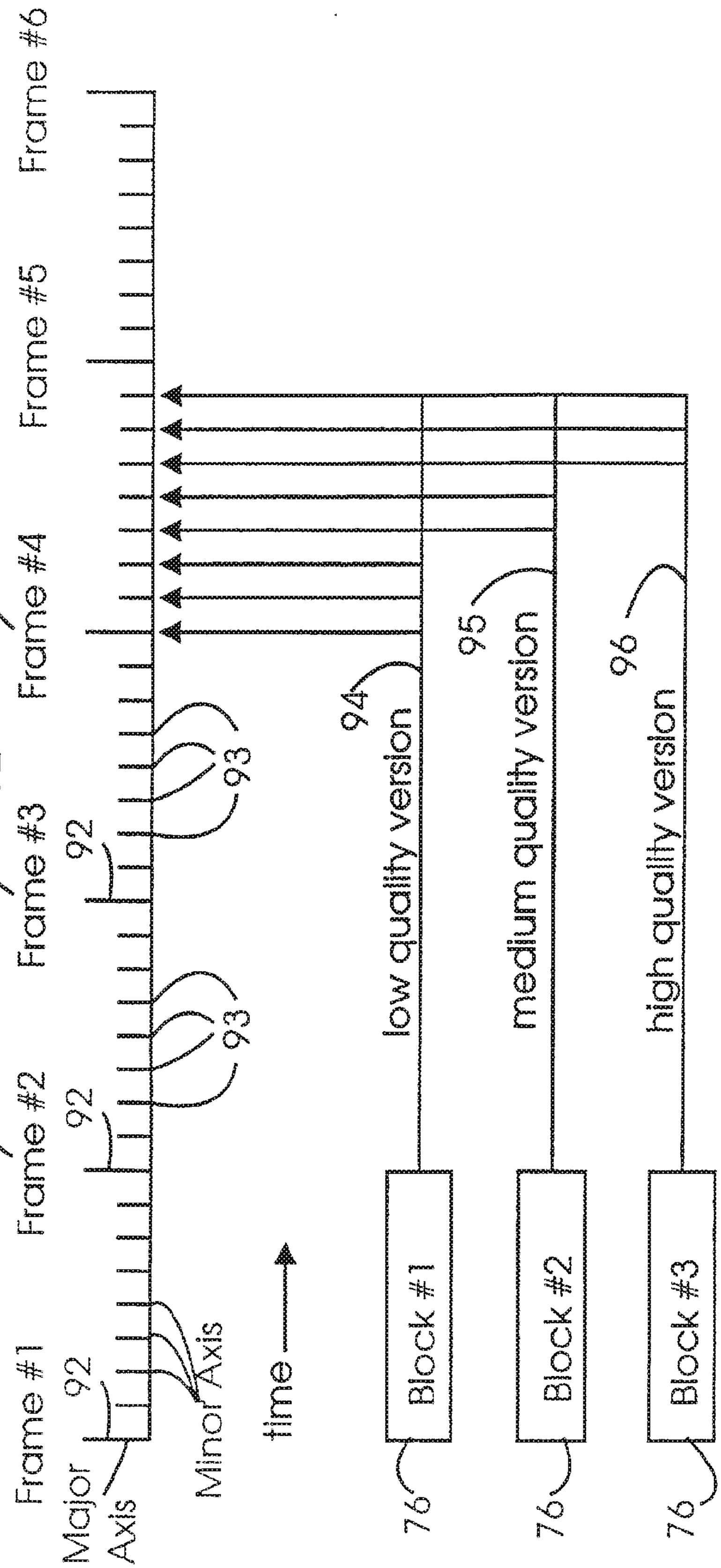

Frame #1
Frame #2
Frame #3
Frame #4
Frame #5
Frame #6
72
72
72
92
92
92
93
93
Major Axis
Minor Axis
time
Block #1
Block #2
Block #3
76
76
76
94
low quality version
95
medium quality version
96
high quality version

FIGURE 9

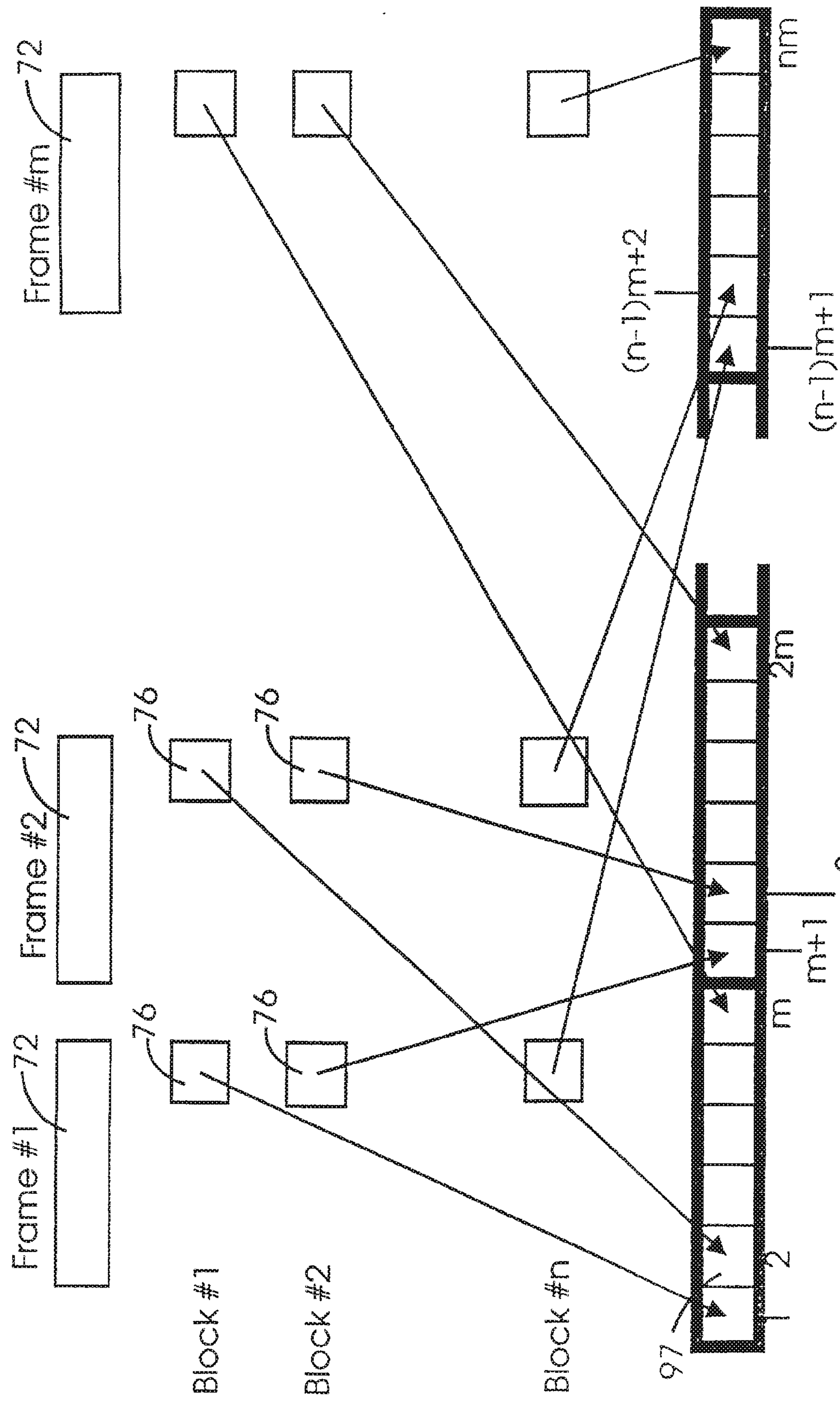

Frame #1
Frame #2
Frame #m
72
76
Block #1
Block #2
Block #n
97
1
2
m
m+1
m+2
2m
(n-1)m+1
(n-1)m+2
nm

FIGURE 10

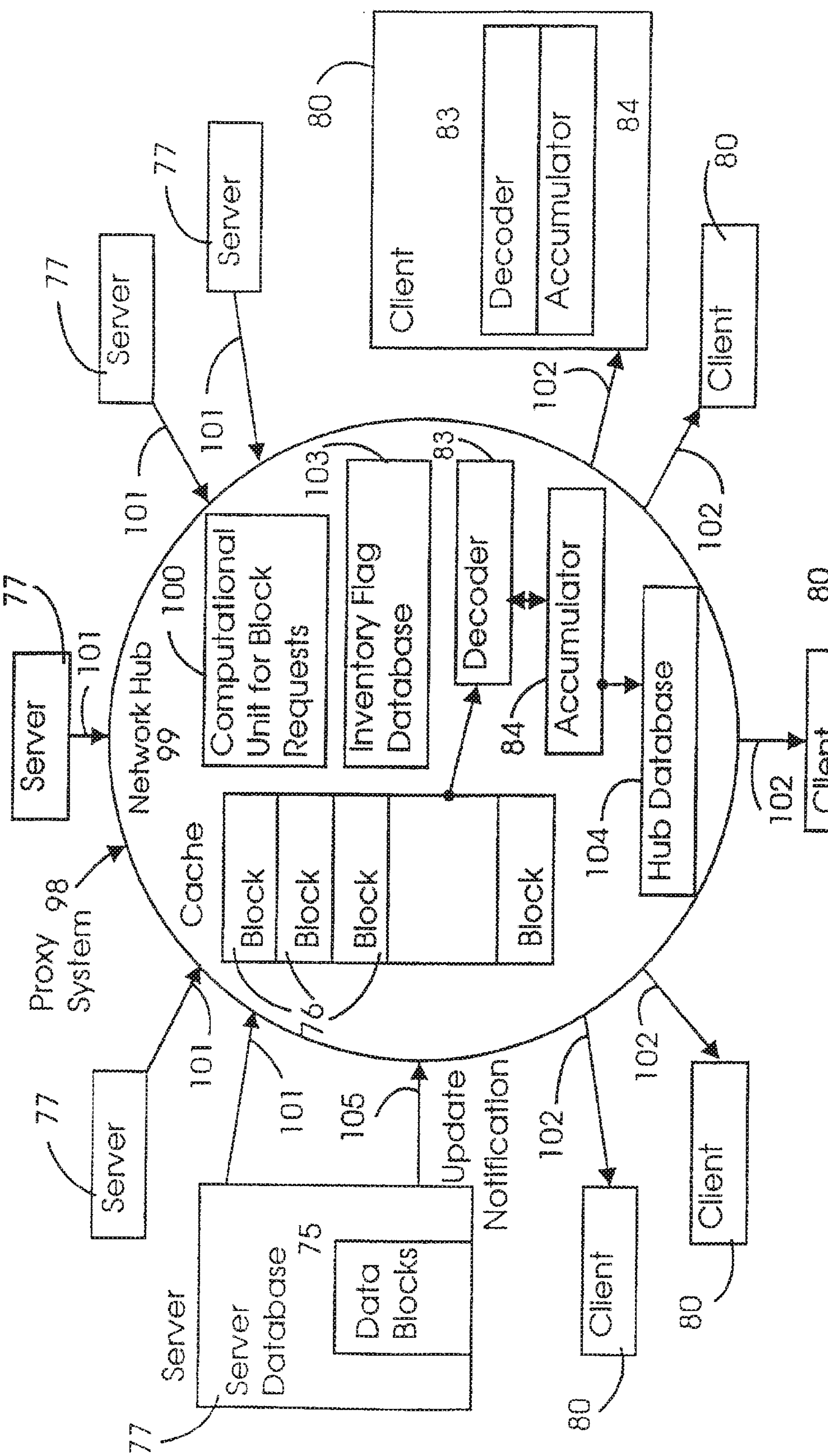

Server
77
Server Database
Data Blocks
75
Update Notification
105
Server
101
Proxy System
98
Network Hub
99
Computational Unit for Block Requests
100
Inventory Flag Database
103
Decoder
83
Accumulator
84
Hub Database
104
Cache
Block
76
Client
80
102
Decoder
Accumulator

FIGURE 11

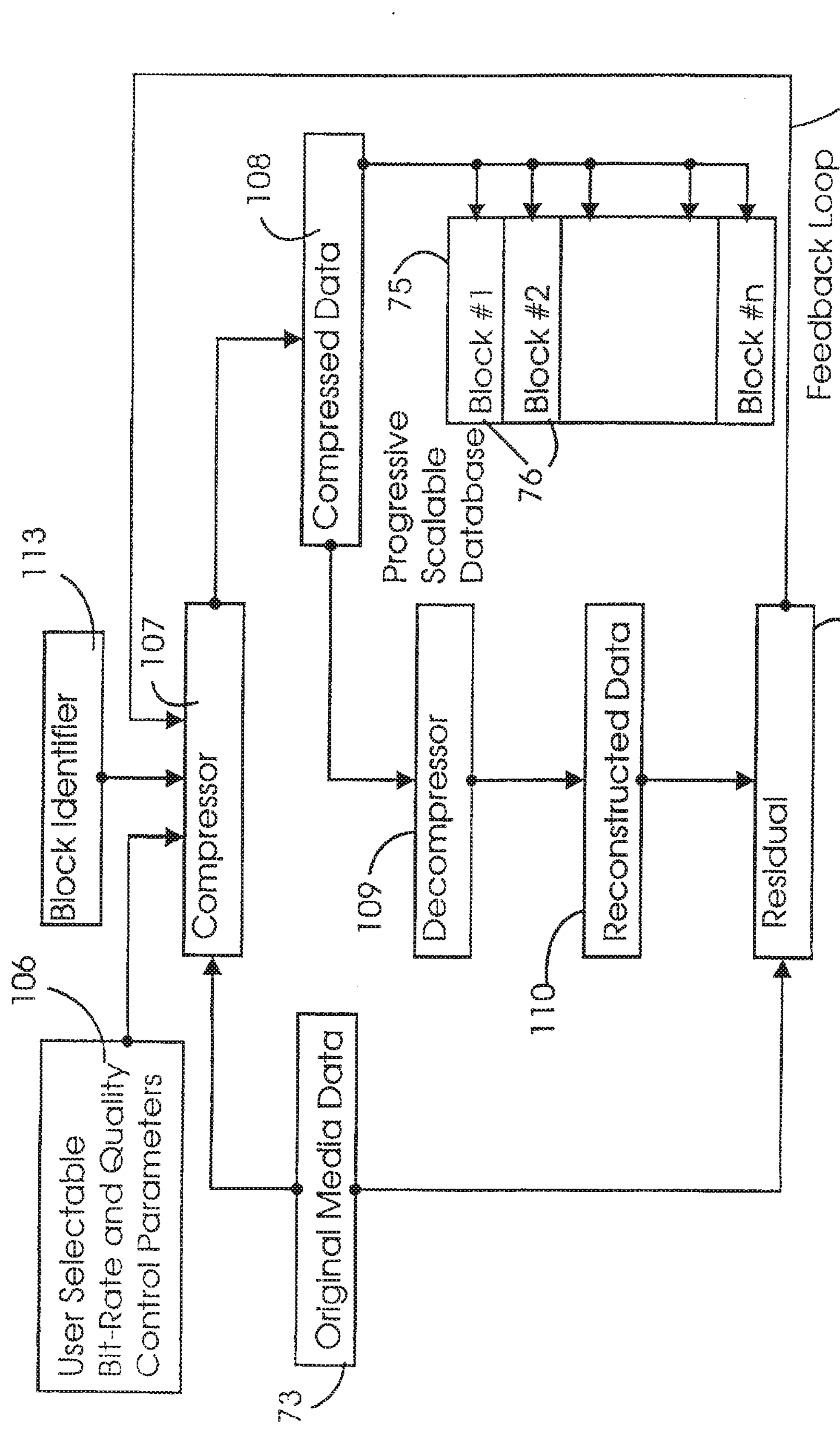

User Selectable Bit-Rate and Quality Control Parameters
106
Block Identifier
113
Compressor
107
Original Media Data
73
Compressed Data
108
Decompressor
109
Reconstructed Data
110
Residual
111
Progressive Scalable Database
Block #1
Block #2
Block #n
75
76
Feedback Loop
112

FIGURE 12

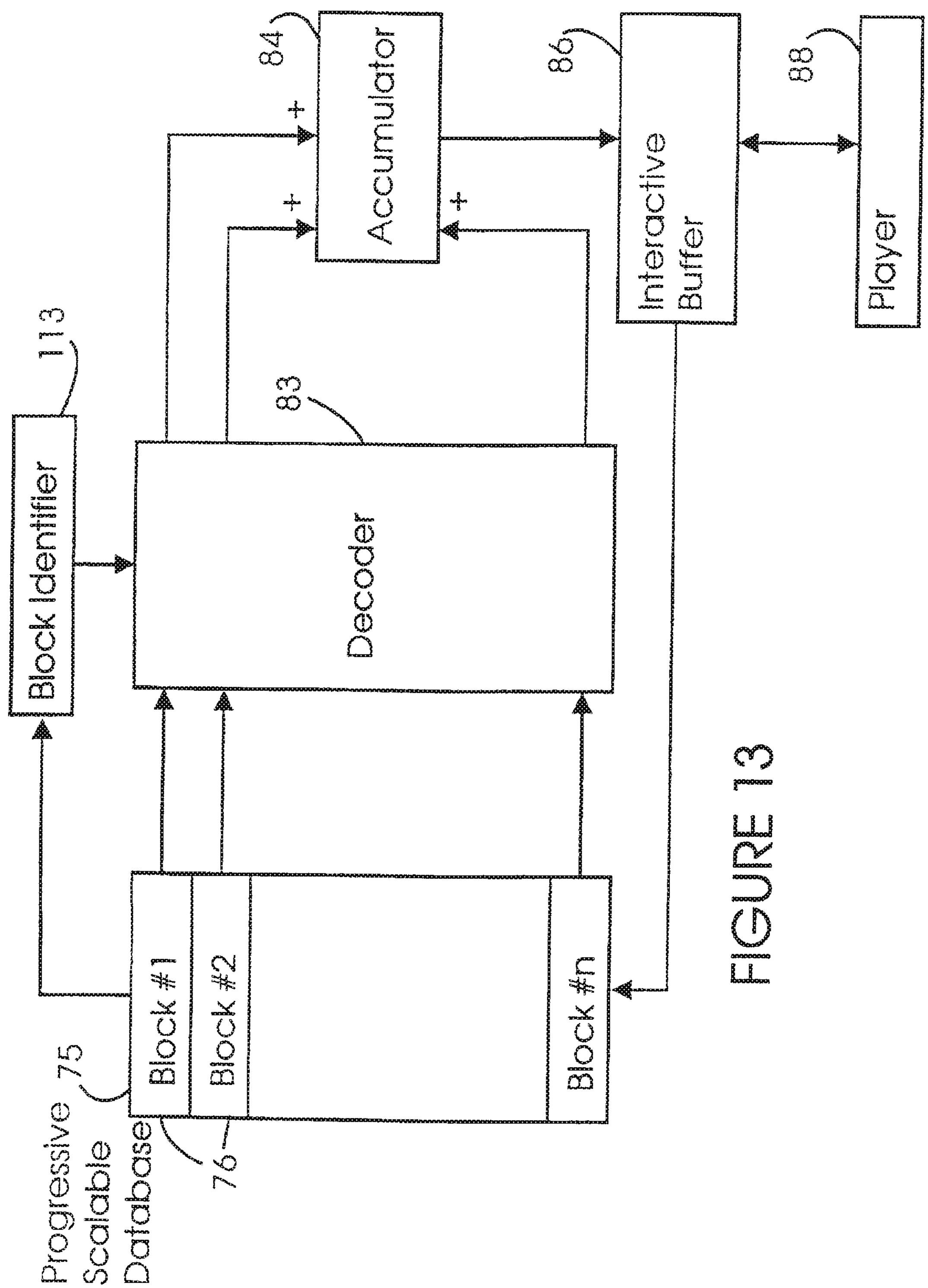

Progressive
Scalable
Database
75
76
Block #1
Block #2
Block #n
Block Identifier
113
Decoder
83
Accumulator
84
+
+
+
Interactive
Buffer
86
Player
88

FIGURE 13

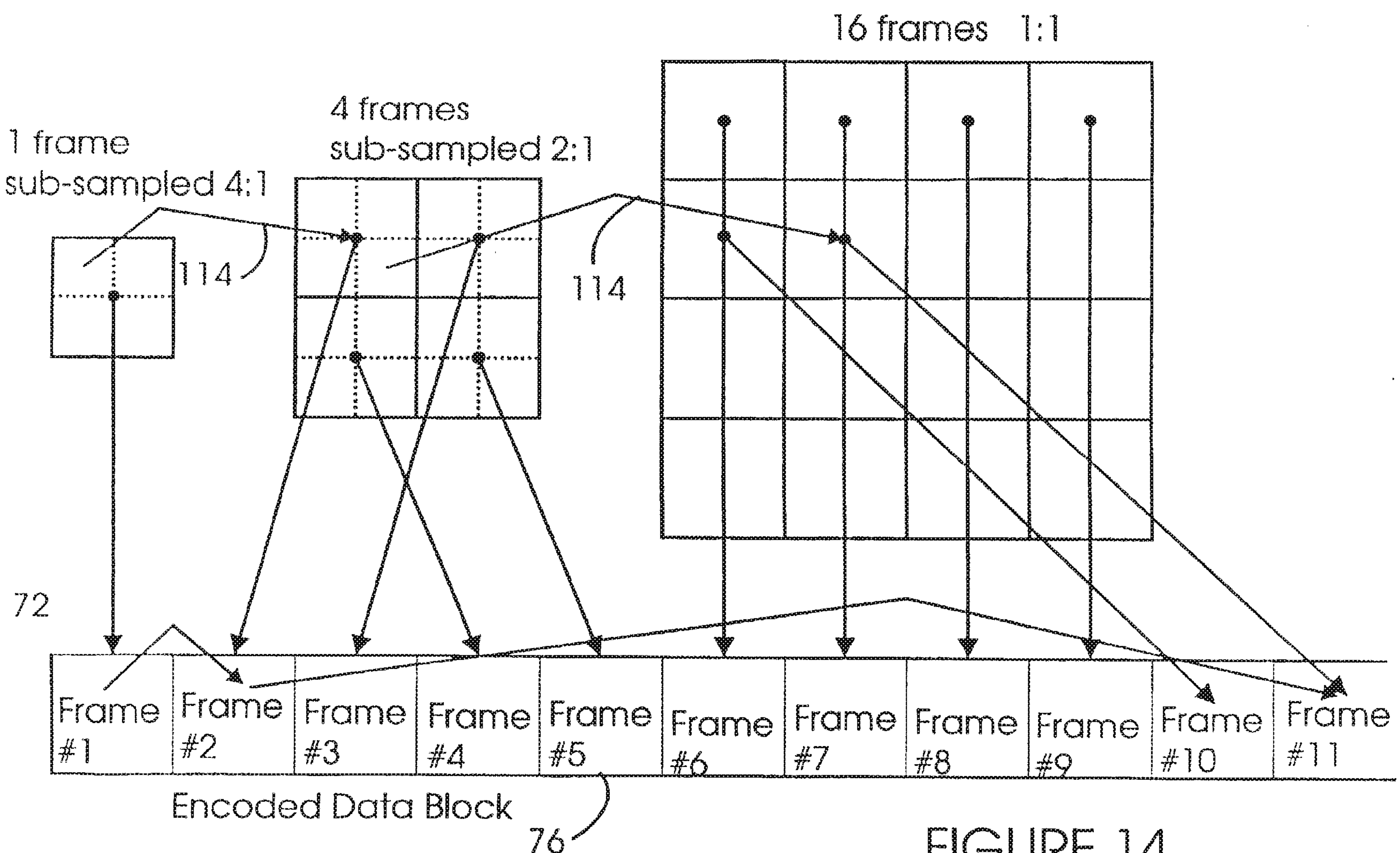

1 frame
sub-sampled 4:1
4 frames
sub-sampled 2:1
16 frames 1:1
114
114
72
Frame #1
Frame #2
Frame #3
Frame #4
Frame #5
Frame #6
Frame #7
Frame #8
Frame #9
Frame #10
Frame #11
Encoded Data Block
76

FIGURE 14

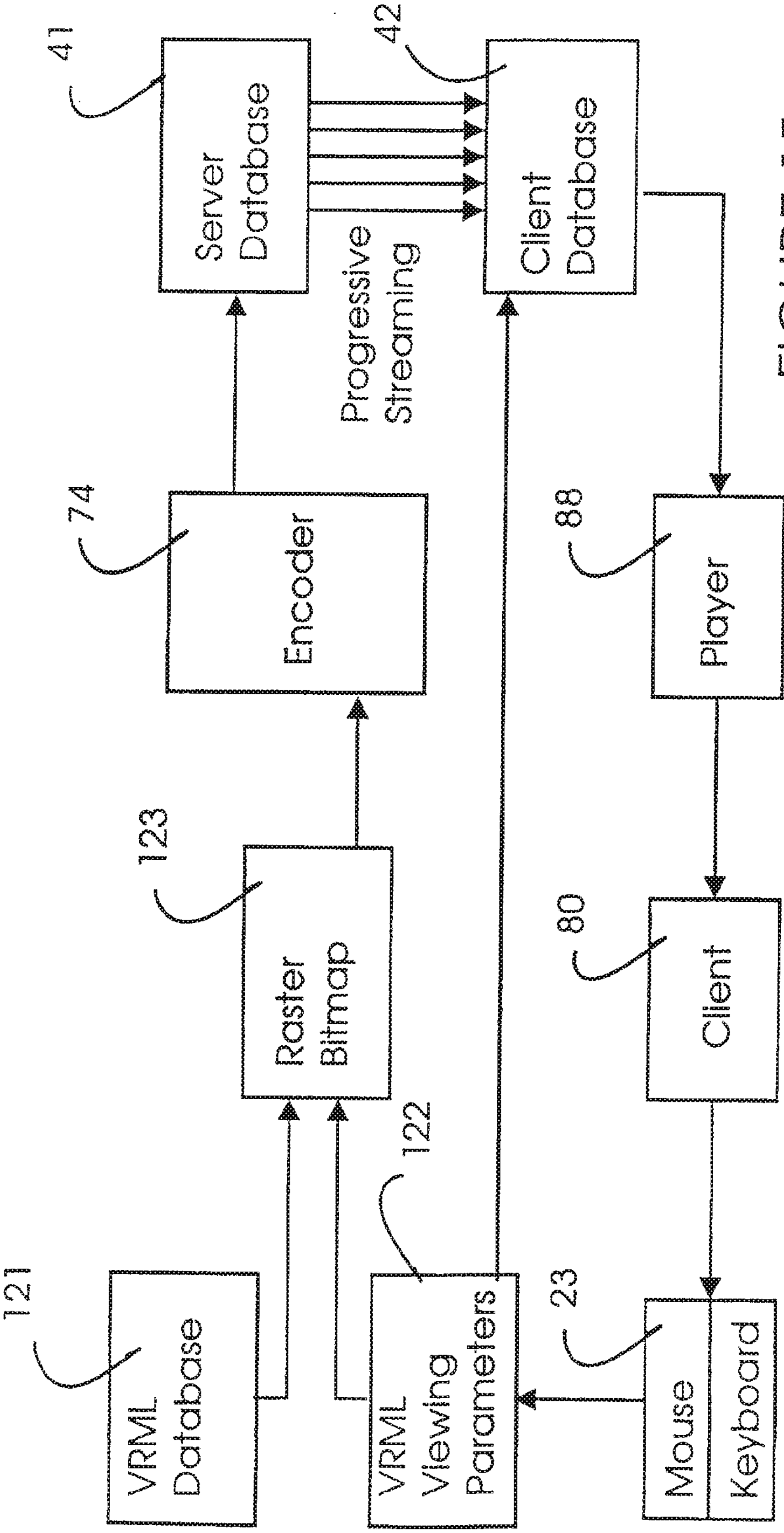

41
Server Database
42
Client Database
Progressive Streaming
74
Encoder
88
Player
123
Raster Bitmap
80
Client
121
VRML Database
122
VRML Viewing Parameters
23
Mouse
Keyboard

FIGURE 15

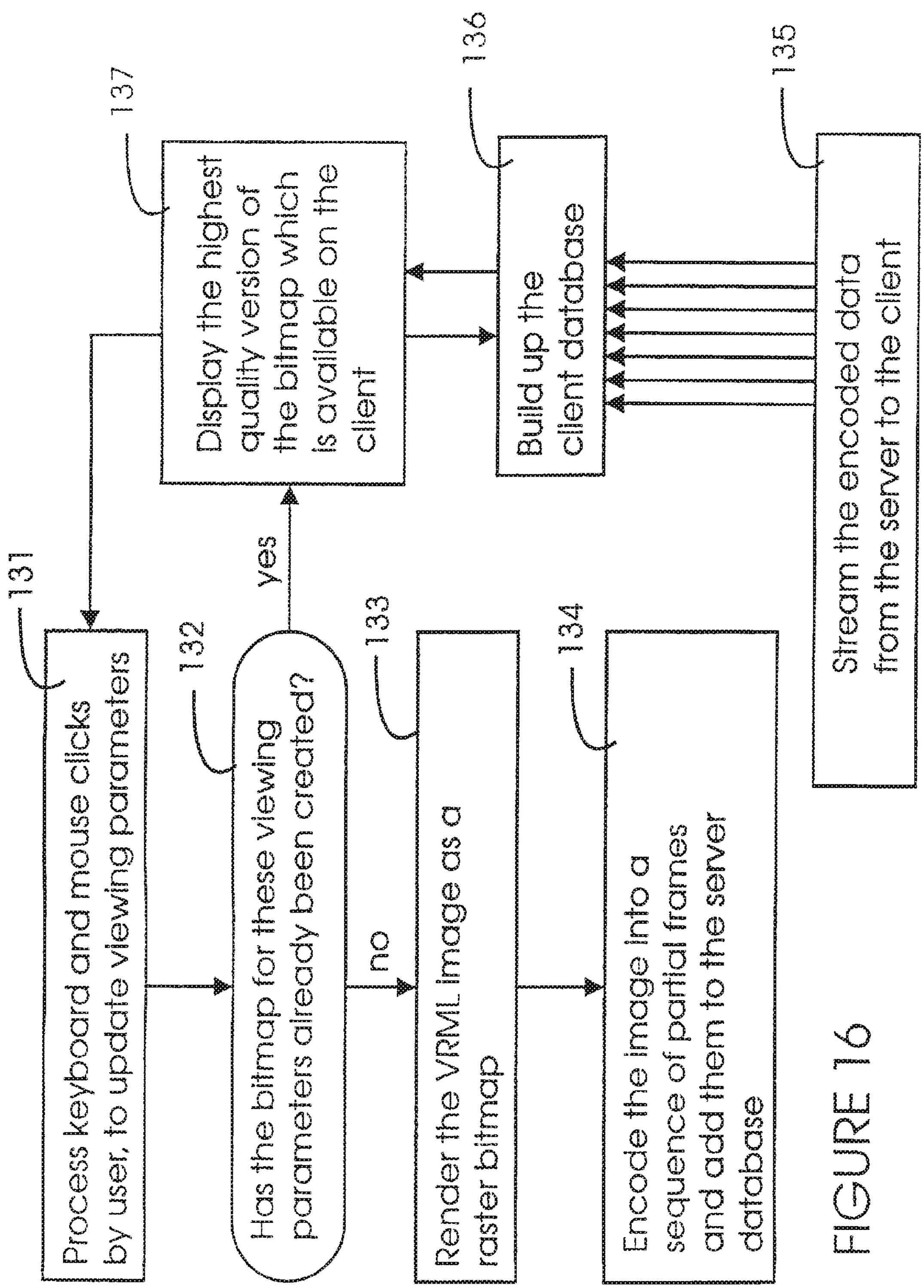

Process keyboard and mouse clicks by user, to update viewing parameters
131
Has the bitmap for these viewing parameters already been created?
132
yes
no
Render the VRML image as a raster bitmap
133
Encode the image into a sequence of partial frames and add them to the server database
134
Display the highest quality version of the bitmap which is available on the client
137
Build up the client database
136
Stream the encoded data from the server to the client
135

FIGURE 16

FIGURE 17

INTERNATIONAL SEARCH REPORT

International application No.
PCT/IL97/00055

A. CLASSIFICATION OF SUBJECT MATTER

IPC(6) :H04N 7/14
US CL :395/200.13; 348/7; 348/6; 455/4.2
According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

U.S. : 395/200.13; 348/7; 348/6; 455/4.2

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	US, 5,414,455 A (HOOPER ET AL.) 09 May 1995, whole reference.	1-112
A,P	US, 5,528,281 A (GRADY ET AL.) 18 June 1996, col. 4-6.	1-112
A,P	US, 5,561,791 A (MENDELSON ET AL.) 01 October 1996, the whole reference.	1-112
A	US, 5,421,031 A (DE BEY) 30 May 1995, see col. 6-10.	1-112
A,P	US, 5,512,934 A (KOCHANSKI) 30 April 1996, col. 3-6.	1-112
A,P	US, 5,544,313 A (SHACHNAI ET AL.) 06 August 1996, the whole reference.	1-112

[X] Further documents are listed in the continuation of Box C. [ ] See patent family annex.

* Special categories of cited documents:

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
06 MAY 1997	26 JUN 1997

Name and mailing address of the ISA/US	Authorized officer
Commissioner of Patents and Trademarks Box PCT Washington, D.C. 20231	ZARNI MAUNG
Facsimile No. (703) 305-3230	Telephone No. (703) 305-3900

Form PCT/ISA/210 (second sheet)(July 1992)*

INTERNATIONAL SEARCH REPORT

International application No.
PCT/IL97/00055

C (Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	US, 5,487,167 A (DINALLO ET AL.) 23 January 1996, the whole reference.	1-112

Form PCT/ISA/210 (continuation of second sheet)(July 1992)*

PCT

WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 6 : **G06F 3/00, 3/14, 13/00, 13/42**	A1	(11) International Publication Number: **WO 97/41504** (43) International Publication Date: 6 November 1997 (06.11.97)

(21) International Application Number: PCT/US97/06982

(22) International Filing Date: 24 April 1997 (24.04.97)

(30) Priority Data:
08/638,350 26 April 1996 (26.04.96) US

(71) Applicant *(for all designated States except US)*: ELOQUENT, INC. [US/US]; Suite 200, 1710 S. Amphlett Boulevard, San Mateo, CA 94402-2703 (US).

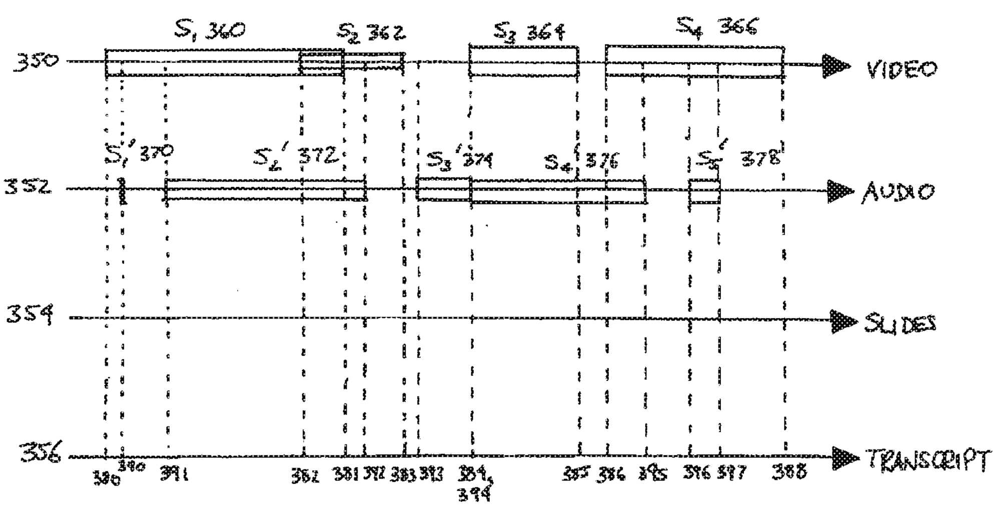
(72) Inventors; and
(75) Inventors/Applicants *(for US only)*: REID, Clifford, A. [US/US]; Apartment 2-710, 2 Townsend Street, San Francisco, CA 94107 (US). GLAZER, David [US/US]; 263 Glenwood Avenue, Woodside, CA 94062 (US).

(74) Agents: BARR, Robert et al.; Weil, Gotshal & Manges L.L.P., Suite 280, 2882 Sand Hill Road, Menlo Park, CA 94025 (US).

(81) Designated States: AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, CA, CH, CN, CU, CZ, DE, DK, EE, ES, FI, GB, GE, HU, IL, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MD, MG, MK, MN, MW, MX, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, TJ, TM, TR, TT, UA, UG, US, UZ, VN, ARIPO patent (GH, KE, LS, MW, SD, SZ, UG), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, ML, MR, NE, SN, TD, TG).

Published
With international search report.
Before the expiration of the time limit for amending the claims and to be republished in the event of the receipt of amendments.

(54) Title: A METHOD AND SYSTEM FOR SYNCHRONIZING AND NAVIGATING MULTIPLE STREAMS OF ISOCHRONOUS AND NON-ISOCHRONOUS DATA



(57) Abstract

A method and system for synchronizing multiple streams of isochronous and non-isochronous data (100) and navigating through the synchronized streams by reference to a common time base (210) and by means of a structured framework of conceptual events provides computer users with an effective means to interact with multimedia programs of speakers giving presentations (400). The multimedia programs consisting of synchronized video, audio, graphics, text, hypertext, and other data types can be stored on a server (130), and users can navigate and play them from a client CPU (110) over a non-isochronous network connection (150).

FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AL	Albania
AM	Armenia
AT	Austria
AU	Australia
AZ	Azerbaijan
BA	Bosnia and Herzegovina
BB	Barbados
BE	Belgium
BF	Burkina Faso
BG	Bulgaria
BJ	Benin
BR	Brazil
BY	Belarus
CA	Canada
CF	Central African Republic
CG	Congo
CH	Switzerland
CI	Côte d'Ivoire
CM	Cameroon
CN	China
CU	Cuba
CZ	Czech Republic
DE	Germany
DK	Denmark
EE	Estonia
ES	Spain
FI	Finland
FR	France
GA	Gabon
GB	United Kingdom
GE	Georgia
GH	Ghana
GN	Guinea
GR	Greece
HU	Hungary
IE	Ireland
IL	Israel
IS	Iceland
IT	Italy
JP	Japan
KE	Kenya
KG	Kyrgyzstan
KP	Democratic People's Republic of Korea
KR	Republic of Korea
KZ	Kazakstan
LC	Saint Lucia
LI	Liechtenstein
LK	Sri Lanka
LR	Liberia
LS	Lesotho
LT	Lithuania
LU	Luxembourg
LV	Latvia
MC	Monaco
MD	Republic of Moldova
MG	Madagascar
MK	The former Yugoslav Republic of Macedonia
ML	Mali
MN	Mongolia
MR	Mauritania
MW	Malawi
MX	Mexico
NE	Niger
NL	Netherlands
NO	Norway
NZ	New Zealand
PL	Poland
PT	Portugal
RO	Romania
RU	Russian Federation
SD	Sudan
SE	Sweden
SG	Singapore
SI	Slovenia
SK	Slovakia
SN	Senegal
SZ	Swaziland
TD	Chad
TG	Togo
TJ	Tajikistan
TM	Turkmenistan
TR	Turkey
TT	Trinidad and Tobago
UA	Ukraine
UG	Uganda
US	United States of America
UZ	Uzbekistan
VN	Viet Nam
YU	Yugoslavia
ZW	Zimbabwe

A METHOD AND SYSTEM FOR SYNCHRONIZING AND NAVIGATING MULTIPLE STREAMS OF ISOCHRONOUS AND NON-ISOCHRONOUS DATA

BACKGROUND OF THE INVENTION

1. Field of the Invention

The present invention generally relates to the production and delivery of video recordings of speakers giving presentations, and, more particularly, to the production and delivery of digital multimedia programs of speakers giving presentations. These digital multimedia programs consist of multiple synchronized streams of isochronous and non-isochronous data, including video, audio, graphics, text, hypertext, and other data types.

2. Description of the Prior Art

The recording of speakers giving presentations, at events such as professional conferences, business or government organizations' internal training seminars, or classes conducted by educational institutions, is a common practice. Such recordings provide access to the content of the presentation to individuals who were not able to attend the live event.

The most common form of such recordings is analog video taping. A video camera is used to record the event onto a video tape, which is subsequently duplicated to an analog medium suitable for distribution, most commonly a VHS tape, which can be viewed using a commercially-available VCR and television set. Such video tapes generally contain a video recording of the speaker and a synchronized audio recording of the speaker's words. They may also contain a video recording of any visual aids which the speaker used, such as text or graphics projected in a manner visible to the audience. Such video tapes may also be edited prior to duplication to include a textual transcript of the audio component recording, typically presented on the bottom of the video display as subtitles. Such subtitles are of particular use to the hearing impaired, and if translated into other languages, are of particular use to viewers who prefer to read along in a language other than the language used by the speaker.

Certain characteristics of such analog recordings of speakers giving presentations are unattractive to producers and to viewers. Analog tape players offer limited navigation facilities, generally limited to fast forward and rewind capabilities. In addition, analog tapes have the capacity to store only a few hours of video and audio, resulting in the need to duplicate and distribute a large number of tapes, leading to the accumulation of a large number of such tapes by viewers.

Advancements in computer technology have allowed analog recordings of speakers giving presentations to be converted to digital format, stored on a digital storage medium, such as a CD-ROM, and presented using a computer CPU and display, rather than a VCR and a television set. Such digital recordings generally include both isochronous and non-isochronous data. Isochronous data is data that is time ordered and must be presented at a particular rate. The isochronous data contained in such a digital recording generally includes video and audio. Non-isochronous data may or may not be time ordered, and need not be presented at a particular rate. Non-isochronous data contained in such a digital recording may include graphics, text, and hypertext.

The use of computers to play digital video recordings of speakers giving presentations provides navigational capabilities not available with analog video tapes. Computer-based manipulation of the digital data offers random access to any point in the speech, and if there is a text transcript, allows the users to search for words in the transcript to locate a particular segment of the speech.

Certain characteristics of state-of-the-art digital storage and presentation of recordings of speakers giving presentations are unattractive to producers and to viewers. There is no easy way to navigate directly to a particular section of a presentation that discusses a topic of particular interest to the user. In addition, there is no easy way to associate a table of contents with a presentation, and navigate directly to section of the presentation associated with each entry in the table of contents. Finally, like analog tapes, CD-ROMs can store only a view hours of digital video and audio, resulting in the need to duplicate and distribute a large number of CD-ROMs, leading to the accumulation of a large number of such CD-ROMs by viewers.

SUMMARY OF THE INVENTION

It is therefore an object of the present invention to provide a mechanism for synchronizing multiple streams of isochronous and non-isochronous digital data in a manner that supports navigating by means of a structured framework of conceptual events.

It is another object of the invention to provide a mechanism for navigating through any stream using the navigational approach most appropriate to the structure and content of that stream.

It is another object of the invention to automatically position each of the streams at the position corresponding to the selected position in the navigated stream, and simultaneously display some or all of the streams at that position.

It is another object of the invention to provide for the delivery of programs made up of multiple streams of synchronized isochronous and non-isochronous digital data across non-isochronous network connections.

In order to accomplish these and other objects of the invention, a method and system for manipulating multiple streams of isochronous and non-isochronous digital data is provided, including synchronizing multiple streams of isochronous and non-isochronous data by reference to a common time base, supporting navigation through each stream in the manner most appropriate to that stream, defining a framework of conceptual events and allowing a user to navigate though the streams using this structured framework, identifying the position in each stream corresponding to the position selected in the navigated stream, and simultaneously displaying to the user some or all of the streams at the position corresponding to the position selected in the navigated stream. Further, a method and system of efficiently supporting sequential and random access into streams of isochronous and non-isochronous data across non-isochronous networks is provided, including reading the isochronous and non-isochronous data from the storage medium into memory of the server CPU, transmitting the data from the memory of the server CPU to the memory of the client CPU, and caching the different types of data in the memory of the client CPU in a manner that ensures continuous display of the isochronous data on the client CPU display device.

BRIEF DESCRIPTION OF THE DRAWINGS

The foregoing and other objectives, aspects, and advantages of the present invention will be better understood from the following detailed description of embodiments thereof with reference to the following drawings.

FIG. **1** is a schematic diagram of the organization of a data processing system incorporating an embodiment of the present invention.

FIGS. **2** and **3** are schematic diagrams of the organization of the data in an embodiment of the present invention.

FIG. **4** is a diagram showing how two different sets of "conceptual events" may be associated with the same presentation in an embodiment of the present invention.

FIGS. **5**, **6** and **9** are exemplary screens produced in accordance with an embodiment of the present invention.

FIGS. **7**, **8**, **10**, and **11** are flow charts indicating the operation of an embodiment of the present invention.

DETAILED DESCRIPTION OF AN EMBODIMENT OF THE INVENTION

Referring now to the drawings, and more particularly to FIG. **1**, there is shown, in schematic representation, a data processing system **100** incorporating the invention. Conventional elements of the system include a client central processing unit **110** which includes high-speed memory, a local storage device **112** such as a hard disk or CD-ROM, input devices such as keyboard **114** and pointing device

116 such as a mouse, and a visual data presentation device 118, such as a computer display screen, capable of presenting visual data perceptible to the senses of a user, and an audio data presentation device 120, such as speakers or headphones, capable of presenting audio data to the senses of a user. Other conventional elements of the system include a server central processing unit 130 which includes high-speed memory, a local storage device 132 such as a hard disk or CD-ROM, input devices such as keyboard 134 and pointing device 136, and a visual data presentation device 138, and an audio data presentation device 140. The client CPU is connected to the server CPU by means of a network connection 150.

The invention includes three basic aspects: (1) synchronizing multiple streams of isochronous and non-isochronous data, (2) navigating through the synchronized streams of data by means of a structured framework of conceptual events, or by means of the navigational method most appropriate to each stream, and (3) delivering the multiple synchronized streams of isochronous and non-isochronous data over a non-isochronous network connecting the client CPU and the server CPU.

An exemplary form of the organization of the data embodied in the invention is shown in FIG. 2 and FIG. 3. Beginning with FIG. 2, the video/audio stream 200 is of a type known in the art capable of being played on a standard computer equipped with the appropriate video and audio subsystems, such as shown in FIG. 1. An example of such a video/audio stream is Microsoft Corporation's AVI™ format, which stands for "audio/video interleaved." AVI™ and other such video/audio formats consist of a series of digital images, each referred to as a "frame" of the video, and a series of samples that make up the digital audio. The frames are spaced equally in time, so that displaying consecutive frames on a display device at a sufficiently high and constant rate produces the sensation of continuous motion to the human perceptual system. The rate of displaying frames typically must exceed ten to fifteen frames per second to achieve the effect of continuous motion. The audio samples are synchronized with the video frames, so that the associated audio can be played in synchronization with the displayed video images. Both the digital images and digital audio samples may be compressed to reduce the amount of data that must be stored or transmitted.

A time base **210** associates a time code with each video frame. The time base is used to associate other data with each frame of video. The audio data, which for the purposes of this invention consists primarily of spoken words, is transcribed into a textual format, called the Transcript **220**. The transcript is synchronized to the audio data stream by assigning a time code to each word, producing the Time-Coded Transcript **225**. The time codes (shown in angle-brackets) preceding each word in the Time-Coded Transcript correspond to the time at which the speaker begins pronouncing that word. For example, the time code **230** of 22.51 s is associated with the word **235** "the." The Time-Coded Transcript may be created manually or by means of an automatic procedure. Manual time-coding requires a person to associate a time code with each word in the transcript. Automatic time coding, for example, uses a speech recognition system of a type well-known in the art to automatically assign a time code to each word as it is recognized and recorded. The current state of the art of speech recognition systems renders automatic time coding of the transcript less economical than manual time coding.

Referring now to FIG. **3**, the set **310** of Slides S1 **311**, S2 **312**, ... that the speaker used as part of the presentation may be stored in an electronic format of any of the types well-known in the art. Each slide may consist of graphics, text, and other data that can be rendered on a computer display. A Slide Index **315** assigns a time code to each Slide. For example, Slide S1 **311** would have a time code **316** of 0 s, S2 **312** having a time code **317** of 20.40 s, and so on. The time code corresponds to the time during the presentation at which the speaker caused the specified Slide to be presented. In one embodiment, all of the Slides are contained in the same disk file, and the Slide Index contains pointers to the locations of each Slide in the disk file. Alternatively, each Slide may be stored in a separate disk file, and the Slide Index contains pointers to the files containing the Slides.

An Outline **320** of the presentation is stored as a separate text data object. The Outline is a hierarchy of topics **321, 322**, .. that describe the organization of the presentation, analogous to the manner in which a table of contents describes the organization of a book. The outline may consist of an arbitrary number of entries, and an arbitrary number of levels in the hierarchy. An Outline Index **325** assigns a time code to each entry in the Outline. The time code corresponds to the time during the presentation at which the speaker begins discussing the topic represented by the entry in the Outline. For example, topic **321**, "Introduction" has entry name "01" and time code **326** of 0 s, topic **322** "The First Manned Flight" has entry name "02" and time code **327** of 20.50 s, "The Wright Brothers" **323** has entry name "021" (and hence is a subtopic of topic **322**) with time code **328** of 120.05 s, and so on. The Outline and the Outline Index may be created by means of a manual or an automatic procedure. Manual creation is accomplished by a person viewing the presentation, authoring the Outline, and assigning a time code to each element in the outline. Automatic creation may be accomplished by automatically constructing the outline consisting of the titles of each of the Slides, and associating with each entry on the Outline the time code of the corresponding Slide. Note that manual and automatic creation may produce different Outlines.

The set **330** of Hypertext Objects **331, 332**, ... relating to the subject of the presentation may be stored in an electronic formats of various types well-known in the art. Each Hypertext Object may consist of graphics, text, and other data that can be rendered on a computer display, or pointers to other software applications, as spreadsheets, word processors, and electronic mail systems, as well as more specialized applications such as proficiency testing applications or computer-based training applications.

A Hypertext Index table **335** is used to assign two time codes and a display location to each Hypertext Object. The first time code **336** corresponds to the earliest time during the presentation at which the Hypertext Object relates to the content of the presentation. The second time code **337** corresponds to the latest time during the presentation at which the Hypertext Object relates to the content of the presentation. The Object Name **338**, as the name suggests, denotes the Hypertext Object's name. The display location **339** denotes how the connection to the Hypertext Object, referred to as the Hypertext Link, is to be displayed on the computer screen. Hypertext Links may be displayed as highlighted words in the Transcript or the Slides, as buttons or menu items on the end-user interface, or in other visual presentation that may be selected by the user.

It may be appreciated by one of ordinary skill in the art that other data types may be synchronized to the common time base in a manner similar to the approaches used to synchronize the video/audio stream with the Transcript, the Slides, and the Hypertext Objects. Examples of such other data types include animations, series of computer screen images, and other specialty video streams.

An Outline represents an example of what is termed here a set of "conceptual events." A conceptual event is an association one makes with a segment of a data stream, having a beginning and end (though the beginning and end may be the points), that represents something of interest. These data segments delineating a set of conceptual events may overlap each other, and furthermore, need not cover the entire data stream. An Outline represents a set of conceptual events that does cover the entire data stream and, if arranged hierarchically, such as with sections and subsections, has sections covering subsections. In the Outline **320** of FIG. **3**, one has the sections 01:"Introduction" **321**, 02:"The First Manned

Flight" **322** , and so on, covering the entire presentation. The subsections 021:"The Wright Brothers" **324**, 022:"Failed Attempts" **324** and so on, represents another coverage of the same segment as 02:"The First Manned Flight" **322**. In accordance with the principles of the present invention, multiple Outlines, created manually or automatically, may be associated with the same presentation, thereby allowing different users with different purposes in viewing the presentation to use the Outline most suitable for their purposes. These Outlines have been described from the perspective of having been created beforehand, but there is no reason, under the principles of the present invention, for this to be so. It should be readily understood by one of ordinary skill in the art that a similar approach would allow a user to create a set of "bookmarks" that denote particular segments, or user-chosen "conceptual events" within presentations. The bookmarks allow the user, for example, to return quickly to interesting parts of the presentation, or to pick up at the previous stopping point.

With reference to FIG. **4**, the implementation of sets of conceptual events may be understood. There are time lines representing the various data streams, as for example, video **350**, audio **352**, slides **354** and transcript **356**. There are two sets of conceptual events or data segments of these time lines shown, S_1 **360**, S_2 **362**, S_3 **364**, S_4 **366**, ... and S'_1 **370**, S'_2 **372**, S'_3 **374**, S'_4 **376** , S'_5 **378**, ..., the first set indexed into the video **350** stream and second set indexed into the audio **352** stream. Thus, the first set S_1 **360**, S_2 **362**, S_3 **364**, etc., would respectively invoke time codes **380** and **381**, **382** and **383**, **384** and **385**, etc., not only for the video **350** data stream, but for the audio **352** , slides **354** and transcript **356** streams. Similarly, the second set S'_1 **370**, S'_2 **372**, S'_3 **384**, etc., would invoke respectively time codes **390** (a point), **391** and **392**, **393** and **394** (**394** shown collinear with **384**, whether by choice or accident), etc., respectively, not only on the audio **352** data stream, but on the video **350**, slides **354** and transcript **356** streams. Consider the following example of a presentation of ice skating performed to music, with voice-over commentaries and slides showing the relative standings of the ice skaters. A first Outline might list each skater and be broken down further into the individual moves of each skater's program. A second Outline might track the musical portion of the audio stream, following the music piece to piece, even movement to movement. Thus, one user might be interested in how a skater performed a particular move, while another user might wish to study how a particular passage of music inspired a skater to make a particular move. Note that there is no requirement that two sets of conceptual events track each other in any way, they represent two different ways of studying the same presentation. Furthermore, the examples showed sets of conceptual events indexed into isochronous data streams; it may be appreciated by someone of ordinary skill in the art that sets of conceptual events may be indexed into non-isochronous data streams as well. As was stated earlier, an Outline for a presentation may be indexed to the slide stream.

Referring now to the exemplary screen shown in FIG. **5**, the exemplary screen **400** shows five windows **410**, **420**, **430**, **440**, **450** contained within the display. The Video Window **410** is used to display the video stream. The Slide Window **420** is used to display the slides used in the presentation. The Transcript Window **430** is used to display the transcribed audio of the speech. The Outline Window **440** is used to display the Outline of the presentation. The Control Panel **450** is used to control the display in each of the other four windows. The Transcript Window **430** includes a Transcript Slider Bar **432** that allows the user to scroll through the transcript, and Next **433** and Previous **434** Phrase Buttons that allow the user to step through the transcript a phrase at a time, where a phrase consists of a single line of the transcript. It also includes a Hypertext Link **436**, as illustrated here in the form of the highlighted words, "Robert Jones", in the transcript. The Outline Window **440** includes an Outline Slider Bar **442** that allows the user to scroll through the outline, and Next **443** and Previous Entry buttons **444** that allow the user to jump directly to the next or previous topic. The Control Panel **450** includes a Video Slider Bar **452** used to select a position in the video stream, and a Play Button **454** used to play the program. It also includes a Slider Bar **456** used to position the program at a Slide, and Previous **457** and Next **458** Slide Buttons used to display the next and previous Slides in the Slide Window **420**. It also includes a Search Box **460** used to search for text strings (e.g., words) in the Transcript.

FIG. **5** shows the beginning of a presentation, corresponding to a time code of zero. The speaker's first slide is displayed in the Slide Window **410**, the speaker's first words are displayed in the Transcript Window **430**, and the beginning of the outline is displayed in the Outline Window **440**. The user can press the play button **454** to begin playing the presentation, which will cause the video and audio data to begin streaming, the transcript and outline scroll in synchronization with the video and audio, and the slides to advance at the appropriate times.

Alternatively, the user can jump directly to a point of interest. FIG. **6** shows the result of the user selecting the second entry in the Outline from Outline Window **440'**, entitled "The First Manned Flight" (recall entry **322** of Outline **320** in FIG. 3). From the Outline Index **327** in FIG. **3**, the system determines that the time code **327** of "The First Manned Flight" is 20.50 s. The system looks in the Slide Index **315** (also in FIG. **3**) and determines that the second slide S2 begins at time code **317** of 20.40 s, and thus the second slide should be displayed in the Slide Window **420'**. The system looks at the Time-Coded Transcript **215** (shown in FIG. **2**), locates the word "the" **235** that begins on or immediately after time code of 20.50 s, and displays that word and the appropriate number of subsequent words to fill up the Transcript Window **430'**. The effect of this operation is that the user is able to jump directly to a point in the presentation, and the system positions each of the synchronized data streams to that point, including the video in Video Window **410'**. The user may then begin playing the presentation at this point, or upon scanning the newly displayed slide and transcript jump directly to another point in the presentation.

Referring now to FIG. 7, the flowchart starting at **600** indicates the operation of an embodiment of the present invention. When the user slides the video slider bar **452** in FIG. 5, the Event Handler **601** in FIG. 7 receives a Move Video Slider Event **610**. The Move Video Slider Event **610** causes the invention to calculate the video frame of the new position of the slider **452**. The position of the video slider **452** is translated into the position in the video data stream in a proportional fashion. For example, if the new position of the video slider **452** is positioned half-way along its associated slider bar, and the video stream consist of 10,000 frames of video, then the 5,000th frame of video is displayed on the Video Window **420**. The invention displays the new video frame **611**, and computes the time code of the new video frame **612**. Using this new time code, the system looks up the Slide associated with the displayed video frame, and displays **613** the new Slide in the Slide Window **410**. Again using this new time code, the system looks up the Phrase associated with the displayed video frame, and displays the new Phrase **614** in the Transcript Window **430**. Again using this new time code, the system looks up the Outline Entry associated with the displayed video frame, and displays the new Outline Entry **615** in the Outline Window **440**. Finally, using this new time code, the system looks up the Hypertext Links associated with the displayed video frame, and displays them **616** in the appropriate place in the Transcript Window **430**.

Referring back to FIG. 5, when the user moves the Slide Slider Bar **456** or presses the Previous **457** and Next **458** Slide Buttons, the Event Handler **601** in FIG. 7 receives a New Slide Event **620**. The New Slide Event causes the system to display the selected new Slide **621** in the Slide Window **420**, and to look up the time code of the new Slide in the Slide Index **622**. Using the time code of the new Slide as the new time code, the system computes the video frame associated with the new time code and displays the indicated video frame **623** in the Video Window. Again using the new time code, the system looks up the Phrase associated with the displayed Slide, and displays the new Phrase **624** in the Transcript Window **430**. Again using the new time code, the invention looks up the Outline Entry associated with the displayed Slide, and displays the new Outline Entry **625** in the Outline Window **440**. Finally, using the new time code, the system looks up the Hypertext Links associated with the displayed Slide, and displays them **626** in the appropriate place in the Transcript Window **430**.

Referring again back to FIG. **5**, when the user moves the Transcript Slider Bar **432** or presses the Next **433** or Previous **434** Phrase Buttons, the Event Handler **601** in FIG. **7** receives a New Phrase Event **630**. The New Phrase Event causes the system to display the selected new Phrase **631** in the Transcript Window **430**, and to look up the time code of the new Phrase in the Transcript Index **632**. Using the time code of the new Phrase as the new time code, the invention computes the video frame associated with the new time code and displays the indicated video frame **633** in the Video Window **410**. Again using the new time code, the invention looks up the Slide associated with the displayed Phrase, and displays the new Slide **634** in the Slide Window. Again using the new time code, the invention looks up the Outline Entry associated with the displayed Phrase, and displays the new Outline Entry **635** in the Outline Window **440**. Finally, using the new time code, the invention looks up the Hypertext Links associated with the displayed Phrase, and displays them **636** in the appropriate place in the Transcript Window **430**.

Referring yet again to FIG. **5**, when the user types a search string into the Search Box **460** and initiates a search, the Event Handler **601** in FIG. **7** receives a Search Transcript Event **640**. The Search Transcript event causes the system to employ a string matching algorithm of a type well-known in the art to scan the Transcript and locate the first occurrence of the search string **641**. The system uses the Transcript Index to determine which Phrase contains the matched string in the Transcript **642**. The system displays the selected new Phrase **631** in the Transcript Window, and looks up the time code of the new Phrase in the Transcript Index **632**. Using the time code of the new Phrase as the new time code, the system computes the video frame associated with the new time code and displays the indicated video frame **633** in the Video Window **410**. Again using the new time code, the system looks up the Slide associated with the displayed Phrase, and displays the new Slide **634** in the Slide Window **420**. Again using the new time code, the system looks up the Outline Entry associated with the displayed Phrase, and displays the new Outline Entry **635** in the Outline Window **440**. Finally, using the new time code, the system looks up the Hypertext Links associated with the displayed Phrase, and displays them **636** in the appropriate place.

Referring to FIG. **5**, when the user moves the Outline Slider Bar **442** or presses the Next **443** or Previous **444** Outline Entry Buttons, the Event Handler **601** in FIG. **7** receives a New Outline Entry Event **650**. The New Outline Entry Event causes the system to display the selected new Outline Entry **651** in the Outline Window **440**, and to look up the time code of the new Outline Entry in the Outline Index **652**. Using the time code of the new Outline Entry as the new time code, the system computes the video frame associated with the new time code and displays the indicated video frame **653** in the Video Window **410**. Again using the new time code, the system looks up the Slide associated with the displayed Outline Entry, and displays the new Slide **654** in the Slide Window **420**. Again using the new time code, the system looks up the Phrase associated with the displayed Outline Entry, and displays the new Phrase **655** in the Transcript Window **430**. Finally, using the new time code, the system looks up the Hypertext Links associated with the displayed Outline Entry, and displays them **656** in the appropriate place in the Transcript Window **430**.

Referring again to FIG. 5, when the user selects a Hypertext Link **436**, the Event Handler **601** in FIG. 7 receives a Display Hypertext Object **660**. The system displays the data object pointed to by the selected Hypertext Link **661**.

Whenever the system is in a stationary state, that is, when no video/audio stream is being played, the system maintains a record of the current time code. The data displayed in FIGS. 4 and 5 always correspond to the current time code. When the user presses the Play Button **454**, the Event Handler **601** in FIG. 5 receives a Play Program Event **670**. The system begin playing the video and audio streams, starting at the current time code. Referring now to FIG. 8, as each new video frame is displayed **700**, the system uses the time code of the displayed video frame to check the Transcript Index, the Slide Index, the Outline Index, and Hypertext Index and determine if the data displayed in the Slide Window **420**, Transcript Window **430**, or Outline Window **440** must be updated, or if new Hypertext Links must be displayed in the Transcript Window **430**. If the time code of the new video frame corresponds to the time code of the next Phrase **710**, the system displays the next Phrase **711** in the Transcript Window **430**. If the time code of the new video frame corresponds to the time code of the next Slide **720**, the system displays the next Slide **721** in the Slide Window **420**. If the time code of the new video frame corresponds to the time code of the next Outline Entry **730**, the system displays the next Outline Entry **731** in the Outline Window **440**. Finally, if the time code of the new video frame corresponds to the time codes of a different set of Hypertext Links than are currently displayed **740**, the system displays the new set of Hypertext Links **741** at the appropriate places on the display in the Transcript Window **430**.

It may be appreciated by one of ordinary skill in the art that the textual transcript may be translated into other languages. Multiple transcripts, corresponding to multiple languages, may be synchronized to the same time base, corresponding to a single video/audio stream. Users may choose which transcript language to view, and may switch among different transcripts in different languages during the operation of the invention.

Furthermore, multiple synchronized streams of each data type may be incorporated into a single multimedia program. Multiple video/audio streams, each corresponding to different video resolution, audio sampling rate, or data compression technology, may be included in a single program. Multiple sets of slides, hypertext links, and other streams of isochronous data types may also be included in a single program. One or more of each data type may be displayed on the computer screen, and users may switch among the different streams of data available in the program.

The present invention is compatible with operating with a collection of many presentations, and to assist users in locating the particular portion of the particular presentation that most interests them. The presentations are stored in a data base of a type well-known in the art, which may range from a simple non-relational data base that stores data in disk files to a complex relational or object-oriented data base that stores data in a specialized format. Referring to the exemplary screen **800** depicted in FIG. **9**, users can issue structured queries or full text queries to identify programs they wish to view. The user types in a query in the query type-in box **810**. The titles of the programs that match the query are displayed in the results box **820**. Structured queries are queries that allow the user to select programs on the basis of structured information associated with each program, such as title, author, or date. Using any of the structured query engines well-known in the art, the user can specify a particular title, author, range of dates, or other structured query, and select only those programs which have associated structured information that matches the query. Full-text queries are queries that allow the user to select programs on the basis of text associated with each program, such as the abstract, transcript, slides, or ancillary materials connected via hypertext. Using any of the full-text search engines known in the art, the user can specify a particular combination of words and phrases, and select only those programs which have associated text that matches the full-text query. Users can also select which of the associated text elements to search. For example, the user can specify to search only the transcript, only the slides, or a combination of both. When the text associated with a program matches the user's query, the user can jump directly to the matched text, and display all of the other synchronized multimedia data types at that point in the program.

Full-text queries can be manually constructed by the users, or they can be automatically constructed by the invention. Such automatically-constructed queries are referred to as "agents." FIG. **10** presents a flow chart of the agent mechanism starting at **900**. When the user displays a program **910**, the system constructs a summary of the program **920**. The summary of the program may be constructed in multiple alternative ways. Each program may have associated with it a list of keywords that describe the major subjects discussed in the program. In this case, constructing the summary simply involves accessing this predefined list of keywords. Alternatively, any text summarization engine well-known in the art may be run across the text associated with program, including the abstract, the transcript, and the slides, to generate a list of keywords that describe the major subjects discussed in the program. This summary is added to the user's profile **930**. The user's profile is a list of keywords that collectively describe the programs that the user has viewed in the past. Each time the user views a new program, the keywords that describe that program are added to the user's profile. In this manner, the agent "learns" which subjects are most interesting to the user, and continues to learn about the user's changing interests as the user uses the system. The agent mechanism also incorporates the concept of memory. Each keyword that is added to the user's profile is labeled with the date at which its associated program was viewed. Whenever the agent mechanism is initiated, the difference between the current date and the date label on each keyword is used to assess the relative importance of that keyword. Keywords that entered the profile more recently are treated as more important than keywords that entered the profile in the distant past. On specified events, such as the user logging into the system, the agents mechanism is initiated **901**. The system creates a query from the current user's profile **940**. The list of keywords in the profile are reorganized into the query syntax required by the full-text search engine. The ages of the keywords are converted into the relative importance measure required by the full-text search engine. The query is run against all of the programs on the server **950**, and the resulting list of programs are presented to the user **960**. This list of programs constitutes the programs which the system has determined may be of interest to the user, based on the user's past viewing behavior.

In addition, users can create their own agents by manually constructing a query that describe their ongoing interest. Each time the agents' mechanism is initiated, the user's manually-constructed agents are executed along with the system's automatically-constructed agent, and the selected programs are presented to the user.

The user can create "virtual conferences" that consist of user-defined aggregations of programs. To create a virtual conference, a user composes and executes a query that selects a set of programs that share a common attribute, such as author, or discuss a common subject. This thematic aggregation of programs can be named, saved, and distributed to other users interested in the same theme.

The user can construct "synthetic programs" by sequencing together segments of programs from multiple different programs. To create a synthetic program, the user composes and executes a query, specifying that the invention should select only those portions of the programs that match the query. The user can then view the concatenated portions of multiple programs in a continuous manner. The synthetic program can be named, saved, and distributed to other users interesting in the synthetic program content.

Referring now to FIG. **11**, which will be used to describe the operation of an embodiment of the present invention across a non-isochronous network connection. This embodiment incorporates a cooperative processing data distribution and caching model that enables the isochronous data streams to play continuously immediately following a navigational event, such as moving to the next slide or searching to a particular word in the transcript.

After the process starts **1000**, when the user first selects a program to play **1001**, the system downloads the selected portions of the non-isochronous data from the server to the client. The downloaded non-isochronous data includes the Slide Index, the Slides, the Transcript Index, the Transcript, and the Hypertext Index. The downloaded non-isochronous data is stored in a disk cache **1010** on the client. The purpose of pre-downloading this non-isochronous data is to avoid having to transmit it over the network connection simultaneously with the transmission of the isochronous data, thereby interrupting the transmission of the isochronous data. The Hypertext Objects are not pre-downloaded to the client; rather, the system is designed to pause the transmission of the isochronous data to accommodate the downloading of any Hypertext Objects. At the end of playing a program, the client disk cache is emptied in preparation for use with another program.

In addition to downloading portions of the non-isochronous data, the system downloads a segment of the isochronous data from the server to a memory cache on the client. The downloaded isochronous data includes the initial segment of the video data and the corresponding initial segment of the audio data. The amount of isochronous data downloaded typically ranges from 5 to 60 seconds, but may be more or less. The downloaded isochronous data is stored in a memory cache **1020** on the client.

When the user presses the Play Button, the Event Handler **1030** receives a Play Program Event **1040**. The system begins the continuous delivery of the isochronous data to the display devices **1041**. Based on the time code of the currently displayed video frame, it also displays the associated non-isochronous data **1042**, including the Transcript, the Slides, and the Hypertext Links. As the system streams the isochronous data to the display devices, it depletes the memory cache. When the amount of isochronous data in the memory cache falls below a specified threshold, the system causes the client CPU to send a request to the server CPU for the next contiguous segment of isochronous data **1043**. This threshold typically works out to be on the order of 5-10 seconds, with a worst-case scenario of 60 seconds. It should be appreciated by one of ordinary skill in art that factors such as network capacity and usage should affect the choice of threshold. Upon receiving this data, the client CPU repopulates the isochronous data memory cache. If, as anticipated, the client CPU experiences a delay in receiving the requested data, caused by the non-isochronous network connection, the client CPU continues to deliver isochronous data remaining in its memory cache in a continuous stream to the display device, until that cache is exhausted.

The method for repopulating the client's memory cache is a critical element in supporting efficient random access into isochronous data streams over a non-isochronous network. The method for downloading the isochronous data from the server to the memory cache on the client is designed to balance two competing requirements. The first requirement is for continuous, uninterrupted delivery of the isochronous data to the video display device and speakers attached to the client CPU. The network connection between the client and server is typically non-isochronous, and may introduce significant delays in the transmission of data from the client to the server. In practice, if the memory cache on the client becomes empty, requiring client to send a request across the network to the server for additional isochronous data, the amount of time needed to send and receive the request will cause the interruption of play of the isochronous data. The requirement for continuous delivery thus encourages the caching of as much data as possible on the client. The second requirement is to minimize the amount of data that is transmitted across the network. In practice, multiple users share a fixed amount of network bandwidth, and transmitting video and audio data across a network consumes a substantial portion of this limited resource. It is anticipated that a common user behavior will be to use the random access navigation capabilities to reposition the program. But the act of repositioning the program invalidates all or part of the data stored in the memory cache in the client. The larger the amount of data that is stored in the memory cache on the client, the more data is wasted upon repositioning the program, and thus the more network bandwidth was wasted in sending this unused data from the server to the client. Thus the requirement for minimizing the amount of data transmitted across the network encourages the caching of as little data as possible on the client.

The present invention balances the need for continuous delivery of isochronous data to the display devices with the need to avoid wasting network bandwidth by implementing a novel cooperative processing data distribution and caching model. The memory cache on the client is designed specifically for compressed isochronous data, and more specifically for compressed digital video data. The caching strategy differs markedly from traditional caching strategies. Traditional caching strategies measure the number of bytes of data in the cache, and repopulate the cache when the number of bytes falls below a specified threshold. By contrast, one embodiment of the present invention measures the number of *seconds* of isochronous data in the memory cache, and repopulates the cache when the number of seconds falls below a specified threshold. Due to the inherent inhomogeneities in video compression, a fixed number of seconds of compressed video data does not correspond to a fixed number of bytes of data. For video data streams that compress into a smaller than average number of bytes per second, the cooperative distribution and caching model reduces the amount of data sent across the network compared to a traditional caching scheme. For video data streams that compress into a larger than average number of bytes per second, the cooperative distribution and caching model guarantees a certain number of seconds of video data cached on the server, reducing the likelihood of interrupted play of the video data stream compared to a traditional caching scheme.

In addition to designing the memory cache to contain a range of a number of seconds of isochronous data, the memory cache employs a policy of *unbalanced look ahead and look behind*. *Look ahead* refers to caching the isochronous data corresponding to "N" seconds into the future. This isochronous data will be delivered to the display device under the normal operation of playing the program. *Look behind* refers to caching the isochronous data corresponding to "M" seconds into the past. This isochronous data will be delivered to the display device under the frequent operation of replaying the previously played few seconds of the program. *Unbalanced* refers to the policy of caching a different amount (that is, a different number of seconds) of look ahead and look behind data. Generally, more look ahead data is cached than look behind data, typically in the approximate ratio of 7:1. It can be appreciated by one of ordinary skill in the art that different caching policies can be employed in anticipation of different common user behaviors. For example, the use of a circular data structure, a structure well-known in the art, may effect this operation.

During program play **1040**, the server sends data to the client at the nominal rate of one second of isochronous data each second. The server adapts to the characteristics of the network, bursting data if the network supports a high burst rate, or steadily transmitting data if the network does not support a high burst rate. The client monitors its memory cache, and sends requests to the server to speed up or slow down. The client also sends requests to the server to stop, restart at a new place in the program, or start playing a different program.

The system administrator can specify how much network bandwidth is available to the system, for each individual program, and collectively across all programs. The system automatically tunes its memory caching scheme to reflect these limits. If the transmitted data would exceed the specified limits, the system automatically drops video frames as necessary.

When the user performs a navigational activity, such as moving to the next slide or searching to a particular word in the transcript, the Event Hander **1030** receives a Navigational Event **1050**. The system computes the time base value of the new position **1051**. It then downloads a new segment of the isochronous data from the server to the memory cache on the client **1052**. The downloaded isochronous data includes a segment of the video data and a corresponding segment of the audio data. The system then displays the video frame corresponding to the current time base value, and the non-isochronous data corresponding to the displayed video frame **1053**.

When the user selects a hypertext link, the Event Handler **1030** receives a Display Hypertext Object Event **1060**. The system pauses the play of the program **1061**. The client CPU requests that the server CPU send the Hypertext Object across the network connection **1062**, and upon receiving the Hypertext Object, causes it to be displayed **1063**.

Referring back to FIG. **1**, the server **130** records the actions of each user, including not only which programs each user viewed, but also which portions of the programs each user viewed. This record can be used for usage analysis, billing, or report generation. The user can ask the server **130** for a usage summary, which contains an historical record of that particular user's usage. A manager or system administrator can ask the server **130** for a summary across some or all users, thereby developing an understanding of the patterns of usage. One might use any of the data mining tools as is known in the art for assisting in this purpose.

The usage record may serve as a guide to restructure old programs or to structure new ones, having learned what works from a presentation perspective and what does not, for example. The usage record furthermore enables the system to notify users of changing data. The list of users who have viewed a program can be determined from the usage records. If a program is updated, the system reviews the usage record to determine which users have viewed the program, and notifies them that the program that they previously viewed has changed.

While the present invention has been described in terms of a few embodiments, the disclosure of the particular embodiment disclosed herein is for the purposes of teaching the present invention and should not be construed to limit the scope of the present invention which is solely defined by the scope and spirit of the appended claims.

Having thus described our invention, what we claim as new and desire to secure by Letters Patent is as follows:

1. A method of manipulating a plurality of streams of isochronous and non-isochronous digital data comprising the steps of:

synchronizing the plurality of streams of isochronous and non-isochronous data by reference to a common time base;

navigating to a position in any one of the plurality of streams using at least one of a sequential and a random access approach available for and adapted to the structure and contents of that stream,

identifying positions for each of the plurality of streams corresponding to the position in the navigated stream; and

simultaneously displaying at least some of the plurality of streams at the positions corresponding to the position in the navigated stream.

2. The method of claim 1, further comprising the step of delivering the plurality of streams of synchronized isochronous and non-isochronous data from a server to a client over a non-isochronous network.

3. The method of claim 1, further comprising the step of caching isochronous data on the client, and modulating the delivery of the isochronous data over the network in a manner that maintains a predetermined range of time's worth of data cached on the client.

4. The method of claim 1, further comprising the step of translating the transcript stream into one or more foreign languages, and including a plurality of such transcripts, each synchronized to a common time base and each independently navigable.

5. A system for interacting with a computerized presentation comprising:

a plurality of isochronous and non-isochronous data streams, wherein each of the plurality of streams are synchronized together by reference to a common time base;

for each of the plurality of data streams, means for at least one of sequential and random access navigation of such data stream, and means for display of such data stream; and

identification means, coupled to each of the navigation means, wherein, given a position in one of the plurality of data streams as pointed to by its associated navigation means, the identification means provides, via the common time base, the corresponding positions in the other of the plurality of data streams.

6. The system of claim 5 further comprising:

a server for storing the plurality of isochronous and non-isochronous data streams;

a client for containing the display and the access navigation means of such data streams; and

a non-isochronous network for delivery of such data streams from the server to the client device;

the client further including a data cache and a modulation means both coupled to the network, wherein one or more of the data streams delivered by the network are stored in the data cache, and further wherein the modulation means maintains a predetermined range of time's worth of data within the data cache.

7. The system of claim 5, wherein one of more of the digital data streams corresponds to a speaker giving an informational or educational presentation.

8. The system of claim 5, wherein at least one of the isochronous data streams includes digital video.

9. The system of claim 5, wherein at least one of the isochronous data streams includes digital audio.

10. The system of claim 5, wherein at least one of the non-isochronous data streams includes slides.

11. The system of claim 5, wherein at least one of the non-isochronous data streams includes hypertext links to related data objects.

12. The system of claim 5, wherein at least one of the non-isochronous data streams includes an outline of the presentation.

13. The system of claim 5, wherein at least one of the non-isochronous data streams includes a transcript of spoken words in the presentation.

14. The system of claim 13, wherein the random access navigation means corresponding to the transcript further includes a full-text search engine.

15. The system of claim 14, further comprising:

a plurality of computerized presentations which may be selected by a user, at least some of the presentations including one or more keywords associated therewith; and

a profiling means which maintains a user profile on each user, the user profile including an aggregation of at least some of the keywords of the presentations selected by the user.

16. A system for interacting with a computerized presentation comprising:

a plurality of isochronous and non-isochronous data streams;

two or more sets of conceptual events, each set indexed into one of the plurality of data streams;

for each of the plurality of data streams, means for navigation and display of such data stream, and for those data streams having a set of conceptual events, the means for navigation including a means for selection of a conceptual event;

an identification means, coupled to each navigation and each display means, wherein, given a selected conceptual event, provides the positions in each of the plurality of data streams corresponding to the event.

17. The system of claim **16** wherein a first set of conceptual events is indexed into an isochronous data stream and a second set of conceptual events is indexed into a non-isochronous data stream.

18. The system of claim **16** further comprising a bookmarking means for ad hoc creation of conceptual events.

FIG. 1

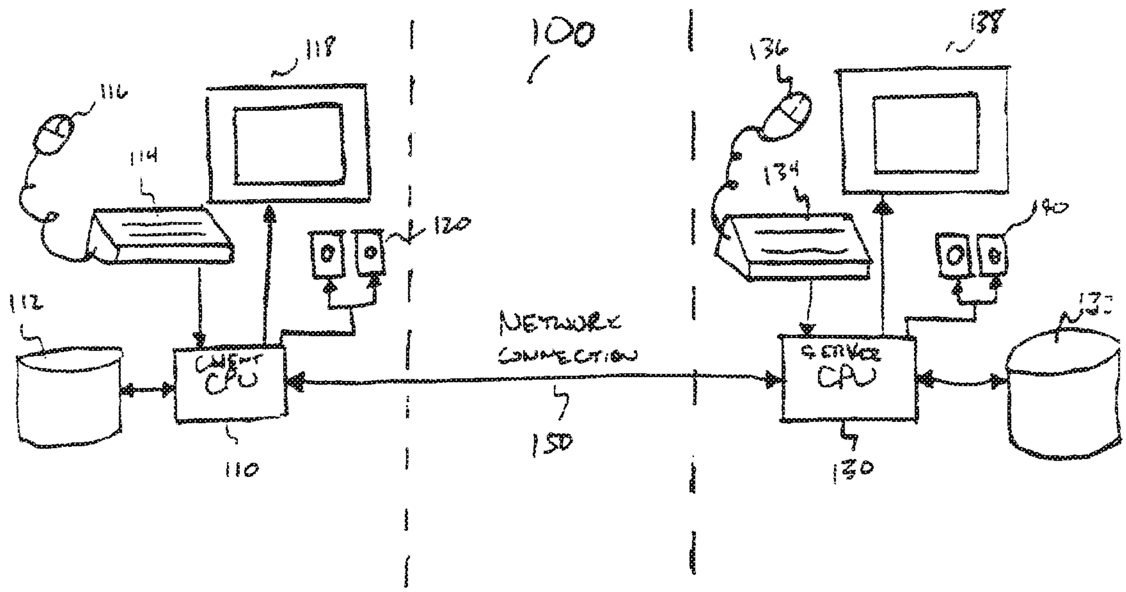

100
116
118
114
120
112
CLIENT CPU
110
NETWORK CONNECTION
150
136
138
134
140
SERVICE CPU
130

FIG. 2

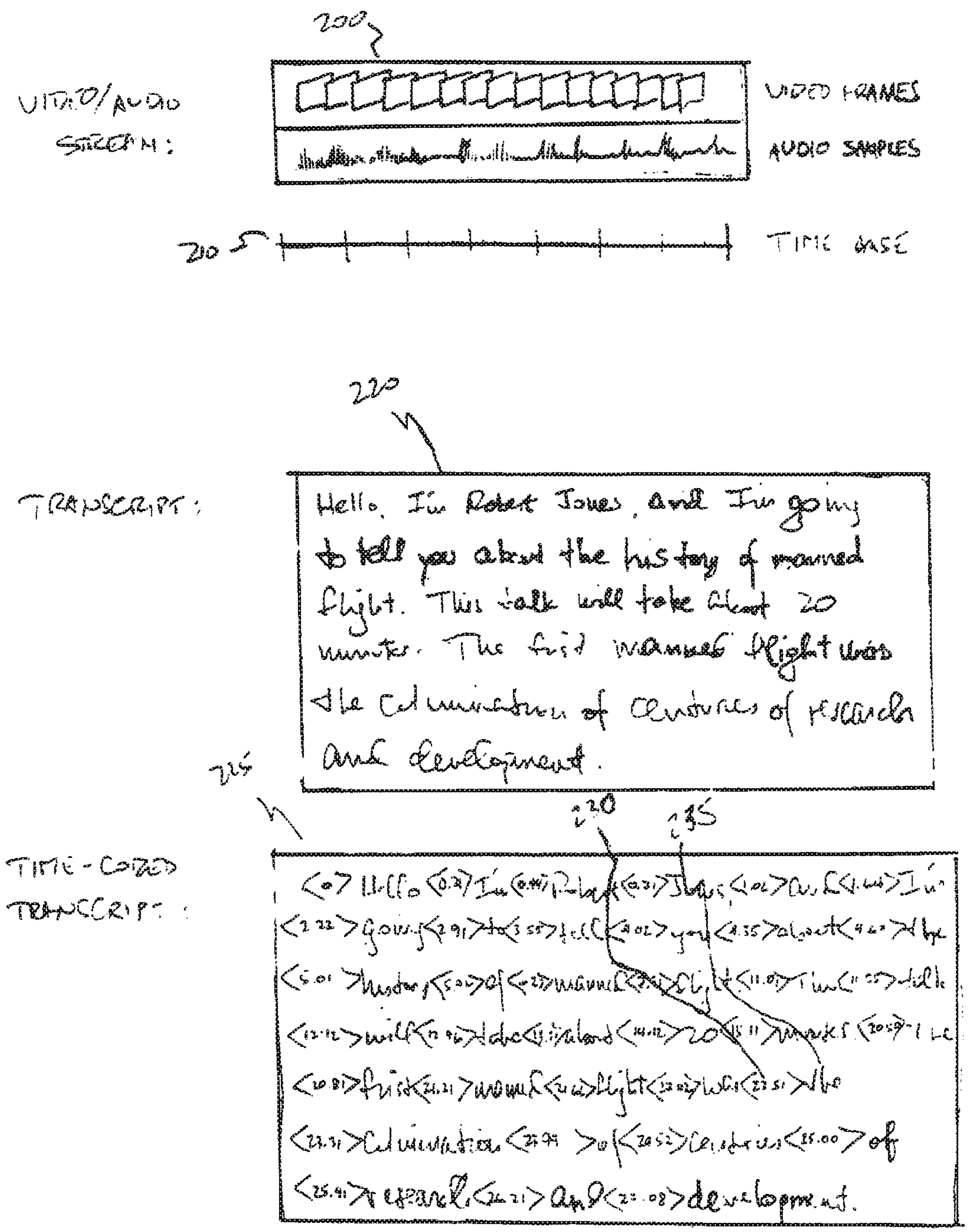
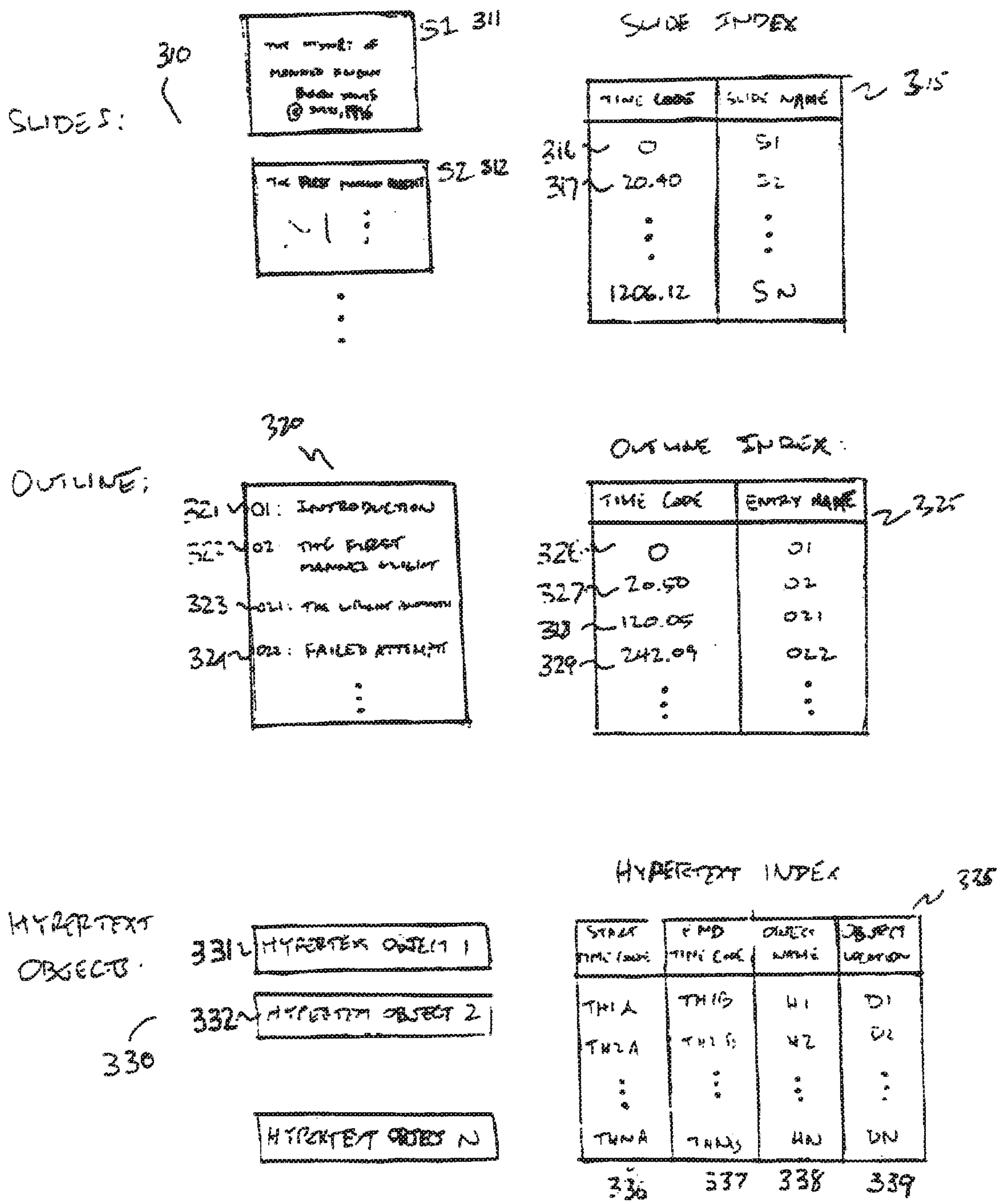

200
VIDEO/AUDIO STREAM:
VIDEO FRAMES
AUDIO SAMPLES
210
TIME BASE
220
TRANSCRIPT:
Hello, I'm Robert Jones, and I'm going to tell you about the history of manned flight. This talk will take about 20 minutes. The first manned flight was the culmination of centuries of research and development.
225
230
235
TIME-CODED TRANSCRIPT:

FIG. 3


SLIDES:
310
S1 311
S2 312
SLIDE INDEX
TIME CODE
SLIDE NAME
315
316
317
0
20.40
1206.12
S1
S2
SN
OUTLINE:
320
01: INTRODUCTION
321
322
323
324
OUTLINE INDEX:
TIME CODE
ENTRY NAME
325
326
327
329
0
20.50
120.05
242.09
01
02
021
022
HYPERTEXT OBJECTS
330
331
332
HYPERTEXT OBJECT 2
HYPERTEXT INDEX
335
336
337
338
339

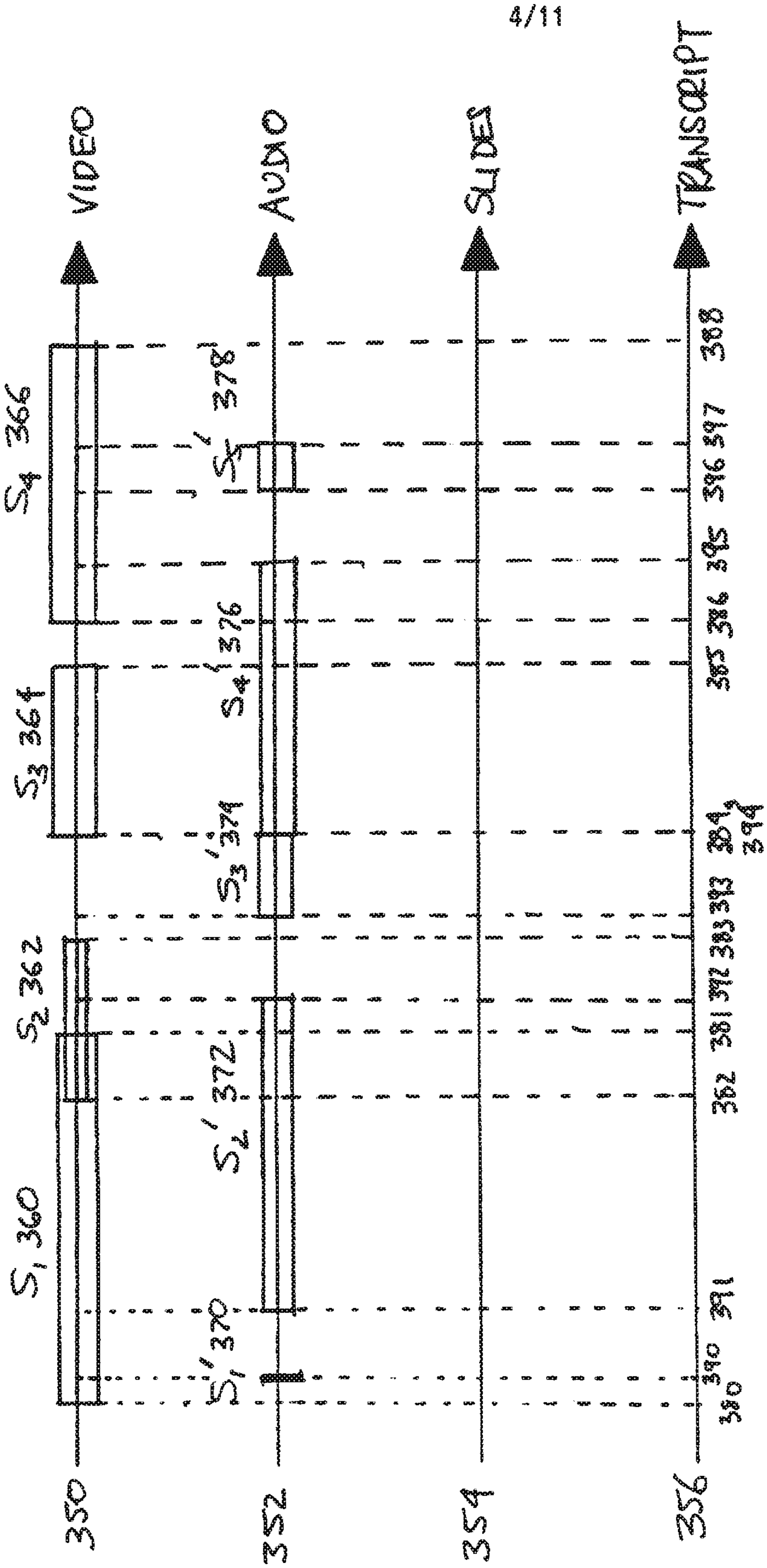

VIDEO
AUDIO
SLIDES
TRANSCRIPT
350
352
354
356
S1 360
S2 362
S3 364
S4 366
S1' 370
S2' 372
S3' 374
S4' 376
S5' 378

FIG. 4

FIG. 5

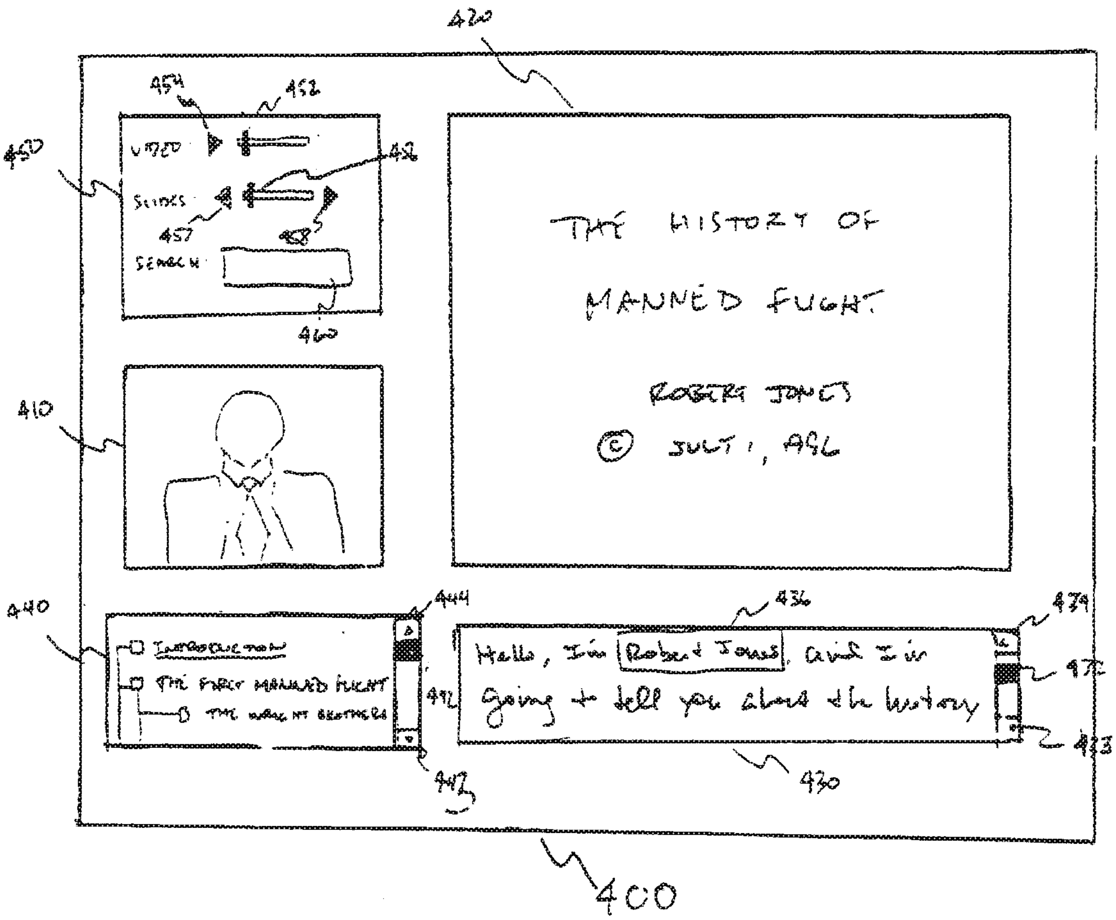

420
454
452
450
VIDEO
456
SLIDES
457
458
SEARCH
460
THE HISTORY OF
MANNED FLIGHT
ROBERT JONES
© JULY 1, 1996
410
440
444
INTRODUCTION
THE FIRST MANNED FLIGHT
THE WRIGHT BROTHERS
442
436
434
Hello, I'm Robert Jones and I'm
going to tell you about the history
432
433
430
400

FIG. 6

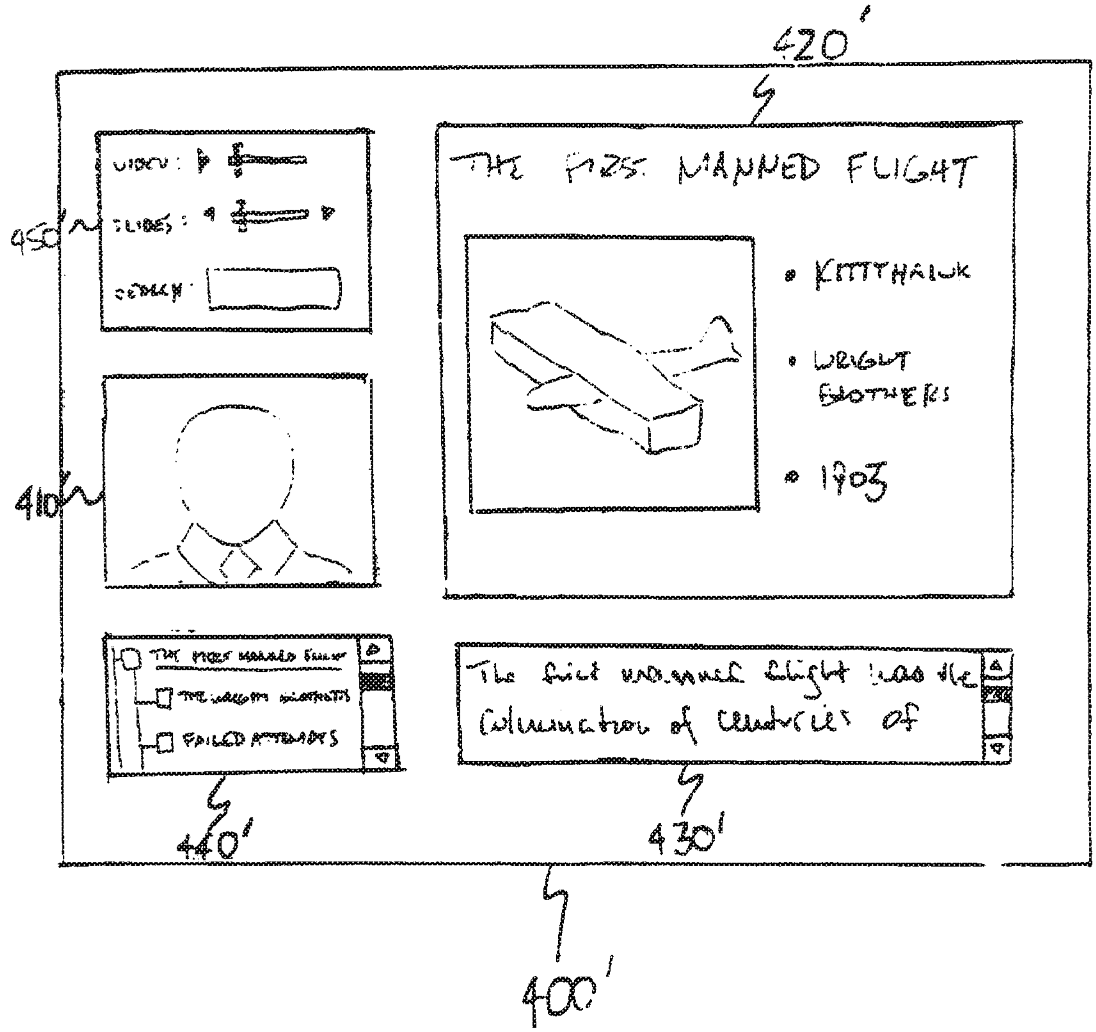

420'
450'
VIDEO:
SLIDES:
SEARCH:
410'
THE FIRST MANNED FLIGHT
• KITTYHAWK
• WRIGHT BROTHERS
• 1903
THE FIRST MANNED FLIGHT
THE WRIGHT BROTHERS
FAILED ATTEMPTS
The first manned flight was the culmination of centuries of
440'
430'
400'

FIG 7

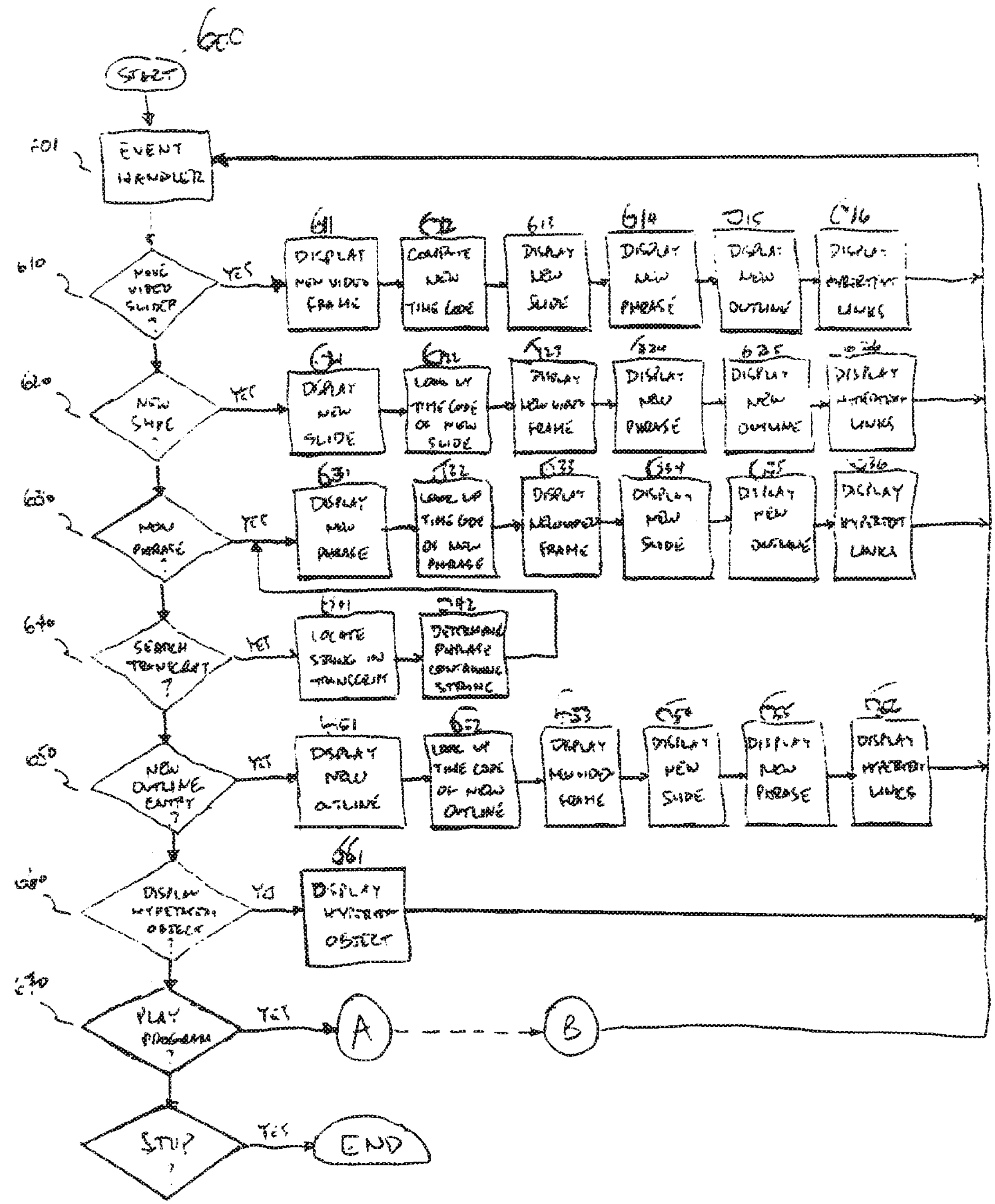

600
START
601
EVENT HANDLER
YES
DISPLAY NEW VIDEO FRAME
COMPUTE NEW TIME CODE
DISPLAY NEW SLIDE
DISPLAY NEW PHRASE
DISPLAY NEW OUTLINE
DISPLAY HYPERTEXT LINKS
NEW SLIDE ?
LOOK UP TIME CODE OF NEW SLIDE
NEW PHRASE ?
LOOK UP TIME CODE OF NEW PHRASE
SEARCH TRANSCRIPT ?
LOCATE STRING IN TRANSCRIPT
DETERMINE PHRASE CONTAINING STRING
NEW OUTLINE ENTRY ?
LOOK UP TIME CODE OF NEW OUTLINE
DISPLAY HYPERTEXT OBJECT
PLAY PROGRAM ?
A
B
STOP ?
END

FIG. 8

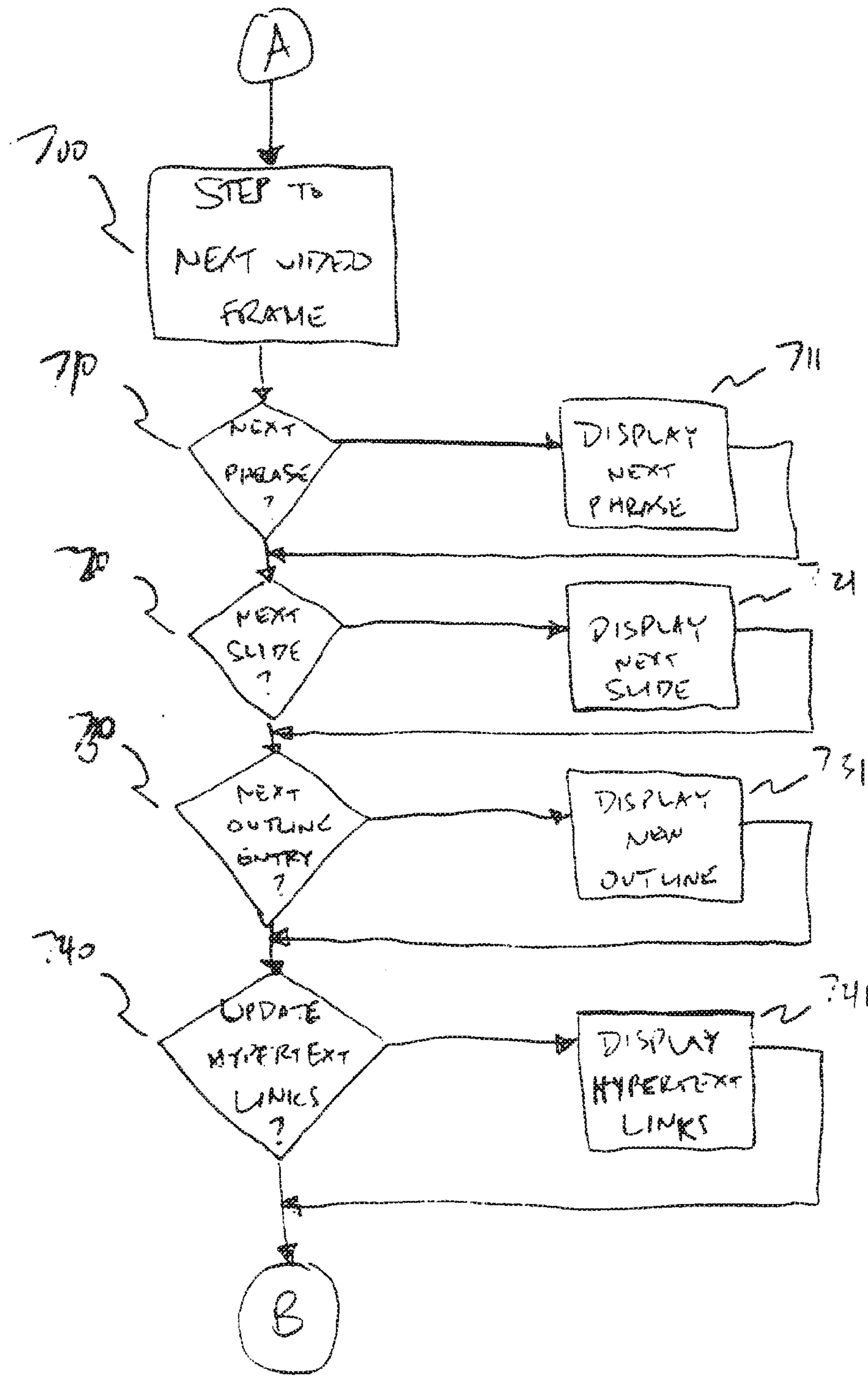

A
700
STEP TO NEXT VIDEO FRAME
710
NEXT PHRASE ?
711
DISPLAY NEXT PHRASE
720
NEXT SLIDE ?
721
DISPLAY NEXT SLIDE
730
NEXT OUTLINE ENTRY ?
731
DISPLAY NEW OUTLINE
740
UPDATE HYPERTEXT LINKS ?
741
DISPLAY HYPERTEXT LINKS
B

FIG. 9

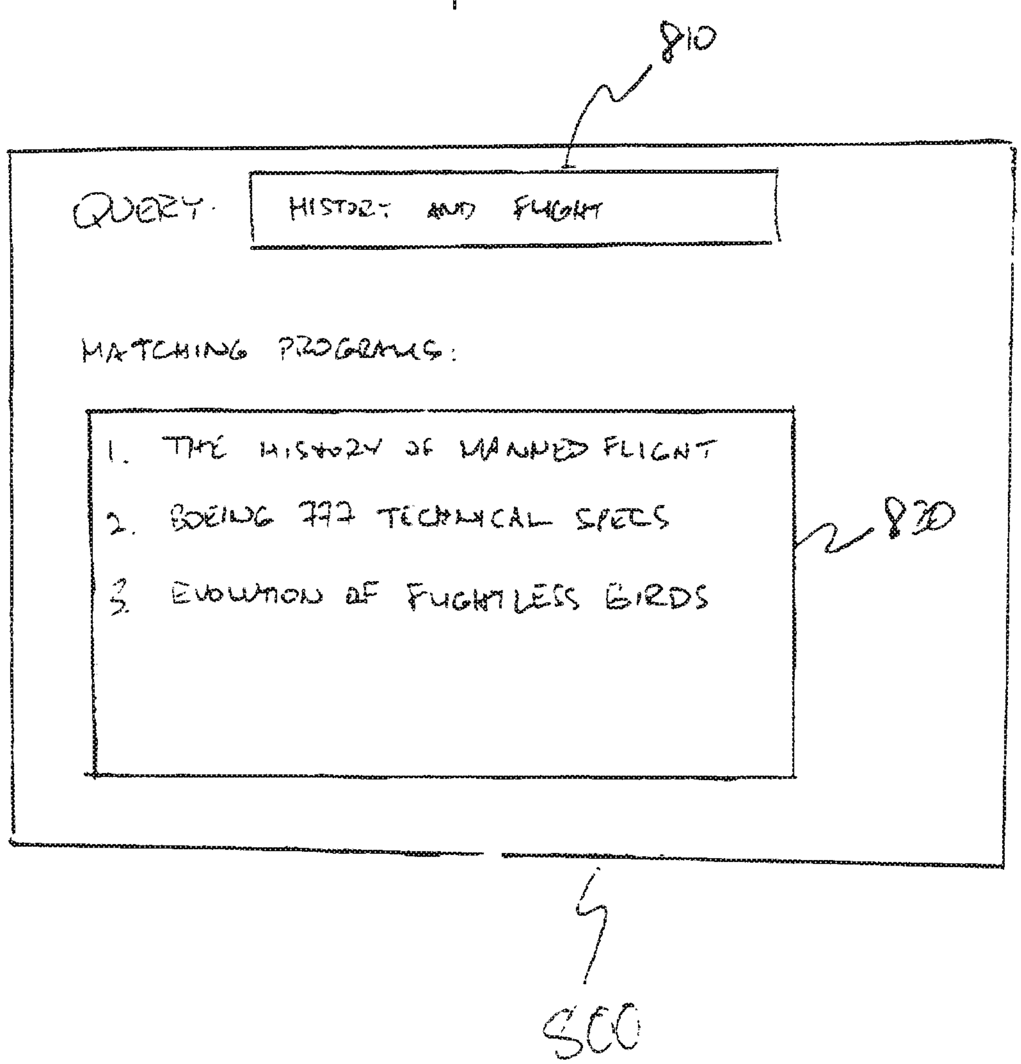

810
QUERY:
HISTORY AND FLIGHT
MATCHING PROGRAMS:
1. THE HISTORY OF MANNED FLIGHT
2. BOEING 777 TECHNICAL SPECS
3. EVOLUTION OF FLIGHTLESS BIRDS
820
800

FIG. 10

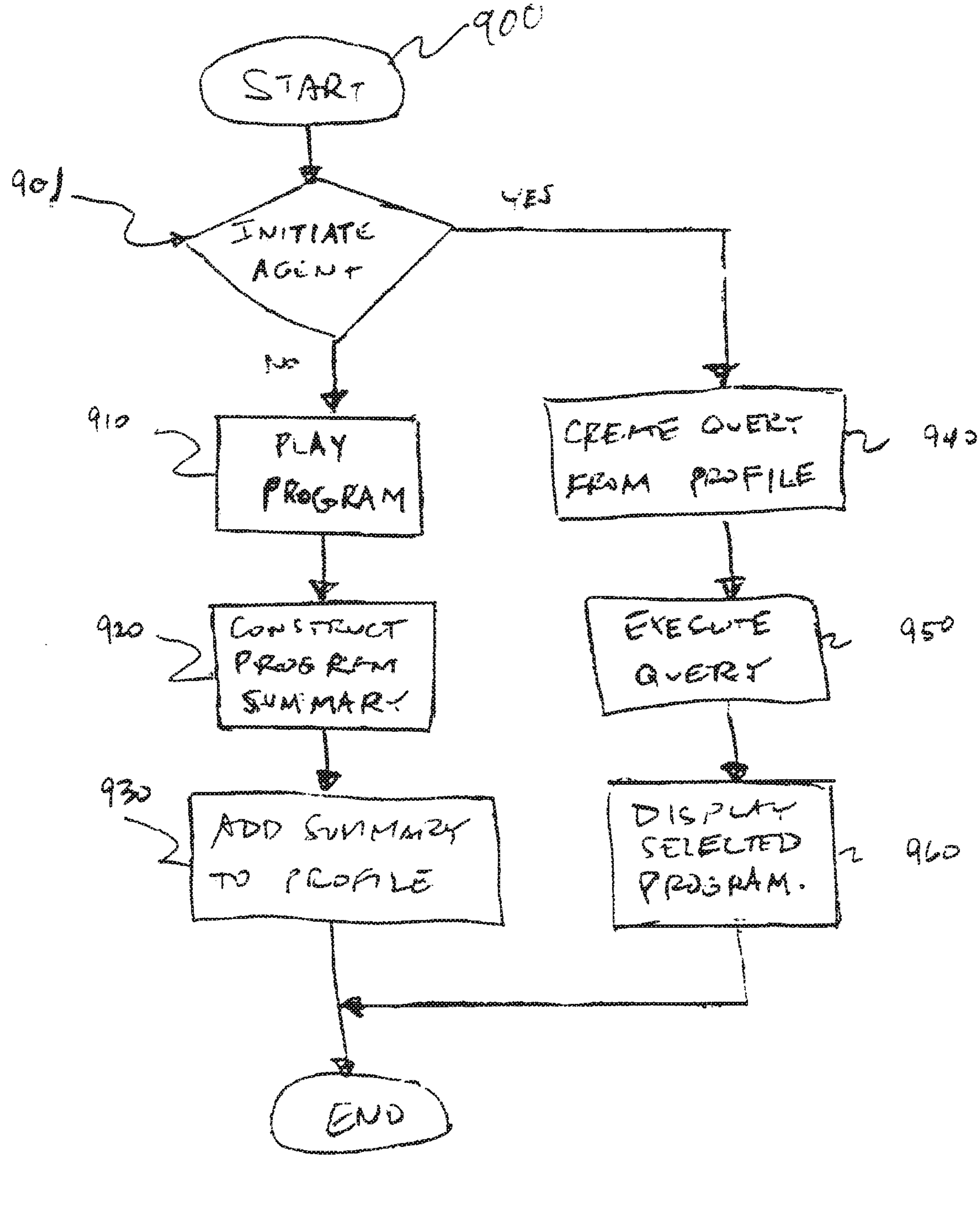
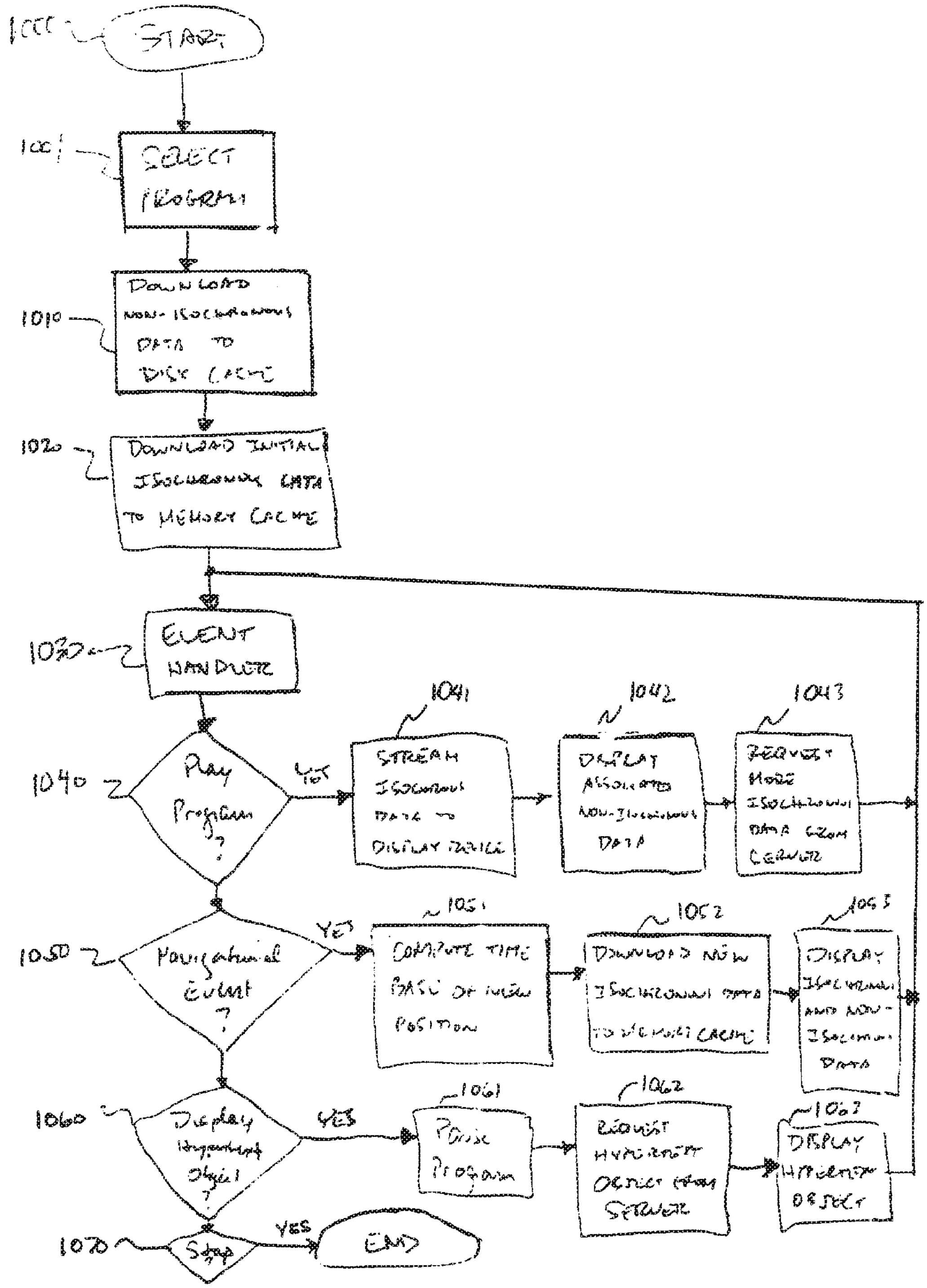

900
START
901
INITIATE AGENT
YES
NO
910
PLAY PROGRAM
920
CONSTRUCT PROGRAM SUMMARY
930
ADD SUMMARY TO PROFILE
CREATE QUERY FROM PROFILE
940
EXECUTE QUERY
950
DISPLAY SELECTED PROGRAM.
960
END

FIG. 11


START
SELECT PROGRAM
DOWNLOAD NON-ISOCHRONOUS DATA TO DISK CACHE
DOWNLOAD INITIAL ISOCHRONOUS DATA TO MEMORY CACHE
EVENT HANDLER
Play Program ?
YES
STREAM ISOCHRONOUS DATA TO DISPLAY DEVICE
DISPLAY ASSOCIATED NON-ISOCHRONOUS DATA
REQUEST MORE ISOCHRONOUS DATA FROM SERVER
Navigational Event ?
YES
COMPUTE TIME BASE OF NEW POSITION
DOWNLOAD NEW ISOCHRONOUS DATA TO MEMORY CACHE
DISPLAY ISOCHRONOUS AND NON-ISOCHRONOUS DATA
Display Hyperlinked Object ?
YES
Pause Program
REQUEST HYPERLINKED OBJECT FROM SERVER
DISPLAY HYPERLINKED OBJECT
Stop
YES
END
1010
1020
1030
1040
1041
1042
1043
1050
1060
1061
1062
1070

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US97/06982

A. CLASSIFICATION OF SUBJECT MATTER

IPC(6) :G06F 3/00, 3/14, 13/00, 13/42
US CL : 395/602, 603, 607, 807, 881, 200.02; 348/482, 515; 370/352
According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

U.S. : 395/602, 603, 607, 807, 881, 200.02; 348/482, 515; 370/352

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

APS, DIALOG

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X,P	US 5,613,909 A (STELOVSKY) 25 March 1997, Figs. 2,4-6,10,12; col. 3, lines 21-60; col. 14, lines 17-67.	1-18
Y	SoftCom LearningNet Multimedia Distance Learning. http://www.softcom.com/Learningnet.html, 1995, Softcom. Inc., see entire document.	1-16
Y	The Re:Viewer Workstation Revolutionary search, retrieval, and organization of litigation discovery data, Produci News Law Technology. vol. 2, no. 6, June 1995, see entire document.	1-16

[X] Further documents are listed in the continuation of Box C. [ ] See patent family annex.

* Special categories of cited documents:

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
24 JULY 1997	05 SEP 1997

Name and mailing address of the ISA/US	Authorized officer
Commissioner of Patents and Trademarks Box PCT Washington, D.C. 20231	MENG-AI AN
Facsimile No. (703) 305-3230	Telephone No. (703) 305-9678

Form PCT/ISA/210 (second sheet)(July 1992)*

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US97/06982

C (Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	RAVINDRAN K. ET AL, Delay Compensation Protocols for Synchronization of Multimedia Data Streams; IEEE transactions on Knowledge and Data Engineering. August 1993. Vol. 5. No. 4. pages 574-589, especially pages 574, and 588.	1-2,5,8-10,16-17
Y,E	US 5,630,117 A (OREN ET AL) 13 May 1997, col. 1, lines 62-64; col. 2, lines 50-55.	1,5,11,16
Y,E	US 5,642,171 A (BAUMGARTNER ET AL) 24 June 1997, Abstract, Fig. 6.	1,5,16
Y	US 5,471,576 A (YEE) 28 November 1995, Abstract, Fig. 2.	1,5,16
Y	US 5,274,758 A (BEITEL ET AL) 28 December 1993, Figs. 10, 14, 43.	1-16
Y	US 5,101,274 A (YOSHIMURA ET AL) 31 March 1992, Abstract	1,5,16
A,P	US 5,619,733 A (NOE ET AL) 08 April 1997, Abstract	1-18
A	US 4,761,781 A (CALVIGNAC ET AL) 02 August 1988, Abstract	1-18

Form PCT/ISA/210 (continuation of second sheet)(July 1992)*

PCT WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 6 : **G06F 3/00, 3/14, 13/00, 13/42**	A1	(11) International Publication Number: **WO 97/41504** (43) International Publication Date: 6 November 1997 (06.11.97)

(21) International Application Number: PCT/US97/06982

(22) International Filing Date: 24 April 1997 (24.04.97)

(30) Priority Data:
08/638,350 26 April 1996 (26.04.96) US

(71) Applicant *(for all designated States except US)*: ELOQUENT, INC. [US/US]; Suite 200, 1710 S. Amphlett Boulevard, San Mateo, CA 94402-2703 (US).

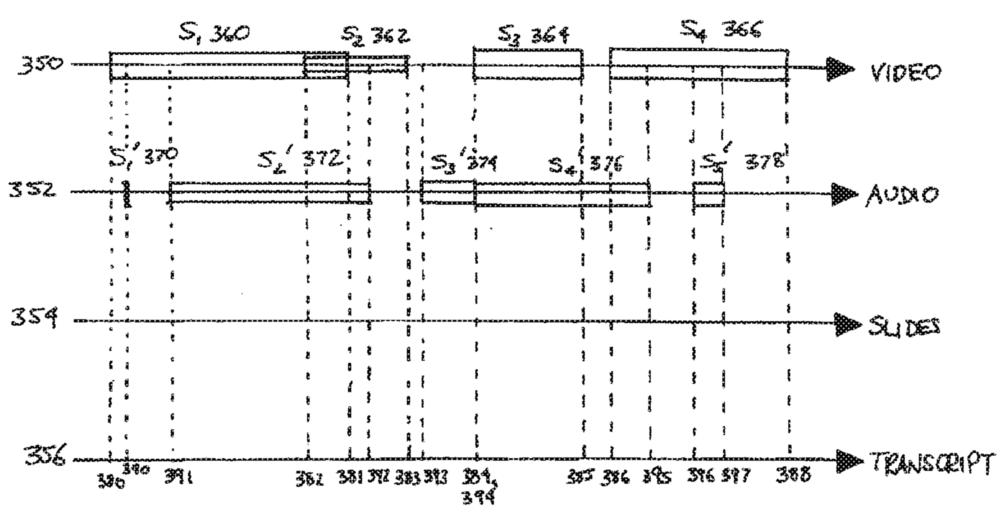
(72) Inventors; and
(75) Inventors/Applicants *(for US only)*: REID, Clifford, A. [US/US]; Apartment 2-710, 2 Townsend Street, San Francisco, CA 94107 (US). GLAZER, David [US/US]; 263 Glenwood Avenue, Woodside, CA 94062 (US).

(74) Agents: BARR, Robert et al.; Weil, Gotshal & Manges L.L.P., Suite 280, 2882 Sand Hill Road, Menlo Park, CA 94025 (US).

(81) Designated States: AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, CA, CH, CN, CU, CZ, DE, DK, EE, ES, FI, GB, GE, HU, IL, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MD, MG, MK, MN, MW, MX, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, TJ, TM, TR, TT, UA, UG, US, UZ, VN, ARIPO patent (GH, KE, LS, MW, SD, SZ, UG), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, ML, MR, NE, SN, TD, TG).

Published
With international search report.
Before the expiration of the time limit for amending the claims and to be republished in the event of the receipt of amendments.

(54) Title: A METHOD AND SYSTEM FOR SYNCHRONIZING AND NAVIGATING MULTIPLE STREAMS OF ISOCHRONOUS AND NON-ISOCHRONOUS DATA



(57) Abstract

A method and system for synchronizing multiple streams of isochronous and non-isochronous data (100) and navigating through the synchronized streams by reference to a common time base (210) and by means of a structured framework of conceptual events provides computer users with an effective means to interact with multimedia programs of speakers giving presentations (400). The multimedia programs consisting of synchronized video, audio, graphics, text, hypertext, and other data types can be stored on a server (130), and users can navigate and play them from a client CPU (110) over a non-isochronous network connection (150).

FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AL	Albania	ES	Spain	LS	Lesotho	SI	Slovenia
AM	Armenia	FI	Finland	LT	Lithuania	SK	Slovakia
AT	Austria	FR	France	LU	Luxembourg	SN	Senegal
AU	Australia	GA	Gabon	LV	Latvia	SZ	Swaziland
AZ	Azerbaijan	GB	United Kingdom	MC	Monaco	TD	Chad
BA	Bosnia and Herzegovina	GE	Georgia	MD	Republic of Moldova	TG	Togo
BB	Barbados	GH	Ghana	MG	Madagascar	TJ	Tajikistan
BE	Belgium	GN	Guinea	MK	The former Yugoslav	TM	Turkmenistan
BF	Burkina Faso	GR	Greece		Republic of Macedonia	TR	Turkey
BG	Bulgaria	HU	Hungary	ML	Mali	TT	Trinidad and Tobago
BJ	Benin	IE	Ireland	MN	Mongolia	UA	Ukraine
BR	Brazil	IL	Israel	MR	Mauritania	UG	Uganda
BY	Belarus	IS	Iceland	MW	Malawi	US	United States of America
CA	Canada	IT	Italy	MX	Mexico	UZ	Uzbekistan
CF	Central African Republic	JP	Japan	NE	Niger	VN	Viet Nam
CG	Congo	KE	Kenya	NL	Netherlands	YU	Yugoslavia
CH	Switzerland	KG	Kyrgyzstan	NO	Norway	ZW	Zimbabwe
CI	Côte d'Ivoire	KP	Democratic People's	NZ	New Zealand		
CM	Cameroon		Republic of Korea	PL	Poland		
CN	China	KR	Republic of Korea	PT	Portugal		
CU	Cuba	KZ	Kazakstan	RO	Romania		
CZ	Czech Republic	LC	Saint Lucia	RU	Russian Federation		
DE	Germany	LI	Liechtenstein	SD	Sudan		
DK	Denmark	LK	Sri Lanka	SE	Sweden		
EE	Estonia	LR	Liberia	SG	Singapore		

A METHOD AND SYSTEM FOR SYNCHRONIZING AND NAVIGATING MULTIPLE STREAMS OF ISOCHRONOUS AND NON-ISOCHRONOUS DATA

BACKGROUND OF THE INVENTION

1. Field of the Invention

The present invention generally relates to the production and delivery of video recordings of speakers giving presentations, and, more particularly, to the production and delivery of digital multimedia programs of speakers giving presentations. These digital multimedia programs consist of multiple synchronized streams of isochronous and non-isochronous data, including video, audio, graphics, text, hypertext, and other data types.

2. Description of the Prior Art

The recording of speakers giving presentations, at events such as professional conferences, business or government organizations' internal training seminars, or classes conducted by educational institutions, is a common practice. Such recordings provide access to the content of the presentation to individuals who were not able to attend the live event.

The most common form of such recordings is analog video taping. A video camera is used to record the event onto a video tape, which is subsequently duplicated to an analog medium suitable for distribution, most commonly a VHS tape, which can be viewed using a commercially-available VCR and television set. Such video tapes generally contain a video recording of the speaker and a synchronized audio recording of the speaker's words. They may also contain a video recording of any visual aids which the speaker used, such as text or graphics projected in a manner visible to the audience. Such video tapes may also be edited prior to duplication to include a textual transcript of the audio component recording, typically presented on the bottom of the video display as subtitles. Such subtitles are of particular use to the hearing impaired, and if translated into other languages, are of particular use to viewers who prefer to read along in a language other than the language used by the speaker.

Certain characteristics of such analog recordings of speakers giving presentations are unattractive to producers and to viewers. Analog tape players offer limited navigation facilities, generally limited to fast forward and rewind capabilities. In addition, analog tapes have the capacity to store only a few hours of video and audio, resulting in the need to duplicate and distribute a large number of tapes, leading to the accumulation of a large number of such tapes by viewers.

Advancements in computer technology have allowed analog recordings of speakers giving presentations to be converted to digital format, stored on a digital storage medium, such as a CD-ROM, and presented using a computer CPU and display, rather than a VCR and a television set. Such digital recordings generally include both isochronous and non-isochronous data. Isochronous data is data that is time ordered and must be presented at a particular rate. The isochronous data contained in such a digital recording generally includes video and audio. Non-isochronous data may or may not be time ordered, and need not be presented at a particular rate. Non-isochronous data contained in such a digital recording may include graphics, text, and hypertext.

The use of computers to play digital video recordings of speakers giving presentations provides navigational capabilities not available with analog video tapes. Computer-based manipulation of the digital data offers random access to any point in the speech, and if there is a text transcript, allows the users to search for words in the transcript to locate a particular segment of the speech.

Certain characteristics of state-of-the-art digital storage and presentation of recordings of speakers giving presentations are unattractive to producers and to viewers. There is no easy way to navigate directly to a particular section of a presentation that discusses a topic of particular interest to the user. In addition, there is no easy way to associate a table of contents with a presentation, and navigate directly to section of the presentation associated with each entry in the table of contents. Finally, like analog tapes, CD-ROMs can store only a view hours of digital video and audio, resulting in the need to duplicate and distribute a large number of CD-ROMs, leading to the accumulation of a large number of such CD-ROMs by viewers.

SUMMARY OF THE INVENTION

It is therefore an object of the present invention to provide a mechanism for synchronizing multiple streams of isochronous and non-isochronous digital data in a manner that supports navigating by means of a structured framework of conceptual events.

It is another object of the invention to provide a mechanism for navigating through any stream using the navigational approach most appropriate to the structure and content of that stream.

It is another object of the invention to automatically position each of the streams at the position corresponding to the selected position in the navigated stream, and simultaneously display some or all of the streams at that position.

It is another object of the invention to provide for the delivery of programs made up of multiple streams of synchronized isochronous and non-isochronous digital data across non-isochronous network connections.

In order to accomplish these and other objects of the invention, a method and system for manipulating multiple streams of isochronous and non-isochronous digital data is provided, including synchronizing multiple streams of isochronous and non-isochronous data by reference to a common time base, supporting navigation through each stream in the manner most appropriate to that stream, defining a framework of conceptual events and allowing a user to navigate though the streams using this structured framework, identifying the position in each stream corresponding to the position selected in the navigated stream, and simultaneously displaying to the user some or all of the streams at the position corresponding to the position selected in the navigated stream. Further, a method and system of efficiently supporting sequential and random access into streams of isochronous and non-isochronous data across non-isochronous networks is provided, including reading the isochronous and non-isochronous data from the storage medium into memory of the server CPU, transmitting the data from the memory of the server CPU to the memory of the client CPU, and caching the different types of data in the memory of the client CPU in a manner that ensures continuous display of the isochronous data on the client CPU display device.

BRIEF DESCRIPTION OF THE DRAWINGS

The foregoing and other objectives, aspects, and advantages of the present invention will be better understood from the following detailed description of embodiments thereof with reference to the following drawings.

FIG. **1** is a schematic diagram of the organization of a data processing system incorporating an embodiment of the present invention.

FIGS. **2** and **3** are schematic diagrams of the organization of the data in an embodiment of the present invention.

FIG. **4** is a diagram showing how two different sets of "conceptual events" may be associated with the same presentation in an embodiment of the present invention.

FIGS. **5**, **6** and **9** are exemplary screens produced in accordance with an embodiment of the present invention.

FIGS. **7**, **8**, **10**, and **11** are flow charts indicating the operation of an embodiment of the present invention.

DETAILED DESCRIPTION OF AN EMBODIMENT OF THE INVENTION

Referring now to the drawings, and more particularly to FIG. **1**, there is shown, in schematic representation, a data processing system **100** incorporating the invention. Conventional elements of the system include a client central processing unit **110** which includes high-speed memory, a local storage device **112** such as a hard disk or CD-ROM, input devices such as keyboard **114** and pointing device

116 such as a mouse, and a visual data presentation device **118**, such as a computer display screen, capable of presenting visual data perceptible to the senses of a user, and an audio data presentation device **120**, such as speakers or headphones, capable of presenting audio data to the senses of a user. Other conventional elements of the system include a server central processing unit **130** which includes high-speed memory, a local storage device **132** such as a hard disk or CD-ROM, input devices such as keyboard **134** and pointing device **136**, and a visual data presentation device **138**, and an audio data presentation device **140**. The client CPU is connected to the server CPU by means of a network connection **150**.

The invention includes three basic aspects: (1) synchronizing multiple streams of isochronous and non-isochronous data, (2) navigating through the synchronized streams of data by means of a structured framework of conceptual events, or by means of the navigational method most appropriate to each stream, and (3) delivering the multiple synchronized streams of isochronous and non-isochronous data over a non-isochronous network connecting the client CPU and the server CPU.

An exemplary form of the organization of the data embodied in the invention is shown in FIG. **2** and FIG. **3**. Beginning with FIG. **2**, the video/audio stream **200** is of a type known in the art capable of being played on a standard computer equipped with the appropriate video and audio subsystems, such as shown in FIG. **1**. An example of such a video/audio stream is Microsoft Corporation's AVI™ format, which stands for "audio/video interleaved." AVI™ and other such video/audio formats consist of a series of digital images, each referred to as a "frame" of the video, and a series of samples that make up the digital audio. The frames are spaced equally in time, so that displaying consecutive frames on a display device at a sufficiently high and constant rate produces the sensation of continuous motion to the human perceptual system. The rate of displaying frames typically must exceed ten to fifteen frames per second to achieve the effect of continuous motion. The audio samples are synchronized with the video frames, so that the associated audio can be played in synchronization with the displayed video images. Both the digital images and digital audio samples may be compressed to reduce the amount of data that must be stored or transmitted.

A time base **210** associates a time code with each video frame. The time base is used to associate other data with each frame of video. The audio data, which for the purposes of this invention consists primarily of spoken words, is transcribed into a textual format, called the Transcript **220**. The transcript is synchronized to the audio data stream by assigning a time code to each word, producing the Time-Coded Transcript **225**. The time codes (shown in angle-brackets) preceding each word in the Time-Coded Transcript correspond to the time at which the speaker begins pronouncing that word. For example, the time code **230** of 22.51 s is associated with the word **235** "the." The Time-Coded Transcript may be created manually or by means of an automatic procedure. Manual time-coding requires a person to associate a time code with each word in the transcript. Automatic time coding, for example, uses a speech recognition system of a type well-known in the art to automatically assign a time code to each word as it is recognized and recorded. The current state of the art of speech recognition systems renders automatic time coding of the transcript less economical than manual time coding.

Referring now to FIG. **3**, the set **310** of Slides S1 **311**, S2 **312**, ... that the speaker used as part of the presentation may be stored in an electronic format of any of the types well-known in the art. Each slide may consist of graphics, text, and other data that can be rendered on a computer display. A Slide Index **315** assigns a time code to each Slide. For example, Slide S1 **311** would have a time code **316** of 0 s, S2 **312** having a time code **317** of 20.40 s, and so on. The time code corresponds to the time during the presentation at which the speaker caused the specified Slide to be presented. In one embodiment, all of the Slides are contained in the same disk file, and the Slide Index contains pointers to the locations of each Slide in the disk file. Alternatively, each Slide may be stored in a separate disk file, and the Slide Index contains pointers to the files containing the Slides.

An Outline **320** of the presentation is stored as a separate text data object. The Outline is a hierarchy of topics **321, 322**, .. that describe the organization of the presentation, analogous to the manner in which a table of contents describes the organization of a book. The outline may consist of an arbitrary number of entries, and an arbitrary number of levels in the hierarchy. An Outline Index **325** assigns a time code to each entry in the Outline. The time code corresponds to the time during the presentation at which the speaker begins discussing the topic represented by the entry in the Outline. For example, topic **321**, "Introduction" has entry name "01" and time code **326** of 0 s, topic **322** "The First Manned Flight" has entry name "02" and time code **327** of 20.50 s, "The Wright Brothers" **323** has entry name "021" (and hence is a subtopic of topic **322**) with time code **328** of 120.05 s, and so on. The Outline and the Outline Index may be created by means of a manual or an automatic procedure. Manual creation is accomplished by a person viewing the presentation, authoring the Outline, and assigning a time code to each element in the outline. Automatic creation may be accomplished by automatically constructing the outline consisting of the titles of each of the Slides, and associating with each entry on the Outline the time code of the corresponding Slide. Note that manual and automatic creation may produce different Outlines.

The set **330** of Hypertext Objects **331, 332**, ... relating to the subject of the presentation may be stored in an electronic formats of various types well-known in the art. Each Hypertext Object may consist of graphics, text, and other data that can be rendered on a computer display, or pointers to other software applications, as spreadsheets, word processors, and electronic mail systems, as well as more specialized applications such as proficiency testing applications or computer-based training applications.

A Hypertext Index table **335** is used to assign two time codes and a display location to each Hypertext Object. The first time code **336** corresponds to the earliest time during the presentation at which the Hypertext Object relates to the content of the presentation. The second time code **337** corresponds to the latest time during the presentation at which the Hypertext Object relates to the content of the presentation. The Object Name **338**, as the name suggests, denotes the Hypertext Object's name. The display location **339** denotes how the connection to the Hypertext Object, referred to as the Hypertext Link, is to be displayed on the computer screen. Hypertext Links may be displayed as highlighted words in the Transcript or the Slides, as buttons or menu items on the end-user interface, or in other visual presentation that may be selected by the user.

It may be appreciated by one of ordinary skill in the art that other data types may be synchronized to the common time base in a manner similar to the approaches used to synchronize the video/audio stream with the Transcript, the Slides, and the Hypertext Objects. Examples of such other data types include animations, series of computer screen images, and other specialty video streams.

An Outline represents an example of what is termed here a set of "conceptual events." A conceptual event is an association one makes with a segment of a data stream, having a beginning and end (though the beginning and end may be the points), that represents something of interest. These data segments delineating a set of conceptual events may overlap each other, and furthermore, need not cover the entire data stream. An Outline represents a set of conceptual events that does cover the entire data stream and, if arranged hierarchically, such as with sections and subsections, has sections covering subsections. In the Outline **320** of FIG. **3**, one has the sections 01:"Introduction" **321**, 02:"The First Manned

Flight" **322** , and so on, covering the entire presentation. The subsections 021:"The Wright Brothers" **324**, 022:"Failed Attempts" **324** and so on, represents another coverage of the same segment as 02:"The First Manned Flight" **322**. In accordance with the principles of the present invention, multiple Outlines, created manually or automatically, may be associated with the same presentation, thereby allowing different users with different purposes in viewing the presentation to use the Outline most suitable for their purposes. These Outlines have been described from the perspective of having been created beforehand, but there is no reason, under the principles of the present invention, for this to be so. It should be readily understood by one of ordinary skill in the art that a similar approach would allow a user to create a set of "bookmarks" that denote particular segments, or user-chosen "conceptual events" within presentations. The bookmarks allow the user, for example, to return quickly to interesting parts of the presentation, or to pick up at the previous stopping point.

With reference to FIG. **4**, the implementation of sets of conceptual events may be understood. There are time lines representing the various data streams, as for example, video **350**, audio **352**, slides **354** and transcript **356**. There are two sets of conceptual events or data segments of these time lines shown, S_1 **360**, S_2 **362**, S_3 **364**, S_4 **366**, ... and S'_1 **370**, S'_2 **372**, S'_3 **374**, S'_4 **376** , S'_5 **378**, ..., the first set indexed into the video **350** stream and second set indexed into the audio **352** stream. Thus, the first set S_1 **360**, S_2 **362**, S_3 **364**, etc., would respectively invoke time codes **380** and **381**, **382** and **383**, **384** and **385**, etc., not only for the video **350** data stream, but for the audio **352** , slides **354** and transcript **356** streams. Similarly, the second set S'_1 **370**, S'_2 **372**, S'_3 **384**, etc., would invoke respectively time codes **390** (a point), **391** and **392**, **393** and **394** (**394** shown collinear with **384**, whether by choice or accident), etc., respectively, not only on the audio **352** data stream, but on the video **350**, slides **354** and transcript **356** streams. Consider the following example of a presentation of ice skating performed to music, with voice-over commentaries and slides showing the relative standings of the ice skaters. A first Outline might list each skater and be broken down further into the individual moves of each skater's program. A second Outline might track the musical portion of the audio stream, following the music piece to piece, even movement to movement. Thus, one user might be interested in how a skater performed a particular move, while another user might wish to study how a particular passage of music inspired a skater to make a particular move. Note that there is no requirement that two sets of conceptual events track each other in any way, they represent two different ways of studying the same presentation. Furthermore, the examples showed sets of conceptual events indexed into isochronous data streams; it may be appreciated by someone of ordinary skill in the art that sets of conceptual events may be indexed into non-isochronous data streams as well. As was stated earlier, an Outline for a presentation may be indexed to the slide stream.

Referring now to the exemplary screen shown in FIG. **5**, the exemplary screen **400** shows five windows **410**, **420**, **430**, **440**, **450** contained within the display. The Video Window **410** is used to display the video stream. The Slide Window **420** is used to display the slides used in the presentation. The Transcript Window **430** is used to display the transcribed audio of the speech. The Outline Window **440** is used to display the Outline of the presentation. The Control Panel **450** is used to control the display in each of the other four windows. The Transcript Window **430** includes a Transcript Slider Bar **432** that allows the user to scroll through the transcript, and Next **433** and Previous **434** Phrase Buttons that allow the user to step through the transcript a phrase at a time, where a phrase consists of a single line of the transcript. It also includes a Hypertext Link **436**, as illustrated here in the form of the highlighted words, "Robert Jones", in the transcript. The Outline Window **440** includes an Outline Slider Bar **442** that allows the user to scroll through the outline, and Next **443** and Previous Entry buttons **444** that allow the user to jump directly to the next or previous topic. The Control Panel **450** includes a Video Slider Bar **452** used to select a position in the video stream, and a Play Button **454** used to play the program. It also includes a Slider Bar **456** used to position the program at a Slide, and Previous **457** and Next **458** Slide Buttons used to display the next and previous Slides in the Slide Window **420**. It also includes a Search Box **460** used to search for text strings (e.g., words) in the Transcript.

FIG. **5** shows the beginning of a presentation, corresponding to a time code of zero. The speaker's first slide is displayed in the Slide Window **410**, the speaker's first words are displayed in the Transcript Window **430**, and the beginning of the outline is displayed in the Outline Window **440**. The user can press the play button **454** to begin playing the presentation, which will cause the video and audio data to begin streaming, the transcript and outline scroll in synchronization with the video and audio, and the slides to advance at the appropriate times.

Alternatively, the user can jump directly to a point of interest. FIG. **6** shows the result of the user selecting the second entry in the Outline from Outline Window **440'**, entitled "The First Manned Flight" (recall entry **322** of Outline **320** in FIG. 3). From the Outline Index **327** in FIG. **3**, the system determines that the time code **327** of "The First Manned Flight" is 20.50 s. The system looks in the Slide Index **315** (also in FIG. **3**) and determines that the second slide S2 begins at time code **317** of 20.40 s, and thus the second slide should be displayed in the Slide Window **420'**. The system looks at the Time-Coded Transcript **215** (shown in FIG. **2**), locates the word "the" **235** that begins on or immediately after time code of 20.50 s, and displays that word and the appropriate number of subsequent words to fill up the Transcript Window **430'**. The effect of this operation is that the user is able to jump directly to a point in the presentation, and the system positions each of the synchronized data streams to that point, including the video in Video Window **410'**. The user may then begin playing the presentation at this point, or upon scanning the newly displayed slide and transcript jump directly to another point in the presentation.

Referring now to FIG. 7, the flowchart starting at **600** indicates the operation of an embodiment of the present invention. When the user slides the video slider bar **452** in FIG. 5, the Event Handler **601** in FIG. 7 receives a Move Video Slider Event **610**. The Move Video Slider Event **610** causes the invention to calculate the video frame of the new position of the slider **452**. The position of the video slider **452** is translated into the position in the video data stream in a proportional fashion. For example, if the new position of the video slider **452** is positioned half-way along its associated slider bar, and the video stream consist of 10,000 frames of video, then the 5,000th frame of video is displayed on the Video Window **420**. The invention displays the new video frame **611**, and computes the time code of the new video frame **612**. Using this new time code, the system looks up the Slide associated with the displayed video frame, and displays **613** the new Slide in the Slide Window **410**. Again using this new time code, the system looks up the Phrase associated with the displayed video frame, and displays the new Phrase **614** in the Transcript Window **430**. Again using this new time code, the system looks up the Outline Entry associated with the displayed video frame, and displays the new Outline Entry **615** in the Outline Window **440**. Finally, using this new time code, the system looks up the Hypertext Links associated with the displayed video frame, and displays them **616** in the appropriate place in the Transcript Window **430**.

Referring back to FIG. 5, when the user moves the Slide Slider Bar **456** or presses the Previous **457** and Next **458** Slide Buttons, the Event Handler **601** in FIG. 7 receives a New Slide Event **620**. The New Slide Event causes the system to display the selected new Slide **621** in the Slide Window **420**, and to look up the time code of the new Slide in the Slide Index **622**. Using the time code of the new Slide as the new time code, the system computes the video frame associated with the new time code and displays the indicated video frame **623** in the Video Window. Again using the new time code, the system looks up the Phrase associated with the displayed Slide, and displays the new Phrase **624** in the Transcript Window **430**. Again using the new time code, the invention looks up the Outline Entry associated with the displayed Slide, and displays the new Outline Entry **625** in the Outline Window **440**. Finally, using the new time code, the system looks up the Hypertext Links associated with the displayed Slide, and displays them **626** in the appropriate place in the Transcript Window **430**.

Referring again back to FIG. **5**, when the user moves the Transcript Slider Bar **432** or presses the Next **433** or Previous **434** Phrase Buttons, the Event Handler **601** in FIG. **7** receives a New Phrase Event **630**. The New Phrase Event causes the system to display the selected new Phrase **631** in the Transcript Window **430**, and to look up the time code of the new Phrase in the Transcript Index **632**. Using the time code of the new Phrase as the new time code, the invention computes the video frame associated with the new time code and displays the indicated video frame **633** in the Video Window **410**. Again using the new time code, the invention looks up the Slide associated with the displayed Phrase, and displays the new Slide **634** in the Slide Window. Again using the new time code, the invention looks up the Outline Entry associated with the displayed Phrase, and displays the new Outline Entry **635** in the Outline Window **440**. Finally, using the new time code, the invention looks up the Hypertext Links associated with the displayed Phrase, and displays them **636** in the appropriate place in the Transcript Window **430**.

Referring yet again to FIG. **5**, when the user types a search string into the Search Box **460** and initiates a search, the Event Handler **601** in FIG. **7** receives a Search Transcript Event **640**. The Search Transcript event causes the system to employ a string matching algorithm of a type well-known in the art to scan the Transcript and locate the first occurrence of the search string **641**. The system uses the Transcript Index to determine which Phrase contains the matched string in the Transcript **642**. The system displays the selected new Phrase **631** in the Transcript Window, and looks up the time code of the new Phrase in the Transcript Index **632**. Using the time code of the new Phrase as the new time code, the system computes the video frame associated with the new time code and displays the indicated video frame **633** in the Video Window **410**. Again using the new time code, the system looks up the Slide associated with the displayed Phrase, and displays the new Slide **634** in the Slide Window **420**. Again using the new time code, the system looks up the Outline Entry associated with the displayed Phrase, and displays the new Outline Entry **635** in the Outline Window **440**. Finally, using the new time code, the system looks up the Hypertext Links associated with the displayed Phrase, and displays them **636** in the appropriate place.

Referring to FIG. **5**, when the user moves the Outline Slider Bar **442** or presses the Next **443** or Previous **444** Outline Entry Buttons, the Event Handler **601** in FIG. **7** receives a New Outline Entry Event **650**. The New Outline Entry Event causes the system to display the selected new Outline Entry **651** in the Outline Window **440**, and to look up the time code of the new Outline Entry in the Outline Index **652**. Using the time code of the new Outline Entry as the new time code, the system computes the video frame associated with the new time code and displays the indicated video frame **653** in the Video Window **410**. Again using the new time code, the system looks up the Slide associated with the displayed Outline Entry, and displays the new Slide **654** in the Slide Window **420**. Again using the new time code, the system looks up the Phrase associated with the displayed Outline Entry, and displays the new Phrase **655** in the Transcript Window **430**. Finally, using the new time code, the system looks up the Hypertext Links associated with the displayed Outline Entry, and displays them **656** in the appropriate place in the Transcript Window **430**.

Referring again to FIG. 5, when the user selects a Hypertext Link **436**, the Event Handler **601** in FIG. **7** receives a Display Hypertext Object **660**. The system displays the data object pointed to by the selected Hypertext Link **661**.

Whenever the system is in a stationary state, that is, when no video/audio stream is being played, the system maintains a record of the current time code. The data displayed in FIGS. **4** and **5** always correspond to the current time code. When the user presses the Play Button **454**, the Event Handler **601** in FIG. **5** receives a Play Program Event **670**. The system begin playing the video and audio streams, starting at the current time code. Referring now to FIG. **8**, as each new video frame is displayed **700**, the system uses the time code of the displayed video frame to check the Transcript Index, the Slide Index, the Outline Index, and Hypertext Index and determine if the data displayed in the Slide Window **420**, Transcript Window **430**, or Outline Window **440** must be updated, or if new Hypertext Links must be displayed in the Transcript Window **430**. If the time code of the new video frame corresponds to the time code of the next Phrase **710**, the system displays the next Phrase **711** in the Transcript Window **430**. If the time code of the new video frame corresponds to the time code of the next Slide **720**, the system displays the next Slide **721** in the Slide Window **420**. If the time code of the new video frame corresponds to the time code of the next Outline Entry **730**, the system displays the next Outline Entry **731** in the Outline Window **440**. Finally, if the time code of the new video frame corresponds to the time codes of a different set of Hypertext Links than are currently displayed **740**, the system displays the new set of Hypertext Links **741** at the appropriate places on the display in the Transcript Window **430**.

It may be appreciated by one of ordinary skill in the art that the textual transcript may be translated into other languages. Multiple transcripts, corresponding to multiple languages, may be synchronized to the same time base, corresponding to a single video/audio stream. Users may choose which transcript language to view, and may switch among different transcripts in different languages during the operation of the invention.

Furthermore, multiple synchronized streams of each data type may be incorporated into a single multimedia program. Multiple video/audio streams, each corresponding to different video resolution, audio sampling rate, or data compression technology, may be included in a single program. Multiple sets of slides, hypertext links, and other streams of isochronous data types may also be included in a single program. One or more of each data type may be displayed on the computer screen, and users may switch among the different streams of data available in the program.

The present invention is compatible with operating with a collection of many presentations, and to assist users in locating the particular portion of the particular presentation that most interests them. The presentations are stored in a data base of a type well-known in the art, which may range from a simple non-relational data base that stores data in disk files to a complex relational or object-oriented data base that stores data in a specialized format. Referring to the exemplary screen **800** depicted in FIG. **9**, users can issue structured queries or full text queries to identify programs they wish to view. The user types in a query in the query type-in box **810**. The titles of the programs that match the query are displayed in the results box **820**. Structured queries are queries that allow the user to select programs on the basis of structured information associated with each program, such as title, author, or date. Using any of the structured query engines well-known in the art, the user can specify a particular title, author, range of dates, or other structured query, and select only those programs which have associated structured information that matches the query. Full-text queries are queries that allow the user to select programs on the basis of text associated with each program, such as the abstract, transcript, slides, or ancillary materials connected via hypertext. Using any of the full-text search engines known in the art, the user can specify a particular combination of words and phrases, and select only those programs which have associated text that matches the full-text query. Users can also select which of the associated text elements to search. For example, the user can specify to search only the transcript, only the slides, or a combination of both. When the text associated with a program matches the user's query, the user can jump directly to the matched text, and display all of the other synchronized multimedia data types at that point in the program.

Full-text queries can be manually constructed by the users, or they can be automatically constructed by the invention. Such automatically-constructed queries are referred to as "agents." FIG. **10** presents a flow chart of the agent mechanism starting at **900**. When the user displays a program **910**, the system constructs a summary of the program **920**. The summary of the program may be constructed in multiple alternative ways. Each program may have associated with it a list of keywords that describe the major subjects discussed in the program. In this case, constructing the summary simply involves accessing this predefined list of keywords. Alternatively, any text summarization engine well-known in the art may be run across the text associated with program, including the abstract, the transcript, and the slides, to generate a list of keywords that describe the major subjects discussed in the program. This summary is added to the user's profile **930**. The user's profile is a list of keywords that collectively describe the programs that the user has viewed in the past. Each time the user views a new program, the keywords that describe that program are added to the user's profile. In this manner, the agent "learns" which subjects are most interesting to the user, and continues to learn about the user's changing interests as the user uses the system. The agent mechanism also incorporates the concept of memory. Each keyword that is added to the user's profile is labeled with the date at which its associated program was viewed. Whenever the agent mechanism is initiated, the difference between the current date and the date label on each keyword is used to assess the relative importance of that keyword. Keywords that entered the profile more recently are treated as more important than keywords that entered the profile in the distant past. On specified events, such as the user logging into the system, the agents mechanism is initiated **901**. The system creates a query from the current user's profile **940**. The list of keywords in the profile are reorganized into the query syntax required by the full-text search engine. The ages of the keywords are converted into the relative importance measure required by the full-text search engine. The query is run against all of the programs on the server **950**, and the resulting list of programs are presented to the user **960**. This list of programs constitutes the programs which the system has determined may be of interest to the user, based on the user's past viewing behavior.

In addition, users can create their own agents by manually constructing a query that describe their ongoing interest. Each time the agents' mechanism is initiated, the user's manually-constructed agents are executed along with the system's automatically-constructed agent, and the selected programs are presented to the user.

The user can create "virtual conferences" that consist of user-defined aggregations of programs. To create a virtual conference, a user composes and executes a query that selects a set of programs that share a common attribute, such as author, or discuss a common subject. This thematic aggregation of programs can be named, saved, and distributed to other users interested in the same theme.

The user can construct "synthetic programs" by sequencing together segments of programs from multiple different programs. To create a synthetic program, the user composes and executes a query, specifying that the invention should select only those portions of the programs that match the query. The user can then view the concatenated portions of multiple programs in a continuous manner. The synthetic program can be named, saved, and distributed to other users interesting in the synthetic program content.

Referring now to FIG. **11**, which will be used to describe the operation of an embodiment of the present invention across a non-isochronous network connection. This embodiment incorporates a cooperative processing data distribution and caching model that enables the isochronous data streams to play continuously immediately following a navigational event, such as moving to the next slide or searching to a particular word in the transcript.

After the process starts **1000**, when the user first selects a program to play **1001**, the system downloads the selected portions of the non-isochronous data from the server to the client. The downloaded non-isochronous data includes the Slide Index, the Slides, the Transcript Index, the Transcript, and the Hypertext Index. The downloaded non-isochronous data is stored in a disk cache **1010** on the client. The purpose of pre-downloading this non-isochronous data is to avoid having to transmit it over the network connection simultaneously with the transmission of the isochronous data, thereby interrupting the transmission of the isochronous data. The Hypertext Objects are not pre-downloaded to the client; rather, the system is designed to pause the transmission of the isochronous data to accommodate the downloading of any Hypertext Objects. At the end of playing a program, the client disk cache is emptied in preparation for use with another program.

In addition to downloading portions of the non-isochronous data, the system downloads a segment of the isochronous data from the server to a memory cache on the client. The downloaded isochronous data includes the initial segment of the video data and the corresponding initial segment of the audio data. The amount of isochronous data downloaded typically ranges from 5 to 60 seconds, but may be more or less. The downloaded isochronous data is stored in a memory cache **1020** on the client.

When the user presses the Play Button, the Event Handler **1030** receives a Play Program Event **1040**. The system begins the continuous delivery of the isochronous data to the display devices **1041**. Based on the time code of the currently displayed video frame, it also displays the associated non-isochronous data **1042**, including the Transcript, the Slides, and the Hypertext Links. As the system streams the isochronous data to the display devices, it depletes the memory cache. When the amount of isochronous data in the memory cache falls below a specified threshold, the system causes the client CPU to send a request to the server CPU for the next contiguous segment of isochronous data **1043**. This threshold typically works out to be on the order of 5-10 seconds, with a worst-case scenario of 60 seconds. It should be appreciated by one of ordinary skill in art that factors such as network capacity and usage should affect the choice of threshold. Upon receiving this data, the client CPU repopulates the isochronous data memory cache. If, as anticipated, the client CPU experiences a delay in receiving the requested data, caused by the non-isochronous network connection, the client CPU continues to deliver isochronous data remaining in its memory cache in a continuous stream to the display device, until that cache is exhausted.

The method for repopulating the client's memory cache is a critical element in supporting efficient random access into isochronous data streams over a non-isochronous network. The method for downloading the isochronous data from the server to the memory cache on the client is designed to balance two competing requirements. The first requirement is for continuous, uninterrupted delivery of the isochronous data to the video display device and speakers attached to the client CPU. The network connection between the client and server is typically non-isochronous, and may introduce significant delays in the transmission of data from the client to the server. In practice, if the memory cache on the client becomes empty, requiring client to send a request across the network to the server for additional isochronous data, the amount of time needed to send and receive the request will cause the interruption of play of the isochronous data. The requirement for continuous delivery thus encourages the caching of as much data as possible on the client. The second requirement is to minimize the amount of data that is transmitted across the network. In practice, multiple users share a fixed amount of network bandwidth, and transmitting video and audio data across a network consumes a substantial portion of this limited resource. It is anticipated that a common user behavior will be to use the random access navigation capabilities to reposition the program. But the act of repositioning the program invalidates all or part of the data stored in the memory cache in the client. The larger the amount of data that is stored in the memory cache on the client, the more data is wasted upon repositioning the program, and thus the more network bandwidth was wasted in sending this unused data from the server to the client. Thus the requirement for minimizing the amount of data transmitted across the network encourages the caching of as little data as possible on the client.

The present invention balances the need for continuous delivery of isochronous data to the display devices with the need to avoid wasting network bandwidth by implementing a novel cooperative processing data distribution and caching model. The memory cache on the client is designed specifically for compressed isochronous data, and more specifically for compressed digital video data. The caching strategy differs markedly from traditional caching strategies. Traditional caching strategies measure the number of bytes of data in the cache, and repopulate the cache when the number of bytes falls below a specified threshold. By contrast, one embodiment of the present invention measures the number of *seconds* of isochronous data in the memory cache, and repopulates the cache when the number of seconds falls below a specified threshold. Due to the inherent inhomogeneities in video compression, a fixed number of seconds of compressed video data does not correspond to a fixed number of bytes of data. For video data streams that compress into a smaller than average number of bytes per second, the cooperative distribution and caching model reduces the amount of data sent across the network compared to a traditional caching scheme. For video data streams that compress into a larger than average number of bytes per second, the cooperative distribution and caching model guarantees a certain number of seconds of video data cached on the server, reducing the likelihood of interrupted play of the video data stream compared to a traditional caching scheme.

In addition to designing the memory cache to contain a range of a number of seconds of isochronous data, the memory cache employs a policy of *unbalanced look ahead and look behind*. *Look ahead* refers to caching the isochronous data corresponding to "N" seconds into the future. This isochronous data will be delivered to the display device under the normal operation of playing the program. *Look behind* refers to caching the isochronous data corresponding to "M" seconds into the past. This isochronous data will be delivered to the display device under the frequent operation of replaying the previously played few seconds of the program. *Unbalanced* refers to the policy of caching a different amount (that is, a different number of seconds) of look ahead and look behind data. Generally, more look ahead data is cached than look behind data, typically in the approximate ratio of 7:1. It can be appreciated by one of ordinary skill in the art that different caching policies can be employed in anticipation of different common user behaviors. For example, the use of a circular data structure, a structure well-known in the art, may effect this operation.

During program play **1040**, the server sends data to the client at the nominal rate of one second of isochronous data each second. The server adapts to the characteristics of the network, bursting data if the network supports a high burst rate, or steadily transmitting data if the network does not support a high burst rate. The client monitors its memory cache, and sends requests to the server to speed up or slow down. The client also sends requests to the server to stop, restart at a new place in the program, or start playing a different program.

The system administrator can specify how much network bandwidth is available to the system, for each individual program, and collectively across all programs. The system automatically tunes its memory caching scheme to reflect these limits. If the transmitted data would exceed the specified limits, the system automatically drops video frames as necessary.

When the user performs a navigational activity, such as moving to the next slide or searching to a particular word in the transcript, the Event Hander **1030** receives a Navigational Event **1050**. The system computes the time base value of the new position **1051**. It then downloads a new segment of the isochronous data from the server to the memory cache on the client **1052**. The downloaded isochronous data includes a segment of the video data and a corresponding segment of the audio data. The system then displays the video frame corresponding to the current time base value, and the non-isochronous data corresponding to the displayed video frame **1053**.

When the user selects a hypertext link, the Event Handler **1030** receives a Display Hypertext Object Event **1060**. The system pauses the play of the program **1061**. The client CPU requests that the server CPU send the Hypertext Object across the network connection **1062**, and upon receiving the Hypertext Object, causes it to be displayed **1063**.

Referring back to FIG. **1**, the server **130** records the actions of each user, including not only which programs each user viewed, but also which portions of the programs each user viewed. This record can be used for usage analysis, billing, or report generation. The user can ask the server **130** for a usage summary, which contains an historical record of that particular user's usage. A manager or system administrator can ask the server **130** for a summary across some or all users, thereby developing an understanding of the patterns of usage. One might use any of the data mining tools as is known in the art for assisting in this purpose.

The usage record may serve as a guide to restructure old programs or to structure new ones, having learned what works from a presentation perspective and what does not, for example. The usage record furthermore enables the system to notify users of changing data. The list of users who have viewed a program can be determined from the usage records. If a program is updated, the system reviews the usage record to determine which users have viewed the program, and notifies them that the program that they previously viewed has changed.

While the present invention has been described in terms of a few embodiments, the disclosure of the particular embodiment disclosed herein is for the purposes of teaching the present invention and should not be construed to limit the scope of the present invention which is solely defined by the scope and spirit of the appended claims.

Having thus described our invention, what we claim as new and desire to secure by Letters Patent is as follows:

1. A method of manipulating a plurality of streams of isochronous and non-isochronous digital data comprising the steps of:

synchronizing the plurality of streams of isochronous and non-isochronous data by reference to a common time base;

navigating to a position in any one of the plurality of streams using at least one of a sequential and a random access approach available for and adapted to the structure and contents of that stream,

identifying positions for each of the plurality of streams corresponding to the position in the navigated stream; and

simultaneously displaying at least some of the plurality of streams at the positions corresponding to the position in the navigated stream.

2. The method of claim 1, further comprising the step of delivering the plurality of streams of synchronized isochronous and non-isochronous data from a server to a client over a non-isochronous network.

3. The method of claim 1, further comprising the step of caching isochronous data on the client, and modulating the delivery of the isochronous data over the network in a manner that maintains a predetermined range of time's worth of data cached on the client.

4. The method of claim 1, further comprising the step of translating the transcript stream into one or more foreign languages, and including a plurality of such transcripts, each synchronized to a common time base and each independently navigable.

5. A system for interacting with a computerized presentation comprising:

a plurality of isochronous and non-isochronous data streams, wherein each of the plurality of streams are synchronized together by reference to a common time base;

for each of the plurality of data streams, means for at least one of sequential and random access navigation of such data stream, and means for display of such data stream; and

identification means, coupled to each of the navigation means, wherein, given a position in one of the plurality of data streams as pointed to by its associated navigation means, the identification means provides, via the common time base, the corresponding positions in the other of the plurality of data streams.

6. The system of claim 5 further comprising:

a server for storing the plurality of isochronous and non-isochronous data streams;

a client for containing the display and the access navigation means of such data streams; and

a non-isochronous network for delivery of such data streams from the server to the client device;

the client further including a data cache and a modulation means both coupled to the network, wherein one or more of the data streams delivered by the network are stored in the data cache, and further wherein the modulation means maintains a predetermined range of time's worth of data within the data cache.

7. The system of claim 5, wherein one of more of the digital data streams corresponds to a speaker giving an informational or educational presentation.

8. The system of claim 5, wherein at least one of the isochronous data streams includes digital video.

9. The system of claim 5, wherein at least one of the isochronous data streams includes digital audio.

10. The system of claim 5, wherein at least one of the non-isochronous data streams includes slides.

11. The system of claim 5, wherein at least one of the non-isochronous data streams includes hypertext links to related data objects.

12. The system of claim 5, wherein at least one of the non-isochronous data streams includes an outline of the presentation.

13. The system of claim 5, wherein at least one of the non-isochronous data streams includes a transcript of spoken words in the presentation.

14. The system of claim 13, wherein the random access navigation means corresponding to the transcript further includes a full-text search engine.

15. The system of claim 14, further comprising:

a plurality of computerized presentations which may be selected by a user, at least some of the presentations including one or more keywords associated therewith; and

a profiling means which maintains a user profile on each user, the user profile including an aggregation of at least some of the keywords of the presentations selected by the user.

16. A system for interacting with a computerized presentation comprising:

a plurality of isochronous and non-isochronous data streams;

two or more sets of conceptual events, each set indexed into one of the plurality of data streams;

for each of the plurality of data streams, means for navigation and display of such data stream, and for those data streams having a set of conceptual events, the means for navigation including a means for selection of a conceptual event;

an identification means, coupled to each navigation and each display means, wherein, given a selected conceptual event, provides the positions in each of the plurality of data streams corresponding to the event.

17. The system of claim **16** wherein a first set of conceptual events is indexed into an isochronous data stream and a second set of conceptual events is indexed into a non-isochronous data stream.

18. The system of claim **16** further comprising a bookmarking means for ad hoc creation of conceptual events.

FIG. 1

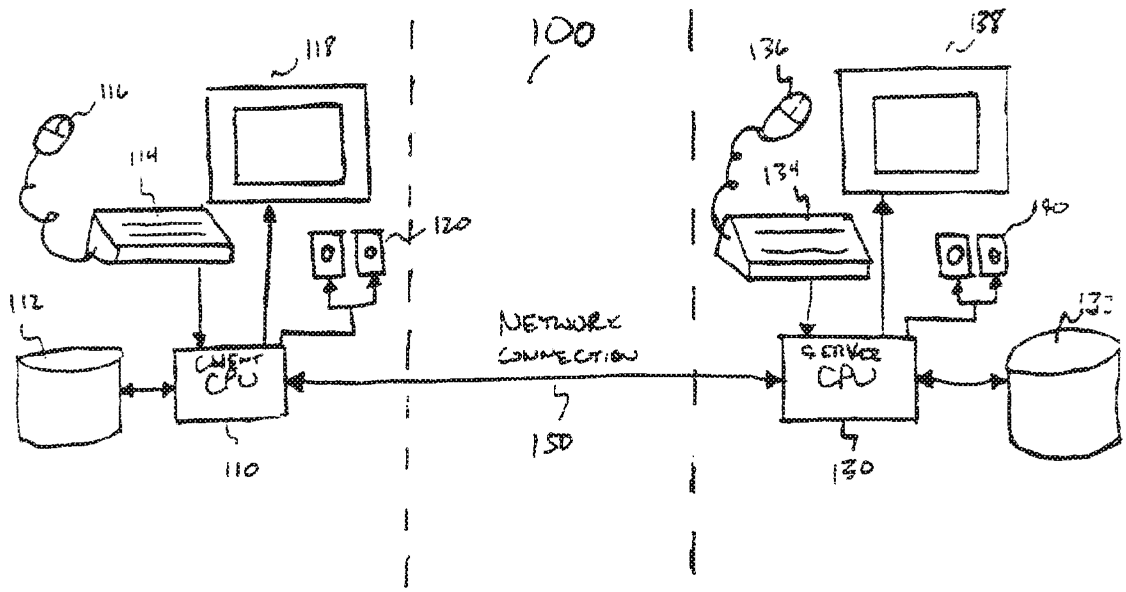

100
116
118
114
120
112
CLIENT CPU
110
NETWORK CONNECTION
150
136
138
134
140
SERVER CPU
130

FIG. 2

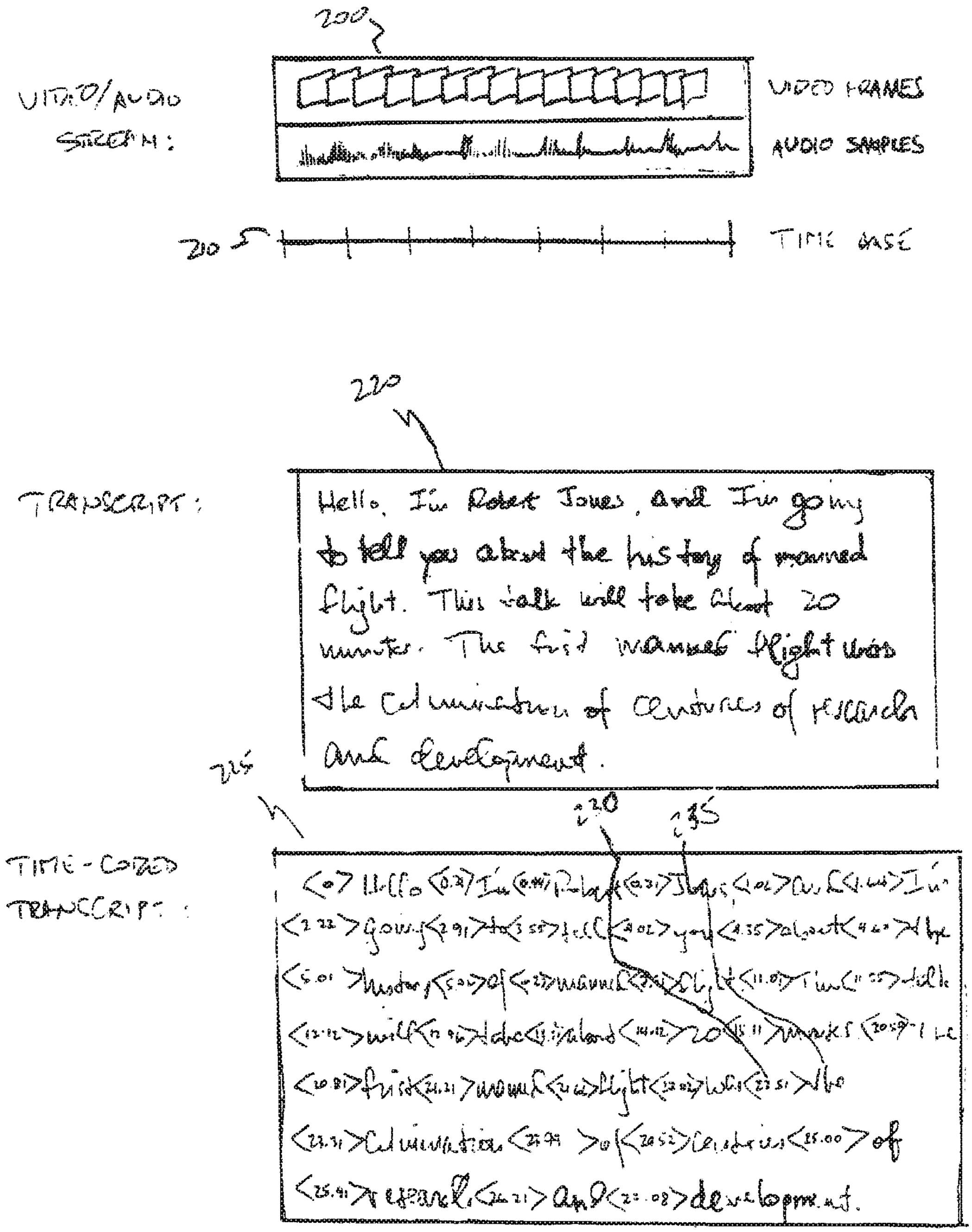
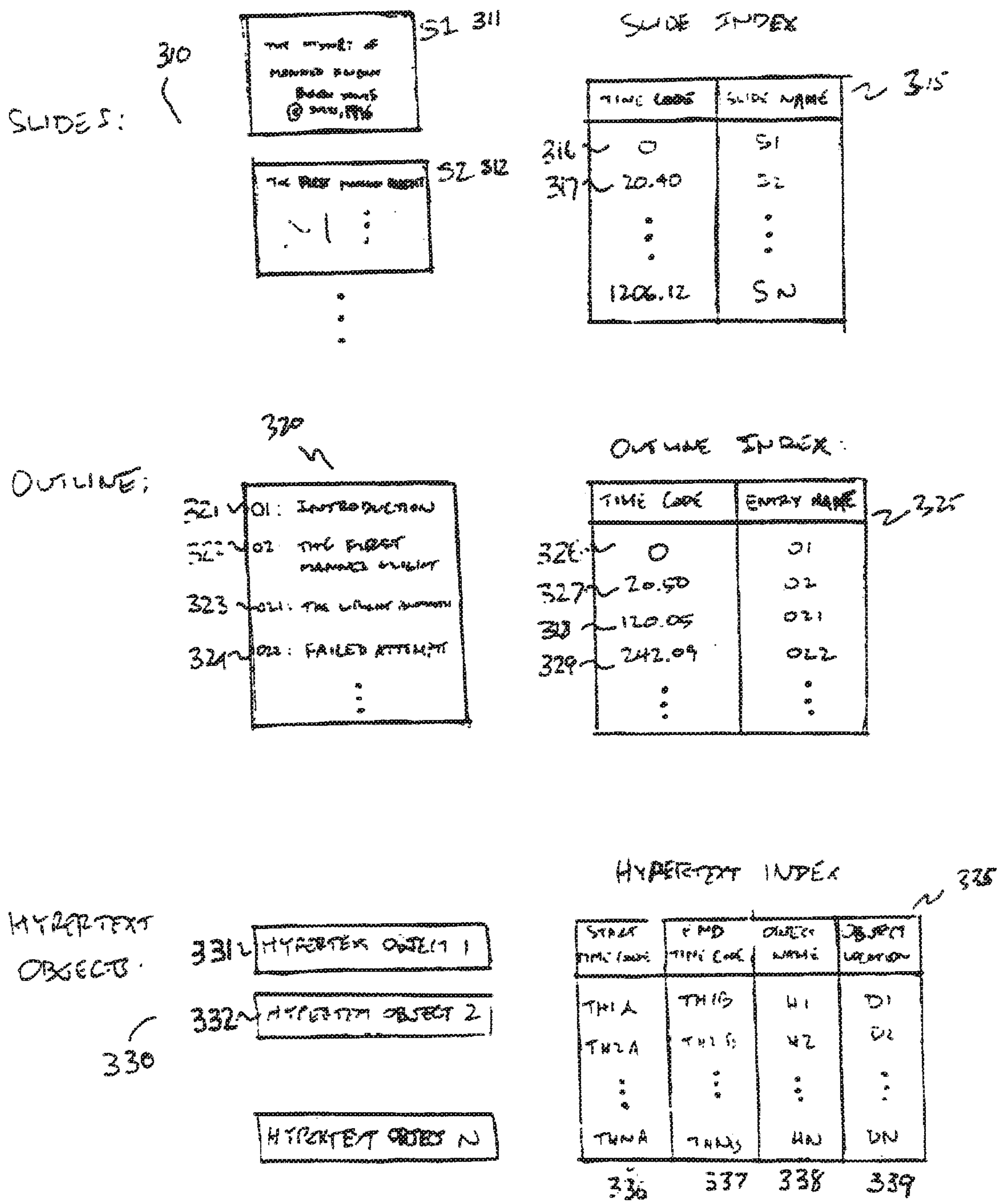

200
VIDEO/AUDIO STREAM:
VIDEO FRAMES
AUDIO SAMPLES
210
TIME BASE
220
TRANSCRIPT:
Hello, I'm Robert Jones, and I'm going to tell you about the history of manned flight. This talk will take about 20 minutes. The first manned flight was the culmination of centuries of research and development.
225
230
235
TIME-CODED TRANSCRIPT:

FIG. 3


SLIDES:
310
S1 311
S2 312
SLIDE INDEX
315
TIME CODE
SLIDE NAME
316
317
0
20.40
1206.12
S1
S2
SN
OUTLINE:
320
01: INTRODUCTION
FAILED ATTEMPT
OUTLINE INDEX:
325
TIME CODE
ENTRY NAME
321
322
323
324
326
327
329
0
20.50
120.05
242.09
01
02
021
022
HYPERTEXT OBJECTS.
330
331
332
HYPERTEXT OBJECT 2
HYPERTEXT OBJECT N
HYPERTEXT INDEX
335
START TIME CODE
END TIME CODE
OBJECT NAME
OBJECT LOCATION
TH1A
TH2A
THNA
H1
H2
HN
D1
DN
336
337
338
339

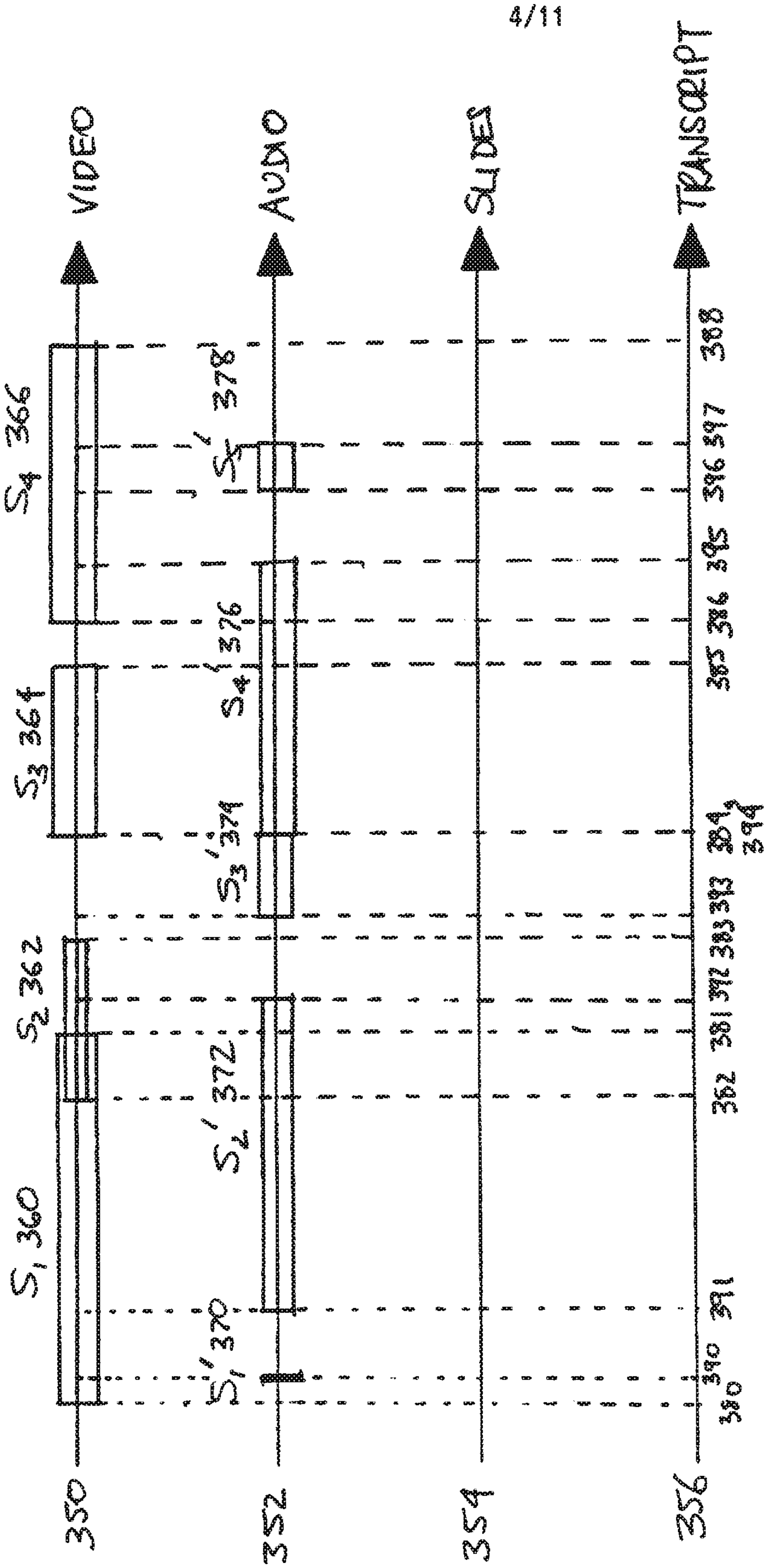

VIDEO
AUDIO
SLIDES
TRANSCRIPT
350
352
354
356
S1 360
S2 362
S3 364
S4 366
S1' 370
S2' 372
S3' 374
S4' 376
S5' 378

FIG. 4

FIG. 5

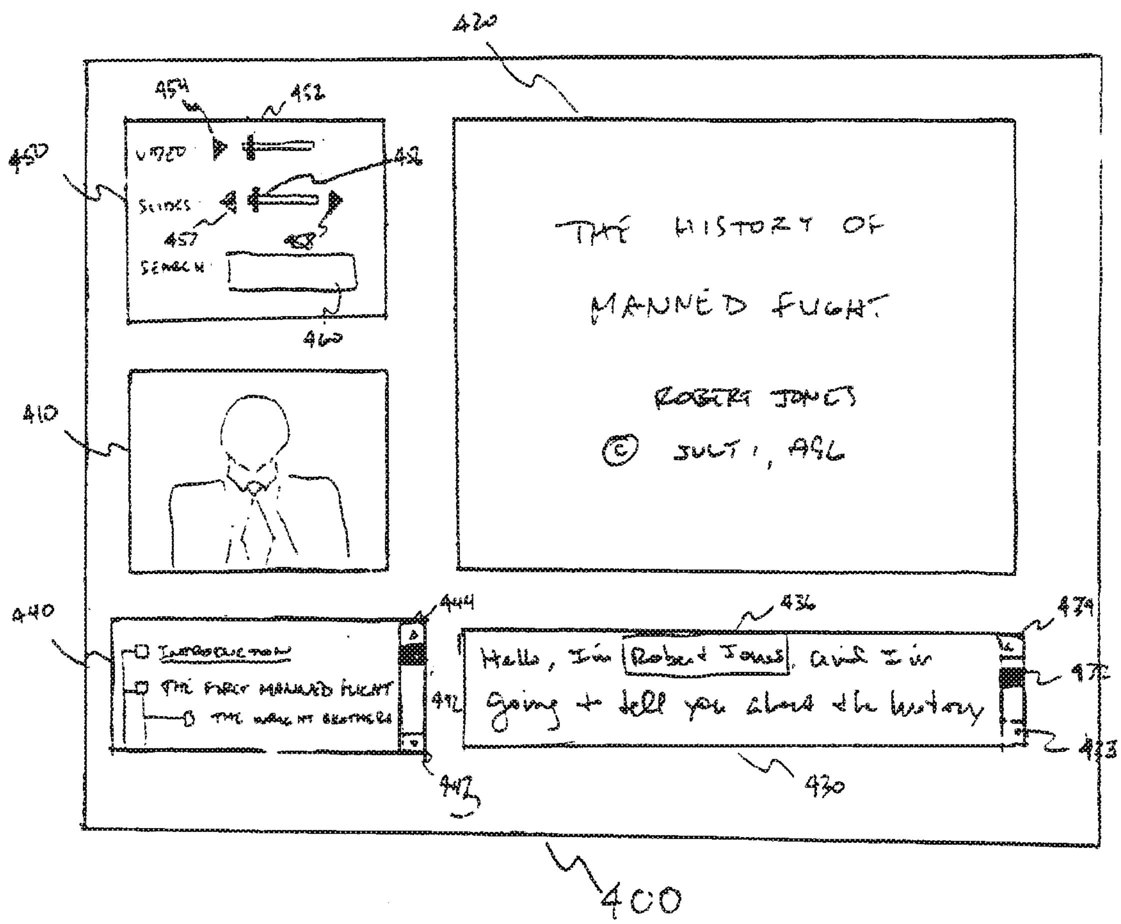

420
454
452
450
VIDEO
SLIDES
SEARCH
457
460
THE HISTORY OF
MANNED FLIGHT
ROBERT JONES
410
440
444
INTRODUCTION
THE FIRST MANNED FLIGHT
THE WRIGHT BROTHERS
436
Hello, I'm Robert Jones, and I'm going to tell you about the history
430
400

FIG. 6

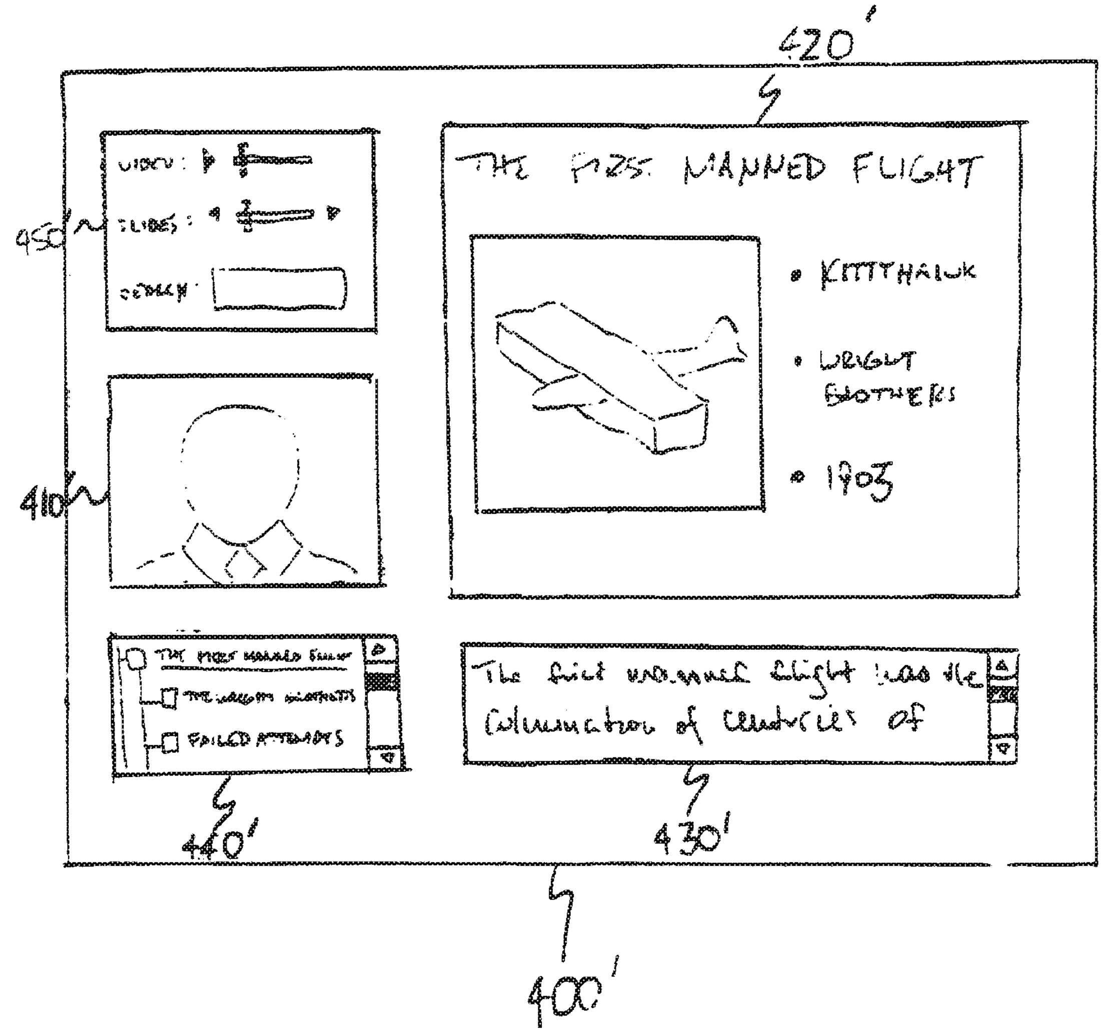

420'
THE FIRST MANNED FLIGHT
KITTYHAWK
WRIGHT BROTHERS
1903
450'
410'
440'
430'
400'
FAILED ATTEMPTS

FIG 7

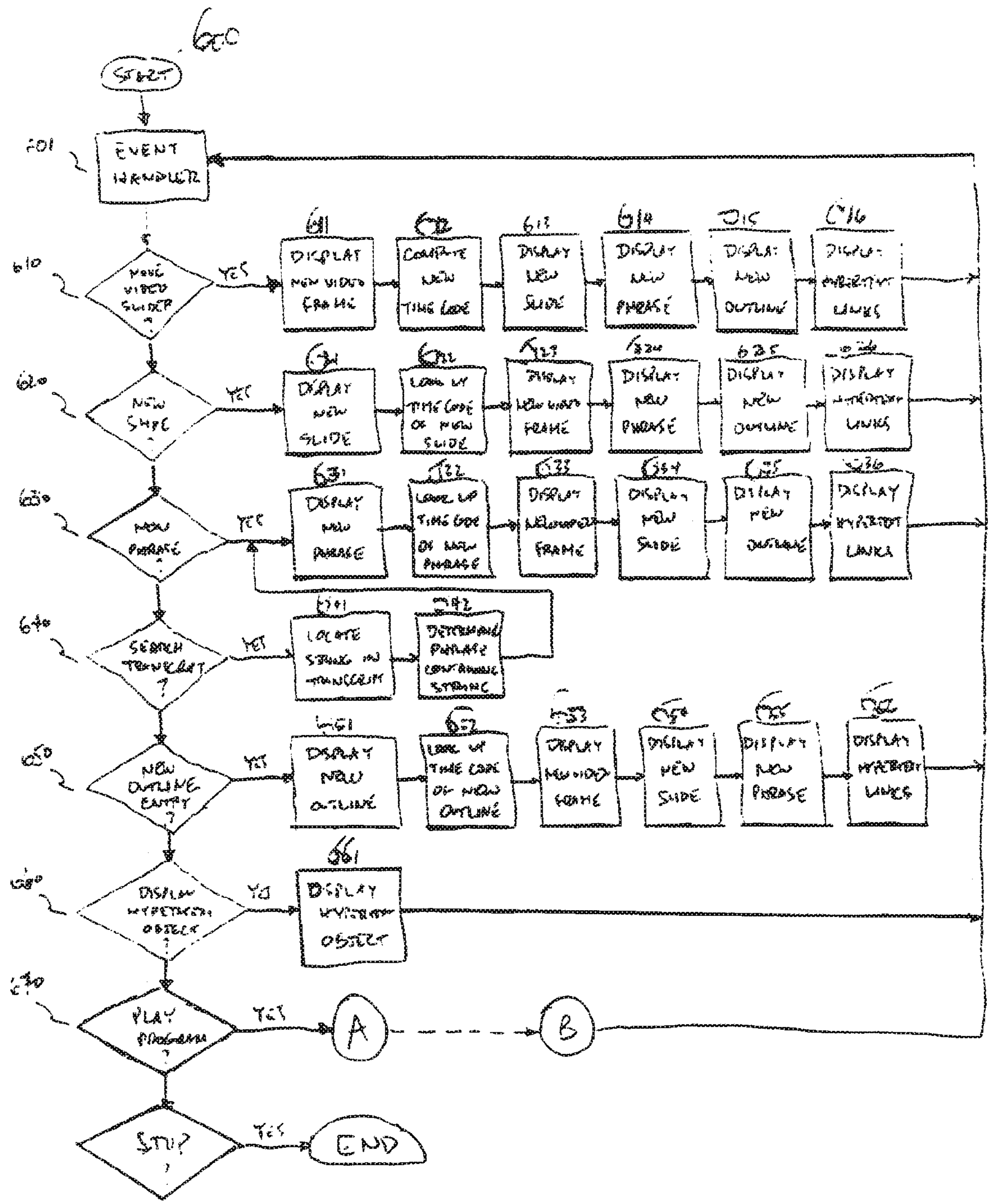

600
START
601
EVENT HANDLER
610
620
NEW SLIDE
630
NEW PHRASE
640
SEARCH TRANSCRIPT ?
650
NEW OUTLINE ENTRY ?
DISPLAY HYPERTEXT OBJECT ?
670
PLAY PROGRAM ?
STOP ?
YES
DISPLAY NEW VIDEO FRAME
COMPUTE NEW TIME CODE
DISPLAY NEW SLIDE
DISPLAY NEW PHRASE
DISPLAY NEW OUTLINE
DISPLAY HYPERTEXT LINKS
LOOK UP TIME CODE OF NEW SLIDE
LOOK UP TIME CODE OF NEW PHRASE
LOCATE STRING IN TRANSCRIPT
DETERMINE PHRASE CONTAINING STRING
LOOK UP TIME CODE OF NEW OUTLINE
DISPLAY HYPERTEXT OBJECT
A
B
END

FIG. 8

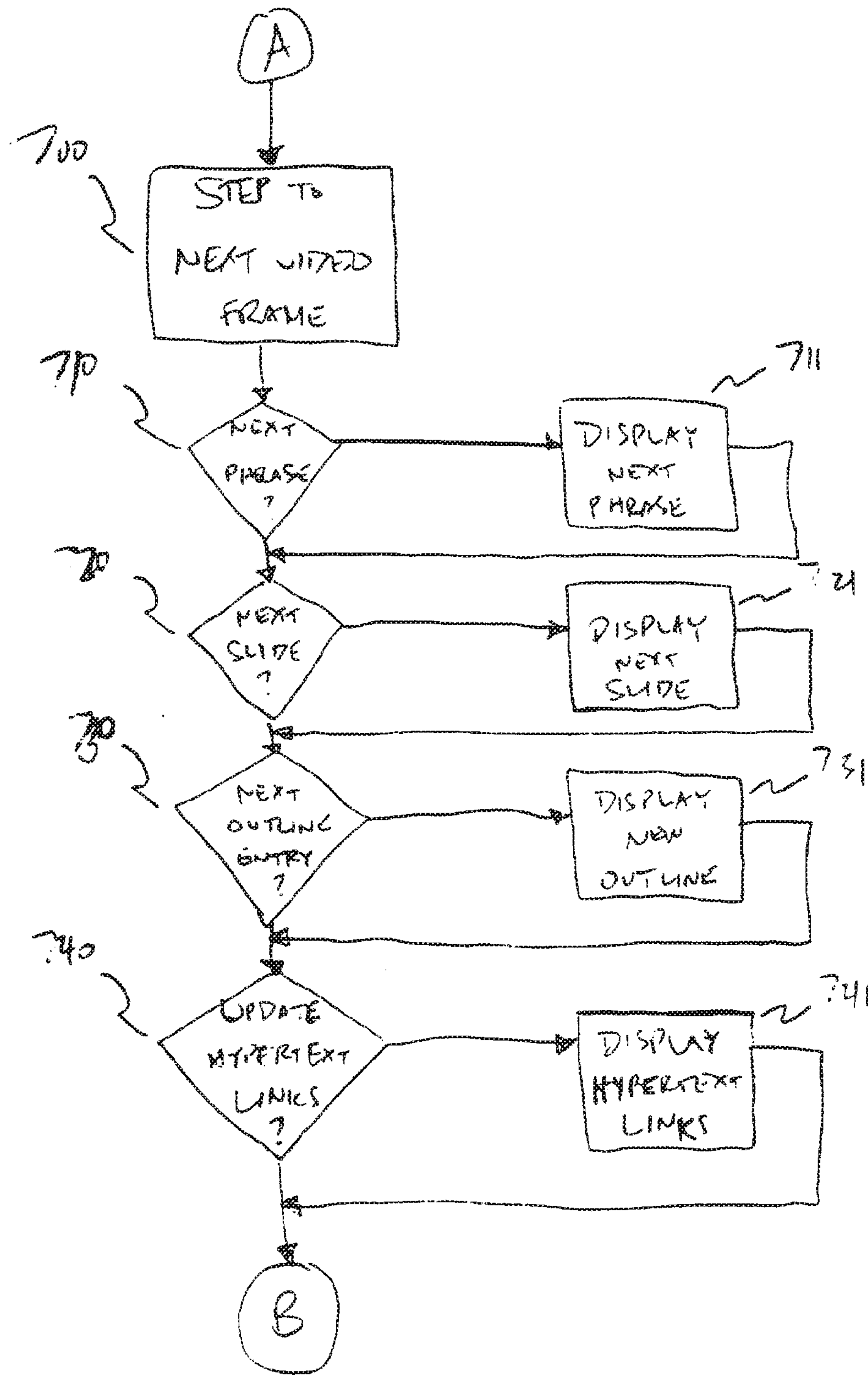

A
700
STEP TO NEXT VIDEO FRAME
710
NEXT PHRASE ?
711
DISPLAY NEXT PHRASE
720
NEXT SLIDE ?
721
DISPLAY NEXT SLIDE
730
NEXT OUTLINE ENTRY ?
731
DISPLAY NEW OUTLINE
740
UPDATE HYPERTEXT LINKS ?
741
DISPLAY HYPERTEXT LINKS
B

FIG. 9

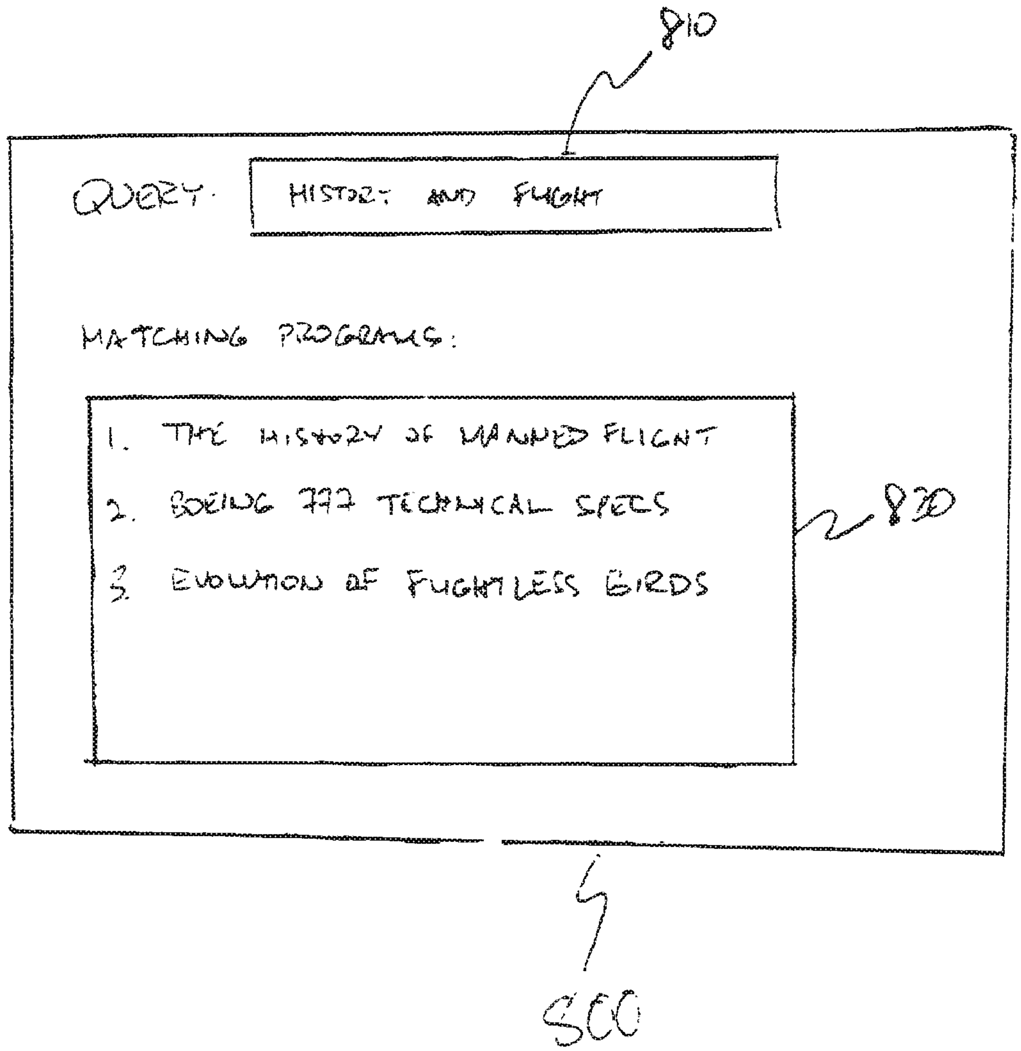

810
QUERY:
HISTORY AND FLIGHT
MATCHING PROGRAMS:
1. THE HISTORY OF MANNED FLIGHT
2. BOEING 777 TECHNICAL SPECS
3. EVOLUTION OF FLIGHTLESS BIRDS
820
800

FIG. 10

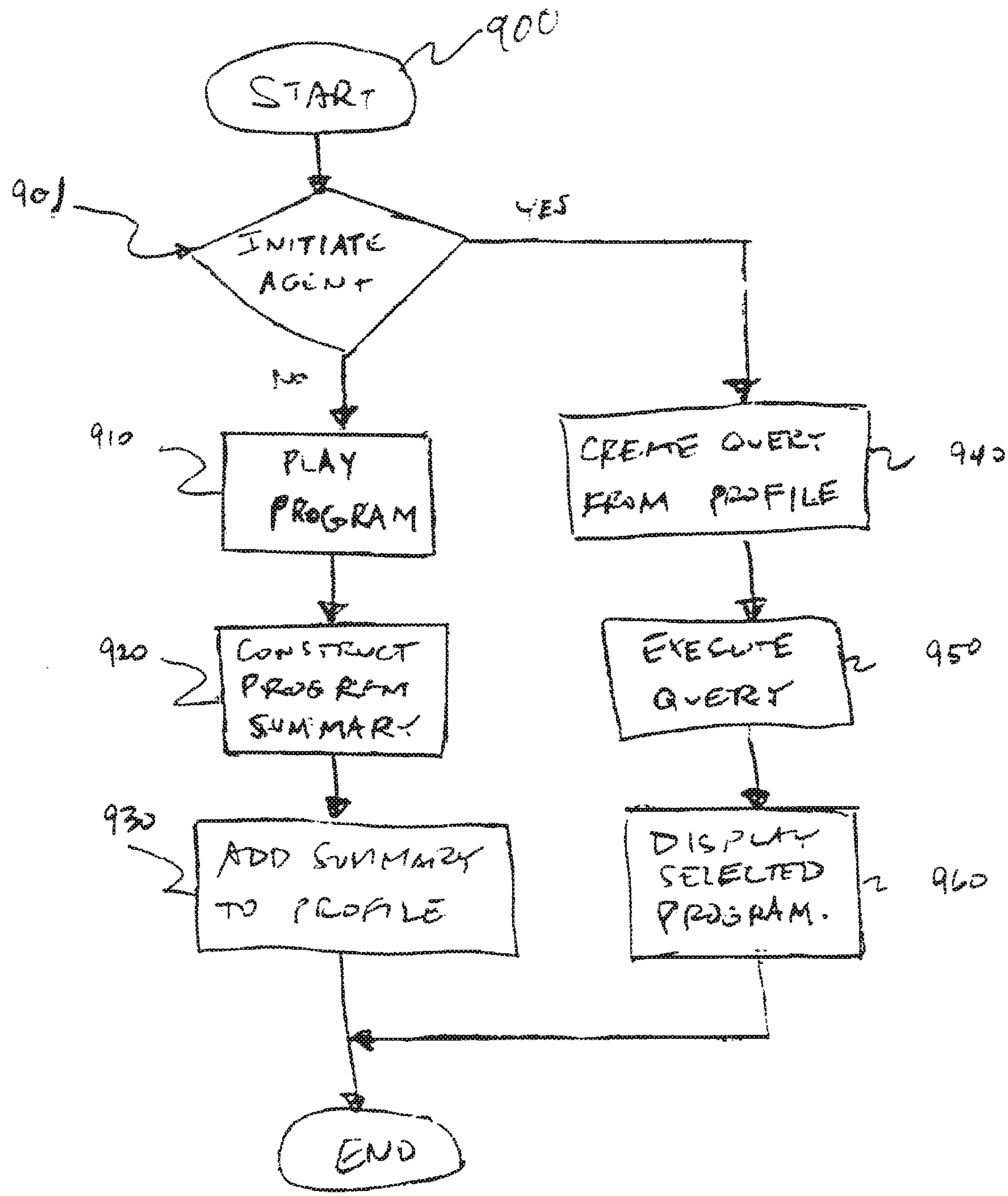
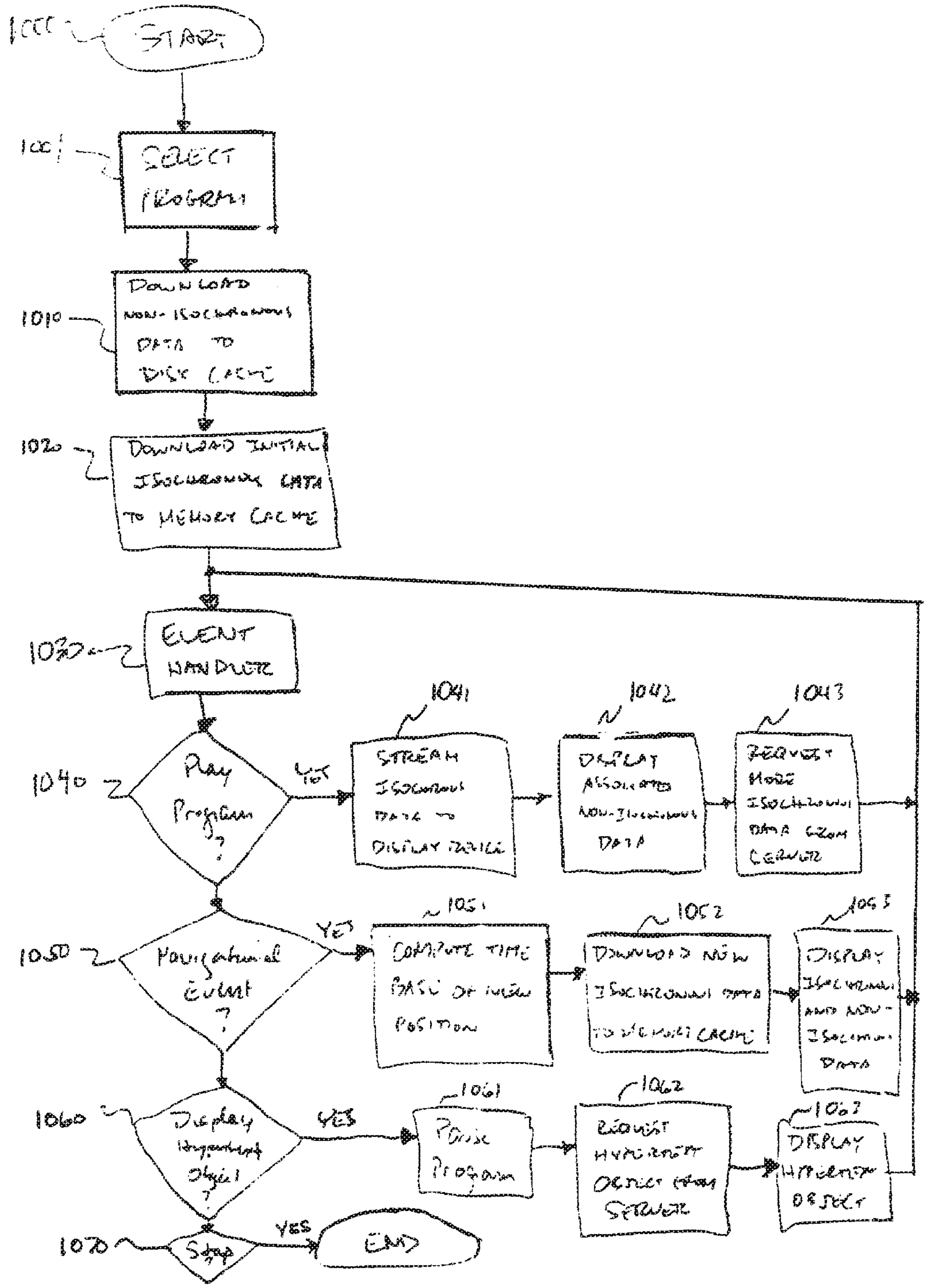

900
START
901
INITIATE AGENT
YES
NO
910
PLAY PROGRAM
920
CONSTRUCT PROGRAM SUMMARY
930
ADD SUMMARY TO PROFILE
CREATE QUERY FROM PROFILE
940
EXECUTE QUERY
950
DISPLAY SELECTED PROGRAM.
960
END

FIG. 11


1000
START
1005
SELECT PROGRAM
1010
DOWNLOAD NON-ISOCHRONOUS DATA TO DISK CACHE
1020
DOWNLOAD INITIAL ISOCHRONOUS DATA TO MEMORY CACHE
1030
EVENT HANDLER
1040
Play Program ?
YES
1041
STREAM ISOCHRONOUS DATA TO DISPLAY DEVICE
1042
DISPLAY ASSOCIATED NON-ISOCHRONOUS DATA
1043
REQUEST MORE ISOCHRONOUS DATA FROM SERVER
1050
Navigational Event ?
YES
1051
COMPUTE TIME BASE OF NEW POSITION
1052
DOWNLOAD NEW ISOCHRONOUS DATA TO MEMORY CACHE
1053
DISPLAY ISOCHRONOUS AND NON-ISOCHRONOUS DATA
1060
Display Hypertext Object ?
YES
1061
Pause Program
1062
REQUEST HYPERTEXT OBJECT FROM SERVER
1063
DISPLAY HYPERTEXT OBJECT
1070
Stop
YES
END

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US97/06982

A. CLASSIFICATION OF SUBJECT MATTER
IPC(6) :G06F 3/00, 3/14, 13/00, 13/42
US CL : 395/602, 603, 607, 807, 881, 200.02; 348/482, 515; 370/352
According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
U.S. : 395/602, 603, 607, 807, 881, 200.02; 348/482, 515; 370/352

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)
APS, DIALOG

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X,P	US 5,613,909 A (STELOVSKY) 25 March 1997, Figs. 2,4-6,10,12; col. 3, lines 21-60; col. 14, lines 17-67.	1-18
Y	SoftCom LearningNet Multimedia Distance Learning. http://www.softcom.com/Learningnet.html, 1995, Softcom. Inc., see entire document.	1-16
Y	The Re:Viewer Workstation Revolutionary search, retrieval, and organization of litigation discovery data, Produci News Law Technology. vol. 2, no. 6, June 1995, see entire document.	1-16

[X] Further documents are listed in the continuation of Box C. [ ] See patent family annex.

* Special categories of cited documents:

"A" document defining the general state of the art which is not considered to be of particular relevance
"E" earlier document published on or after the international filing date
"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)
"O" document referring to an oral disclosure, use, exhibition or other means
"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
"&" document member of the same patent family

Date of the actual completion of the international search
24 JULY 1997

Date of mailing of the international search report
05 SEP 1997

Name and mailing address of the ISA/US
Commissioner of Patents and Trademarks
Box PCT
Washington, D.C. 20231
Facsimile No. (703) 305-3230

Authorized officer
MENG-AI AN
Telephone No. (703) 305-9678

Form PCT/ISA/210 (second sheet)(July 1992)*

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US97/06982

C (Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	RAVINDRAN K. ET AL, Delay Compensation Protocols for Synchronization of Multimedia Data Streams; IEEE transactions on Knowledge and Data Engineering. August 1993. Vol. 5. No. 4. pages 574-589, especially pages 574, and 588.	1-2,5,8-10,16-17
Y,E	US 5,630,117 A (OREN ET AL) 13 May 1997, col. 1, lines 62-64; col. 2, lines 50-55.	1,5,11,16
Y,E	US 5,642,171 A (BAUMGARTNER ET AL) 24 June 1997, Abstract, Fig. 6.	1,5,16
Y	US 5,471,576 A (YEE) 28 November 1995, Abstract, Fig. 2.	1,5,16
Y	US 5,274,758 A (BEITEL ET AL) 28 December 1993, Figs. 10, 14, 43.	1-16
Y	US 5,101,274 A (YOSHIMURA ET AL) 31 March 1992, Abstract	1,5,16
A,P	US 5,619,733 A (NOE ET AL) 08 April 1997, Abstract	1-18
A	US 4,761,781 A (CALVIGNAC ET AL) 02 August 1988, Abstract	1-18

Form PCT/ISA/210 (continuation of second sheet)(July 1992)*

PCT

WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 6 : H04L 29/06 | A2

(11) International Publication Number: WO 97/44942

(43) International Publication Date: 27 November 1997 (27.11.97)

(21) International Application Number: PCT/US97/08679

(22) International Filing Date: 22 May 1997 (22.05.97)

(30) Priority Data:
60/018,256 24 May 1996 (24.05.96) US

(71) Applicant: NARRATIVE COMMUNICATIONS CORP. [US/US]; 204 Second Avenue, Waltham, MA 02154 (US).

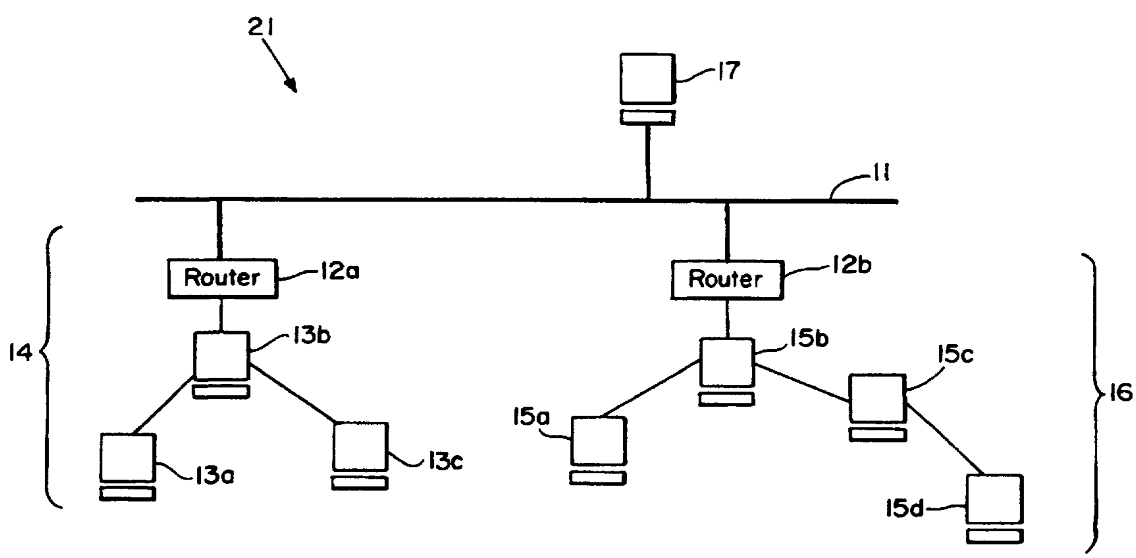
(72) Inventors: KLIGER, Scott, A.; 18 Jacob Amsden Road, Westborough, MA 02581 (US). MIDDLETON, Thomas, M., III; 25 Burditt Avenue, Hingham, MA 02043 (US). WHITE, Gregory, T.; 31 Old Billerica Road, Bedford, MA 01730 (US).

(74) Agents: WAKIMURA, Mary, Lou et al.; Hamilton, Brook, Smith & Reynolds, P.C., Two Militia Drive, Lexington, MA 02173 (US).

(81) Designated States: AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, CA, CH, CN, CU, CZ, DE, DK, EE, ES, FI, GB, GE, GH, HU, IL, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MD, MG, MK, MN, MW, MX, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, TJ, TM, TR, TT, UA, UG, UZ, VN, YU, ARIPO patent (GH, KE, LS, MW, SD, SZ, UG), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, ML, MR, NE, SN, TD, TG).

Published
Without international search report and to be republished upon receipt of that report.

(54) Title: COMPUTER METHOD AND APPARATUS FOR OBJECT STREAMING



(57) Abstract

In a distributed computing environment, a data stream is formed of a sequence of requested objects. The defined order of the sequence of objects is determined from a client request for data. The order may be a default order, or, alternatively, the server may track client criteria to determine the order. For example, the server (17) may track objects previously transmitted in the stream to the client (13) such that there is no duplication of objects. In other instances, the server may select an object from a class of objects, depending upon object quality, bandwidth, client location, and other client-specific criteria. The server compiles and transmits the object data stream in real-time (on-the-fly) based on the criteria. Buffering of data with pausing to rectify buffer debt is provided by the client.

FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AL	Albania
AM	Armenia
AT	Austria
AU	Australia
AZ	Azerbaijan
BA	Bosnia and Herzegovina
BB	Barbados
BE	Belgium
BF	Burkina Faso
BG	Bulgaria
BJ	Benin
BR	Brazil
BY	Belarus
CA	Canada
CF	Central African Republic
CG	Congo
CH	Switzerland
CI	Côte d'Ivoire
CM	Cameroon
CN	China
CU	Cuba
CZ	Czech Republic
DE	Germany
DK	Denmark
EE	Estonia
ES	Spain
FI	Finland
FR	France
GA	Gabon
GB	United Kingdom
GE	Georgia
GH	Ghana
GN	Guinea
GR	Greece
HU	Hungary
IE	Ireland
IL	Israel
IS	Iceland
IT	Italy
JP	Japan
KE	Kenya
KG	Kyrgyzstan
KP	Democratic People's Republic of Korea
KR	Republic of Korea
KZ	Kazakstan
LC	Saint Lucia
LI	Liechtenstein
LK	Sri Lanka
LR	Liberia
LS	Lesotho
LT	Lithuania
LU	Luxembourg
LV	Latvia
MC	Monaco
MD	Republic of Moldova
MG	Madagascar
MK	The former Yugoslav Republic of Macedonia
ML	Mali
MN	Mongolia
MR	Mauritania
MW	Malawi
MX	Mexico
NE	Niger
NL	Netherlands
NO	Norway
NZ	New Zealand
PL	Poland
PT	Portugal
RO	Romania
RU	Russian Federation
SD	Sudan
SE	Sweden
SG	Singapore
SI	Slovenia
SK	Slovakia
SN	Senegal
SZ	Swaziland
TD	Chad
TG	Togo
TJ	Tajikistan
TM	Turkmenistan
TR	Turkey
TT	Trinidad and Tobago
UA	Ukraine
UG	Uganda
US	United States of America
UZ	Uzbekistan
VN	Viet Nam
YU	Yugoslavia
ZW	Zimbabwe

COMPUTER METHOD AND APPARATUS FOR OBJECT STREAMING

REFERENCE TO CO-PENDING APPLICATION

This application claims the benefit of a United States Provisional Application Serial No. 60/018,256 filed May 24, 1996.

BACKGROUND

"Distributed computing" makes use of a computer network formed out of one or more computers loosely coupled together to allow processes on different computers to communicate with each other and to provide services for each other. One of the most common paradigms of distributed computing is known as the "client-server model", in which consumers of services are called "clients", and make requests of service providers, called "servers".

In object oriented distributed computing, there is a notion of computer entities called "objects". Each object comprises a particular state and a set of defined behaviors. The state is represented by data maintained by the object. The behavior is specified in terms of operations that the object can perform with the operations, typically realized by executable code. Conceptually, the data and the code are inextricably bound together in the object. Objects may be "persistent", that is, they may continue to exist even though they are inactive or the computer on which they exist has failed or has been turned off. Further, objects may issue requests for services to other objects as well as supply services.

Typically, data is held in linear files on a server. When a client requests that data or a part thereof, a connection is formed between the data source (server) and delivery (client) point.

In the prior art there are in general two different types of servers. The first, known as a web server, typically stores data files of a number of different types. Web servers typically communicate with clients over a network such as the Internet using the well known TCP/IP protocol. The second type of server, known as a streaming media server, stores and transmits media files of various types.

More particularly, the web servers presently in use typically store data files in a format known as Hyper Text Markup Language(HTML). HTML permits the web servers to handle container files which reference other files of varying formats. Using HTML, a given web document may include content information in various formats and may also refer to other files by including reference information known as a Uniform Reference Locator (URL). URL's specify the location of remote servers at which files referenced in the HTML file may be located.

Upon receipt of an HTML file from the original web server, a client then must access each document referenced from its source. Each such request typically requires a full cycle of communication with a remote server, including opening a connection socket with the remote server, requesting that the file be transferred, waiting for the file to download, closing the connection, and then, finally parsing the file. To render a given web page may therefore require many such cycles.

The other type of server, known as a streaming media server, has been developed to be particularly suited for multimedia of various types. Such servers may handle single data types, such as a RealAudio™ file, or may include mixed media types, in formats such as NetShow™ (RealAudio™ is a trademark of Progressive Networks, Inc., and NetShow™ is a trademark of Microsoft Corporation). In any event, media files are typically laid out in a linear fashion in a single file. Thus, when the client requests a file from a streaming server, a socket is simply opened and delivery of data is begun.

The client may perform a caching or buffering operation prior to actual play back of the media file. This ensures that the media file is played back to the user of the client computer in a continuous stream. In particular, the client may calculate in advance an amount of data that it must have on hand prior to actually beginning to render the media file, so that the user has an impression of continuous delivery of the media.

In such a linear streaming server, files may be formatted in advance with a specific communication transfer bandwidth in mind. For example, a Real Audio file may have been compressed for receipt at a baud rate such as 14.4 kilo bits per second (kbps). Another file would be made available for optimum playback at 28.8 kbps. These different file formats provide for allowances in playing back data such that it is rendered in a continuous fashion at the respective rates.

In streaming media server, the connection remains open with the server during the full duration of the play back of the file. Thus, for example, even on a high speed network connection such as a T1 line, if the media file is a ten minute audio file, then the connection will remain open for ten minutes, even though the available information transfer rate on a T1 line is much greater than the audio bandwidth.

In addition, one other disadvantage of streaming media servers is that they typically implement a lossy type of compression algorithm. Thus, if network traffic increases after file download has begun, bits may be dropped and the quality of the presentation is adversely affected.

Therefore, with streaming media files, the content must typically be specific to each type of client at the targeted bit rate. In addition, the streaming media server is occupied for the real time duration of the media clip, and the presentation may experience degradation based upon the amount of latency in the network.

SUMMARY OF THE INVENTION

Briefly, in the present invention, rather than interpreting container objects that contain references to other objects that must be retrieved by the client opening multiple connections with various servers, and rather than specifying the streaming of data in a single access request, the present invention provides for transmission of data as a stream of objects. In particular, in response to a user or application request for a file, the client issues a request for the file in the form of a sequence of desired objects. The request is presented to the server as a single request that includes a list of multiple objects to be returned to the client.

On the server side, the request is pulled together according to what the client has requested. The requested objects are then sent together in a single stream to the client. To accomplish this, the server analyzes the request to locate the particular objects.

Objects that are available locally to the server are simply added to the outgoing stream, however, the server may also need to query back end file servers and other sources for objects that may be located at other computers in the network. The server then assembles these objects together and provides them to the client in a single object stream.

In one implementation of the invention, typically the default implementation, the server may analyze the reqeust and assemble objects based upon a predefined order, such as specified by an object map file located at the server.

In another implementation of the invention, the server may assemble objects "on-the-fly", based upon client-specific criteria. The present invention thus also permits the mixture of different objects in an object stream, depending upon the particular client or client request. The assembly of object streams may thus occur dynamically based upon any number of client-specific criteria, such as the objects already available to the client, the communication channel bandwidth, the desired presentation quality, client buffer capability, or other parameters associated with the client or the communications channel.

For example, the server may maintain a log of objects already sent to the client, and send only those objects which are not already available at the client computer.

The client may also specify an object class together with information that enables the server to determine which member object of the class is to be placed in the stream.

Such information may include the communication bandwidth, graphical resolution, physical location, or other information which varies from client to client.

The server may also send objects of a particular quality targeted to particular clients. The selection of objects may depend upon client parameters such as observed network latency in real time or desired object quality.

The system may also include buffering, so that the rendering of the set of objects may be delayed until a sufficient amount of data is received at the client.

BRIEF DESCRIPTIONS OF THE DRAWINGS

The foregoing and other objects, features and advantages of the invention will be apparent from the following more particular description of preferred embodiments and the drawings in which like reference characters refer to the same parts throughout the different views. The drawings are not necessarily to scale, emphasis instead being placed upon illustrating the principles of the invention.

Fig. 1 is a block diagram of a computer network employing the present invention.

Fig. 2 is a schematic flow diagram of one embodiment of the present invention.

Fig. 3 illustrates how a server dynamically assembles an object stream in response to a client request.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENT

Illustrated in Fig. 1 is a block diagram of a general computer network 21. A plurality of computers 13, 15, 17 are coupled across a communication channel 11 for communication amongst each other. Various subsets of the computers may themselves form a local area network (LAN) or other local network 13, 15. Each of the various local networks are coupled through a respective router 12a, 12b to channel 11. This enables communication from one local network to another across the channel 11 to form what is known as an "internet". In the preferred embodiment, the present invention is employed on what has become known as the Internet (an international computer network linking an estimated 35 million people to approximately 4.8 million host computers or information sources).

In the preferred embodiment the computers 13, 15, 17 or digital processors employing the present invention are of the PC or mini computer type, or the like, having processing capabilities of the Intel XX386 processing chip or better. The communication channel 11 is a typical telephone line or other transmission/communication cable handling a 28,800 baud data rate or the like.

A sequence of steps are undertaken by the computers 13, 15, 17 to transfer data in the form of a stream of objects according to the present invention. For example, a given client computer 13a may issue a request for computer data to another computer 17, which acts as a server.

Referring to Fig. 2, the client 13a is a computer that executes an application (or other user interaction) for which certain data needs to be made present. In steps 100 and 102, the client receives an object stream request from the application that includes a global identification of an object map which indicates certain objects existing in the network 21 that the client application program seeks. In particular, the object stream request includes a list, or linear sequence of objects identified a global object identification number. The client 13a transmits the initial request to the server 17, which enters state 200 to transfer back a global identification of the object map listing subject objects.

The client next determines whether the object map is stored locally. Thus, the client 13a checks multiple local memories such as hardware caches, working memory, CD-Rom's and local network memories for the object map in state 108.

If the object map is found locally, then the client analyzes the object map. If the object map is not found locally, then the client requests the object map from the server in state 106.

In response to this request, the server enters states 202, 203 and 204 to (i) initialize a log of object identifications for this client, (ii) initialize a steady state connection with the client, and (iii) transmit the object map identified by the client 13a.

The client 13a then analyzes the object map in state 108. This is accomplished by the client processing each object referred in on the object map, as indicated by the analyze loop of states 110,112 and 114.

For each such object, the client first determines, in state 110, whether the object is in local cache. If not, then the client 13a adds the identification of the object to a request block. As long as the block is not full, this loop continues with the client adding an identification of each object not found in local cache in state 112.

When the block is full in state 114, the client transmits a request to the server for the block of objects as a list of object identifications.

Upon receipt of this request (i.e., a block of object indentifications), the server 17 then initiates the assembly of a data stream and compiles the stream of objects based on (i) the sequence of requested objects and/or (ii) quality of the objects.

To accomplish this the server constructs blocks of objects to form the data stream, in states 206 through 214. In constructing the blocks, the server 17 maintains a log of objects being transmitted to fulfill the client's request. Specifically, in states 206 and 208, for each object in the block, the server 17 first determines from the log whether the object has previously been transmitted to the particular client 13a. If so, the server 17 enters state 214 to prevent that object from being placed in the current block. Therefore, states 110 and 212 are entered only for the objects that the client is actually in need of. In state 210, then, the server only fetches objects from remote servers which are actually needed by the client 13a. This provides a significant performance advantage, especially where the objects must be obtained elsewhere in the network 21. As such, the server 17 constructs blocks of data and hence compiles the stream of data for the client 13a in real time, i.e., on the fly, without duplication of any one object throughout the total data stream.

On the receiving end, the client 13a receives the object stream in state 116. More accurately, the data of the object is then delivered to the requesting application of the client for further processing and use.

In summary, for each such object request by the client 13a, an object streaming type communication from the server 17 to client 13a is performed. In particular, a request for a sequence of non-local objects is made by the client 13a and transmitted to the server 17. In turn, the server 17 initializes a log of objects according to object identifications, and then fulfills the request for objects by transmitting the requested objects, in sequence order, which have not previously been transmitted to that client 13a as indicated by the log.

It should be understood that the objects may be computer data structures of various types and formats. The objects may, for example, be compiled in any number of ways within the object oriented computing model well known in the art. For example, objects may include text, graphics, audio, video, and other types of digitized information. Furthermore, objects may be complete data files or only portions of such files. In addition, the objects themselves may be classes that consist of a number of objects grouped together. The object request must then typically include information to specify which member of the object class is desired by the client 13a.

For example, an object class may consist of various versions of a particular graphic object. The selected version may depend upon the desired quality of the final graphical rendering of the object desired by the client computer 13a. Quality may also have other meaning such as bandwidth, graphical resolution, number of colors, available client memory cache size, and other class criteria that depend upon the type of client computer 13a.

In addition, object classes may depend upon various other client-specific information such as domains, for example. In this scenario, when the client computer 13a is located in one area of the country, such as Massachusetts, a different object may be returned than when the client 13b is located in California, although each of the clients 13a, 13b actually requested the same global object.

Fig. 3 is a diagram illustrating how an example object stream 300 may be assembled by the streaming servers 17, and in particular showing how the object stream does not have to originate from a single source or a single file.

Here the client 13a requests and receives a particular object map 301 consisting of a list of object identifications such as the list (ID1,ID4,ID7,ID6,ID2, ID3), where each object identification IDx indicates a global address for a particular object. According to the process already described in connection with Fig. 2, the client 13a first creates a request block 302 taking into account any local objects 303 which it may already have available. In the particular illustrated example, the client 13a already has local objects (O1, O6) available locally.

After compiling the request block 302, the request block 302 is sent to the streaming server 17. Streaming server 17 receives the request block 302 and then assembles a stream block 310a for the client 13a. It should be understood that other stream blocks 310b may also be constructed for other clients 13b at the same time as the block being constructed for client 13a.

For example, assuming a first time interaction between the client 13a and server 17 in a given stream, the server

17 typically has a default data stream order regardless of user interaction on the client 18 side.

In a first modification, the server 17 may have available certain information concerning the client. Because the server 17 may compose the data stream on-the-fly, the default stream order may be changed in accordance with prior history of requests made on the server 17. That is, on the server processor, an analysis of multiple prior usage of server objects by other clients is made. The server 17 changes the default objects stream 300 order to the closest substitution (i.e., object map) based on demographics of clients 13a having prior server data/object usage. To accomplish such an analysis, a neural network may be employed.

In yet another modification, the streaming server 17 may create the stream block 310a from information that is known about the specific client 13a.

For example, the server 17 may maintain a log 311a of objects that have already been provided to client 13a as well as any available local objects 312 local to the server 17. In the illustrated example the log 311a indicates objects (ID1,ID6) as already having been provided to client 13a. In addition, the available local objects 312 include (O4,O12).

It should be understood, as previously described, that a particular object such as object O4 may actually consist of a class definition, as illustrated, wherein a number of objects comprise the class. For example, object 04 here is actually an object class (O4a,Ob4,...O4x). The streaming server 17 thus also receives together with the object identification request block 302 information as to which particular member of class O4 is appropriately provided to the client 13a.

The streaming server 17 then assembles the stream block 310a as illustrated, which includes all of the objects that the client 13a has requested. In the illustrated example this includes objects (O4a, O7, O2, O12, O3). After assembly of the stream block 310a the streaming server 17 then provides the stream block 310a as a single object stream 300 to the client 13a.

During assembly of the stream block 310a the streaming server 17 may not have all the necessary objects available in the local object list 312. In such instances the streaming server 17 must query other remote server computers 15a, 15b connected to the network 21 in order to locate the objects. This is done in a manner which is well known in the art by the streaming server 17 making requests of the remote computers 15a, 15b to provide their respective objects that they have made available. In Fig. 3, objects (O7, O2, O4) are located at computer 15a and object (O3) at computer 15b.

An identical object map may be operated on in a different manner by server 17 for client 13b. In particular, although the object map 301b is the same as the objects map 301a, because a different list of object 303b is available to client 13b, the request block 302b created by client 13b will have different object identification numbers (ID4, ID6, ID3). Furthermore, the client parameter(s) provided with the request block by client 13b may very well be different for that provided by client 13a. Thus, an object of a particular quality may be targeted to client 13b which is different than as that supplied to client 13a. For example, when clients 13a and 13b request that object O4 be provided to them, client 13a may actually receive object O4a and client 13b may receive object O4b, despite the fact that the reference O4 was a common global object identification.

The object classes may also be defined as bandwith selectable objects. In particular, the resulting data stream 300 may be assembled based upon observed client bandwidth availability. Therefore, depending on a periodically calculated data transfer rate, the client 13a or server 17 may choose to request or transmit a data stream of different objects. That is, for a given requested object 04, a specific version 04a, 04b...04x may be selected for transfer from the server 17 to the client 13a as a function of available bandwidth. This function is termed "bandwidth scalability" of the object stream.

In another possible implementation, "object specific compression" is provided by the server, such that on an object by object basis, the object data stream 300 may be compressed one object at a time depending upon client criteria. In the preferred embodiment, the server 17 determines which compressed object version to include in the object data stream 300 at the time of compilation.

The client 13a also manages the amount of data being delivered, i.e., throughput to a client 13a. In particular, this is useful to determine whether there is enough data being timely transferred; that is, whether data is being consumed on the client 13a end faster than the server 17 is delivering the requested objects.

In the preferred embodiment, the client 13a builds a map of uncompressed data consumption. The map tells how fast data is being consumed (used by the client 13a application). The client then measures the throughput (client receipt) of data in real time and contrasts that with the formulated map. Based on the comparison, the client 13a is able to determine how much of the object stream data 300 should be read by a buffer 320a at a time. Thus, the client 13a effectively maps the physical compressed object data stream 300 and logical consumption of data at each point in time.

The client 13a continually monitors the real data throughput versus data consumption. The client 13a wants the delivery-to-consumption ratio to be greater than one so that the throughput (supply) is keeping up with the consumption (demand). When the delivery-to-consumption ratio is less than one, there is more data being consumed than the amount of data being delivered, such that there is a so-called "buffer debt" on the client 13a side. In that case, the transmission of data needs to take advantage of pause points in the object data stream 300, so that for a period of time, data is not being transmitted over the communication channel 11. In turn, the buffer 320a is allowed to fill up with the requested data and thus decrease or solve the "buffer debt", to assist with proper real time delivery of objects.

The pause points may be pre-defined by the author of the object content, or the pause points may be determined on the fly in accordance with the client's ability to consume data.

In the preferred embodiment this latter implementation is accomplished as follows. The client 13a at each time point, t, computes object data consumption. A running average of throughput such as number of bytes received divided by total time in seconds is employed. An adaptive running average or weighted average or the like may also be used to compute the data consumption. This is also known as the physical throughput at a given time, i, (i.e., (pt_i).

The total logical data (tld) optimally needed at a point in time t is calculated as follows:

$$tld = \sum_{i=o}^{t} rate\ map(i)$$

Buffer debt is then calculated as follows:

$$buffer\ debt = \sum_{i=t}^{end} (tld(i) - pt_i)$$

The client 13a then calculates the buffer debt from time to time. For a buffer debt greater than 0 the client calculates a maximum wait time

maxwait = buffer debt ÷ physical throughput

which equals the number of seconds of pausing needed to rectify the buffer debt. At each wait point or pause point in the object data stream 300, the client 13a will then wait a minimum of the maxwait time or the maximum time allowed at that wait point. For a buffer debt of less than zero, the server 17 transmission of data is ahead of the consumption and thus no pausing during the transmission of the object data stream 300 is warranted.

EQUIVALENTS

While the invention has been particularly shown and described with reference to a preferred embodiment thereof, it will be understood by those skilled in the art that various changes in form and details may be made therein without departing from the spirit and scope of the invention as defined by the appended claims.

For example, the log maintained by the server 17 for preventing duplication of objects in the transmitted object data stream 300 to the requesting client may be implemented with tables, cache memory and the like. In the case of caching, a sparse representation of the most recent transmitted objects is maintained in the log. Other caching or log implementations are suitable and are understood to be in the purview of one skilled in the art.

Another example is the term "local to the client". "Local" means in various memories including hard drives, cache memories, CD's and other working memories in the local network involving the client 13a.

CLAIMS

1. In a computer network having a plurality of digital processors loosely coupled to a communication channel for communication among the digital processors, a method of transmitting data from one digital processor to a second digital processor comprising the steps of:

providing a server processor;

providing a requesting processor;

coupling a communication channel between the server processor and requesting processor to enable communication between the server processor and requesting processor;

in the requesting processor, forming a request for a desired sequence of objects;

transmitting the formed request across the communication channel from the requesting processor to the server processor;

in the server processor, in response to receipt of the request, assembling and transmitting, across the communication channel to the requesting processor, a data stream based on the desired sequence of objects, such that a set of objects in a defined order is transmitted in the data stream from the server processor to the requesting processor.

2. A method as claimed in Claim 1 wherein the step of assembling and transmitting a data stream includes:

in the server processor, recording an indication of objects being transmitted to the requesting processor in response to the request; and

based on the recording, preventing plural and subsequent transmission of an object indicated on the recording by deleting from the data stream objects indicated on the recording such that the transmitted set of objects is formed only of objects which have not been previously transmitted to the requesting processor in response to the request as indicated in the recording.

3. A method as claimed in Claim 1 wherein the step of assembling and transmitting a data stream includes:

in the server processor for each of certain objects, providing multiple versions of the object;

for each of the certain objects, in one of the server processor and requesting processor, determining one version of the object to be optimal for transmission in terms of available bandwidth of the communication channel; and

using the determined version of the object in the set of objects transmitted in the data stream.

4. A method as claimed in Claim 1 wherein the step of forming a request in the requesting processor includes:

for each object in the desired sequence, determining whether the object is locally stored; and

omitting from the request those objects determined to be locally stored but maintaining sequence ordering of the objects in the request.

5. A method as claimed in Claim 1 further including the step of monitoring a rate at which the requesting processor uses requested objects such that rate at which the requesting processor receives requested objects transmitted from the server processor is sufficiently fast to prevent the data stream from