VideoZoom Spatio-Temporal Video Browser

John R. Smith, *Member, IEEE*

***Abstract*—We describe a system for browsing and interactively retrieving video over the Internet at multiple spatial and temporal resolutions. The VideoZoom system enables users to start with coarse, low-resolution views of the sequences and selectively zoom-in in space and time. VideoZoom decomposes the video sequences into a hierarchy of view elements, which are retrieved in a progressive fashion. The client browser incrementally builds the views by retrieving, caching, and assembling the view elements, as needed.**

By integrating browsing and retrieval into a single progressive retrieval paradigm, VideoZoom provides a new and useful system for accessing video over the Internet. VideoZoom is suitable for digital video libraries and a number of other applications in which streaming methods provide insufficient quality of video, video downloading introduces large latencies, and generating video summaries is difficult or not well integrated with video retrieval tasks.

***Index Terms*— Coding, digital libraries, image and video segmentation for interactive services, progressive retrieval, video signal processing, video summaries and storyboards.**

I. Introduction

DIGITAL video is finding important roles in many online applications in education, news, science, advertising, and entertainment, and as a result, has the potential to become one of the dominant media types on the Internet. However, at present, the Internet is not well suited for video. The typical end-to-end data rates are not sufficient for obtaining high quality streaming video. Furthermore, the Internet does not provide the necessary quality of service (QoS) guarantees that are needed to support real-time video transmission [1]. Some of these shortcomings are being addressed by the Internet2 and Next Generation Internet Initiative (NGII) [2] efforts, whose objectives are to develop new technologies for the Internet infrastructure, increase network capacity, and include provisions for guaranteeing QoS. However, due to the relatively high bandwidth and network resource requirements of video, new access paradigms need to be investigated.

In order to deal with current video access problems on the Internet, new methods are needed that allow more efficient browsing, retrieval, and remote interaction. The predominant methods of accessing video on the Internet are streaming, video file downloading, and video browsing. Streaming and downloading are designed to address the issues of remote and local playback, respectively, and are not well suited for interactive navigation. Downloading video for local playback is problematic for most browsing tasks since the large video files typically result in significantly large access latencies.

This is especially problematic if the user needs to download many video sequences in order to find the relevant content. Video streaming avoids the initial latencies by playing the video back directly over the Internet. However, the result is often of poor quality in terms of frame rate and fidelity. Video summarization and browsing is useful for identifying relevant video content. However, strong separation of the browsing and retrieval data increases the net amount of data needed to be transmitted over the network. Furthermore, video summary browsing does not solve the subsequent quality versus latency problem in the retrieval stage.

A. Video Applications

Improving the ability to browse and interactively retrieve video over the Internet has the potential to greatly benefit many applications. For example, consider a digital video library that presents the user with a number of video sequences that result from a search. Ultimately, the user is interested in some of the items and wants to view the relevant segments at high-resolution. Given the option of downloading versus not-downloading, it is difficult for the user to quickly identify the relevant content. Given the option of streaming versus not-streaming, it is difficult for the user to obtain or view the relevant content at high-resolution.

We propose that progressive video retrieval is a useful paradigm for accessing digital video libraries over the Internet. In progressive retrieval, the user initially retrieves coarse views of the sequences. Then, by drilling-down into different temporal segments, the user is able to simultaneously identify the relevant content and retrieve it at high resolution.

The progressive video retrieval paradigm is illustrated in Fig. 1. Initially, many of the different temporal segments of the video are possibly relevant (PR). Based on the initial low-resolution views, as shown in Fig. 1(a), the user identifies some of the segments as nonrelevant (NR), possibly relevant (PR), or relevant (R) to varying degrees by building up details, as shown in Fig. 1(b). The user repeats the process of zooming in on the segments of most interest, until the final relevant (R) content is obtained at high-resolution. Identifying the nonrelevant (NR) segments early using progressive retrieval allows better allocation of network and client storage resources.

The progressive retrieval paradigm is especially well suited for many types of video content on the Internet that do not fit into traditional video abstraction, summarization, and

Manuscript received September 9, 1998; revised April 6, 1999. The associate editor coordinating the review of this paper and approving it for publication was Prof. Wayne Wolf.

The author is with IBM T. J. Watson Research Center, Hawthorne, NY 10532 (e-mail: jrsmith@watson.ibm.com).

Publisher Item Identifier S 1520-9210(99)05836-8.

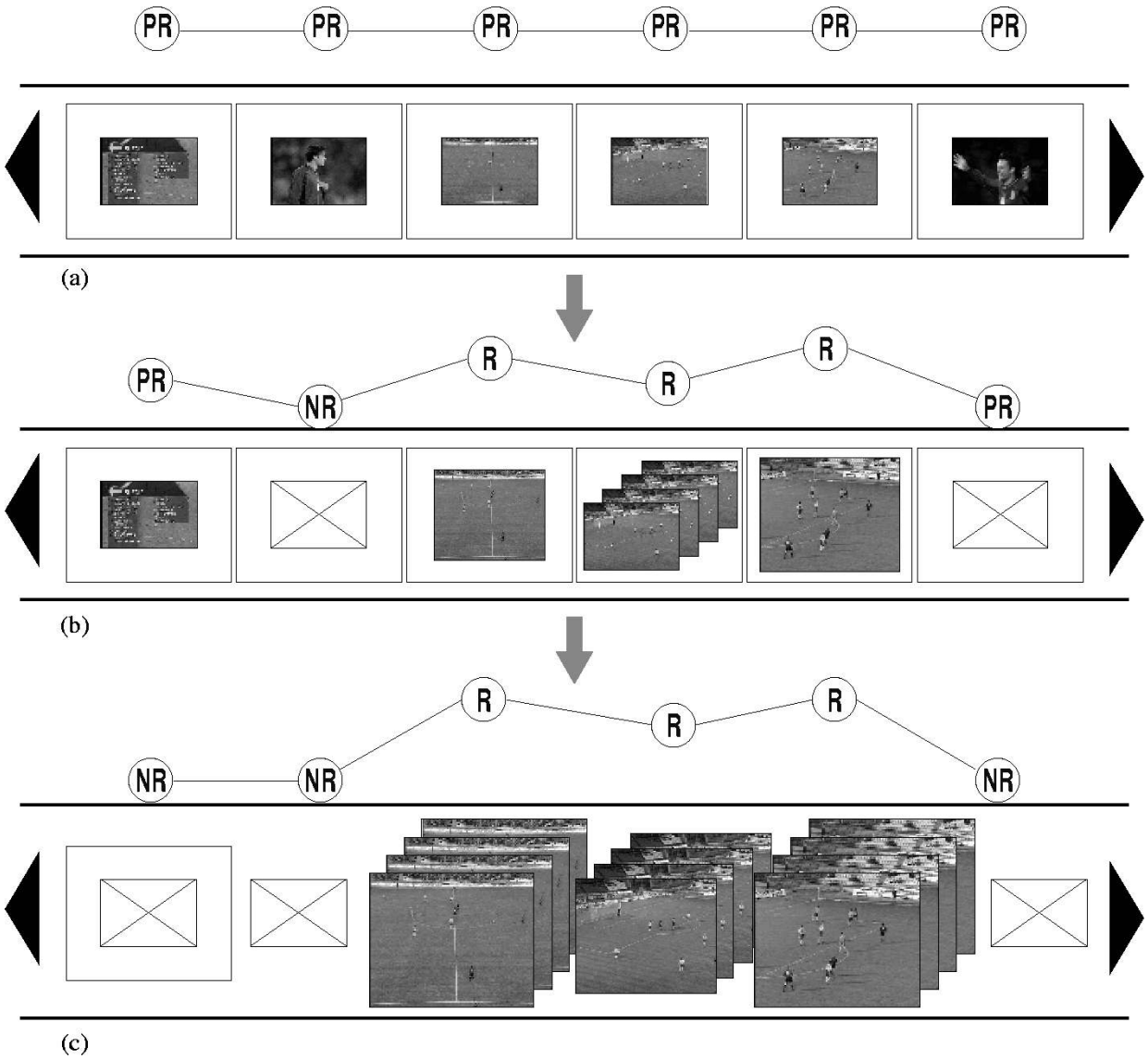

Fig. 1. In progressive video retrieval, at each retrieval iteration, the different temporal segments are identified as being nonrelevant (NR), possibly relevant (PR), and relevant (R) to varying degrees.

browsing paradigms. For example, the National Aeronautics and Space Administration (NASA) has deployed satellites that acquire image sequences of the earth and solar system. NASA publishes the sequences on the Internet to allow further study by scientists. The extreme ultraviolet imaging telescope (EIT) on the solar and heliospheric observatory (SOHO) satellite acquires image sequences of the solar transition region and inner corona of the sun in four wavelengths [3].

Often in accessing these sequences, scientists are looking for various spatio-temporal phenomena. Downloading the sequences over the Internet is not practical due to the enormous amount of data (>96 GB/year/instrument). Video streaming does not provide sufficient quality due to the high-resolution of the images (1K pels $\times$ 1K pels/frame). Since these sequences do not satisfy any notion of shots, camera breaks or scenes, video abstraction, and summarization methods cannot be used. Progressive retrieval allows the users to browse the solar sequences at various levels of spatial and temporal detail and zoom-in on the segments of most interest.

B. Related Work

The efficient storage, search, and retrieval of video has been the focus of a number of projects related to digital video libraries [4]–[7]. In the Informedia digital library project, Wactlar *et al.* investigated the integration of video and speech analysis for indexing and querying of video [4]. Wolf *et al.* [6] deployed an education-centric digital video library on the Web, which uses automatic video abstraction methods to develop video summaries for remote browsing. In the Vision project, Lei *et al.* [5] developed a digital video library that uses shot detection and keyframe selection for indexing and browsing.

A common framework for searching, browsing, and retrieving video has emerged across a number of digital video library projects [7]. As shown in Fig. 2, in digital video libraries, video is typically handled separately in the ingestion process by subsystems for compression, analysis,, and indexing. When online, the video is also accessed separately in the different tasks for searching, browsing,, and retrieval. Consequently,

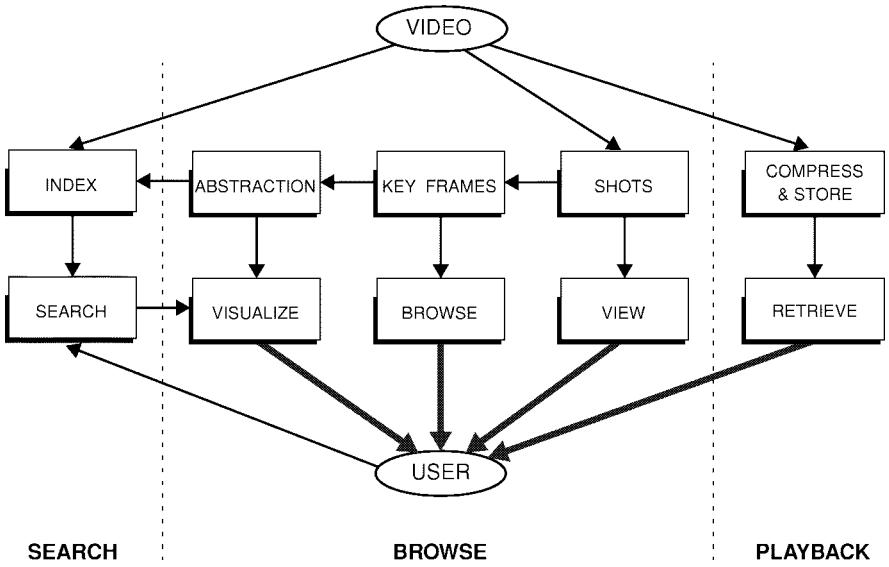

Fig. 2. The traditional processing of video for digital video libraries and other online applications strongly separates the tasks for searching, browsing, and playback.

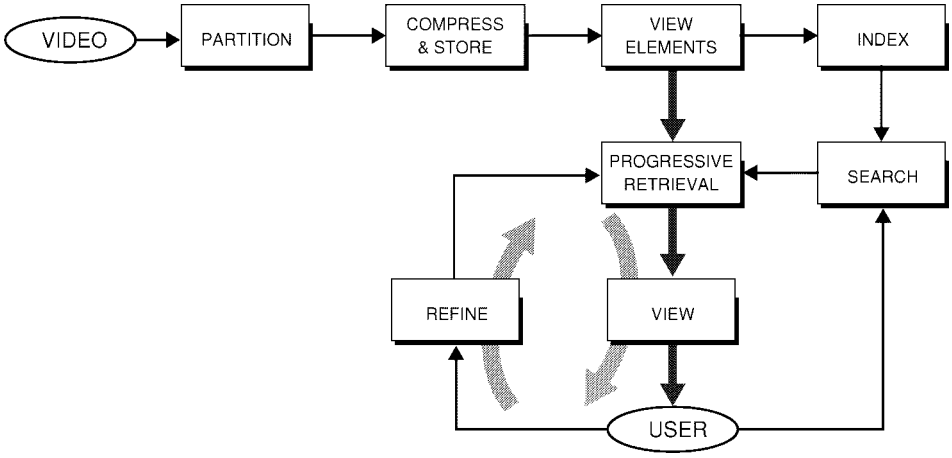

Fig. 3. The progressive retrieval paradigm integrates browsing and retrieval into a combined task in which the user iteratively and selectively refines the video by building details of the segments of most interest.

the separate treatment and distinct data results in a loss of efficiency in accessing video from remote digital video libraries. We next describe the query, browsing, representation, and retrieval processes of digital video libraries in more detail.

1) Video Query: The initial access to the digital video library typically occurs through a query interface. Recently, many video query methods have been investigated that search against low-level visual features such as color, texture, shape, and motion [8]–[12]. Querying of the spatial and temporal information in video sequences is also important and has been investigated in a number of video database projects [13]–[15]. Once the query is made to the digital video library, the system typically returns browse data to the user. The browse data may take the form of keyframe lists, video shot thumbnails, video posters, scene transition graphs, and so forth. The objective of the browse stage is to allow the user to identify the relevant items.

2) Video Browsing: Many recent methods of video browsing have been investigated that combine automated video analysis and visualization [16]–[18]. The common tools used by many analysis methods include shot detection and keyframe selection [19]–[21]. These tools are often used to construct video abstractions that provide higher-level summaries of the video content. Yeung and Yeo [17] developed a method for clustering the keyframes to construct a scene-transition graph that provides a visualization of the recurring scenes and the transitions between them. Zhong *et al.* [18] developed a video browser that organizes temporal segments of the video into a hierarchy. The system allows the user to browse the video at increasing temporal resolution by drilling down deeper into the hierarchy.

3) Video Representation and Retrieval: The representation and coding of video is important for storage and retrieval in the digital video library. There are many video coding formats commonly used on the Internet, such as MPEG-1, MPEG-2, MPEG-4, motion-JPEG, Realvideo, Quicktime, and AVI. Both MPEG-4 and Realvideo handle the streaming playback of the coded video over the Internet [22]. MPEG-1 and MPEG-2

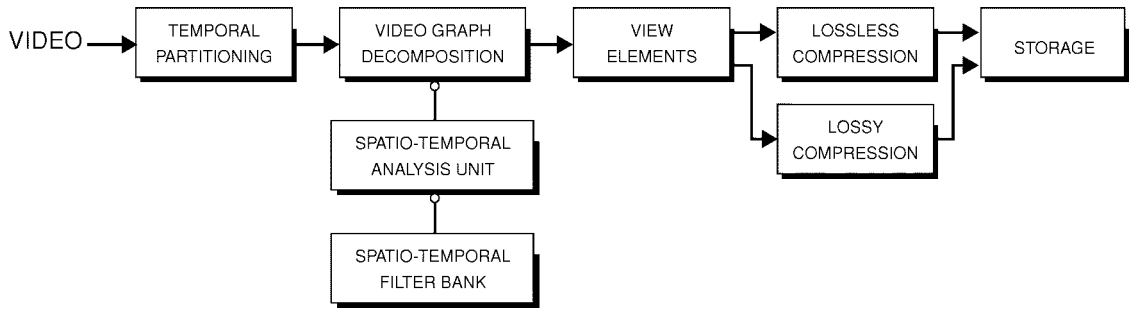

Fig. 4. The video ingestion process first partitions the video in time, then decomposes the partitions into view elements, and finally compresses and stores the view elements.

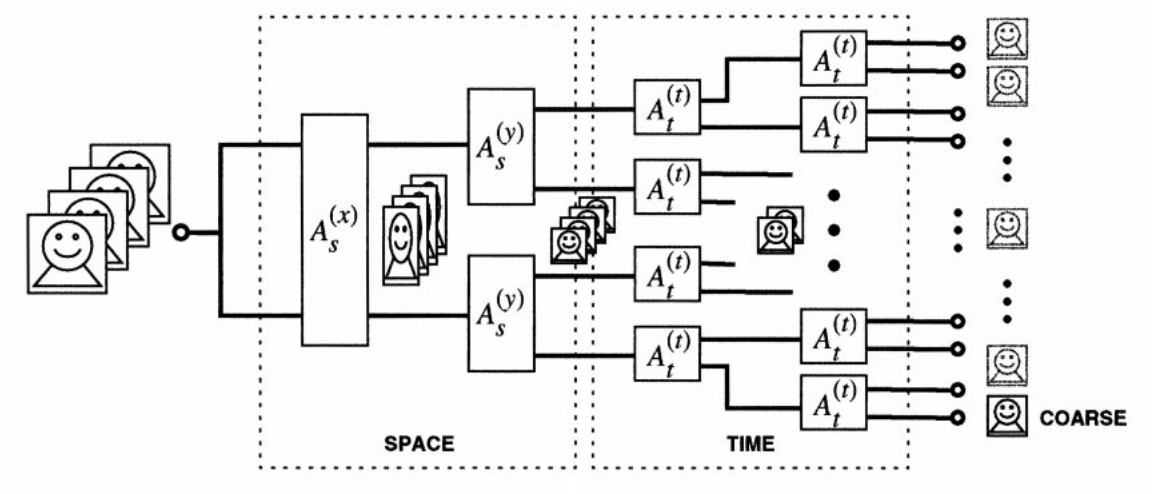

Fig. 5. The spatio-temporal analysis unit decomposes each input group of four frames in space ($A_s^{(x)}$, $A_s^{(y)}$) and time ($A_t^{(t)}$, $A_t^{(t)}$) to generate a total of 16 view elements, one of which is a coarse view of the group.

provide mechanisms for scalable coding that embed both base and enhancement layers within a single coded stream. MPEG-2 provides different profiles for spatial, temporal, and SNR scalability in which the clients can decode the video streams at different levels of detail in space, time and quality, respectively [23].

A number of projects have focused on using wavelets for video coding [24]–[26]. One advantage is that wavelets embed multiple versions of the video sequences at different spatial and temporal resolutions in the compressed stream. As with MPEG scalability, this can provide spatial and temporal scalability for adapting to network and client resources. In addition, wavelet coding can also be used to enable progressive retrieval of video in which wavelet subbands are retrieved as needed to build up details of the video sequence at the client.

4) Progressive Retrieval: Progressive retrieval has been found to be very useful for image retrieval applications [27]–[29]. In progressive image retrieval, the objective is to initially transmit a coarse, low-resolution view of the image and allow the user to subsequently retrieve additional residual information. Smith *et al.* [28] developed a progressive image retrieval system that uses a space and frequency graph to progressively retrieve large images, high-resolution documents, and maps. The system allows users to zoom-in on the spatial areas of interest.

In progressive retrieval of video, there are a number of ways in which the user can refine the video sequences [30]. Given a coarse view of the video, the user can fill in details in space or time. In spatial refinement, the added data increases the size or spatial resolution of each video frame. In temporal refinement, the added data increases the frame-rate.

We propose a new progressive video retrieval paradigm in which the browse and retrieval phases are integrated in a progressive video retrieval system, as shown in Fig. 3. The users selectively zoom-in on the segments of the video by retrieving *view elements* from a *video view element graph*. The *video graph* generates and manages the view elements and allows the client applications to retrieve, cache, re-use view elements to synthesize the multiple spatial- and temporal-resolution views of the video sequences.

C. Outline

In this paper, we describe the VideoZoom system for progressive retrieval of video over the Internet. The paper is organized as follows. In Section II, we present the video view element representation which is generated and managed using a video graph. In Section III, we describe two types of video graphs that are useful for progressive retrieval. In Section IV, we describe the process for compressing the view elements in the video graph. In Section V, we describe video view access

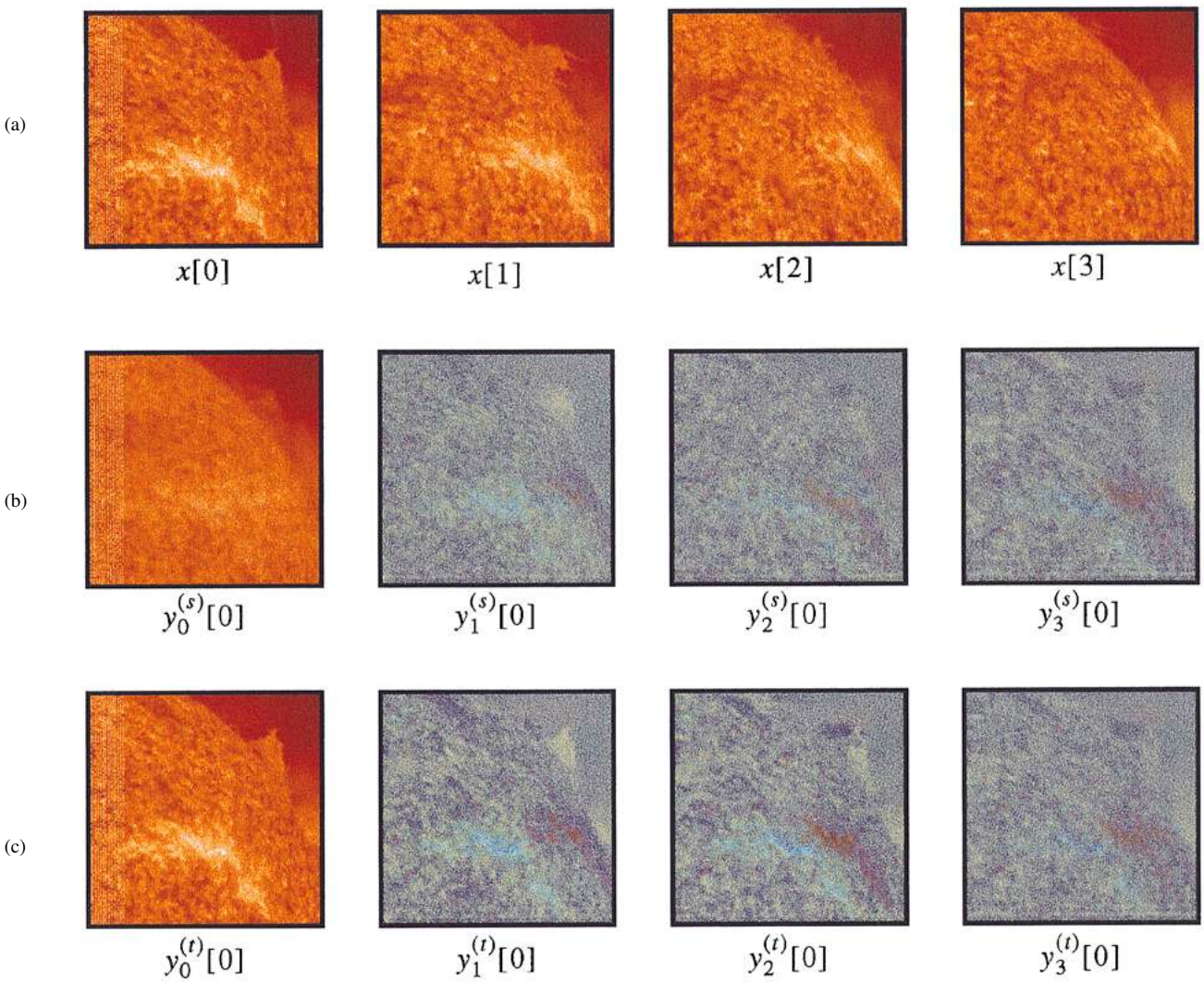

Fig. 6. Different filter banks are used in spatial and temporal analysis in order to generate good coarse views: (a) sequence of four solar images, (b) temporal analysis with filtering in the coarse path, and (c) temporal analysis without filtering in the coarse path.

process in which view elements are combined to synthesize views at different levels of spatial and temporal resolution. Finally, in Section VI, we describe the process for interactively and progressively retrieving views of the video sequences over the Internet by retrieving, caching, and assembling the video view elements.

II. Video Representation

The VideoZoom system uses a video graph to decompose the sequences and represent them using view elements. As show in Fig. 4, the ingestion process first partitions the video temporally, then decomposes each partition using a video graph (Fig. 8 shows an example of a uniform video graph). The partitioning operation segments the sequences into fixed temporal units of 4^D frames, where D is the depth of the video graph, and typically, we choose $D = 3$. The video graph is constructed by integrating the spatial and temporal filter bank building blocks. Overall, the view elements generated by the video graphs roughly correspond to subbands with different locations and sizes in spatial- and temporal-frequency.

The terminal view elements in the video graph are initially compressed using adaptive lossy compression. Keeping in mind the applications for scientific image sequences, such as the SOHO solar image sequences describe earlier, we also develop an optional lossless compression method that allows the reconstruction at full resolution without loss.

A. Spatio-Temporal Decomposition

In order to decompose each temporal segment, we use separable wavelet filter banks with different filters in space and time. Each spatio-temporal analysis unit, as depicted in Fig. 5, is composed from the basic spatial and temporal analysis building blocks, which are illustrated in Fig. 7.

B. Spatio-Temporal Analysis Unit

The spatial A_s and temporal A_t analyzes are performed separately on the x- and y-dimensions ($A_s^{(x)}$, $A_s^{(y)}$) of each video frame, respectively, and temporally across frames ($A_t^{(t)}$). Due to separability, the order in which the analyzes are

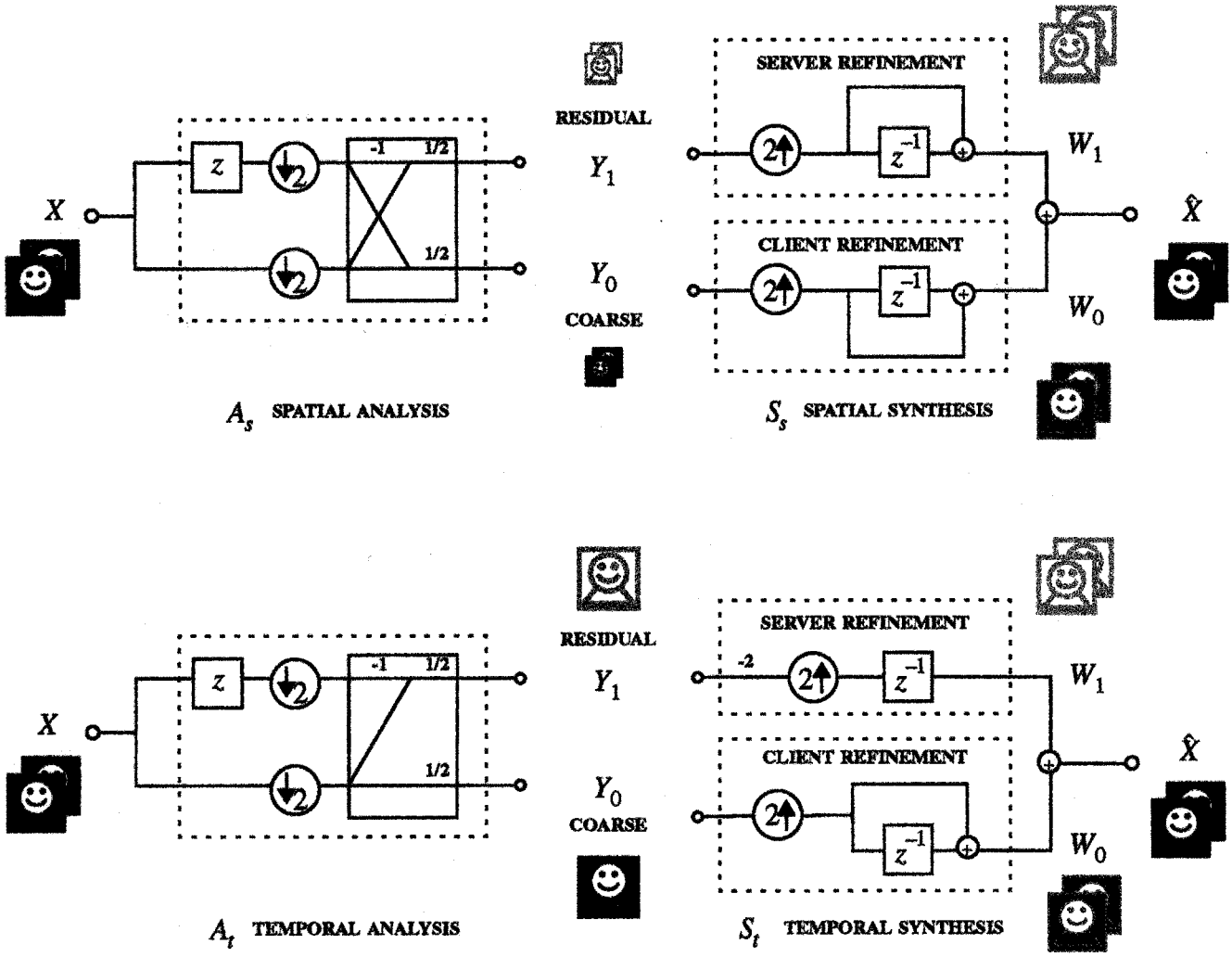

Fig. 7. Analysis and synthesis filter banks form the building blocks for temporal and spatial zooming. Each analysis filter bank splits the input X into coarse Y_0 and residual Y_1 view elements. The synthesis filter banks reconstruct the input $\hat{X}$ from the view elements in a two-step process: (a) spatial analysis and synthesis filter banks and (b) temporal analysis and synthesis filter banks.

performed does not affect the overall decomposition. As shown in Fig. 5, in each unit of spatio-temporal analysis, segments consisting of four frames are reduced once on each spatial dimension (x, y) and twice in time (t, t) to generate 16 view elements. The coarse sequence has 1/4 resolution in space and 1/4 resolution in time, as depicted in the bottom-right of Fig. 5.

1) Spatial versus Temporal Blurring: Each pass of a spatial and temporal filter bank decomposes the input into two view elements: one coarse view and one residual view element. The reason for using different filters in space and time is to obtain good coarse views. The temporal filter bank avoids filtering in the coarse path in order to allow one of the frames to serve directly as the coarse view. However, in space, it is beneficial to filter before subsampling in order to avoid anti-aliasing artifacts.

We illustrate this for the sequence of four solar images, $X = (x[0], \cdots, x[3])$, shown in Fig. 6(a). We decompose the sequence two times in time and compare the results of using different analysis filter banks. Fig. 6(b) shows the poor blurred coarse view, $y_0^{(s)}[0]$, that results from using an analysis filter bank that filters in time across pairs of frames at each iteration. Fig. 6(c) shows that without filtering in the coarse path, the decomposition simply selects the first frame, $x[0]$, as the coarse view, $y_0^{(t)}[0]$. This coarse view, $y_0^{(t)}[0]$, provides a more useful representation of the sequence.

One drawback of obtaining the coarse view by simply subsampling is that the energy is not compacted well into the coarse view. As a result, the residual view elements $(y_1^{(t)}[0], y_2^{(t)}[0], y_3^{(t)}[0])$ contain more energy than those of $(y_1^{(s)}[0], y_2^{(s)}[0], y_3^{(s)}[0])$. This potentially affects the compression of the view elements, but is necessary to obtain good coarse views in time.

2) Analysis and Synthesis: We describe the analysis and synthesis filter banks in the polyphase domain [31]. Each analysis filter bank decomposes its input X into a coarse Y_0 view of X and a residual Y_1 view element as follows:

$$\begin{pmatrix} Y_0(z) \\ Y_1(z) \end{pmatrix} = \mathbf{H}_p(z) \begin{pmatrix} X_0(z) \\ X_1(z) \end{pmatrix} \tag{1}$$

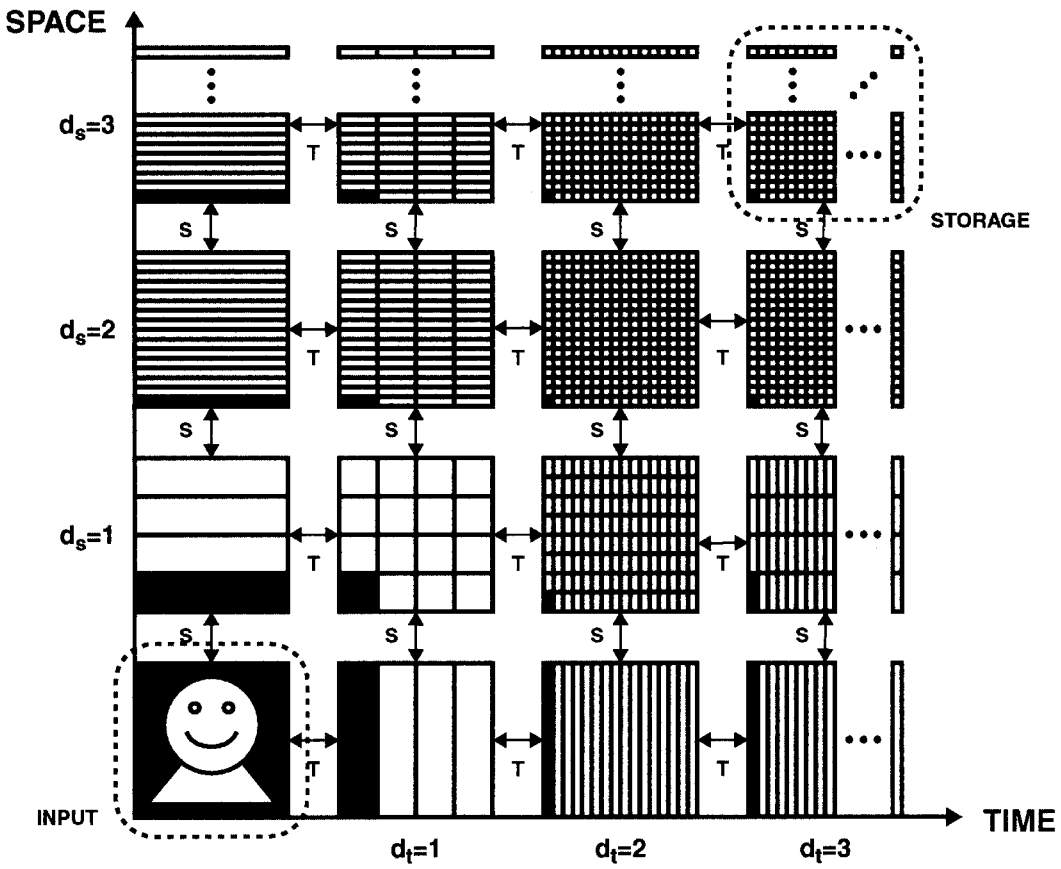

Fig. 8. Uniform video graph with depth $D = 3$ generates and stores 4096 view elements including one coarse view which, for an input sequence of size $(w \times h \times n)$ has size $(w/8 \times h/8 \times n/64)$, where w = width, h = height and n = number of frames. The uniform video graph has 16 coarse views and provides 20 unique paths for zooming from the most coarse view to the full-resolution sequence.

where X_0 and X_1 are the polyphase components of X given by $X(z) = X_0(z^2) + z^{-1}X_1(z)$. Each synthesis filter bank reconstructs $\hat{X}$ from Y_0 and Y_1 as follows:

$$\hat{X}(z) = (1 \quad z^{-1})\mathbf{G}_p(z^2)\begin{pmatrix} Y_0(z^2) \\ Y_1(z^2) \end{pmatrix}. \tag{2}$$

We have perfect reconstruction of $\hat{X} = X$ when $\mathbf{G}_p(z^2)\mathbf{H}_p(z) = \mathbf{I}$.

a) Spatial filter bank: We define the spatial analysis polyphase matrix $\mathbf{H}_p^{(s)}(z)$ and spatial synthesis polyphase matrix $\mathbf{G}_p^{(s)}(z^2)$ as follows:

$$\mathbf{H}_p^{(s)}(z) = \begin{pmatrix} 1/2 & 1/2 \\ 1/2 & -1/2 \end{pmatrix} \quad \text{and} \quad \mathbf{G}_p^{(s)}(z^2) = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}. \tag{3}$$

b) Temporal Filter Bank: We define the temporal analysis polyphase matrix $\mathbf{H}_p^{(t)}(z)$ and temporal synthesis polyphase matrix $\mathbf{G}_p^{(t)}(z)$ as follows:

$$\mathbf{H}_p^{(t)}(z) = \begin{pmatrix} 1 & 0 \\ 1/2 & -1/2 \end{pmatrix} \quad \text{and} \quad \mathbf{G}_p^{(t)}(z^2) = \begin{pmatrix} 1 & 0 \\ 1 & -2 \end{pmatrix}. \tag{4}$$

Note that, as described earlier, using $\mathbf{H}_p^{(t)}(z)$, the coarse view Y_0 is simply a subsampled version of X generated without any filtering.

3) Progressive Refinement: The spatial and temporal synthesis filter banks are used to synthesize the views of the video sequence from the view elements. We partition the synthesis and refinement operations into two steps, one of client refinement, and the other, of server refinement. In progressive retrieval, the first step uses the view elements already at the client to approximate the requested view. The second step corrects the approximation by filling in missing details. In this way, in an interactive environment, when the user clicks to retrieve a higher resolution view of a segment, step one is carried out immediately to provide initial feedback. Then, step two, which requires retrieval of additional residual view elements over the network, is carried out afterwards to correct the view.

a) Spatial refinement: Given a coarse view Y_0, spatial details are added by retrieving residual data Y_1 and building the view in two steps.

1) *Client spatial refinement*: $W_0(z^2) = (1 + z^{-1})Y_0(z^2)$. Up-sample Y_0 by two and add it to a shifted, up-sampled version of itself.
2) *Server spatial refinement*: $W_1(z^2) = (1 - z^{-1})Y_1(z^2)$. Retrieve Y_1, upsample by two and subtract a shifted,

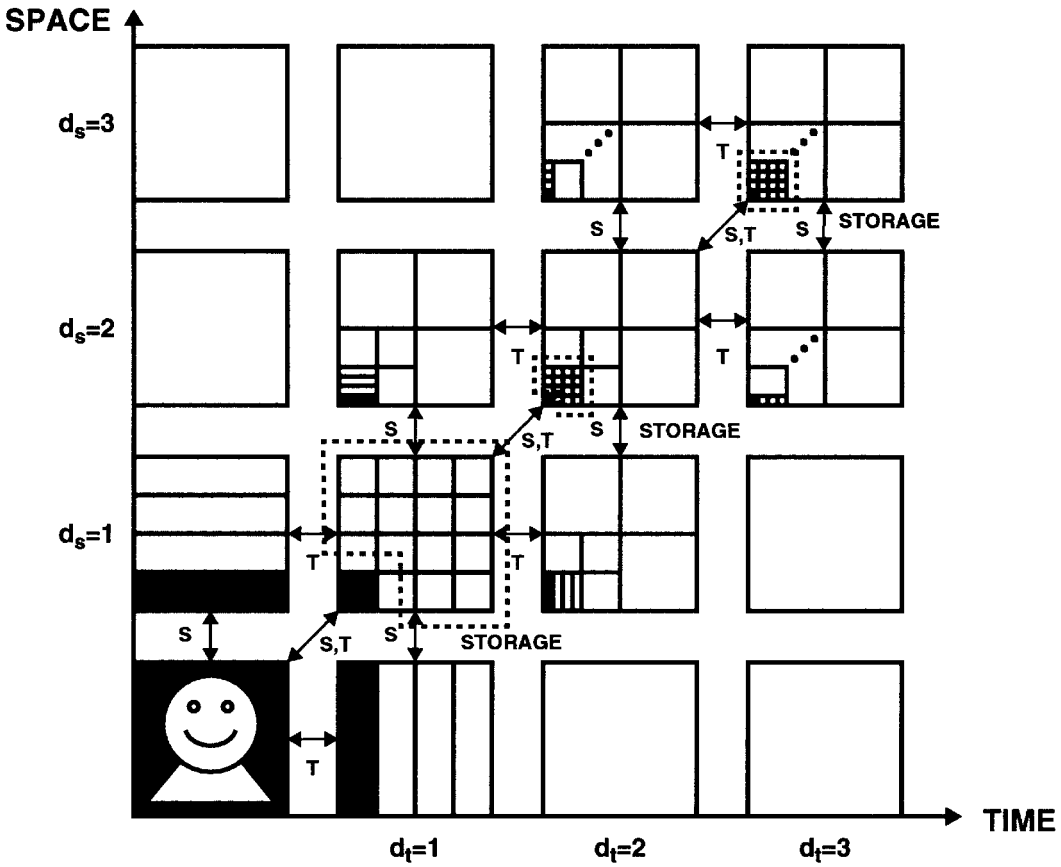

Fig. 9. Wavelet video graph with depth $D = 3$ generates and stores 37 view elements including one coarse view which of size $(w/8 \times h/8 \times n/64)$. The wavelet video graph has ten coarse views and provides eight unique paths for zooming from the most coarse view to the full-resolution sequence.

up-sampled version of itself, and add to the output of step one.

b) Temporal refinement: Given a coarse view Y_0, temporal details are added by retrieving residual data Y_1 and building the view in two steps.

1) *Client temporal refinement*: $W_0(z^2) = (1+z^{-1})Y_0(z^2)$. Up-sample Y_0 by two and add it to a shifted, up-sampled version of itself.
2) *Server temporal refinement*: $W_1(z^2) = -2z^{-1}Y_1(z^2)$. Retrieve Y_1, multiply by -2, up-sample by two, shift by one, and add to the output of step one.

III. Video Graph

Given the spatial and temporal analysis and synthesis building blocks, we cascade the filter banks using graph-structured cascades, as illustrated in Figs. 8 and 9. To generate a decomposition of depth $D = 3$, i.e., $(d_s = 3, d_t = 3)$, the spatio-temporal analysis unit of Fig. 5 is iterated three times. We distinguish between two types of video graphs: the uniform video graph (Fig. 8) and the wavelet video graph (Fig. 9). The uniform video graph provides higher granularity of view elements and more options for refinement.

Figs. 8 and 9 show the analysis paths in which the view elements are generated by analyzing the input sequence along the spatial and temporal dimensions. Likewise, they show the synthesis paths in which the view elements can be assembled to synthesize views of the sequence. In these figures, the darkened view elements correspond to the different coarse views. Typically, these are the views that are of interest to the users. The remaining view elements are residual view elements that add details in building the views of higher resolution in space or time.

A. Uniform Video Graph

In the uniform video graph, the iterative decomposition is carried out on all of the view elements at each stage, i.e., $(d_s, d_t) = (0, 0) \cdots (2, 2)$. As shown in Fig. 8, the uniform video graph generates and stores a total of 4096 view elements. Overall, the uniform video graph allows the synthesis of 16 different coarse views with different resolutions in space and time. Of the stored view elements, one serves as the most coarse representation of the sequence. It has a 1/64th resolution in space and 1/64th in time. For example, given a sequence of $t_0 = 64\,000$ frames of size $w_0 \times h_0 = 512 \times 512$, the uniform video graph generates a coarse view with $t_3 = 100$ frames of size $w_3 \times h_3 = 64 \times 64$.

B. Wavelet Video Graph

In the wavelet video graph, the iterations are carried out only on the coarse view at each stage. As shown in Fig. 9, the

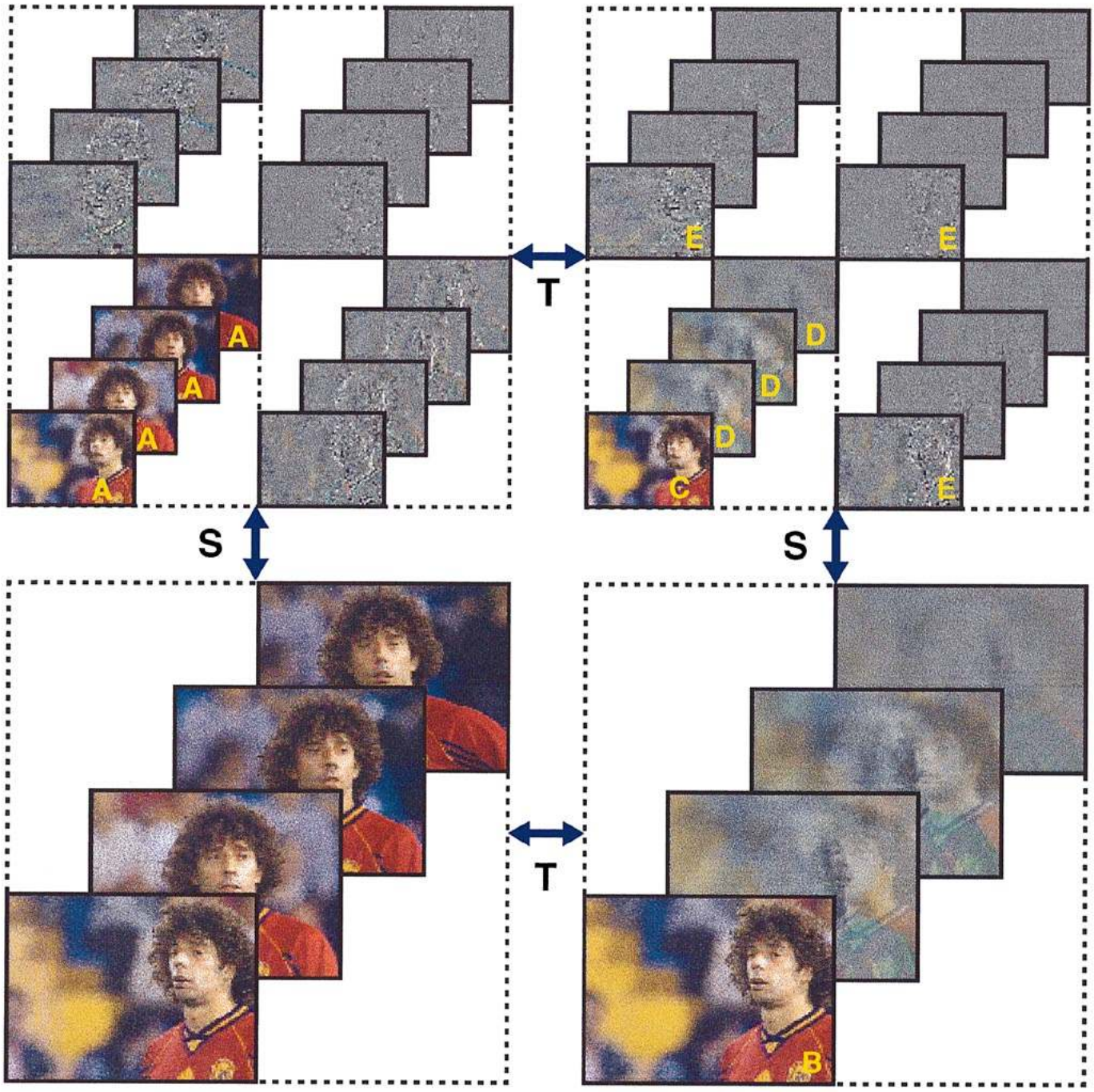

Fig. 10. Applying one unit of spatio-temporal analysis to a sequence of four soccer images generates the 16 view elements depicted in the upper-right, with one coarse view C. These view elements can be used to synthesize views A and B, which have different resolutions in space and time as shown above.

wavelet video graph generates and stores a total of 37 view elements. Overall, the wavelet video graph allows the synthesis of ten different coarse views with different resolutions in space and time. As in the uniform case, of the stored view elements, the coarse view has a 1/64th resolution in space and 1/64th.

IV. Video View Element Compression

In order to represent the video sequence using the video graphs, we compress and store a complete and nonredundant set of view elements. Using the uniform video graph, we do this by compressing the 4096 terminal view elements. Using the wavelet video graph, we compress the 37 wavelet view elements. We develop both lossy and lossless compression methods for compressing the video graphs. In lossy compression, we adaptively compress each stored view element using JPEG compression. To compress the video graph without loss, we first compress the view elements using adaptive lossy compression, then reconstruct the sequence using the compressed view elements, and compress the error sequence using lossless compression.

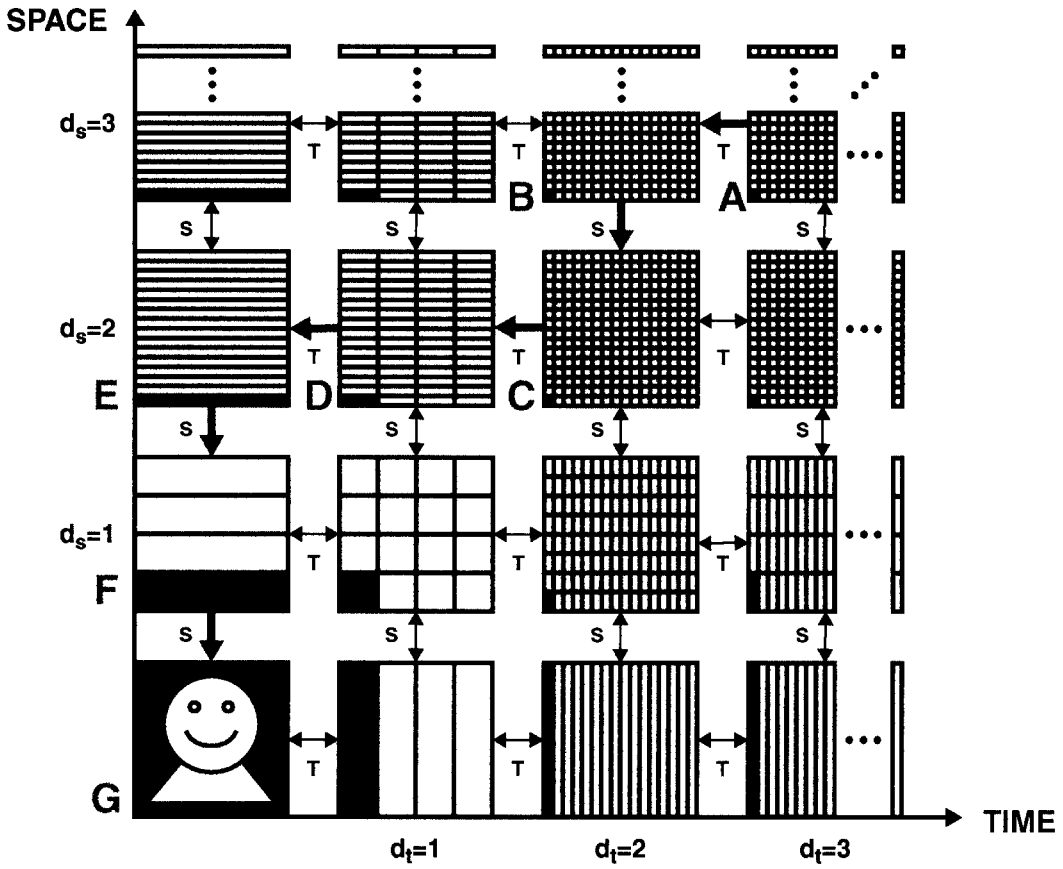

Fig. 11. In this example progressive retrieval session, the user initially retrieves the coarse view (A), then zooms-in in time (B), space (C), time (D, E), and space (F, G) to build up details of the video sequence.

A. Lossy Video Graph Compression

The lossy view element compression system takes advantage of an independent coding of the view elements in order to optimize compression performance. Independent coding allows the compression system to adapt to the spatio-temporal energy of the video sequence by varying the JPEG compression factor according to the rate-distortion characteristics of the view element.

With independent coding, the optimal adaptive compression requires that each view element operates at the same rate-distortion tradeoff [32]. In this way, we need to search over different values of the rate-distortion tradeoff in order to best satisfy the compression criterion. The compression criterion is expressed as maximizing the fidelity for a given target bit-rate [33].

1) Lossy View Element Compression: We use JPEG [34] to provide the underlying coding of the view elements. For coding the residual view elements, we modify the JPEG quantization matrices to have the same scale factor for all of the terms of the 8×8 DCT [35]. The compression algorithm consists of the following steps.

1) Use either the uniform or wavelet video graph to generate a complete and nonredundant set of view elements $\{V_i\}$.
2) Using JPEG, we compress each view element using eight JPEG quality factors: $Q = \{20, 45, 65, 75, 80, 88, 95, 97\}$. For each view element V_i and quality factor Q_j, we compute the size of the compressed data R_{ij} and the distortion D_{ij}.
3) We then select for each V_i the triple: (Q_i^*, R_i^*, D_i^*) that solves the constrained optimization problem.
   - $\min \sum_i D_i^*$, such that $\sum_i R_i^* \leq R_T$ (minimize the total distortion for a total data size of R_T), by iterating over λ and by solving the following unconstrained problem.
   - $\min \sum_i (D_i^* + \lambda R_i^*)$, such that $\sum_i R_i^* \leq R_T$.
4) We then compress each view element V_i using its selected quality factor Q_i^*.
5) We store the compressed view elements $\{\hat{V}_i\}$.

Fig. 10 illustrates the results of applying one unit of spatio-temporal analysis. We can see that the analysis compacts a significant amount of the energy into the single coarse view (labeled C). The remaining view elements in the upper-right have less energy. Adaptive lossy compression of the video graph adjusts the quality factor according to the characteristics of each view element.

B. Lossless Video Graph Compression

For the lossless compression, we compress the difference of the reconstructed and original sequence at full resolution. The lossless compression algorithm consists of the following steps.

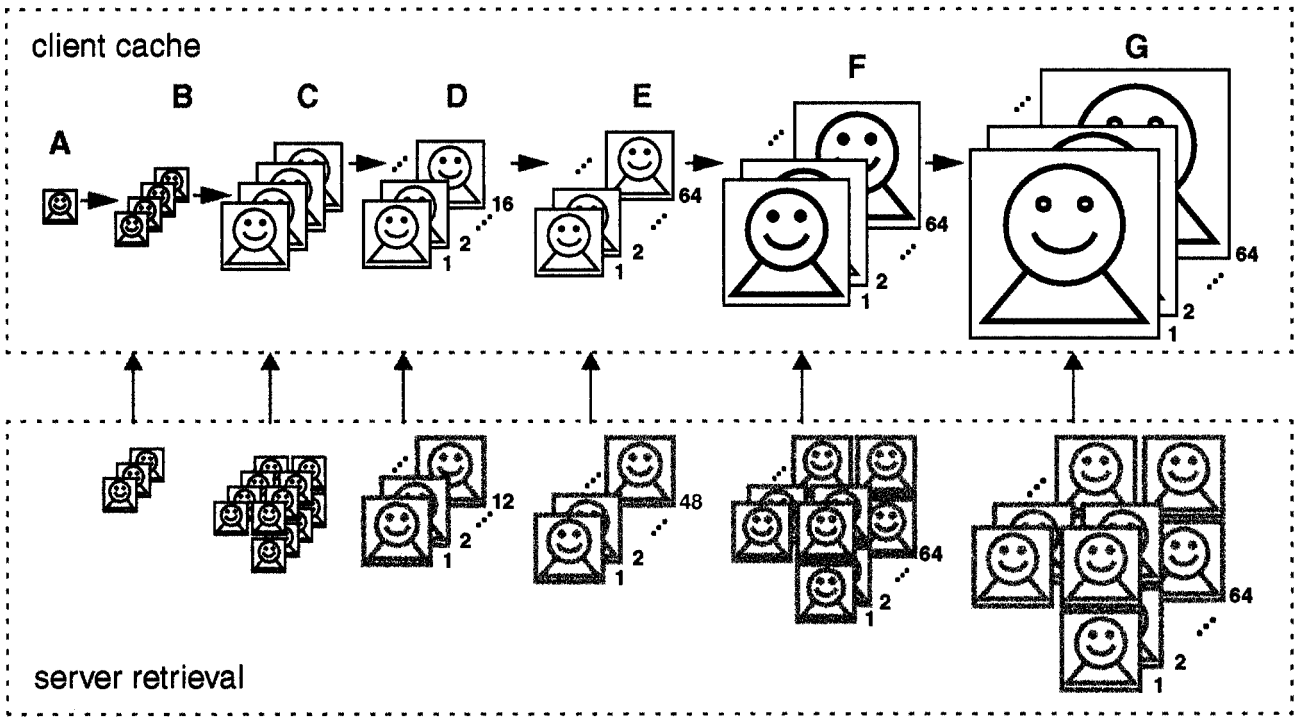

Fig. 12. In progressive video retrieval, the VideoZoom system retrieves the residual view elements from the server and combines with those in the client cache to synthesize the higher-resolution views.

1) Use either the uniform or wavelet video graph to generate a complete and nonredundant set of view elements $\{V_i\}$.
2) Lossily compress the view elements using selected JPEG quality factors $\{Q_i^*\}$ as described above to give compressed view elements $\{\hat{V}_i\}$.
3) Reconstruct the sequence $\hat{x}[i]$ at full-resolution from the compressed view elements $\{\hat{V}_i\}$.
4) For each original frame $x[i]$ at full-resolution, compute the error of the reconstructed frame, $\epsilon[i] = x[i] - \hat{x}[i]$.
5) Compress $\epsilon[i]$ using LZW lossless coding.

The difference sequence allows the full-resolution sequence to be reconstructed by adding the reconstructed lossy sequence to the difference sequence. Since the difference sequence does not have much energy, it compresses better without loss than the straight full-resolution sequence.

V. Video View Access

The video graph enables a number of views of each segment of the video sequence to be accessed from the compressed and stored view elements. As show in Fig. 8, the uniform video graph provides 16 views of each temporal segment of the video sequence. As show in Fig. 9, the wavelet video graph provides ten views of each segment. In order to access a requested view, the system needs to identify the view elements that intersect with the view.

A. View Element Intersection

The view elements correspond to tiles in spatial- and temporal-frequency. Each view element has a location and size in spatial-frequency $R_s = (x_s, y_s, w_s, h_s)$, and location and size in temporal-frequency $R_t = (t_t, w_t)$. As such, each requested view corresponds to a region in spatial- and temporal-frequency: $R = \{R_s, R_t\}$. The system uses this information to determine which stored view elements are needed to construct a requested view.

For each video segment, the stored view elements in both the uniform and wavelet video graphs form complete and nonredundant tilings in spatial- and temporal-frequency. Therefore, if a stored view element intersects with a requested view, it must be used in synthesizing the requested view. Given the request for a view $v_i = \{R^i\}$, the system determines the intersection of v_i with each stored view element $v_j = \{R^j\}$ as follows:

$$R^i \cap R^j = \begin{cases} 1, & R_s^i \cap R_s^j \text{ and } R_t^i \cap R_t^j \\ 0, & \text{otherwise} \end{cases} \tag{5}$$

where

$$R_s^i \cap R_s^j = \begin{cases} 0, & x_s^i \geq x_s^j + w_s^j \text{ or } y_s^i \geq y_s^j + h_s^j \text{ or} \\ & x_s^j \geq x_s^i + w_s^i \text{ or } y_s^j \geq y_s^i + h_s^i \\ 1, & \text{otherwise} \end{cases} \tag{6}$$

and

$$R_t^i \cap R_t^j = \begin{cases} 0, & x_t^i \geq x_t^j + w_t^j \text{ or } x_t^j \geq x_t^i + w_t^i \\ 1, & \text{otherwise.} \end{cases} \tag{7}$$

Once the system identifies the necessary view elements, the system processes them to construct the view. The transition paths of the video graph dictate the processing steps. Note that the view elements may undergo several steps of spatial (S) and/or temporal (T) synthesis.

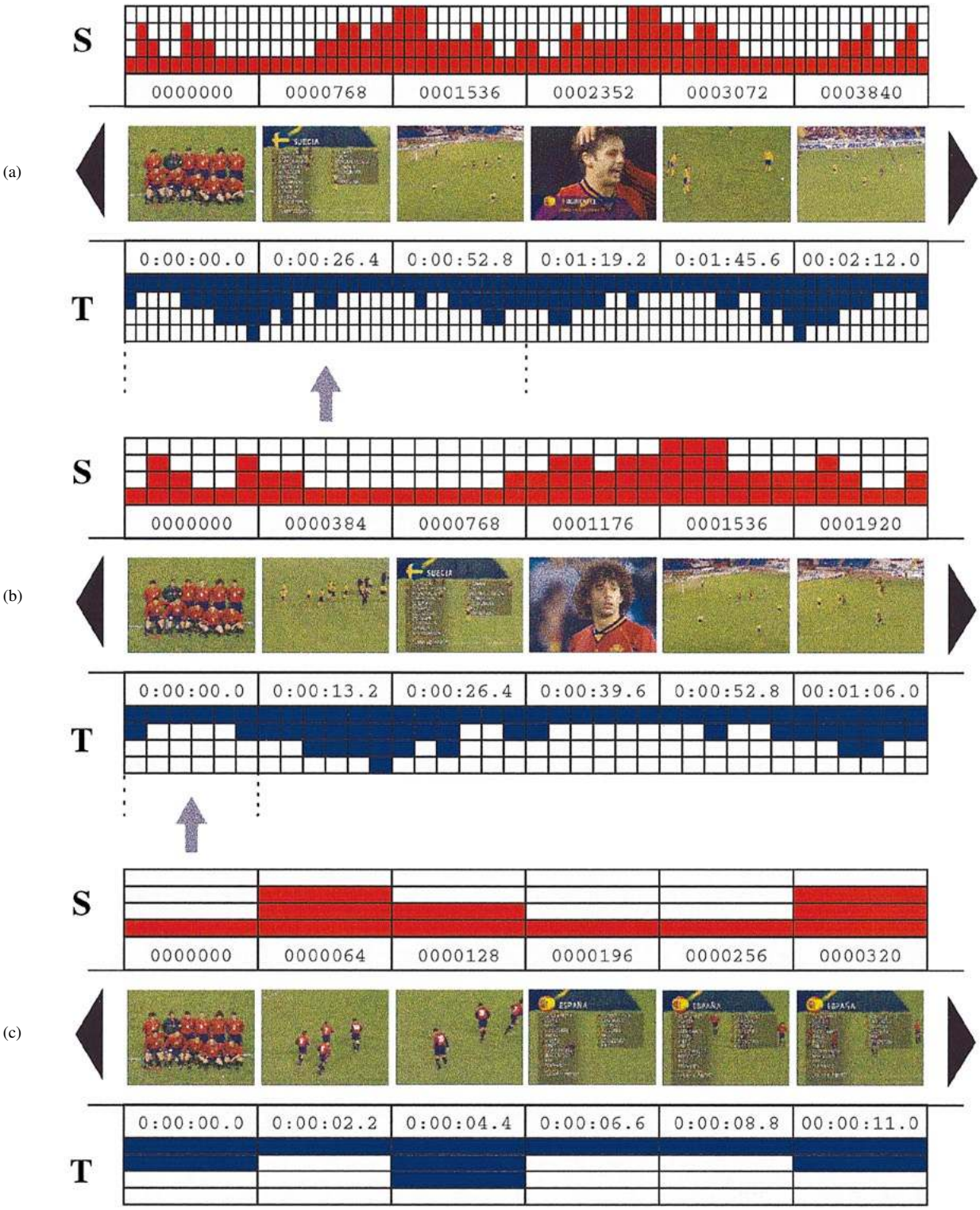

Fig. 13. In each stage above, the spatial (S) resolution of each segment is indicated by the top bar graph (d_s), and the temporal (T) resolution is indicated by the bottom bar graph (d_t). In this example, the user has selected different spatial and temporal resolutions for each segment and is drilling down in time to view the sequence.

B. View Synthesis

Fig. 10 shows how the view elements are combined to synthesize views of the soccer sequence. By combining C with residual view elements E, the system zooms in in space to create view B. By combining C with residual D, the system zooms in in time to create view A.

VI. Progressive Video Retrieval

In a progressive retrieval session, the user initially retrieves coarse views of the video sequence. Then, as the user zooms in on different segments, the system retrieves additional view elements and synthesizes the new views. The uniform and wavelet video graphs differ in the number of coarse views and

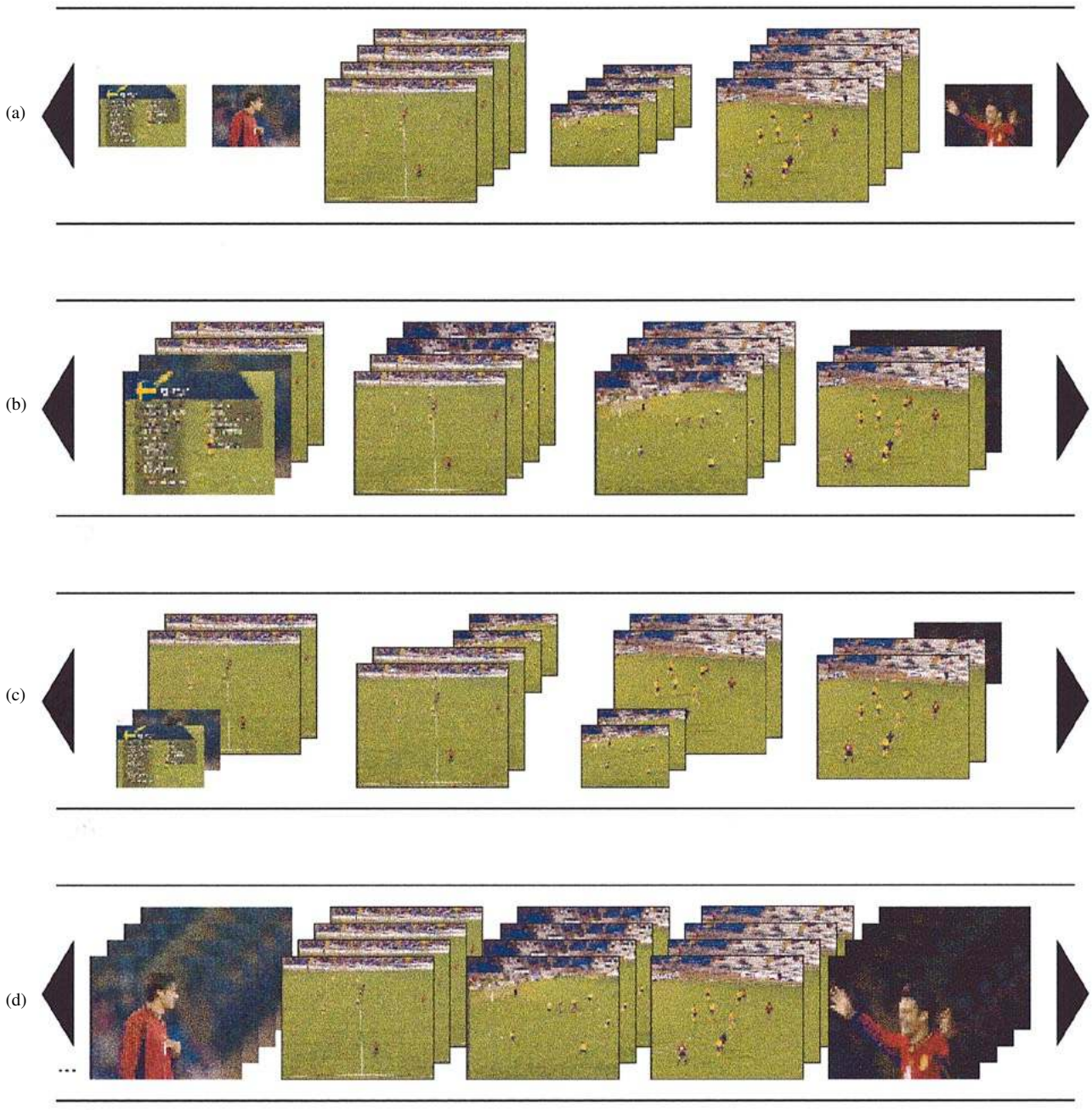

Fig. 14. VideoZoom provides three ways for playing back the video during progressive retrieval: (a) retrieved views, (b) variable rate playback, (c) variable scale playback, and (d) client-refined full-scale playback.

number of unique zooming paths they provide. The uniform video graph in Fig. 8 has 16 coarse views and provides 20 unique paths for zooming from the most coarse view to the full-resolution sequence. The wavelet video graph in Fig. 9 has ten coarse views and provides eight unique paths for zooming from the most coarse view to the full-resolution sequence.

A. Client View Element Caching

During progressive retrieval, the client application caches the view elements as the user zooms in. This allows them to be re-used in synthesizing views at higher resolutions as the user continues zooming in.

Figs. 11 and 12 shows an example progressive retrieval session using the uniform video graph. The user initially retrieves the coarse view (A), then zooms in in time (B), then space (C), then time (D, E), and then space (F, G) again to build up details of the video sequence. The retrieved view elements for this session are shown in Fig. 12. In the first zoom-in, in time, $A \rightarrow B$, the client retrieves three residual view elements from the server and combines these with coarse view A to synthesize the view B. In the second zoom, in space, $B \rightarrow C$, the client retrieves 12 residual view elements from the server and combines these with coarse view B to synthesize

the view C. We can see that B has a higher resolution in time than A and C has higher resolution in space than B.

B. Zooming Controls

Using the VideoZoom system, the user controls the zooming process by selecting the spatial and temporal resolution of each segment. The zooming controls are illustrated in Fig. 13. In each stage of progressive retrieval, the spatial (S) resolution of each segment is indicated by the top bar graph (d_s), and the temporal (T) resolution is indicated by the bottom bar graph (d_t) in Fig. 13(a)–(c). This example, the user has selected different spatial and temporal resolutions for each segment and is drilling down in time to view the sequence. Fig. 13(a) shows a view of a sequence 3840 frames. The spatial and temporal resolution of each segment of 64 frames is indicated by the resolution bar graphs. Fig. 13(b) shows a view of the first 1920 frames of the sequence at $2\times$ temporal resolution. Fig. 13(c) shows a view of the first 320 frames of the sequence at $16\times$ temporal resolution.

C. Video Playback

VideoZoom provides three ways for playing back the video during progressive retrieval. As illustrated in the example of Fig. 14(a), the user has retrieved a number of views of the video sequence at different spatial and temporal resolutions. In variable rate playback, VideoZoom scales the views at the client to full spatial resolution and plays the frames back with a fixed temporal delay between each frame, as illustrated in Fig. 14(b). With variable rate playback, the effect is to speed-up the sequence where temporal details are missing and slow-down the sequence where there are more temporal details. In variable scale playback, the VideoZoom does not scales the views at the client to full spatial resolution, as illustrated in Fig. 14(c). Finally, with client-refined full-scale playback, VideoZoom scales the views to full resolution in space and scale without the missing detailed view elements, as illustrated in Fig. 14(d). The effect is to playback the video at full scale and rate but at lower viewing quality due to missing view elements.

VII. Summary

The VideoZoom system allows the browsing and interactive retrieval of video sequences over the Internet at multiple spatial and temporal resolutions. The video sequences are decomposed into hierarchies of video view elements, which are retrieved in a progressive fashion. The client browser builds the views of the video sequences by retrieving, caching, and assembling the view elements, as needed. This allows the user to quickly browse the video over the Internet by starting with coarse, low-resolution views and by efficiently and selectively zooming in along the temporal and spatial dimensions. VideoZoom is suitable for digital video libraries and a number of other applications in which streaming methods provide insufficient quality of video, video downloading introduces large latencies, and generating video summaries is difficult or not well integrated with video retrieval tasks.

References

[1] S. Jacobs and A. Eleftheriadis, "Real-time video on the Web using dynamic rate shaping," in *Proc. IEEE Int. Conf. Image Processing (ICIP),* Santa Barbara, CA, Oct. 1997.

[2] NGI, "Next generation internet initiative," Office Comput., Inform. Commun., Tech. Rep. NGI Concept paper, July 1997.

[3] J.-P. Delaboudinire *et al.*, "EIT: Extreme-ultraviolet imaging telescope for the SOHO mission," *Solar Phys.,* vol. 162, nos. 1/2, pp. 291–312, Dec. 1995.

[4] H. Wactlar, M. G. Cristel, Y. Gong, and A. G. Hauptman, "Lessons learned from building a terabyte digital video library," *IEEE Comput. Mag.,* vol. 32, no. 2, pp. 66–73, Feb. 1999.

[5] W. Lei, S. Gauch, J. Gauch, and K. M. Pua, "Vision: A digital video library," in *Proc. ACM Int. Conf. Digital Libraries,* 1996, pp. 19–27.

[6] W. Wolf, Y. Liang, M. Kozuch, H. Yu, M. Phillips, M. Weekes, and A. Debruyne, "A digital video library on the World Wide Web," in *Proc. ACM Int. Conf. Multimedia (ACMMM),* Nov. 1996, pp. 433–434.

[7] J. R. Smith, "Digital video libraries and the Internet," *IEEE Commun. Mag.,* special issue on the Next Generation Internet, vol. 37, no. 1, pp. 92–99, Jan. 1999,

[8] M. Flickner, H. Sawhney, W. Niblack, J. Ashley, Q. Huang, B. Dom, M. Gorkani, J. Hafner, D. Lee, D. Petkovic, D. Steele, and P. Yanker, "Query by image and video content: The QBIC system," *IEEE Computer,* vol. 28, no. 9, pp. 23–32, Sept. 1995.

[9] S. W. Smoliar and H. Zhang, "Content-based video indexing and retrieval," *IEEE Multimedia Mag.,* pp. 62–72, Summer 1994.

[10] J. R. Smith and S.-F. Chang, "An image and video search engine for the World-Wide Web," in *In Storage and Retrieval Storage & Retrieval for Still Image and Video Databases V,* I. K. Sethi and R. C. Jain, Eds., *Proc. SPIE*, vol. 2670, pp. 84–95, Feb. 1997.

[11] S.-F. Chang, W. Chen, H. Meng, H. Sundaram, and D. Zhong, "VideoQ: An automated content based video search system using visual cues," in *Proc. ACM Int.. Conf. Multimedia (ACMMM),* Nov. 1997.

[12] A. Nagasaka and Y. Tanaka, "Automatic video indexing and full-video search for object appearances," in *Visual Database Systems II,*, E. Knuth and L. M. Wegner, Eds. Amsterdam, The Netherlands: Elsevier, 1992, pp. 113–127.

[13] E. Hwang and V. S. Subrahmanian, "Querying video libraries," *J. Vis. Commun. Image Represent.,* vol. 7, no. 1, pp. 44–60, Mar. 1996.

[14] E. Oomoto and K. Tanaka, "OVID: Design and implementation of a video-object database system," *IEEE Trans. Knowl. Data Eng.,* vol. 5, Aug. 1993.

[15] M. Davis, "Media streams: An iconic language for video annotation," *Telektronik 4.93: Cyberspace,* 1993.

[16] B.-L. Yeo and M. M. Yeung, "Retrieving and visualizing video," *Commun. ACM,* vol. 40, no. 12, pp. 43–52, Dec. 1997.

[17] M. M. Yeung and B.-L. Yeo, "Video visualization for compact presentation and fast browsing of pictorial content," *IEEE Trans. Circuits Syst. Video Technol.,* vol. 7, pp. 771–785, Oct. 1997.

[18] D. Zhong, H. Zhang, and S.-F. Chang, "Clustering methods for video browsing and annotation," in *IS&T/SPIE Symp. Electronic Imaging: Science and Technology—Storage & Retrieval for Image and Video Databases IV,* San Jose, CA, Feb. 1996, pp. 239–246.

[19] B.-L. Yeo and B. Liu, "Rapid scene analysis on compressed video," *IEEE Trans. Circuits Syst. Video Technol.,* vol. 5, pp. 533–544, Dec. 1995.

[20] J. Meng, Y. Juan, and S.-F. Chang, "Scene change detection in a MPEG compressed video sequence," in *Proce. IS&T/SPIE Symp. Electronic Imaging: Science & Technology,* San Jose, CA, Feb. 1995.

[21] G. Ahanger and T. D. Little, "A survey of technologies for parsing and indexing digital video," *J. Vis. Commun. Image Represent.,* vol. 7, no. 1, pp. 28–43, Mar. 1996.

[22] O. Avaro, P. Chou, A. Eleftheriadis, C. Herpel, and C. Reader, "The MPEG-4 systems and description languages: A way ahead in audio visual information representation," *Signal Process.: Image Commun.,* Special Issue on MPEG-4, vol. 4, no. 9, pp. 385–431, May 1997.

[23] J. L. Mitchell, W. B. Pennebaker, C. E. Fogg, and D. J. LeGall, "MPEG video compression standard," *Digital Multimedia Standards Series.* New York,: International Thomson, 1996.

[24] C. I. Podilchuk, N. S. Jayant, and N. Farvardin, "Three-dimensional subband coding of video," *IEEE Trans. Image Processing,* vol. 4, Feb. 1995.

[25] H. J. Barnard, "Image and video coding using a wavelet decomposition," Ph.D. thesis, Tech. Univ. Delft, Delft, The Netherlands, 1994.

[26] W.-L. Hsu and H. Derin, "3-D adaptive wavelet packet for video compression," in *IEEE Proc. Int. Conf. Image Processing (ICIP),* 1995, vol. 10, pp. 602–605.

[27] A. Ortega, Z. Zhang, and M. Vetterli, "A framework for the optimization of a multiresolution remote image retrieval system," in *Proc. IEEE INFOCOM,* Toronto, Ont., Canada, 1994, pp. 672–679, .

[28] J. R. Smith, V. Castelli, and C.-S. Li, "Adaptive storage and retrieval of large compressed images," in *Storage & Retrieval for Image and Video Databases VII,* M. M. Yeung, B.-L. Yeo, and C. A. Bouman, Eds., *Proc. SPIE*, vol. 3656, pp. 467–487, Jan. 1999.

[29] V. Castelli, L. D. Bergman, C.-S. Li, J. R. Smith, and A. Thomasian, "Progressive content-based retrieval of satellite image database through Internet," in *IEEE Tyrrhenian Int. Workshop on Digital Commun,* Ischia, Italy, Sept. 1998.

[30] J. R. Smith, "VideoZoom: A spatio-temporal video browser for the Internet," in *Multimedia Storage and Archiving Systems III,* C. C. J. Kuo, S.—F. Chang, and S. Panchanathan, Eds., *Proc. SPIE*, vol. 3527, Nov. 1998, pp. 212–222.

[31] M. Vetterli and J. Kovačević, *Wavelets and Subband Coding.* Englewood Cliffs, NJ: Prentice Hall, 1995.

[32] Y. Shoham and A. Gersho, "Efficient bit allocation for an arbitrary set of quantizers," *IEEE Trans. Acoust., Speech, Signal Processing,* vol. 36, Sept. 1988.

[33] K. Ramchandran and M. Vetterli, "Best wavelet packet bases in a rate-distortion sense," *IEEE Trans. Image Processing,* vol. 2, June 1993.

[34] G. K. Wallace, "The JPEG still picture compression standard," *IEEE Trans. Consumer Electron.,* vol. 38, Feb. 1992.

[35] W. B. Pennebaker and J. L. Mitchell, *JPEG Still Image Data Compression Standard.* New York: Van Nostrand Reinhold, 1992.

John R. Smith (S'85–M'97) received the Ph.D. degree in electrical engineering from Columbia University, New York, NY, in 1997, where he developed several content-based query systems, including VisualSEEk, SaFe, and the WebSEEk image and video search engine.

He joined the IBM T. J. Watson Research Center, Hawthorne, NY, in 1997, and is currently a Research Staff Member. He is investigating problems in multimedia and multidimensional data management, compression, access, and retrieval.

Dr. Smith received the Eliahu I. Jury award from Columbia University for outstanding achievement as a graduate student in the areas of systems communication or signal processing in 1997.