

# VideoZoom: A Spatio-temporal video browser for the Internet

John R. Smith  
IBM T.J. Watson Research Center  
30 Saw Mill River Road  
Hawthorne, NY 10532  
jrsmith@watson.ibm.com

## ABSTRACT

*We describe a system for browsing and interactively retrieving video sequences over the Internet at multiple spatial and temporal resolutions. Each video sequence is decomposed into a hierarchy of video view elements that are retrieved in a progressive fashion. The client browser builds the views of the video sequence 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 a coarse, low-resolution view and by selectively zooming-in along the temporal and spatial dimensions. We demonstrate that the video view element method is able to represent and deliver the video in a compact form while significantly speeding up the access and progressive retrieval over the Internet.*

## KEYWORDS

Video browsing, video databases, progressive retrieval, system optimization for search and retrieval, storage hierarchy, scalable approaches in storage and network delivery.

## 1 INTRODUCTION

Digital video is likely to become one of the dominant media types on the Internet since video has found important roles in many applications in education, news, science, and entertainment.<sup>Smi98</sup> However, the Internet is currently not well suited for video. Typical end-to-end data rates are not sufficient for high quality video. Furthermore, the Internet does not provide the necessary quality of service (QoS) guarantees to support real-time video transmission.<sup>JE97</sup> Some of these shortcomings are being addressed in the Internet2 and Next Generation Internet Initiative (NGII)<sup>NGI97</sup> efforts. Their objectives are to develop new technologies for the Internet infrastructure that increase network capacity and include provisions for guaranteeing QoS.

However, new methods are needed that allow more efficient browsing, retrieval and interaction with video. The predominant methods of accessing video on the Internet are video streaming and video file downloading. Neither method is well-suited for remote browsing of the video sequences. Downloading large video files results in a significant initial latency for viewing and browsing, which prevents interactivity. On the other hand, video streaming avoids the initial latency by playing the video back directly. However, the result is often of poor quality in terms of frame rate and fidelity. We propose a hybrid solution that combines elements of progressive downloading and streaming in order to address these problems.

Improving the ability to browse and interactively retrieve video over the Internet greatly benefits many applications. For example, consider an application that presents the user with a number of video sequences that result from a search in a digital video library. Ultimately, the user wants to view portions

of the video content at high-resolution. Given the options of downloading or streaming, it is difficult for the user to either quickly identify the content of most interest, or to obtain the final high-resolution content. By allowing the user to quickly retrieve coarse versions of the sequences, and by providing facilities for the user to progressively retrieve additional detail data, the user can efficiently zoom-in on the content of most interest.

This access paradigm is well suited for a variety of video content on the Internet. For example, some of the missions of National Aeronautics and Space Administration (NASA) use satellites to periodically acquire images of the earth and solar system to generate sequences over time. Many of these video sequences are published on the Internet to allow them to be studied by scientists and students all over the world. For example, the Solar and Heliospheric Observatory (SOHO) satellite uses an Extreme ultraviolet Imaging Telescope (EIT) to acquire images of the solar transition region and inner corona of the sun in four wavelengths.<sup>Dea95</sup>

Often in accessing these sequences, scientists are looking for various spatio-temporal phenomena. Downloading the sequences over the Internet is not a reasonable option for most users due to the enormous amount of data. Furthermore, video streaming does not provide sufficient quality for the video sequences to be studied. However, providing tools for interactive and progressive retrieval improves access to these video sequences. Initially, the user retrieves coarse versions of the sequences. Subsequently, the user zooms-in in space and/or time by retrieving additional information to build high-resolution views of the content of most interest. This approach minimizes the amount of data transmitted over the Internet, speeds-up access to the video sequences and allows viewing of the desired content at high-resolution.

### 1.1 Related work

Recently, approaches based on video abstraction and progressive retrieval have been investigated for improving video access. In video abstraction, summaries of the video, such as a set of extracted key-frames,<sup>Wo196,AL96</sup> or a video scene-graph,<sup>YY97</sup> are first retrieved to the user. This allows the user to visually inspect the smaller amount of abstract data before retrieving the full-resolution data. Typically, the browse data and video sequence data are coded and transmitted separately. However, in the case of progressive coding, a coarse version of the video serves to predict the full resolution data. In this way, the video is first browsed using the coarse data, then in the subsequent retrieval of the full-resolution video, less data needs to be transmitted. However, with current methods of video abstraction and progressive coding, the user has little ability to iteratively refine the retrieved video to build details of the portions of most interest.

There are a number of ways in which the user would like to refine the video sequence in progressive retrieval over the network. For example, given a coarse version of the video, the user retrieves additional data to fill in details either in space, in time, or in quality. 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. By allowing the user to selectively refine portions of the video in space or time, the user more easily zoom-ins on the details of interest.

We propose the VideoZoom system as a new framework for progressive retrieval of video over the Internet. VideoZoom decomposes each video sequence into a set of spatio-temporal view elements. The view elements are able to represent the video in a compact form while enabling fast access to views of the video with various spatial and temporal resolutions. The view element framework also allows great flexibility in progressive retrieval since the client application retrieves, caches, and assembles the view elements as needed to build details of the video in space or time.

## 1.2 Outline

In this paper we describe the VideoZoom system for storing, accessing, and progressively retrieving video sequences. In section 2 we present VideoZoom view element framework. In section 3 we describe the process for representing and compressing video sequences using view elements. In section 4 we describe the process of accessing views by selectively decompressing and assembling view elements. In section 5 we describe the process for interactively and progressively retrieving views of the video sequences by retrieving, caching and assembling view elements. Finally, in section 6, we describe the implementation of the VideoZoom browser client.

## 2 VIDEO REPRESENTATION

The underlying video representation, compression and storage framework for VideoZoom is based on a spatio-temporal transformation. The video sequence is decomposed jointly in spatial- and temporal-frequency using a separable wavelet filter bank. The spatial and temporal building blocks are integrated to form a graph-structured filter bank. Overall, the video graph generates a hierarchy of video view elements that correspond to spatial- and temporal-frequency subbands with various locations and resolutions in space and time.

The frequency analyses are performed separately on the x- and y-dimensions ( $A_X, A_Y$ ) of each video frame, respectively, and temporally across frames ( $A_T$ ). Due to separability, the order in which the analyses are performed does not affect the overall decomposition. In each unit of spatio-temporal analysis, consecutive groups of frames are reduced once on each dimension in space and twice in time, as illustrated in Figure 1. This generates an output consisting of 16 view elements per each group of four input frames. The coarse version of the video sequence generated by the spatio-temporal analysis unit has 1/4 resolution in space and 1/4 resolution in time, as depicted in the bottom-right of Figure 1.

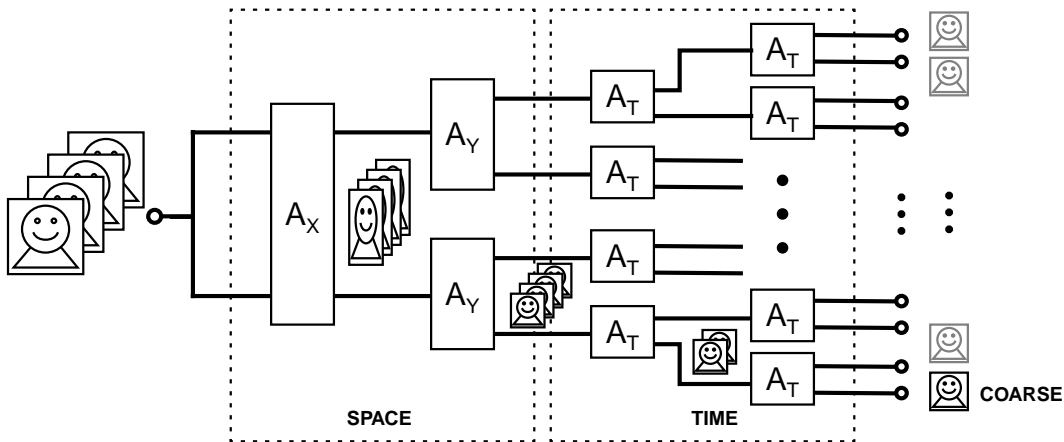


Figure 1: The spatio-temporal analysis unit decomposes the input video sequence in groups of four frames into 16 view elements, one of which is a coarse version of the group.

The spatio-temporal analysis unit depicted in Figure 1 is composed from the basic analysis building blocks illustrated in Figure 2. In general, given an input signal  $X$ , the basic analysis building block generates a coarse  $Y_0$  version of  $X$  and a residual  $Y_1$  as follows:

$$\begin{pmatrix} Y_0(z) \\ Y_1(z) \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 1/2 & -1/2 \end{pmatrix} \begin{pmatrix} 1 \\ z \end{pmatrix} X(z). \quad (1)$$

Note that the coarse signal  $Y_0$  is simply a sub-sampled version of  $X$  generated without any filtering. By combining  $Y_0$  and  $Y_1$  using the basic synthesis building block depicted in Figure 2, the input  $X$  is perfectly reconstructed as follows:

$$\hat{X}(z) = (1 \ z^{-1}) \begin{pmatrix} 1 & 0 \\ 1 & -2 \end{pmatrix} \begin{pmatrix} Y_0(z^2) \\ Y_1(z^2) \end{pmatrix}. \quad (2)$$

We apply the basic analysis and synthesis filter bank separately to each dimension in space and time, and cascade the filter banks to further decompose the video sequence. The advantages of the filter bank depicted in Figure 2 are the efficiency of the synthesis block and its suitability for progressive retrieval. For example, given a coarse view  $Y_0$  of the video sequence, the more detailed view is generated by retrieving the additional residual data  $Y_1$ . Note that we build this view in two steps:

1. Up-sample  $Y_0$  by two and add it to a shifted, up-sampled version of itself.
2. Retrieve  $Y_1$ , multiply by  $-2$ , up-sample by two, shift by one, and add to the output of step one.

The first step itself perfectly reconstructs the even terms of  $X$  along the dimension of interest and approximates the odd terms. The second step corrects the odd terms. In this way, in an interactive environment, when the user clicks to retrieve a higher resolution view, step one is carried out immediately to approximate the view and to provide initial feedback. Then, step two, which depends on retrieving  $Y_1$  over the network, is carried out to correct the view. We contrast Figure 2 with a Haar filter bank in which the coarse version is derived from the average of the odd and even terms and the residual is derived from the difference of the odd and even terms. In the Haar case, step one above requires the same processing, however, step two needs to apply the correction to all terms in the approximate view derived in step one.

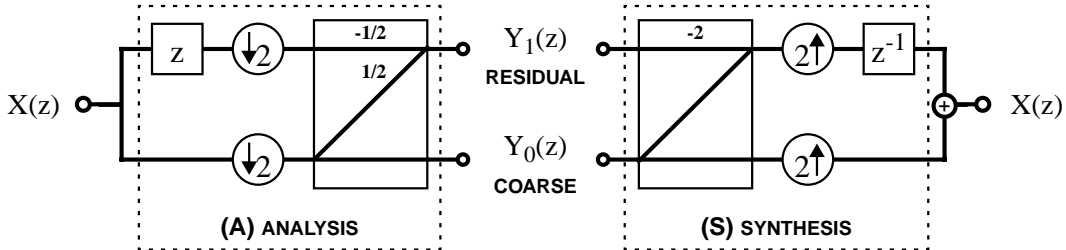


Figure 2: The basic analysis and synthesis filter bank forms the building blocks for the spatio-temporal analysis and synthesis units. In the analysis block the input is split into coarse and residual view elements. In the synthesis block the input is reconstructed from the view elements without loss.

In order to fully decompose the video sequence, we cascade the spatio-temporal analysis units to form a video graph of depth  $D$ , as illustrated in Figure 3. We derive  $D$  from the spatial size  $w_0 \times h_0$  of the original video and the desired coarse version spatial size  $w_D \times h_D$ , as follows:

$$D = \frac{1}{2} \log_2 \left( \frac{w_0 \times h_0}{w_D \times h_D} \right).$$

For example, given a video sequence with a frame size of  $w_0 \times h_0 = 512 \times 512$  and a desired coarse version size of  $w_D \times h_D = 64 \times 64$ , we generate the video graph of depth  $D = 3$  depicted in Figure 3.

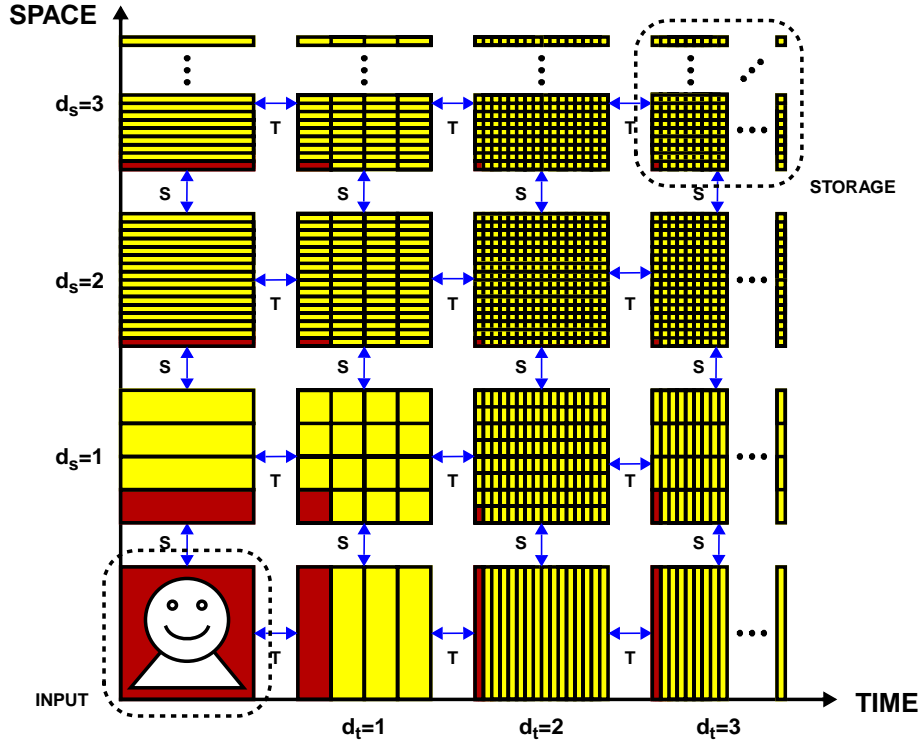


Figure 3: Example video graph with depth  $D = 3$ . In this example, the input video sequence has a size of  $(w \times h \times n)$ , where  $n$  is the number of frames. For purposes of compression and storage we represent the sequence by the 4,096 view elements in the upper right, each of which has a size of  $(w/8 \times h/8 \times n/64)$ .

The video graph in Figure 3 shows the multiple paths in which the view elements are generated by analyzing the input video along the spatial and temporal dimensions. Likewise, there are multiple paths in which the view elements are assembled to synthesize the original video sequence. Within the video graph, the darkened view elements correspond to coarse versions of the video sequence with various resolutions in space and time. Typically, these are the views that are of interest to the users. The remaining view elements are residual view elements that are used to add details in building the views.

In order to represent the video sequence, we compress and store the complete and non-redundant set of view elements with a spatial and temporal depth =  $D$  (i.e.,  $D = 3$ ). We use this set of view elements to build the views in the video graph by following the spatial and temporal synthesis paths. This set of view elements has a number of advantages for compression, access and progressive retrieval. First, we achieve good rate-distortion performance by independently compressing each of the view elements in this set. Second, in terms of access, the high granularity of the view element set minimizes the work required for synthesizing views in the video graph. Finally, in terms of progressive retrieval, the high-granularity also minimizes the amount of data transmission over the network, maximizes re-use of cached view elements at the client, and minimizes the work required for synthesizing views at the client. We address each of these features in more detail, starting with compression.

### 3 VIDEO COMPRESSION

The video compression system takes advantage of the independent coding of the view elements in order to optimize compression performance. Independent coding allows the compression system to adapt to the spatio-temporal features of the video sequence by allocating bits according to the specific rate-distortion characteristic of each view element. Given independent coding, we know that any optimal allocation of bits to the view elements requires that each of the view elements operates at the same rate-distortion trade-off.<sup>SG88</sup> In this way, we need only to search over different values of the rate-distortion trade-off in order to best satisfy the compression criteria, where the criteria are expressed in one of two ways: either maximize the fidelity for a given target bit-rate, or minimize the bit-rate for a target fidelity.<sup>VK95</sup>

#### 3.1 Compression algorithm

Currently, in VideoZoom we use JPEG<sup>Wa192</sup> to provide the underlying coding of the view elements. For coding the high-frequency, or residual, view elements, we modify the JPEG quantization matrices to have the same scale factor for all of the terms of the  $8 \times 8$  DCT.<sup>PM92</sup> The compression algorithm consists of the following steps:

1. We use the video graph to generate the set of view elements with spatial depth  $d_s = D$  and temporal depth  $d_t = D$ .
2. We compress each view element a number of times using JPEG with different quality factors. For each view element  $V_i$  and quality factor  $Q_j$ , we compute the size of the compressed data  $R_{ij}$  and the resulting distortion  $D_{ij}$  of the decompressed view element to obtain the triple  $(Q_{ij}, R_{ij}, D_{ij})$ .
3. We then select for each view element  $V_i$  the triple:  $(Q_i^*, R_i^*, D_i^*)$  that solves one of the following constrained optimization problems depending on the objectives of the user:
  - (a)  $\min \sum_i D_i^*$ , such that  $\sum_i R_i^* \leq R_T$  (minimize the total distortion for a target bit rate  $R_T$ ), or
  - (b)  $\min \sum_i R_i^*$ , such that  $\sum_i D_i^* \leq D_T$  (minimize the total bit-rate for a target distortion  $D_T$ ),

by iterating over values of the rate-distortion tradeoff  $\lambda$  and by solving one of the following unconstrained problems:

- (a)  $\min \sum_i (D_i^* + \lambda R_i^*)$ , such that  $\sum_i R_i^* \leq R_T$ , or
- (b)  $\min \sum_i (R_i^* + \lambda D_i^*)$ , such that  $\sum_i D_i^* \leq D_T$ , respectively.

4. Finally, we compress each view element  $V_i$  using its selected quality factor  $Q_i^*$  and store the compressed view elements.

#### 3.2 Compression evaluation

We evaluate the VideoZoom compression algorithm in compressing a sequence of solar images acquired by the EIT telescope on the SOHO satellite. Figure 4 illustrates the results of applying one unit of spatio-temporal analysis to an example sequence of four solar images. We can see in the upper-right that the analysis compacts a significant amount of the energy into the single coarse version view element (labeled  $C$ ). By combining  $C$  with the three residual view elements labeled  $A$ , VideoZoom is able to zoom-in temporally by synthesizing the view labeled  $A'$ . Alternatively, by combining  $C$  with the three residual view elements labeled  $B$ , VideoZoom is able to zoom-in spatially by synthesizing the view labeled  $B'$ .

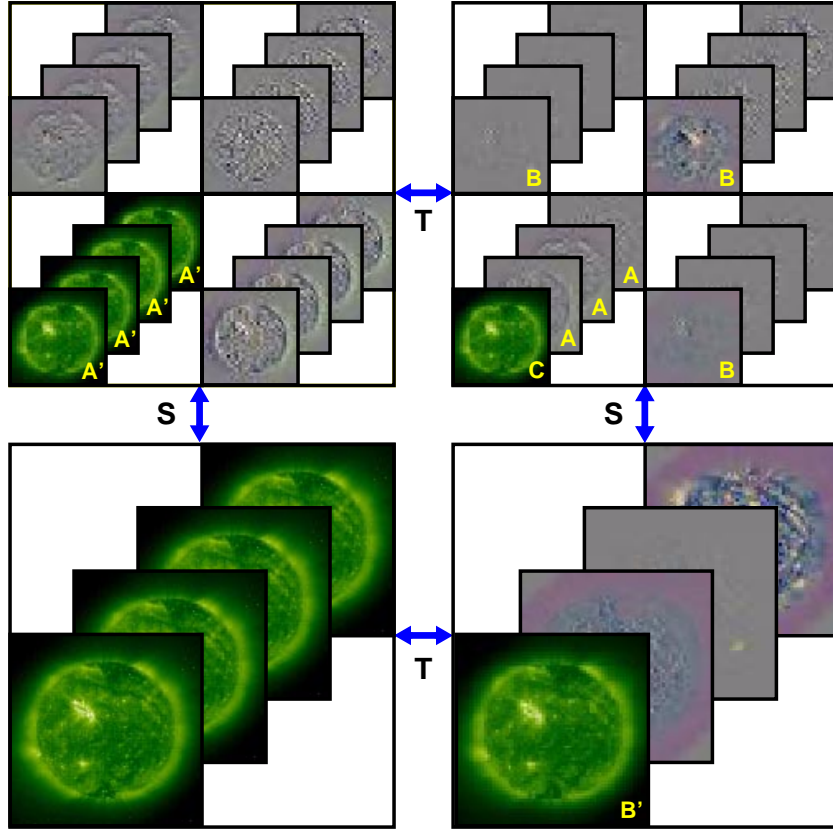


Figure 4: The results of applying one unit of spatio-temporal analysis to a sequence of four solar images.

We compare the VideoZoom compression algorithm to motion-JPEG. In motion-JPEG, each frame in the full-resolution sequence (lower-left of Figure 4) is compressed separately using JPEG. In VideoZoom, each view element in the upper-right is compressed separately using JPEG as described above. Figure 5 shows that the VideoZoom attains better compression performance than motion-JPEG. The improvement results from VideoZoom’s exploitation of temporal redundancy. For example, the temporal analysis in the video graph compacts the energy into the coarse view element, as illustrated in bottom-right of Figure 4.

#### 4 VIDEO VIEW ACCESS

VideoZoom also has advantages over motion-JPEG, MPEG<sup>Gal91</sup> and three-dimensional subband coding in terms of compressed-domain functionality. Since VideoZoom partitions the sequence into highly-granular and independent view elements, there is greater flexibility in synthesizing views. Neither motion-JPEG or MPEG provide very good compressed-domain functionality.

In both cases, the DC-terms of the  $8 \times 8$  DCTs provide coarse versions of the intra-frame coded frames. However, since each  $8 \times 8$  DCT is coded as a group, neither method makes it easy to extract views with higher spatial resolution. Typically, three-dimensional subband coding does not decompose the video sequence deeply in space or time. For example, the scheme proposed in<sup>PJF95</sup> generates 352 subbands for a block of 64 frames compared to 4,096 for VideoZoom. This results in less flexibility in generating views with different spatial and temporal resolutions.

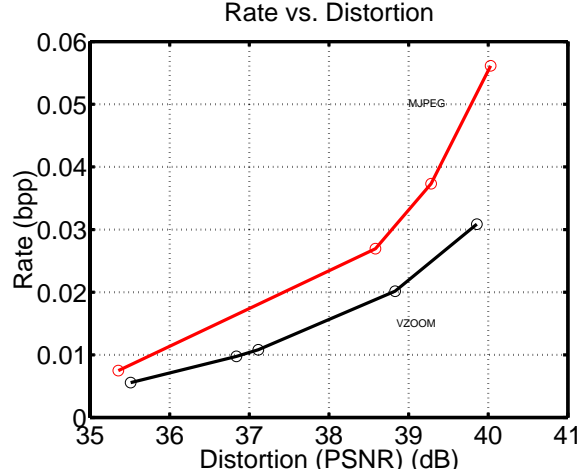


Figure 5: Comparison of rate-distortion performance of VideoZoom and motion-JPEG in compressing a sequence of  $512 \times 512$  EIT solar images.

In application, the coarse views are more likely to be retrieved since the user needs to pass through these views in order to access more the more detailed views. As a result, VideoZoom achieves greater efficiency in supporting typical access patterns by storing the set of view elements at maximum depth (i.e.,  $d_s = d_t = D$ ). For example, the view with maximum-depth in space ( $d_s = D$ ) and time ( $d_t = D$ ) is accessed with zero synthesis cost. The synthesis of other coarse views requires minimal processing of the few relevant view elements. In general, in synthesizing any view, VideoZoom does not need to process data that is irrelevant to the view.

## 5 INTERACTIVE VIDEO RETRIEVAL

The high-granularity of the stored view element set has additional benefits in progressive retrieval. First, only relevant view elements need to be transmitted over the network. That is, view elements are cached at the client, and are re-used whenever possible in building the views. Also, the high-granularity of the stored view elements allows that the accessed view elements do not contain irrelevant data. This allows VideoZoom to minimize the amount of data transmission. Second, due to the separability of the spatio-temporal synthesis, the client has great opportunity for re-using the retrieved view elements.

We examine an example interactive video retrieval session Figure 6. The user starts by retrieving the coarse view  $A$ . This coarse view is displayed to the user and is stored at the client. The user continues to zoom-in by requesting view  $B$ . In order to provide this view, VideoZoom retrieves three additional residual view elements and combines them with  $A$  using spatial synthesis. Next, the user zooms-in by requesting view  $C$ . In order to provide this view, VideoZoom retrieves three more residual view element and combines them with  $A$  using spatial synthesis. The user continues zooming-in in time and space until a full-resolution sequence in space and time is formed. In each zoom after the first coarse view has been retrieved, only residual view elements are retrieved.

## 6 IMPLEMENTATION

We are currently implementing the VideoZoom browser shown in Figure 7. The VideoZoom browser constructs a video graph cache at the client in order to facilitate the interactive retrieval process. As the user progressively retrieves the views of the video sequence, the view elements are retrieved and stored in the client’s video graph cache. As views are requested, the client application determines what

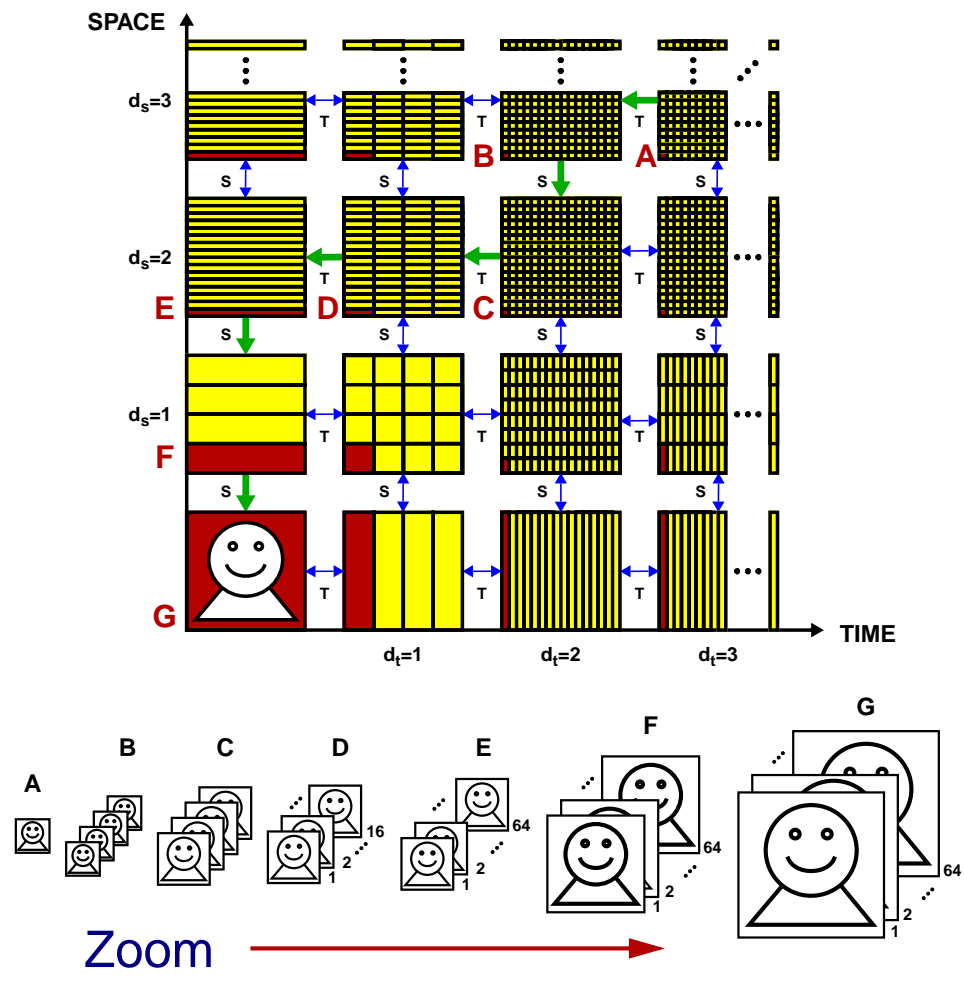


Figure 6: Example interactive video retrieval session. The user starts by retrieving the coarse view (A) and continues to zoom-in in time (B) and space (C) to build-up details of the video sequence.

additional view elements are needed to be combined with the cached view elements in order to synthesize the views.

In the VideoZoom user interface, the user is presented with two panels: the navigation panel and the viewer panel. The navigation panel allows the user to select the spatial and temporal resolution of the views. After clicking on a view, the appropriate view elements are retrieved to the client and are assembled to synthesize the view. The view is then displayed in the viewer panel. The view panel gives the user controls over the playback, pause, forward, reverse, and speed.

## 7 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

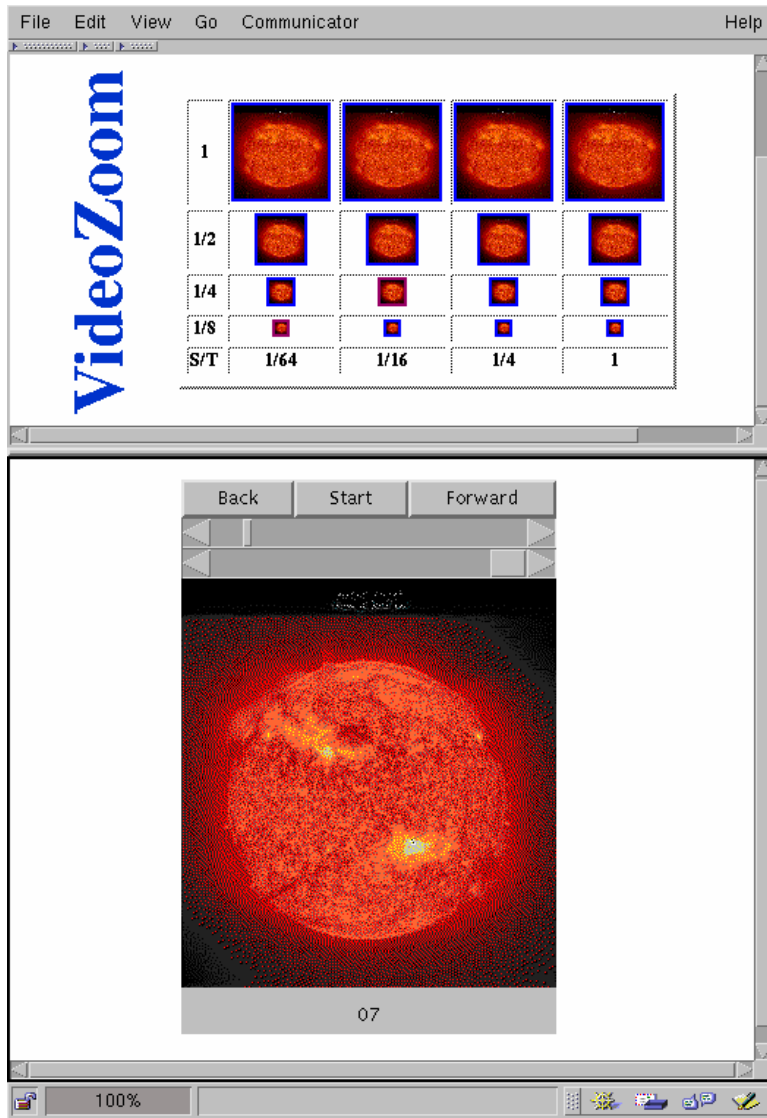


Figure 7: VideoZoom client video sequence browser.

by efficiently and selectively zooming-in along the temporal and spatial dimensions. We demonstrated that the video view element method represents the video in a compact form while significantly speeding up the access and progressive retrieval of views over the Internet.

## 8 REFERENCES

- [AL96] G. Ahanger and T. D. Little. A survey of technologies for parsing and indexing digital video. *Journ. Visual Comm. and Image Rep.*, 7(1):28 – 43, March 1996.
- [Dea95] J.-P. Delaboudinire and et al. EIT: Extreme-ultraviolet imaging telescope for the SOHO mission. *Solar Physics*, 162(1/2):291–312, December 1995.
- [Gal91] D. Le Gall. MPEG: A video compression standard for multimedia applications. *Commun. ACM*, 34(4), April 1991.
- [JE97] S. Jacobs and A. Eleftheriadis. Real-time video on the Web using Dynamic Rate Shaping. In *IEEE Proc. Int. Conf. Image Processing (ICIP)*, Santa Barbara, CA, October 1997. IEEE.
- [NGI97] NGI. Next generation internet initiative. Technical Report NGI Concept paper, Office for Computing, Information, and Communications, July 1997.
- [PJF95] C. I. Podilchuk, N. S. Jayant, and N. Farvardin. Three-dimensional subband coding of video. *IEEE Trans. Image Processing*, 4(2), February 1995.
- [PM92] W. B. Pennebaker and J. L. Mitchell. *JPEG still image data compression standard*. Van Nostrand Reinhold, New York, NY, 1992.
- [SG88] Y. Shoham and A. Gersho. Efficient bit allocation for an arbitrary set of quantizers. *IEEE Trans. Acoust., Speech, Signal Processing*, 36(9), September 1988.
- [Smi98] J. R. Smith. Digital video libraries and the Internet. *IEEE Communications Mag.*, 1998. Special issue on the Next Generation Internet, To appear.
- [VK95] M. Vetterli and J. Kovačević. *Wavelets and Subband Coding*. Prentice-Hall, Inc, Englewood Cliffs, NJ, 1995.
- [Wal92] G. K. Wallace. The JPEG still picture compression standard. *IEEE Trans. Consum. Electr.*, 38(1), February 1992.
- [Wol96] W. Wolf. Key frame selection by motion analysis. In *IEEE Proc. Int. Conf. Acoust., Speech, Signal Processing (ICASSP)*, Atlanta, GA, May 1996.
- [YY97] M. M. Yeung and B.-L. Yeo. Video visualization for compact presentation and fast browsing of pictorial content. *IEEE Trans. Circuits Syst. for Video Technol.*, 7(5):771 – 785, Oct 1997.