Video Tapestries with Continuous Temporal Zoom

Connelly Barnes[1] Dan B Goldman[2] Eli Shechtman[2,3] Adam Finkelstein[1]
[1]Princeton University [2]Adobe Systems [3]University of Washington

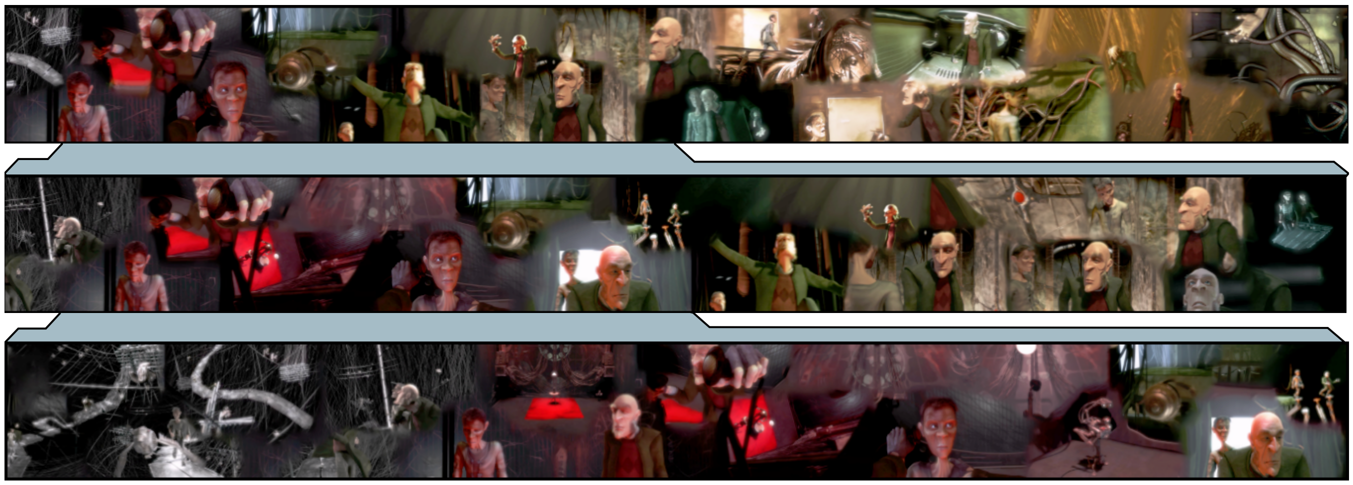


Figure 1: *A multiscale tapestry represents an input video as a seamless and zoomable summary image which can be used to navigate through the video. This visualization eliminates hard borders between frames, providing spatial continuity and also continuous zooms to finer temporal resolutions. This figure depicts three discrete scale levels for the film* Elephants Dream *(Courtesy of the Blender Foundation). The lines between each scale level indicate the corresponding domains between scales. See the video to view the continuous zoom animation between the scales. For Copyright reasons, the print and electronic versions of this paper contain different imagery in Figures 1, 4, 6, and 7.*

Abstract

We present a novel approach for summarizing video in the form of a multiscale image that is continuous in both the spatial domain and across the scale dimension: There are no hard borders between discrete moments in time, and a user can zoom smoothly into the image to reveal additional temporal details. We call these artifacts *tapestries* because their continuous nature is akin to medieval tapestries and other narrative depictions predating the advent of motion pictures. We propose a set of criteria for such a summarization, and a series of optimizations motivated by these criteria. These can be performed as an entirely offline computation to produce high quality renderings, or by adjusting some optimization parameters the later stages can be solved in real time, enabling an interactive interface for video navigation. Our video tapestries combine the best aspects of two common visualizations, providing the visual clarity of DVD chapter menus with the information density and multiple scales of a video editing timeline representation. In addition, they provide continuous transitions between zoom levels. In a user study, participants preferred both the aesthetics and efficiency of tapestries over other interfaces for visual browsing.

CR Categories: I.3.6 [Computing Methodologies]: Computer Graphics—Methodology and Techniques

Keywords: Patch-based synthesis, video summarization

1 Introduction

In recent years, the number of videos available in digital form has increased dramatically. However, due to the large volumes of data involved, it is difficult for a user to locate a scene in even a single video, or quickly comprehend the content of a video by looking at an overview. For example, a DVD film typically contains 200,000 frames, and there may even be 20 times more raw footage before it is edited into its final form [Bernstein 1994]. To help people efficiently browse, comprehend, and navigate video, both commercial software and research systems have explored how to best summarize video in the form of static images.

In the commercial domain, video navigation tools include simple scene selection interfaces, such as are found in DVD menus, and timelines that contain a sequence of images, as found in video editing software such as Adobe Premiere. In the case of DVD menus, a selection of thumbnails representing pre-selected scenes are shown to the user. In the case of timelines, one thumbnail may be shown for every shot of the video, or a sequence of thumbnails may be shown at spatial intervals corresponding linearly to the times of each scene. Each of these representations incurs a tradeoff: DVD menus are easily comprehensible, but they may be incomplete, omitting important temporal details. Video editing timelines are more complete because they contain more of the details – every shot is represented as a separate block – and they support zooming in to view more temporal detail, but they usually lack coherent transitions between zoom levels, and can be visually confusing when viewed at coarser zoom levels.

In this paper, we present a novel approach to visualize timelines and browse through scenes of a video that attempts to provide the comprehensibility of a DVD menu with the completeness and ease of navigation of a video editing timeline. Previous work in video summarization has mostly focused on selecting important frames – called "keyframes" – and arranging them in layouts with hard borders. However, such systems generally ignore one or more critical aspects of the user experience in browsing through video. First, the use of hard borders between rectangular frames

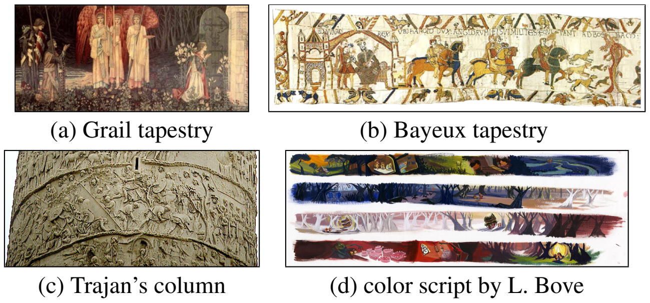

(a) Grail tapestry (b) Bayeux tapestry (c) Trajan's column (d) color script by L. Bove

Figure 2: *Our method is inspired by artistic techniques depicting action without hard borders, such as medieval tapestries (a-b), Roman friezes (c), and color scripts for cinematography (d).*

is an artifact of how film has historically been represented, and is not necessarily an optimal visual representation of video. The hard borders between frames introduce additional distracting and dominant image content, making discrete representations harder to visually parse. In contrast, our approach is inspired by the techniques that artists have historically used to depict action in a continuous manner without hard borders, in art forms such as medieval tapestries, architectural friezes, and color scripts for cinematography (Figure 2). Second, for applications like video editing, it should be possible to present a summary online, in real-time. Third, we believe users should be able to zoom in to a summary to expose fine-scale temporal details. As we have argued that abrupt discontinuities in the spatial layout are objectionable, we also believe that abrupt discontinuities in temporal zooming are also undesirable, because they can be disorienting to the viewer. Animating the transitions between scales helps the user assimilate the context and orientation of the transition [Cockburn et al. 2008]. While some previous systems satisfy some of our goals, no system satisfies all of them.

We present a unified framework for creating seamless summaries of video that allow the user to continuously zoom in to expose more temporal detail. We call these summaries *multiscale tapestries*. An example tapestry is shown in Figure 1.

Our contributions are: A set of criteria for evaluating the qualities of an optimal summary; summaries seamlessly blended across their width with implicit region-of-interest preservation; a multiscale representation with continuous zoom across temporal scales, and; rendering of tapestries in real-time with limited precomputation.

1.1 Criteria

To construct our multiscale video tapestries, we propose four important qualities that are desirable in a visual summary of video, in rough priority order:

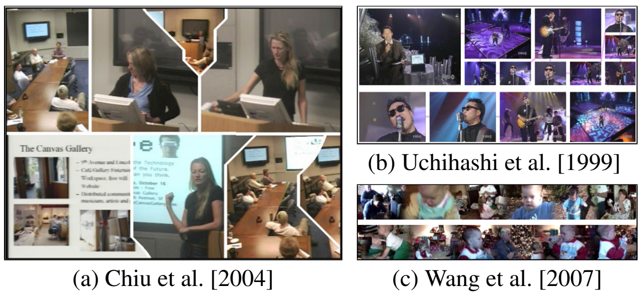

(a) Chiu et al. [2004] (b) Uchihashi et al. [1999] (c) Wang et al. [2007]

Figure 3: *Example arrangements in previous work: (a) Voronoi layout of keyframes, (b) rectangular layout, (c) soft-border collage.*

First, it should be *coherent*, depicting only visual entities that appear in the source video. This criterion motivates our desire to eliminate frame boundaries, since such structures do not exist in the source contents.

Second, it should be roughly *chronological*, presenting events in a spatial order that corresponds to their temporal order in the film. In our work as in most other such interfaces, we use a left-to-right ordering corresponding to Roman language reading order, but top-to-bottom and right-to-left orderings are trivially feasible. Although our horizontal layout doesn't fit all window shapes, it streamlines navigation by naturally mapping "left" and "right" to "before" and "after."

Third, it should be *continuous* in scale space, with visually smooth transitions from coarse to fine temporal scales.

Fourth, it should be *complete*, containing as much of the unique visual content in the source video as possible given the available pixel budget. Since our tapestries have vastly fewer pixels than the source video, we prioritize the most unique content over repeated content.

We observe that two of these desiderata for tapestries, *coherence* and *completeness*, have been previously formalized using the notion of *bidirectional similarity* (BDS), and successfully applied to the related problems of image collage and video skims [Simakov et al. 2008]. Furthermore, recent methods for approximating bidirectional similarity have made such image synthesis algorithms practical for high-quality interactive use [Barnes et al. 2009]. Thus, our system builds upon the synthesis algorithm of Simakov et al. and search algorithm of Barnes et al. as its foundations, applying additional constraints and energy terms to impose the remaining criteria of chronological ordering and scale-space continuity.

1.2 Overview

After discussing related work (Section 2), we first review the concept of bidirectional similarity in the context of a single-scale tapestry (Section 3). Our multi-scale approach is then divided into three stages, for tractability. In the first stage (Section 4.1), our system selects a set of representative keyframes at each zoom scale. In the second stage (Section 4.2), static tapestries are constructed at each scale. In the third and final stage (Section 4.3), our system interpolates zoom animations between the discrete scales. We then present implementation details and results (Section 5), and the findings of our user study (Section 6). Finally, we will revisit how our criteria are resolved in each stage, limitations, and future work (Section 7).

2 Related work

There is a large body of work on the problems of video summarization and navigation. A recent survey paper [Truong and Venkatesh 2007] comprehensively reviews these techniques, which can be broadly divided into two classes: *video skims*, and *image summaries*. Video skims [Smith and Kanade 1995; Smith and Kanade 1997; Ma and Zhang 2002; Kang et al. 2006] summarize a longer video with a shorter summary video. These skims can be useful for navigation, but they potentially require that the user observe the video for a long time, and do not naturally exploit wide screen real estate as do the timelines or scene selection interfaces that we envision. Therefore, we restrict further discussion to methods that create static image summaries.

An important early work on visualizing video is [Davis 1995]. This work used hierarchies of summarization and annotation icons to represent collections of video, and like our work was inspired by comic art and paintings. The work also introduced new video representations, such as the *videogram*, a summary image showing only the center vertical line from each frame, and time line icons: 3D stacks visualizing the spacetime volume.

Other early work on summarizing video as images focused on methods for selecting the important keyframes to be shown in the summary. For example, Dementhon et al. [1998] used curve splitting in a feature space to determine keyframes, and allowed users to subdivide finer to expose more temporal detail. Subsequent work detected keyframes by greedy and global methods that consider how much change has occurred in various feature spaces, and used clustering and coverage methods that try to ensure that the summary represents the same set of content as the original video [Truong and Venkatesh 2007]. Work on selection of keyframes also often relied on detecting shot boundaries, which is now considered a mature research area [Kraaij et al. 2004]. In the context of skeletal animation, Assa et al. [2005] utilized motion capture data to produce effective visual summaries of human actions. Our approach does use keyframes to limit computation, but employs a consistent energy function for keyframe selection and static tapestry composition. Previous methods such as [Shipman et al. 2003; Wang et al. 2007] have explored using multiple scales of temporal detail, however, our method is the first to offer continuous zooming between scales.

Other research has emphasized the importance of the composition of a static summary, examples of which are seen in Figure 3. For panoramic videos, Taniguchi et al. [1997] automatically detected panoramas and key frames, and used an adaptive method to pack the resulting images to make effective use of space. Comic-book style layouts [Uchihashi et al. 1999; Boreczky et al. 2000] employed variation in sizes and tiling patterns of rectangular frames to create more appealing summaries. For news video, collages have been used to create more interesting layouts [Christel et al. 2002]. By detecting important regions and using Voronoi diagrams, Chiu et al. [2004] created non-rectangular, arbitrary layouts which effectively pack the salient information into a limited space. For browsing into large numbers of video clips, [Kim et al. 2006] explored a mosaic representation, where each video is visualized as a small tile.

In each of these layout approaches, discrete moments in time have been separated by hard borders. Wang et al. [2007] found that users prefer video summaries with soft blending between adjacent frames over hard boundaries. In later work, the soft blending was also extended to use arbitrary shapes for the layout of the summary [Yang et al. 2008; Mei et al. 2009]. However, these methods still construct the summary by a sequence of independently motivated steps: selecting regions of interest within keyframes based on saliency maps, shot detection, and camera motion detection, extracting and resizing these keyframes, computing optimal seams, and applying blending. In contrast, we formulate the optimal summary image using just three stages with tightly related optimizations. Moreover, our method supports coherent and continuous zoom and can be computed quickly enough for interactive synthesis.

A number of tools are available for making collages and creating soft blendings for summaries. Previous work has demonstrated collage construction using graph cut and gradient domain techniques [Agarwala et al. 2004; Rother et al. 2005; Rother et al. 2006]. Optimal boundaries between collage regions can be detected by graph cuts [Kwatra et al. 2003], minimum cuts, or simply pasted in the gradient domain without optimization. More recently, Sivic et al. [2008] showed that by using a large database of images, photorealistic transitions can be made between abutting images.

Concurrently, [Correa and Ma 2010] have developed a technique for interactively creating seamless summaries of video. They extract a panoramic background plate, and compose matted foreground objects on the background. Unlike our system, their work focuses more on videos that convey specific actions, and they focus on interactive authoring and animation facilities. Our work in contrast focuses on summarizing long films with a fully automatic method, and we allow for continuous zoom.

We draw inspiration for our tapestries not only from ancient art but also from the "color scripts" used in animation production. These sketches, intended to depict the color palette of a film from beginning to end, are sometimes composed as semi-continuous strips representing multiple scenes [Hauser 2008]. See Figure 2 for one example.

Our multiscale tapestries are based on the bidirectional similarity summarization approach [Simakov et al. 2008], as it provides us with a theoretical framework for how to optimally compute a tapestry of any input video. When employed for image summarization, this method can merge repetitive or redundant structures and implicitly defines saliency via uniqueness.

3 Single-scale tapestries

For simplicity, we begin by considering an objective for tapestries at a single zoom scale. The objective function for the optimal single-scale tapestry is inspired by the work of Simakov et al. [2008], which defined an objective function to be minimized for image retargeting. This objective contains two complementary error terms, $d_{complete}$ and d_{cohere}, which ensure that every patch in the source image is found in the target image and vice versa. The complete objective is

$$d_{BDS}(S,T) = \overbrace{\frac{1}{N_S}\sum_{s\subset S}\min_{t\subset T} D(s,t)}^{d_{complete}(S,T)} + \overbrace{\frac{1}{N_T}\sum_{t\subset T}\min_{s\subset S} D(s,t)}^{d_{cohere}(S,T)} \qquad (1)$$

where S is the source, or original image, T is the target image, small rectangular image patches s and t are sampled from the source and target images, and the number of source and target patches are N_S and N_T, respectively. The patches are of fixed size: we use 7x7 patches in our implementation. The distance $D(s,t)$ is the distance in color space between these square patches: we use L_2 distance in RGB space. When retargeting, the source image S will have different dimensions than the retargeted image T. Simakov et al. [2008] also used the same energy function to *shorten* videos, by letting S and T represent video volumes of different dimensions, and s and t represent small space-time patches (boxes).

Our single-scale tapestry approach uses the same definition of optimality given by Equation 1, with the following changes to the defined terms: We start with an input video S with dimensionality three, to produce a summary image (tapestry) T with dimensionality two. We take image patches s and t, so the sum in d_{cohere} is taken over all $p \times p$ image patches in the target image T, and the sum in $d_{complete}$ is taken over all $p \times p$ image patches in *every frame* of the source video S. We wish to enforce at least a loose time ordering in the final summary, with time advancing approximately from left to right. This is effected by introducing an additional term to our distance function $D(s,t)$ between two patches s and t that maps the time dimension of the video S to the x-axis of the summary T:

$$D(s,t) = D_{color}(s,t) + \alpha(\tau_s - \beta x_t)^2 \qquad (2)$$

where τ_s denotes time in the input video, x_t denotes horizontal position in the output tapestry, α is a user parameter that controls how strictly time must increase linearly from left to right, and β is a space-time proportionality factor chosen so the right side of the tapestry T coincides with the last frame of the input video S. Note that D_{color} is the color space distances between the patches as described above.

In principle, given an input video S, the best summary T of a user-specified resolution could be found by optimizing Equation (1). However, this is in general an NP-hard problem, so efficient approximations must be introduced to solve it effectively. Simakov et al. [2008] proposed an iterative, guided algorithm that starts with an initial guess and iteratively refines T to produce a target image with minimal error. This iterative algorithm was recently accelerated by Barnes et al. [2009] to perform image synthesis at interactive rates. Our implementation relies on these approaches, but naïve application alone would still be intractable. It would be inefficient to perform the optimization on the entire input video S, because in most cases this video cannot even fit in core memory. Even with sufficient memory, the optimization would be far too slow even for a preprocessing step, much less a real-time interface.

For this reason, we split the optimization into two stages, starting at the level of entire frames and then proceeding to the finer level of image patches within frames. First, we find a subset of keyframes from the input video that optimizes a frame-level approximation of the objective (1) over the set of all input frames. This can also be viewed as clustering over the set of frames. Second, using only these keyframes as a restricted domain S we optimize Equation 1 at the level of small image patches. Since the clustering stage is the bottleneck computation, it can optionally be replaced by a simple constant-rate sampling of the input frames, or even using manual frame selection.

4 Multi-scale tapestries

Unfortunately, the formulation of single-scale tapestries described in the previous section does not incorporate our criterion of continuity between scales. Indeed, at two different scales, the subsets of keyframes chosen in the first stage may not even intersect. And even if they do, we are still faced with the challenge of interpolating smoothly between two images of different size and significantly differing content.

Thus, we must modify our objectives to handle multi-scale tapestries, and add a third stage for synthesizing intermediate tapestries between scales. In the subsections that follow we describe in detail each stage of multi-scale tapestry generation. Single-scale tapestry generation is essentially identical to the first two stages, but without the subset constraint ensuring that keyframes from each zoom level are present in the finer zoom levels.

4.1 Keyframe clustering

In our automatic clustering process, we wish to find a fixed number of key frames n_1 for the coarsest level (we use $n_1 = 24$). For finer levels $i = 2, 3, \ldots, d$, where d is the number of levels to reach a maximum temporal sampling rate (e.g., 2 fps), we increase the number of keyframes n_i geometrically ($n_i = n_{i-1} * \rho$, where $\rho = 2$ in our implementation).

Given the objective of Equation (1), we perform our clustering as an offline precomputation. Suppose we wish to choose the set of keyframes K at zoom level 1 that minimizes the objective function. To perform this clustering efficiently we need to be able to evaluate Equation (1) quickly. Let $f_1, \ldots, f_n$ represent discrete frames from the input video. We precompute an $n \times n$ asymmetric affinity matrix $\mathbf{A}$, where A_{ij} is one direction in our objective function (1):

$$A_{ij} = \frac{1}{N_{f_i}} \left(\sum_{s \subset f_i} \min_{t \subset f_j} D_{color}(s,t) + \alpha(\tau_{f_i} - \tau_{f_j})^2 \right) \tag{3}$$

D_{color} represents the color distance between s and t, and τ_{f_i} and τ_{f_j} denote the times associated with the frames i and j. Note that the key frames in K have known time values, so we use their

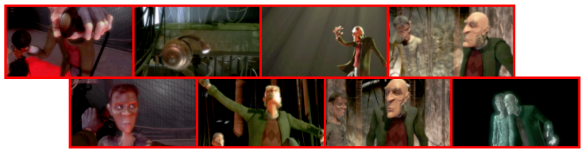


Figure 4: *The brickwork layout used as an initial guess for the optimization. Courtesy of the Blender Foundation.*

associated times directly instead of the spatial time mapping βx_t in Equation (2). Moreover, where $(\tau_{f_i} - \tau_{f_j})^2$ exceeds some threshold it will always dominate the patch color differences and therefore we can speed up the preprocessing step by omitting the computation of the affinity matrix outside a central diagonal band.

Given the affinity matrix $\mathbf{A}$ and a set of key frames $K = \{K_1, \ldots, K_m\}$, we approximate Equation (1) as follows:

$$d_{BDS}(K) = \overbrace{\frac{1}{n}\sum_{i=1}^{n} \min_{j=1\ldots m} A_{i,K_j}}^{d_{complete}(K)} + \overbrace{\frac{1}{m}\sum_{i=1}^{m} \min_{j=1\ldots n} A_{K_i,j}}^{d_{cohere}(K)} \tag{4}$$

The second term $d_{cohere}(K)$ is zero, as the diagonal elements of $\mathbf{A}$ are zero: each of the chosen key frames also exists in the set of input frames. Note that this is only an upper bound on the true objective, as for $d_{complete}$ the min operator in Equation 3 is taken only over the domain of a single keyframe rather than the collection of keyframes.

We employ the k-medoids clustering algorithm [Berkhin 2002] to minimize this objective. The key frames at each level are found by taking the known key frames at the previous level, and solving for the best new key frames to add to the layout, while existing frames are not removed or changed. Thus the key frames at each level are a subset of the frames at the next level.

This automated keyframe selection process is usually effective at highlighting salient information, but it can optionally be replaced with simple sub-sampling in time (e.g. 2 fps), or the user can manually choose key frames to tell the story better. Uniform subsampling offers the advantage that the final tapestry proceeds approximately linearly with time, which may be desirable for video editing applications where the duration of events is important. Moreover, it requires no special processing. Manual selection has the benefit of human higher-level comprehension of characters, plot, and story arc. We have found the some combination of these works best: In our highest-quality results, we perform automatic clustering, but then a user manually adjusts the key frames slightly to create a tapestry with improved composition and more semantically meaningful contents. Figure 8 compares the results from these three approaches.

4.2 Discrete tapestries

Given the key frames, we solve for the optimized tapestry independently at each scale by minimizing the objective in equation (1) using the method of Simakov et al. [2008], which takes downhill steps in the objective function (this time at the patch level, as opposed to the frame level as in the previous section). We use a brickwork layout as our initial guess: the key frames are arranged in two rows, in increasing temporal order, as shown in Figure 4. Then we retarget this input image by a factor of 75% in the y direction, encouraging repetitive structures to merge and seamless blending to occur. This process was found through experimentation; we tried scaling in x and even more in y, and found that this process makes the best compromise between saving space and preserving important structures. Note that although we use a particular layout as our initial guess, only patches from the keyframes themselves are used in the S term, so the regions of the initial guess which overlap multiple keyframes will be blended or compressed away by the retargeting process.

This method successfully condenses the tapestry, eliminating redundant image features, condensing repeating elements, and blending image regions seamlessly. However, it can sometimes destroy high-level semantic information such as faces, so we introduce a weighting factor for each patch [Simakov et al. 2008] and increase the weight of patches that overlap faces, as detected by the method of Bourdev and Brandt [2005].

For efficiency, we compute tapestries in discrete tiles of about 500 pixels wide. Although this could in principle affect the output, we find it has little impact in practice, because the time component of the distance function in Equation 2 limits interaction between regions of the tapestry that are very far apart. We avoid seams between tiles by overlapping the output tiles, and in the retargeting process we introduce a hard constraint that the colors in the overlap region must match the previously computed tiles. The implementation of these hard constraints is similar to those of Barnes et al. [2009], except that here after each iteration of the retargeting algorithm, we composite feathered copies of the known tiles using alpha-blending to avoid abrupt visual discontinuities.

When rendering tapestries in high quality mode, all tiles for all zoom levels are precomputed. When rendering tapestries as a lower resolution interactive process, we calculate tiles on-demand as the user visits each part of the zoom hierarchy. Even when running in the lower resolution interactive mode, the retargeting process takes about 0.5 second to run on each tile, so we cache the tiles, and to avoid pauses in the user interface, we use a background thread to pre-cache tiles in a small neighborhood of the user's current position in the x direction and in scale space.

4.3 Continuous zoom

At this point we have computed discrete tapestries at a fixed set of zoom levels. To satisfy our desideratum of continuity in the animation between discrete levels, we now wish to fill in the space-time volume between each pair of the discrete tapestries in a temporally coherent way. Let us label a small tapestry at zoom level i as A and a larger tapestry at level $i+1$ as B: Our goal is to complete the space-time region between A and B.

The algorithm for filling this region proceeds as follows: First, our system identifies corresponding regions between the two discrete tapestries A and B. These regions are animated linearly across the gap between A and B. Finally, remaining regions are filled in using bidirectional similarity synthesis [Barnes et al. 2009]. The paragraphs that follow provide additional details.

Finding correspondences. In Section 4.1 we guaranteed that the frames in A are a subset of the frames in B. Thus we can find correspondences between A and B, again using the fast correspondence algorithm of Barnes et al. [2009]. This establishes dense correspondences between small square patches of size $p \times p$.

Identifying coherent corresponding regions. Next we wish to find corresponding regions between the two tapestries that are above some threshold size. We create an edge graph E and choose large connected components of at least size R. The edge graph E has vertices consisting of the coordinates in A, and the horizontal and vertical edges are set only if the correspondence field is smooth, i.e. the field vector has changed less than a threshold. This gives us large corresponding regions between the two images, which we call "islands." In the space-time region we are filling in between A and B, we use the corresponding "island" regions as hard constraints, and they are constrained to animate with constant linear velocity. Note that it may be objectionable if a keyframe changes from the top to the bottom row (or vice versa) during zooming. Therefore, we optionally run a second pass of keyframe clustering requiring contiguous sets of keyframes to be even numbers.

Retargeting. Next, given the constraints, we fill in the unconstrained parts of the space-time volume, shown in magenta in Figure 5-left. We begin at the larger image B, and iteratively retarget down to smaller resolutions until we reach the size of A. For each resolution, we repeatedly run the summarization algorithm [Simakov et al. 2008] to retarget down the current image, with the source image S in Equation (1) as the larger tapestry B. We omit the $d_{complete}$ term from the objective, because the input domain is changing as we transition to the next discrete scale. We also accelerate the process by retargeting only at the finest scale.

Boundary conditions. Because our initial guess begins at B, the boundary condition at the bottom of the space-time volume is implicitly enforced. However, we also need to ensure that the animation sequence ends at the smaller tapestry A. To accomplish this we use as our input domain S a weighted combination of patches from A and B, linearly interpolating the weighting factor so the animation converges to A at the last frame. We also grow the islands by inflating them as we approach A in the space-time volume.

An example of a temporally coherent space-time interpolation is shown in Figure 5-right. As in the previous section, we compute this online by using overlapping tiles.

5 Implementation and results

Results showing discrete tapestry scales are shown in Figures 1 and 6. The continuous zoom operation is shown in our video.

The running time and space used by our algorithm varies according to the level of approximations made. The clustering preprocess for a short film (10 mins, sampled at 2fps) takes roughly 75 minutes: 15 minutes for the actual clustering and 60 minutes to compute the square affinity matrix. For clustering frames in full-length films, the computation of the affinity matrix dominates the running time. As an approximation, the entire affinity matrix need not be computed, and only a band within a threshold distance of the main diagonal need be computed, as argued in Section 4.1. Nevertheless, even with this acceleration, precomputation of the affinity matrix requires 6 hours for a 75 minute film. Note that the clustering process is optional and can be omitted, or could be replaced with a more efficient process.

Once the keyframes are chosen, high resolution high quality tapestries can be computed offline. Each discrete tile (500 pixels wide) takes about 10 seconds to compute. The continuous zoom interpolation uses tiles of the same size, and it takes roughly another 10 seconds to compute the zoom animation between scales. Alternatively, the interface can be run at interactive rates at a somewhat lower resolution and quality. Our interactive implementation computes 500 pixel-wide tiles of the discrete tapestry in about 0.5 seconds each, and each zoom tile can be computed in about 2 seconds. As described previously, to avoid pauses in the user interface, we prefetch tiles in the neighborhood of the user's current scale space region. Therefore the user can move around the scale space interactively. We show in the accompanying video that due to the interactive rendering rates, the user can change keyframes and an updated tapestry can be recomputed interactively.

6 User study

To evaluate the effectiveness of our approach we performed a user study focusing on the task of finding an event in a familiar film.

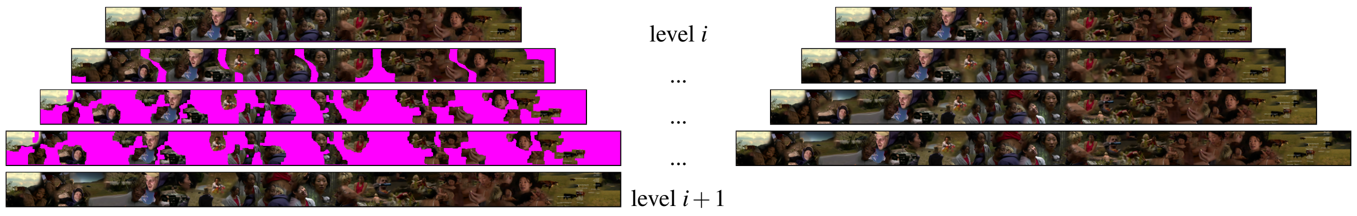
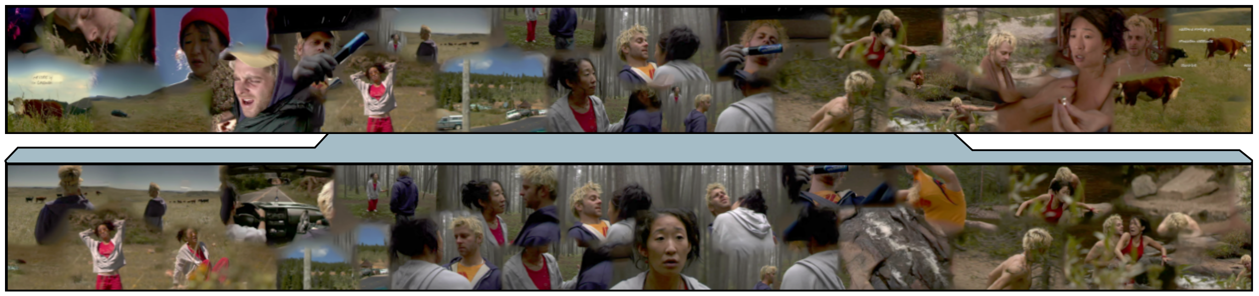
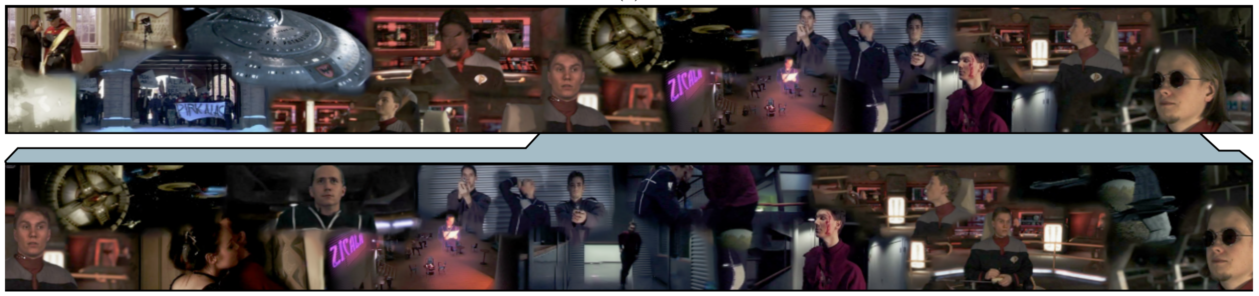


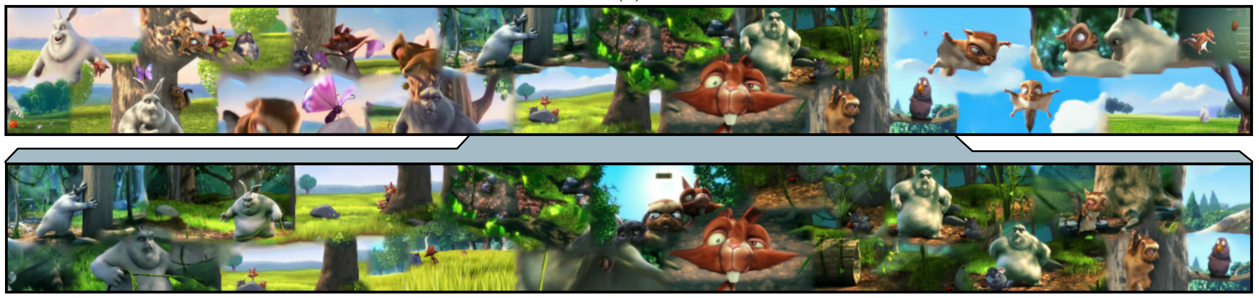
Figure 5: *Continuous zooming.* **Left:** *"Island" constraints on the animated zoom sequence between a discrete tapestry at a smaller zoom value (top) and a larger zoom value (bottom). Regions present in both images are constrained to move with constant, linear velocity. Magenta regions are unconstrained. The interpolation starts with the bottom image as initial guess, and retargets the image to smaller sizes, holding the linearly-animated islands as hard constraints. Islands expand as the smaller image is reached, enforcing convergence.* **Right:** *Results of the retargeting where the unconstrained (magenta) regions have been synthesized. The sequence zoom exhibits temporal coherence.*



(a)



(b)



(c)

Figure 6: *Discrete tapestries for various films: (a)* Kind of a Blur *(© Skunk Creek Productions), (b)* Star Wreck *(Energia Productions, Creative Commons license) and (c)* Big Buck Bunny *(Blender Foundation, Creative Commons license). The lines between each zoom level indicate the extent of the corresponding regions at each scale.*

We invited 22 individuals familiar with the films *Star Wars Episode IV: A New Hope* (1977) and *The Matrix* (1999) to find events with a visual and a temporal component, such as "the first appearance of Luke Skywalker" or "the last appearance of Obi-Wan Kenobi." Each participant was trained to use the three different interfaces shown in Figure 7: Our tapestries, a keyframe filmstrip, and a simple scrollbar. In all three cases, a preview window above the interface showed the frame corresponding to the current x location of the mouse in the interface. The training and practice session used a third film, *The Lord of the Rings: The Fellowship of the Ring* (2001). We first asked users to find events in *Star Wars* using each of the three interfaces. We next tested the importance of the zoom animation by switching it off for some tasks, replacing it with an instantaneous transition between discrete tapestries. In the final section of the study, users freely chose their favorite of the three interfaces when finding events in *The Matrix*. Users were quizzed on their familiarity with the events in *Star Wars* both before and after the study, and were given a post-study questionnaire to evaluate different interfaces.

During the study we timed the performance on each task, but found no statistically relevant difference in task performance: The average task time was about 30 seconds, but varied dramatically between individuals, probably because of differing familiarity with the subject material and comfort with video browsing interfaces in general. (In a previous pilot study we included a fourth interface

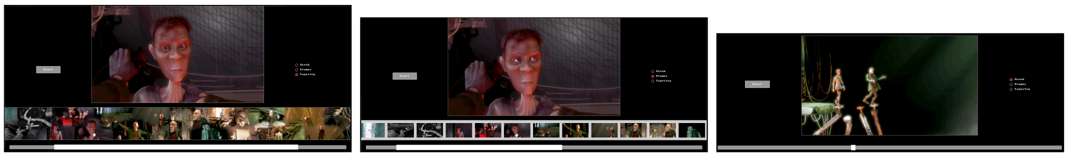

(a) Tapestries (b) Keyframe filmstrip (c) Scrollbar

Figure 7: *Three interfaces presented to users in a study. Footage from* Elephants Dream*, courtesy of The Blender Foundation*

consisting of DVD chapter menus and fast-forward/rewind controls, but task times for this interface were consistently 3 to 4 times longer than the others, so we omitted it from the final study to limit user frustration.)

However, users strongly preferred tapestries over other interfaces, both by their own evaluation and by their preferences when given free choice of interface:

- 95% of participants (21 of 22) ranked tapestries as the most aesthetically pleasing of the three interfaces, while one participant ranked the keyframe interface first.
- 68% of participants (15 of 22) found tapestries the easiest to use for the event-finding task, 23% (5 of 22) ranked keyframes the easiest, and 9% (2 of 22) ranked tapestries and other interfaces tied for ease-of-use.
- 73% of participants (16 of 22) preferred using tapestries with the continuous zoom animation instead of discrete jumps.
- 73% of participants (16 of 22) used tapestries more than other interfaces when given their choice of interface.
- 73% of participants (16 of 22) preferred tapestries as their overall favorite interface of the three. 18% (4) preferred keyframes, and 9% (2) ranked frames and tapestries as tied.

Users also gave valuable feedback for improving our interface for zooming through scale space. Some users who were most familiar with the subject material and with video browsing felt that the zoom animation was too lengthy, slowing down their search time. The zoom transition time used in our study was 1.2 seconds, but several works suggest that shorter zoom transitions between 0.3 and 1.0 seconds are more appropriate. Thus, the results shown in the video use shorter zoom transitions of .6 seconds.

7 Discussion and Future Work

In each of the stages of our algorithm, we attempt to preserve all of our four criteria described in Section 1.1. In keyframe selection, BDS is optimized under the constraints that entire frames are accepted or rejected, while chronological ordering and continuity between scales are maintained using subset constraints. In static tapestry generation, our system again optimizes for bidirectional similarity, applying an additional temporal distance term to loosely enforce chronological ordering. Here scale-space continuity is not explicitly enforced, but rather inherited implicitly from the subset constraints of keyframe selection. Finally, when constructing zoom animations, the continuity constraint is enforced by constructing and interpolating "islands," the coherence term from BDS is optimized to fill in the remaining space, and chronological ordering is inherited implicitly from the static tapestries.

It may seem to be a deficiency that our framework has no explicit cut detection preprocess as in many other common video analysis techniques. However, in professionally edited film, most cuts are intended to be invisible [Murch 1995], so we place no special significance on the location of a cut: only the similarity between frames matters. For example, if a scene contains a rapid series of shots alternating between two characters, it is plausible to summarize it using as few as one keyframe for each character, rather than a keyframe for each shot.

Keyframe selection. Our keyframe selection algorithm offers a simple theoretical solution that optimizes the same energy function as discrete tapestry synthesis. However, our system does not depend on using this keyframe selection mechanism, and indeed other keyframe selection methods may be advantageous for specific applications. For example, uniform keyframe selection offers the twin advantages of speed – no precomputation is necessary – and a roughly linear mapping from the x coordinate of the tapestry to time. Manual keyframe selection offers the advantages of human cognition and aesthetics to choose visually appealing and informative keyframes. Figure 8 shows a comparison of three tapestries generated from same source content but with three different keyframe selection mechanisms.

Failure cases. Although the bidirectional similarity synthesis algorithm eliminates and blends redundant visual content, it often happens that two adjacent keyframes in an initial layout have significantly different visual contents. In such cases, the algorithm fails gracefully by creating a blurred and feathered stitch along their boundary. Our method uses face detection as a preprocess: When this fails we can see some faces squashed in the final tapestry. But the cost of false positives is low, so we can use an extreme portion of the ROC curve to estimate face regions, and the only artifact is that some non-face regions may fail to be compacted. Nonetheless, the face detector may fail to fire on unusual faces (i.e. robots, bearded faces, cartoon faces). Face detection is not the only option for preserving structures: our method supports external identification of other important objects or regions within the video that could be supplied either manually or using other computer vision techniques (e.g. salient region detectors and object detectors).

Another extension would be to create animated tapestries, in which the region underneath the mouse begins "playing" a short sequence or multiple sequences within the tapestry in order to visualize the dynamic content of that scene. The work of [Correa and Ma 2010] achieves a similar animation effect by animating the extracted foreground objects. Other improvements to the tapestry generation could be achieved by adding a boundary compatibility term for adjacent keyframes to encourage even more contiguous tapestries.

In this work we proposed an optimization that attempts to minimize an underlying energy function by introducing several stages of approximations. In the future we would like to explore the feasibility of constructing a tapestry by minimizing a single unified objective function directly, rather than multiple successive stages with slightly different approximations.

In conclusion, we have presented a new video summarization technique that constructs spatially continuous and zoomable visualizations of video. We presented an objective function for such summaries and optimize this objective using a series of stages, computing the final stages in real-time. We hope that this technique will prove useful for video navigation, video editing, and the presentation of video search results.

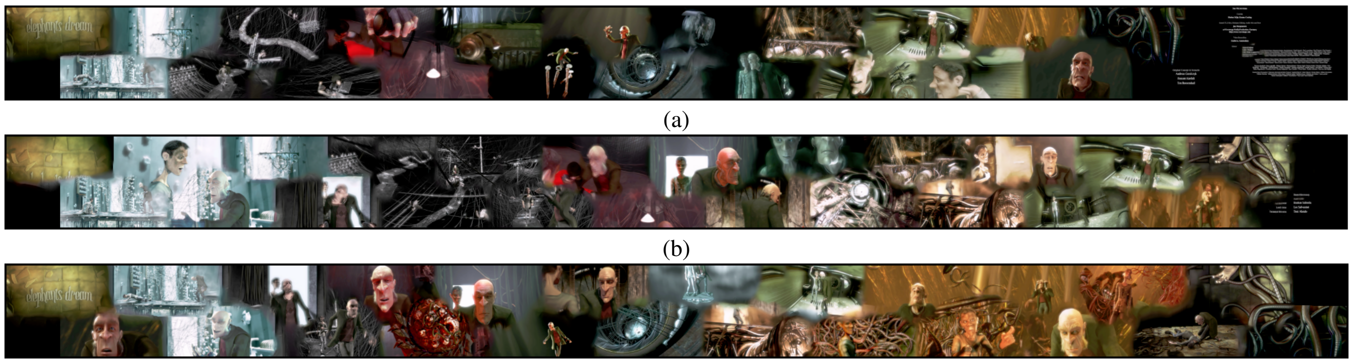


Figure 8: *Comparison of key frame selection methods for the film Elephants Dream: (a) Uniform sampling. Note that many frames are dark or otherwise contain little information. (b) Clustering. The visual information is denser and less redundant. For example, the end credits have been reduced to a single cluster, while the bright gray part on the left was expanded to include the two dominant characters (missed by the uniform sampling). (c) Manual selection. The information density is yet higher, and the characters' facial expressions are highlighted.*

Acknowledgements

We thank Lorelay Bove for the color script. The short film *Kind of a Blur* is © Skunk Creek Productions. The films *Elephants Dream* and *Big Buck Bunny* are licensed as Creative Commons by the Blender Foundation. The film *Star Wreck* is licensed as Creative Commons by Energia Productions. This work was sponsored in part by Adobe and by the NSF grant IIS-0511965.

References

AGARWALA, A., DONTCHEVA, M., AGRAWALA, M., DRUCKER, S., COLBURN, A., CURLESS, B., SALESIN, D., AND COHEN, M. 2004. Interactive digital photomontage. *ACM Trans. Graphics 23*, 3, 294–302.

ASSA, J., CASPI, Y., AND COHEN-OR, D. 2005. Action synopsis: pose selection and illustration. In *ACM Intl. Conference on Computer Graphics and Interactive Techniques*, 667–676.

BARNES, C., SHECHTMAN, E., FINKELSTEIN, A., AND GOLDMAN, D. 2009. PatchMatch: A Randomized Correspondence Algorithm for Structural Image Editing. *ACM Trans. Graphics 28*, 3.

BERKHIN, P. 2002. *Grouping Multidimensional Data: A survey of clustering data mining techniques*. Springer.

BERNSTEIN, S. 1994. *Film Production, Second Edition*. Focal Press.

BORECZKY, J., GIRGENSOHN, A., GOLOVCHINSKY, G., AND UCHIHASHI, S. 2000. An interactive comic book presentation for exploring video. In *Proceedings of SIGCHI*, ACM, 185–192.

BOURDEV, L., AND BRANDT, J. 2005. Robust object detection via soft cascade. In *IEEE CVPR 2005*, vol. 2.

CHIU, P., GIRGENSOHN, A., AND LIU, Q. 2004. Stained-glass visualization for highly condensed video summaries. In *IEEE ICME 2004*.

CHRISTEL, M., HAUPTMANN, A., WACTLAR, H., AND NG, T. 2002. Collages as dynamic summaries for news video. In *ACM Multimedia*, 561–569.

COCKBURN, A., KARLSON, A., AND BEDERSON, B. B. 2008. A review of overview+detail, zooming, and focus+context interfaces. *ACM Comput. Surv. 41*, 1, 1–31.

CORREA, C. D., AND MA, K.-L. 2010. Dynamic video narratives. *ACM Trans. Graphics 29*, 3.

DAVIS, M. 1995. *Media streams: representing video for retrieval and repurposing*. PhD thesis, Wesleyan University.

DEMENTHON, D., KOBLA, V., AND DOERMANN, D. 1998. Video summarization by curve simplifiation. In *ACM Multimedia*, 211–218.

HAUSER, T. 2008. *The Art of Wall-E*. Chronicle Books LLC.

KANG, H., MATSUSHITA, Y., TANG, X., CHEN, X., HEFEI, P., AND BEIJING, P. 2006. Space-time video montage. In *CVPR06*, 1331–1338.

KIM, K., ESSA, I., AND ABOWD, G. D. 2006. Interactive mosaic generation for video navigation. In *ACM Multimedia*, 655–658.

KRAAIJ, W., SMEATON, A., OVER, P., AND ARLANDIS, J. 2004. Trecvid 2004-an overview. In *TRECVID video retrieval online proceedings*.

KWATRA, V., SCHDL, A., ESSA, I., TURK, G., AND BOBICK, A. 2003. Graphcut textures: Image and video synthesis using graph cuts. *ACM Trans. Graphics 22*, 3, 277–286.

MA, Y., AND ZHANG, H. 2002. A model of motion attention for video skimming. In *Proc. Image Processing, Int'l Conf.*, vol. 1, I–129–I–132.

MEI, T., YANG, B., YANG, S., AND HUA, X. 2009. Video collage: presenting a video sequence using a single image. *The Visual Computer 25*, 1, 39–51.

MURCH, W. 1995. *In the Blink of an Eye: A Perspective on Film Editing*. Silman-James Press, Los Angeles.

ROTHER, C., KUMAR, S., KOLMOGOROV, V., AND BLAKE, A. 2005. Digital tapestry. In *IEEE CVPR*, I: 589–596.

ROTHER, C., BORDEAUX, L., HAMADI, Y., AND BLAKE, A. 2006. Autocollage. *ACM Trans. Graphics 25*, 3, 847–852.

SHIPMAN, F., GIRGENSOHN, A., AND WILCOX, L. 2003. Generation of interactive multi-level video summaries. In *ACM Multimedia*, 392–401.

SIMAKOV, D., CASPI, Y., SHECHTMAN, E., AND IRANI, M. 2008. Summarizing visual data using bidirectional similarity. In *CVPR 2008*.

SIVIC, J., KANEVA, B., TORRALBA, A., AVIDAN, S., AND FREEMAN, W. 2008. Creating and exploring a large photorealistic virtual space. In *IEEE CVPR Workshops, 2008.*, 1–8.

SMITH, M., AND KANADE, T. 1995. *Video skimming for quick browsing based on audio and image characterization*. Technical Report CMU-CS-95-186, School of Computer Science, Carnegie Mellon University.

SMITH, M., AND KANADE, T. 1997. Video skimming and characterization through the combination of image and language understanding techniques. In *1997 IEEE CVPR*, 775–781.

TANIGUCHI, Y., AKUTSU, A., AND TONOMURA, Y. 1997. PanoramaExcerpts: Extracting and packing panoramas for video browsing. In *ACM Multimedia*, 427–436.

TRUONG, B. T., AND VENKATESH, S. 2007. Video abstraction: A systematic review and classification. *ACM Trans. Multimedia Comput. Commun. Appl. 3*, 1, 3.

UCHIHASHI, S., FOOTE, J., GIRGENSOHN, A., AND BORECZKY, J. 1999. Video manga: generating semantically meaningful video summaries. In *ACM Multimedia*, ACM, 383–392.

WANG, T., MEI, T., HUA, X.-S., LIU, X., AND ZHOU, H.-Q. 2007. Video collage: A novel presentation of video sequence. In *ICME*, IEEE, 1479–1482.

YANG, B., MEI, T., SUN, L.-F., YANG, S.-Q., AND HUA, X.-S. 2008. Free-shaped video collage. *Multi-Media Modeling (MMM)*, 175–185.