Schematic Storyboarding for Video Visualization and Editing

Dan B Goldman[1] Brian Curless[1] David Salesin[1,2] Steven M .Seitz[1]

[1]University of Washington [2]Adobe

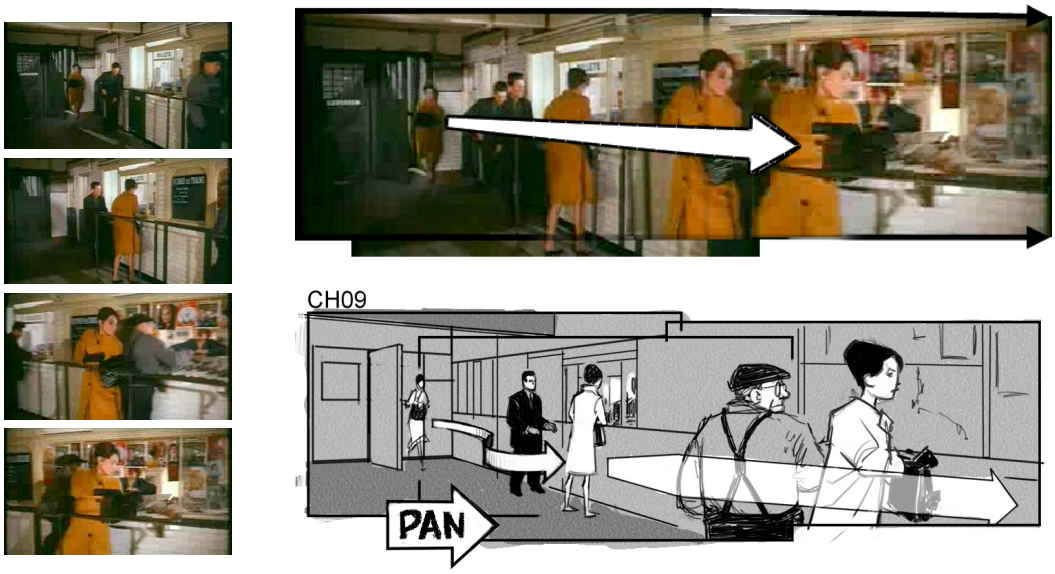

Figure 1 Left: Four still frames of a shot from the 1963 film *Charade*. Top: Schematic storyboard for the same shot, composed using the four frames at left. The subject appears in multiple locations, and the 3D arrow indicates a large motion toward the camera. The arrow was placed and rendered without recovering the 3D location of the subject. Bottom: Storyboard for the same shot, composed by a professional storyboard artist. (Credit: Peter Rubin.)

Abstract

We present a method for visualizing short video clips in a single static image, using the visual language of storyboards. These *schematic storyboards* are composed from multiple input frames and annotated using outlines, arrows, and text describing the motion in the scene. The principal advantage of this storyboard representation over standard representations of video — generally either a static thumbnail image or a playback of the video clip in its entirety — is that it requires only a moment to observe and comprehend but at the same time retains much of the detail of the source video. Our system renders a schematic storyboard layout based on a small amount of user interaction. We also demonstrate an interaction technique to scrub through time using the natural spatial dimensions of the storyboard. Potential applications include video editing, surveillance summarization, assembly instructions, composition of graphic novels, and illustration of camera technique for film studies.

CR Categories: I.3.8 [Computer Graphics]: Applications I.4.9 [Image Processing and Computer Vision]: Applications

Keywords: Storyboards, graphical representation of motion, video visualization, video interaction

1 Introduction

Video editing is a time-consuming process, in some part due to the sheer volume of video material that must be repeatedly viewed and recalled. Video editing software offers only token mnemonic assistance, representing each video segment with only a single frame, typically defaulting to the segment's first frame. But this frame is often unrepresentative of the video content, and in any case does not illustrate other important aspects of the video, such as camera and subject motion.

Indeed, this is not a new problem; since the beginning of motion pictures, it has been necessary for filmmakers to communicate with each other and their crew members about moving compositions, and they have invented a special type of diagram — the *storyboard* — to address it. Although the dictionary definition of a storyboard is just a sequence of still frames representing a moving sequence, storyboard artists have developed a distinct visual vocabulary to concisely summarize and annotate moving compositions. Filmmakers may use the term, "storyboard," to refer to either type of illustration, so we employ the term, *schematic storyboards*, specifically to describe the annotated storyboard.

Schematic storyboards are static — like a filmstrip — but organized and annotated to convey continuity and directionality — like an animation. A key advantage of schematic storyboards is that a significant time interval of the video content can be observed instantaneously. In contrast, the simple act of observing an animated display takes a certain length of time: A ten-minute shot generally takes ten minutes of a user's time to observe in its entirety. Of course, one can always fast-forward through a video (in effect scaling the data along the temporal axis) but as playback speed increases, it becomes more difficult to observe details of the motion.

In addition, schematic storyboards are particularly well-suited for

applications in which many clips must be observed and mentally processed in parallel, such as video editing. An animated display is awkward to use in such applications, since the human visual system can be quickly overwhelmed by even small numbers of video streams playing simultaneously: A rapid motion in one video may distract the observer's attention from small but important motions in another video playing simultaneously.

Finally, we employ schematic storyboards in a novel interface for *scrubbing*, or controlling the playback of the video. In most video viewing tools, a user clicks and drags on a horizontal slider to scrub forward and backward in time. But the left-to-right motion of the slider is unrelated to the motion of the camera or subject, which may move vertically, along a complex curve, or simply in the opposite direction: right-to-left. We propose the use of a storyboard as a temporal interface, such that clicking and dragging on different elements of the storyboard scrubs through time in a manner that creates the impression of directly manipulating the camera or subject.

In this work, we restrict ourselves to the problem of visualizing footage already segmented into *shots*, where a shot is a sequence of frames captured over a continuous interval of time. We have developed a system to assist the user in creating a schematic storyboard that summarizes an entire shot in a still image, using a visual representation that can communicate high-level aspects of motion of camera and subject, without requiring a moving display. As a still image, it can be printed out on a piece of paper or a booklet for reference on a soundstage or on location. It can also be used as an interface for sliding a viewer along the temporal axis of the original footage in an intuitive fashion.

The problem of automatically producing storyboards from video is a difficult and wide-ranging challenge, spanning aspects of computer vision, AI, visualization, and human-computer interface. This work addresses only a small part of that broad spectrum, interleaving some user interaction for frame selection and tracking with automated layout, annotation and compositing. Because of this user interaction, we emphasize applications such as stock footage search, in which video clips may be re-used in multiple settings over long periods of time, thus amortizing the cost of the user interaction over many uses.

Our contributions include introducing a formal summary of the visual language of storyboards as a tool for visualization, and incorporating many of these principles as part of a semi-automated layout and rendering system. Although there are many books on storyboards, we believe this is the first concise "ruleboook" for the use of storyboard annotations. We also introduce an interactive interface for moving along the time axis of a video using the natural spatial axes of a storyboard.

In Section 2 we discuss some previous work in related areas. Section 3 summarizes some of the more common notations of storyboards, and Section 4 concerns the translation of these notations into an automated system, which is described in detail in Section 5. Results and extended applications are given in Sections 6 and 7, respectively.

2 Related work

Artists and scientists employ a variety of methods to illustrate motion in static images. Cutting [2002] catalogued five distinct solutions historically employed by artists: dynamic balance, multiple stroboscopic images, affine shear/forward lean, photographic blur, and image and action lines, and Ward [1979] discusses the role of body pose in the depiction of motion. Cartoonists employ these and other tools for depicting the temporal axis, including "speedlines," adjusting the shape and size of panels, and bleeding panels off the page [McCloud 1993]. Masuch *et al.* [1999] have applied speedlines to computer animation, and Kawagishi *et al.* [2003] incorporate additional techniques such as geometric deformation. Joshi and Rheingans [2005] explored the use of speedlines to illustrate the evolution of features in volumetric simulations. Kim and Essa [2005] use these and other techniques to create a nonphotorealistic expression of motion in a video.

We have chosen the art and techniques of production storyboards as an ideal iconography for video visualization purposes. Storyboards have been used since the dawn of filmmaking [Hart 1999] to articulate and communicate concepts of image composition and scene blocking. Ours is not the first work to explicitly adopt this visual language: Nienhaus and Döllner [2003] previously adopted storyboard-style 3D arrows to depict dynamics in 3D scenes, but our work is the first to apply storyboard techniques to video visualization.

Salient Stills [Teodosio and Bender 1993; Teodosio and Bender 2005] represents one of the first works to attempt video summarization in a single image. In particular, Massey and Bender [1996] noted the applicability of Salient Stills for conveying camera and subject motion. More recently, Freeman and Zhang [2003] used stereo imaging to merge multiple frames of video into a single image as if they occupied the same space simultaneously. Agarwala *et al.* [2004] seamlessly merged multiple images using a variety of user-specified criteria, and demonstrated the application of their system to several types of time-varying scenes. A variant of this work was explored by Wexler and Simakov [2005]. Assa *et al.* [2005] considered the problem of pose or keyframe selection for composing a storyboard-like extended frame. A novel approach to reconciling multiple planar projections was presented by Zelnik-Manor *et al.* [2005]. Like our work, all these systems combine multiple moments in time into a single still image, but from a single viewpoint and without the schematic annotations of our system.

Other systems do consider multiple frames taken from different points of view into a single composite image, including multiperspective panoramas [Wood et al. 1997], multiple-center-of-projection images [Rademacher and Bishop 1998], and manifold projection [Peleg and Herman 1997], all of which create composite images with multiple viewpoints. However, these works assume static scenes and use large numbers of viewpoints to create a near-continuous change of perspective, whereas we are using a small number of keyframes of a dynamic scene.

Although it is not the focus of our work, our problem is related to that of video abstraction or summarization, which attempts to create a compact abstract (either still or animated) of a large collection of video. The literature in this topic is large, but Li *et al.* [2001] surveyed the field. [**TODO**: say more about this work?] Irani and Anandan [1998] created a system for summarizing and indexing surveillance video that shares some common goals with our work. The PanoramaExcerpts system [Taniguchi et al. 1997] summarizes large collections of video using both single frames and panoramic mosaics. Jojic *et al.* [2003] have demonstrated a user interface for video by decomposition into layers. Our work attempts to extend the expressiveness of these types of static summaries using storyboard annotations.

We assume that our video material has already been segmented into individual shots. This can be done manually, but dozens of automatic methods have also been developed [Adjeroh et al. 1997; Nicolas et al. 2004; Heng and Ngan 2001; Cheong and Huo 2001; Vlachos 2000; Lee et al. 2001].

3 The visual language of storyboards

We propose visualization of video in a single static storyboard diagram. This schematic storyboard will be designed to communicate high-level motion of the observer and observed objects, abstracting away details that may be less important for understanding motion.

At the same time, the storyboard should relate in an intuitive way to the original video. We use the term *schematic storyboard* to describe storyboards that combine both pictorial and diagrammatic elements such as arrows and frame outlines.

The rules of composition for these elements are complex and fluid, but we have identified some of the key stylistic conventions shared by several books on traditional storyboarding [Simon 2000; Begleiter 2001; Hart 1999; Block 2001; Katz 1991]. We also received assistance from Peter Rubin, a visual effects art director and former professional storyboard artist, who created the hand-drawn storyboards in this paper and refined our understanding and implementation of storyboard notation.

We attempt to formalize some of the specific techniques used by storyboard artists in the remainder of this section. Our summary spans a broad range of storyboard conventions, some of which are used only rarely. In Section 5 we will narrow our scope, applying a few key idioms to generate storyboards from video with a small amount of user input.

In the discussion that follows, we refer to the primary object of interest as the "subject" of a shot. The subject is often — but not always — in the foreground, and may be moving relative to the static environment.

Key frames. Storyboards typically depict several "key" moments in the time span of a shot. The depicted moments in time represent some or all of the following qualities:

- extrema of motion,
- "representative" poses of a subject, which are in some sense similar to a large range of the observed poses,
- clarity of expression or pose, and
- dynamic balance, suggesting the motion in progress.

Also, different objects or individuals in the scene may be depicted at different moments in time in order to more fully optimize these criteria.

Extended frame. An extended frame is an arrangement of multiple frames on the two spatial axes of a screen or page. The frames are arranged so that the background appears continuous. Typically, standard planar projections are used, but different regions of the extended frame may have different perspective projection. Changes in perspective are typically hidden in featureless regions or at architectural boundaries. Figure 1 features one such extended frame composition.

In contrast to multiperspective panoramas [Wood et al. 1997], a storyboard is intended to be viewed in its entirety at a single orientation. Therefore, even if the best alignment between multiple frames includes a rotation, all frames in an extended frame are placed on the page or screen without rotation. (One rare exception is when the camera undergoes a large and rapid change of roll angle over the duration of a shot.) See Figure 4 for a comparison of alignments with and without rotations.

Note that arranging many frames in a single extended frame may sacrifice clarity and legibility. Therefore, storyboard artists use extended frame sparingly, typically only when the camera and/or subject are moving smoothly and the image composition changes from the beginning to the end of the shot. However, extended frame compositions are split into smaller segments or even individual frames if the resulting composition would sacrifice clarity. Such a confusing composition may result from:

- poor alignment between frames,
- large scale changes between frames due to changes in focal length or motion of the camera, or
- motion of the camera or subject that "backtracks," so that distant moments in time would obscure each other.

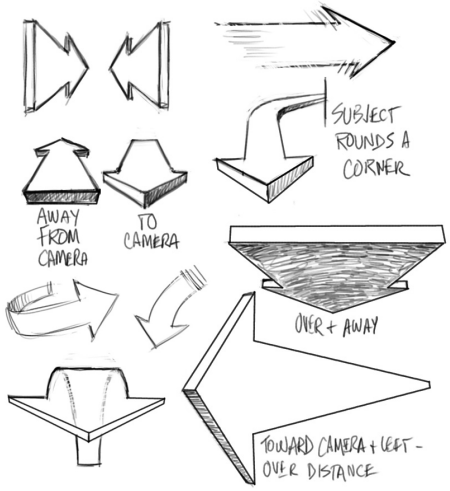

Figure 2 Arrows. A variety of arrow styles used by storyboard artists. Credit: Peter Rubin.

In addition, storyboard artists may break an extended frame into several segments in order to avoid wasted space on the page. Figure 8 uses multiple segments due to large forward motion of the camera, while Figure 10 uses multiple segments due to the subject backtracking in her path, first travelling left to right and then right to left.

Motion arrows. Storyboard artists often augment the subjects of the frames with 3D arrows that roughly follow the path of motion of the camera or subject. These arrows are usually rendered as if they were solid or semi-transparent objects in the scene itself, using different line styles or shading to distinguish the motion paths of different subjects. Motion arrows provide a more definitive sense of direction of motion than speedlines, motion blur, or some of the other mechanisms previously discussed. Furthermore, they can describe additional degrees of freedom, having both thickness and "twist" that may vary over the length of the arrow. See Figure 2 for a few of the many styles of motion arrows in storyboards. Many storyboards observe the following conventions for motion-arrow depiction:

- Arrows are piecewise smooth, emphasizing broad motion rather than small details.
- Arrows never obscure important objects in the scene.
- For subjects rotating about their direction of translation ("rolling") — e.g., a banking aircraft — the arrow twist varies over its length, and maintains alignment with the horizontal or vertical plane of the subject.
- For subjects that do not roll — e.g., a person or animal — the arrow twist may either be aligned to the subject's horizontal or vertical plane, or aligned so as to maximize the thickness of the arrow as seen from the camera.
- The arrow's width in the image is approximately proportional to the subject's size in the image. A change in perspective may be exaggerated to emphasize the motion.
- Arrows are omitted if the motion is short or self-evident.
- When multiple subjects move in the same general path, a single arrow may be used to represent their aggregate motion.
- If the subject referred to by an arrow is ambiguous, the arrow may include a textual label. (This is often the case for arrows indicating camera motion, since the camera itself is not visible.)

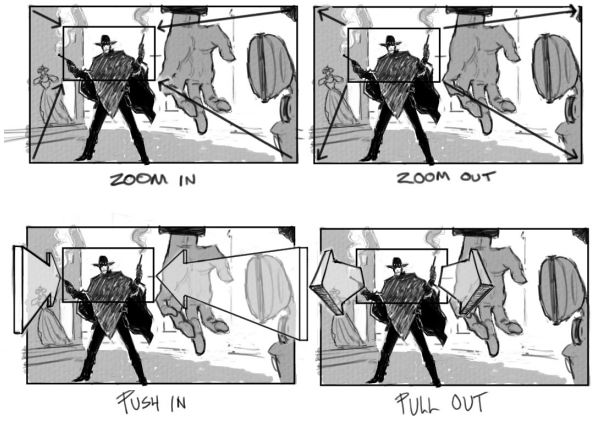

Figure 3 Zoom lines and dolly arrows. Storyboards illustrating changes in focal length (top) and camera motion (bottom). Credit: Peter Rubin.

- Arrows indicating camera motion typically cross or meet the frame boundary.

Zoom lines. Changes in focal length (zooms) are denoted by concentric sets of frame lines, using 2D arrows to indicate the direction of zoom (see Figure 3). The frame lines are typically unbroken (even if they intersect important details of the image), but the arrow heads and tails may be offset from the corners of the frames in order to improve legibility.

Depth ordering. The subjects depicted in extended frames are composed in depth order, with closer subjects appearing in front of more distant subjects, regardless of temporal ordering. Motion arrows are also rendered in depth order, unless they would be too heavily obscured by closer objects. The background of the storyboard is depicted as a continuous scene, hiding changes in perspective in featureless areas or along architectural boundaries.

Sketch style. Storyboards are typically rendered by hand using pencil or charcoal. These are often rendered quickly without significant detail, texture, or shading. Often the dominant subject is rendered in the most detail, with static background objects rendered more loosely.

4 Computer-generated schematic storyboards

Generating storyboards automatically is a broad challenge, entailing difficult problems in computer vision, non-photorealistic rendering, visualization and even activity and emotion recognition. As a step towards solving this problem, we first propose the following conceptual pipeline:

- Tracking: Determine the motion of the camera and subject.
- Segmentation: Classify pixels as belonging to a rigid background or moving subject.
- Keyframe selection: Determine frames to appear in the composition.
- Extended frame layout: Arrange the frames into one or more extended frame segments.
- Annotation layout: Determine the placement of arrows and zoom lines.
- Compositing: Combine the elements in a plausible depth ordering.
- Sketch rendering: Optionally output the results in a non-photorealistic style.

In this work we largely limit our scope to the latter four items of this pipeline, that of arranging and rendering storyboard elements in a clear and appealing composition that effectively communicates camera and subject motion. We identify and address the following challenges:

- *Frame alignment.* In a storyboard extended frame, we must create a continuous composition from multiple frames in which the center of projection varies. This is related to panoramic mosaics and multiperspective imaging, with the additional constraint that the final composition may not rotate the images with respect to the page or screen. However, since our goal is expressiveness, not realism, we may permit some distortion of objects in the scene.
- *3D annotations with unknown camera.* We must introduce 3D annotations with incomplete 3D information. The challenge here is determining the salient elements of motion; note that even when the camera is calibrated, it is not straightforward to determine the 3D trajectory of all the moving, deforming objects in the scene.
- *Maximize visual clarity.* We seek to lay out elements and render them in styles that maximize the visibility and clarity of the presentation.

Finally, we point out that while a human storyboard artist can use arbitrary visual elements to improve a composition, our system is constrained to represent images and motions that actually appeared in the input video.

In Section 5, we discuss in detail the algorithms employed to meet these goals.

5 System overview

Our system is broken into the following stages, which are performed in sequence: keyframe selection, feature tracking and labelling, extended frame layout, compositing, sketch rendering, and annotation layout. The algorithms for each stage are described in detail in subsections that follow.

Our system considers all objects in the scene to belong to either the background (rigid, possibly viewed from a moving camera), or a single subject. The case of multiple subjects is a natural extension of our framework.

5.1 Tracking and keyframe selection

In our prototype system, keyframe selection and feature tracking and labelling are performed manually: The user first chooses the frames to be included in the composition. Then she identifies corresponding points that appear in multiple frames. To assist in the segmentation phase, she also labels each such feature as belonging to a subject (moving) or background (static) object. Relatively few points are needed (about a dozen per frame). For each of the results presented in this paper, the time required for user input was approximately 5 to 10 minutes.

5.2 Extended frame layout

We first solve for an alignment of all keyframes in a single coordinate system. That is, we find transformations $\mathbf{M}_i$ for each keyframe i, such that the features are nearly aligned, under the constraint that $\mathbf{M}_i$ is comprised of a uniform scale and translation $s_i\mathbf{x}+\mathbf{t}_i$. Rotations are intentionally omitted from this transformation. Figure 4 shows the difference between a composition using rotations and a composition without rotations. Note that although the composition with rotations has better geometric alignment between features, it gives an impression that the camera is rolling with respect to the horizon, even though the camera is held at a constant orientation with respect to the horizon. The composition without rotations has poorer alignment, but it gives a more correct impression of the camera motion.

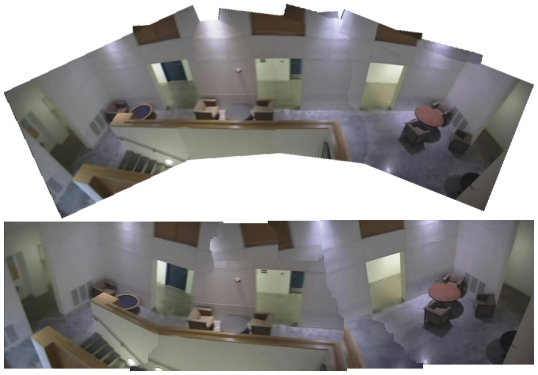

Figure 4 Top: extended frame layout with rotations. Bottom: extended frame layout with no rotations. The rotationless composition has poorer alignment but gives a more accurate impression of the camera motion.

Consider a single pair of frames i and j, with N background features. Let us denote the location of feature n in these frames as $\mathbf{f}_{in}$ and $\mathbf{f}_{jn}$. The obvious approach is to solve for the least-squares transformation between the sets of points $\mathbf{f}_i = \{\mathbf{f}_{in} \forall n\}$ and $\mathbf{f}_j = \{\mathbf{f}_{jn} \forall n\}$ using only uniform scale and translation, but this can produce degenerate solutions. Consider for example the case in which two points undergo a ninety-degree rotation with respect to the camera, appearing on a horizontal line in one frame and a vertical line in the next; the best scale between these frames in the least-squares sense is 0, even if the distance between the points is unchanged.

An alternate approach is to find correspondences with rotations, then "undo" the rotations. Our approach is simply a modification of a method due to Horn [1988] and refined by Umeyama [1991], in which we have substituted the optimal rotation R with the identity matrix. Indeed, when there is no rotation between the two sets of feature points, this transformation is the optimal least-squares uniform scale and translation between the points.

Specifically, we compute the centroids and standard deviations of the features in each frame in the standard way:

$$\bar{\mathbf{f}}_i = \sum_n \mathbf{f}_{in}/N \tag{1}$$

$$\sigma_i = \sum_n \|\mathbf{f}_{in} - \bar{\mathbf{f}}_i\|^2/N \tag{2}$$

Then the relative scale between frames i and j is computed as the ratio of the standard deviations of the feature positions, and the relative translation is computed as the difference of scaled centroids:

$$s_{i\to j} = \sigma_j/\sigma_i \tag{3}$$

$$\mathbf{t}_{i\to j} = \bar{\mathbf{f}}_j - s\bar{\mathbf{f}}_i \tag{4}$$

We denote this transformation $\mathbf{M}_{i\to j}$. (Note that all features are not visible in all frames, so for each pair of frames we recompute $\bar{\mathbf{f}}$ and σ using the subset of feature points visible in both frames.)

After computing the transformations between temporally adjacent pairs of frames, we assign each frame a transformation $\mathbf{M}_i$ in a global extended frame coordinate system. The first frame is arbitrarily assigned to lie at the origin with a scale of 1, and each successive frame is transformed by appending the transformation from the previous frame:

$$\mathbf{M}_0 = \mathbf{I} \tag{5}$$

$$\mathbf{M}_i = \mathbf{M}_{(i-1)\to i} \circ \mathbf{M}_{i-1} \tag{6}$$

This placement is not globally optimal, since small errors between pairs of frames may accumulate over the length of a sequence. But we have found these errors acceptable as long as temporally distant frames do not overlap — that is, as long as the camera does not pass over the same background multiple times.

Extended frame segments. As noted in Section 3, a single extended frame composition is not always the optimal representation for a given shot. We therefore split a shot into multiple *segments* as needed to handle both large changes in scale, and self-intersecting trajectories of the subject. Each segment is represented as a single extended frame.

Our system provides several tests to determine how to break a shot into segments. One is the *scale* test that determines if the scale ratio between the largest and smallest frame in a segment is greater than a threshhold T_s. The second is an *overlap* test that determines if the subject's path will cross over itself. The overlap test sweeps out the path of the subject in the extended frame coordinate system, failing when the percentage of pixels overlapping between successive intervals of the path is greater than a threshhold T_o. For all results shown in this paper, we used the constant threshholds $T_s = 1.5$ and $T_o = .25$ for these tests.

It is not necessary to globally optimize the frame segmentation, since there is no implicit penalty for very long or short segments. Therefore, our system employs a greedy algorithm, accumulating successive frames into a segment until one of the above tests fails, at which point the current frame becomes the first frame of a new segment. In addition, a failure of the overlap test often indicates an extremum of subject motion. Therefore, the system duplicates the last frame of the previous segment as the first frame of the new segment, so that both segments visually represent a time interval including the extremum.

When all the segments have been computed, the segments are rescaled so that the first frame j of each segment has scale $s_j = 1$.

In a typical storyboard, separate extended frame segments are laid out left to right and top to bottom. Our segment layout approach adopts the spirit of extended frame layout: We offset each segment either to the right of or below the previous segment, choosing the offset direction that is closer to the direction of camera movement. This avoids segment layouts in which the segment layout and extended frame layout proceed in opposite directions.

5.3 Segmentation and compositing

We composite the extended layout before adding other 3D annotations such as arrows.

When assembling our storyboard composite, we must balance competing goals. Where regions of background overlap, we would like to create a seamless composite, as in a panoramic mosaic. Where regions of subject appear, we would like to ensure that the subject appears in the final composite. Finally, where multiple subjects overlap, we would like them to appear in the proper depth priority, with closer subjects occluding more distant ones.

We treat compositing as a labelling problem, in which the label assigned to a pixel determines the frame from which we draw the color of that pixel. Our system first computes mattes for the subjects using the GrabCut matting algorithm [Rother et al. 2004], initialized using the bounding boxes of the subject feature points. These mattes are then transformed into the extended frame coordinate system, where they are used as hard constraints for a second graph cut optimization, using the Photomontage approach of Agarwala *et al.* [2004] to determine the locations of seams between frames. Where multiple mattes overlap, we define a depth ordering of the frames using the relative 2D size of the subject to indicate its distance to the camera. Note that this approximation for depth ordering is not generally a valid one: for example, a car viewed from the side appears smaller when viewed at the same distance from the front. However, we've found that it works well in practice.

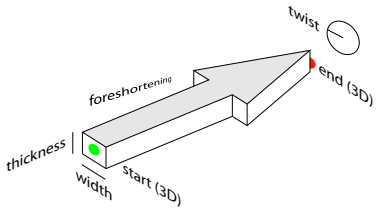

Figure 5 The ten degrees of freedom of a straight 3D arrow.

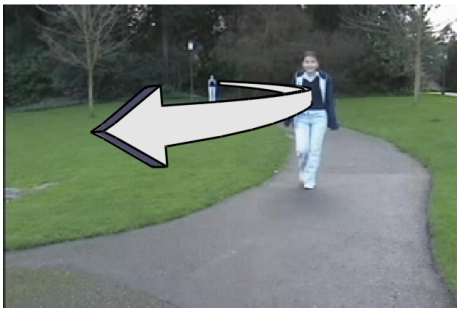

Figure 6 A motion arrow for an approaching figure.

5.4 Sketch rendering

For some applications, this literal composite of the input frames is the ideal representation of the video content. However, we also provide a non-photorealistic filter that can be applied to the composite before annotations are added, giving the image an appearance similar to a hand-drawn sketch (see Figures 9 and 10). Our NPR filter "inks" high-contrast edges using a thresholded gradient magnitude image, and shades the background using a combination of color transformations and multiplicative patterns. This representation improves the visibility of arrows and other notations under a wide variety of image contents. Since it abstracts the contents of the images, it may also be used in applications for which the animatic video content may be a proxy for the final footage. For example, in animation preproduction, two types of rough cuts provide an early guide to the dynamics of a scene: a "rip-o-matic" is a cut composed of footage from other films, and an "animatic" is a cut composed using very rough models and animation.

5.5 Annotation layout: camera motion

Camera motion within a segment is annotated using an outline of the first frame boundary and arrows leading from the corners of this outline to the corners of the boundary of the last frame of the segment. For translational motions, only two outermost arrows are drawn, so that the internal arrows do not cross over the center of the frame.

5.6 Annotation layout: subject motion

We annotate subject motion using 3D curved arrows, composited atop the extended frame. Such arrows have many degrees of freedom; Even a straight arrow with no curvature or varying twist has ten geometric degrees of freedom: three each for starting and ending position, two for breadth and thickness, and one for twist angle. In addition, each arrow could be rendered using a different amount of foreshortening, determined by the camera's focal length. (See Figure 5.)

To simplify the situation, in our current system we choose a single 3D coordinate system and camera projection for all motion arrows, with the eye point $\mathbf{E}$ at the origin, the negative z-axis pointing into the page, and a user-selected focal length.

We create a separate arrow for each frame range in a segment in which the subject is continuously visible. The arrows are placed using 3D non-uniform b-splines, with control vertices constrained by the locations of the 2D subject features. However, we do not generally have enough information about the scene to compute true three-dimensional locations of the features. Furthermore, our keyframes do not share a common 3D coordinate system. Therefore we employ a pseudo-3D estimation using the 2D distributions of the subject features in the extended frame coordinate system.

First, the 2D centroids $\bar{\mathbf{f}}$ and standard deviations σ of the subject features in each frame are computed in the global extended frame coordinate system. We assume that the dominant changes in size of the subject are due to motion towards or away from the camera. We use the standard deviation of the subject feature points as an estimate of the size of the subject in each frame. If we know the distance to the subject in one frame, we can estimate its distance in any other frame using similar triangles: $d_j/d_i = \sigma_i/\sigma_j$. Therefore, we need only to provide the distance to the subject in a single frame. We assume the subject is at its closest in the frame with the largest standard deviation, $i_{min} = \text{argmax}_i \sigma_i$. The distance $d_{i_{min}}$ of the subject at this closest frame is specified as a system constant. (This constant affects only the scale of the reconstructed scene along the z axis, and therefore does not affect the final rendering, modulo numerical precision. We used $d_{i_{min}} = 500$ for all our results.) The subject centers are approximated as the point along the ray from the camera through the feature centroid $\bar{\mathbf{f}}_i$ at distance d_i.

These estimated 3D points form the control vertices of a cubic non-uniform b-spline, using a knot vector containing multiple end knots in order to interpolate the starting and ending position. For splines with fewer than 4 control vertices, the spline degree is reduced as necessary.

The resulting 3D spline forms the backbone of the 3D arrow. In order to align the arrow geometry to this backbone, we also construct a coordinate frame at each point along the spline. As described in Section 3, a twisting arrow signifies a subject rotating around its velocity vector, and since this is an uncommon case, we presently generate arrows that do not twist along their length. Our system determines these coordinate frames as follows: Given the 3D point $\mathbf{P}_H$ and tangent $\mathbf{T}_H$ at the head of the arrow, we compute a perpendicular vector $\mathbf{U}_H = \mathbf{T}_H \times (\mathbf{E} - \mathbf{P}_H)$, where $\mathbf{E}$ is the center of projection. This vector is used to construct the normal vector everywhere along the arrow: $\mathbf{N}_t = \mathbf{U}_H \times (\mathbf{E} - \mathbf{P}_t)$. (Degeneracies of these cross products are rare, but can be handled as special cases.)

Finally, the arrow endpoints are offset from the subject positions along the spline by a small amount to improve visibility of the subjects.

The width w and thickness h of the arrows (i.e., the dimension along which the arrowhead flares, and the perpendicular dimension) are set to be linearly proportional to the standard deviation of the features at the closest frame:

$$w = \alpha \sigma_{i_{min}}$$
$$h = \beta \sigma_{i_{min}}$$

For all results shown in this paper, we used $\alpha = 0.5$ and $\beta = 0.125$. Figure 6 shows an arrow constructed by our system for an approaching figure.

5.7 Intersegment motion annotations

As described above, a shot may be split into multiple extended frame segments, either because of widely varying scales across the shot, or because of overlapping motion of the subject. For both

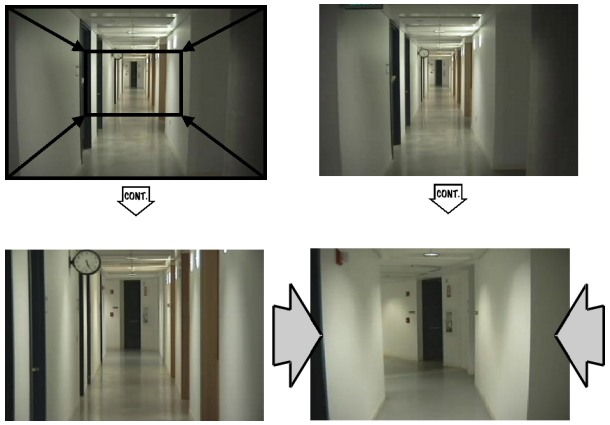

Figure 7 Left: storyboard notation for a zoom in (increasing focal length). Right: storyboard notation for a dolly in (camera traveling forward).

types of segment transitions, our system renders small arrows labelled "CONT." to indicate continuity between these segments.

Scale changes between segments can occur either because of motion into or out of the image plane (known to cinematographers and camera operators as a *dolly*), or because of a change in focal length (known as a *zoom*). At present we do not detect the difference between these camera operations, but allow the user to choose the appropriate annotations for each scale change.

Scale changes due to zoom are annotated using zoom lines. Consider two adjacent frames i and $i+1$ that have been placed in successive segments A and B. To represent a zoom-in between A and B, the system draws the outline of frames i and $i+1$ in the coordinate frame of i. The transformation between the two outlines is simply $\mathbf{M}_{i\to(i+1)}$. For a zoom-out, the procedure is similar, but this time the system draws the outlines atop frame $i+1$, using the transformation $\mathbf{M}_{(i+1)\to i} = \mathbf{M}_{i\to(i+1)}^{-1}$. Finally, the corresponding corners of the frame outlines are connected using 2D arrows (see Figure 7 left).

Our system denotes scale changes due to dollying using 3D arrows at both left and right sides of a frame. For camera motion into the plane of the image (forward motion), receding arrows are drawn on frame $i+1$ in segment B. For camera motion out of the plane of the image (backward motion), approaching arrows are drawn on frame i in segment A. The placement of these arrows is such that either the tail of the arrow touches the earlier frame or the head of the arrow touches the later frame, always pointing "forwards" in time across the transition (see Figure 7 right).

6 Results

In previous sections we have shown a few preliminary results from our system. In this section we provide several more illustrating the range of our system. A few additional results are shown in our supplementary video.

The "Running Boy" sequence is a 20-second home video taken using a handheld digital camera at 320×240 resolution, 15 frames per second, and compressed with MPEG-4. Figure 8a shows the 8 user-selected key frames, and Figure 8b shows the mattes produced automatically using GrabCut. About a dozen feature points were manually selected in each of the key frames and labeled as subject or background. Although the mattes automatically produced by the GrabCut algorithm are low quality, often cutting out large sections of the subject's body, they are sufficient to seed the graph-cut composite shown in Figure 8c. The final schematic storyboard, with 3D arrows and zoom lines overlaid atop the composite, is shown in Figure 8d.

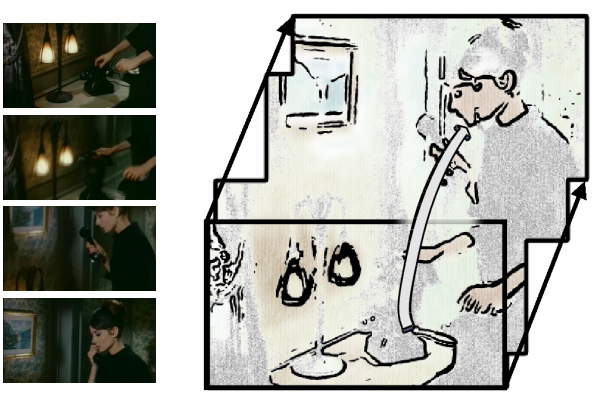

Figure 9 Schematic storyboard for the telephone sequence.

Figure 10 Four frames from the "walkleftright" sequence, and a schematic storyboard composed using these frames.

Figures 9 and 10 show two non-photorealistic examples. In Figure 10 the system has broken this scene into two extended frame segments (connected with a "CONT." arrow) since the subject crosses over her own path in the latter half of the shot.

Additional examples are shown in Figure 11. These shots were all extracted from the film *Charade*, digitized at 320×240 resolution and 30 frames per second, and compressed with MPEG-4. Users spent 5-10 minutes on each storyboard, whereas the sketches on the right of Figure 11 took a professional storyboard artist 10-30 minutes each to complete, using Corel Painter and Photoshop.

7 Video browsing and editing

One particularly compelling application for schematic storyboards is a clip interface for use in nonlinear video editing software. Such software is now widely available and affordable for consumers. But two of nonlinear video editing's fundamental interface paradigms — representing shots using a single key frame and temporal search using a timeline — have remained nearly unchanged since the earliest professional nonlinear editing systems were created a quarter-century ago [Rubin 2005]. However, schematic storyboards can be employed as an alternative to the timeline or jog/shuttle dial as a graphical interface for "scrubbing" through the time axis of a video clip.

We have developed an intuitive interaction paradigm for storyboards allowing rapid selection of moments in a video clip that leverages the spatial relationships and representation of motion that storyboards offer. Clicking on any point in the storyboard with the

Figure 8 Source frames (a) and GrabCut mattes (b) for the "Running Boy" shot. The mattes are not very accurate, but we can still use them to produce a composite (c). The complete schematic storyboard (d) includes a variety of additional annotation.

mouse displays a specific time in the video, and dragging the mouse with the mouse button depressed results in a continuous playback either forwards or backwards in time. Different parts of the storyboard invoke subtly different actions:

Motion arrows. Clicking or dragging on a subject motion arrow retrieves a frame of the video corresponding to the spline parameter at that position of the arrow. Thus, dragging towards the head of an arrow moves forward in time, while dragging towards its tail moves backwards in time.

Background pixels. Clicking or dragging on a pixel not on a motion arrow retrieves the frame of the video in which the selected pixel is closest to the center of frame. Thus dragging the mouse left or right across a pan storyboard gives the impression of dragging the camera itself left or right.

When rendering the storyboard, our system pre-renders selection buffers for these different regions and the associated temporal transformations, so that the retrieval of the selected frame occurs at interactive rates.

We have prototyped a video editing interface using this technique, enabling the user to select in and out points of a clip and drag them to a timeline in arbitrary order.

8 Discussion

We have presented a system for transforming video clips into static visualizations using the visual language of storyboards. Our contributions include: introducing the iconography of storyboards as a visualization tool for video, representation of visual motion using 3D arrows without requiring true 3D reconstruction, and an interactive interface for scrubbing video using the natural spatial relationships of a storyboard.

We believe that schematic storyboards can provide effective visualizations for presentation in both inherently static print media and dynamic digital media. In contrast with animated or filmstrip displays, a storyboard depiction allows rapid visual comprehension for a variety of tasks involving selection, comparison, and manipulation of motion data.

A wide range of tasks involving video can benefit from the concise summary of motion afforded by schematic storyboards. In a few instances, automated storyboard generation could replace illustrations presently drawn by hand. For instance, textbooks on film studies already use production storyboards to illustrate film techniques [Katz 1991], and where the original storyboards are unavailable, they may be reproduced by hand [Hart 1999]. Similarly, instruction manuals typically contain illustrations of a physical process that must clearly represent the assembly of an object or the operation of a device.

Furthermore, because storyboards are more abstract than filmstrips or animations, they can be especially useful for tasks in which multiple videos must be viewed, compared, and selected. For example, stock footage galleries can be logged and indexed using storyboards for easy reference and access.

Although the level of user effort and expertise required to create a storyboard using our system is much lower than that of a professional storyboard artist, some user effort is still required. Therefore, our system is less well-suited to situations in which the video or sto-

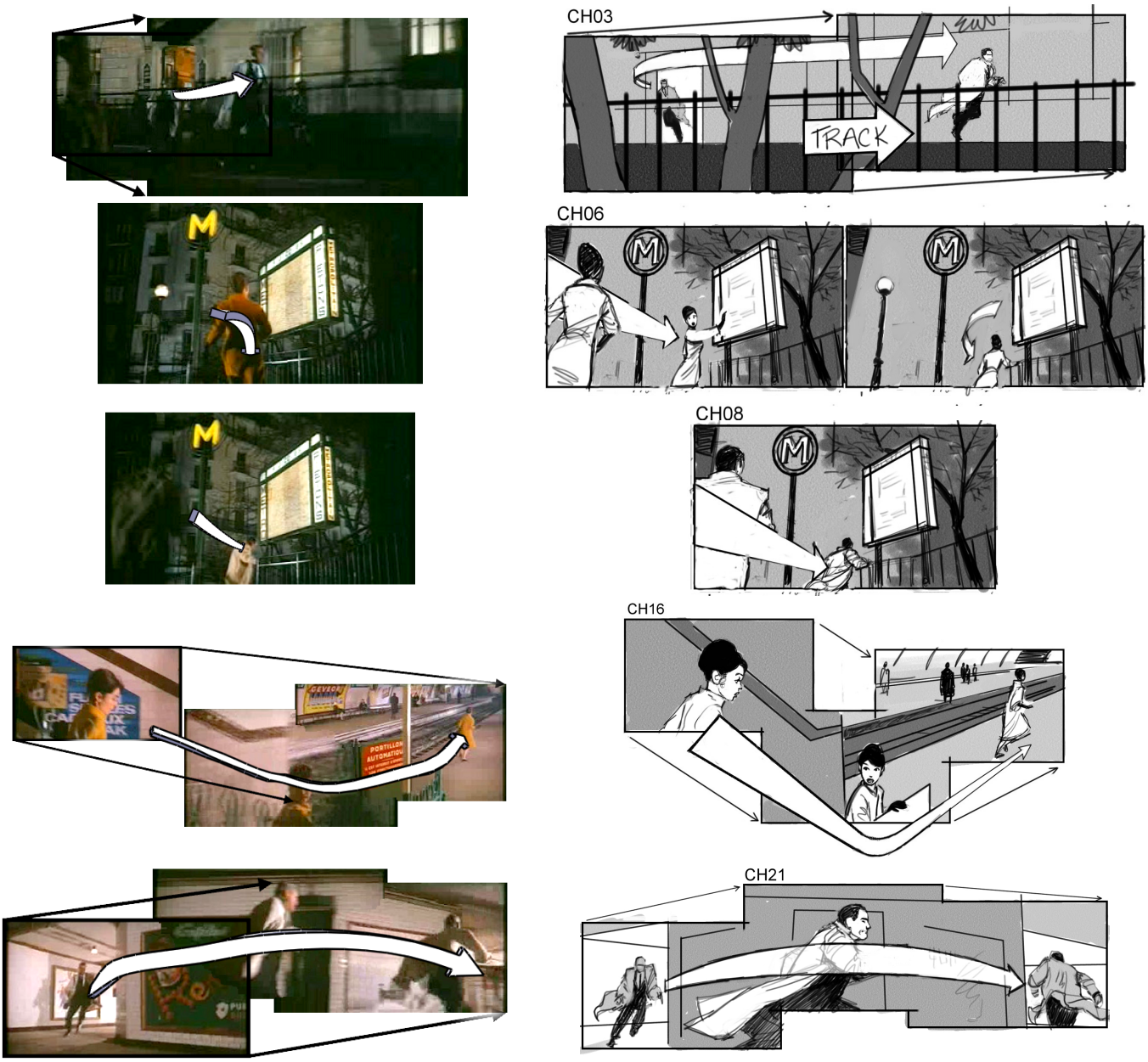

Figure 11 Schematic storyboards for a chase sequence from *Charade*. Left: results produced by our system. Right: results produced by hand. (Credit: Peter Rubin)

ryboard is only viewed once or twice. For example, storyboards are conceivably useful in the initial "assembly" stage of editing documentary footage, in which large volumes of data must be screened and culled to select shots that will be used in the first cut [Sangster 2005]. In other applications, such as surveillance, a storyboard could act as a high-level preview of video material for pre-screening of unusual activity [Irani and Anandan 1998]. However, both scenarios require annotation of a large volume of potentially low-value video material. Further automation of our system may bring these applications within reach.

We summarize the limitations of our system in three broad categories: fundamental limitations of the concept, limits of the algorithms used, and limitations of the present implementation.

Since it is based predominantly on existing storyboard iconography, our system shares some of the fundamental limitations of storyboards themselves. For example, storyboards do not use a standard indication for the duration of a shot or the speed of a motion, other than generic text labels such as "fast" or "slow", or the duration in seconds or frames. Furthermore, long and complex shots may be split into many segments, with confusing annotations and a spatial layout that may waste screen space.

The algorithms we have chosen for image layout can occasionally result in objects being duplicated, partially or completely omitted, or interrupted by visible seams. Although we believe the results are satisfactory for visualization purposes, additional manual or automated methods could be applied to eliminate remaining artifacts. For example, the segmentation process can be improved by taking advantage of motion metrics, as demonstrated by Kumar *et al.* [2005]. In any event, the image quality of our storyboards is always limited by that of the input video. One avenue for improvement is to utilize other image sources — such as still photos or hand sketches — to replace the degraded video frames.

Finally, our prototype implementation is lacking some obvious niceties, such as support for multiple subjects, depth estimation that

is more robust to changing orientation and occlusions, smoothing of complex arrow paths, and more intelligent layout avoiding overlapping annotations. Also, as with conventional sliders, there may be fewer pixels in the storyboard than there are frames in the video, so some frames of the source video are not directly accessible through the scrubbing interface. Although our present system does not address these issues yet, we believe most of them can be addressed using relatively straightforward techniques.

Schematic storyboards offer many other avenues for further research. Our principal aim is to improve the quality of our results and bring them closer to the appearance of hand-drawn storyboards. One possible improvement is to perform a global layout optimization, and selectively employ semi-transparent arrows to improve visibility.

One problem inherent in multi-perspective mosaics such as those shown here is that no placement of seams can completely hide changes of perspective or parallax. One possible solution is to encourage seams along architectural [Zelnik-Manor et al. 2005] or subject/background boundaries. Another option is to apply small amounts of distortion to the images in order to improve their alignment [Jia and Tang 2005]. And although our system provides a non-photorealistic filter, we believe these results can be improved by taking advantage of motion cues to render differently moving regions in different styles.

Schematic storyboards can in principle be augmented as adaptive entities that change their appearance depending on context or size. For example, when surveying a large number of storyboards for assembly or editing purposes, the storyboards may be more visible using a more abstract rendering emphasizing one or two frames with simplified color schemes and 2D motion arrows. Dynamically-generated storyboards could also change their detail level and layout depending on a user's level of interest in a segment or region of time. Schematic storyboard style may also prove especially useful for depicting motion data without a unique viewpoint, such as motion capture data. In this setting, the choice of viewpoint is an important free variable.

Although some of the tasks performed by the user in our system may be automatable for certain types of input, many real-world videos contain rapidly moving objects with motion blur, occlusions, and severe compression artifacts that can confound tracking algorithms. Nonetheless, we believe computer vision approaches are on the threshhold of tackling these issues robustly. Fully automating our system could enable its use at the consumer level, for applications ranging from internet video search to home video editing. [**TODO**: combine with earlier discussion of operating range?] We hope to identify appropriate algorithms for this purpose.

Acknowledgments

The authors would like to thank filmmaker Ann Coppel and editors Cindy Sangster and Nic Anastassiou for helpful discussions about film editing practice, and Nodira Khoussainova for her acting. Very special thanks to Peter Rubin for providing artwork and for improving our understanding of storyboard iconography. *Charade* was obtained from the Internet Movie Archive, `http://www.archive.org`. This work was supported by Adobe, Microsoft, Pixar, the Washington Research Foundation, the University of Washington Animation Research Labs, and National Science Foundation grant IIS-0413198.

References

ADJEROH, D., LEE, M., AND ORJI, C. 1997. Techniques for fast partitioning of compressed and uncompressed video. *Multimedia Tools and Applications 4*, 2 (March), 225–243.

AGARWALA, A., DONTCHEVA, M., AGRAWALA, M., DRUCKER, S., COLBURN, A., CURLESS, B., SALESIN, D., AND COHEN, M. 2004. Interactive digital photomontage. *ACM Transactions on Graphics (Proc. SIGGRAPH) 23*, 4, 294–301.

ASSA, J., CASPI, Y., AND COHEN-OR, D. 2005. Action synopsis: Pose selection and illustration. *ACM Transactions on Graphics (Proc. SIGGRAPH) 24*, 3, 667–676.

BEGLEITER, M. 2001. *From word to image: storyboarding and the filmmaking process*. Michael Wiese Productions.

BLOCK, B. A. 2001. *The Visual Story: Seeing the Structure of Film, TV, and New Media*. Focal Press.

CHEONG, L., AND HUO, H. 2001. Shot change detection using scene-based constraint. *Multimedia Tools and Applications 14*, 2 (June), 175–186.

CUTTING, J. E. 2002. Representing motion in a static image: constraints and parallels in art, science, and popular culture. *Perception 31*, 1165–1193.

FREEMAN, W. T., AND ZHANG, H. 2003. Shape-time photography. In *Proc. Computer Vision and Pattern Recognition*, 151–157.

HART, J. 1999. *The art of the storyboard: storyboarding for film, TV and animation*. Focal Press.

HENG, W., AND NGAN, K. 2001. An object-based shot boundary detection using edge tracing and tracking. *Journal of Visual Communication and Image Representation 12*, 3 (September), 217–239.

HORN, B., HILDEN, H., AND NEGAHDARIPOUR, S. 1988. Closed-form solution of absolute orientation using orthonormal matrices. *Journal of the Optical Society of America A 5*, 7, 1127–1135.

IRANI, M., AND ANANDAN, P. 1998. Video indexing based on mosaic representations. *IEEE Transactions on Pattern Analysis and Machine Intelligence 86*, 5 (May), 905–921.

JIA, J., AND TANG, C.-K. 2005. Eliminating structure and intensity misalignment in image stitching. In *Proc. International Conference on Computer Vision*.

JOJIC, N., BASU, S., PETROVIC, N., FREY, B., AND HUANG, T., 2003. Joint design of data analysis algorithms and user interface for video applications. presented at NIPS 2003 workshop on Machine Learning in User Interface, extended abstract at: `http://research.microsoft.com/workshops/MLUI03/jojic.html`.

JOSHI, A., AND RHEINGANS, P. 2005. Illustration-inspired techniques for visualizing time-varying data. In *IEEE Visualization*, 86.

KATZ, S. D. 1991. *Film directing shot by shot: visualizing from concept to screen*. Michael Wiese Productions.

KAWAGISHI, Y., HATSUYAMA, K., AND KONDO, K. 2003. Cartoon blur: nonphotorealistic motion blur. In *Proc. Computer Graphics International*, 276–281.

KIM, B., AND ESSA, I. 2005. Video-based nonphotorealistic and expressive illustration of motion. In *Proc. Computer Graphics International*, 32–35.

KUMAR, M. P., TORR, P. H. S., AND ZISSERMAN, A. 2005. Learning layered motion segmentations of video. In *Proc. International Conference on Computer Vision*, 33–40.

LEE, M., YANG, Y., AND LEE, S. 2001. Automatic video parsing using shot boundary detection and camera operation analysis. *Pattern Recognition 34*, 3 (March), 711–719.

LI, Y., ZHANG, T., AND TRETTER, D. 2001. An overview of video abstraction techniques. Tech. Rep. HPL-2001-191, HP Laboratories.

MASSEY, M., AND BENDER, W. 1996. Salient stills: process and practice. *IBM Systems Journal 35*, 3,4, 557–574.

MASUCH, M., SCHLECHTWEG, S., AND SCHULTZ, R. 1999. Speedlines: depicting motion in motionless pictures. In *ACM SIGGRAPH 99 Conference abstracts and applications*, 277.

MCCLOUD, S. 1993. *Understanding Comics: The Invisible Art*. HarperCollins.

NICOLAS, H., MANAURY, A., BENOIS-PINEAU, J., DUPUY, W., AND BARBA, D. 2004. Grouping video shots into scenes based on 1d mosaic descriptors. In *Proc. International Conference on Image Processing*, I: 637–640.

NIENHAUS, M., AND DÖLLNER, J. 2003. Dynamic glyphs – depicting dynamics in images of 3D scenes. In *Third International Symposium on Smart Graphics*, 102–111.

PELEG, S., AND HERMAN, J. 1997. Panoramic mosaics by manifold projection. In *Proc. Computer Vision and Pattern Recognition*, 338.

RADEMACHER, P., AND BISHOP, G. 1998. Multiple-center-of-projection images. In *Proc. SIGGRAPH*, 199–206.

ROTHER, C., KOLMOGOROV, V., AND BLAKE, A. 2004. "GrabCut" – interactive foreground extraction using iterated graph cuts. *ACM Transactions on Graphics (Proc. SIGGRAPH) 23*, 3, 309–314.

RUBIN, M. 2005. *Droidmaker: George Lucas and the digital revolution*. Triad Publishing. 327,338.

SANGSTER, C., 2005. Personal Communication.

SIMON, M. 2000. *Storyboards: Motion in Art*. Focal Press.

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

TEODOSIO, L., AND BENDER, W. 1993. Salient video stills: Content and context preserved. In *Proc. ACM International Conference on Multimedia*, 39–46.

TEODOSIO, L., AND BENDER, W. 2005. Salient stills. *ACM Transactions on Multimedia Computing, Communications, and Applications 1*, 1 (February), 16–36.

UMEYAMA, S. 1991. Least-squares estimation of transformation parameters between two point patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence 13*, 4, 376–380.

VLACHOS, T. 2000. Cut detection in video sequences using phase correlation. *IEEE Signal Processing Letters 7*, 7 (July), 173–175.

WARD, J. 1979. *Perception and Pictorial Representation*, vol. 1. Praeger, New York, ch. 13, "A piece of the action: Moving figures in still pictures", 246–271.

WEXLER, Y., AND SIMAKOV, D. 2005. Space-time scene manifolds. In *Proc. International Conference on Computer Vision*, 858–863.

WOOD, D. N., FINKELSTEIN, A., HUGHES, J. F., THAYER, C. E., AND SALESIN, D. H. 1997. Multiperspective panoramas for cel animation. In *Proc. SIGGRAPH*, 243–250.

ZELNIK-MANOR, L., PETERS, G., AND PERONA, P. 2005. Squaring the circle in panoramas. In *Proc. International Conference on Computer Vision*, 1292–1299.