Re-Cinematography: Improving the Camerawork of Casual Video

MICHAEL L. GLEICHER and FENG LIU
University of Wisconsin-Madison

This article presents an approach to postprocessing casually captured videos to improve apparent camera movement. *Re-cinematography* transforms each frame of a video such that the video better follows cinematic conventions. The approach breaks a video into shorter segments. Segments of the source video where there is no intentional camera movement are made to appear as if the camera is completely static. For segments with camera motions, camera paths are keyframed automatically and interpolated with matrix logarithms to give velocity-profiled movements that appear intentional and directed. Closeups are inserted to provide compositional variety in otherwise uniform segments. The approach automatically balances the tradeoff between motion smoothness and distortion to the original imagery. Results from our prototype show improvements to poor quality home videos.

Categories and Subject Descriptors: H.5.1 [**Information Presentation**]: Multimedia Information Systems—*Video*

General Terms: Design, Experimentation, Human Factors

Additional Key Words and Phrases: Image stabilization, casual video, cinematography

ACM Reference Format:

Gleicher, M. L. and Liu, F. 2008. Re-cinematography: Improving the camerawork of casual video. ACM Trans. Multimedia Comput. Commun. Appl. 5, 1, Article 2 (October 2008), 28 pages. DOI = 10.1145/1404880.1404882 http://doi.acm.org/10.1145/1404880.1404882

1. INTRODUCTION

The increasing availability of video capture devices means that one is always on hand to capture an interesting moment. It also leads to a growing amount of video that is created without the planning and artistry required to make *good* video: when a baby takes his first steps, a proud parent is lucky to remember to take off the lens cap, never mind set up a fluid-head tripod or remember what they learned in film school (had they gone). Such casually captured videos often record precious memories but are also difficult to watch. Our goal is to create tools that can postprocess these videos to improve them. In this paper we introduce *Re-Cinematography,* an approach that processes recorded video clips to improve one aspect of their quality: the apparent camera movements.

Casual video[1] is often made with unplanned and poorly executed camera motions. The best-known problem is shakiness from a hand-held camera. These undesirable high-frequency movements can be

[1]We prefer the term *casual video* to the more common term *home video* as it is more correctly descriptive. Some home video enthusiasts take great care and planning their work, and many videos in settings outside the home are taken casually.

This research was sponsored in part by NSF grant IIS-0416284.

Authors' address: Department of Computer Sciences, University of Wisconsin, 1210 W. Dayton St., Madison, WI 53706.

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or direct commercial advantage and that copies show this notice on the first page or initial screen of a display along with the full citation. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, to republish, to post on servers, to redistribute to lists, or to use any component of this work in other works requires prior specific permission and/or a fee. Permissions may be requested from Publications Dept., ACM, Inc., 2 Penn Plaza, Suite 701, New York, NY 10121-0701 USA, fax +1 (212) 869-0481, or permissions@acm.org.
© 2008 ACM 1551-6857/2008/10-ART2 $5.00 DOI 10.1145/1404880.1404882 http://doi.acm.org/10.1145/1404880.1404882

removed by *image stabilization*. Image stabilization adjusts each frame to change the apparent movement between frames. These transformations typically damage the individual frames through distortions, lost portions of the frame, and loss of detail. Removing unwanted shakiness usually makes this an acceptable tradeoff. Almost all video cameras implement some image stabilization, and widely available software solutions can provide even more aggressive stabilization (i.e., larger changes).

Re-cinematography extends image stabilization, adjusting video frames so that the resulting apparent camera motion follows cinematic conventions. Each movement in the original video is replaced by one that better implements its intent. Portions of the video where there is no intentional movement are made to appear as if the camera is completely static; larger, more intentional movements in the original are replaced with movements that appear as if the camera was on a properly damped tripod. Closeups are inserted to provide compositional variety in otherwise uniform segments. Each of these changes is implemented by determining and applying appropriate transformations to the video frames. Like stabilization, transforming the frames involves tradeoffs between image and motion quality. Re-Cinematography considers these tradeoffs explicitly.

In an earlier paper [Gleicher and Liu 2007], we introduced the re-cinematography approach, presenting a system that adjusted the frames of the video such that the apparent camera motions better followed cinematic conventions. In this paper, we provide an improved approach. We begin by more carefully considering the goals of good casual video, more explicitly including the videographer's intent and compositional variety. This new analysis leads to a more complex set of goals for the desired results of re-cinematography. We provide an extended set of methods that are capable of achieving these new goals. Also, our experience with the original re-cinematography system has lead us to improve many of the methods used in it.

1.1 Overview

Consider a video clip of a baby crawling to a bookcase and "pulling up" on it (Figure 1). This casual video was taken spontaneously, using a handheld camera and no planning (it was hard to predict the baby's movement). While in-camera image stabilization has prevented most small jitter, the camera movement wanders distractingly. Also, the composition offers no variety, it is a 60-second "medium shot."

Re-cinematography transforms this clip into something more resembling "good video." In addition to removing any remaining jitter, it replaces the wandering camera movements with ones that appear intentional. For durations of video where the camera is simply wandering (most likely due to the unsteady hand of the videographer), the resulting video appears as if the camera is completely static. For durations of video where the camera makes significant movements, the movements are replaced by fluid, direct goal-directed movements like one would make intentionally with the camera mounted on a fluid-head tripod. To add compositional variety, the system applies synthetic closeups in long, static segments.

The re-cinematography process transforms clips of video. First, a preprocess determines the camera transformations and identifies the most salient object in each frame. The re-cinematography system then analyzes this data to segment the clip into durations where consistent processing can be applied to improve the motion of each. The system then determines a new camera motion for each segment, in a manner that maintains the continuity. Finally, another process examines the resulting motion and applies improvements.

To understand how motions are computed for segments, consider the video clip of a skier (Figure 2) taken with a hand-held digital camera panning to follow the skier. The segmenter correctly identifies this as a single long movement. However, unlike a video made by a professional (with considerable camera experience and a fluid-head tripod), this casual video has small jitters (from being handheld)

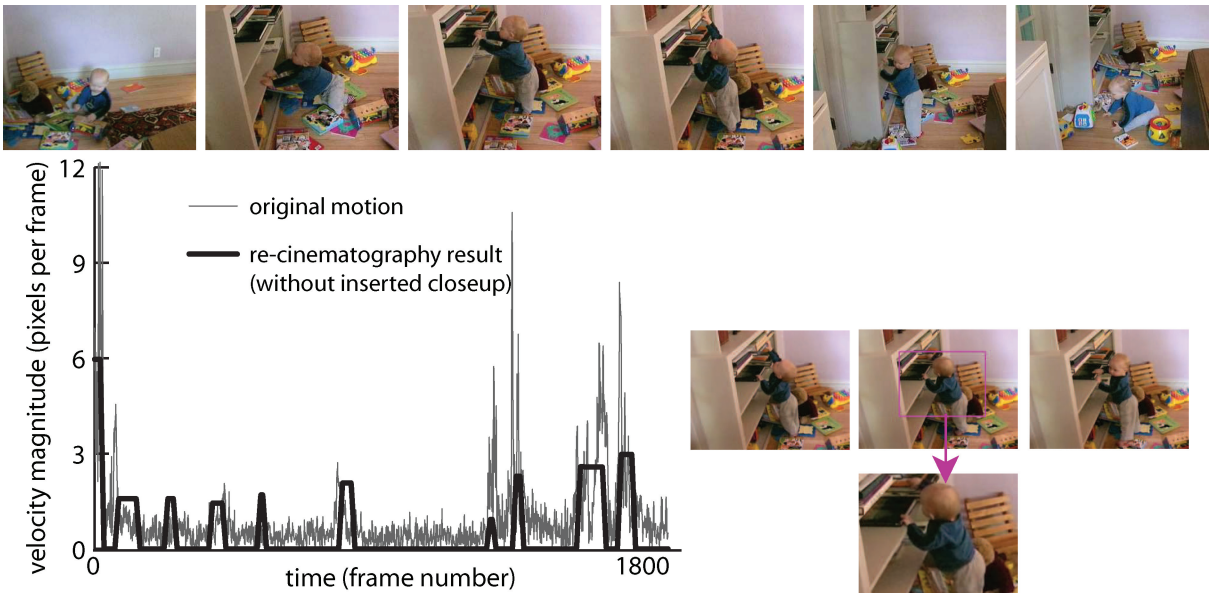


Fig. 1. (top) Selected frames from a one minute clip of a toddler crawling and "pulling up" on a bookcase. (lower left) Velocity profile graph of the camera movement. Even with the camera's built-in stabilization, the original camerawork (thin line) is wandering and undirected. The velocity profile of the Re-cinematography result (thick line) shows how Re-Cinematography segments the clips into a series of static segments where camera motion is removed, connected by transitions that move with proper velocity profiles (ease-in and -out). (lower right) To break the monotony of a 12 second segment with little camera movement, Re-Cinematography creates a synthetic closeup complete with proper transitions.

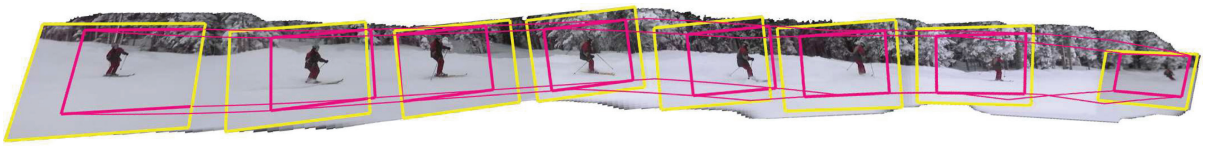


Fig. 2. Schematic of the Re-cinematography process. Conceptually, an image mosaic is constructed for the video clip and a virtual camera viewing this mosaic is keyframed. Yellow denotes the source camera path, magenta (dark) the keyframed virtual camera.

and does not have a fluid, goal-directed motion. Image stabilization (including what is built into the camera) address the former problem; Re-cinematography addresses both.

Re-cinematography can automatically turn this video clip into a pan, similar to what one would obtain using a good tripod. Conceptually, the method uses interframe motion estimation to build an image mosaic that completely stabilizes the camera movement to create a panoramic image, and then animates a virtual camera that views this mosaic. In practice, an implementation need not form the panoramic image (except to create diagrams like Figure 2).

Re-cinematography avoids the limitations of image mosaicing by breaking video into a series of shorter segments, and choosing appropriate virtual camera motions for each. These motions are created automatically for each video segment based on the observed camera motions: nearly static source video results in no motion; larger camera movements are replaced with goal-directed, velocity-profiled paths. The motions are created by keyframing the camera paths. Problems in direct paths are addressed by inserting keyframes. For example, our approach uses *motion salience* to determine what is likely to

be important to the viewer and insures that these objects stay on screen. Other constraints include limitations on the scene analysis and amount of distortion introduced by the transformations.

Re-cinematography is a novel approach to improving the apparent camera motion in casual videos. Specific contributions include:

—An analysis of the goals of casual video quality and the translation of these artistic ideas into mathematical description suitable for implementation.

—A local mosaic approach that allows mosaic-based techniques to be applied to a wider range of video content.

—A model for creating camera motions directly as 2D transformations that maps cinematographic ideals onto implementable mathematical foundations such as matrix exponential interpolation.

—An approach that explicitly considers the tradeoffs in video motion alteration with methods that plan new virtual camera motions that optimize the tradeoff between motion quality and image quality.

A re-cinematography system first preprocesses input video to perform motion estimation and important object identification. Given these three inputs, re-cinematography proceeds in the following steps. First, the video is broken into short segments, each corresponding to a local mosaic, and initial virtual camera motions are created for these segments that insure continuity between them (Section 5). Second, each segments' camera motions are optimized independently to best trade off image and motion quality (Section 6). Finally, the camera motions are used to transform each frame, and in-painting is applied to fill uncovered regions.

Throughout the presentation, tunable parameters are denoted by $p_?$, with the standard values given. Experimenting with parameter settings is easy as the planning process is very fast (a few seconds for a two-minute video). In practice, we have found that little empirical tuning has been necessary.

Because the source video footage tends to be of low quality, the information that can be extracted automatically is limited. Issues such as poor lighting, bad sensors and lenses, severe compression, camera shake make casual video difficult for computer vision algorithms. Re-cinematography relies on limited information about the source video: in addition to the motion estimation used by stabilization, it also uses an assessment of the motion and motion salience as a predictor of what may be important in each frame.

2. RELATED WORK

The use of 2D image transformations to change apparent camera motion has a long tradition. Lab shots that moved the film during optical printing [Bordwell and Thompson 1997] evolved into the complex effects used in animation where flying is simulated by moving the animation camera over a panoramic background [Johnston and Thomas 1981]. Re-cinematography is inspired by Wood et al. [1997], which introduced the camera-over-panorama idea in computer graphics. Unlike their work, we use real video rather than 3D models, a full projective transformation, and do not build or distort panoramas.

Image or video stabilization, of which re-cinematography is a generalization, uses image movement to change apparent camera motions. The methods are ubiquitous and implemented in almost all consumer video cameras, and the literature on the subject is too vast to survey here. Image stabilization comprises three main components: motion estimation, motion smoothing and image production. Our innovation is in the second step. Prior work on motion smoothing falls into three categories: causal (or online) low-pass filters [Litvin et al. 2003], general low-pass filters [Matsushita et al. 2006], and mosaic-based approaches. [Yan and Kankanhalli 2002] is similar to our work in that they combine low-pass filtering in the space of the mosaic, and even use splines to implement these filters. However, our work goes beyond low-pass filtering to create motion paths that follow cinematographic conventions and adapt

to the interpreted intention of the source video. We also more fully consider the ramifications of the mosaic-based approach.

Mosaic-based video stabilization is impractical for cleanup of general casual videos because of their significant nonprojective camera motions [Matsushita et al. 2006]. Our approach reaps its benefits by applying it when it can stabilize a shot, and extending it to create good camera movements. Other applications of mosaicing have inspired us. For example, Irani et al. [1995] provide numerous applications. They introduce "dynamic mosaics" that include some of the original camera motion. Proscenium [Bennett and McMillan 2003] uses mosaic-based stabilization to provide a video editing tool. Salient Stills [Teodosio and Bender 2005] create single frame summaries of video clips. Irani and Anandan [1998] use mosaics to summarize videos for indexing. Dony et al. [2005] use mosaics to create storyboards. We adopt their approach of breaking video into shorter segments to avoid the limits of mosaicing.

Prior work uses knowledge of cinematography to plan camera movements. These works inspired us by demonstrating different domains and applications. Camera motion in 3D environments has been considered by many; see Christie et al. [2005] for a survey. Virtual videography [Heck et al. 2007] determines virtual camera movements given a static camera source video in the specific case of a chalkboard lecture. We attempt to apply similar reasoning to a very different class of source material. Video retargeting [Liu and Gleicher 2006] also considers choosing new cinematography. However it considers transforming feature films, not casual video, and therefore must assume that the source camera motion is the intentional work of a skilled filmmaker and should be preserved.

Another set of prior work performs camera motion understanding on casual videos. Kender and Yeo [2000] and Pan and Ngo [2004] use it for content understanding, while Osian and Van Gool [2004] use it for shot detection. However, only Yan and Kankanhalli [2002] have used it as we do, for informing motion alteration. Their motion analysis is limited to detect video segments with significant shaking, while our more extensive analysis identifies different types of motions and quality of motion estimates to better inform the choice of new camera motions.

Re-cinematography does not consider editing (Section 3): the resulting video clips are the same length as the source. Several prior systems have considered automating the editing of home video. Notable examples include IMCE [Adams et al. 2005], Hitchcock [Girgensohn et al. 2000], Silver [Casares et al. 2002], and AVE [Hua et al. 2004]. Video editing involves selecting and assembling clips of video, whereas recinematography aims to improve the quality of the clips. The two approaches could complement one another nicely. For example, Hitchcock uses motion estimation to find bad camera motions that it discards, instead it could use re-cinematography to repair them.

Recent work, such as Mei et al. [2005] and Achanta et al. [2006], attempts to determine the intent of a videographer in capturing a shot to better inform postproduction automation. Such information could be use to better inform re-cinematography. At present, however, re-cinematography only attempts to deduce the intent of the camera motion.

3. GOOD VIDEO

To go beyond simply removing artifacts (like shakiness) from video, we must first develop an understanding of the goals of good video so they can be translated into an implementable approach. We emphasize that filmmaking is an art: that rules and conventions developed over the past century are only suggestions. Skilled filmmakers may bend these rules (intentionally) to achieve a desired effect, but this is different than a causal videographer ignoring the rules, or simply failing to implement them.

Re-cinematography is concerned with the camera motion within a single clip of video taken as a single run of a camera. The input is therefore continuous in time and viewpoint. It is, in film terminology,

a single shot.[2] While filmmakers often introduce discontinuities, known as cuts, such editing must be done with care such that the viewer can understand the sudden change in time and space. Ironically, discontinuities that are too small are jarring to viewers and therefore known as "jump cuts." Because we begin with a single continuous stream of video, it is difficult to introduce a discontinuity in the camera motion that is large enough to avoid a jump cut. Similarly, it is difficult to introduce temporal discontinuities without enough understanding of the content of the video to know what portions might be discarded. For these reasons, re-cinematography performs no editing (in the filmmaking sense): it processes individual clips (shots), preserving the continuity of camera motion and the duration of the clip.

3.1 Cinematography

Cinematography is the art of choosing what the viewer in a film or video sees through camera positioning and movement. Good video uses the control of viewpoint effectively. Guidelines and conventions developed over the past century suggest what camera work will be effective.

Literature on film helps indirectly. General books on the film arts (such as Bordwell and Thompson [1997]), or more specifically about cinematography (such as Katz [1991] or Arijon [1991]), discuss cinematography and the aesthetic considerations of camera movement. Texts such as Block [2001] and Brown [2002] give more technical discussion and a relationship to low-level perceptual issues. These latter discussions are more readily translated into computational frameworks.

To borrow terminology from Brown [2002], camera movements should be motivated and intentional. If the filmmaker exerts control over the viewer's viewpoint they should have a reason so the viewer can more easily follow. This suggests that small adjustments should be avoided, that is, that the camera is "locked down" unless it makes a significant movement. When there are movements, they should be goal-directed (since they have an intent), rather than wandering.

To create these carefully planned camera motions, filmmakers often use camera supports. A good support, like a fluid-head tripod, makes nonmoving shots stable and that moving ones have continuity in velocity and direction. See Brown [2002] for an array of supports used by filmmakers to create camera movements.

3.2 Cinematography in Casual Video

Casual video is quite different than traditional filmmaking for a number of reasons [Chalfen 1987; Kirk et al. 2007]. Casual videographers usually lack the skill, the equipment (i.e., camera supports), or the planning that "good" filmmaking does. Often casual videographers want to fully experience the event they are capturing, which precludes the amount of attention they can devote to proper cinematography.

Therefore, casual video generally contains "excessive" amounts of camera movement (relative to traditional film) within its shots [Chalfen 1987]. "How to do it" (HTDI) books on producing good video always advise videographers to use tripods and to use camera motion sparingly [Ang 2005]; [Brandon 2005], but these suggestions are rarely followed [Chalfen 1987]. Some of this stems from the fact that casual video often contains long shots, connecting many viewpoints, whereas more traditional filmmaking might use a sequence of shots instead. However, many of the camera movements are unintentional. Those that are intentional are often poorly executed.

These observations inform re-cinematography. First, since casual video shots are long and may contain many different types of camera movements, a "one-size-fits-all" approach (like standard image stabilization) does not necessarily apply. Instead, we must consider different durations of the clip (that

[2]Most modern camcorders clearly denote individual shots, however for legacy footage there is a rich literature on automatically determining shot breaks.

we call *segments*) that have very different movements, and therefore different appropriate transformations. Re-cinematography therefore breaks shots into shorter segments based on their movement.

Generally, re-cinematography prefers to avoid camera movements. When the method identifies segments where a static camera is appropriate, it removes all movement. This includes not only segments where the camera is nearly static, but also segments where the camera wanders around without any meaningful direction. Creating such static segments is given priority, with the exception of zooms.

Zooms are of particular importance in casual video. For one, they are quite prevalent: "There was a frequent use of the zoom technique. The majority of newer cameras have a zoom lens built into the body, and home movie makers seem to feel they should use it." ([Chalfen 1987], p. 65). Secondly, they are most certainly intentional, users rarely press the zoom button by accident. Third, zooms are often used to provide a close-up of an important object (in traditional film, a closeup shot would more often be used), and are therefore semantically significant. In fact, identifying zoom-and-hold patterns has been used for automated video interpretation [Kender and Yeo 2000; Pan and Ngo 2004]. Finally, zooms provide a mechanism for creating compositional variety without introducing discontinuities.[3]

For segments where the camera does move (including zooms), re-cinematography attempts to make the movements look intentional: they should follow directed paths and move with velocity-profiled paths between goals. By velocity-profiled, we mean that it moves with a constant velocity, except possibly for an initial acceleration or deceleration (Section 4.4). Other details of cinematography, such as size constancy and leading, are also considered in our algorithm design.

Without an understanding of the motivation for the original camera motion, re-cinematography relies on the source movement as a basis for a new one, effectively predicting what the movement might have been if the videographer had a properly damped camera support. This reliance on the source motion is convenient as practical considerations limit how far re-cinematography can deviate from it.

4. TECHNICAL FOUNDATIONS

4.1 Motion Estimation

This section reviews some of the basic mathematics of image and video stabilization in order to clarify our notation.

Consider two images a and b from a video. The stabilization transformation $\mathbf{S_b}(a) \in \mathcal{R}^2 \rightarrow \mathcal{R}^2$ maps the positions of points in frame a to where they appear in frame b, if those points were in the same position in the world. If we apply $\mathbf{S_b}(a)$ to the image of frame a, the result would be the view of the world of a taken from the viewpoint of b. Were we to apply the corresponding transform to each frame of the video, the result would be a video taken from the single viewpoint of b. There would be no camera motion, the images have been stabilized.

An image created by combining a collection of images all transformed to a common coordinate system is called a *mosaic*. We refer to the common image as the *base frame*.

For arbitrary scenes and arbitrary camera movements, the stabilization transformation can be complex. However, in two important cases, the transformation will be a 2D projection: if the world is a single plane, or the camera rotates around its optical center [Szeliski 1996]. A 2D projection

$$x' = \frac{s_1 x + s_2 y + s_3}{s_7 x + s_8 y + 1}, \quad y' = \frac{s_4 x + s_5 y + s_6}{s_7 x + s_8 y + 1}$$

is an 8 parameter transformation that can be expressed as a linear transformation in homogeneous coordinates, that is, a 3×3 matrix. This 2D projective transformation is also known as a *homography*.

[3]The use of a zoom transition is less desirable and more "amateurish" than a *proper* cut (c.f. [Ang 2005], p. 74–75). However, making such continuity cuts is more difficult, especially without well-planned shots to cut between.

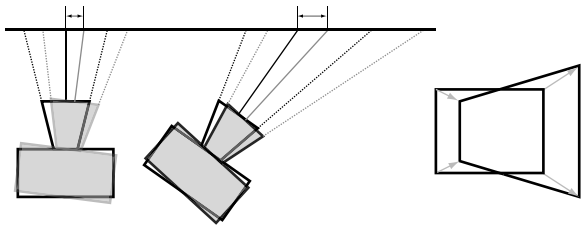


Fig. 3. Identical camera rotations can lead to very different motions on the base plane. The movements are identical in local coordinates. To measure the local transformation, we measure the distance moved by each corner of the identity image. This would be the same for both cameras to the left.

In the event that the restrictions to camera rotation or a flat world do not apply, a 2D projection is only an approximation to the real stabilization transformation. This simplified model is preferred because it is mathematically convenient (matrix operations allow for manipulation), has proven to be a reasonable approximation to many real situations, and a plethora of techniques exist for motion estimation (that is, estimating $\mathbf{S_b}(a)$ from the images it relates). Historically, all image stabilization methods (with the notable exception of Buehler et al. [2001]) use either homographies or a more restricted class of transformations that are a subset of them.

Motion estimation typically performs best between similar images. Most compute transformations between adjacent video frames, that is, $\mathbf{S}_i(i+1)$, and determine other transformations between other pairs by composition,

$$\mathbf{S}_b(a) = \text{if } (a > b) \prod_{i=b}^{a-1} \mathbf{S}_i(i+1), \quad \text{if } (a < b) \prod_{i=b}^{a+1} \mathbf{S}_{i-1}^{-1}(i). \tag{1}$$

The parameters of the homographies are difficult to interpret directly [Gleicher 1997]. A four-point representation, which measures the displacement of the four corners of the source image, gives more meaningful metrics in units of pixels. Making these measurements in local coordinates (Figure 3), that is, between pairs of frames, avoids issues from the non-linearity and scene dependence of the transformations.

The sum of the corner vectors (the arrows in Figure 3) gives a measurement of apparent camera movement, while the sum of their magnitudes measures the amount of change between frames. This measurement is used in several re-cinematography steps.

4.2 Local Mosaics

Mosaic image stabilization is attractive because it gives the appearance of a locked down camera. Unfortunately, it is impractical for longer videos and complex motions for a number of reasons [Matsushita et al. 2006]. Therefore, we limit our use of mosaics to short durations of video. We call this a *local mosaic* as we consider a mosaic around the locality of a given frame.

One limit of mosaics for longer video segments is error accumulation in Equation (1). Even the best homography estimation will have errors, due to either misregistration or a poor fit to the planar projective model. As the sequence grows longer, small interframe errors are amplified as they are composed in Equation (1). Therefore, for each homography we compute an *unreliability score*. The score is used in two ways: first, homographies with very high scores are considered too unreliable to use. Second, the length of a local mosaic is limited such that the sum of the unreliability is below a threshold.

The unreliability score is computed by "triple-frame checking." Our system estimates both interframe homographies $\mathbf{S}_i(i+1)$ and homographies between spaced frames $\mathbf{S}_i(i+2)$. The checking score confirms that the result of two inter-frame homographies is similar to the result of the wider spacing by computing the difference matrix $\mathbf{S}_{i-1}(i)\mathbf{S}_i(i+1)\mathbf{S}_{i-1}^{-1}(i+1)$ and measuring the maximum displacement it makes to the four corners of the frame. It is desirable to velocity correct the unreliability scores because errors in homographies are less noticeable when they are small relative to the overall motion. However, we have found such a correction ineffective because the velocity estimates depend on the homographies that we are trying to assess. In practice, the segmentation process (Section 5.1) breaks the video into short enough segments such that a single local mosaic applies.

Our approach does not actually form the mosaic image (except to explanatory figures). This avoids issues in merging images. The transformations $\mathbf{S}_i(j)$ give us a common coordinate system, conceptually the coordinate system of the local mosaic, for the duration of the video. When the video is segmented into local mosaics, we denote the base frame for a particular video frame as $b(t)$.

4.3 Apparent Camera Motion

If we consider the mosaic to be the "world," then the viewpoint or camera is a transformation $\mathbf{C}$ that maps from the mosaic to the video (i.e., the camera's film plane). Assuming a standard camera model, $\mathbf{C}$ is a homography [Szeliski 1996].

To view a frame of a video from a particular camera $\mathbf{C}$, the video frame using the stabilization transformation is then transformed to the screen by the camera:

$$\mathbf{M}(t) = \mathbf{C}(t)\mathbf{S}_{b(t)}(t).$$

If $\mathbf{M}(t)$ is the identity matrix ($\mathbf{I}$), the viewer sees exactly the source video frame. If $\mathbf{C}(t) = \mathbf{I}$, then the viewer is looking at the "center" of the mosaic (i.e. directly at where the base frame was). The inverse of the camera transform can be viewed as a quadrilateral on the base frame. Intuitively, this is the part of the world the camera "sees."

A video appears to have camera motion when static objects in the scene move. As the mosaic defines the static objects, the motion of $\mathbf{C}(t)$ creates a camera movement. Note that if we are viewing the original video, $\mathbf{M}(t) = \mathbf{I}$, so that $\mathbf{C}(t) = \mathbf{S}_{\mathbf{b(t)}}{}^{-1}(t)$. In this case, the virtual camera motion is following the motion of the source video.

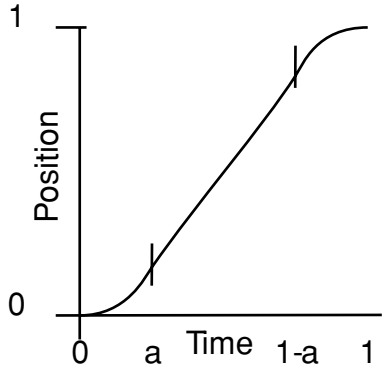
It is important to recognize that the result is $\mathbf{M}(t)$. The intermediate transformations, $\mathbf{C}(t)$ and $\mathbf{S}_{b(t)}(t)$, might be wild distortions, providing their composition produces a reasonable transformation.

The problem of re-cinematography is to choose $\mathbf{C}(t)$. Standard image stabilization chooses it to be a low-pass filtered version of $\mathbf{S}^{-1}(t)$ (although it is rarely implemented this way). Mosaic-based stabilization chooses $\mathbf{C}(t)$ to be constant.

4.4 Camera Motion

Rather than model the 3D geometry of the virtual camera (as in Irani et al. [1994]), we create camera motions directly in the space of the homography matrices $\mathbf{C}(t)$. The desire to control apparent velocity profiles of a 3D camera (Figure 3) suggests a local linearization of the transformations using an exponential map. The idea, introduced to the graphics community by Alexa [2002], takes the logarithm of the transformation matrices before applying linear operations (such as interpolation).

An intuition behind the use of the exponential map considers a camera moving with constant velocity. Between every frame, the transformation in the camera's local coordinate system is the same, for example 1 degree to the right. The interframe transform is applied by matrix multiplication, so if $\mathbf{A}$ is



$$d(u) = \begin{cases} \frac{u^2}{2a-2a^2} & u < a \\ \frac{a}{2-2a} + \frac{u-a}{1-a} & a \le u \le 1-a \\ 1 - \frac{(1-u)^2}{2a-2a^2} & u > 1-a \end{cases}$$

Fig. 4. The ease curve accelerates with a constant acceleration, maintains a constant velocity, then decelerates.

the initial transformation and $\mathbf{T}$ is the transformation between steps, after n steps the transformation would be $\mathbf{AT}^n$. To interpolate between transformations $\mathbf{A}$ and $\mathbf{B}$ from $t = 0$ to 1 with a constant velocity (in terms of camera rotation), we compute $\mathbf{A}(\mathbf{A}^{-1}\mathbf{B})^t$. Taking the logarithm of this expression would allow it to be re-written as:

$$e^{(1-t)\log\mathbf{A}+t\log\mathbf{B}}. \tag{2}$$

That is, constant-velocity interpolation of transformations can be achieved by linear interpolation of their logarithms. However, because $\mathbf{A}$ and $\mathbf{B}$ are matrices and multiplication does not commute, $e^{\mathbf{A}}e^{\mathbf{B}} \neq e^{\mathbf{A}+\mathbf{B}}$ [Govindu 2004], so this is only an approximation. [Kavan et al. 2006] shows that although interpolation of matrix logarithms is not constant velocity, the non-constancy is small, bounded, and correctable if need be. In practice, the effect is too small to be visible in re-cinematography (except in numerical displays of the results such as Figure 8).

Abrupt accelerations and decelerations of the camera are jarring to viewers. An instant acceleration to a constant velocity creates a noticeable artifact. To avoid this, we use an "ease-in/ease-out" function as given by Liu and Gleicher [2006] to warp time so that the interpolations begin with a period of constant acceleration, move with a period of constant velocity, and end with a period of constant deceleration. This function is shown in Figure 4. Alternatively, smoother splines can be used for interpolating the matrix logarithms. We will discuss this in Section 6.6.

5. MOTION SEGMENTATION AND GENERATION

After preprocessing, the first step in the re-cinematography process is to break the source video into segments and to determine an initial camera motion for each segment. Segmentation is designed to achieve two goals: each segment must be short enough that a single local mosaic can be applied, and each segment should contain a single type of camera movement that can be processed. The parameters for segmentation create subjectively different results, such as shorter segments. The segmentation process is fast enough (under a second for several minutes of video) that parameters can be tuned as desired.

5.1 Segment Finding

The segmentation process first identifies three types of segments from the motion: bad segments, zoom segments, and static segments. These segments are identified in that order to insure the priority is maintained (e.g., if something is part of a bad segment, it should not be part of any other segment).

These initial segments are created such that there are sufficient gaps ($p_{mingap} = 10$ frames) between them to make transition segments.

"Bad" segments are parts of the video where the motion estimation is too unreliable to be used. These segments are identified by finding frames whose un-reliability scores (Section 4.2) are above a threshold ($p_{bh} = 40$ pixels). Frames within a short window ($p_{bhw} = \pm 3$ frames) are also considered bad. Bad frames are grouped into bad homography segments. For bad segments, re-cinematography sets $\mathbf{M}(t)$ equal to the identity as we cannot alter the bad frames reliably.

Next, zoom segments are identified. Zooming is defined as image motions where there is significant change in image scale. Because of perspective effects, different parts of the image might be changing in size at differing rates due to camera movements. For each frame, we first compute a "zoom velocity." The corners positions of the image are transformed to their positions in a subsequent frame. The ratio of the perimeters of these two quadrilaterals is used to provide an estimate of the overall scale change. The logarithm of this ratio is used as the velocity estimate as it provides a zero centered parameter with equal magnitudes for equivalent zooms in and out. In practice, the velocity estimate is computed over a 5-frame window to reduce noise.

A zoom segment is defined to be a duration of video where at least $p_{minZoom} = 10$ frames have a zoom velocity either all above or all below a threshold $p_{zoom} = 1.015$. In our implementation, filtering allows a small number of frames to be below the threshold. Our zoom criteria does not explicitly distinguish zooms and camera motions in the view direction (e.g., moving the camera forward is the same as zooming in).[4] In practice, treating forward/backwards motions as zooms has no ill-effect in our system. Either type of motion is significant and should be treated as such. Zoom segments are chosen so there are sufficient gaps between them and previously identified bad segments.

Third, the segmenter identifies static segments. Static segments are those where the resulting camera motion is not significant. A static segment does not imply that the camera has a small velocity during the segment. On the contrary, shaking and wandering have high velocities but are not significant motions. A static segment is therefore defined to be any series of frames where the movement between the first frame and any other is less than a threshold. Movement is measured by the displacement of the image center. The threshold is based on the distance (in pixels) that this point moves ($p_{staticDist} = 15$ pixels), and there is a minimum duration for static segments ($p_{minStatLen} = 30$ frames).

The segmenter prefers to make static segments as long as possible. Of all the possible static segments, it chooses an appropriate set (i.e., subject to the limitations of having proper gaps between them) using a greedy algorithm to maximize the lengths.

At this point, the segmenter has identified bad, zoom, and static segments. These segments are not overlapping, and have appropriate gaps between them. The remaining frames, including the short gaps enforced between segments, are grouped into segments of type "moving." By construction, any two segments of type other than moving always have a moving segment between them. This is important as there must be a transition between each of these segment types to maintain continuity. Our system attempts to avoid moving segments that are too short to prevent motions that are too abrupt. Static segments are shortened if necessary to accommodate this restriction.

For each segment, the base frame is the middle frame. Also, segments that contain too much error are considered too long and broken.

[4]The previous systems for analyzing home video we are aware of, such as Kender and Yeo [2000] and Pan and Ngo [2004], do not make this distinction either.

5.2 Initial Motion Creation

Once the motion has been broken into a series of segments, initial motions for these segments are chosen. For each of the four types of segments (bad, zoom, static, moving), the initial motion is chosen differently.

For bad segments, we choose $\mathbf{M}(t) = \mathbf{I}$, as without reliable homography information the best we can do for the boundary is to mimic the motion of the source video. Note that for bad segments, we do not use the stabilizations so that there are no meaningful camera transformations.

For each static segment, our algorithm computes a single constant value for $\mathbf{C}(t)$ to be used over the segment's duration. The constant camera creates a "locked down" look for these segments where the camera was close to being static. The identity is used as an initial choice for the matrix, $\mathbf{C}(t) = \mathbf{I}$. This has the effect of showing the base frame unaltered. Because the amount of movement in static frames is typically small, this is often a reasonable choice, although it may be adjusted using techniques discussed later (Section 6.4).

The camera motion for moving and zoom segments is created by keyframing. For both segment types, we initially determine the required transformations at the ends of the segment, and keyframe interpolation on the camera matrices is performed using the matrix exponentials as described in Section 4.4. Additional keys to interpolate may be added to refine these trajectories in later steps (Section 6.5).

For zoom segments, we set the initial and final keyframes such that $\mathbf{M}(t) = \mathbf{I}$. This creates a motion that achieves the same ends as the identified zoom segment, but with a much more directed path.

Moving segments connect other segments, therefore their beginning and end key frames must be chosen to provide continuity with the neighboring segments. The camera motions of the neighboring segments must be chosen first, and these are used to determine the keys for the connecting moving segments. We describe how the end of the segment is chosen, the beginning is done similarly.

If we denote the moving segment in question by index i, we denote the the base frame of the moving segment as frame index b_i, b_{i+1} is the base frame of the next segment, t_b is the beginning of the moving segment, and t_e be the end (so $t_e + 1$ is the beginning of the next segment). The end keyframe of the moving segment is therefore $\mathbf{C}_i(t_e)$.

If the moving segment is either followed by a bad segment, or is at the end of the clip (so it is not followed by any segment), we choose $\mathbf{C}_i(t_e)$ such that $\mathbf{M}(t_e) = \mathbf{I}$. Otherwise, we choose the key by extrapolating the movement of the next segment to the appropriate time, and then converting the camera motion from being relative to the next segment to being relative to the moving segment:

$$\mathbf{C}_i(t_e) = \mathbf{C}_{i+1}(t_e)\, \mathbf{S}_{b+1}(t_e)\, \mathbf{S}_b^{-1}(t_e). \tag{3}$$

This expression computes the key by transforming the extrapolated camera matrix into the video coordinate system, then transforming it into the coordinate system of the moving segment. For static segments, the extrapolation is easy as the camera matrix is constant. For zoom segments, the keyframe interpolation process must be used to extrapolate beyond the segment boundary.

6. MINIMIZING LOST IMAGE INFORMATION

The methods of the previous section suggested transformations that when applied to the video images will create good apparent camera motions. However, the proposed transformations did not consider their effects on the resulting images.

When we transform the image by $\mathbf{M}(t)$, we change it in potentially damaging ways. We consider three different types of problems that may be introduced. If a problem is considered unacceptable, some step must be taken. After a discussion of the various problem types, we describe methods that adjust segments to avoid these problems.

6.1 Offscreen Problems

The transformation may move portions of the source image outside of the frame of the result. Losing a portion of the image (necessarily near an edge) this way is generally not a problem unless either a large fraction of the image is lost, or the lost part of the image is important. This latter case is quite critical, and is handled explicitly.

Our re-cinematography system identifies important objects in each image (Appendix A.2). If any portion of an identified region is moved off screen by the transformation, the transformation is considered a problem. The distance in pixels that the farthest corner of the important object bounding box extends beyond the frame is used as a measure to prioritize the importance of fixing the particular frame.

Offscreen problems are handled in three ways. First, the camera transform for static segments is chosen such that there are no offscreen problems (Section 6.4). Second, keys are inserted to alter the motion in dynamic segments to avoid offscreen problems (Section 6.5). Third, a separate pass can be run to fix remaining off-screen problems (Section 6.7).

6.2 Uncoverage Problems

The transformation might map places outside of the source image to inside the frame of the result, leading to empty areas. We call these regions *uncovered* areas. Left empty (usually black), these areas are not only ugly but the edge moves with the high frequencies that have been removed from the camera motion. In-camera stabilization avoids these effects by capturing "extra" pixels (called overscan) as a buffer and restricting the amount of movement to be small enough that this is sufficient.

In software, methods can use pixels from other frames to provide information about what information lies outside the source video frame, allowing uncovered areas to be filled in. The basic methods, such as Hansen and Mcdowel [2001] and Litvin et al. [2003], use the transformations that relate the video frames to copy pixels from neighboring frames to fill uncovered regions. More recent methods, such as Wexler et al. [2007] and Matsushita et al. [2006], use alignment along the edges to avoid artifacts from parallax and in scene motion.

Our implementation provides filling based on other frames. Neighboring frames (a 3-second window in either direction) are stabilized to the current frame and alpha-blended with more distant frames drawn first. Unfortunately, such filling is often ineffective. Because the transformations proposed by re-cinematography can be large, the filling algorithm may need to look quite far into the future or past to find a frame that provides the necessary information. During this duration of time, it is quite possible that the scene has changed enough that the pixels of the alternate frame are no longer appropriate to use for filling. The artifacts often appear as "ghosts" of moving objects (including shadows).

The alternative to filling is to adjust the transformations such that there is no uncovered area. This can be done with minimal impact to the motion by scaling the source image to be larger, effectively creating overscan. The problem with scaling is that it creates additional offscreen areas. Another problem with scaling is that the scale factor cannot be changed rapidly, otherwise unintended apparent zooming will be introduced. A final option for handling uncovered areas is to change the motion to give up some smoothness to reduce the amount of uncovered area.

To quantify the amount of uncoverage, a penalty is associated with a transformation as the percentage of the resulting frame that is uncovered. Methods that automatically adjust segments to reduce uncovered penalties are described in Section 6.4 and Section 6.5. In practice, a combination of these methods to reduce the amount of uncovered areas are used in conjunction with filling, as the latter approach is most effective for small uncovered areas.

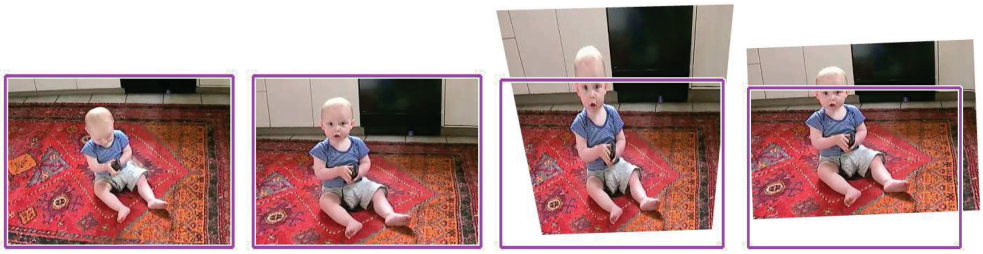


Fig. 5. The projective transformation can stabilize the carpet between two frames (left 2 images), but stretches objects out of this plane (middle right). The rightmost image shows the similarity transform closest to the projective homography.

6.3 Distortion and Resampling Problems

Nonuniform transforms may introduce unpleasant distortions. They may appear as unnatural viewing directions, or will stretch portions of the scene that violate the projective assumptions, as shown in Figure 5. To quantify the distortion of a transform, we compute the distance to the closest similarity transform, where the distance is defined by the corner displacements.

The distortion penalty is considered in the segment optimization methods below to insure that objectionable distortions are not introduced.

The resampling used to implement the transformation may result in a loss of detail, especially if some amount of zoom is used. Given that much casual video is often presented in a small format (e.g., on a video sharing site, a portable media player), these details might be lost anyway. Our implementation does not explicitly consider resampling losses.

6.4 Optimizing Static Segments

With a static shot, the amount of motion in the shot is necessarily small, so the amount of problems generated because of the motion is likely to be small as well. The composition of the shot is set by the single camera matrix used by all of the frames. The standard choice, to set $\mathbf{C}(t) = \mathbf{I}$ (i.e., to show the base frame unaltered, and all other frames relative to this) is often effective. While the other frames in the shot may move to create penalties, they generally do not move far enough to cause bad distortion. Because the base frame covers the entire image, there is guaranteed to be coverage for each pixel during the shot, so infill methods are likely to be effective.

As static shots get longer and/or there is significant motion near the edges of the screen, infilling may not work well. In such cases, we may choose to zoom in on the static shot, choosing a camera transformation that insures that every frame is completely covered. To do this, each frame's boundary is transformed to the base frame and the intersection is found. The transformation is chosen so that the frame is filled from within this intersection. While this guarantees that there will be no uncovered areas over the shot, it is also quite conservative. We constrain the scaling such that identified salient objects are not moved offscreen (Figure 6), however, this may mean uncovered areas remain.

Paradoxically, the place where this zooming is often most needed, when filling fails because there is too much action near the edges, is not well handled by zooming either because the edges are moved offscreen. In our implementation, we have not automated the test for the effectiveness of filling. Therefore, our system cannot make an informed choice as to whether filling or scaling is a more appropriate choice. By default, it chooses to fill the uncovered regions, and gives the user the option of requesting that scaling be done if they are unsatisfied with the filled results.

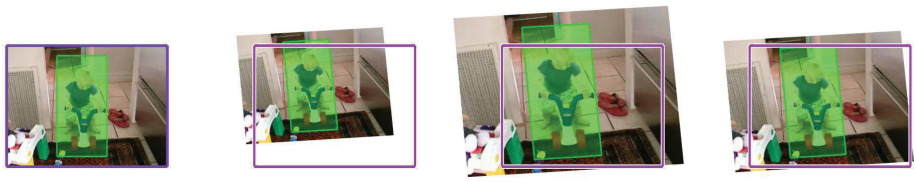


Fig. 6. (a) The initial frame with its salience bounding box shown in green. (b) The proposed camera motion has a large uncovered region (on many of its frames). (c) Scaling to fill the uncovered region removes some of the salient image part. (d) The salience onscreen constraint prevents too large of a scale factor.

6.5 Optimizing Dynamic Segments

For dynamic segments (moving and zoom segments), the camera motion is created by keyframe interpolation. Initially, keys are set at the beginning and end of each segment (Section 5.2). Interpolating between beginning and ending keys creates camera movements with direct, velocity-profiled paths. Such paths are ideal in terms of our goals for motion (Section 3), but may deviate considerably from the actual camera motion. These large deviations lead to transformations that are far from the identity and therefore may create image problems. Therefore, we alter the motion to avoid the problem, trading motion quality for image quality.

The system performs optimization to tradeoff between motion quality (directness and smoothness) and image quality (measured by the penalties for distortion and uncovered areas). The straightforward way to solve this optimization would be to pose it numerically by defining a metric of motion smoothness, and solving the highly nonlinear optimization problem (because the metrics depend on the exponential trajectories). We consider such an approach to be impractical, especially since the metrics for image penalties are inexact. Instead, we pose the problem as a constrained optimization, with metrics chosen to be easy to solve.

To adjust a dynamic segment, we seek the motion that is as smooth as and as close to the originally proposed motion as possible, subject to the constraint that we have no objectionable image problems. Our metric for smoothness is the number of inserted keys. While this measure is quite crude as it doesn't take into account how far an introduced key is from the smooth path, it does lead to simple and efficient solutions to the constrained optimization. Also, in practice it leads to motion paths with desirable properties (Section 6.6).

Inserting a keyframe has a benefit to image quality, but a potential cost to motion quality, as illustrated in Figure 7. For example, inserting a new keyframe at time j $\mathbf{C}(j) = \mathbf{S}_b^{-1}(j)$ makes $\mathbf{M}(j) = \mathbf{I}$ so the penalty on the frame is zero. Frames close to j (between j and the adjacent keyframes) will also have their penalties reduced. On the other hand, inserting the key frame breaks the directness of the camera's path.

Key insertion provides a simple, greedy algorithm for solving the constrained optimization problem. Our algorithm identifies the frame with the worst image loss and inserts a key, repeating until there are no more frames with bad enough losses to be considered a problem. The algorithm is constrained to keep a minimum spacing between keys ($p_{minKeySpace} = 30$ frames). The simple algorithm's parameters are the minimum key spacing ($p_{minKeySpace} = 30$ frames), and the thresholds below which penalties are not considered a "problem." There are separate penalty thresholds for salience offscreen ($p_{so} = 0$, i.e. it is, unacceptable for any important region to be moved offscreen), distortion ($p_d = 20$ pixels of corner movement), and uncovered area ($p_{uc} = 15\%$ of the screen area). Rather than determining scaling factors between the different penalty terms, we choose a strict ordering: we choose salience offscreen

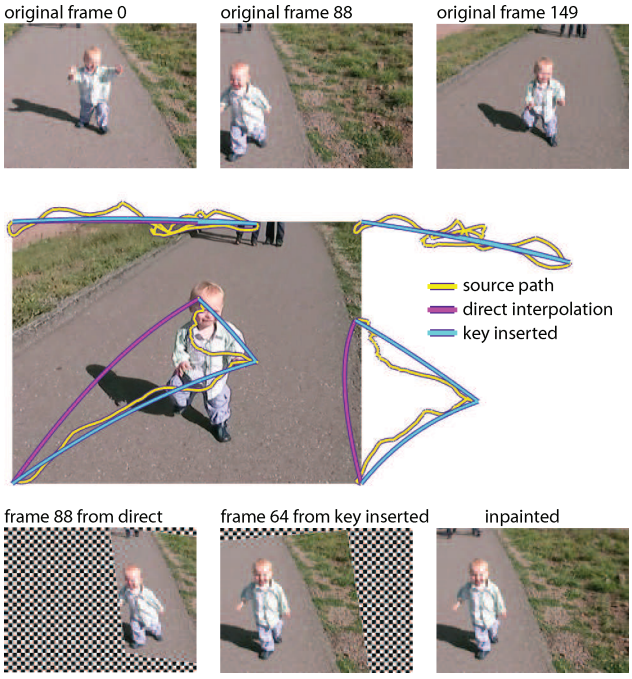


Fig. 7. 5-second segment walking backwards tracking a toddler. Both subject and camera are weaving significantly. Direct interpolation provides a smooth path, but creates an extremely uncovered and distorted frame. Inserting a key at this frame causes it to appear as in the source. The smaller problems that remain can be addressed with inpainting.

problems first, distortion penalties second, and uncoverage penalties last. Note that since the penalties are created by large transformations, frames with high values for one penalty often have high values for the others.

While choosing the keys such that $\mathbf{C}(j) = \mathbf{S}_b^{-1}(j)$ minimizes the error on frame j, it is not necessarily the best choice to avoid penalties on in-between frames. In particular, uncoverage penalties on non-key frames can be reduced by scaling up the keyframe images, just as with static shots. We use the same method to choose a scaling that prevents losing important information. We use a single scale factor for all keys in the segment to avoid wobbles in the zoom. However, the use of a single scale is often insufficient as different parts of the movement may exhibit different problems, and the single scale must be chosen for the worst case over the whole duration.

6.6 Interpolation Types

Inserting a key into a constant-velocity interpolation creates a velocity discontinuity. If the direction does not change, the discontinuity is hard to notice even if there is a large discontinuity in magnitude. We explain this by observing that most of the velocity discontinuities are about the same magnitude as the velocity changes that occur between each pair of frames in the source motion (see Figure 8). Directional changes in velocity are noticeable, but often seem intentional. This is because a change in virtual camera motion direction is almost always caused by a corresponding change in the source camera's motion, which was probably motivated by something. Meandering source paths become crisply directed virtual camera paths, with similar targets.

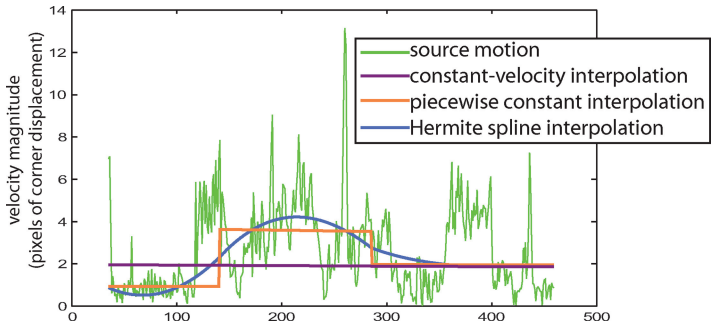


Fig. 8. Graphs of the magnitude of the camera velocities for the paths in Figure 9. The green line measures the velocities of the source motion, purple is a constant-velocity interpolation between the endpoints. The orange and blue paths add 2 keyframes and interpolate with piecewise-constant-velocity and Hermite splines respectively.

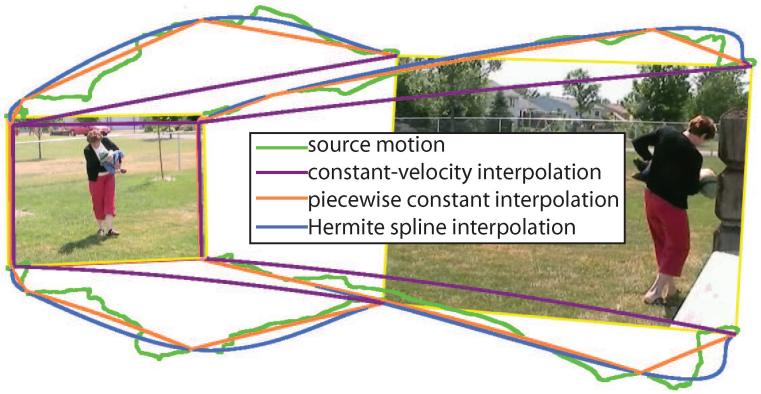


Fig. 9. Camera paths illustrated for a moving segment in the coordinate system of its base frame. The green path shows the movements of the corners of the image from the source motion. The purple paths show the constant-velocity interpolation between the endpoints. The orange and blue paths add 2 keyframes and interpolate with piecewise constant-velocity and Hermite splines respectively.

Velocity discontinuities seem to be noticeable and objectionable when the camera starts or stops. Therefore, we apply an ease curve (Section 4.4) when a moving segment connects to a static one.

As an alternative to constant-velocity interpolation, velocity-continuous interpolation is implemented using Hermite cubic splines to interpolate the matrix logarithms. While these splines remove velocity discontinuities, they also remove the velocity constancy. The tradeoff is subjective, and the effects are surprisingly subtle. Our implementation uses constant-velocity interpolation by default. An example of camera paths is illustrated in Figure 9. The corresponding velocity profiles are shown in Figure 8.

6.7 Postprocess Cleanup of Offscreen Errors

The algorithms described in the previous sections do not guarantee that identified salient objects are always visible. In situations where the above algorithms fail to adequately keep important objects on screen, we use a optional cleanup pass over the computed motions (after their optimization). The process strictly enforces that identified regions remain on screen, at the expense of all other metrics. By limiting this postprocess pass to translations only, we can create an optimization problem that is sufficiently easy

to solve. Translation is sufficient because other transformations required for smoothness are already accounted for in the motion.

For each frame, we compute a translation $\mathbf{T}(t)$ that is concatenated with the computed motion (i.e., the transformation applied to frame t is $\mathbf{T}(t)\ \mathbf{M}(t)$) such that the translation places any identified salient region on the screen, and is as smooth as possible. Because this optimization problem is separable, we can treat each axis (x and y) separately. Here we describe the optimization for x, y is handled equivalently.

For each frame t on which there is an identified salient region, we compute the minimum and maximum translation that will place the salient region completely on the screen. Even if the salient region is on the screen, there will be a range of translations that can be applied to keep it there. This provides a (potentially sparse) set of constraints $a(t_i) \leq x(t_i) \leq b(t_i)$, where i indexes the frames with identified salient regions, and a and b are the determined minimum and maximum translations for this frame. The objective function measures smoothness using finite differences

$$\begin{aligned} &\text{minimize } \textstyle\sum_{t \in \text{frames}} (x[t-1] - 2 * x[t] + x[t+1])^2 \\ &\text{such that } a(t_i) \leq x(t_i) \leq b(t_i), \quad i \in \text{frames w/known salience.} \end{aligned} \tag{4}$$

This quadratic program can be solved using standard methods [Nocedal and Wright 2006]. In practice, we find use a simple greedy approximation algorithm. Figure 12 shows an example.

7. SYNTHETIC CLOSEUPS

The long (in duration) shots of casual video can be monotonous. Filmmakers usually prefer to use a sequence of shorter shots to provide some degree of variety. "Cutting in" a closeup shot provides visual interest and a sense of connection to the subject. Re-cinematography creates synthetic closeups to add visual variety to long segments. It is not immune to degradation of image quality due to digital zoom.

Creating a closeup shot synthetically requires addressing two problems. First the subject of the closeup must be identified and tracked across the frames. In our implementation, we use the same regions identified as important to prevent offscreen errors (Section 6.1). This provides the position for each frame, but since it identifies the most salient object on each frame independently (Appendix A.2), it may jump between objects. In practice, handling noise in object localization is important anyway (see below), so this switching is handled in the motion synthesis process.

Once the subject is identified on each frame, a camera motion is created that zooms in on the subject over the duration of the video. This motion should track the object, but should also move smoothly in the world. To achieve this, we perform filtering in the common coordinate system of the base frame of the local mosaic: the center point of the salient region on each frame is projected onto the local mosaic, this trajectory is filtered, and then projected back into each image. This process factors out the motion of the original camera. The determined trajectory is then used as the center of the zoom for the closeup.

Our implementation filters the trajectory first with a median filter, to reject outliers, and then with a Gaussian to create a smooth motion. The scale of the zoom is chosen such that a zoom centered at the filtered trajectory is as large as possible but still includes the identified objects.

Once the closeup motion is determined, there is the question of how to use it. "Cutting it in" is likely to be a jump cut. Instead, we make a transition between the original motion and the closeup, blending between the two using matrix logarithms. Such transition zooms are prevalent in casual videos, although they are aesthetically less desirable than a proper cut [Ang 2005].

Our re-cinematography system can identify candidates for automatic closeup creation by finding long, relatively static, segments ($p_{cuseg} = 20$ *seconds*) where there is an identified region of interest. In practice, we find that zoom candidates are best identified manually as the use of a closeup is more of a

"storytelling" choice, requiring an appropriate subject. The system provides assisted modes where the user can identify either a segment, or a time range, in which a closeup should be inserted.

8. EXPERIMENTS

Our re-cinematography prototype is implemented on PCs running Windows XP using compressed AVI files. Motion estimation, homography evaluation, and motion salience are computed as a preprocess. Our implementation of these standard components is inefficient. State of the art systems provide near real-time performance [Bevilacqua and Azzari 2006; Nistér 2005; Osian and Van Gool 2004].

The main steps of re-cinematography are performed in our interactive system and are all much faster than real time. The system caches the video in texture memory for efficient playback and precomputes all $O(n^2)$ stabilizations, meaning that memory becomes an issue after (approximately) 2700 frames on a machine with 2G of memory. After the video is read from disk (which happens faster than real time) none of the steps (including the $O(n^2)$ step) take more than a few seconds (for an entire 2-minute video), when the data fits into memory.

Our prototype implementation applies the transformations to the images using the graphics hardware via OpenGL. Video frames are applied as textures to polygons that are transformed according to $\mathbf{M}(t)$. Uncovered area filling (Section 6.2) is performed by using the stabilization transformations for neighboring frames, and drawing the textured polygons in farthest (temporally) to nearest order. By caching the video in texture memory, we can achieve real-time previews, even on notebook computers.

Most of our test data (and all of the nonsynthetic examples in this article) were recorded using a Sanyo Xacti C6 camera. It records Quicktime MPEG4 at 640x480 at 30 frames per second. All examples use automatic white balance and exposure. Most examples were originally shot using the camera's digital image stabilization, which does degrade input quality. In addition to real examples, we experimented with videos created specifically for testing. These examples illustrate the differences between re-cinematography and filtering. Some examples were footage shot in our lab, while others were created synthetically using a compositing tool. The latter have the advantage that we can compute the homographies directly, without relying on motion estimation.

8.1 Comparative Tests

Because most of our examples were shot with in-camera image stabilization, the re-cinematography results are effectively a comparison with current technology. For a more fair comparison, we also have implemented the low-pass filter method from Matsushita et al. [2006], however we apply it within the same video processing framework that is used for re-cinematography. The same motion estimation and in-painting implementation are used for both approaches.

Figure 10 shows a synthetic example simulating a videographer filming a UFO weaving over a city. The "videographer" tries to keep the saucer centered (i.e., following the bobbing), but has an unsteady hand (simulated by adding white noise). Image stabilization methods can remove the noise, but leave the smooth (approx 1hz) zig-zagging of the camera, which can be quite sea-sickness inducing—even with a larger filter kernel than suggested in Matsushita et al. [2006]. Also, the saucer is no longer tracked perfectly, and has an unsteady wobble. The re-cinematography result has a camera that moves in straight pans, leaving a smoothly weaving saucer. The resulting camera path began as a direct interpolation between the beginning and end position, with a key inserted where the salient object was maximally out of the frame. Because of the noise, the key is inserted such that the path of the camera is not precisely horizontal.

The smooth oscillations appear in real examples as well. Figure 11 shows a video taken while walking down a corridor. With standard image stabilization, the oscillations due to the videographer's stride are very apparent. Our approach removes these oscillations. Also note that the forward motion is not

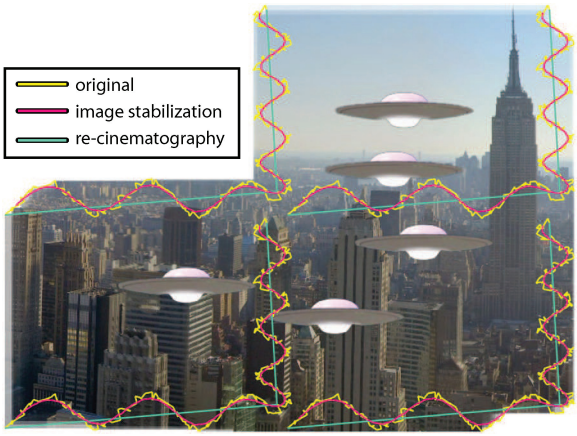


Fig. 10. Synthetic video example of a flying saucer zig-zagging over New York. The video simulates a videographer tracking the saucer. Lines in this mosaic image show the paths of the corner of the video images. Yellow is the original (noisy) path. Magenta shows the result of low-pass filtering from [Matsushita et al. 2006]. Cyan shows the re-cinematography result.

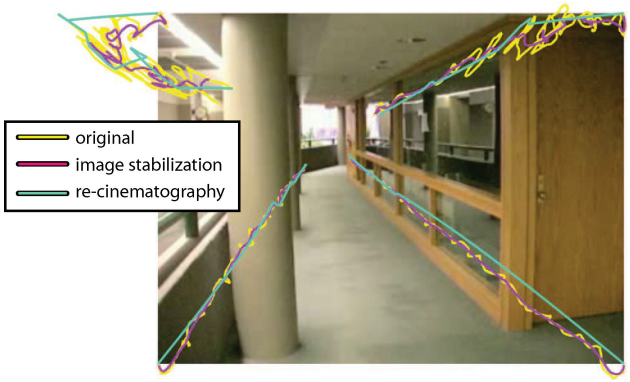


Fig. 11. Video taken walking down hallway. Image corner paths shown using the first frame as base. Yellow is the original motion, Magenta is filtered using the method of [Matsushita et al. 2006], and Cyan is the re-cinematography solution.

well represented by the homographic model, yet our system can still create a good result by inserting keyframes periodically.

We have also compared our results with DeShaker (www.guthspot.se/video/deshaker.htm.), a Shareware postprocess image stabilization program. The software's filtering-based approach is able to remove the high-frequency movements on almost all of our examples. Re-cinematography offers a clear advantage in making the static segments more stable. For videos with static segments connected by short motions (which are very common among our sample videos), the stable static shots with directed connections in the re-cinematography results are particularly compelling.

For moving segments, the differences between DeShaker and re-cinematography is more subjective. DeShaker's smooth motions are quite different than the point-to-point motions produced by our approach. In both systems, the worst artifacts are the uncovered screen areas as neither system offers an advanced in-painting scheme. DeShaker's implementation of scale-to-cover is problematic as it varies the scale too rapidly causing rapid oscillations in size (this problem is noted in the manual). At the opposite extreme, our scale-to-cover solution picks a single scale factor for each segment

(to avoid oscillations in size) which is often insufficient, especially since the scale is chosen conservatively. An advanced inpainting method such as Matsushita et al. [2006] or Wexler et al. [2007] is a better cure for uncovered regions.

Improvements over the Original Re-Cinematography System: There are some notable differences between the methods described in this article, and the original system described in Gleicher and Liu [2007]. The improvements were motivated by problems in results noticed in informal evaluation of the original system. For example, the improved segmenting method (Section 5.1) that considers zoom avoids problems in cases where the previous method would create a long dynamic segment that performed the zooming operations at inappropriate times (beginning a zoom at the beginning of a shot, rather than where the zoom actually began). The postprocess cleanup (Section 6.7) was a direct reaction to viewers expressing the importance of keeping important objects on-screen—even if other ideals were violated. In general, the visual differences between the results of the old and new systems are very subtle when both make good choices. However, the new system (appears to) fail less often. Anecdotally, every example that worked with the old system works with the new one, but we have many examples where the old system fails but the new one achieves acceptable results.

8.2 Subjective Evaluation

We have experimented with our implementation on a collection of home videos. The results are difficult to convey in still images. They exhibit very stable static segments, and motions appear directed and intentional. Pictures from these examples appear throughout the article. A particularly challenging example is shown in Figure 12.

The moving segments produced by the system achieve their intended effect. The desirability of these effects is a subjective question. Our initial informal evaluation suggests that viewers appreciate the "improved" motion, especially when the original source video was shaky. However, this may just be confirming what camera manufacturers already know: that image stabilization is a good thing.

Viewer preference of re-cinematography, or of cinematography in general, is difficult to assess for a number of reasons. The camera movements are only one aspect of how a video makes its impression on the viewer. And ultimately, the effectiveness (or enjoyability) of a film or video is the overall impression. Almost by definition, the effects of cinematography should be subtle (they should not detract from the content of the video). Often, the effects build up over time, small changes in a number of individual shots lead to an overall effect. This makes difficult the assessment of clips out of context.

8.3 Formal Pilot Study

To gain better insight into the subjective reactions to re-cinematography, we ran a pilot study in addition to our more casual demonstrations. Subjects were shown a selection of clips in both source and processed forms, and asked about their preferences. Only after they had provided their initial reactions were they asked more detailed questions about their preferences. Our study had six subjects who were all graduate students, none of whom had extensive video production experience. The clips chosen for the study were from the author's collection, the study subjects had no connection to the people or places in the clips. Subjects were shown 5 pairs of clips, in random order. For each clip the subjects were shown both the source footage and the re-cinematography result, in random order. After being given the opportunity to give their initial reactions, subjects were asked the following:

(1) Did you notice differences between the two videos?
(2) Which one do you prefer?
(3) Why do you like the one you picked?
(4) Did you notice any artifacts in the videos?

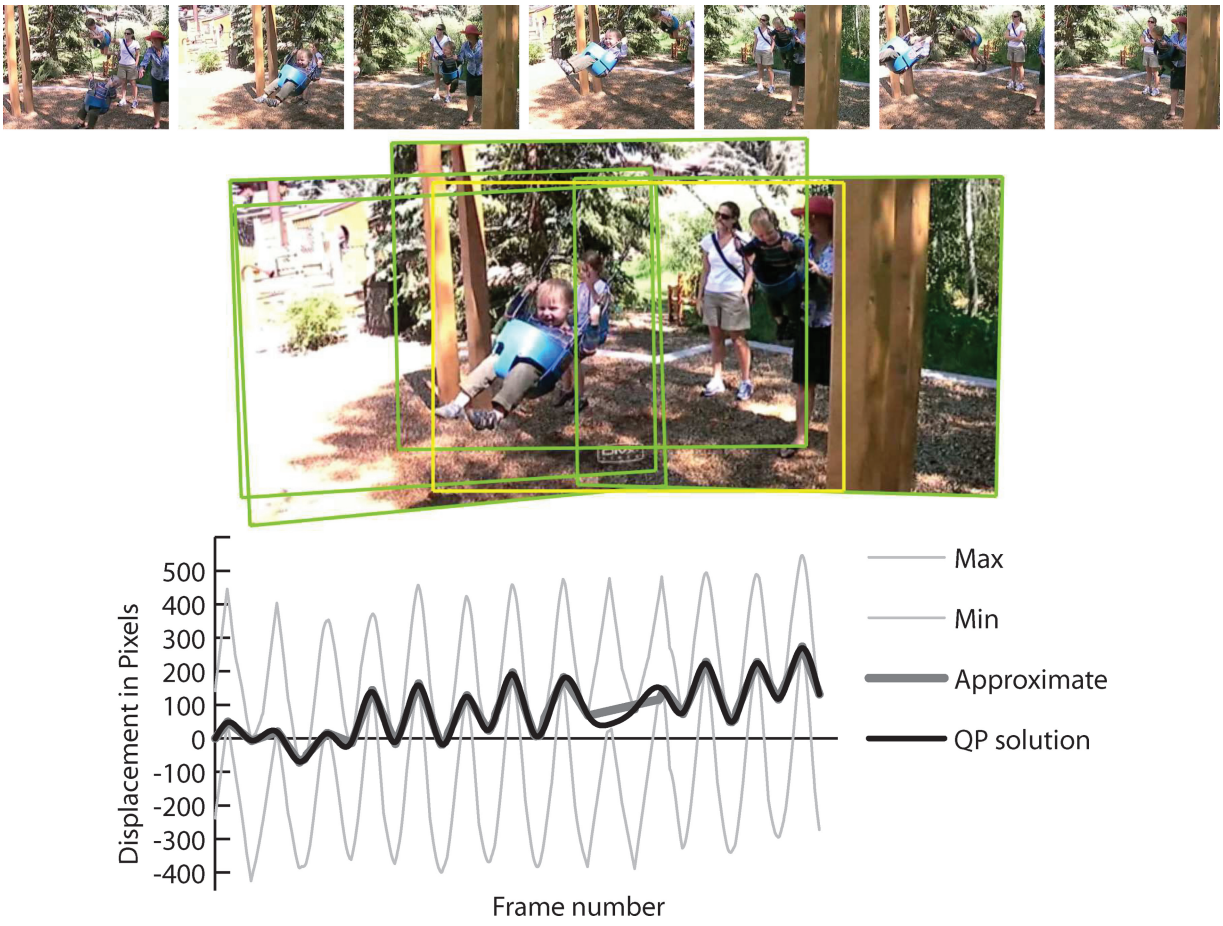


Fig. 12. (top) Frames from a 30-second source video tracking a child on a swing. (middle) A mosaic of several frames (overlayed, not blended) showing the range of movement in the video. Green boxes show extreme frames, the yellow rectangle outlines the position of the base frame. (bottom) Using a single static shot framed to the base frame leads to the important object (the swinging boy) going out of the frame. Motion is added using the cleanup process (Section 6.7) to keep him on screen. The graph indicates the minimum and maximum amount of x translation for each frame that puts the important object on screen, as well as the solution for the best translation motion that meets these boundaries. Two solutions are shown: one exactly solves the quadratic program of Equation (4); the other is the approximate solution computed by our implementation.

Fig. 13. Beginning and end frames of a zoom segment shown in Figure 14.

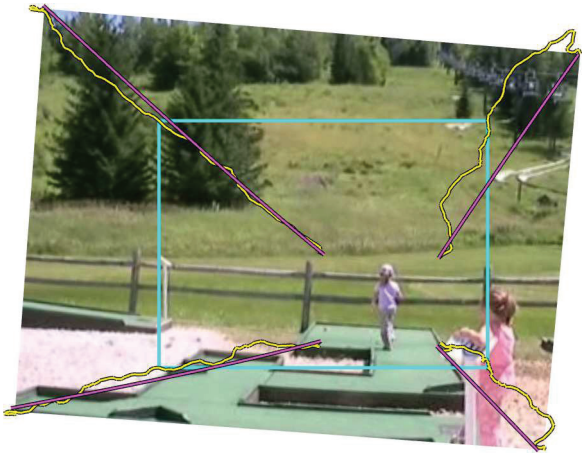


Fig. 14. Visualization of re-cinematography of zoom segment. The first frame is shown transformed to the base frame (whose boundary is shown in cyan). Yellow paths trace the corners of the original video, magenta (darker) traces the corners of the resulting camera motion.

Our pilot study confirmed the subjectivity, and the influence of context. For example, one example clip contained a girl bouncing on a "bungie swing." The source video tracked the girl, leading to very large tilting motions, while the re-cinematography result stabilized the clip (so that the girl moved relative to a static frame). Some people preferred the source, indicating that the camera motion gave them a feeling of the action, while others felt the movement was hard to watch. We would expect the preference to be affected by context: as the clip grows longer, the novelty of the camera movement might grow more tiresome; when cut between other clips, the movement might be jarringly different (or a refreshing change); if the viewer had developed an emotional connection with the subject (or had one because they knew her), they might prefer camera work that allowed them to better focus on her reaction; etc.

In the pilot study, all of our subjects noticed the visual artifacts from bad in-painting without prompting. Their reaction to the artifacts varied: for example, 5 out of 6 respondents preferred the re-cinematography result for Figure 7, despite the fact that they all noticed the inpainting artifacts. On other videos, people reacted negatively to blurring (both from inpainting and resampling). However, how they viewed the tradeoff of these artifacts versus the motion improvement varied. The sensitivity to artifacts, as well as how willing a viewer is to accept them as a tradeoff, is clearly subjective, personal, and context dependent.

8.4 Failures and Limitations

Re-cinematography creates a new video by transforming an existing one. This places a number of fundamental limitations on the technique. Because re-cinematography is limited to transforming the frames of the source video, if it is not in the source video, re-cinematography cannot put it into the result. Furthermore, the amount of change that can be applied to a frame is limited, which in turn limits the amount of change that can be made to the motion.

A richer class of transformations (as in Buehler et al. [2001]) would allow larger changes without distortion. However, poor quality source video makes estimating rich motion models difficult, and interpolation of more complex transformations may be problematic. Our simple inpainting implementation

also limits the amount of change that can be made to a frame. Recent inpainting methods [Matsushita et al. 2006; Wexler et al. 2007] would provide improved performance.

Another limitation in re-cinematography is the reliance on motion and importance estimation. While state-of-the-art methods may provide better performance than our prototype, no method can be completely reliable as some video is inherently ambiguous.

Given the limitations, we recognize that re-cinematography cannot help all videos. We consider a result from our system a failure if it produces a result that is worse than the source. The system should recognize places where it cannot help.

One category of failure is when the system incorrectly assesses a bad homography as reliable. This leads to transformations that give visual artifacts, including distortions and small jiggling. These errors come from poor motion estimation and scenes not well modeled by the projective assumptions. In future work, we believe that it is possible to make a reliable assessment of both of these kinds of errors and to use this to restrict the amount of transformation used.

Another class of failure is when the system makes a bad tradeoff, for example, choosing to show a large uncovered area rather than a less smooth motion. Part of the problem is that these tradeoffs often depend on the image, the application, and the viewer's preference. A related class of failure is when the system misses an opportunity to make an improvement because it is being too conservative.

Our approach, like any stabilization approach, makes tradeoffs in image quality to achieve better motion quality. There is an implicit assumption that some loss of image quality is worthwhile to gain an improvement in motion. Re-cinematography discards image detail, particularly when it zooms in. Given that much casual video is often presented in a small format (e.g., on a video sharing site, a portable media player, etc.), these details might be lost anyway.

We have implicitly decided to preserve the "style" of casual videos: because we do no editing, we preserve the long (in duration) shots with considerable camera motion. While re-cinematography can improve how this style is implemented by improving the camera movements, it keeps the (aesthetically questionable) casual style. Books on filmmaking stress the importance of obtaining the correct footage to allow for proper editing. However, casual video is effectively defined by the fact lack of the planning for good filmmaking.

9. DISCUSSION

Our re-cinematography approach can improve the camera dynamics in a wide range of casual videos. Wandering handheld motions are turned into smooth directed movements that exhibit ease-in and out where appropriate, and the camera stays rigidly fixed when this is similar enough to the source movement. Better evaluation is important to understand the performance and utility of our system.

Our current system is almost completely automated. A more appropriate use of the technology may be as a more interactive authoring tool where a user has more control over the process. A user can make decisions about quality tradeoff parameters based on their preferences and knowledge of the video's content and application. The user can also provide high level insight based on the semantics of the video and the desired storytelling in the result. By definition, this latter information cannot be obtained without the user.

Ultimately, our approach is limited by the quality of the source footage. No post-process is likely to be a replacement for good planning, skillful camera work, proper camera supports, and artistic talent. Re-cinematography will not turn casually captured video clips into masterpieces of film. However, the techniques presented in this paper can effectively improve the apparent camera dynamics in casual video.

A. APPENDIX: VIDEO PREPROCESSING METHODS

Re-cinematography relies on motion estimates and important object identification from the source video. These standard computer vision components are performed as a preprocess to our system. Here we describe the methods we have used for our experiments. These components could easily be replaced by more state of the art methods.

A.1 Motion Estimation

Motion estimation is common to a number of image, video, and vision applications and has been studied extensively. Szeliski's recent survey [2006] is encyclopedic and insightful. In this paper, we use a homography to describe the geometrical relationship between every two consecutive frames. A homography is a 2D perspective matrix described by 8 parameters. The relationship between corresponding points in two images I^t and I^{t-1} can be described as follows:

$$\begin{bmatrix} sx^t \\ sy^t \\ s \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & 1 \end{bmatrix} \begin{bmatrix} x^{t-1} \\ x^{t-1} \\ 1 \end{bmatrix}. \tag{5}$$

We use a feature-based method to estimate the homography between every two consecutive frames in a video. Among many local feature descriptors, SIFT [Lowe 2004] is reported to perform best by recent work [Mikolajczyk and Schmid 2005]. We use SIFT points as features. The best candidate match for each SIFT point in one image is found by identifying its nearest neighbor in the other image. As suggested by the authors, a criterion of a good SIFT match can be the ratio between the distance of the closest neighbor to that of the second closest neighbor. Since there are normally a large number of SIFT points in each frame, it is time consuming to search for the closest neighbor globally. Also, consecutive frames of video usually have small changes. Therefore, local searching is both more efficient and accurate. We currently use a tile based method. The image is divided into uniform tiles, and the SIFT points are binned to each tile. When searching for the matching SIFT points, only the points in the corresponding tile will be compared. Once we find the candidate matching pairs, we use RANSAC [Fischler and Bolles 1981] method to robustly estimate the homography.

A.2 Important Object Identification

Truly measuring importance would require semantic and intentional understanding. To estimate importance, we use heuristic rules from image and video retargeting [Suh et al. 2003; Chen et al. 2003; Liu and Gleicher 2006]: identifiable objects (especially faces) and visually salient regions are likely to be important. Our implementation detects faces using the method of Viola and Jones [2001]. For salience, research suggests that motion salience is a strong cue [Rosenholtz 1999], corresponding with our intuition that moving objects (i.e., with motion relative to the camera motion) are likely to be interesting. We compute motion saliency as the contrast between the local motion vectors computed by optical flow, and the global motion vectors given by the homographies [Liu and Gleicher 2006]. To get consistent motion saliency across frames, motion saliency is space-time filtered by an isotropic filter due to the camera motion. Alternatively, this isotropic filtering can be implemented by first aligning the frames in the filter window to the filtered frame, and then applying a space-time Gaussian filter, which can take advantage of the linearity and therefore operate quickly.

Given the motion saliency, we detect from each frame an moving object by finding the smallest rectangle containing a certain amount of the total saliency. We use a greedy method to search for the moving object. First, we find a 20×20 rectangle containing the maximal saliency in a brute-force way. For speed, we build a summed area table beforehand so that the total saliency value in the rectangle can be calculated in a constant time [Crow 1984]. Second, we grow the rectangle in one of the four

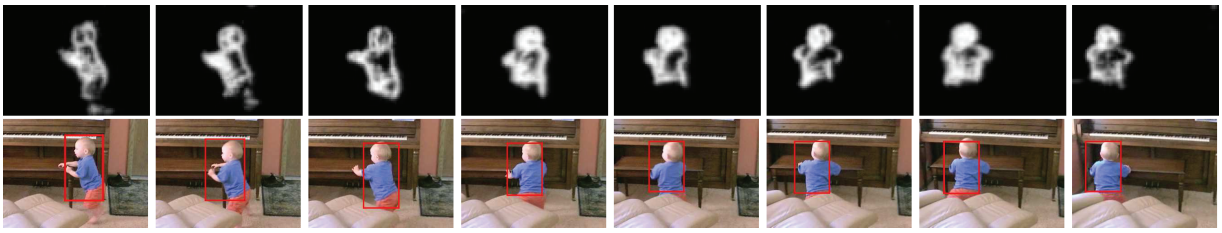


Fig. 15. Top: motion saliency, bottom: moving objects indicated by red rectangles.

directions, left, right, top, and bottom. We select the direction where growing can increase the mean importance value in the rectangle most significantly. This step is repeated until the rectangle contains sufficient saliency. An example of motion saliency and moving objects is illustrated in Figure 15.

To compute the salience bounding box for a frame t, we consider a window of frames $t \pm p_{sbw}$ ($p_{sbw} = 8$ frames). This implements the cinematographic principle of leading: it gives a moving object space so that the viewer can see where it is going. For each frame i in the window, its salience object box is transformed to t via $\mathbf{S}_t(i)$. A histogram counts the number of frames with salience in each region of the image. The salience bounding box is determined from the histogram cells with counts above a threshold.

ACKNOWLEDGMENTS

We thank Rachel Heck for her assistance in preparing demonstrations and Stephen Wright for providing the real solutions for the quadratic programming problem shown in Figure 12. The city background image in Figure 10 is by S. Schultz and used under a Creative Commons license.

REFERENCES

Achanta, R., Yan, W.-Q., and Kankanhalli, M. 2006. Modeling intent for home video repurposing. *IEEE Multimedia 13*, 46–55.

Adams, B., Venkatesh, S., and Jain, R. 2005. IMCE: Integrated media creation environment. *ACM Trans. Multimed. Comput. Comm. Appl 1*, 211–247.

Alexa, M. 2002. Linear combinations of transformations. In *Proceedings of the ACM SIGGRAPH International Conference on Computer Graphics and Interactive Techniques*. 380–387.

Ang, T. 2005. *Digital Video Handbook*. Dorling Kindersley.

Arijon, D. 1991. *Grammar of the Film Language*. Silman-James Press.

Bennett, E. P. and McMillan, L. 2003. Proscenium: A framework for spatio-temporal video editing. In *Proceedings of the 11th ACM International Conference on Multimedia (MULTIMEDIA'03)*. ACM, New York, NY, 177–184.

Bevilacqua, A. and Azzari, P. 2006. High-quality real time motion detection using ptz cameras. In *Proceedings of the IEEE International Conference on Video and Signal Based Surveillance (AVSS'06)*. 23.

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

Bordwell, D. and Thompson, K. 1997. *Film Art: An Introduction*. McGraw-Hill.

Brandon, B. 2005. *The Complete Digital Video Guide*. Reader's Digest.

Brown, B. 2002. *Cinematography: Theory and Practice: Imagemaking for Cinematographers, Directors & Videographers*. Butterworth-Heinemann.

Buehler, C., Bosse, M., and McMillan, L. 2001. Non-metric image-based rendering for video stabilization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. Vol. 2. 609–614.

Casares, J., Long, A. C., Myers, B. A., Bhatnagar, R., Stevens, S. M., Dabbish, L., Yocum, D., and Corbett, A. 2002. Simplifying video editing using metadata. In *Proceedings of the 4th Conference on Designing Interactive Systems: Processes, Practices, Methods, and Techniques*. ACM, London, England, 157–166.

Chalfen, R. 1987. *Snapshot Versions of Life*. Bowling Green State University Press.

Chen, L.-Q., Xie, X., Fan, X., Ma, W.-Y., Zhang, H.-J., and Zhou, H.-Q. 2003. A visual attention model for adapting images on small displays. *Multimed. Syst. 9*, 4, 353–364.

CHRISTIE, M., MACHAP, R., NORMAND, J.-M., OLIVIER, P., AND PICKERING, J. 2005. Virtual camera planning: A survey. In *Proceedings of Smart Graphics*. 40–52.

CROW, F. C. 1984. Summed-area tables for texture mapping. In *Proceedings of the ACM SIGGRAPH International Conference on Computer Graphics and Interactive Techniques*. 207–212.

DONY, R., MATEER, J., AND ROBINSON, J. 2005. Techniques for automated reverse storyboarding. *IEE J. Vision, Image Signal Process. 152*, 4, 425–436.

FISCHLER, M. A. AND BOLLES, R. C. 1981. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Comm. ACM 24*, 6, 381–395.

GIRGENSOHN, A., BORECZKY, J., CHIU, P., DOHERTY, J., FOOTE, J., GOLOVCHINSKY, G., UCHIHASHI, S., AND WILCOX, L. 2000. A semi-automatic approach to home video editing. In *Proceedings of the Annual ACM Symposium on User Interface Software and Technology*. 81–89.

GLEICHER, M. 1997. Projective registration with difference decomposition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 331–337.

GLEICHER, M. AND LIU, F. 2007. Re-cinematography: Improving the camera dynamics of casual video. In *Proceedings of the 15th International Conference on Multimedia*.

GOVINDU, V. M. 2004. Lie-algebraic averaging for globally consistent motion estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

HANSEN, M. AND MCDOWEL, L. 2001. Apparatus and method for removing blank areas from real-time stabilized images by inserting background information. U.S. Patent 6211913.

HECK, R., WALLICK, M., AND GLEICHER, M. 2007. Virtual videography. *ACM Trans. Multimed. Comput. Comm. Appl. 3*, 1, 4.

HUA, X.-S., LU, L., AND ZHANG, H.-J. 2004. Optimization-based automated home video editing system. *IEEE Trans. Circ. Syst. Video Tech. 14*, 5 (May), 572–583.

IRANI, M. AND ANANDAN, P. 1998. Video indexing based on mosaic representations. *Proc. IEEE 86*, 5, 905–921.

IRANI, M., ANANDAN, P., AND HSU, S. 1995. Mosaic based representations of video sequences and their applications. In *Proceedings of the International Conference on Computer Vision*. 605–611.

IRANI, M., ROUSSO, B., AND PELEG, S. 1994. Recovery of ego-motion using image stabilization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 454–460.

JOHNSTON, O. AND THOMAS, F. 1981. *The Illusion of Life: Disney Animation*. Abbeville Press.

KATZ, S. D. 1991. *Film Directing Shot by Shot: Visualizing from Concept to Screen*. Michael Wiese Productions.

KAVAN, L., COLLINS, S., O'SULLIVAN, C., AND ZARA, J. 2006. Dual quaternions for rigid transformation blending. Tech. Rep. TCD-CS-2006-46, Trinity College Dublin.

KENDER, J. AND YEO, B.-L. 2000. On the structure and analysis of home video. In *Proceedings of ACCV*.

KIRK, D., SELLEN, A., HARPER, R., AND WOOD, K. 2007. Understanding videowork. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 61–70.

LITVIN, A., KONRAD, J., AND KARL, W. 2003. Probabilistic video stabilization using kalman filtering and mosaicking. In *Proceedings of the IS&T/SPIE Symposium on Electronic Imaging, Image, and Video Comm.* 663–674.

LIU, F. AND GLEICHER, M. 2006. Video retargeting: Automating pan and scan. In *ACM Multimed.* 241–250.

LOWE, D. G. 2004. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vision 60*, 2, 91–110.

MATSUSHITA, Y., OFEK, E., GE, W., TANG, X., AND SHUM, H.-Y. 2006. Full-frame video stabilization with motion inpainting. *IEEE Trans. Pattern Anal. Mech. Intel. 28*, 7, 1150–1163.

MEI, T., HUA, X.-S., AND ZHOU, H.-Q. 2005. Tracking users' capture intention: A novel complementary view for home video content analysis. In *Proceedings of ACM Multimedia*. 531–534.

MIKOLAJCZYK, K. AND SCHMID, C. 2005. A performance evaluation of local descriptors. *IEEE Trans. Pattern Anal. Mech. Intel. 27*, 10, 1615–1630.

NISTÉR, D. 2005. Preemptive RANSAC for live structure and motion estimation. *Machine Vision Appl. 16*, 5, 321–329.

NOCEDAL, J. AND WRIGHT, S. J. 2006. *Numerical Optimization*, 2nd ed. Springer.

OSIAN, M. AND VAN GOOL, L. 2004. Video shot characterization. *Machine Vision Appl. 15*, 172–177.

PAN, Z. AND NGO, C.-W. 2004. Structuring home video by snippet detection and pattern parsing. In *Proceedings of the ACM SIGMM Workshop on Multimedia Information Retrieval*. 69–76.

ROSENHOLTZ, R. 1999. A simple saliency model predicts a number of motion popout phenomena. *Vision Research 39*, 19, 3157–3163.

SUH, B., LING, H., BEDERSON, B. B., AND JACOBS, D. W. 2003. Automatic thumbnail cropping and its effectiveness. In *Proceedings of the Annual ACM Symposium on User Interface Software and Technology*. 95–104.

Szeliski, R. 1996. Video mosaics for virtual environments. *IEEE Comput. Graph. Appl. 16*, 2, 22–30.

Szeliski, R. 2006. Image alignment and stitching: A tutorial. Tech. rep. MSR-TR-2004-92, Microsoft Research.

Teodosio, L. and Bender, W. 2005. Salient stills. *ACM Trans. Multimed. Comput. Commun. Appl. 1*, 1, 16–36.

Viola, P. and Jones, M. 2001. Rapid object detection using a boosted cascade of simple features. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 511–518.

Wexler, Y., Shechtman, E., and Irani, M. 2007. Space-time completion of video. *IEEE Trans. Pattern Anal. Mech. Intel. 29*, 3, 463–476.

Wood, D. N., Finkelstein, A., Hughes, J. F., Thayer, C. E., and Salesin, D. H. 1997. Multiperspective panoramas for cel animation. In *Proceedings of the ACM SIGGRAPH International Conference on Computer Graphics and Interactive Techniques*. 243–250.

Yan, W.-Q. and Kankanhalli, M. S. 2002. Detection and removal of lighting and shaking artifacts in home videos. In *Proceedings of ACM Multimedia*. 107–116.

Received January 2008; revised May 2008; accepted May 2008