Video Retargeting: Automating Pan and Scan

Feng Liu
Department of Computer Sciences
University of Wisconsin, Madison
1210 West Dayton Street
Madison, Wisconsin, 53706
fliu@cs.wisc.edu

Michael Gleicher
Department of Computer Sciences
University of Wisconsin, Madison
1210 West Dayton Street
Madison, Wisconsin, 53706
gleicher@cs.wisc.edu

ABSTRACT

When a video is displayed on a smaller display than originally intended, some of the information in the video is necessarily lost. In this paper, we introduce *Video Retargeting* that adapts video to better suit the target display, minimizing the important information lost. We define a framework that measures the preservation of the source material, and methods for estimating the important information in the video. Video retargeting crops each frame and scales it to fit the target display. An optimization process minimizes information loss by balancing the loss of detail due to scaling with the loss of content and composition due to cropping. The cropping window can be moved during a shot to introduce virtual pans and cuts, subject to constraints that ensure cinematic plausibility. We demonstrate results of adapting a variety of source videos to small display sizes.

Categories and Subject Descriptors

I.4.9 [**Image Processing and Computer Vision**]: Applications

General Terms

Algorithms, Human Factors

Keywords

Video retargeting, Video editing, Mobile multimedia, Importance estimation

1. INTRODUCTION

Viewing video on small screens is becoming increasingly common as portable devices become more capable and popular. Unfortunately, most source material is originally intended for larger displays, such as televisions and theater screens. If such video is presented naïvely, by simply scaling it to fit the small screen, important parts of the image become too small to see. To make matters worse, small displays often have different aspect ratios than larger ones, requiring either an anisotropic "squish" or padding the video to fill the display. Small displays are limited to display less content than larger ones; our goal is to enable effective small display by retaining what is important.

This paper considers the problem of *video retargeting,* that is, adapting a video so that it is better suited for viewing on a display different than was originally intended. Video retargeting applies two operations to each frame of a video: cropping, which discards information outside of a window and disturbs the composition of the image; and scaling, which loses details of the image especially as objects become too small to recognize, and distorts the image if the scaling is anisotropic.

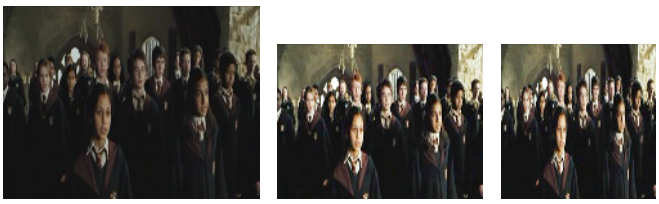
In this paper, we introduce an approach for automatically retargeting video to displays of different sizes and aspect ratios. This intelligent retargeting solution uses the video content to determine how to best combine cropping and scaling: unimportant aspects of the frame are cropped away so that more important content appears at a larger scale. We cast the retargeting as an optimization problem: what new video least damages the content of the original video. By moving the cropping window, video retargeting can create virtual pans and cuts to better portray dynamic shots. While our focus is on adapting edited films and videos for small displays, the methods are also applicable for automatically adapting wide format videos (such as feature films) to other aspect ratios (such as standard television).

Cropping discards considerable information. Not only is the content of the cropped portion lost, but we also lose the intended composition of the original frame. Composition is important in video as filmmakers use it in subtle ways to convey emotion and story. However, for small devices the alternative, downsampling the image to a tiny size that where objects are potentially too small to be recognized, is often worse. In essence, we choose to selectively lose some information from cropping in the hope of avoiding losing all information from scaling. Examples are shown in Figure 1.

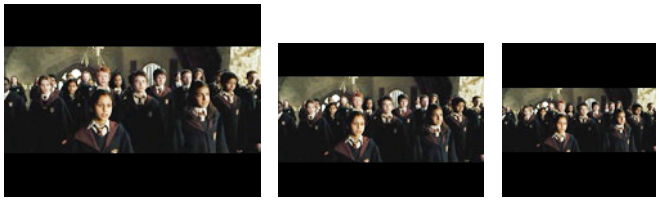
In video, the motion on screen has significance beyond individual frames. Not only do objects move, but also filmmakers move the camera to achieve their desired goals. These latter effects are often subtle, yet significant: a zoom-in to create a feeling of connection or the timing of cuts to establish pacing. Therefore, video retargeting must not only consider how each frame is cropped, but also how this cropping affects motion. Rather than computing the cropping for each frame, we must be careful not to introduce new motions that will be obvious artifacts or significantly destroy

MM'06, October 23–27, 2006, Santa Barbara, California, USA.

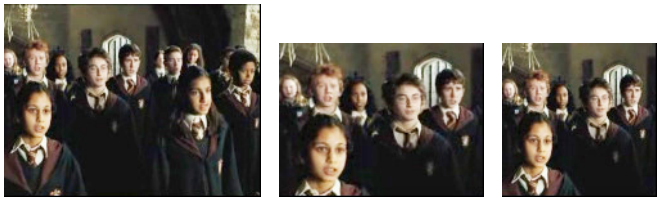
(a) Original DVD Frame 480 × 200 pixels



(b) Retarget by scaling



(c) Retarget by letterboxing



(d) Retarget by our approach

Figure 1: An example of retargeting a frame of a feature film from a widescreen (2.4:1) DVD to a variety of other devices, both naïvely and using our approach. Targets are 200 × 150, 160 × 120, and 120 × 120 pixels respectively. Note that while both the left and center targets have the same 4:3 aspect ratio, the different sizes lead to different retargeting solutions. Source © 2004 Warner Home Video.

important existing motions. In particular, the cropping window may be moved either smoothly to introduce a virtual pan or discontinuously to introduce a virtual cut. However, the introduction of the virtual camera motions must be *cinematically plausible:* they must look as if they could have been part of the original film.

Implementing video retargeting involves two components. A system must determine what is important in the video then choose a retargeting that preserves it. In Section 3 we define a framework where heuristic penalties measure the amount of information lost due to a retargeting operation. Video retargeting attempts to find the cropping window for each frame that minimizes this penalty over the entire video. In Section 4 we describe how we find solutions to this minimization that enforce constraints of cinematic plausibility. The final sections of the paper present examples and discuss results.

Automation is important in video retargeting because there is an ever expanding set of potential target displays. Manual retargeting may provide better results for a specific target, but our automated video retargeting allows video to be adapted to a wide variety of devices. For each different display size and aspect ratio, the system can determine the best way to retarget the video.

The main contribution of this paper is an approach for video retargeting that works for edited source material such as feature films. To do this, we also contribute a novel importance metric that accounts for object and camera motion, a problem formulation that penalizes various types of loss and methods for minimizing these penalties subject to the constraints of cinematic plausibility.

1.1 Overview and Example

To motivate and explain our methodology we consider a feature film in the common 2.4:1 aspect ratio[1]. To display the entire frame of the film on a standard television screen (4:3 aspect ratio), one would either need to "squeeze" the image considerably (Figure 1(b)), or display the frame as a narrow band across the frame, padding the frame with blank space (known as "letterboxing," see Figure 1(c)). As an alternative, the sides of the film image might be cropped off, creating an image with 4:3 aspect ratio. When this copped image is shown on the television display it utilizes the full height of the screen and is commonly called a *fullscreen* presentation.

Fullscreen presentations of movies are created by a process called *pan and scan* where an operator chooses how to crop the image to the required aspect ratio. To our knowledge, this process is done manually, and typically only involves cropping the left and right edges. The pan-and-scan operator chooses what is important in each frame and moves the cropping window accordingly. Our work automates this process, and extends it to crop from all edges of the frame.

Pan-and-scan is problematic as it discards a portion of the original frame, eliminating parts of the image and destroying the composition of the image. However, a face that is 1/10th of the vertical size of the frame (or in fullscreen presentation) would be only 1/18th of a screen tall in letter box. On a large display, a viewer may be willing to sacrifice the size, but on a small display even 1/10th of the image may make the face too small to be recognizable, so a viewer may be more willing to sacrifice the edges of the image such that the face is recognizable.

We can be more aggressive in our cropping than traditional pan and scan by also cropping vertically. While we may lose more of the original image content and composition, such aggressive cropping may be justified for a small display as the important face may not be recognizable otherwise. By appropriately cropping a video, we can *retarget* it to a smaller display. Cropping may be useful even if the source and target video have the same aspect ratio.

Our automatic video retargeting process considers each shot of the video independently (§4). First, it determines what are the important parts of each frame (§3). Then for each shot it determines how to crop the shot so that it best

[1]Modern feature films are presented in a variety of aspect ratios, almost all are wider than the 16:9 aspect of "widescreen" television.

preserves the content of the original video. Choosing the cropping requires balancing between a number of competing goals, such as showing as much of the image as possible yet not shrinking the important aspects of the image too much. To perform this balancing, our approach assigns penalties for many types of information loss, and determines the cropping by choosing it such that the weighted sum of the penalties are minimized (§4).

In some cases, a single cropping rectangle is not the best choice for an entire shot. For example, if an important object moves across the source frame, the cropping window can move with the object so that it is always in view at an appropriate size. Moving the cropping window creates effects similar to moving the camera during filming. Our approach (§4.2) limits the movements it considers to ones that are *cinematically plausible,* that is, ones that appear as if they may have been part of the original film. In general, we are conservative in not violating the "rules" taught to beginning filmmakers: an experienced filmmaker may violate these rules to create a dramatic effect, but the retargeting process should not introduce dramatic effects not present in the original film. Similarly, our approach considers discontinuous movements of the cropping rectangle to introduce cuts, however these cuts are applied cautiously to ensure their cinematic plausibility.

Our implementation of video retargeting first pre-processes the source video to identify shot boundaries and to identify the important aspects of it (§3). Once pre-processing is complete, different target video sizes can be produced. Given a target video size, the system considers each shot, finding the best static cropping and virtual camera motions, and selects the one that minimizes information loss. Given a pre-processed source video, the system can generate videos retargeted for a variety of different target sizes. For example, we retarget DVDs of feature films to sizes appropriate for small media players , PDAs and cellular phones.

2. RELATED WORK

Feature films are retargeted for television through pan and scan. To our knowledge, this is done manually by an operator who decides what to crop. Historically, pan and scan has only cropped the left and right edges of the frame to use a fixed scaling. Now that pan and scan is performed digitally, operators can apply more general zooming if they want to. Historically, automation has been a minor issue as pan and scan has only been important in converting films to television which only needs to be done once and for a single target. The range of portable device sizes and aspect ratios make automatic video retargeting important.

A few authors have considered adapting videos to small devices. Fan et al.[7] and Wang et al. [30] present systems that retarget video to small devices. Both use an importance model to determine important aspects of the frame and apply cropping to remove less important content. Neither importance model accounts for global motion, nor do they consider cinematic plausibility in either their importance model or cropping rules. Because of this, these prior works are most likely inappropriate for our intended application: automatic retargeting of edited video such as feature films.

Automatic retargeting has been successful for still images. Several authors, including Chen et al. [5] and Suh et al. [27], analyze image content to determine important aspects and crop away less important image regions so that the important regions appear at a larger scale. While work on automatic still image retargeting inspired and informs our approach, the video retargeting problem is more challenging for a number of reasons including: determining the important aspects of a video is difficult (§3), what is important changes dynamically, and video retargeting has the opportunity to introduce virtual camera motions.

The image retargeting approach of Santella et al.[25] relies on eye-tracking to determine the important region. It introduces a number of heuristics for preserving image composition in addition to information content, similar to our efforts to preserve aspects of the video. We must consider camera motions and other aspects of filmmaking.

An automatic retargeting system requires some "knowledge" of the art of filmmaking so that the alterations it introduces appear natural rather than as obvious artifacts of the retargeting process. While books for beginning filmmakers, such as Cantine et al.[4], provide some basic rules on filmmaking, more advanced texts (such as Katz[14] or Arijon[1]) show both the limited nature of such rules, as well as how good filmmakers may break them to achieve desired effects. This has made the art of filmmaking difficult to codify. There have been two domains where it has been discussed: creating 3D camera movements (including cuts) for cinematography of virtual worlds (such as He et al.[9] and Bares et al.[2]) and automatic video editing.

Systems for automatic video editing have been presented in a few focussed domains. For personal video collections, the focus is on selecting clips from a library Girgensohn et al.[8]. Automatic editing of classroom videos is a similar problem to video retargeting as it transforms a video stream (potentially from multiple cameras) into a video of the same length by selecting parts of them. Successful systems, such as Rui et al.[24], Bianchi[3] and Heck et al.[10], all use heuristics to determine what is most important to show, and then crop away less important aspects (either by zooming a camera in real time or by cropping a pre-recorded frame). Unfortunately, the methods used in classroom automatic video editing systems do not generalize easily as they rely on the known structure of classroom lectures and the unedited source material. Our automatic retargeting work applies to a wide range of video, including edited source material such as films and television shows.

An alternative to cropping and scaling for retargeting is to apply non-linear distortions. This has been considered for still images in Liu et al.[15] and Setlur et al.[26]. Applying such approaches to video is challenging and may introduce odd visual effects.

3. IMPORTANCE

The basic idea of video retargeting is to preserve what is important in video by removing what is not. To automate retargeting, we must have some method of identifying what is important in the video. Our approach is to define penalties for various forms of important information loss and to find retargeting solutions that minimize these penalties. This approach provides a flexible framework: we can easily add new types of penalties as we devise better mechanisms for measuring the important content lost in a video.

Automatically determining what is important in a video is difficult and subjective. To truly determine what is most important to show on screen requires an understanding of

the story of a film: a seemingly unimportant element might be critical later on. To make matters worse, different viewers may have different opinions on what is most important.

Automatic image retargeting must also determine what is important in an image without semantic understanding. Recent results in still image retargeting [5, 15, 26, 27] suggest two mechanisms work well: specifically recognizable objects (such as faces) are usually important, and regions of the image that are most likely to attract low-level visual system are likely to be important.

Determining importance in video is more difficult than determining importance in its constituent still images for two reasons: first, there is motion in the frame that carries meaning; and second, the image is part of a larger context of the overall video[2]. In our current work, we consider the first issue, but not the latter.

The heuristics we use for video retargeting without semantic understanding fall into two categories that extend what has been applied to still image retargeting. First, we consider the temporally local information of a given frame, that is the content of the image and its motion. If a part of the frame is likely to attract a viewer's attention, we consider it important to preserve. The methods from still images, salience and object identification, are extended with motion features in Section 3.1. Second, we consider more general types of loss due to retargeting. Retargeting should avoid things that always lose information, or create results that a filmmaker would be unlikely to do (or would only do if they wanted to call attention to something). In Section 3.2, we describe the specific heuristics used in our prototype. Our extensible framework allows for new penalties to be added as we develop a richer set of importance determining mechanisms.

The user can add hints to inform the system what is important. This is valuable as they may have semantic or contextual information that is unavailable to a purely automatic system, and their effort can be amortized over a set of retargeting targets. None of the examples in this paper employ hints.

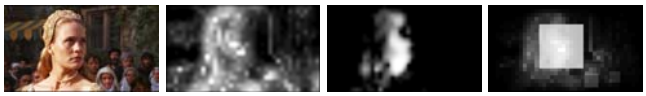
3.1 Local Importance

Our first heuristic considers each frame without regard to its context. For each image, we compute an *importance map* that for each pixel measures how important that pixel is likely to be. Our approach is similar to what is used in still image retargeting: we combine image salience and specific object detection. For video, we must extend the salience model to account for motion: certain types of motion are very likely to draw a viewer's attention.

The importance map is temporally local: it considers the content of the frame and a bounded number of neighboring frames (for motion estimation). It is created as a weighted linear combination of three components: the image saliency (S_I), motion saliency (S_M) and detected object saliency (S_F). The first two components are discussed in subsequent sections. An example of an importance map is shown in Figure 2.

In our current implementation, the only detected objects are faces. Faces are almost always important, are quickly recognized by viewers, and usually attract attention. They are also easily detected automatically. Our implementation

[2]Including the audio, consider a narration that says "look at the left side of the image."



(a) (b) (c) (d)

Figure 2: Salience map for a frame. (a) original frame, (b) image salience S_I without center weighting (c) motion salience S_M (d) resulting map $S = S_I + S_M + S_F$. Note the center weighting and that the small faces are weighted less than the large one. Not all small faces are detected. Source video © 2001 MGM.

uses the method of [29]. The object salience map S_F has value zero for pixels that are not part of a face, and a value proportional to the area of the face if it is part of one. This implements the heuristic that larger faces are likely to be more important.

The importance map is center weighted to add the heuristic that things in the center of the image are more likely to be important. Therefore, the importance map S is computed as

$$S = \omega_I S_I + \omega_M S_M + \omega_F S_F$$
$$\omega_I + \omega_M + \omega_F = 1$$

where the ω are the weights for each salience component.

3.1.1 Image Salience

Research in vision science, such as [19, 20], indicates that saliency can be measured by low-level feature contrast. Based on this, several methods have been proposed to detect saliency from an image. Itti et al. [13, 12] detect local spatial feature discontinuities in a static image pyramid with a fixed number of scales, as feature contrast maps, and combine them into a single saliency map. Ma et al. [18] provide a simpler saliency metric based on heuristics observation how people perceive images. Hu et al. [11], transform image features into a 2D space through a polar transformation, and identify regions by estimating the subspaces. They consider both the region feature contrast and its geometric properties to determine the saliency. Our implementation uses an implementation of Ma's contrast based method [18] that weights the center of the image more as described in [15]. Basically, the method applies a band-pass filter in a perceptually uniform color space.

3.1.2 Motion Salience

Moving objects usually attract attention [23]. Therefore, given the observed motion (the optical flow that assigns a motion vector to each pixel) with an image, we would expect that regions with motion are likely to be salient. However, the simple approach of only considering the amount of motion at each pixel, such as in [17], is insufficient. People are good at factoring out global motion, such as that induced by head or camera movements, therefore motion *contrast*, not magnitude, is a better predictor of salience [23].

For video retargeting, we use a new motion salience scheme that is based on motion contrast. Given the optical flow associated with an image, we first use dominant motion estimation to segment the image into a global background[3] and foreground regions. The salience for each pixel is then

[3]The dominant motion may not *semantically* be the background.

proportional to the magnitude difference between its motion and the background motion. For pixels that are in the background, there will be no difference. To reduce noise from the fitting of a global motion model, we explicitly set these values to zero. So given the optical flow M_V and the global motion model M_G, both of which provide 2D vectors for each pixel, we compute the motion salience S_M at foreground pixel i, j as:

$$S_M(i,j) = \frac{\|M_V(i,j) - M_G(i,j)\|}{dMV_{max}}$$

where dMV_{max} is the maximal difference between a motion vector and the global motion vector.

Our implementation uses the Lucas and Kanade method [16] to calculate optical flow in each frame. We model the dominant global motion using a restricted affine model with four parameters ($zoom$, $rotate$, pan, $tilt$)

$$M_G(x,y) = \begin{bmatrix} zoom & rotate \\ -rotate & zoom \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} pan \\ tilt \end{bmatrix}.$$

In practice, the model seems to fit well enough for most camera motions observed in regular videos and is easy to estimate. Because we are not performing registration, more detailed models are not necessary.

Our implementation uses the iterative method of [21] to simultaneously perform segmentation and parameter estimation. A variant of the expectation maximization (EM) algorithm, the method alternates estimating the assignment of pixels into foreground and background and determining the parameters of the global motion of the background model. The model is initialized by assigning all pixels to the background. The following steps are then alternated until the model sufficiently fits the motion of the assigned pixels, or too few pixels are left in the background:

1. Estimate the background motion parameters given the current set of pixels assigned to the background. To fit the model to the motion vectors, a linear least squares problem is formed and solved using singular value decomposition (SVD).

2. For each pixel (i, j), compare its motion $M_V(i, j)$ to the motion predicted by the global model $M_G(i, j)$. If the magnitude of the difference is small, assign it to the background set, otherwise assign it to the foreground set.

Because of errors in motion estimation, segmentation, and background model fitting, the computed motion salience map S_M is often noisy. Band-pass filtering to reduce noise seems appropriate, as a pixel's salience should be similar to pixels on the same object since objects move coherently. However, a simple filter disregards the fact that neighboring pixels may be on different objects so filtering reduces contrast between groups. Therefore, we apply a Bilateral Filter [28] that takes into account similarity in value as well as proximity. We apply a Bilater Filter in space and time, to the motion salience map. Similar to a 3D Gaussian filter, the Bilateral Filter also takes the weighted average of each pixels' neighbors (in space and time) but compute the weights as a Gaussian function of distance in space, time, *and* value.

Figure 3: Cut faces appear unnatural (left), so our system adds a penalty to avoid them. This often causes the faces to appear at the edge of the frame (center) so the system penalizes this as well, leading to more natural framings (right). Source video © 2003 20th Century Fox.

3.2 Penalties for Information Loss

Given a source video of dimensions $w_{source} \times h_{source}$ and a target video dimension $w_{targ} \times h_{targ}$, retargeting chooses a subwindow of the source $\mathcal{W}$ that is subsequently scaled to fit the target. For convenience, rather than parameterizing $\mathcal{W}$ by the corner position (we place the origin at the upper left), width and height, we choose to parameterize it by corner position, scale factor s, and anisotropy of scale s_a

$$\mathcal{W} = (x, y, w, h) = (x, y, w_{targ} * s, h_{targ} * s * s_a).$$

For a unity scale factor ($s = s_a = 1$), the source pixels are copied directly to the target. We do not consider enlargement ($s < 1$).

Given a window, we can compute a penalty value for the amount of "information" that is destroyed on a particular frame. The total penalty is computed as a weighted sum of a number of individual penalties. Each penalty p_i has an associated weight ω_i. At present, our system considers the following penalties:

Information cropped: the sum of the importance of the pixels of the source video that do not appear in the resulting image

$$p_x(\mathcal{W}) = \frac{\sum_{(x,y)\notin\mathcal{W}} S(x,y)}{S_{total}},$$

where $S(x, y)$ is the importance value at pixel (x, y) and S_{total} is the total importance in the shot.

Scaling: as the scale factor increases, information is lost due to downsampling. We penalize larger scale factors:

$$p_s(\mathcal{W}) = |s - 1.0|^3,$$

where s is the scale factor. We choose a cubic cost function to avoid drastic downsampling.

Pixel Aspect Ratio: while the system can squeeze the image with an anisotropic scale, such distortions look odd and are penalized:

$$p_a(\mathcal{W}) = |s_a - 1.0|^3,$$

where s_a is the aspect ratio change factor. We choose a cubic cost function again to avoid drastic distortion.

Face cut cost: cutting off part of a face is unattractive, and generally not done by filmmakers [1]. If the window intersects a located face, we create an image that is clearly an artifact of the retargeting process losing cinematic plausibility (Figure 3). For each detected face, we penalize the system with a constant cost if part of the face is cut outside the window.

Edge crowding cost: by itself, the face cut penalty tends to cause faces to appear adjacent to the end of the window which appears unnatural (Figure 3). We penalize the system for putting a face too close to the edge of the frame:

$$p_e(\mathcal{W}) = \begin{cases} 1.0 - d/d_{max}, & d \leq d_{max}; \\ 0, & \text{else.} \end{cases},$$

where d is the smallest distance between the face edge and the window edge, and d_{max} is a constant, set empirically in our system to 15 pixels.

Pan and Cut costs: motions of the window between frames can cause the loss of motion information, so they are penalized. This is discussed in more detail in Sections 4.2 and 4.3.

User hint costs: if the user specifies that a particular position in the frame is important, a window is penalized if it does not contain the point. Note: none of the examples in this paper include user specified hints.

There is a weight ω_i for each penalty and for each salience component. This set of weights serves to tune the importance determination process to balance all of the factors. In principle, these weights could be tuned to account for viewer preferences. In practice, we have determined a set of weights empirically and used them for all of our experiments.

One feature of our approach is that the set of penalties and salience terms is extensible. New ones can be added easily, potentially improving the quality of the results.

4. OPTIMIZING RETARGETING

The video retargeting problem is to choose a cropping window (x, y, s, s_a) for each video frame. As described, the optimal retargeting would be to chose the window such that the penalty is minimized. However, performing this optimization on each frame independently may cause the cropping window to move around, inducing motion that appears like camera motion. Therefore, we must limit the motions introduced by retargeting. Induced motions must be *cinematically plausible:* they must not be something that a filmmaker would be unlikely to use, as this would yield results that would appear as obvious artifacts. We also should consider the preservation of the motion in the original video as it is part of the video's content that should be preserved.

Our approach considers each shot (a duration of the video taken from a continuous viewpoint) independently. Between shots, there is already a discontinuity, so we need not worry about providing continuity across shot boundaries (ramifications of this are considered in Section 6). Our implementation automatically breaks a long video into a sequence of shots using the algorithm of [22]. This simple algorithm is efficient and robust against object and camera motions. First, a color histogram is computed for the consecutive frames. Color is quantized to improve performance. A shot boundary is detected whenever the histogram intersection between two neighboring frames is below a threshold.

To ensure cinematic plausibility, video retargeting considers each shot as a unit. Rather than solving a large constrained-optimization problem to determine the window for each frame in the shot, we restrict the motion to be one of three shot types that are most common in film: a crop (the window remains constant over the shot); a pan (the window moves smoothly over the duration of the shot); and a cut (a discontinuity is introduced and the shot is divided into two independent shots). Each shot type has a small number of parameters (for example, a crop's parameters are (x, y, s, s_a)). These shot types are detailed in subsequent sections.

For each shot, video retargeting chooses the cropping window for each frame such that the sum of the penalties for all of the frames is minimized, subject to the constraint that the motion of the cropping window is one of the three shot types. To implement this constrained optimization for a given shot, for each of the three shot types we determine the parameter values that give the lowest penalty. Then, from these three options, we select the one that has the lowest overall penalty. For each shot type, we use a different algorithm to find the parameter values that minimize the penalty.

4.1 Cropping

The simplest method for performing retargeting is to select a single cropping window for each shot. By not having the cropping window move within the shot, retargeting does not introduce any visual motion. Motion in the source video, both object and camera movements, are preserved.

Choosing the optimal cropping for a shot requires minimizing the penalties across the entire shot. The set of potential cropping window forms a 4-dimensional space to search, including the position of the lower left corner of the rectangle, (x, y), the scale factor, s, and the anisotropy, s_a (a scaling factor on the width that allows the aspect ratio of the cropping window to vary from the aspect ratio of the target). In performing cropping, we minimize the weighted sum of the penalties (§3.2). Because of the discrete and nonlinear nature of this objective, we have chosen to optimize it by a brute-force search through the parameter space. Specifically, we step the cropping window position (x, y) pixel by pixel. We iterate the anisotropy factor s_a from 0.8 to 1.2 with step size 0.1, and the scale factor s from 1.0 multiplying by 1.1 until the cropping window outgrows the source frame.

To efficiently compute the information lost cost for all of the different windows, we first sum the importance over each shot to create a single map. For each window, the information lost can be computed in constant time using a summed area table [6].

4.2 Virtual Pans

Introducing camera motion must be done carefully to avoid introducing artifacts or changing the existing cinematography. For example, we avoid introducing zooms (changing the scale s and s_a in a shot) as they have strong effects on the viewer. In practice, we limit ourselves to the most common camera motion: a horizontal pan. Real cameras create pans by turning on their tripod. For us, a horizontal pan means that the horizontal position of the window x changes over time. We restrict each shot to contain a single pan to avoid a "ping-pong" effect. Examples of pans are shown in Figure 4 and Figure 8.

Real cameras have mass and do not accelerate instantly. Rapid accelerations are so noticeable and disconcering to viewers that real tripods for film and video are almost always damped to prevent this. Therefore instant accelerations of the window would be an obvious artifact of retargeting and

(a) Static Cropping

(b) Virtual Pan

Figure 4: Three frames from a shot where our system uses a pan to better show moving objects. With a static shot (a), the system must either show the characters at a small scale, or crop them. With a pan (b) the system can show the characters across the entire shot. Source video © 2004 Warner Home Video.

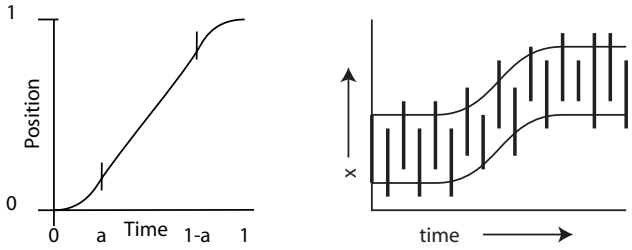


Figure 5: Left: the interpolation function has zero initial and final derivatives, constant acceleration and deceleration, and constant velocity in the middle. Right: the optimal window is determined for each frame and a smooth curve is fit.

must be avoided. To do this, we define the horizontal position of the window to be a function of an appropriate form. The function depends on 4 parameters: x_1 the initial value, x_2 the ending value, t_1 the time that the motion begins, and t_2 the time that the motion ends. Specifically, the camera does not move until t_1, then accelerates, holds a constant velocity, then decelerates until t_2, when it holds constant

$$x(t) = \begin{cases} x_1 & t < t_1 \\ (1 - d(\frac{t-t_1}{t_2-t_1}))x_1 + d(\frac{t-t_1}{t_2-t_1}))x_2 & t_1 \leq t \leq t_2 \\ x_2 & t > t_2 \end{cases} .$$

Where $d(u)$ is the interpolation function (depicted in Figure 5)

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

where a is a constant. Note that for a given t, $x(t)$ is linear in all parameters except t_1 and t_2.

To find the optimal horizontal pan, we need to search the 7 parameter space $(x_1, x_2, y, s, a, t_1, t_2)$ to find the values that minimize the penalty over the shot. The brute force search used in the previous section is intractable. Instead, we approximate the ideal solution by first finding the optimal window on each frame individually (using the methods of the previous section) and then fitting the interpolation curve to this data, as schematized in Figure 5.

Given the optimal window for each frame, we perform fitting by creating constraints that the top, left, bottom and right edges of the window on each frame are equal to their ideal counterparts. Given the optimal window for each frame, we perform the fitting with the goal that the top, left, bottom and right edges of the window on each frame are close to their ideal counterparts as possible. We fit the curve in a least squares sense. For a given t_1 and t_2, this is a linear least squares problem and can be solved analytically. We exhaustively search the valid range of t_1 and t_2, finding the best values of the other 5 parameters for each, and choose the one with the lowest residual.

Once the parameters of the optimal panning motion are computed, the penalty for the retargeted shot using this movement can be computed by evaluating the penalty terms (§3.2) and adding an addition "pan" penalty that discourages the addition of camera motion

$$p_p = w_p * (t_2 - t_1),$$

where w_p is a pan penalty coefficient.

4.3 Virtual Cuts

Retargeting can introduce a cut to break a shot into two. Such cuts must be introduced carefully: discontinuous camera motions can disorient the viewer if they are not done correctly, and even good cuts effect the pacing of the video.

In principle, we could treat each new sub-shot as a shot and perform the entire retargeting optimization recursively, potentially dividing each sub-shot further. In practice, we restrict ourselves to performing a single cut on each shot, since finer divisions would have too drastic an impact on the pacing of the result. We also restrict each subshot to be retargeted with a crop (the window does not move over the subshot). This facilitates determining if the cut is good as we need not worry about motion matching issues.

In introducing a cut we must avoid creating a cut that stands out as obtrusive to the viewer. We restrict introduced cuts such that the resulting sub-shots are of sufficient length (63 frames) because shots shorter than this have a noticeable effect on pacing and are only used very intentionally by filmmakers. Secondly, we avoid making a "jump cut," the phenomena where a cut is very jarring to a viewer if the difference between the two subshots it divides is not sufficiently different [4].

A cut applied to a shot has nine parameters: the four parameters of the windows of each subshot, and the time of the cut t_c. To find the optimal retargeting we use an approximation that incorporates some of the constraints that avoid a jump cut. We note that to be sufficiently different, one of the subshots must come from the left part of the frame, while the other must come from the right. We then use the following three steps:

1. For each frame, we compute a penalty for the best window for the left and right sides, $L(t)$ and $R(t)$.

2. We determine t_c and which side is first by exhaustively trying all reasonable values of t_c and both orderings to see which leads to the minimum. For example, the

Figure 6: A shot where our system inserts a cut. Top: DVD frame. Bottom: 2 frames from each of the sub-shots. The horse (left) only moves (and therefore becomes salient) in the later part of the shot. Source video © 2004 Warner Home Video.

left-first penalty for a given t_c is

$$P(t_c, left) = \sum_{i=1}^{t_c} L(i) + \sum_{i=t_c}^{n} R(i).$$

3. We then use the method of Section 4.1 on each subshot.

An example where our system introduces a cut is shown in Figure 6.

Introducing a cut disturbs the continuity of the original shot. Also, our simple method of determining the cut does not account for other preconditions for quality cuts such as cutting on action [1, 4, 14]. Therefore, we add an additional penalty p_c to the penalty computed from the optimal cut shot. To further discourage jump cuts, the penalty is proportional to the image similarity between the two subshots as measured by color histogram intersection [22]. In computing the total penalty for applying a cut to a shot, we discount the information cropped penalties for the two subshots to account for the fact some of the cropped information will appear in the other subshot.

5. EXAMPLES

We have implemented the video retargeting methods described in the previous sections on Windows-based PCs. Analysis of the video to determine the local importance is time consuming as we use inefficient open source implementations of face finding and optical flow computation. We perform it as an offline pre-process. Once the importance information is pre-computed, retargeting requires roughly 3-5 times real time with the unoptimized prototype.

We have used our system to retarget various content from DVDs to a variety of sizes and aspect ratios. For all of our tests, the source comes from "widescreen" DVDs that letterbox the original theatrical presentation. The aspect ratios of the source material range from 1.85:1 to 2.4:1 (these DVDs appear letterboxed even on 16:9 TVs). The systems performs as we would expect: choosing static crops, pans and cut insertion where appropriate.

target size	crops	pans	cuts	failures
120×120	141	66	7	22
160×120	147	63	4	12
200×150	147	67	0	9

Table 1: Shot choices and failure rates on the test set.

Despite its simple importance models and heuristics, our system often chooses good retargeting solutions. In films, many shots have a clear focus of attention (as directors want to facilitate an audience's comprehension), and our system can appropriately choose tighter croppings. In other cases, such as an action scene, importance information is spread over more of the screen, and our system shows more of the frame.

On some shots, our system fails to produce a retargeting solution that conveys the original intent of the shot. On some of these failures, the system cannot be blamed: the source material might be such that there is no way to preserve its content in a smaller size. Even a manual pan and scan operator would need to make a difficult decision to discard something important. Other failures can be directly traced to system components, e.g. the face detector failing to identify a poorly lit face, or our heuristics being insufficient.

To assess our system's results, we selected 3 movies that were known to be difficult for fullscreen presentation based on web critiques of the full screen versions. For efficiency, and to test the system's ability to handle different source size, we first downsampled the movies to various sizes. From these films we used 214 shots for our tests, and retargeted each to three different targets: two different sizes of 4:3 displays (200×150 pixels and 160×120 pixels), and a square display like a PDA (120×120 pixels). These target sizes were chosen because their relatioship to the source material is similar to the relationship of current devices to the most common source material.

The shot choices made by our system are listed in Table 1. Our system chooses static crops the majority of the time. As the target grows larger, the frequency that cuts are used decreases. For all of the target sizes, the average anisotropy used was 1.1, or a 10 percent "squish."

Result quality was assessed subjectively by the authors. We consider a retargeting solution for a shot to be a failure if we prefer a naïve scaling to that solution. On the test set, our system failed less than 10% of the time. Failures have two forms: cropping of an important part of the image, and introduction of a cinematically implausible artifact. Because the baseline for comparison is a maximal scale, we cannot fail by scaling too much.

Camera motions are the largest cause of failures due to cinematic implausibility. Despite our efforts to prevent jump cuts, almost one third of the cuts made by our system were distracting, suggesting that we must be even more conservative about applying cuts. In rare cases pans create cinematic implausibility by changing the direction of the apparent motion of a moving object. The most common form of failure, however, is when the system removes an important element of the image. This happens when either the system fails to identify something as important, or incorrectly decides to focus on some other part of the image.

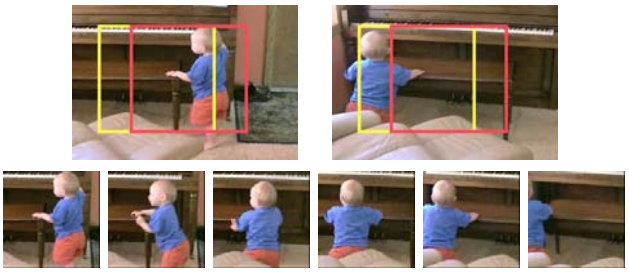


Figure 7: A clip from a digital camera is retargeted. Despite no identifyable faces and jerky handheld motion, the system still correctly applies a pan. Top: range of pan shown on source frames. Bottom: frames from retargeted result. Source video © 2006 M. Gleicher.

5.1 Home Videos

The approach described in this paper addresses the retargeting of edited videos such as films and television. Our approach makes some assumptions that the video is "good." For example, we assume that the video is broken into shots and that within each shot the camera work is intentional and should be preserved. Amateur "home" videos often fail to meet these criteria. Digital still cameras and video camera cell phones usually take short clips of video. Retargeting these clips is important as they are often timely and so users want to transmit them. Retagreting not only makes the resulting videos more appropriate for portable displays, it can also make them smaller.

Preliminary experiments show that our retargeting system can be applied to short clips from small digital cameras. Despite the jerky, hand-held camera movements these clips tend to exhibit, the importance finding algorithm provides reasonable results. An example of a retargeted clip where a virtual pan is used is shown in Figure 7.

We expect that a real solution for amateur video clip retargeting will require new methods that do not make assumptions about the quality of the source cinematography. However, the importance model from this paper should still be applicable, as will the general approach of seeking to create cinematically plausible virtual camera work.

6. DISCUSSION

Video retargeting is fundamentally limited: we necessarily must throw away information when presenting a video on a smaller device. Our ability to do this well is fundamentally limited by the fact that what is important to preserve depends not only on the low-level visual information that *may* be available to an automatic system, but also high level aspects of the underlying story.

At present, our system relies on only basic, easy to obtain information about the video's content. We have chosen to use only salience, face detection, and dominant motion direction as this information can be obtained with simple, robust algorithms. In a sense, our prototype is an experiment of how well we can do video retargeting based on this limited information.

Incorporating more information poses a number of challenges. First, we need a method of obtaining the new information; second, we need mechanisms for evaluating how much information is lost in a retargeted presentation; and third, we need mechanisms for incorporating these new metrics into our optimization. For example, at present, our system identifies faces in the image, but does not distinguish between them. Computer vision techniques could find facial motion to identify which face is speaking, allowing the heuristic that the speaker is more important. Aside from the obvious caveat that the heuristic is imperfect (sometimes another character's reaction is more important), this new type of information suggests not only new penalty terms, but also new optimization methods that will encourage "cut on speaker change."

There is one category of information that we feel is particularly important for improving the quality of retargeting: coverage. At present, our approach treats each salient pixel and face as independent. There is no notion of something being the same across frames. If our system were to identify when important image features are common across multiple frames, we could introduce metrics that measure the *coverage* of a retargeting. That to give importance that objects in a shot are seen at *some* time. For example, in a static scene, the system might introduce a virtual pan so that the entire frame is seen more closely.

Presently, our system treats shots independently. Performing analysis across multiple shots may be useful for a number of reasons. For example, information given in one shot may be less important to repeat in a later one; or since a filmmaker may use a repeated shot composition to convey continuity or repetition or allow the viewer to witness change, cropping these similar shots differently breaks this rhythm. Therefore, performing video retargeting "globally" over all shots poses a number of technical challenges to scale the optimization process as well as a development of schemes for using the global information.

Better evaluation is necessary for assessing video retargeting. Assessing the quality of the results of our system is difficult. Displaying a video at a small size necessarily discards information, and the choice in how to do this is subjective. Manual pan-and-scan is sufficiently criticized that letterbox presentation is preferred by most movie fans, although we feel that the tradeoffs make aggressive retargeting using cropping important for small devices.

7. ACKNOWLEDGMENTS

This work was supported in part by NSF grants IIS-0097456 and IIS-0416284. Figures 1, 4, 6, and 8 are from *Harry Potter and the Prisoner of Azkaban* © 2004 Warner Home Video. Figure 3 is from *Ever After* © 2003 20th Century Fox. Figure 2 is from *The Princess Bride* © 2001 MGM.

References

[1] D. Arijon. *Grammar of the Film Language*. Silman-James Press; Reprint edition, 1991.

[2] W. Bares and J. Lester. Intelligent multi-shot visualization interfaces for dynamic 3d worlds. In *Proceedings of the 1999 International Conference on Intelligent User Interfaces*, pages 119–126, 1999.

[3] M. Bianchi. Automatic video production of lectures using an intelligent and aware environment. In *MUM '04: Proceedings of the 3rd international conference on Mobile and ubiquitous multimedia*, pages 117–123, New York, NY, USA, 2004. ACM Press.

(a) First and last frames of DVD shot.

(b) Frames from retargeted shot 160×120.

Figure 8: A virtual pan applied. The original shot is a complex camera motion with different characters moving at different times. Our motion salience method (§3.1.2) can distinguish object and camera motion. The system uses a virtual pan (§4.2) to better include the active characters. The range of the cropping window is indicated on the source frames. Source video © 2004 Warner Home Video.

[4] J. Cantine, B. Lewis, and S. Howard. *Shot by Shot; A Practical Guide to Filmmaking*. Pittsburgh Filmmakers, 1995.

[5] L.-Q. Chen, X. Xie, X. Fan, W.-Y. Ma, H.-J. Zhang, and H.-Q. Zhou. A visual attention model for adapting images on small displays. *ACM Multimedia Systems Journal*, 9(4):353–364, November 2003.

[6] F. C. Crow. Summed-area tables for texture mapping. In *Proc. of SIGGRAPH 84*, pages 207–212, 1984.

[7] X. Fan, X. Xie, H.-Q. Zhou, and W.-Y. Ma. Looking into video frames on small displays. In *Proc. of ACM Multimedia*, pages 247–250, 2003.

[8] A. Girgensohn, J. Boreczky, P. Chiu, J. Doherty, J. Foote, G. Golovchinsky, S. Uchihashi, and L. Wilcox. A semi-automatic approach to home video editing. In *Proceedings of UIST*, pages 81–89, 2000.

[9] L. He, M. Cohen, and D. Salesin. The virtual cinematographer: A paradigm for automatic real-time camera control and directing. *Proceedings of SIGGRAPH 96*, pages 217–224, August 1996.

[10] R. Heck, M. Wallick, and M. Gleicher. Virtual videography. *ACM Transactions on Multimedia Computing, Communications and Applications*, 3(1), 2007. to appear.

[11] Y. Hu, D. Rajan, and L.-T. Chia. Robust subspace analysis for detecting visual attention regions in images. In *Proc. of ACM Multimedia*, pages 716–724, 2005.

[12] L. Itti and C. Koch. Computational modeling of visual attention. *Nature Reviews Neuroscience*, 2(3):194–203, March 2001.

[13] L. Itti, C. Koch, and E. Niebur. A model of saliency-based visual attention for rapid scene analysis. *IEEE Trans. Pattern Anal. Mach. Intell.*, 20(11):1254–1259, November 1998.

[14] S. Katz. *Film Directing: Shot by Shot : Visualizing from Concept to Screen*. Michael Wiese Productions, 1991.

[15] F. Liu and M. Gleicher. Automatic image retargeting with fisheye-view warping. In *Proc. of the 18th annual ACM symposium on User interface software and technology*, pages 153–162, 2005.

[16] B. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proc. of International Joint Conference on Artificial Intelligence*, pages 674–679, 1981.

[17] Y.-F. Ma and H.-J. Zhang. A model of motion attention for video skimming. In *Proc. IEEE ICIP*, pages 129–132, 2002.

[18] Y.-F. Ma and H.-J. Zhang. Contrast-based image attention analysis by using fuzzy growing. In *Proc. of ACM Multimedia*, pages 374–381, 2003.

[19] H. Nothdurft. Salience from feature contrast: additivity across dimensions. *Vision Research*, 40(11-12):1183–1201, 2000.

[20] H. Nothdurft. Salience from feature contrast: variations with texture density. *Vision Research*, 40(23):3181–3200, 2000.

[21] W. R. and H. T. Fast camera motion analysis in mpeg domain. In *Proc. of IEEE ICIP*, pages 691–694, 1999.

[22] M. Rasheed, Z.and Shah. Scene detection in hollywood movies and tv shows. In *Proc. of IEEE CVPR*, pages 343–348, 2003.

[23] R. Rosenholtz. A simple saliency model predicts a number of motion popout phenomena. *Vision Research*, 39(19):3157–3163, 1999.

[24] Y. Rui, L. He, A. Gupta, and Q. Liu. Building an intelligent camera management system. In *Proceedings of ACM Multimedia Conference*, 2001.

[25] A. Santella, M. Agrawala, D. DeCarlo, D. Salesin, and M. Cohen. Gaze-based interaction for semi-automatic photo cropping. In *Proc. of CHI*, 2006.

[26] V. Setlur, S. Takagi, R. Raskar, M. Gleicher, and B. Gooch. Automatic image retargeting. In *Proc. of International Conference on Mobile and Ubiquitous Multimedia*, 2005.

[27] B. Suh, H. Ling, B. B. Bederson, and D. W. Jacobs. Automatic thumbnail cropping and its effectiveness. In *Proc. of the 16th annual ACM symposium on User interface software and technology*, pages 95–104, 2003.

[28] C. Tomasi and R.Manduchi. Bilateral filtering for gray and color images. In *Proc. of IEEE ICCV 98*, pages 839 – 846, 1998.

[29] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Proc. of IEEE CVPR*, pages 511–518, 2001.

[30] J. Wang, M. Reinders, R. Lagendijk, J. Lindenberg, and M. Kankanhalli. Video content presentation on tiny devices. In *Proc. of IEEE Conference on Multimedia and Expo*, pages 1711–1714, 2004.