Open access • Proceedings Article • DOI:10.1109/CVPR.2014.429

Time-Mapping Using Space-Time Saliency — Source link

Feng Zhou, Sing Bing Kang, Michael F. Cohen

Institutions: Carnegie Mellon University, Microsoft

Published on: 23 Jun 2014 - Computer Vision and Pattern Recognition

Topics: Rendering (computer graphics), Salience (neuroscience) and High dynamic range

Time-Mapping Using Space-Time Saliency

Feng Zhou
Carnegie Mellon University
http://www.f-zhou.com

Sing Bing Kang
Michael F. Cohen
Microsoft Research
{sbkang, mcohen}@microsoft.com

Abstract

We describe a new approach for generating regular-speed, low-frame-rate (LFR) video from a high-frame-rate (HFR) input while preserving the important moments in the original. We call this time-mapping*, a time-based analogy to high dynamic range to low dynamic range spatial tone-mapping. Our approach makes these contributions: (1) a robust space-time saliency method for evaluating visual importance, (2) a re-timing technique to temporally resample based on frame importance, and (3) temporal filters to enhance the rendering of salient motion. Results of our space-time saliency method on a benchmark dataset show it is state-of-the-art. In addition, the benefits of our approach to HFR-to-LFR time-mapping over more direct methods are demonstrated in a user study.*

1. Introduction

High-frame-rate (HFR) cameras such as the FastCam[1] and Phantom[2] are capable of capture rates of $1,000$ fps or higher at HD video resolution. These cameras are typically used to create slow-motion playback by replaying the captured frames at a much slower rate. These are expensive specialized cameras intended for visualization and analysis of high-speed events such as automotive crash tests and explosions. There are now consumer-grade cameras such as the GoPro[3] and iPhone 5S[4] which are capable of 120 fps video capture at 720p. In addition, there is a Kickstarter project called "edgertronic[5]" where the goal is a relatively inexpensive high-speed camera capable of $18,000$ fps.

Although the obvious purpose for HFR cameras is to produce slow-motion video, we address a different question. Given HFR video, can one produce a superior regular-speed low-frame-rate (LFR) video, than is possible from normal LFR video input? By regular-speed, we refer to maintaining the overall pacing of the original input, *i.e.*, a 10 second event should still playback in 10 seconds, although short intervals may deviate from this constraint. This is analogous to the conversion from high-dynamic-range (HDR) imagery to the low-dynamic-range (LDR) of current monitors for display. This HDR-to-LDR conversion, often called tone-mapping has been well studied. In this paper we explore HFR-to-LFR conversion, *i.e.*, *time-mapping*, with the goal of producing superior normal speed LFR video from the richer set of input frames available from HFR capture. To our knowledge, this is the first paper to ask and attempt to answer this specific time-mapping question.

Clearly, one can (almost) reproduce low-frame-rate (30 fps) video by simply averaging over HFR blocks of frames representing each $1/30^{th}$ of a second. But can one do bet-

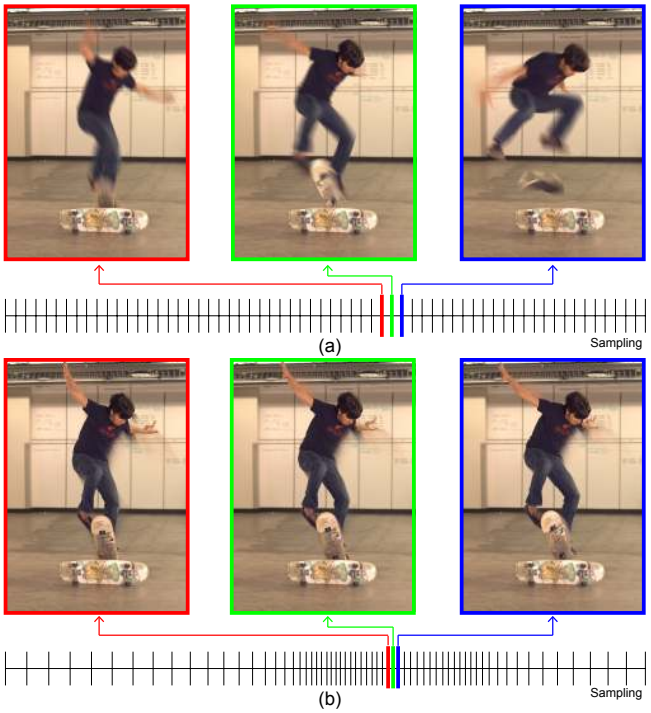


Figure 1. Example of time-mapping. (a) Uniform sampling using a box filter. Note the blur throughout each frame. (b) Sampling and filtering using our system. The sampling is denser in the middle to stretch out a little the more interesting part of the video.

[1] http://www.photron.com/
[2] http://www.visionresearch.com/
[3] http://gopro.com/
[4] http://www.apple.com/
[5] http://www.kickstarter.com

ter? For example, the rich temporal details captured by a HFR camera may be lost in this averaging step. Consider the performance capture of the skateboarding maneuver shown in Fig. 1a, where three frames are synthesized by uniformly averaging the HFR video frames over $1/30^{th}$ of a second intervals. The $1/30^{th}$ of a second intervals as well as the temporal integration miss and blur the split-second key moment. Alternatively, given three sequential HFR frames, the key moment is preserved but we now need to consider how to display these frames in such a way as to both depict this moment and also maintain the pacing of the real event.

To accomplish this goal, we devised methods for: (1) predicting what is interesting or salient in the video, (2) retiming to retain as many salient frames as possible while minimizing time distortion, and (3) temporal filtering to combine frames minimizing loss of detail while avoiding strobing. Methods for retiming and filtering make use of the results of the proposed space-time saliency measure.

As an example, shown in Fig. 1b, the output video is generated using more densely sampled frames where the faster and more interesting part of the skateboarder's performance can be better visualized at a higher temporal resolution. To enhance the motion perception of the fast moving object (*e.g.*, hand and torso), a saliency-based filter is applied to synthesize motion blur while keeping the foreground sharp.

We validate our space-time saliency approach using a benchmark dataset, and subsequently our overall approach through a user study. Results show that our approach is capable of generating visually better regular-speed video than achieved from standard LFR video. In this paper, we assume that the camera is stationary.

2. Related work

In this section, we briefly review related work in motion saliency, video re-timing, and temporal filtering.

Motion saliency. The ability to predict where a human might fixate in an image or a video is of interest in the vision community. While many models have been proposed in image domain (see [2] for a review), computing spatio-temporal saliency for videos is a relatively unexplored problem. Most existing motion saliency methods built upon image attention models by taking into account simple motion cues. For instance, Guo *et al*. [7] adopt an efficient method based on spectral analysis of the frequencies in the video. Similarly, Cui *et al*. [4] analyze the Fourier spectrum of the video along X-T and Y-T planes. Seo and Milanfar [21] propose using self-similarity in both static and space-time saliency detection. Rahtu *et al*. [17] apply a sliding window on video frames to compare the contrast between the feature distribution of consecutive windows. Recently, Rudoy *et al*. [20] model the continuity of the video by predicting the saliency map of a given frame, conditioned on the map from the previous frame. Compared to previous work, our motion saliency method combines various low-level features with region-based contrast analysis.

Video re-timing. Our method for re-timing is closely related to time-lapse video and video summarization techniques for condensing long videos into shorter ones. (See [24] for a review.) By sampling the video non-uniformly with user control, Bennett and McMillan [1] are able to generate compelling-looking time-lapse videos. In an interesting twist, Pritch *et al*. [16] present a time-lapse technique to shorten videos, while simultaneously showing multiple activities which may originally occur at different times. Kang *et al*. [10] generate a space-time video montage by analyzing both spatial and temporal information distributions in a video sequence. By sweeping an evolving time front surface, Rav-Acha *et al*. [19] manipulate the time flow of a video sequence to generate special video effects. Unlike these techniques that significantly modify the original spatio-temporal video structure, our HFR video sampler is designed to produce natural regular-speed videos.

Temporal filtering. Temporal filtering techniques have been well-studied for removing noise and aliasing artifact in video signals and for creating motion effects [14] in the context of computer animation. For instance, Fuchs *et al*. [5] investigate the effect of different temporal sampling kernels in creating enhanced movies for HFR input. Bennett and McMillan [1] experiment with various virtual shutters that synthetically extend the exposure time of time-lapse frames. By simulating motion blur in stop-motion videos, Brostow and Essa [3] are able to approximate the effect of capturing moving objects on film. Recently, temporal filtering has also been used to obtain high-resolution videos [22] from multiple low-resolution inputs.

In the related work on motion magnification, special spatial or temporal filters are used to amplify subtle motions. Wang *et al*. [27], for example, introduce the cartoon animation filter to create perceptually appealing motion exaggeration. Liu *et al*. [13] compute flow and amplify subtle motions and visualize deformations that would otherwise be invisible. On the other hand, Wu *et al*. [28] apply a linear Eulerian magnification method to reveal hidden information in video signal. Another variant of this work, a phase-based method [26] has recently been shown to achieve larger magnifications and with less noise. We focus on retaining important moments that lie within the original HFR video.

3. Space-time saliency

The key to effective time-mapping of high-frame-rate to regular-speed low-frame-rate counterparts is the ability to identify frames that are more informative. We propose a ro-

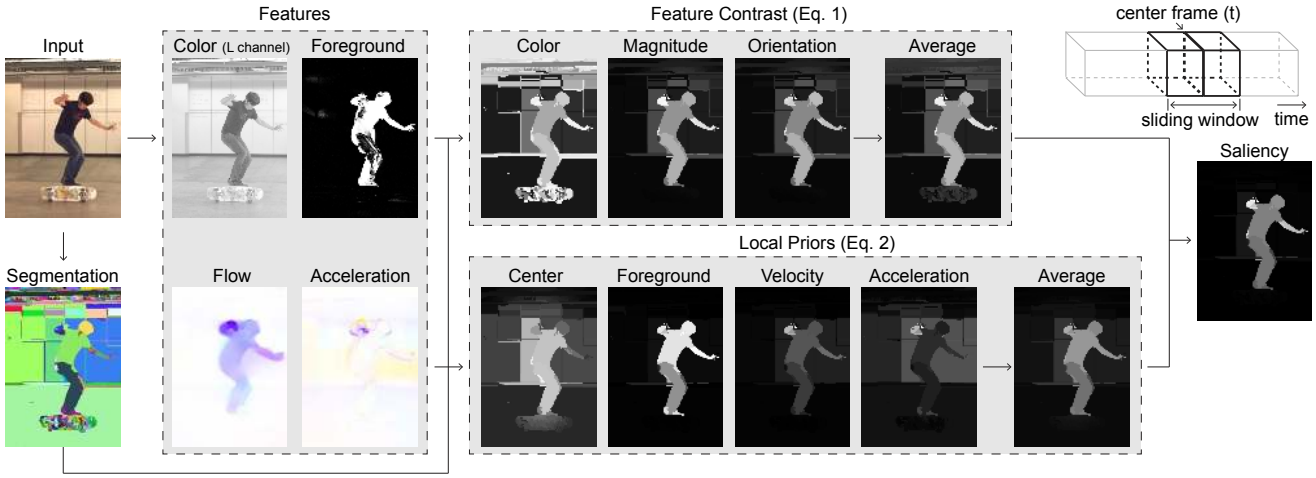


Figure 2. The pipeline for computing space-time saliency.

bust bottom-up saliency model to rate spatial-temporal regions in a video. The process of computing our space-time saliency measure is summarized in Fig. 2. First, the video is segmented into spatio-temporal regions; for each region, various appearance and motion features are extracted. The saliency measure for each region is a combination of feature contrast (Section 3.2) and local prior (Section 3.3).

3.1. Video pre-processing

Prior to estimating saliency, we over-segment the input video into color-coherent *spatio-temporal regions* (STRs). We leverage a streaming method [29] to segment video frames on-the-fly. To capture region of interests in different spatial scales, we construct a l-level pyramid of segmentation by adopting different scale parameters in the usage of [29]. (In our work, $l = 4$.) After the initial segmentation pass at each scale, we refine the result by merging adjacent STRs that are too small but have similar colors.

3.2. Feature contrast

Our space-time saliency measure is defined based on contrast in both appearance and motion features. In other words, a spatio-temporal region may be salient because its color or motion is different from its neighbors. This design decision is inspired by sensitivity of the human visual system to contrast in visual signal, and shared by the recent advance in image saliency [9, 15].

At each scale level[6], given an STR (denoted by $r_{c,t}$) within a sliding window[7] centered at frame t, we compute three feature vectors. First, we compute its color statistics in CIE Lab color space, which is perceptually uniform. The appearance of $r_{c,t}$ is computed as a color histogram, $\mathbf{x}_{c,t}^{col}$, normalized to be of unit length. The second and third features are based on the motion of the STR's pixels. Given the pixel-wise optical flow [12] between consecutive frames, the motion distribution of $r_{c,t}$ is encoded in two descriptors: $\mathbf{x}_{c,t}^{mag}$ a normalized histogram of the flow magnitude, and $\mathbf{x}_{c,t}^{ori}$ the distribution of flow orientation.

The contrast of each STR ($r_{c,t}$) is measured as the sum of its feature distances to other STRs ($r_{i,t}$), *i.e.*,

$$u_{c,t} = \sum_{f} \sum_{r_{i,t} \neq r_{c,t}} |r_{i,t}| w(r_{c,t}, r_{i,t}) \|\mathbf{x}_{c,t}^{f} - \mathbf{x}_{i,t}^{f}\|, \qquad (1)$$

where $f = \{col, mag, ori\}$ denotes one of the three features, each of which is weighted by two factors. $|\cdot|$ is the STR size and $w(r_{c,t}, r_{i,t}) = \exp(\frac{-\|\mathbf{p}_{c,t} - \mathbf{p}_{i,t}\|^2}{\sigma_s})$ measures the proximity between the center-of-mass, $\mathbf{p}_{c,t}$ and $\mathbf{p}_{i,t}$, of the STRs $r_{c,t}$ and $r_{i,t}$. Note that all these are computed over the current temporal window. As with [9], we set $\sigma_s = 0.04$ with the pixel coordinates normalized to $[0, 1]$. A larger $r_{i,t}$ closer to $r_{c,t}$ contributes more in the estimation of $u_{c,t}$.

3.3. Local prior

In addition to the contrast between an STR and its neighbors, we also compute a prior term based only on the STR's characteristics itself. There is a natural bias towards specific parts of the video, *e.g.*, the center of the frame or foreground objects. We encode these *priors* as part of saliency, to complement feature contrast. Our priors are based on location, velocity, acceleration, and foreground probability.

It has been shown that [25] humans watching a video are biased towards the center of the screen. We represent this bias by assigning a constant Gaussian falloff weight, $v_i^{cen} = \exp(-9(x_i^2 + y_i^2))$, for each pixel i with a normalized coordinate (x_i, y_i) centered at the frame origin. To encode preference for fast moving pixels and rapid changes

[6] To keep the notation clean, we remove the scale superscript $j = 1, \cdots, l$ from $r_{c,t}^j$ until the fusion step.

[7] Our STR features for each frame are computed within a sliding temporal window (200 frames) instead of over the entire sequence.

in direction, we compute for each pixel its velocity v_i^{vel} and acceleration v_i^{acc}, respectively from the optical flow. Finally, since we want to bias foreground objects with higher saliency, we compute the foreground probability v_i^{fg} of each pixel by subtracting the background from the frame. In our work, our camera is stationary; hence, the background is approximated as a per-pixel temporal median filtering.

Given the various pixel attributes, the local prior of each STR $r_{c,t}$ is computed as the average prior of the pixels i within $r_{c,t}$, *i.e.*,

$$v_{c,t} = \sum_a \frac{1}{|r_{c,t}|} \sum_{i \in r_{c,t}} v_i^a, \tag{2}$$

where $a \in \{cen, vel, acc, fg\}$ denotes one of the four pixel attributes.

3.4. Saliency fusion

All the processing described so far (to model each STR based on its appearance and motion features) is done at each level of the hierarchy. To generate a more robust space-time saliency value, we fuse over all the levels $j = 1, \cdots, l$. In our work, $l = 4$. The final saliency score $s_{i,t}$ for each pixel i at frame t is then computed by combining these responses in a linear fusion scheme, *i.e.*,

$$s_{i,t} = \sum_{j=1}^{l} u^j_{c(i),t} v^j_{c(i),t},$$

where $u^j_{c(i),t}$ and $v^j_{c(i),t}$ measure the feature contrast and local prior of the STR $r^j_{c(i),t}$ computed by Eq. 1 and Eq. 2 respectively. The subscript $c(i)$ of the two terms denotes the index of the STR containing pixel i. The superscript $j = 1, \cdots, l$ indicates the STR is generated in the j^{th} scale of the segmentation pyramid.

Compared with other methods (see Fig. 5a), our model generates saliency maps that have sharper object and motion boundaries and with less background artifacts. It provides a more robust and stable saliency measure for re-timing and filtering high-speed videos.

4. Video re-timing

Time-mapping consists of re-timing followed by rendering via filtering. In this section, we describe how we make use of saliency information for re-timing, *i.e.*, mapping input frames to output frames. The goal is to slightly slow down the action to focus on highly salient regions at the cost of slightly speeding up other portions. Thus we are constraining the overall length of the video to mimic real-time, and also trying to maintain a pacing that still feels real.

More specifically, we apply an optimal dynamic programming approach to sample representative frames based on overall saliency per frame. To reduce the effect of image noise on saliency, and hence the re-timing curve, we perform a final temporal smoothing (Section 4.2).

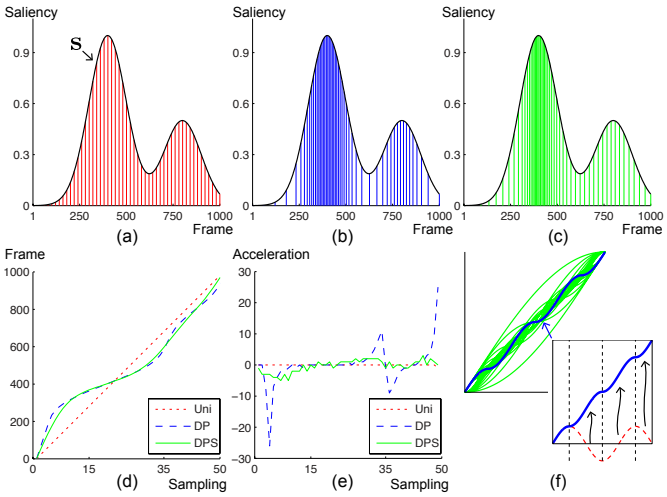


Figure 3. Re-timing a synthetic 1000-frame saliency curve to 50 frames. (a) Uniform sampling (Uni). (b) DP-based sampling. (c) Smoothed version (DPS). (d) Input-output frame mapping functions. (e) 2^{nd} derivative of the frame mapping functions. (f) 20 monotonic bases (the green curves in the upper-left) used for smoothing, where an example base (the blue curve) is generated by piece-wisely transforming the function sin (the red curve).

4.1. Saliency-based sampling

The "importance" of frame i is computed to be the average saliency score $\mathbf{s} \in \mathbb{R}^n$ over all its pixels. Note that each element of $\mathbf{s}$ is between 0 and 1. We define the frame saliency curve to be the variation of frame importance over frame number. The user can set the weight of saliency on re-timing: $s_i \leftarrow \gamma s_i + (1 - \gamma)$, with parameter $\gamma \in [0, 1]$. If $\gamma = 0$, we have uniform sampling (Fig. 3a), while $\gamma = 1$ results in total reliance on saliency for sampling. In our examples, we set γ empirically to a value of 0.5 resulting in a good balance between importance and maintaining actual speed playback.

Suppose the input and desired effective output frame rates are f_0 and f_1, respectively. This is equivalent to extracting m frames from the original n frames such that $m = \frac{f_1}{f_0} n$. We cast the problem of video re-timing as finding $\mathbf{p} \in \{1 : n\}^m$, the optimal m-frame sampling of the saliency curve $\mathbf{s} \in \mathbb{R}^n$.

As an illustration, consider the synthetic example in Fig. 3a, where we need to sample 50 frames on a 1000-frame saliency curve. The ideal $\mathbf{p}$ should sample more densely around the peaks and more sparsely at the troughs. In principle, the cumulative saliency between successive frames in $\mathbf{p}$, *i.e.*, $a_i = \sum_{j=p_i}^{p_{i+1}-1} s_j \equiv \bar{s}$, should be a constant equal to $\bar{s} = \frac{1}{m-1} \sum_{i=1}^{n} s_i$. In practice, this strict criterion can be relaxed to minimize the following sum of least-square errors:

$$\min_{\mathbf{p}} \sum_{i=1}^{m-1} (a_i - \bar{s})^2 + \lambda \Delta_i, \tag{3}$$

where a regularization term Δ_i weighted by λ is introduced to penalize a large jump at i^{th} step,

$$\Delta_i = \begin{cases} \frac{|p_i - p_{i+1}|}{\beta}, & \text{if } |p_i - p_{i+1}| < 3\beta, \\ \infty, & \text{otherwise.} \end{cases}$$

In the experiments, we empirically constrained the maximum step size to be three times the average sampling gap, *i.e.*, $3\beta = 3\frac{n}{m-1}$. Due to its additive nature, Eq. 3 can be globally minimized by dynamic programming (DP). As shown in Fig. 3b, the DP method generates a non-uniform sampling that adapts to the saliency.

4.2. Smoothing

The proposed DP sampler optimally subsamples frames from a high-speed input given its saliency curve. In practice, however, the saliency curve can change dramatically at motion boundaries or contain random variations due to image noise. Consequently, the generated sampling function may be locally noisy, yielding noticeable artifacts in the regular speed output. As shown by the blue dashed line in Fig. 3d, the DP sampling function has several sharp peaks in its 2^{nd} derivative (Fig. 3e). Since the DP-based method is based on pairwise errors (Eq. 3) between samples, it is not possible to enforce 3^{rd} smoothness and curvature on the solution. To handle this limitation, we instead apply a smoothing step.

Given a sampling function $\mathbf{p} \in \mathbb{R}^m$, the goal of smoothing is to find a monotonic approximation $\mathbf{q} \in \mathbb{R}^m$ that remains in a similar global pattern as $\mathbf{p}$ while leaving out the rapid change in fine-scale structure. Inspired by the work [18] on approximating time warping functions, we parameterize the sampling function $\mathbf{q} = \mathbf{Q}\mathbf{a}$ as a linear combination of k monotonic bases, $\mathbf{Q} = [\mathbf{q}_1, \cdots, \mathbf{q}_k] \in \mathbb{R}^{m \times k}$ weighted by $\mathbf{a} \in \mathbb{R}^k$. The monotonic bases are generated by piecewisely shifting and scaling sin and cos functions. For instance, the bottom-right corner in Fig. 3f illustrates an example basis. In our experiments, we found that 20 monotonic bases are sufficient.

Given the bases $\mathbf{Q}$ and the input sampling function $\mathbf{p}$, we optimize $\mathbf{a}$ to minimize the following reconstruction error weighted by saliency score $\mathbf{s_p}$ on the sample position:

$$\begin{aligned} \min_{\mathbf{a}} \quad & \|(\mathbf{p} - \mathbf{Q}\mathbf{a}) \odot \mathbf{s_p}\|^2 + \alpha \|\mathbf{L}\mathbf{Q}\mathbf{a}\|^2, \\ \text{s.t.} \quad & \mathbf{F}\mathbf{Q}\mathbf{a} \geq \epsilon > 0, \mathbf{a}^T\mathbf{1} = 1, \end{aligned} \tag{4}$$

where $\mathbf{L} \in \mathbb{R}^{m \times m}$ is the 2^{nd} differential operator and $\|\mathbf{L}\mathbf{Q}\mathbf{a}\|^2$ penalizes the curvature of the approximation $\mathbf{Q}\mathbf{a}$, whose quality can be controlled by the parameter α. To enforce monotonicity of $\mathbf{Q}\mathbf{a}$, we constrain its gradient $\mathbf{F}\mathbf{Q}\mathbf{a} \in \mathbb{R}^m$ to be positive, where $\mathbf{F} \in \mathbb{R}^{m \times m}$ is the 1^{st} differential operator. To prevent repeated frames in the output video, we set a small threshold ϵ on the minimum of gradient. The optimum of Eq. 4 can be efficiently found by solving a small-scale quadratic programming. Fig. 3c shows the sampling result refined by this smoothing step. Compared to DP, the new DPS result has a similar global shape (Fig. 3d) but much smaller local acceleration (Fig. 3e).

5. Temporal filtering

We approximate standard film camera capture with a box filter over the temporal span between frame times[8]. Given a high-speed video, we can also simulate a very short exposure for each frame using a delta filter (*i.e.*, selecting a frame without any blending with other frames). While the delta filter retains the most details, the lack of visible motion blur can produce a discontinuous strobing effect. We introduce two new saliency-based temporal filters for rendering the regular-speed video output.

Adaptive box filter (BoxA). The adaptive box filter simulates shorter-time exposures at more important moments to retain more temporal details. The synthetic exposure at each frame is $w_i = (1 - s_i^\alpha)w_0$, which is a saliency-based exponential falloff curve whose tail length matches w_0. Note that w_0 is the actual exposure window associated with the regular-speed video. The parameter α allows the user to configure the curve and we set $\alpha = 1$ in the experiments.

Saliency-based motion-blur filter (SalBlur). Motion blur is a strong perceptual cue. Our goal in designing SalBlur for high-speed video is to combine the advantages of box and delta filters, *i.e.*, retain the blur for salient motion, while keeping foreground as clear in the original as possible. Given the saliency map, SalBlur renders the video frame in three steps as shown in Fig. 4.

First, we remove the holes and artifacts in the original saliency map through bilateral filtering [23]. A direct filtering of the saliency map might blur the object's boundary. To keep the edge sharp, we compute the pixel-wise kernel of the filter from the original frame.

Second, we compute a binary motion blur mask for each frame based on the refined saliency map and optical flow. The pixels with non-zero mask values correspond to the historic position of important regions that need to be blurred. For example, the bottom-right of Fig. 4 shows five pixels (circles) on a 1-D slice at current frame t_0 and the optical flow (arrows) computed for the previous three frames t_1, t_2 and t_3. Although the third pixel (the filled red circle) locates outside the human body at frame t_0, it very likely belongs to the important region at previous frames (the filled red square) according to the motion history. Following this intuition, we calculate for each pixel i the difference, $r(i,t) = s_{f(i,t)} - s_i$, between its saliency value s_i at current frame t_0 and the ones $s_{f(i,t)}$ at the position $f(i,t)$

[8] We ignore issues such as rolling shutter here.

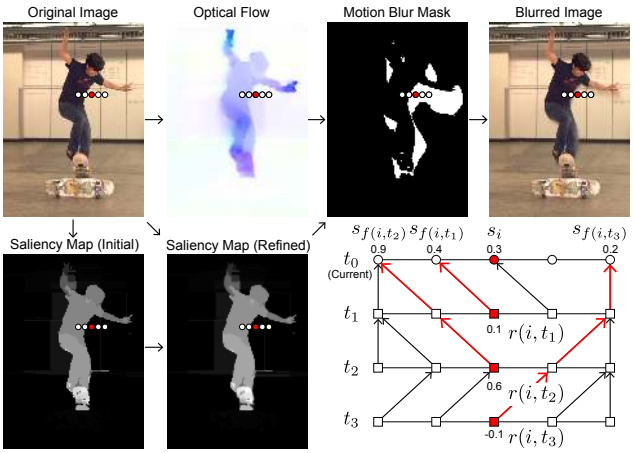


Figure 4. The pipeline of the salient-based motion-blur filter (SalBlur). The bottom-right corner illustrates the computation of motion blur mask for the five pixels (circles) on the 1-D slice at current frame (t_0) using the optical flow (arrows) in the previous three frames (t_1, t_2, t_3). The number of $s_{f(i,t)}$ and s_i indicate the saliency value at different pixel position and $r(i,t) = s_{f(i,t)} - s_i$ computes the saliency difference.

following the flow starting from frame t. A positive $r(i,t)$ indicates a more salient event happened at pixel i at some previous frame t before. We assign a non-zero mask value to i if $r(i,t)$ is positive for some t within the exposure window. In the case of Fig. 4, the mask at the third pixel is non-zero because $r(i,t_1) = 0.1$ and $r(i,t_2) = 0.6$.

The final result is generated by applying a box filter only on the pixels with non-zero values in the mask.

6. Results

We first show our space-time saliency method is state-of-the-art based on a benchmark dataset. This is also justification of its use in re-timing and temporal filtering to generate the output video. To subjectively evaluate the overall effectiveness of our system, we show results of a user study.

6.1. Comparisons using Weizmann database

In the first experiment, we test our proposed saliency method on the Weizmann human action database [6]. This database contains 84 video sequences of nine people performing ten actions. The ground-truth foreground mask of each frame is provided by the authors of this dataset.

We compare our method against three image saliency methods [8, 11, 9] and four video saliency ones [21, 17, 7, 4]. We took the implementation of all the methods from the authors' websites except for [4], for which we implemented according to the paper. We evaluate the accuracy of the computed saliency of each video frame using the ground-truth segmentation mask in the same manner as described in [2]. Given a threshold $t \in [0,1]$, the regions whose saliency values are higher than t are marked as foreground. The segmented image is then compared with the ground-truth mask to obtain the precision and recall values. The average precision-recall curve is generated by combining the results from all the video frames.

A visual comparison of different motion saliency methods is shown in Fig. 5a. As can be seen, the saliency map produced by the proposed method is more visually consistent with the shape, size, and location of the ground truth segmentation map than the maps generated by the other methods. Fig. 5b shows the precision-recall curves. Our method significantly improves on previous motion saliency methods [21, 17, 7, 4] and the image saliency ones [8, 11] by a significant margin. The most competitive method to ours is [9]. However, [9] lacks the fundamental mechanism for enforcing temporal coherence of the saliency map across the video, which is important for our problem.

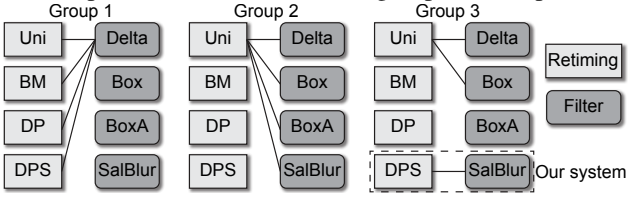
6.2. Evaluation of our system

In the second experiment, we investigate the performance of the proposed system for re-timing and filtering high-speed videos. Given the lack of ground truth to evaluate video quality, we collect a number of high-speed videos and design a user study where subjects compare the quality of differently generated outputs.

Fig. 6a lists the ten videos recorded using a Phantom high-speed camera at between 500 and 1000 fps. These videos (representative frames shown in Fig. 6b) cover a variety of human behaviors and object interactions occurred in different indoor and outdoor scenarios.

To establish baselines for re-timing, we implemented the non-uniform sampling method developed by Bennett and McMillan (BM) [1] as well as a simple uniform sampler (Uni). To evaluate our temporal filters, we compare against a delta filter (Delta, using sampled unmodified original frames) and a box filter (Box, uniformly integrating pixel values within a fixed-length window). In total, we have four different re-timing schemes (Uni, BM, DP, and DPS) and four filters (Delta, Box, BoxA, SalBlur) for comparison. Recall that BoxA and SalBlur, which rely on our saliency measure, are described in Section 5. Implemented in Matlab on a PC platform with 3.6GHz Intel Xeon and 16GB memory, our system takes less than a second to re-time each video and ten seconds for rendering each frame.

Given a significant number of sampling techniques and filters and the need to make the user study manageable for subjects, we selected only a subset of combinations for comparisons. There are three groups of comparisons:



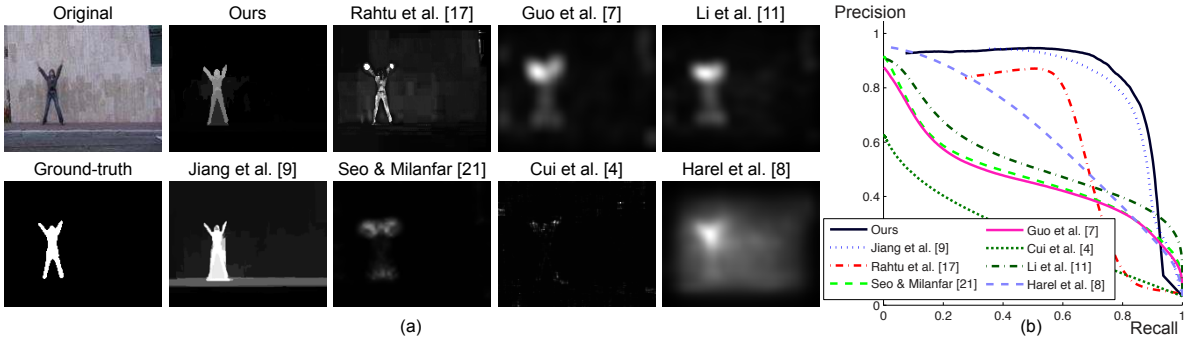


Figure 5. Comparison of different saliency algorithms on the Weizmann dataset. (a) Saliency maps. (b) Precision-recall curves.

These combinations yield 120 video pairs (12 method combinations with 10 sequences). In the user study, the video pairs and their ordering are randomized.

Thirteen subjects (ten men and three women) took part in the user study. Each subject is asked to compare two videos at a time; the two videos were generated using different re-timing method and filter. Each video is shown one after another. The subject is asked to select the video that seems more "informative" or pleasing. To counter the possible bias towards picking the second result, the same result pair would appear once more in the user study but in the reversed order. On average, each participant took about 70 minutes to evaluate all 240 pairs of video results (120 unique pairs repeated).

The re-timing and filtering results of two examples are illustrated in Fig. 7a and Fig. 7b respectively. The results of the user study are summarized in Fig. 7c-e, where the mean and variance of user preferences are plotted for each pair of method combinations. Although video quality assessment is a strong subjective task, we can conclude from Fig. 7c that most users preferred the non-uniform sampling results rather than the simple uniform sampling.

Without smoothing, the proposed DP method received more preferences than BM with a small margin. This is because the lack of smoothness causes some videos to appear unnatural. For instance, the second row of Fig. 7a compares the four sampling methods, where the DP sampling bar abruptly switches in sampling rates for several areas. However, the DP results were largely improved by the smoothing step in DPS.

From Fig. 7d, we found the new BoxA filter is preferred much more than the conventional box filter. This appears to show the effectiveness of adapting the filter length so that the exposure was kept low to reduce motion blur at interesting moments. It is surprising to us that Delta is more popular than Box and BoxA. Based on subject feedback, the attractiveness of Delta is visual clarity of frames. Our SalBlur filter is similar to Delta in that it is able to retain the sharp motion and object boundaries in the output vides. However, SalBlur is also able to introduce motion blur for fast moving objects, which is an important perceptional cue for the case like the "Soccer Kick" example (second row in Fig. 7b). Using frames as-is will result in the strobing effect. Fig. 7e shows that the subjects prefer the results of our new system (DPS + SalBlur) over those of simple direct techniques.

7. Concluding remarks

We have presented a system for *time-mapping*, *i.e.*, converting a HFR video into a regular-speed LFR video while retaining detail in the original video. We propose a new space-time saliency technique, shown to be state-of-the-art in performance on a benchmark dataset. This new saliency technique is the basis of retiming and temporal filtering. A user study shows that our system is very promising in generating pleasing and informative video outputs.

There are several future directions to our work. Currently, we assume the camera is stationary. As future work, we could remove camera motion as a preprocess by determining and tracking background features, while realizing this is also a difficult problem. In addition, we implemented time-mapping using whole frame selection. One challenging extension is to analyze and time-map separate objects differently in the scene. Another direction is to correlate effective retiming and filters with high-level content (*e.g.*, indoor vs. outdoor, type of activity, and object identity).

Acknowledgements. Our initial investigation with Eduardo Gastal on processing high-speed videos helped to provide focus on our time-mapping project. We would also like to thank him and Patrick Meegan for capturing the high-speed video clips used in this paper.

References

[1] E. P. Bennett and L. McMillan. Computational time-lapse video. *ACM Trans. Graph.*, 26(3):102, 2007.

[2] A. Borji, D. N. Sihite, and L. Itti. Salient object detection: A benchmark. In *ECCV*, 2012.

[3] G. J. Brostow and I. A. Essa. Image-based motion blur for stop motion animation. In *SIGGRAPH*, pages 561–566, 2001.

[4] X. Cui, Q. Liu, and D. N. Metaxas. Temporal spectral residual: fast motion saliency detection. In *ACM MM*, 2009.

Name	FPS	#Frames	Size
Glitter	1000	3614	800 × 1280
Laptop Smash	1000	2359	800 × 1280
Skate	500	1050	800 × 576
Dirt Fall	1000	5000	800 × 1280
Cone Kick	1000	1222	498 × 640
Soccer Kick	1000	4022	448 × 832
Soccer Juggle	1000	2000	680 × 512
Car	800	944	800 × 1280
Face Slap	1000	2501	768 × 1024
Cricket	500	1655	647 × 896

(a)

(b)

Figure 6. The high-speed video sequences used for experimental study. (a) Video statistics. (b) Key frames.

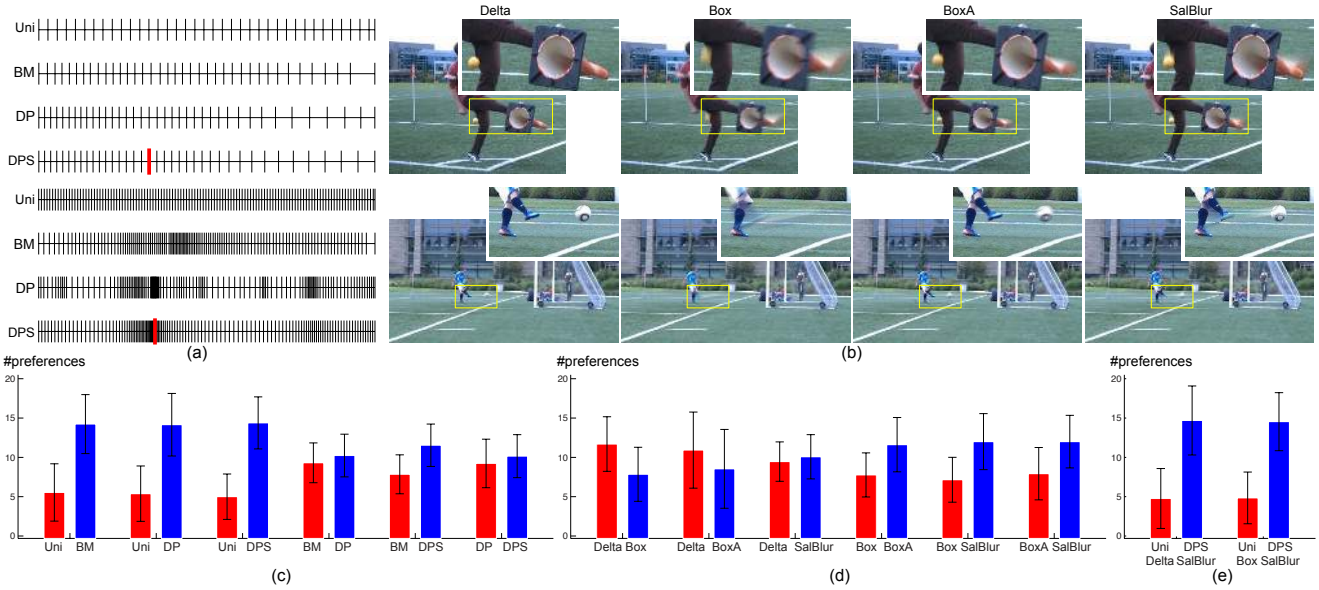


Figure 7. Comparison on rendering high-speed video in regular frame rate. (a) Comparison on the re-timing of two examples, where the red bold line in the DPS sampling bar indicates the position of the filtering results shown in (b). (c-e) Pairwise preferences averaged over 13 participants for the generated regular-speed videos using different (c) video re-timers, (d) temporal filters and (e) entire systems.

[5] M. Fuchs, T. Chen, O. Wang, R. Raskar, H.-P. Seidel, and H. P. A. Lensch. Real-time temporal shaping of high-speed video streams. *Computers & Graphics*, 34(5):575–584, 2010.

[6] L. Gorelick, M. Blank, E. Shechtman, M. Irani, and R. Basri. Actions as space-time shapes. *IEEE Trans. Pattern Anal. Mach. Intell.*, 29(12):2247–2253, 2007.

[7] C. Guo, Q. Ma, and L. Zhang. Spatio-temporal saliency detection using phase spectrum of quaternion Fourier transform. In *CVPR*, 2008.

[8] J. Harel, C. Koch, and P. Perona. Graph-based visual saliency. In *NIPS*, 2006.

[9] H. Jiang, J. Wang, Z. Yuan, T. Liu, and N. Zheng. Automatic salient object segmentation based on context and shape prior. In *BMVC*, 2011.

[10] H.-W. Kang, Y. Matsushita, X. Tang, and X.-Q. Chen. Space-time video montage. In *CVPR*, 2006.

[11] J. Li, M. D. Levine, X. An, X. Xu, and H. He. Visual saliency based on scale-space analysis in the frequency domain. *IEEE Trans. Pattern Anal. Mach. Intell.*, 35(4):996–1010, 2013.

[12] C. Liu. *Beyond pixels: exploring new representations and applications for motion analysis*. PhD thesis, MIT, 2009.

[13] C. Liu, A. Torralba, W. T. Freeman, F. Durand, and E. H. Adelson. Motion magnification. *ACM Trans. Graph.*, 24(3):519–526, 2005.

[14] F. Navarro, F. J. Serón, and D. Gutierrez. Motion blur rendering: State of the art. *Comput. Graph. Forum.*, 30(1):3–26, 2011.

[15] F. Perazzi, P. Krähenbühl, Y. Pritch, and A. Hornung. Saliency filters: Contrast based filtering for salient region detection. In *CVPR*, 2012.

[16] Y. Pritch, A. Rav-Acha, and S. Peleg. Nonchronological video synopsis and indexing. *IEEE Trans. Pattern Anal. Mach. Intell.*, 30(11):1971–1984, 2008.

[17] E. Rahtu, J. Kannala, M. Salo, and J. Heikkilä. Segmenting salient objects from images and videos. In *ECCV*, 2010.

[18] J. O. Ramsay and B. W. Silverman. *Functional Data Analysis*. 2005.

[19] A. Rav-Acha, Y. Pritch, D. Lischinski, and S. Peleg. Evolving time fronts: Spatio-temporal video warping. Technical report, 2005.

[20] D. Rudoy, D. Goldman, E. Shechtman, and L. Zelnik-Manor. Learning video saliency from human gaze using candidate selection. In *CVPR*, 2013.

[21] H. Seo and P. Milanfar. Static and space-time visual saliency detection by self-resemblance. *J. Vis.*, 9(12), 2009.

[22] E. Shechtman, Y. Caspi, and M. Irani. Space-time super-resolution. *IEEE Trans. Pattern Anal. Mach. Intell.*, 27(4):531–545, 2005.

[23] C. Tomasi and R. Manduchi. Bilateral filtering for gray and color images. In *ICCV*, 1998.

[24] B. T. Truong and S. Venkatesh. Video abstraction: A systematic review and classification. *ACM Trans. on Multimedia Comput.*, 3(1), 2007.

[25] P.-H. Tseng, R. Carmi, I. Cameron, D. Munoz, and L. Itti. Quantifying center bias of observers in free viewing of dynamic natural scenes. *J. Vis.*, 9(7), 2009.

[26] N. Wadhwa, M. Rubinstein, F. Durand, and W. T. Freeman. Phase-based video motion processing. *ACM Trans. Graph.*, 32(4):80, 2013.

[27] J. Wang, S. M. Drucker, M. Agrawala, and M. F. Cohen. The cartoon animation filter. *ACM Trans. Graph.*, 25(3):1169–1173, 2006.

[28] H.-Y. Wu, M. Rubinstein, E. Shih, J. V. Guttag, F. Durand, and W. T. Freeman. Eulerian video magnification for revealing subtle changes in the world. *ACM Trans. Graph.*, 31(4):65, 2012.

[29] C. Xu, C. Xiong, and J. J. Corso. Streaming hierarchical video segmentation. In *ECCV*, 2012.