Vis Comput (2017) 33:749–759
DOI 10.1007/s00371-017-1372-y

ORIGINAL ARTICLE

# High-speed video generation with an event camera

**Han-Chao Liu**[1] · **Fang-Lue Zhang**[2] · **David Marshall**[3] · **Luping Shi**[4] · **Shi-Min Hu**[1,3]

Published online: 10 May 2017


**Abstract** The event camera is a kind of visual sensor that mimics aspects of the human visual system by only recording events when the light intensity on a pixel changes. This allows for an event camera to possess high temporal resolution and makes it able to capture fast motion. However, an event camera lacks information for all pixels within a scene, especially color information. In this paper, we aim to recover a typical scene in which the foreground undergoes high-speed motion which can be approximated by a planar motion, and the background is static. We demonstrate how to use the event camera to generate high-speed videos of 2D motion augmented with foreground and background images taken from a conventional camera. We match an object obtained for a static image to frames formed by the event stream, from the event camera, based on curve saliency, and we build a parametric model of affine motion to create image sequences. In this work, we are able to restore scenes of very fast motion such as falling or rotating objects and string vibration.

**Electronic supplementary material** The online version of this article (doi:10.1007/s00371-017-1372-y) contains supplementary material, which is available to authorized users.

✉ Shi-Min Hu
shimin@tsinghua.edu.cn

1 TNList, Department of Computer Science and Technology, Tsinghua University, Beijing, China

2 School of Engineering and Computer Science, Victoria University of Wellington, Wellington, New Zealand

3 School of Computer Science and Informatics, Cardiff University, Cardiff, UK

4 Center for Brain-Inspired Computing Research (CBICR), Department of Precision Instrument, Tsinghua University, Beijing, China

**Keywords** Event camera · Image matching · Video enhancement · Sensor fusion

## 1 Introduction

A conventional camera captures motion as a sequence of images. Recording high-speed motion at a high frame rate requires expensive high-speed cameras with a very short exposure time. High-speed motion recording is also expensive in computation and storage as the recorded image sequence contains enormous redundant information such as a static background. However, in the human visual system there is no concept of a frame. New information is only received when part of a scene changes [16]. Such changes are encoded as neural spikes, which are sufficient for us to sense motion in the world.

An event camera mimics the biological retina. The idea was first proposed by Mahowald and Mead (1991) and practical products for research use are now available [6]. An event camera only captures variation of light intensity instead of image sequences. The data output by an event camera is in the form of an event stream, and an event is generated only when the relative light intensity change on a pixel is above a threshold. As it does not record a whole image, the event camera yields a very high temporal resolution of a few microseconds.

Current event camera research focuses on reconstructing grayscale images from event streams [4,12,19], while [7] restore grayscale images and videos from event streams in high-speed and high-dynamic-range scenes. We find that the reconstructed images, though high in temporal quality, still contain many visual artifacts and are bereft of color information. A few works [17,21] have focussed on tracking tasks in high-speed scenes alone and utilized a hybrid system that combines a conventional camera and an event camera. How-

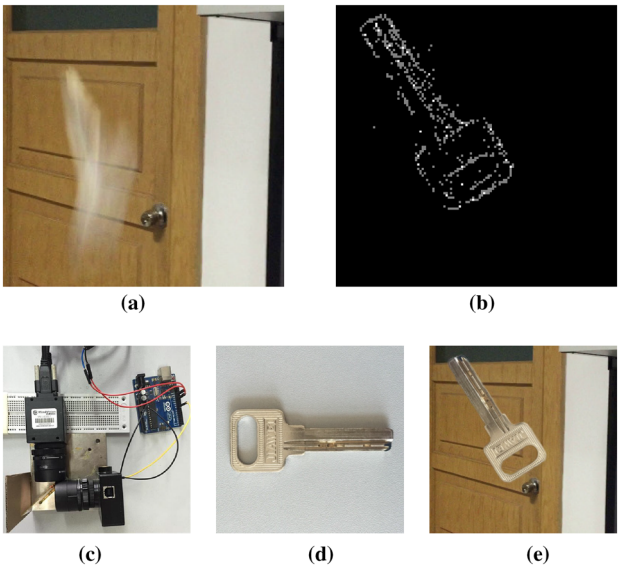


**Fig. 1** **a** A severely blurred image taken by a conventional low-frame-rate camera. **b** A frame generated by an event camera, where nonzero pixels represent events. **c** Our hybrid system that combines a conventional camera and an event camera. **d** An additional static foreground image. **e** A frame in our synthesized video with a frame rate of around 2k fps, which reveals fast motion and *color*

ever, they do not really make full use of images from the conventional camera.

In our work, we partially overcome the above drawbacks by retaining both motion and visual image color content. We propose a pipeline to generate high-speed videos with an event stream from an event camera and several images from a conventional camera. We note that events form sharp edges and salient curves of objects in motion because moving edges undergo the most intense light changes (Fig. 1b). Therefore, we can match images from two sensors by aligning edges. Motivated by this observation, we focus on recovering a typical scene. We assume that the scene has a static background so as not to trigger excessive events and the foreground object undergoes high-speed 2D affine motion. This means that a static foreground image (Fig. 1d) and one or several background images (Fig. 1a) are sufficient to compensate for the loss of color and texture in the event stream. It should be noted that on limitation of these assumptions is that 3D motion, such as a rolling textured ball, is not able to be captured. To our knowledge, the proposed approach is the first to fuse event data and images to generate video frames for high-speed motion.

In our framework, the recorded event stream is sampled into frames to ensure enough boundary information in each frame. We acquire static foreground and background images from the same view with a conventional camera. Using these source images, we extract salient curves including edges in the foreground and complete the background. An affine motion model is obtained by aligning the foreground and event frames with an incremental refinement approach, based on the proposed measure of curve saliency. With this affine model, we are able to transform colored foreground and background to the event frames and composite video frames (Fig. 1e).

The synthesized videos achieve high temporal resolution (thousands of frames per second) which are able to restore some fast motion, such as falling of objects and string vibration. In the work, we use the dynamic vision sensor (DVS) [14], a version of the event camera that has a spatial resolution of 128 × 128 pixels. We justify the necessity of static foreground images rather than blurred images taken from low-frame-rate consumer level cameras by implementing a video enhancing method based on deblurring techniques. We also compare our matching method with [17], which tracks objects using a point-based approach.

## 2 Related work

Though inspired by human vision, the practical issue how to make best use of the event camera remains a challenge. Kim et al. [12] reconstructed super-resolution images from an event camera under pure rotational motion. They estimated the camera rotation and image gradient in parallel, and reconstructed images by posing the problems as the solution to a Poisson equation. Bardow et al. [4] further developed event-based optical flow and Reinbacher et al. [19] reconstructed intensity images using manifold regularization. While their work shows that intensity information of a scene is still retained with an event camera recording, color and delicate texture information cannot be recovered. Their work requires that the event camera moves continuously to allow for background capture within the scene. Also, the reconstructed videos still have many artifacts such as noise points and blurred regions on the boundaries because of inaccurate estimation.

Hybrid systems with a conventional camera and an event camera have also been devised to take the advantages of both sensors. Saner et al. [21] made such a system and performed registration between two sensors. However, they only presented the high-speed tracking of point trajectories and did not really fuse the images and the event stream. Our approach has also focused on such a hybrid system.

Ben and Nayar [8] first used a hybrid system in which a low-resolution camera tracks the motion path with high temporal resolution. They used the motion trajectory to estimate the motion blur kernel and achieved in deblurring images. Tai et al. [22] later used such a system to correct deblurred videos by both estimating the blur kernel and using back-projection of low-resolution images. Gupta et al. [10] and Ancuti et al. [2] enhanced the spatial and/or temporal resolution of

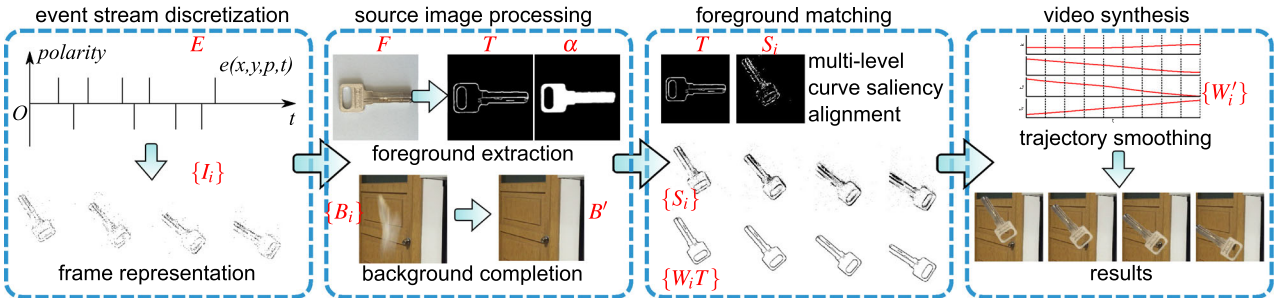


**Fig. 2** Our hybrid camera system high-speed video generation pipeline

a low-quality video with some high-resolution still images. Unfortunately, those techniques cannot handle our problem as our low-temporal-resolution image sequence lacks explicit texture information. We compare of method to [8] in Sect. 8.

The domains of image and shape registration provided approaches for our method that matches objects in two sensors. Irani and Anandan [11] presented image alignment of different sensors based on correlation. Baker and Matthews [3] presented a good evaluation of image alignment methods derived from the well-known Lucas–Kanade algorithm. We adopt the inverse compositional method for our matching approach which is based on curve saliency. Also our work bears similarity to event-based pattern tracking approaches. Ni et al. [17] introduced a fast shape tracking algorithm using an event camera. They adopted an event-based iterative closest points (ICP) registration method, which yields smaller error and is more robust in the presence of fast object motion. However, in very complex scenes, their approach is highly sensitive to noise and algorithm tuning parameters. We make a comparison to this method in Sect. 8.

## 3 Overview—Our hybrid camera system high-speed video generation framework

In our framework, we assemble a hybrid system combining a conventional camera and an event camera. A beam splitter provides the same scene for both sensors, with their optical axes and respective views aligned (Fig. 1c). As a result, they practically capture the same scene. Our objective is to match the events to the gradient information in order to transfer the high-speed motion to colored scenes (Fig. 2).

The system inputs the event stream $E$ recording the high-speed motion of the foreground, and source images for obtaining foreground pixels $F'$ and background pixels $B'$. The source images include (1) a static foreground image $F$ and (2) a single background image of simple texture $B$ or several video frames $\{B_i\}$. We use GrabCut [20] to extract the foreground object pixels $F'$ from $F$, and refine the boundary with an alpha mask $\alpha$ using alpha matting [13]. We obtain the complete background pixels $B'$ either by completing the occluded region of $B$ using PatchMatch [5], or using interactive mosaicing [1] with multiple frames $\{B_i\}$. In some cases, $F'$ and $B'$ can be acquired from one source image of the scene.

The recorded event stream $E$ is first discretized into frames $\{I_i\}$ with fixed number of events $N$, which ensures enough motion information in each frame. As events are very sparse, frames need refinement. For each frame $I_i$ , we warp adjacent frames and aggregate them. Thus, we form a curve saliency map $S_i$ for the time point, the value on each pixel in our curve saliency map representing the possibility that a curve of strong contrast exists. We then use a hierarchical segmentation technique [18] to extract pixels of boundaries and high saliency curves in color/texture in $F'$, which we denote as $T$.

As previously stated, we assume a 2D affine transformation between images in two sensors. We align two images with an incremental refinement method based on curve saliency maps $S_i$, $T$, which we have described above, for both event camera frames and source images, and we obtain warp chain $\{W_i\}$, each from $T$ to $S_i$. We refine the matching by applying a low-pass filtering on parameters in the transformation matrix. We interpolate warps $\{W_i\}$ for time points with constant intervals and obtain $\{W_i'\}$. Finally we use alpha blending to synthesize the warped foreground pixels $W_i'F'$ and background $B'$ to generate each frame in high-speed videos.

## 4 Event stream and image preprocessing

### 4.1 Frame generation

*Event stream discretization* An event camera records motion in event streams. Each event is in the form of $e(x, y, p, t)$, where $x$, $y$ are pixel addresses of the event, $p$ is the polarity of light intensity change and $t$ is the timestamp measured to within an accuracy of microseconds. An event is generated when the relative intensity change exceeds a threshold $\theta$.

$$\left|\log \frac{I(x, y, t)}{I(x, y, t_p)}\right| \geq \theta \tag{1}$$

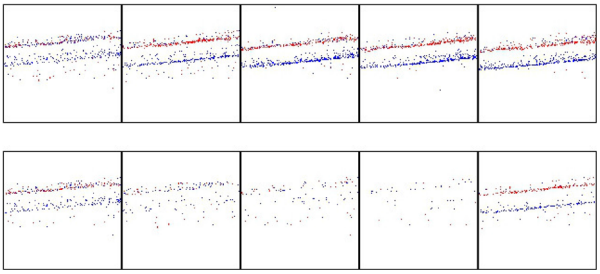

**Fig. 3** Event stream of a vibrating string. *Upper*: sliced in fixed number of events, $N = 500$. *Lower*: sliced in fixed time interval, $\Delta t = 0.5$ ms. The moment that the string has zero velocity is adaptively skipped if sliced with fixed number of events. The real scene is the example *guitar string*, shown in Fig. 7

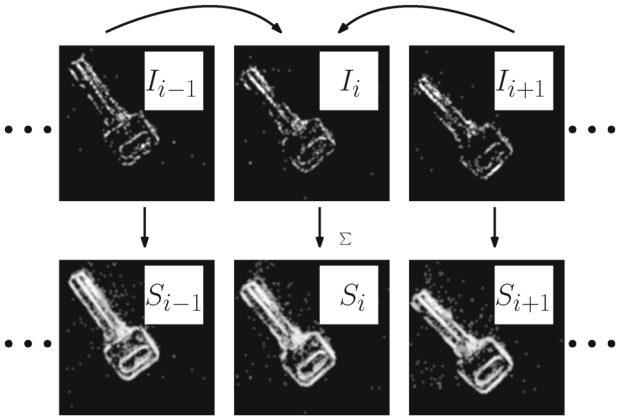


**Fig. 4** In raw frames $I_i$, the active pixels are sparse. We construct curve saliency maps $S_i$ to combine information of multiple frames

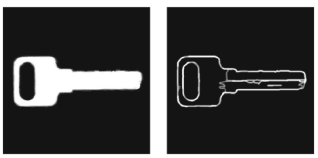


**Fig. 5** **a** alpha mask $\alpha$ using matting. **b** curve saliency map $T$ using hierarchical segmentation. The source image of foreground is shown Fig. 1d.

$t_p$ denotes the timestamp of a previous event. We assume high contrast between foreground (object in motion) and background; thus, when a static event camera captures scenes, only edges and high-contrast texture of moving objects and regions of light changes will fire events. As events accumulate over a short interval, a frame (in the sense of a normal camera) forms.

We first transform the event stream into a more traditional video frame representation. One approach is to generate images (video frames) with events over a fixed time period. However, in this way some frames may remain blank when the scene is static as they contain no events. An asynchronous event camera enables us to generate frames with a fixed number of events [21]. This adaptive method ensures that every frame has enough events and boundary information for later matching process (Fig. 3).

To achieve this, we record $\{t_i\}$, the middle timestamp of time window formed by a fixed number of events $N$. We later interpolate motion to create a constant frame rate video. Usually $N$ can be set empirically by judging the boundary completeness of the foreground in an event frame (Fig. 7). Our approach is relatively robust to the choice of $N$ as we later consider consecutive frames. For each example video, our choice of $N$ is listed in Sect. 6. In each sliced frame $I_i$, the value of a pixel is set as the number of events of during the time window, which is analogous to intensity integration.

*Frame refinement* The generated frames $\{I_i\}$ may still contain fragmented curves that lack strong constraints for the later matching process (Fig. 4). Also, the event stream typically contains considerable noise, which from a raw frame cannot be readily distinguished from 'active' pixels. We, therefore, introduce a measure called curve saliency, which gives large weight to pixels that indicate high possibility of real curves. We, also, propose a method to construct a map of curve saliency for each frame by aggregating warped adjacent frames.

For each frame $I_i$, we set a neighborhood $\Omega$ consisting of its adjacent frames. Successive frames have overlapping events and thus are temporally correlated. We warp those frames and aggregate them on the center frame (Fig. 4). The value of a pixel in the curve saliency map $S_i$ is simply the sum of values on all warped frames.

$$S_i(\mathbf{p}) = \sum_{j \in \Omega(i)} I_j(W_{ji}\mathbf{p}) \tag{2}$$

where $\mathbf{p} = (x, y)^T$. $W_{ji}$ is the affine transformation from the $j$th frame to the center $i$th frame. This process can be intuitively interpreted to be the integration of multiple consecutive frames that fills missing parts on the raw frame.

We warp frames based on the Lucas–Kanade alignment algorithm [3] with a common Sum of Squares of Distances (SSD) error function. When warping frame $I_j (j > i)$, for example, the warping matrix $W_{j,i}$ is initialized with $W_{j-1,i}$, the previous warping result of the most adjacent frame. In practice, we set the neighborhood size to 5–10 pixels.

### 4.2 Source image processing

A conventional camera captures images from the same perspective as the event camera. In nearly all cases, objects in motion are seriously blurred when taken simultaneously. Therefore, we have to independently capture static images of the objects.

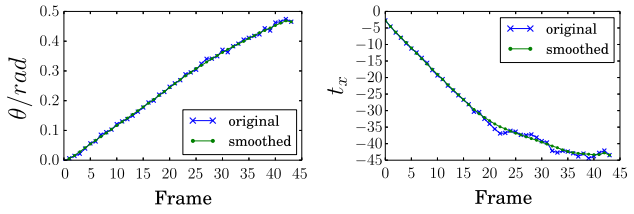


**Fig. 6** Each component in the parametric motion model is smoothed to achieve temporal consistency. Rotation and translation along $x$-axis are displayed, in the case of the *falling key*

*Foreground extraction* We extract the foreground pixels $F'$ and alpha mask $\alpha$ with smoothed edges from the source image $F$ with regard to the object in motion (Fig. 5). We follow a similar approach in [9]. We first use GrabCut [20] to roughly segment the foreground by indicating foreground region and background with strokes. Strokes have to be carefully labeled if the foreground in $F$ is accompanied by a textured background (*e.g.,* source image of the *guitar string* and *spinning fan* cases is shown in Fig. 7) instead of an independent monochromatic one (Fig. 1d). We extend the segmentation edge to generate a trimap specifying the uncertain region and use alpha matting proposed in [13] to obtain more precise foreground pixels $F'$ and a mask with smooth edges $\alpha$. In order to get a representation corresponding to the saliency map in event camera frames, we use hierarchical segmentation [18] to extract curves of high saliency including boundaries in $F'$ and obtain saliency map $T$. In contrast to alternative gradient-based approaches, hierarchical segmentation avoids extracting curves in the cluttered image regions. The value of a pixel defines the curve saliency in this map.

*Background completion* For simple uncluttered and predictable backgrounds (such as case *guitar string* in Fig. 7), we fill in the missing part of source image $B$ using PatchMatch [5] to get a full $B'$. For more complex backgrounds (Fig. 1a), we utilize image mosaicing [1], to interactively composite a background $B'$ from the source image sequence $\{B_i\}$. The main idea here is that occluded part of the background can be replaced with the region in another non-occluded aligned image.

## 5 Foreground alignment

### 5.1 Matching method

Our goal is to find the warp chain $\{W_i\}$ from the curve saliency map of template $T$ to $\{S_i\}$. The transformation can be represented as

$$W = \begin{pmatrix} 1+p_1 & 1+p_2 & p_5 \\ 1+p_3 & 1+p_4 & p_6 \\ 0 & 0 & 1 \end{pmatrix} \tag{3}$$

let $\boldsymbol{p} = (p_1, p_2, p_3, p_4, p_5, p_6)^T$ and we minimize the error between normalized $S$, $T$.

$$E(\boldsymbol{p}) = \sum_{(x,y)} (T(W(x, y; \boldsymbol{p})) - S(x, y))^2 \tag{4}$$

To solve it, we use the inverse compositional algorithm based on the Lucas–Kanade method [3], which iteratively refines the warp until its error converges. In order to achieve a improved result, we adopt a multi-scale search.

$T$ and $S_i$ are first scaled to a similar size. It is possible that the curve saliency map obtained by hierarchical segmentation yields regions that are too thin, which results in a flat region of minima of the error function. To avoid this, we expand a nonzero region of $T$ by applying a Gaussian blur in advance to ensure $S$ and $T$ have approximately same width of curves. We manually give an initial $\boldsymbol{p}_0$ for $S_1$ and solve for Eq. (4), and afterward, $W_i$, the warp from $T$ to $S_i$, is initialized using a previous result $W_{i-1}$.

We do not always apply a full affine transformation to the scenes. In some cases, the event stream only recovers motion along a specific dimension. In other cases, we have prior knowledge about the motion. In both cases, we reduce $\boldsymbol{p}$ to a suitable form of motion.

### 5.2 Trajectory smoothing

The warp chain $\{W_i\}$ that depicts motion should be temporally consistent. However, the generated $\{W_i\}$ contains jitter on each element along time axis. This is mainly due to (1) not accounting for temporal inconsistency in the error function and (2) the bandwidth of $T$ and $S_i$ being identical, meaning there is scope for the warp to stretch. To overcome the above, we perform a low-pass filtering on $\{W_i\}$ and eliminate the high-frequency component (Fig. 6). We do not wish to intentionally smooth the path but merely reduce the unwanted high-frequency jitter. We use convolution-based smoothing method [15]. In order to choose a suitable filter kernel size, $\boldsymbol{p}$ is first transformed into $(M_x, M_y, t_x, t_y, \theta, s)^T$ with scaling, translation, rotation and shear and then transformed back to matrix form $W$ after filtering.

### 5.3 Motion interpolation and synthesis

Since we generate the sequence of event camera frames based on the fixed number of events, we have to transform it back into an image sequence with constant time intervals for display.

Given the timestamp of each frame $\{t_i\}$, we interpolate smoothed $\{W_i\}$ using linear interpolation and obtain $\{W'_i\}$ with constant intervals. The new time interval between frames $\Delta t$ ($1/\Delta t$ being the video frame rate) is set approximately to be the minimum of $\Delta t_i$. We mainly want

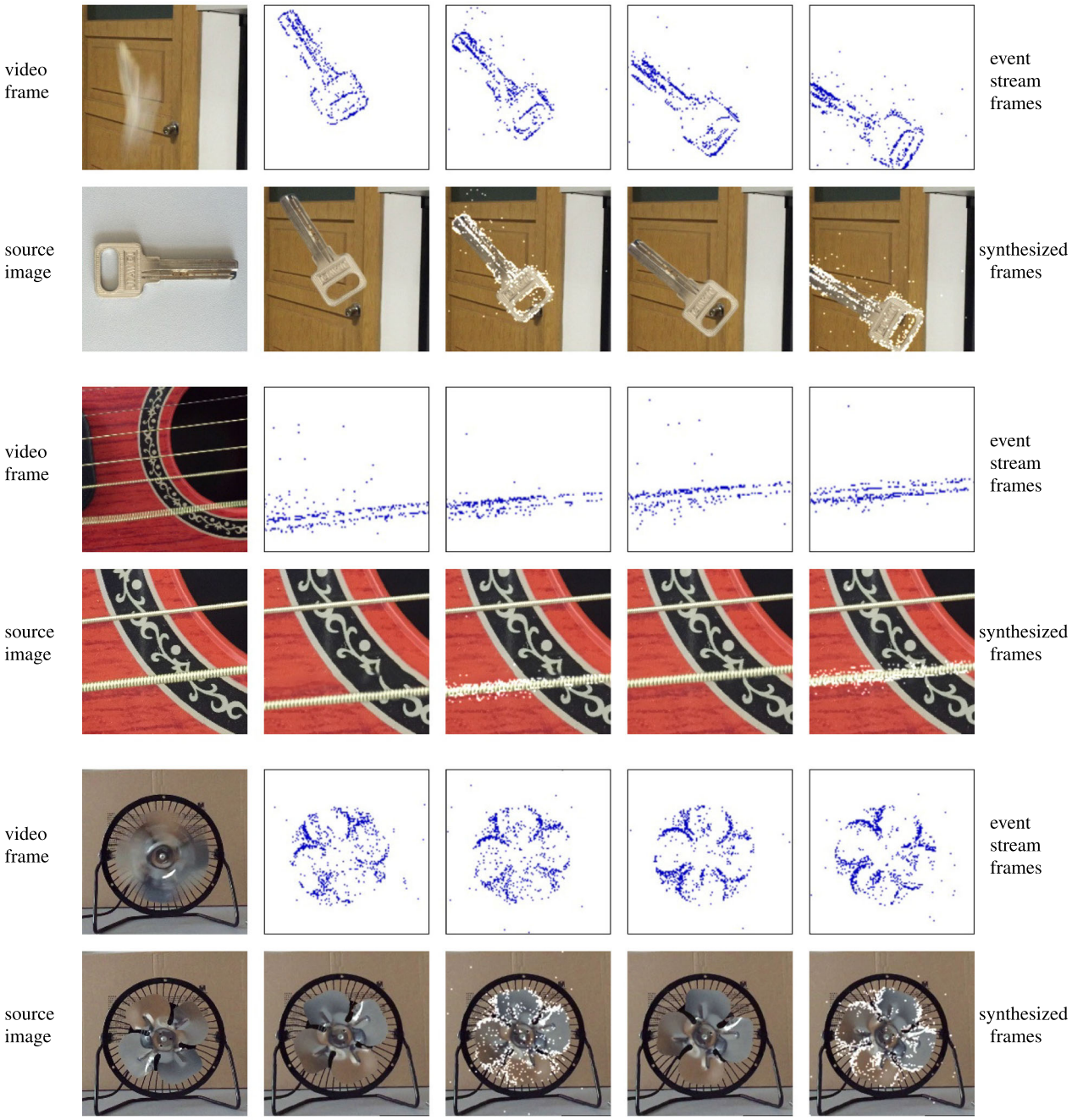


**Fig. 7** Examples: from *top* to *bottom falling key, guitar string* and *spinning fan*. Events (white pixels) are plotted on the final synthesized frames to highlight the matching results

to recover slow motion during moments that have been skipped by discretization with fixed number of events. Note that we are not able to recover the motion which is at a higher frequency than the current event sampling rate. We composite each frame $I'_i$ in the high-speed video using alpha matting-based composition $I'_i = W'_i(\alpha F') + (W'_i(1-\alpha))B'$.

## 6 Results

We restore scenes of some very fast motion and generate high-speed videos using our method. They are shown in the supplementary videos. In order to better illustrate the results, we decrease the display speed. Event streams are displayed with fewer events per frame and at a standard frame rate

**Table 1** Choice of parameters used in the examples

| | Number of events per frame $N$ | Temporal resolution $\Delta t$/ms |
|---|---|---|
| *Falling key* | 500 | 0.5 |
| *Guitar string* | 400 | 0.2 |
| *Spinning fan* | 800 | 1 |

(30 fps). Choices of $N$ and temporal resolution $\Delta t$ of each video are listed in Table 1.

*Restoration of affine motion* A key is falling freely in the air, and we assume a full affine motion. The synthesized video has a temporal resolution of 0.5 ms and a frame rate of 2k fps. While a camera at a normal frame rate (30 fps) captures scenes with severe blur, the event camera frames clearly show the trajectory. As the key is blurred, we take an independent image of the static foreground. We complete the background using mosaicing. We replace the occluded part with the corresponding region of another aligned background image. The synthesized video is approximately 67 times faster than the normal (30 fps) video rate. We construct curve saliency maps for event camera frames, and we get connected boundaries of the foreground.

*Restoration of translational motion* We set a guitar string tuned to vibrate at around 50 Hz. When capturing at 30 fps using a normal camera, the image is blurred and the exact location of the string is difficult to find. We are able to restore the accurate location of the string and generate a high-speed video. Due to the simple texture of the string and its background, we only use one high-quality image that is not blurred. We directly extract foreground from the source image and complete the background using PatchMatch. As we only care about the motion of vibration, we choose a translational motion model and disregard irrelevant parameters. The equivalent frame rate is 5 k fps which is approximately 167 times faster than the standard video rate. Noise points are significantly reduced in the constructed curve saliency maps by integrating consecutive frames. Plotting the displacement along the time axis, we can also recover amplitude attenuation of the vibration.

*Restoration of rotational motion* A fan is spinning at approximately 320 rpm. When recording at 30 fps using a normal camera, the contour of blades is almost invisible. We take images of the fan when it is still and another series of images in which the blade is at different location. We use interactive mosaicing to generate a complete background. We also disregard irrelevant parameters and focus mainly on rotation. The contour of the fan blade rises above from unrelated events in curve saliency maps. We attain a high-speed video of 1k fps, about 33 times faster than the standard video rate. We can also calculate the rotational speed of the fan from the warp chain.

## 7 Extension: revealing intensity changes

Although our method primarily restores fast motion, the frame sequence generated from an event stream can also reveal intensity variation if the scene is static.

When dealing with light intensity variation, we discretize the event stream with fixed time intervals. Polarity information is used here. First, regions undergoing fast intensity changes can be matched to the source image with a similar boundary matching. Then, according to Eq. (1), polarity can be integrated in the sense of log intensity.

$$\log I_i(x, y) = \log I_{i-1}(x, y) + (n_{ON} - n_{OFF})\theta \tag{5}$$

where $I_i(x, y)$ denotes the intensity value on $(x, y)$ in $i$th frame. $n_{ON}, n_{OFF}$ are number of ON and OFF events in the time window of $i$th frame. As we do not really know the exact value of $\theta$, we only restore scenes of qualitative intensity changes, similar to the color magnification [23]. Log intensity is then transformed into relative intensity change, and changes along time axis are added to the source image. We manipulate intensity changes in YIQ space.

In real cases, events are very sparse at high temporal resolution (e.g., 0.1 ms), so the event stream frames should be downsampled sufficiently to reveal the pattern of variation. Moreover, we should have some prior knowledge about the change to ensure a reasonable scene restoration.

*Restoration of periodic intensity changes* An LED driven by pulse width modulation (PWM) is periodically changing brightness. Video captured in 30 fps shows the constant maximum value in a point on the LED. With an event camera we are able to restore the changing scene (Fig. 8). We first match the region of LED by aligning edges. We set an extended alpha mask to introduce the effect of light halo. Events in the region of interest are downsampled to a single point to reveal pattern of changes. As we have prior knowledge that intensity is constant from beginning to end, we filter out low-frequency disturbance in the frequency domain due to noise. The recovered video shows periodic intensity changes. The event stream is sliced into frames each 0.1 ms. We reveal a frequency of periodic change of 494 Hz, compared to the ground truth 490 Hz. The video obtained about 333 times faster than the normal video rate.

## 8 Discussion

### 8.1 Comparisons with other techniques

*Comparison with enhancement by motion deblurring* The work of Tai et al. [22] with less motion blur can be generated with a high-frame-rate low-resolution camera and a low-frame-rate high-resolution camera. Since they simultaneous

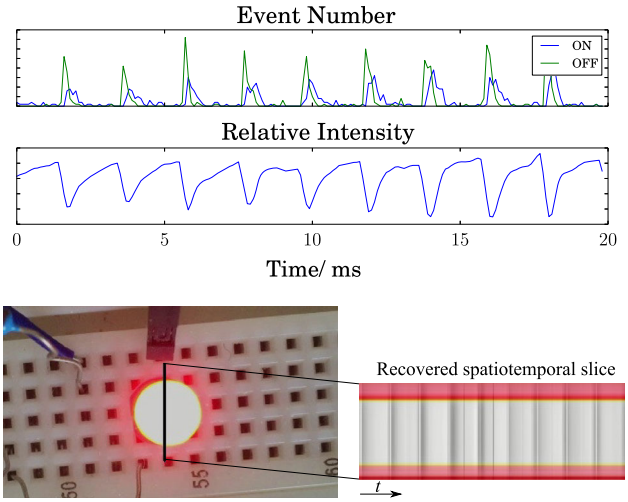


**Fig. 8** Example: *blinking LED*. The event stream has a very high temporal resolution of 0.1 ms, but the events are very sparse. We spatially downscale the region of interest sufficiently to reveal intensity variation, which is clear in the spatiotemporal slice

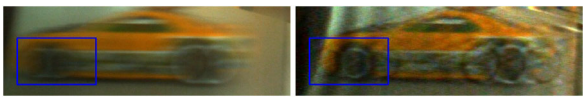

**Fig. 9** A simultaneous capture of a toy car under translational motion and the deblurred foreground. While some parts have been correctly recovered, it suffers from ringing artifacts

record both cameras they do not need additional images of static scenes to be taken. The first camera functions similarly to the event camera in our hybrid system, so we investigated whether the motion information brought by the event camera can be used to perform motion deblurring.

We follow the idea in [8] which discusses the deblurring of moving objects with the assistance of a low-resolution camera tracking motion. In the scene of a running toy car, we have a parametric model with one-dimensional translation. We simultaneously record image sequence from the conventional camera and the event camera and we try to deblur the foreground. We estimate the blur kernel using our consecutive curve saliency maps instead of optical flow.

The result (Fig. 9) shows that some blurred parts of the toy car are recovered, but the result still posses some significant ringing artifacts. Also, the edges still merge with the background. This can be explained since (1) we estimate a rough global kernel instead of a spatial-variant kernel, and (2) the motion estimation using consecutive curve saliency maps ($128 \times 128$) does not reach a sufficient resolution. The deblurred image increases the difficulty for the following segmentation and matting process, and affects the visual effects of the synthesized image sequence. Consequently, an image of the static scene, though requiring some additional effort, is greatly beneficial for generating good results.

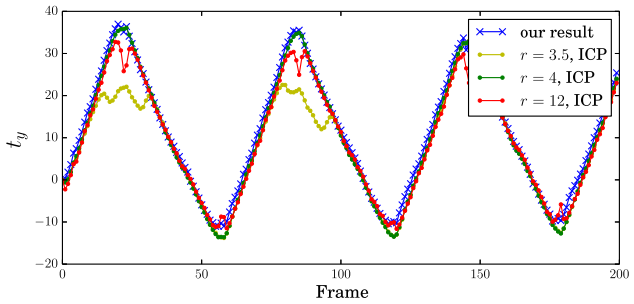


**Fig. 10** Matching results for *guitar string* using our method and point-based approach [17]

*Comparison with point-based matching* There are usually two types of image alignment. One is performed directly on whole images, and the other is based on key points such as feature points. In our method, we actually transform a set of sparse points into an image of edge map thus adopt direct alignment. Ni et al. [17] performed a point-based matching in their tracking application. They first obtain a set of points representing the boundary of the object and then use approach of iterative closest point (ICP) to find point correspondence. We compare the matching results on translation component in the example *guitar string*.

To illustrate the difference, we did not apply trajectory smoothing any of these results. We find that the point-based approach is more sensitive to noise and parameters, especially the largest acceptable distance $r$ between corresponding points in the event stream and the model. It has to be set carefully, which means that noise level has to be correctly estimated. In Fig.10, small and large estimation of $r$ yields bad results. The noise level varies in different cases so the suggested value does not well apply in all cases. Our method well handles noise by assigning large weights to events on real edges and performing image-based alignment. As a result, we avoid tedious parameter tuning of other methods.

### 8.2 User study

So far, there are no available public datasets that record high-speed motion in event streams along with video ground truth. In order to evaluate our synthesis results, we build a dataset to conduct a user study. Ten subjects, including 5 males and 5 females, aged 18–28, participated in this user study. They have no prior knowledge about event cameras. They are asked to simultaneously watch the event stream and synthesized video in the four example cases described in this paper. Then they gave scores to the video they watched according to the following criteria: motion rationality of synthesized videos, video quality and scene comprehensibility (to what extent the synthesized video assists scene comprehension compared

**Table 2** Average scores in the evaluation of our work

| | Motion rationality | Video quality | Scene comprehensibility |
|---|---|---|---|
| *Falling key* | 2.3 | 3.8 | 3.7 |
| *Guitar string* | 4.0 | 4.2 | 4.1 |
| *Spinning fan* | 2.8 | 3.6 | 3.8 |
| *Blinking LED* | 3.7 | 3.5 | 4.4 |

with mere event stream). Each score ranges from 1 (poorest) to 5 (best). The average scores are listed in Table 2.

The overall video quality of our generated results is acceptable, as is indicated in the above results. The criterion of scene comprehensibility has high scores in all cases, meaning that subjects find the scenes generated by our synthesized videos much easier to understand. Subjects reported high scores in the cases *guitar string* and *blinking LED*, meaning that they have been well recovered and look natural. However, the motion rationality of the *falling key* and *spinning fan* examples received a relatively low score. Subjects reported that the key in 2D motion looks a little unnatural and the restored motion of the fan contains jitter. Our evaluation shows that a full motion such as homographic or 3D transformation and motion smoothness regularity should be considered for future work to make recovered motion more natural.

### 8.3 Limitations

While we demonstrate many promising results, our method has several limitations. First, we can only recover large and smooth motion for objects with simple shapes. The matching is performed in the original spatial resolution of the event camera and we did not apply super-resolution estimation such as in [12] so we only recover large motion. We filter on the parameters with the assumption of smooth motion; in this way, we also filter out jitter that possibly comes along with the large motion.

Secondly, the event discretization in our method may contribute errors. We adaptively skip those moments, but they are later interpolated as very slow motion. However, it is tolerable when we set $N$, events per frame to a suitable value.

Thirdly, we cannot deal with objects that have deformation or undergo 3D rigid motion. We only assume an affine model for the motion. Intensity reconstruction approaches well handle these cases, although they can only recover part of the scene. Additionally, our matching accuracy can further be improved by adding geometric constraints in the error function. An example of failed matching result appears in the last frame of the *falling key* when part of the foreground contour falls out of view (Fig. 7).

Last, we cannot determine the true light intensity change but only the overall pattern. Note that we only recover the periodic pattern of light changes in example *blinking LED*, which is similar to [23] that magnifies color changes.

## 9 Conclusion

We have proposed a framework that uses the high-temporal-resolution event stream of an event camera and foreground and background images to generate high-speed videos. We have introduced a measure of curve saliency for matching images from the two sensors, and we construct curve saliency maps with warping and hierarchical segmentation. Our results show that by using our method, the event camera has potential applications in high-speed motion restoration, visualization and measurement.

**Acknowledgements** We thank all the anonymous reviewers for their helpful and constructive comments. This work was supported by the Natural Science Foundation of China (Project Number 61521002), the Joint NSFC-ISF Research Program (project number 61561146393), Research Grant of Beijing Higher Institution Engineering Research Center and Tsinghua-Tencent Joint Laboratory for Internet Innovation Technology. Luping Shi was supported in part by the Beijing Municipal Science and Technology Commission (Z151110000090000), the Study of BrainInspired Computing of Tsinghua University (20141080934) and SuZhou-Tsinghua innovation leading program 2016SZ0102.

## References

1. Agarwala, A., Dontcheva, M., Agrawala, M., Drucker, S., Colburn, A., Curless, B., Salesin, D., Cohen, M.: Interactive digital photomontage. In: ACM Transactions on Graphics (TOG), vol. 23, pp. 294–302. ACM (2004)
2. Ancuti, C., Haber, T., Mertens, T., Bekaert, P.: Video enhancement using reference photographs. Vis. Comput. **24**(7), 709–717 (2008)
3. Baker, S., Matthews, I.: Lucas-kanade 20 years on: A unifying framework. Int. J. Comput. Vis. **56**(3), 221–255 (2004)
4. Bardow, P., Davison, A.J., Leutenegger, S.: Simultaneous optical flow and intensity estimation from an event camera. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 884–892 (2016)
5. Barnes, C., Shechtman, E., Finkelstein, A., Goldman, D.B.: PatchMatch: a randomized correspondence algorithm for structural image editing. ACM Trans. Graph.(Proc SIGGRAPH) **28**(3), 24 (2009)
6. Barth, F.G., Humphrey, J.A., Srinivasan, M.V.: Frontiers in Sensing: From Biology to Engineering. Springer, Berlin (2012)
7. Barua, S., Miyatani, Y., Veeraraghavan, A.: Direct face detection and video reconstruction from event cameras. In: Applications of Computer Vision (WACV), 2016 IEEE Winter Conference on, pp. 1–9. IEEE (2016)
8. Ben-Ezra, M., Nayar, S.K.: Motion-based motion deblurring. IEEE Trans. Pattern Anal. Mach. Intell. **26**(6), 689–698 (2004)
9. Cheng, M.M., Zhang, F.L., Mitra, N.J., Huang, X., Hu, S.M.: Repfinder: finding approximately repeated scene elements for image editing. ACM Trans. Graph. **29**(4), 83:1-8 (2010)

10. Gupta, A., Bhat, P., Dontcheva, M., Deussen, O., Curless, B., Cohen, M.: Enhancing and experiencing spacetime resolution with videos and stills. In: Computational Photography (ICCP), 2009 IEEE International Conference on, pp. 1–9. IEEE (2009)
11. Irani, M., Anandan, P.: Robust multi-sensor image alignment. In: International Conference on Computer Vision, pp. 959–966 (1998)
12. Kim, H., Handa, A., Benosman, R., Ieng, S.H., Davison, A.: Simultaneous mosaicing and tracking with an event camera. In: Proceedings of the British Machine Vision Conference. BMVA Press (2014). doi:10.5244/C.28.26
13. Levin, A., Lischinski, D., Weiss, Y.: A closed-form solution to natural image matting. IEEE Trans. Pattern Anal. Mach. Intell. **30**(2), 228–242 (2008)
14. Lichtsteiner, P., Posch, C., Delbruck, T.: A 128 × 128 120 db 15 $\mu$ s latency asynchronous temporal contrast vision sensor. IEEE J. Solid-State Circuits **43**(2), 566–576 (2008)
15. Matsushita, Y., Ofek, E., Tang, X., Shum, H.Y.: Full-frame video stabilization. In: Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on, vol. 1, pp. 50–57. IEEE (2005)
16. Meister, M., Berry, M.J.: The neural code of the retina. Neuron **22**(3), 435–450 (1999)
17. Ni, Z., Ieng, S.H., Posch, C., Benosman, R.: Visual tracking using neuromorphic asynchronous event-based cameras. Neural Comput. **27**(4), 1–29 (2015)
18. Paris, S., Durand, F.: A topological approach to hierarchical segmentation using mean shift. In: Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on, pp. 1–8. IEEE (2007)
19. Reinbacher, C., Graber, G., Pock, T.: Real-time intensity-image reconstruction for event cameras using manifold regularisation. In: Proceedings of the British Machine Vision Conference. BMVA Press (2016)
20. Rother, C., Kolmogorov, V., Blake, A.: Grabcut: Interactive foreground extraction using iterated graph cuts. ACM Trans. Graph. (TOG) **23**(3), 309–314 (2004)
21. Saner, D., Wang, O., Heinzle, S., Pritch, Y., Smolic, A., Sorkine-Hornung, A., Gross, M.H.: High-speed object tracking using an asynchronous temporal contrast sensor. In: VMV, pp. 87–94 (2014)
22. Tai, Y.W., Du, H., Brown, M.S., Lin, S.: Correction of spatially varying image and video motion blur using a hybrid camera. IEEE Trans. Pattern Anal. Mach. Intell. **32**(6), 1012–1028 (2010)
23. Wu, H.Y., Rubinstein, M., Shih, E., Guttag, J., Durand, F., Freeman, W.T.: Eulerian video magnification for revealing subtle changes in the world. ACM Trans. Graph. **31**(4), 1–8 (2012)

**Han-Chao Liu** is an undergraduate student at Tsinghua University. He is currently interested in computer graphics, image and video processing.

**Fang-Lue Zhang** is a lecturer at Victoria University of Wellington. He received his doctoral degree from Tsinghua University in 2015 and bachelor degree from Zhejiang University in 2009. His research interests include image and video editing, computer vision and computer graphics. He is a member of ACM and IEEE.

**David Marshall** has been working in the field of computer vision since 1986. In 1989, he joined Cardiff University as lecturer and is now Professor of Computer Vision in the School of Computer Science and Informatics. David's research interests include articulated modeling of human faces, models of human motion, statistical modeling, high-dimensional subspace analysis, audio/video image processing and data/sensor fusion. He has published over 150 papers and one book in these research areas and has attracted over £4M in research funding over his academic career. He is currently Head of the Visual Computing Research Group and director of the Human Factors Technology Centre. http://users.cs.cf.ac.uk/Dave.Marshall/

**Luping Shi** is a National distinguish professor, found director of center for brain inspired computing research(CBICR) at Tsinghua university(THU), China. He is a SPIE fellow. He received Doctor of Science from University of Cologne, Germany in 1992. He joined THU in 2013 to found CBICR, which is currently only one center in China to study on brain inspired computing in all directions including fundamental theory, chip, software, system and application platform.

**Shi-Min Hu** is currently a professor in the department of Computer Science and Technology, Tsinghua University, Beijing. He received the PhD degree from Zhejiang University in 1996. His research interests include digital geometry processing, video processing, rendering, computer animation and computer-aided geometric design. He has published more than 100 papers in journals and refereed conference. He is Editor-in-Chief of Computational Visual Media, and on editorial board of several journals, including IEEE Transactions on Visualization and Computer Graphics, Computer Aided Design and Computer & Graphics.