Computer Vision and Image Understanding xxx (2011) xxx–xxx

Contents lists available at SciVerse ScienceDirect

Computer Vision and Image Understanding

journal homepage: www.elsevier.com/locate/cviu

Content aware video manipulation

Moshe Guttmann *, Lior Wolf **, Danny Cohen-Or

The Blavatnik School of Computer Science, Tel-Aviv University, Tel-Aviv 69978, Israel

ARTICLE INFO

Article history:
Received 22 April 2010
Accepted 6 May 2011
Available online xxxx

Keywords:
Video and image retargeting
Video optimization

ABSTRACT

Content aware video manipulation (CAVM) is a method for the analysis and recomposition of video footage, by means of content analysis and adaptive video warping.

One main motivation of CAVM is "video retargeting", a process that visually alters an existing video while considering the relative importance of its various regions. CAVM video retargeting aims at preserving the viewers' experience by maintaining the information content of important regions in the frame, while altering the video dimensions. Other applications include commercial real-estate allocations, time and space content summary, and content deletion (in both time and spatial domain).

In this paper we introduce an efficient algorithm for the implementation of CAVM. It consists of two stages. First, the video is analyzed to detect the importance of each pixel in the frame, based on local saliency, motion detection and object detectors. Then, a transformation manipulates the video content according to the aforementioned analysis and application dependent constraints. The visual performance of the proposed algorithm is demonstrated on a variety of video sequences, and compared to the state-of-the-art in image retargeting.

1. Introduction

With the increasing flexibility of video usage, the need for content based video manipulation arises. For example, mobile LCD displays of various dimensions are now ubiquitous, and there is an acute need for conversion systems that can alter video content to fit a variety of displays, which are smaller than originally intendant by the creators of the video.[1]

These mobile displays come at many aspect ratios. The current industry solutions for video aspect ratio altering are basic and not very effective. They include: blunt aspect ratio free resizing; cropping the middle of the video; resizing while preserving the aspect ratio by adding black stripes above and below the frame; and keeping the center of the frame untouched while warping the sides. In fact, it is common nowadays to have printed lines on movie-cameras' screens that mark the region that will be visible in the frame after it would be cropped to the aspect ratio of a regular 4:3/16:9 TV screen. No commercial solution exists which examines the content of the video.

Our CAVM system examines the footage, and assigns a saliency score to each pixel in the video. An optimized transformation of the input video to a manipulated version is then calculated. The transformation adheres to the application-based constraints and respects the saliency score. The algorithm is designed to work efficiently in an online manner, and performs in real-time on live streaming video input. The saliency score is composed of three basic components: spatial gradient magnitude, a face detector and a block-based motion detector. The optimization stage amounts to solving a sparse linear system of equations. It considers spatial constraints as well as temporal ones, leading to a smooth coherent user experience.

1.1. Previous work

In previous decades, vast image processing research tackled the down-sampling and up-sampling problem. These classical methods, however, are not "content aware"—they apply the same local operator everywhere across the image, oblivious to the semantics of the image and to the varying importance and sensitivity to distortion of each image region.

Recently, with the ever increasing need to alter the dimensions and aspect ratio of images and videos, the subject of retargeting has regained an increased academic attention, and a number of contributions have been published. Suh et al. [2] considered the problem of cropping optimal thumbnails from an input image. Although the task is different from that of retargeting (thumbnails are used for easy access, not to convey the entire content of an image), some of the components in their system have reappeared in later retargeting systems. Most notably, defining an importance

* Corresponding author.
** Principal corresponding author.
E-mail addresses: guttm@cs.tau.ac.il (M. Guttmann), wolf@cs.tau.ac.il (L. Wolf), dcor@cs.tau.ac.il (D. Cohen-Or).

[1] A preliminary version of this work was published in the Proceedings of the 11th IEEE International Conference on Computer Vision (ICCV-07), 2007 [1].

1077-3142/$ - see front matter © 2011 Published by Elsevier Inc.
doi:10.1016/j.cviu.2011.05.010

measure based on both a local low-level saliency measures, such as the one introduced by Itti and Koch [3], and on high level object detectors, such as the face detector of Viola and Jones [4].

An extended cropping mechanism, applied to video is presented by Liu and Gleicher [5]. They propose a per-frame score for cropping windows that also takes into account aspect ratio distortions and information lose due to down-sampling. Time-wise, they penalize cropping window shifts that do not correspond to plausible camera motion. Compared to our system their system is limited to displaying a sequence of cropped windows.

A non-photorealistic solution for retargeting stills is proposed by Setlur et al. [6]. Their system is based on the background/foreground separation, and relies heavily on their capability to solve the separation problem. In their system the foreground is clipped on the resized background, where the missing background regions are filled using inpainting. Another non-photorealistic system is the one of Liu and Gleicher [7], where an optimal fish-eye transformation is computed based on the image content.

Several recent systems introduce photorealistic solutions for content-aware warping of still images, see [8] for a survey. Gal et al. [9] present a method to modify an arbitrarily image warp while preserving the shape of important region by constraining their deformation to be a similarity/rigidity transformation. Inspired by their system, we introduced a content aware video retargeting system [1] that defines the video warping as a solution of a sparse linear system of equations. Unlike Gal et al.'s our innovative system is fully automatic. Following these two [10] introduced a local solver for image warping somewhat close in spirit to the solution of [1] with an added local saliency measure (gradient based). Unlike [1], their system cannot separate between the warping of the *X* and *Y* axis. While this is desirable, it does limit the applicability and robustness of the solution. The work of [11] extends the system proposed by [1] into the realm of scalable video retargeting, i.e. an expensive preprocessing is performed on an input video, and then only a fast efficient computation is needed for a specific retargeting task. They present comparable visual results to [1] while their storage expenses are greatly reduced (due to the scalable manner of their preprocessing system).

A different, discrete, approach is proposed by Avidan and Shamir [12], where the retargeting is applied by reducing the width (or the height) of the image by one pixel at a time, through the deletion of a vertical (horizontal) connected path of low importance pixels. An extension to video retargeting using graph-cuts was introduced by Rubinstein et al. [13]. They consider the video sequence as a 3D space-time cube, where the goal is to remove 2D seams from the 3D-cube to achieve a video retargeting solution. Similar to the original "Seam Carving" work [12] it is a discrete solution, and in addition to necessitating the entire shot, it is also computationally expensive. In contrast, our system is designed in a scalable manner for online video, and in particular for streaming video, and allows the implementation of real-time video manipulating systems.

Following the optimization approach introduced by [1], [14] suggests a more elaborated linear system for retargeting that includes added manual saliency constraints and introduces an image-processing enhancements in the warping stage. [15] and later [16] address the important temporal coherence problem of retargeting video. In [15] the motion constraints are constructed based on camera motion estimation and added to the linear system to ensure frame aliment; Later on in [16] a similar concept is used, however, instead of a simple mesh based warping, the optimal cropped window over the entire video is also computed. Later we show that our camera motion extension manages camera motion scenarios in a related way.

2. System overview

Our CAVM system, described in Fig. 1, consists of two main stages. A computation of the saliency matrix and a mapping calculation stage. Then, the calculated warping is rendered by a forward-mapping technique.

Given a new frame we compute a per-pixel importance measure. This measure (see Section 3) is a combination of three factors: A simple, gradient based, local saliency; an off-the-shelf face detector; and a high-end motion detector.

The optimization of the mapping function from the source frame to the manipulated target frame is set through a linear system of equations. Each pixel (i,j) at frame t is associated with two variable $x_{i,j,t}$, $y_{i,j,t}$ that determines its location on the manipulated frame. Horizontal and vertical warps are optimized independently using the same technique. In order to persevere content integrity, and to avoid distortion, the horizontal post-warp location is first encouraged to have the same coordinates as the warp of the pixel just below it $(x_{i+1,j,t})$, and the pixel preceding it $(x_{i,j,t-1})$. Penalties are also placed in order to encourage a distance of 1 from the warp of its left neighbor $(x_{i,j-1,t})$.

For obvious reasons, it is impossible to satisfy all of the requirements above and still uphold the manipulated target's specifications (e.g, fit into smaller retargeting dimensions). In order to satisfy these specification, we add weights to these space preserving energy terms that are proportional to the pixel importance values. For example, a pixel with high importance is mapped to a distance of approximately 1 from its left neighbor, while a pixel of less saliency is mapped closer to its neighbor. Note that time smoothness is taken into consideration by the above mentioned relation between $x_{i,j,t}$ and $x_{i,j,t-1}$, in order to generate a continuous natural-looking video sequence.

Our algorithm is designed for streaming video. Therefore, time smoothness and motion analysis considerations are limited to the previous frames only. Such considerations need only apply to frames of the same shot, and an online shot detection mechanism is employed. We also present an extensions that manipulates a video in an off-line manner, the entire video shot at once.

The shot-segmentation mechanism is similar in spirit to the motion-discontinuity based method of [17], where the block matching operation is replaced with the efficient algorithm of [18]. First, motion estimation by the method of [18] is applied to each 16×16 pixels macro-block. A shot boundary is detected wherever the number of blocks for which the motion estimation

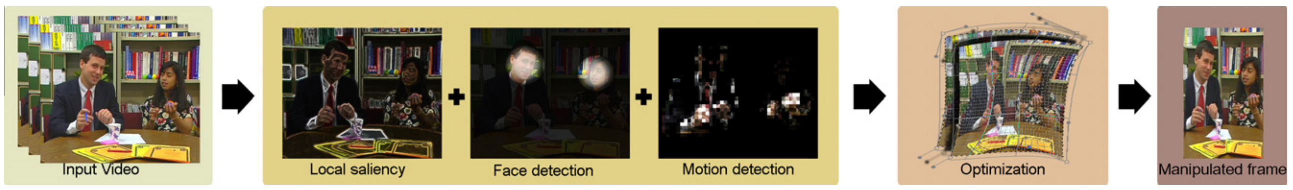

Fig. 1. System overview. A saliency score is computed for each frame based on the gradient magnitude, object detection and motion detection. Next, an optimization stage recovers the CAVM warp. The warp is then applied to the original frame.

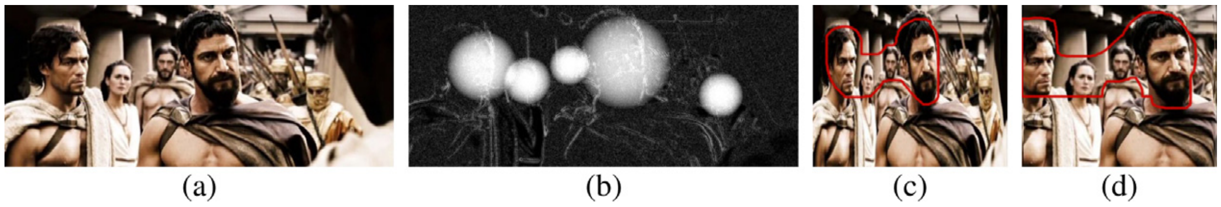

Fig. 2. Retargeting example on a frame from the movie (a). The saliency map, with the detected faces imposed (b). The retargeting result to half the width without face detection (c). Retargeting with face detection (d). The entire retargeted shot compared to bicubic interpolation is available in the supplemental material [19].

fails, exceeds a threshold. This combination is efficient, robust and uninfluenced by object and camera motion.

3. Saliency

We aim at modeling the visual attention by mapping each pixel with a saliency measure, where ϵ value is considered an unattractive pixel and 1 is an important pixel. In order to locate the salient regions in the video, we construct a three way saliency measure, combined to create the per-pixel saliency. The combined saliency measure is then used by the optimization stage to produce a saliency-preserving manipulated video. Next we describe the three saliency measures and their linear combination.

Let S be the content preservation (saliency) matrix. Each entry in the matrix S represents the saliency of a single pixel in the source frame I. Values varies between ε and 1, where ε values are, content wise, non-important pixels. the following formula is computed elementwise:

$$S = \max\left(\varepsilon, \min\left(S_E + \sum_i S_F^i + S_{MD}, \mathbf{1}\right)\right), \tag{1}$$

where the summation is over all detected faces in the frame.

3.1. Local saliency

We employ the simplest measure of the local information content in the frame. Namely, we use the L_2-norm of the gradient

$$S_E = \sqrt{\left(\frac{\partial}{\partial x} I\right)^2 + \left(\frac{\partial}{\partial y} I\right)^2} \tag{2}$$

(all gray values are scaled to be between zero and one).

Previous work, e.g. [2,12], consider several alternative energy measures. However, we found the efficient L_2-norm of the gradient to, generally, perform no worse than other, more complex, saliency measure functions.

3.2. Face detection

Human perception is highly sensitive to perspective changes in faces, more specifically to frontal/profile portraits. In order to avoid deforming frontal/profile portraits we employ the Viola and Jones [4] face detection mechanism.

The detector returns a list of detected faces. Each detected face i has a 2D center coordinate F_p^i and a radius F_r^i. The face detection score of each pixel is a function of the distance from the faces' center: $D_i(x,y) = \left\| F_p^i - (x,y) \right\|_2$, and it is given by the cubic function:

$$\widehat{S}_{Fi}(x,y) = 1 - \frac{-D_i(x,y)^3 + .5 \cdot D_i(x,y)^2}{-\left(F_r^i\right)^3 + .5 \cdot \left(F_r^i\right)^2} \tag{3}$$

The cubic function (normalized between 0 and 1) is used to weigh the importance of the face as an almost constant function with a drastic fall near the end of the face. This allows some flexibility at the brim of the face whilst avoiding face deformation.

We further introduce a face rescaling measure.

$$F_{rn}^i = \frac{F_r^i}{\max(C_{Width}, C_{Height})}$$
$$S_F^i(x,y) = \widehat{S}_F^i(x,y)\left(1 - 2.5 \cdot \left(F_{rn}^i\right)^4 - 2.5 \cdot \left(F_{rn}^i\right)^2\right) \tag{4}$$

The implicit saliency of a detected face is rescaled in relation to the area it occupies in an $C_{Width} \times C_{Height}$ pixels frame. A **1** factor is used where the size of the face is relatively small, while extremely large faces tend to be ignored. This rescaling factor prevents a distorted zooming effect on large faces.

In view of the fact that we constrain smoothness over columns, a detected face also prevents thinning the regions below it. Therefore, human bodies are shrunk less, as necessitated Fig. 2.

3.3. Motion detection

Moving objects in video draw most of the viewers' attention and are content-wise important. Using a motion detection mechanism we mange to manipulate the video while preserving the temporal context.

The motion detectors suggested by [20] is implemented. The selected algorithm is efficient and effective, although little known. Let the frame be partitioned into $N \times N (N = 8)$ pixel square blocks and A_{uv} denote the (u,v)th block. The pixel coordinate (x,y) is in A_{uv} if $(u-1)N + 1 \leqslant x \leqslant uN$ and $(v-1)N + 1 \leqslant y \leqslant vN$. define $x' = (x)_{mod\ N}$ and $y' = (y)_{mod\ N}$.

For each block A_{uv}, the total intensity of the block at frame t is calculated: $A_t(u,v) := \sum_{x'=1}^{N}\sum_{y'=1}^{N} I_t(x,y)$. Then, the normalized "circular shift moments" in the x and y directions $mx_t^j(u,v), my_t^j(u,v)$ are computed for $j = 0, \ldots, N-1$.

$$mx_t^j(u,v) = \frac{\sum_{x'=1}^{N} (x-j)_{mod\ N} \cdot \sum_{y'=1}^{N} I_t(x,y)}{A_t(u,v)}$$
$$my_t^j(u,v) = \frac{\sum_{y'=1}^{N} (y-j)_{mod\ N} \cdot \sum_{x'=1}^{N} I_t(x,y)}{A_t(u,v)} \tag{5}$$

A motion in block (u,v) is detected if the maximum absolute difference in any of the computed moments between two consecutive frames is larger than a threshold. i.e., no motion is detected if for all j, $\left| mx_t^j(u,v) - mx_{t+1}^j(u,v) \right| < \chi$ and $\left| my_t^j(u,v) - my_{t+1}^j(u,v) \right| < \chi$. In our system we set χ to 0.3.

The motion based saliency $S_{MD}(x,y)$ is rescaled between zero and one according to the motion in block $A_{(\lfloor x/N \rfloor)(\lfloor y/N \rfloor)}$.

As can be seen in Fig. 3 moving objects gain saliency, thus seizing a larger area in the retargeted video.

4. Optimization

We cast the problem of finding the optimal mapping between the source image and the manipulated target image as a sparse set of linear equations, which we solve in a least squares manner.

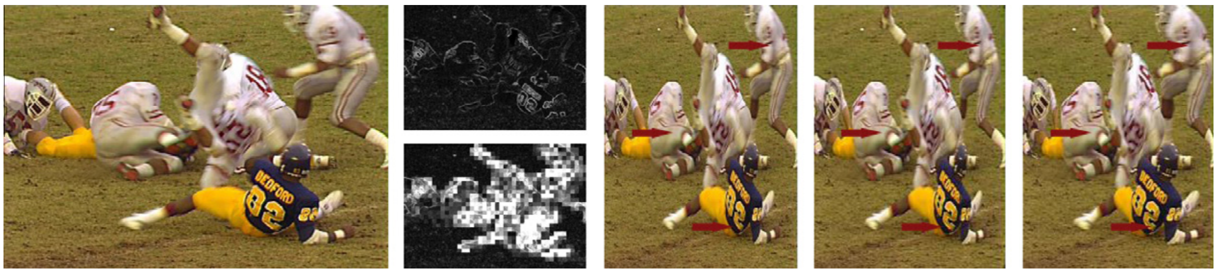

Fig. 3. From left to right. Retargeting a frame taken from the MPEG committee standard benchmark video "football". Saliency maps with (bottom) and without (top) motion based saliency. the result of bicubic interpolation to half the width. retargeting without motion based saliency. Retargeting result of the full clip is available in the supplemental material [19].

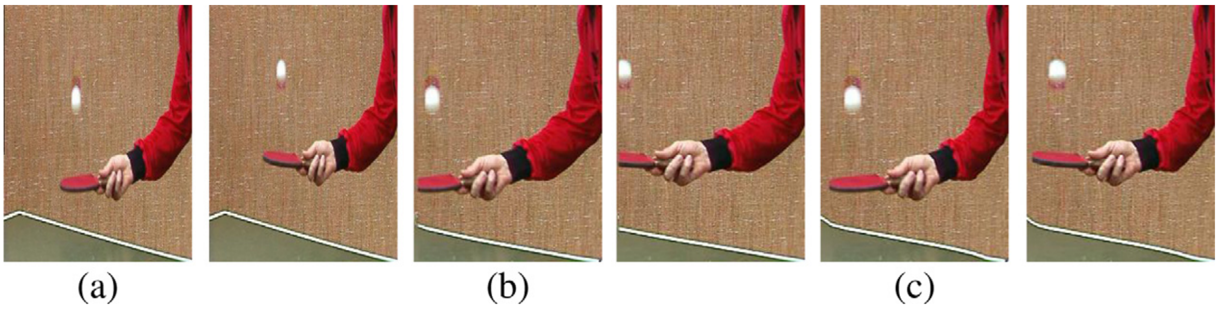

Fig. 4. Retargeting examples on frames from the ITU-T video "tennis" with and without time smoothness. The left image depicts the retargeting results of frame 10, and the right of frame 15. Retargeting results are shown for bicubic interpolation (a), frame by frame retargeting (b), and time smoothed retargeting (c). Time smoothing prevents the video from "jumping around".

A more natural formalization is to cast the problem as a constrained linear system. This way one can guarantee that no pixel falls out of bounds and that the mapping preserves the order of the pixels along the scan lines in the image. However, the solution to the unconstrained system is much more efficient and, in practice, the mappings recovered using an unconstrained system of equations do not contain noticeable artifacts due to changes in the order of the pixels.

In the manipulation process a pixel (i,j) in frame t of the video is being mapped into a pixel in frame t of the output video with some computed location $(x_{i,j,t}, y_{i,j,t})$. Hence, there is twice the number of variables ($x_{i,j,t}$ and $y_{i,j,t}$) to solve for, as the number of pixels in the input video. We compute the y variables separately from the computation of the x variables, using the same linear system describe below. The mapping computation is done one frame at a time, and so our systems of equations has approximately the same number of unknowns as the number of pixels in one input frame.

Consider the problem of recovering the new x-axis locations $x_{i,j,t}$ of pixels $(i,j), j = 1,\ldots,C_{Width}$, $i = 1,\ldots,C_{Height}$, in frames $t = 1,\ldots,C_{Duration}$. The problem of determining $y_{i,j,t}$ is the transpose of this problem and is solved similarly. Also, as a running example, we retarget a video into a smaller display screen, i.e. a mapping of the input frame to a smaller one, width $C_{TargetWidth} < C_{Width}$. Other applications will follow.

There are four types of constraints. First, we constrain each pixel to be at a fixed distance from its left and right neighbors. Second, each pixel needs to be mapped to a location similar to the one of its upper and lower neighbors. Third, the mapping of a pixel at time t needs to be similar to the mapping of the same pixel at time $t-1$. The forth constraint fits the warped locations to the dimensions of the target video frames.

Importance modeling If a pixel is not "important" it can be mapped close to its left and right neighbors hence blending with them. An "important" pixel, however, needs to be mapped far from its neighbors, thus a region of important pixels is best mapped into a region of a similar size. We formulate these insights into equations stating that every pixel should be mapped at a horizontal distance of 1 from its left and right neighbors. These equations are weighed such that equations associated with pixels with higher importance-score are more influential on the final solution.[2] The first type of equations is therefore:

$$S_{i,j,t}(x_{i,j,t} - x_{i,j-1,t} - 1) = 0 \quad (6)$$
$$S_{i,j,t}(x_{i,j+1,t} - x_{i,j,t} - 1) = 0 \quad (7)$$

where S is the saliency matrix of Eq. (1), with explicit time index. Note that the equation looking right from pixel $(i,j-1)$ can be combined with the equation looking left from pixel (i,j) to form one equation:

$$(S_{i,j-1,t} + S_{i,j,t})(x_{i,j,t} - x_{i,j-1,t} - 1) = 0 \quad (8)$$

Boundary substitutions In order to make the new image fit in the new dimensions we add a constraint defining the first pixel in each row of each frame $(i,1,t)$ to be mapped to the first row in the retargeted video, i.e., $\forall i, \forall t\, x_{i,1,t} = 1$. Similarly, the last pixel of each row is mapped to the boundary of the remapped frame: $\forall i, \forall t x_{i,C_{Width},t} = C_{TargetWidth}$.

The mapping of the first and last pixel in each row is known, we therefore eliminate the associated variables with the actual values, wherever $x_{i,1,t}$ or $x_{i,C_{Width},t}$ appear in Eq. (8).

Spatial and time smoothness. It is important to have each column of pixels in the input image mapped within the boundaries of a narrow strip in the retargeted image. Otherwise, the image looks jagged and distorted . These type of constraint are weighted uniformly, and take the simple form of:

$$W^s(x_{i+1,j,t} - x_{i,j,t}) = 0 \quad (9)$$

[2] More precisely, since we solve in a least-squares manner an equation arising from a pixel of importance $S_{i,j,t}$ is as influential as $\frac{S_{i,j,t}^2}{S_{i',j',t'}^2}$ equations arising from a pixel of importance $S_{i',j',t'}$.

In our system $W^s = 1$. In order to prevent drifting, we also add a similar restriction that constraints the last pixel of each column to have the same displacement as the first one in the column.

$$W^s(x_{1,j,t} - x_{C_{Height},j,t}) = 0 \quad (10)$$

Preserving the mapping function continuity between adjacent frames is necessary, as is stated in the following constraint Fig. 4:

$$W^t(x_{i,j,t} - x_{i,j,t-1}) = 0 \quad (11)$$

4.1. Speed optimization

Computing a per-pixel location mapping function is slow and requires large amounts of memory, instead we represent the input frame using a mesh, then we compute an optimally deformed mesh Fig. 5, such that we practically solve the optimization system on a down-sampled problem. Next we compute the per-pixel mapping function by interpolation of the mesh. Further more, we can use the low-resolution mesh to perform the forward mapping on the GPU, saving precious CPU processing time. We were able to achieve further performance boost by down-sampling the height by a much larger factor than the width (in virtue of the constraint limiting the vertical movement Eq. (10)). Practically, we down-sampled the width by a factor of 16 while the height is down sampled by a factor of 48. An image of 640 × 480 pixels is thus solved by sampling only 40 × 10 pixels. On a Core2 Duo 2.4Ghz we were able to solve these equation systems in 4–10 ms (C++code using UMFPACK [21]). For comparison we refer to [13], they state that a 400 × 300 pixels video having 400 frame, on similar hardware, will take between 10 and 20 min, or 1500–3000 ms per frame.

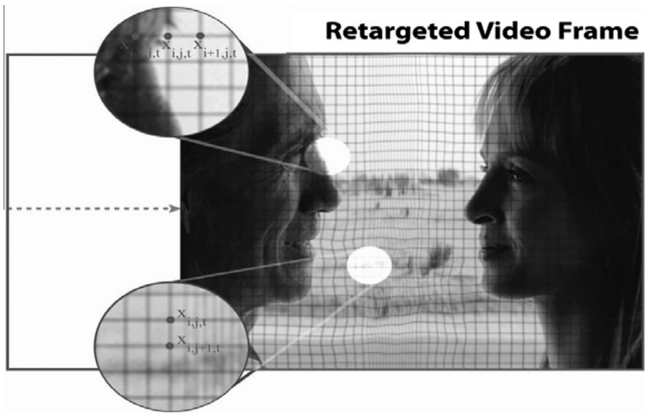

Fig. 5. Frame from a video with a mesh grid on top. The deformed mesh in the middle of the image reviles the retargeting process.

4.2. off-line shot-by-shot solution

The online system proposed above may produce suboptimal solution on complex video scenes. Consider a scene where an object moves across the frame from one side to the other, for example a man crossing the road, or an airplane flying across the sky. An online architecture is somewhat limited to the solution found at the beginning of the shot, in the example above the object of interest moves across the shot, thus damaging the visual performance of the retargeting process.

To overcomes these kind of complex scenes, an off-line (per shot) solution can be used Fig. 6, which solves for the displacement of the entire shot at once. Notice that replacing the per-frame solver with a per-shot solver increases the number of variables in a T times factor (where T is the number of frames in the shot).

An online architecture solution, treats each frame as a stand-alone problem with the addition of a time smoothness constraint to create a continues flow between consecutive frames. The time constraint allows only slight changes in the retarget solution between consecutive frames Eq. (11) thus, limiting our global retargeting solution. In our example, since in the first frame the salient object is on one side of the frame, and in the last frame the salient object is in the opposite side, our solution will be sub-optimal, since it cannot adapt to the immense change in the shot. With the introduction of the global solution solver, we see how the globally retargeted video performs as a cropped

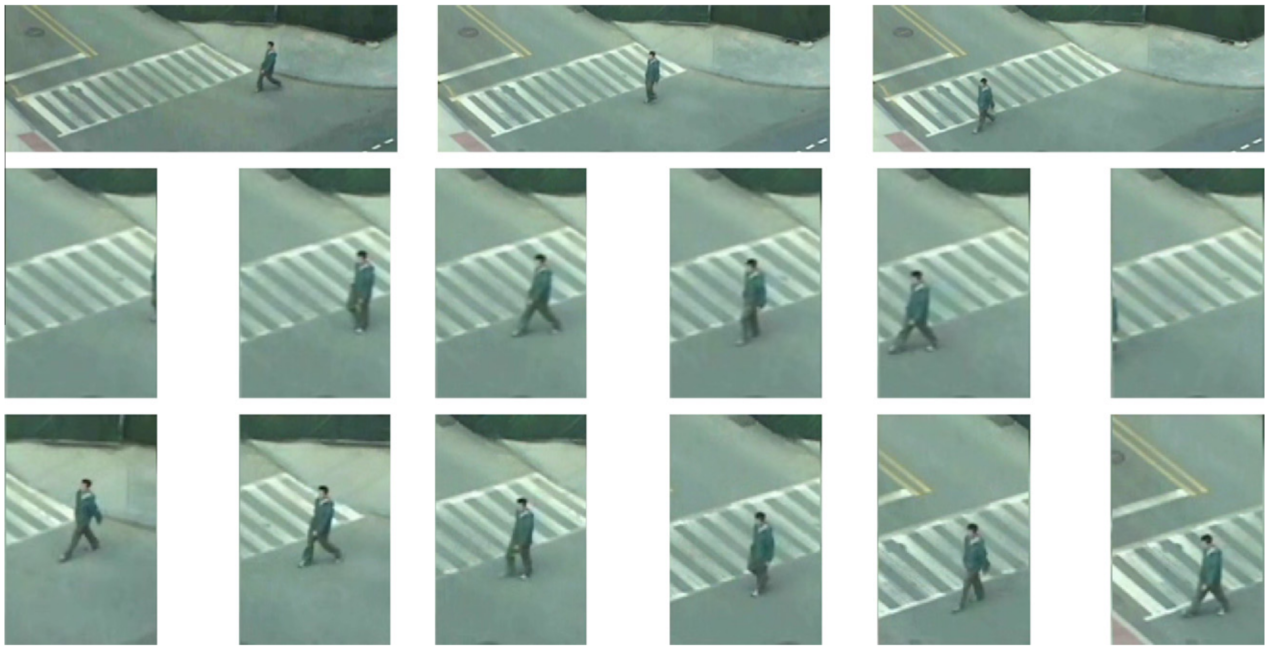

Fig. 6. Top: Original video from a surveillance camera. Middle: Retargeted video to quarter-width using a global solution. Bottom: Retargeted video to quarter-width using a global solution with a pan like camera motion. The videos are available in the supplemental material [19].

window in the middle of the frame, therefore capturing more than half of the mans' walk. This in contrast of the online solver which will be "stuck" on the right side of the frame due to the time-smoothness constraint.

The constraints involved in the per-shot linear system are the same as in the per-frame system, where in the per-shot system the time constraint becomes:

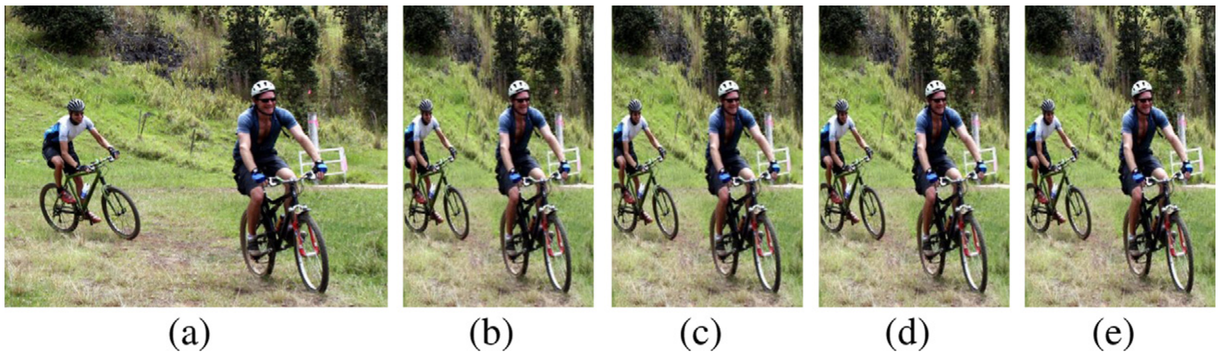

Fig. 7. Retargeting example. (a) original, (b) retargeted frame, notice the utilization of the frame area, for obvious reasons it cannot be achieved using an optimal cropped window. (c) retargeting with added saliency noise (uniform distribution, variance 0.05); (d) noise: uniform distribution, variance 0.10; (e) noise: uniform distribution, variance 0.15.

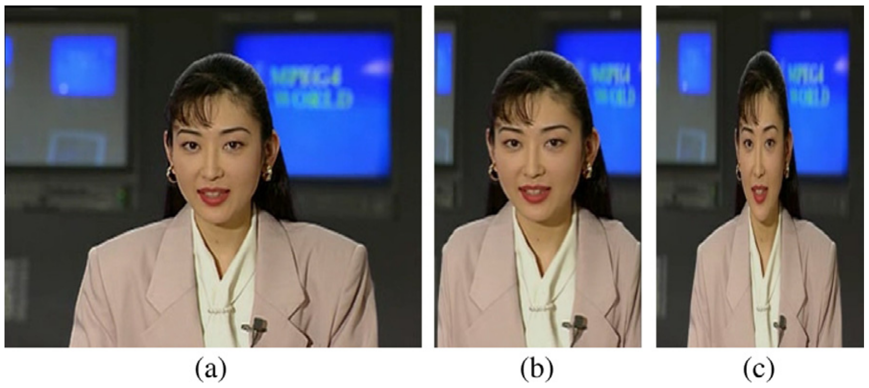

Fig. 8. Retargeting example. (a) original frame from the ITU-T video sequence "Akiyo"; (b) the downsized frame achieved by our method; (c) a conventionally down-sampled frame. See also supplemental material [19].

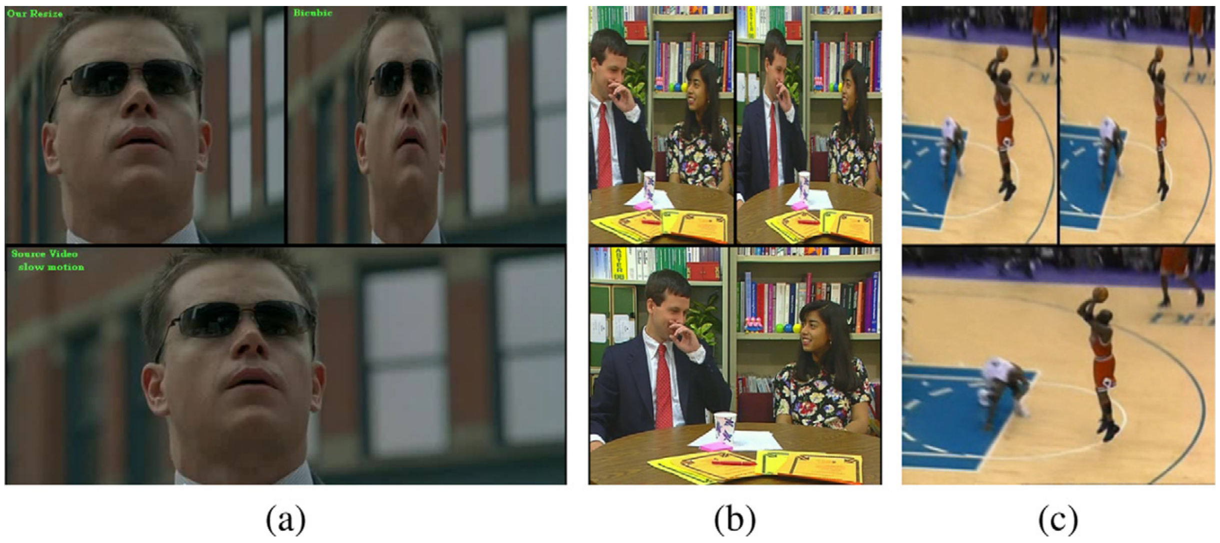

Fig. 9. (a) Retargeting a frame taken from a motion picture; (b) Retargeting a frame taken from the MPEG committee standard benchmark video "paris"; (c) Retargeting a frame taken from a basketball video. The original frame is shown at the bottom of each triplet, and a bicubic interpolation is shown on the top-right. Our retargeting method (top left of each triplet) prevents much of the thinning effect of rescaling and preserves the visual details. Please refer also to the supplemental material [19].

$$W^t(x_{i,j,t} - x_{i,j,t-1}) = 0 \quad (12)$$

that is, the position in the previous frame $x_{i,j,t-1}$ becomes an unknown.

4.3. Video model-camera motion

The warping operator induces "camera motion" by shifting the entire frame a few pixels at a time. It is desirable to employ this motion in a way which imitates the ego-motion of an actual

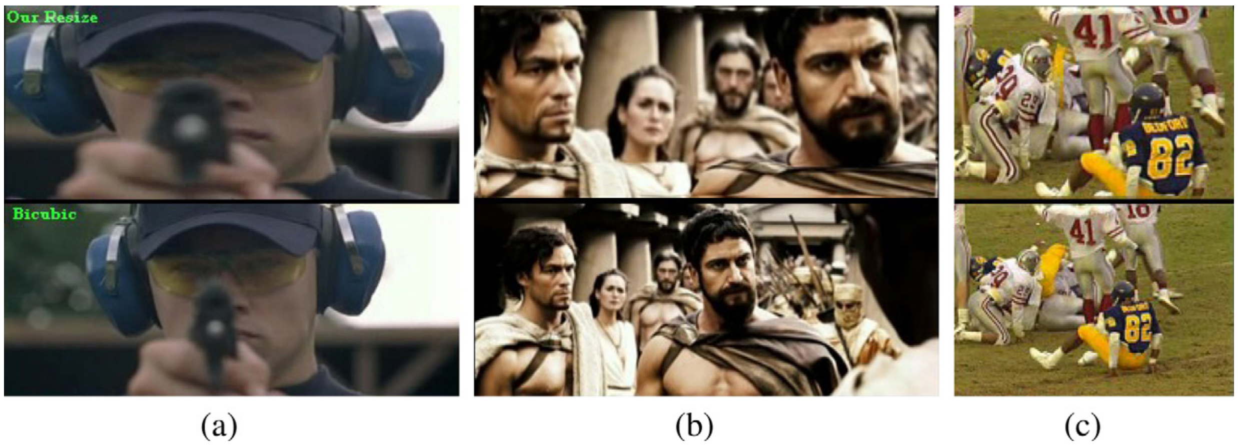

Fig. 10. Three down-sizing results. A bicubic interpolation is shown at the bottom of each pair. Our retargeting method (top) applies a non-homogenous zoom on the objects of interest.

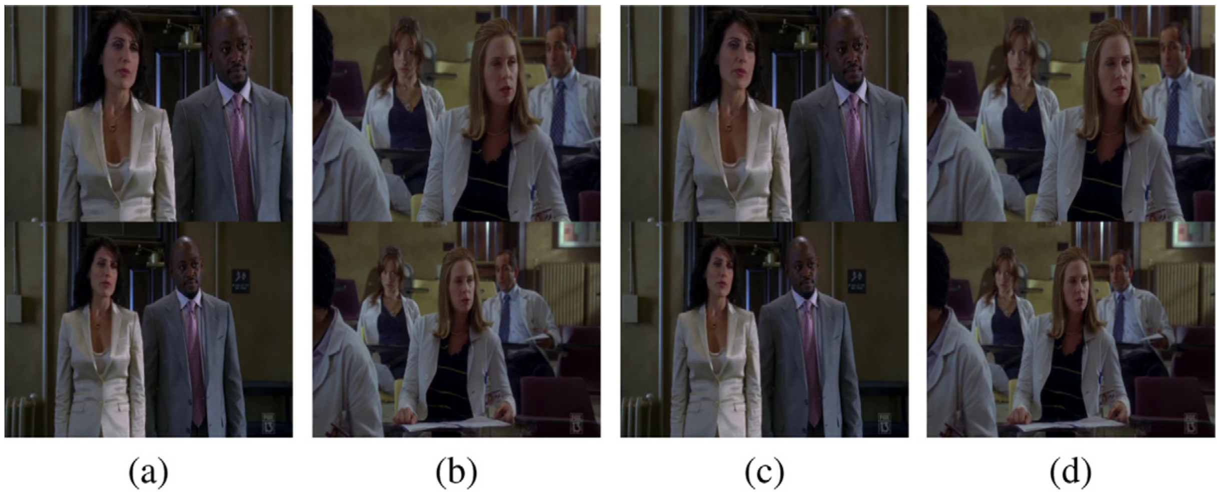

Fig. 11. Four down-sizing results from a TV series. A bicubic interpolation is shown at the bottom of each pair. Our retargeting method (top) applies a non-homogenous zoom on the objects of interest, see also the supplemental material [19].

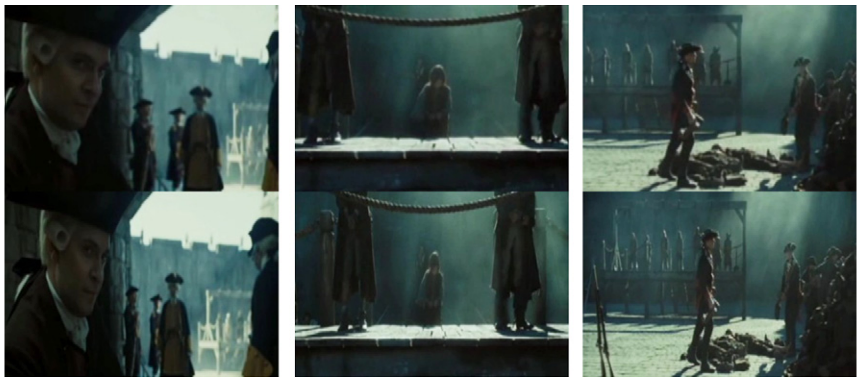

Fig. 12. Three down-sizing results from a movie. A bicubic interpolation is shown at the bottom of each pair. Our retargeting method (top) applies a non-homogenous zoom on the objects of interest. The video shots are available in the online supplemental material [19].

camera, while retaining the salient objects inside the visible frame. To simulate a pan/tilt motion, we add one translation D_t variable per frame t. When retargeting on the x-axis, the variable D_t allows a pan like motion between two consecutive frames, for retargeting the y-axis a tilt like motion is induced. [16] use temporal coherence preservation on every mesh-vertex on their grid, we however use one variable for the entire reposition of the cropped and warped window. In both cases the reposition variable is part of the global optimization scheme and is found automatically, uniquely, our approach eludes the need for an optical flow module.

We sustain the time smoothness by limiting the translation between two consecutive frames using a new set of soft constraint Eq. (13), to produce a coherent output video. The rest of the constraints remain the same as in the per-shot system:

$$\begin{aligned}
&\forall_{1\leqslant i\leqslant\ C_{Height}} x_{i,1,t} = 1 + D_t \\
&\forall_{1\leqslant i\leqslant\ C_{Height}} x_{i,C_{Width},t} = C_{NewWidth} + D_t \\
&D_t - D_{t-1} = 0 \\
&W^t(x_{i,j,t} + D_t - x_{i,j,t-1} - D_{t-1}) = 0
\end{aligned} \tag{13}$$

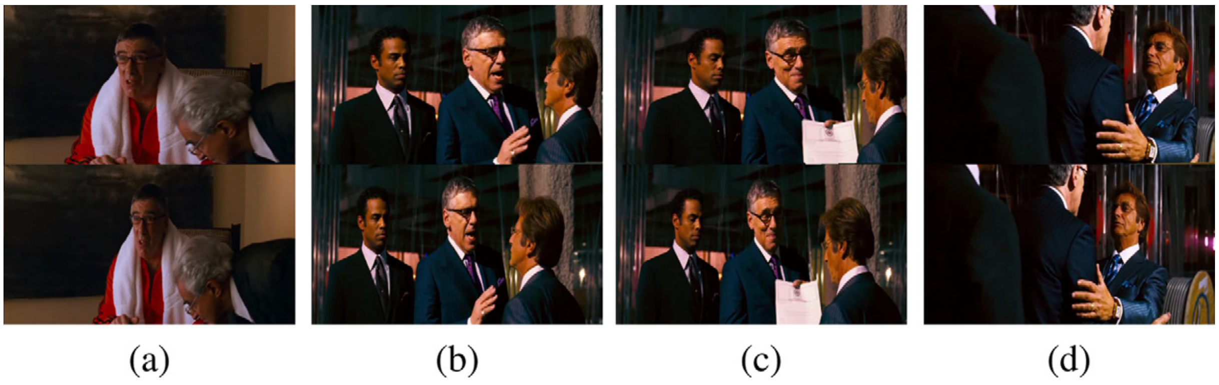

Fig. 13. Four down-sizing results from a movie. A bicubic interpolation is shown at the bottom of each pair. Our retargeting method (top) applies a non-homogenous zoom on the objects of interest, see also [19].

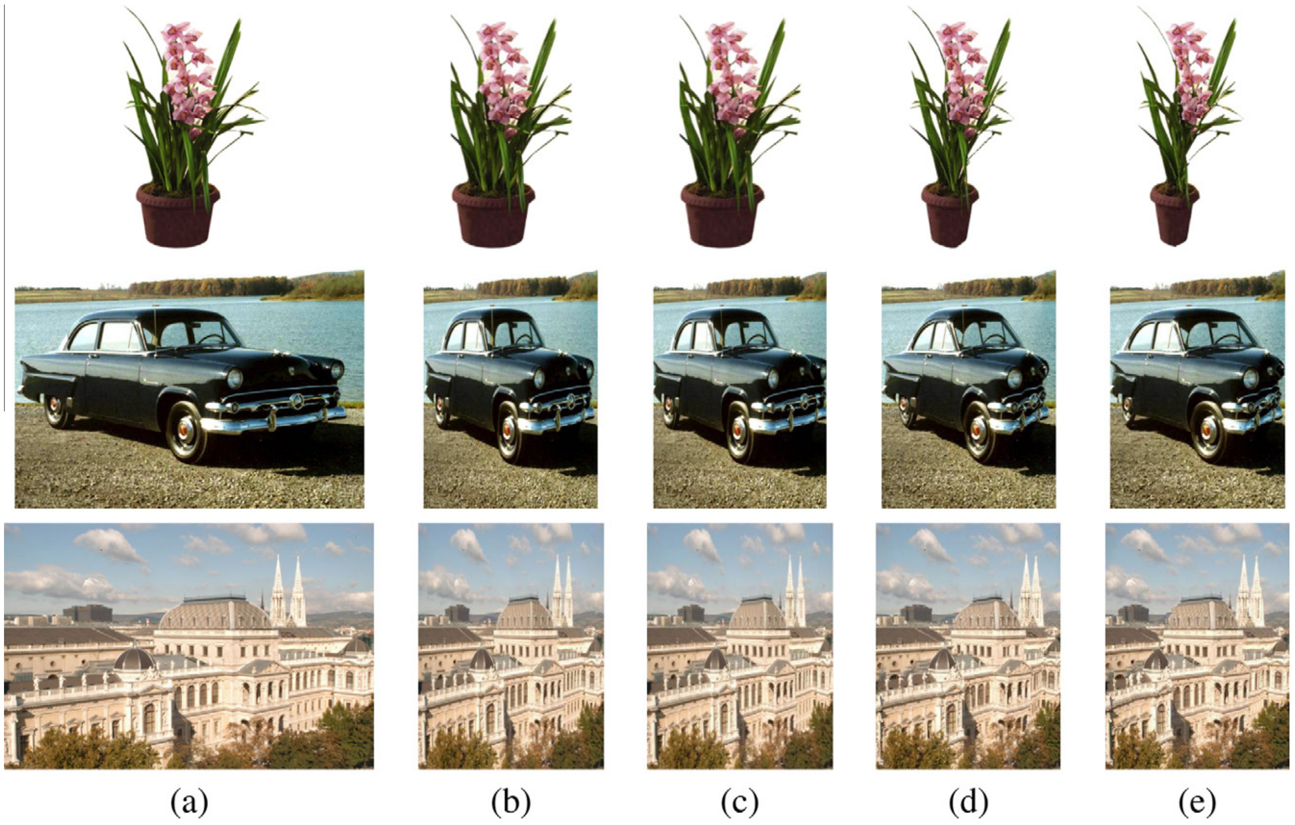

Fig. 14. Comparison with [12] Seam Carving technique. (a) Original, (b) our retarget using L1-norm to half the width, (c) retarget using L2-norm, (d) Seam Carving using L1-Norm, (e) Seam Carving using L2-Norm. Our method is more robust to the selection of the Saliency measure (subtle difference between (a) and (b)), and because of its continuous nature, creates less jagged lines (note the jagged lines in the Ford image and on the buildings on columns (d) and (e)). The advantages of optimizing the entire mapping at once are also apparent when examining the flowerpot.

5. CAVM-Applications and results

The applications of our CAVM system are demonstrated next. First we review our results on video retargeting and compare it to state-of-the-art in the image/video retargeting field. We also demonstrate video expansion and real-estate creation. The applicability of the CAVM system is exhibited on other assorted tasks, such as video abstraction, object removal and displacement (both

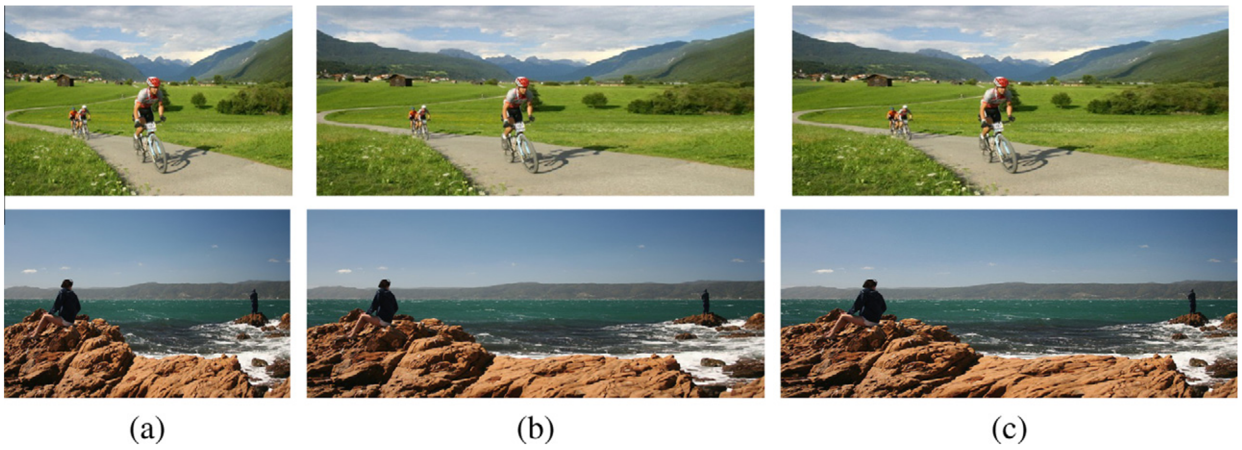

Fig. 15. (a) Original image. (b) Enlarged image Rubinstein et al. [13]. (c) Our retargeting method.

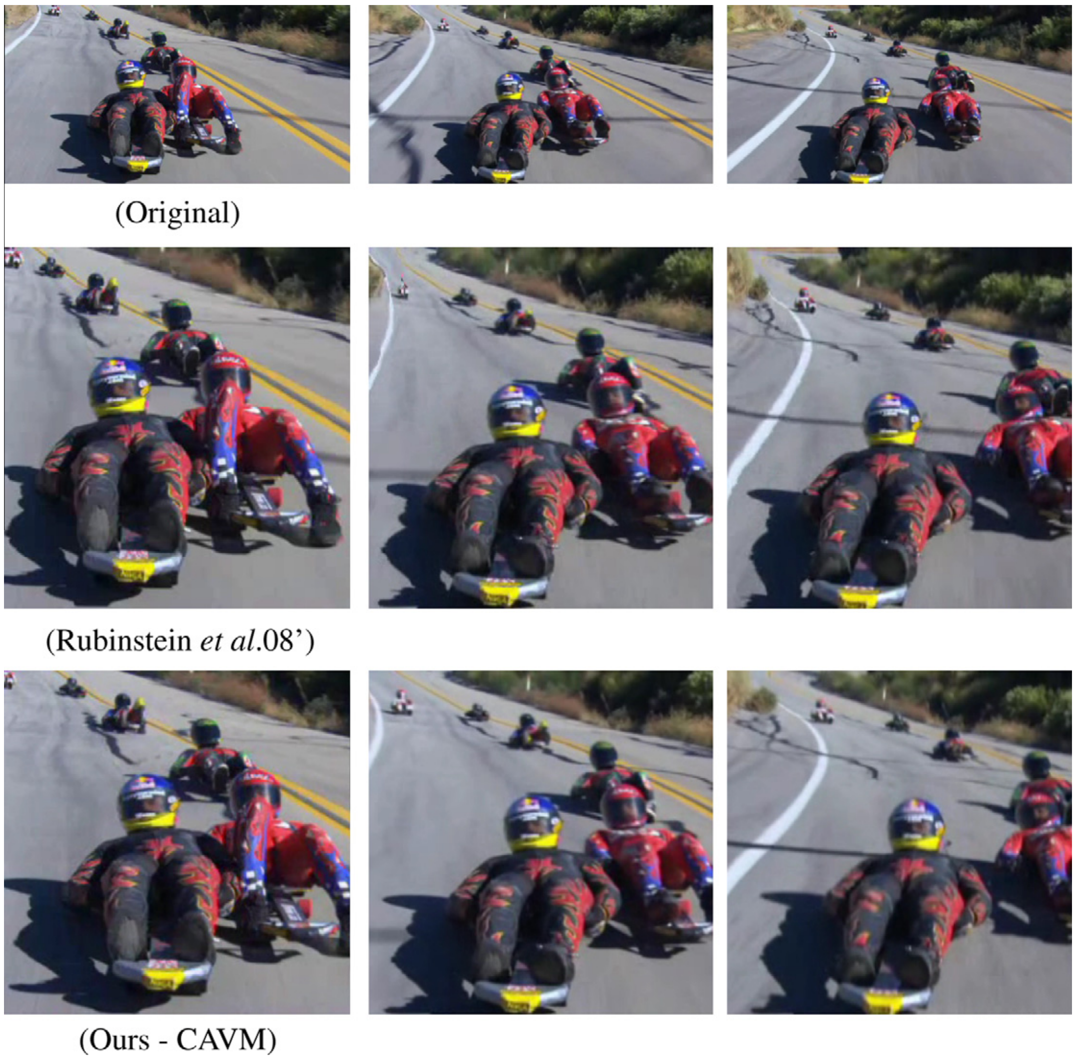

Fig. 16. Comparison with [13] Video Seam Carving technique. Frames (1,19,40) from the video "RoadSki" used by [13]. Observe the distortion of white lines and the disappearance of the rider in the background in [13] retargeted video. Please refer to the online supplemental material [19].

spatial and temporal planes), and freeform image/video boundary optimization.

5.1. Video retargeting, results and comparison

We have experimented with our algorithm on a large number of videos, and tested several video retargeting applications.

Altering the aspect ratio. Example videos for the task of altering the aspect ratio are shown in Figs. 9, 11, 12 (see the accompanied supplemental material for retargeted videos [19]). The format of the retargeted videos is as follows: each frame is divided into three sub frames. The bottom one is the original video frame. The top right sub-frame is the result of applying bicubic interpolation to obtain a new frame of half the input width. The top-left figure is the result of our retargeting algorithm.

Note that while our algorithm does not explicitly crop that frame, whenever the unimportant regions in the frame lie away from the frame's center, an implicit cropping is applied. See for example the retargeting results in Fig. 8. Many of the pixels at the left and right sides of the input frames are mapped to the first and last few columns of the retargeted frames, hence disappear.

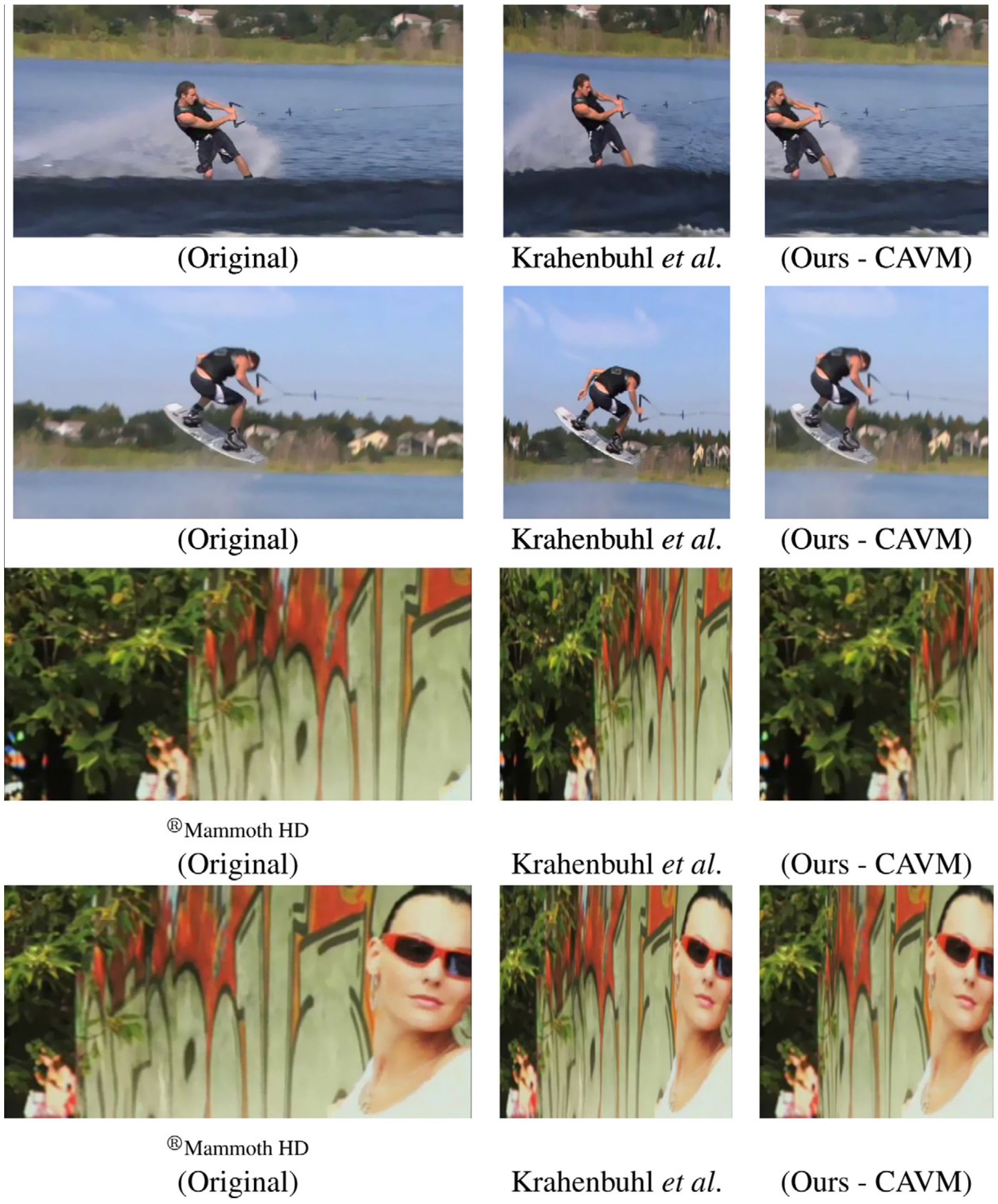

Fig. 17. Comparison with Krahenbuhl et al. [14]. Please refer to the supplemental material [19].

A real-time video retargeting performance (25fps) is achieved on an Intel Core 2 Duo 2.4Ghz, running Windows XP SP2, by C++standalone software that incorporates UMFPACK (sparse solver) with a GPU based implementation to perform the forward mapping.

Saliency measure One of the biggest issues with saliency based image/video retargeting is the selection of the saliency function. We demonstrate the robustness of the optimization model to auxiliary noise in the saliency function, see Fig. at>7–9. As is evident in this experiment, when adding to the saliency score white noise of variance up to 0.15, the visual quality remains adequate.

Down-sizing results Examples of down-sampling while preserving the aspect ratio, are shown in Figs. 10 and 1312,13 and supplemental material [19]. The *x* and the *y* warps were computed independently on the original frame and then applied together to create the output frames. As can be seen, there is a strong zooming in effect in our results, as necessitated by the need to display large enough objects on a small LCD-screen.

Comparison with Seam Carving [12,13] Fig. 14 compares our image retargeting method with the one of [12] while Fig. 16 compares our video retargeting method with the one of [13]. We emphasize the speed differences between the two methods, Rubinstein et al. indicate a 400 × 300 pixels video takes between 1.5 s and 3 s (excluding saliency computation), while we optimize a similar size frame in 4–10 ms. We feel that in both image and video retargeting our method is less susceptible to artifacts and produce smoother visual results.

Comparison with optimization based video retargeting [14,16,15] Fig. 17 compares our video retargeting method with the one of [14] while they do have temporal constraints their system lacks the ability to create a panning like window, and as such cannot adapt to fast changing shots. Fig. 18 compares our video retargeting method with the one of [15] and [16]. Both [15] and [16] employ an ego-motion module that allows the retargeted video to adapt to fast changing scenes. It is interesting to see that even

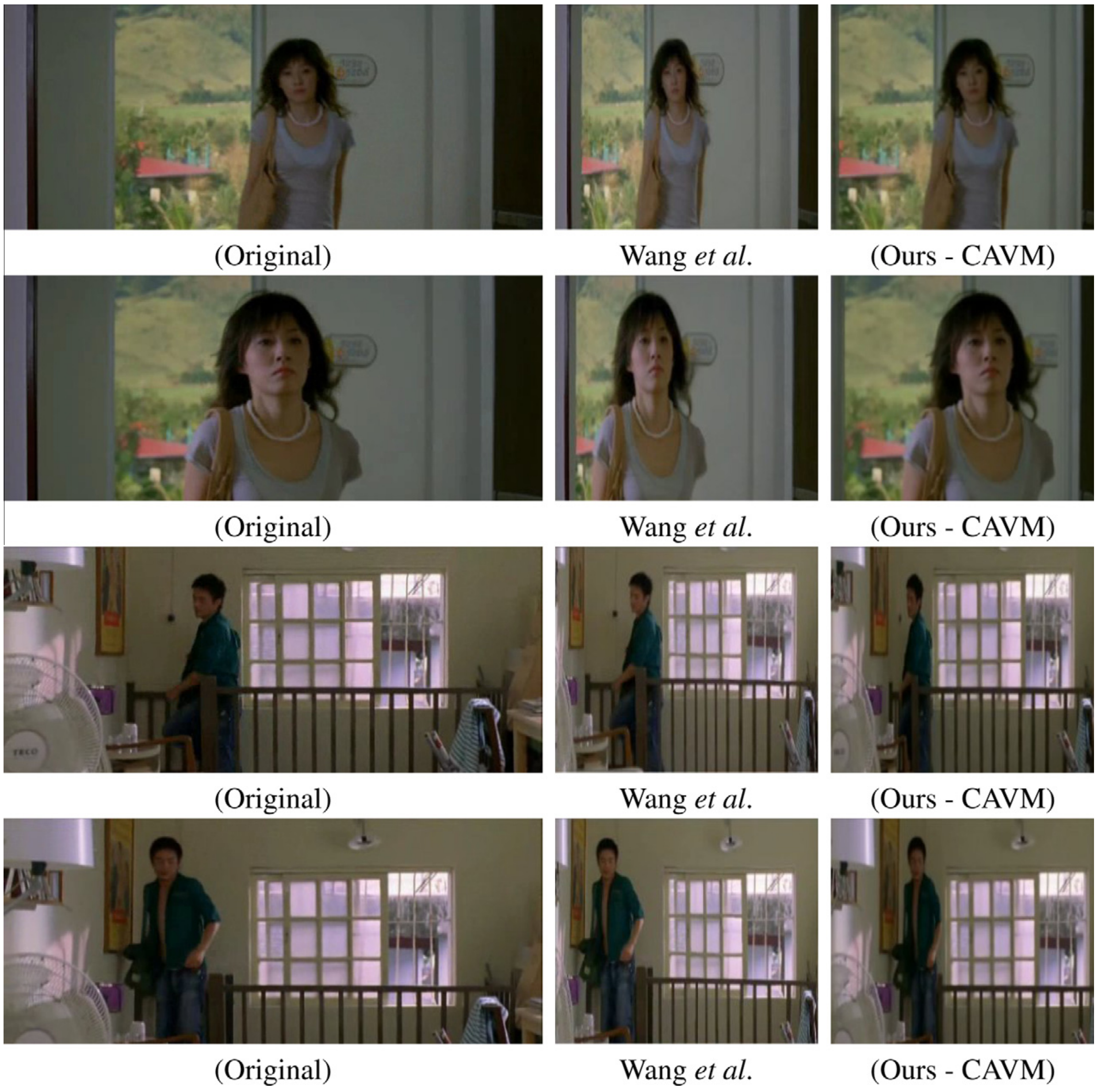

Fig. 18. Comparison with Wang et al. [15] and [16] respectively. The video shots are available as part of the supplemental material [19].

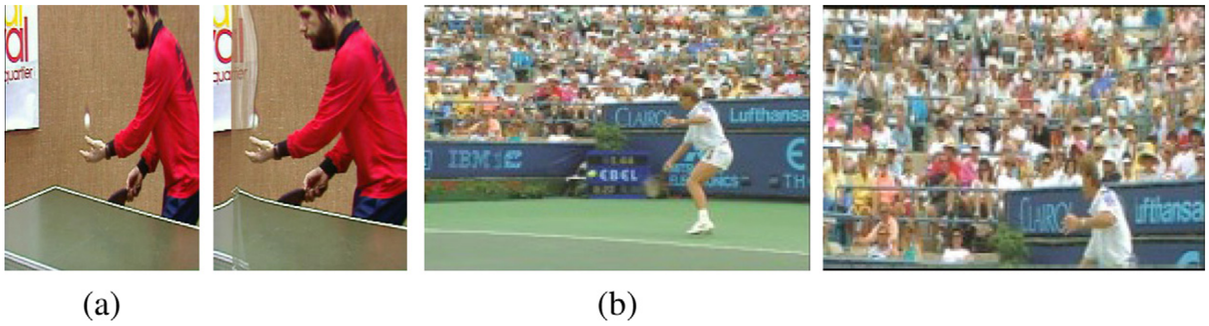

Fig. 19. Failure cases. (a) Frame 55 from the ITU-T video "tennis". (left) Bicubic sampling to half width. (right) Our solution. The artifacts were caused by the zoom-out camera motion. (b) A tennis sequence. (left) Bicubic interpolation to half width and height. (right) The crowd part of the frame has many large-magnitude gradients that tilted our solution.

though our system does not incorporate a camera-motion module, it is able to create the same panning window effect and adapt to the changing scene.

5.2. failure cases

The saliency model we employ and the optimization may be "fooled". An example of a failure case due to the optimization stage is evident in Fig. 19a. The camera zoom-out motion creates a conflict between the need to preserve time smoothness and the need to update the salient region. This results in a noncontinuous warp, creating a visible artifacts. This phenomena can be detected by analyzing the warp or using a dedicated zoom-out motion detector, when occurs, the continuity constraints (Eq. (11)) should be relaxed.

Another flaw, one which demonstrates the limitations of our importance-measure is revealed in Fig. 19b. High energy textures in the crowd area endorse the algorithm to give it more space in the retargeted frame, thus taking cardinal frame area from the tennis court. Similar problems are solved by incorporating other object recognition techniques to the saliency score and by creating genre-optimized scores.

The term "global motion" is used to describe occurring 2D image motion caused by camera motion. Estimating accurately the affine global motion is a classical task, for which efficient algorithms exist. Currently, we chose not to incorporate it into our system. However, having a black-box global motion estimator, it is fairly easy to do so. Assuming a global affine motion M_G has been estimated, it can be added to the previous frame warping result, thus compensating for the camera motion. Then the optimization problem is solved similarly to the stationary model.

5.3. Media expansion

Video expansion is obtainable with the help of our system. In such a case, however, the desired output depends on the application. In one application, which is the one considered by [12] and [13] for stills, the task is to keep the original size of the salient objects, while enlarging the images by filling-in the less salient areas with unnoticeable pixels. For such a task, our algorithm can work with mild modifications Eq. (14). Fig. 15 demonstrates and compares [13] image inflation by a fixed factor of one and a half on the width while the height remains unchanged.

A related task is foreground emphasis through non-homogenous warping in-place, where the dimensions of the video remain the same (salient objects are increased in size on the expense of less salient regions). To apply our method in these cases, we need to alter Eq. (8) to have the preferred inflating ratio on the right-hand-side. If this desired inflation is given by the user or by some heuristic, this is a simple modification.

$$S_{i,j,t}(x_{i,j,t} - x_{i,j-1,t} - desiredScale) = 0 \qquad (14)$$

5.4. Video synopsis

Video synopsis aims at taking a step toward sorting through video for summary and indexing and is especially beneficial for surveillance cameras and web-cam. Classical video summarization techniques have focused on selecting a subset of representative key-frames or, alternatively, a collection of short video cuts. Current research tendency is to compose new frames from the source video [22–24]. This current trend is sometimes coupled with mosaicing techniques to mitigate camera motion, and some of the most recent contributions aim at generating interactive mosa-

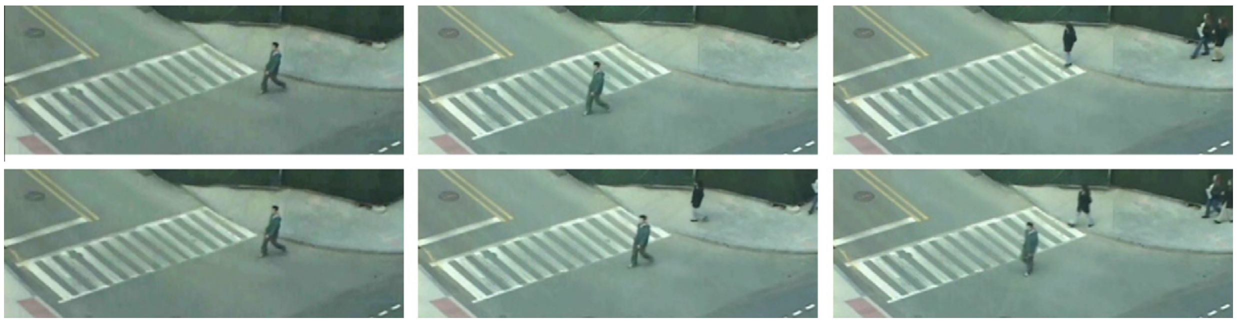

Fig. 20. (top) Source frames from a surveillance camera video. (bottom) Frames from the time-retargeted output video.

ics for quick and layered indexing of video content [25,26]. These methods generate key-frames or key-clips and create a seamless mosaic of these clips using different texture-synthesis approaches. Below, we focus on generating the summarized clips, which can be coupled with independent mosaicing methods.

Given a video of a certain length the task is to create a new video of a much shorter length. We propose to tackle this task in a similar manner to our retargeting method, i.e. to compute an optimal per-pixel time warping via a linear system of equations. The first and last frames on the input are to be mapped to the first and last output frames. Each pixel in the input video will be mapped to a location in the output video that is similar to that of its spatial neighbors, where important pixels are mapped to distinct locations from their time-line neighbors.

We consider a video as a 3D spatio-temporal cube. Each pixel is mapped to a new temporal location. Pixels of salient regions should stay together (both on X and Y axis) and the target time is set by constraining the last frame to be mapped to time $C_{TargetTime}$. The following equation system is used to solve this time-retarget problem. Video synopsis results are shown in Figs. 20 and 21.

$$\begin{aligned}
&\forall_{1\leqslant i\leqslant\ C_{Height}}\forall_{1\leqslant j\leqslant\ C_{Width}}T_{i,j,1}=1\\
&\forall_{1\leqslant i\leqslant\ C_{Height}}\forall_{1\leqslant j\leqslant\ C_{Width}}T_{i,j,C_{Duration}}=C_{TargetTime}\\
&W^s(T_{i,j,t}-T_{i+1,j,t})=0\\
&W^s(T_{i,j,t}-T_{i,j+1,t})=0\\
&S_{i,j,t}(T_{i,j,t}-T_{i,j,t-1}-1)=0
\end{aligned}\tag{15}$$

Stressing the limits of the video synopsis solution is evident in Fig. 22, where we stretched our solution to its limits. The number of output frames required from the system is too little for this complex video, and the outcome is a 'ghosting' effect on the summarized video output. Future improvement of the system could be in the form of visually pleasing stitching mechanism, maybe similar to the one introduces by [26].

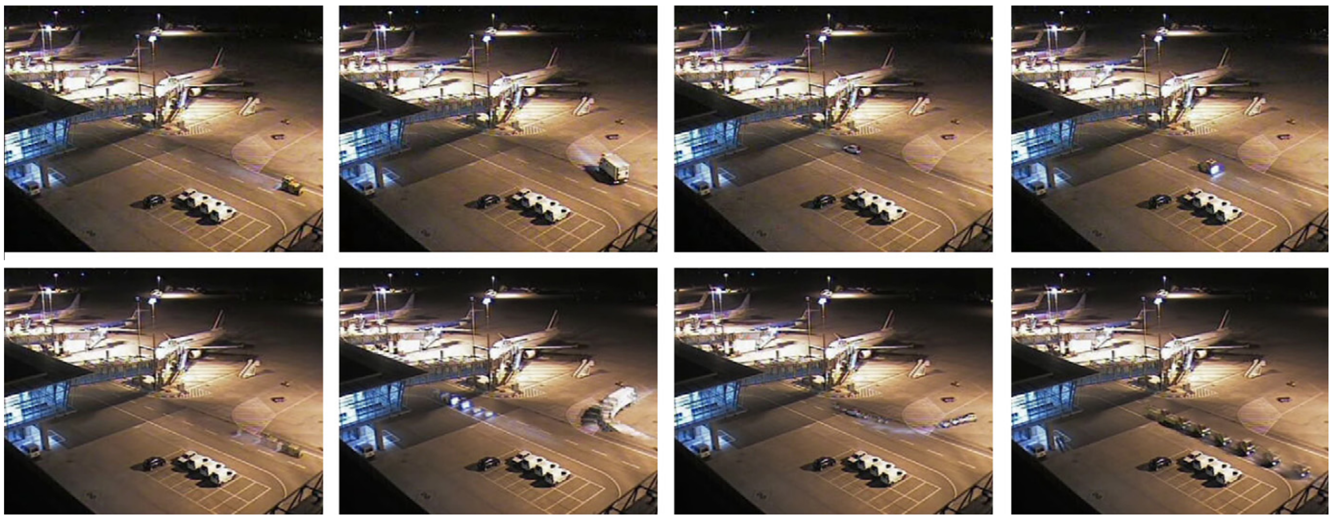
Fig. 21. (top) Source frames from a surveillance camera video. (bottom) Frames from the time-retargeted output video.

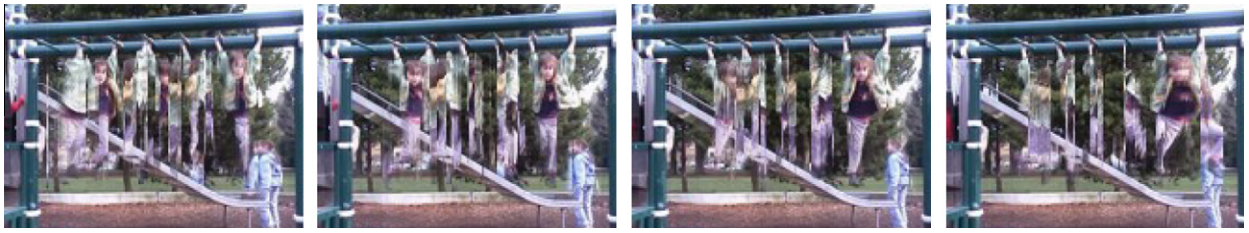
Fig. 22. Four frames out of the summarized 16 frames produced for the video "Lena", originally 242 frames.

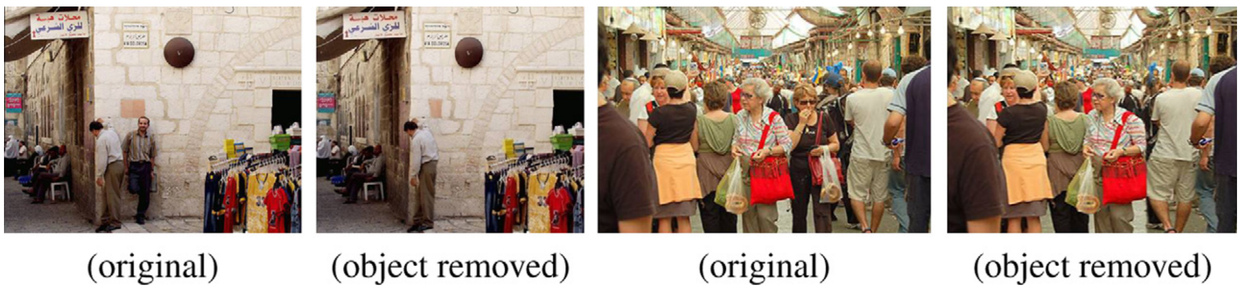

Fig. 23. Object removal. On the left pair the standing man is removed. On the right pair, a woman is removed.

5.5. Object removal

The next task we consider is the removal of a selected object from an image or a video sequence, see Fig. 23. First, the object's width is estimated by: $C_{ObjectWidth} = ArgMax_i \sum_{j=1}^{C_{Width}} M_{i,j}$. Then, the following equation system is formed, where $M_{i,j}$ is a binary pixel map, indicating whether the object is to be removed.

$$\begin{aligned}
&\forall_{1\leqslant i\leqslant C_{Height}} x_{i,1} = 1\\
&\forall_{1\leqslant i\leqslant C_{Height}} x_{i,C_{Width}} = C_{Width} - C_{ObjectWidth}\\
&(1 - M_{i,j})S_{i,j}(x_{i,j} - x_{i,j-1} - 1) = 0\\
&M_{i,j}W^r(x_{i,j} - x_{i,j-1}) = 0\\
&W^s(x_{i,j} - x_{i+1,j}) = 0\\
&W^s(x_{1,j} - x_{C_{Height},j}) = 0
\end{aligned} \tag{16}$$

where W_r is a large weight to ensure complete removal.

The spatial object removal can be transformed into temporal object removal by having removal constraints of the form $M_{i,j,t} W^r(x_{i,j,t} - x_{i,j,t-1}) = 0$.

5.6. Object displacement

On occasion, ones needs to relocate an object in within an image whilst minimizing the distortion caused by a standard mesh stretching. For this purpose we revised the retargeting scheme to allow relocation of a selected object. Let $M_{i,j}$ be a binary per-pixel selection mask, then the solution of the following equation system shifts the object by D^X pixels (Fig. 24). Of course relocating an object in within a video is also possible, see for example Fig. 25 and online supplemental material [19].

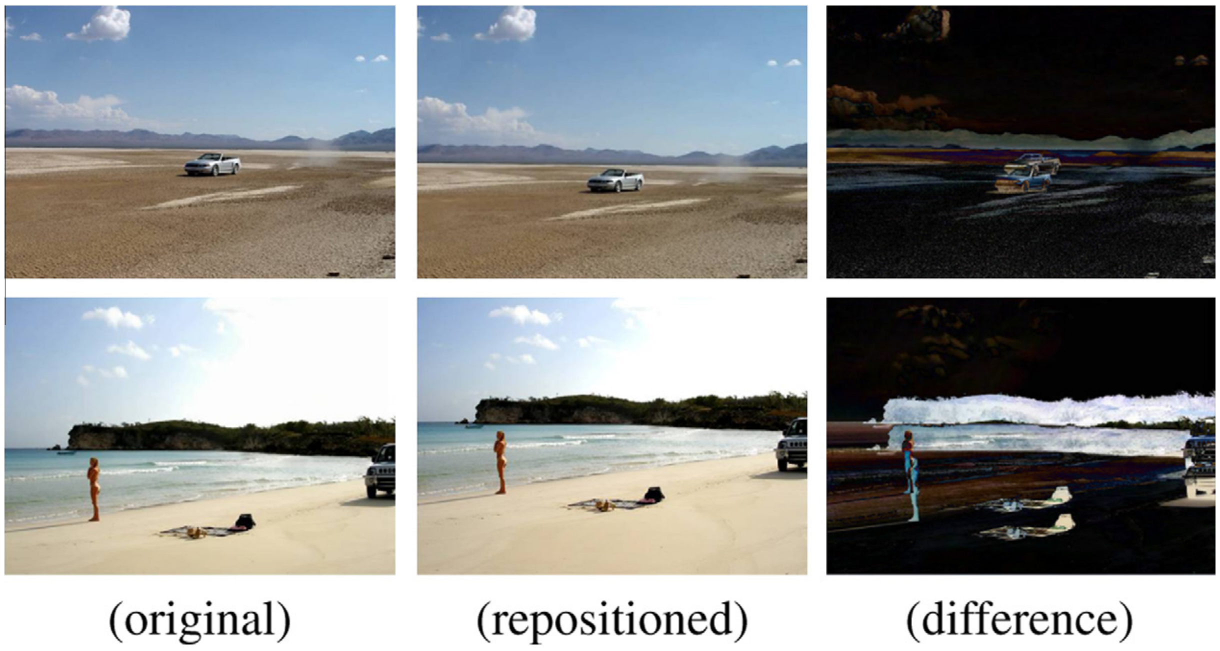

Fig. 24. In the top image the car is repositioned. In the bottom image the girl is repositioned.

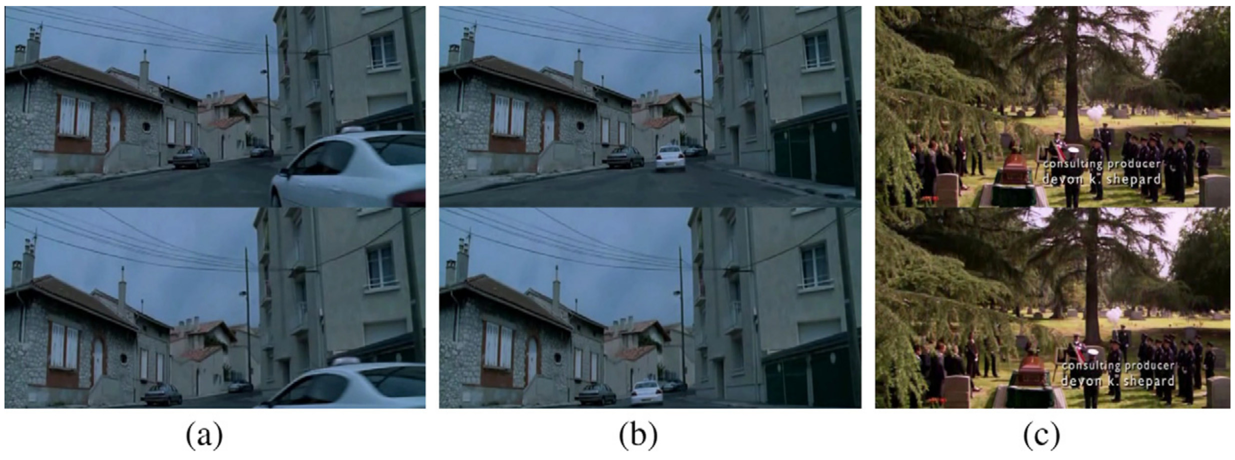

Fig. 25. (a and b) Top: frames from a movie. Bottom: The parked car is moved downwards and to the left, while avoiding visual distortion. (c) Top: a frame from a TV series. Bottom: we moved the tree to right nonetheless the video remains undistorted. Video content is available online [19].

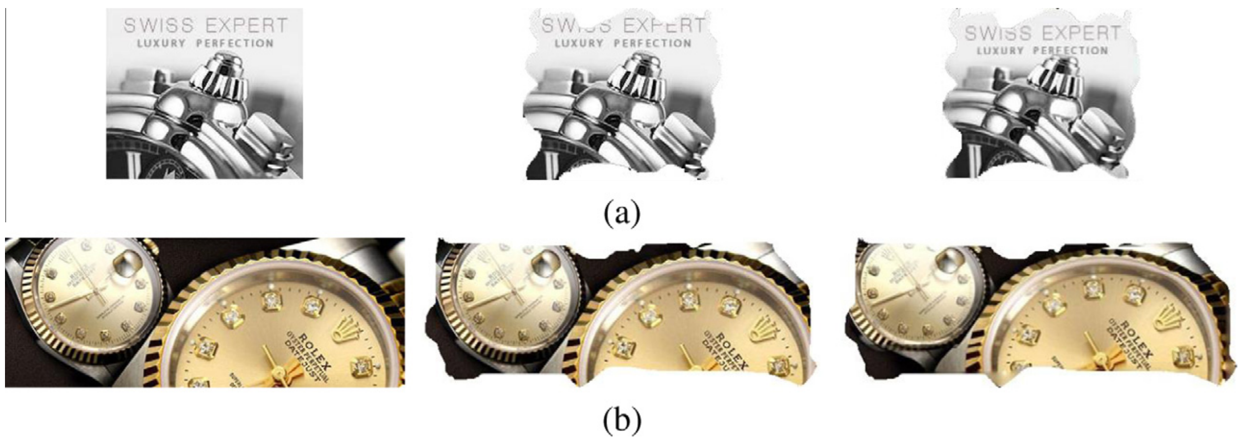

Fig. 26. Left: original image. Middle: non-rectangular mask. Right: Optimized image. (a) Notice that on the optimized image the title is more apparent. (b) In the optimized image, the two watches are squeezed towards one another making the title and indicator visible.

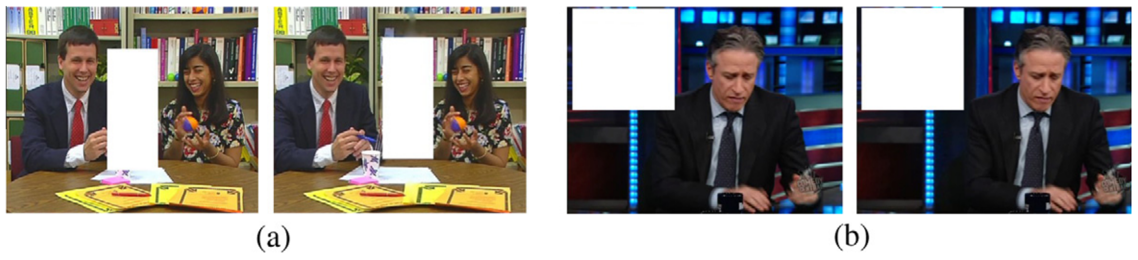

Fig. 27. Left: original image. Advertisement real-estate marked by a white rectangle. Right: Optimized real-estate. (a) Single frame from the MPEG committee video "Paris". (b) Single frame for the TV show "The daily Show".

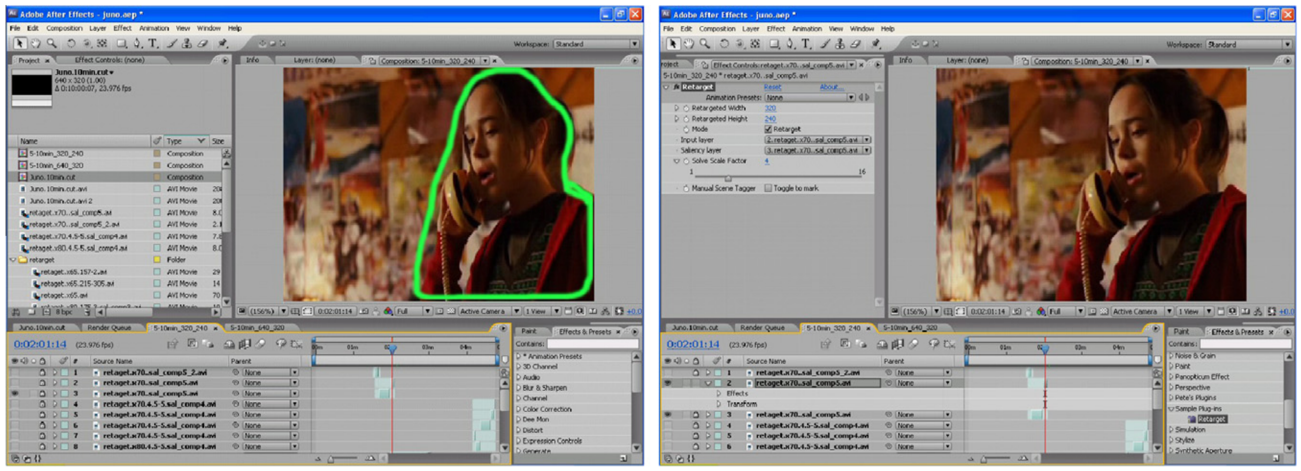

Fig. 28. A screenshot of our "Adobe® After Effects" plugin in action. Left: The bright green area marks the salient object. Right: the retargeted video output. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

$$
\begin{aligned}
&\forall_{1\leqslant i\leqslant C_{Height}} x_{i,1} = 1\\
&\forall_{1\leqslant i\leqslant C_{Height}} x_{i,C_{Width}} = C_{Width}\\
&(1 - M_{i,j})S_{i,j}(x_{i,j} - x_{i,j-1} - 1) = 0\\
&W^r M_{i,j}(x_{i,j} - (j + D^X)) = 0\\
&W^s(x_{i,j} - x_{i+1,j}) = 0\\
&W^s(x_{1,j} - x_{C_{Height},j}) = 0
\end{aligned}
\tag{17}
$$

5.7. Image boundary optimization

In the advertisement world, "real-estate" is a free area in within an image, or a video, where an ad can be located. Advertisements are commonly a rectangle banner (or "flash" object) located on a vacant space of the web page. Unfortunately vacant real-estate on images and video are generally non-rectangular. We present a unique approach, mapping the rectangular advertisement to a non-rectangular area. We demonstrate how to optimize an ad for shapeless areas, in which we first optimize horizontally, and then apply the same scheme for the vertical optimization. Let L_i be the left border, i.e. $L_i = d_i$ implies that the left pixel in row i should be mapped to location d_i (on the X-axis). The right border R_i is defined accordingly. The following equation system solves the above optimization problem.

$$\begin{aligned}
&\forall_{1\leqslant i\leqslant\ C_{Height}} x_{i,1} = L_i\\
&\forall_{1\leqslant i\leqslant\ C_{Height}} x_{1,C_{Width}} = R_i\\
&S_{i,j}(x_{i,j} - x_{i,j-1} - 1) = 0\\
&W^s(x_{i,j} - x_{i+1,j}) = 0\\
&\forall_{j s.t. j>max(L_{j'})\wedge j<min(R_{j''})} W^s(x_{1,j} - x_{C_{Height},j}) = 0
\end{aligned} \tag{18}$$

Fig. 26 demonstrate the technique of optimizing a non-rectangular banner area. It is worth noting that the retargeted areas that are most pleasing visually are natural or texture areas, while geometrically structured areas are more prone to visual artifacts.

5.8. Real-estate expansion

Not always a suitable sized ‘‘real-estate‘‘ can be found, on those occasions one needs to enlarge a small area for the favor of a large advertisement, this without harming the salient regions of the video. We present the following expansion scheme, that respects the salient regions of the video, to solve the aggrandizement problem. Assume we already have the small sized video real-estate in a binary pixel map $M_{i,j,t}$ we want to expand the marked regions whilst preserving the saliency $S_{i,j,t}$ in the video. We follow with a linear system, where the target growth factor is marked by F^{re}. A few ‘‘real-estate‘‘ manipulations are shown in Fig. 27.

$$\begin{aligned}
&\forall_{1\leqslant i\leqslant\ C_{Height}} x_{i,1} = 1\\
&\forall_{1\leqslant i\leqslant\ C_{Height}} x_{i,C_{Width}} = C_{Width}\\
&(1 - M_{i,j,t})S_{i,j,t}(x_{i,j,t} - x_{i,j-1,t} - 1) = 0\\
&M_{i,j,t}W^r(x_{i,j,t} - x_{i,j-1,t}) = F^{re}\\
&W^s(x_{i,j,t} - x_{i+1,j,t}) = 0\\
&W^t(x_{i,j,t} - x_{i,j,t+1}) = 0
\end{aligned} \tag{19}$$

6. Summary

We introduced a system for content aware non-homogenous video manipulation, whilst respecting the saliency and preserving the user experience. The proposed system is efficient enough for real-time processing: both the saliency and the optimization stage are computed in under 33ms for Standard-Definition video on an Intel Core2 duo 2.4Ghz.

The framework is flexible, and is able to provide solution for several video manipulation tasks: video down-sampling, aspect ratio alterations, non-homogenous video expansion, real-estate optimization and creation. We also demonstrated innovative applications using the same framework to achieve video abstractions, object removal from a video and object displacement in a video based on respective saliency.

The optimization seems suitable and efficient for all of the above mentioned tasks. However, it relies on the suitability of the computed saliency measure. The components currently employed in the saliency cannot accurately capture the human sense of importance. Therefore, the main remaining goal is finding the salient regions in a video.

To allow professional users (e.g. video editor) to alter the automatic saliency map, we developed an ‘‘Adobe® After Effects’’ plugin. As can be seen in Fig. 28 the user can manually correct the saliency map, thus producing a better optimized video.

The suggested framework is robust and flexible enough to apply to a variety of real world video and image manipulation challenges beyond saliency based manipulations. For example, we have recently extended this framework to allow semi-automatic conversion of conventional 2D video to stereoscopic video [27].

References

[1] L. Wolf, M. Guttmann, D. Cohen-Or, Non-homogeneous content-driven video-retargeting, in: Proceedings of the Eleventh IEEE International Conference on Computer Vision (ICCV-07), 2007.

[2] B. Suh, H. Ling, B.B. Bederson, D.W. Jacobs, Automatic thumbnail cropping and its effectiveness, in: UIST ’03: Proceedings of the 16th Annual ACM Symposium on User Interface Software and Technology, ACM Press, New York, NY, USA, 2003, pp. 95–104.

[3] L. Itti, C. Koch, Comparison of feature combination strategies for saliency-based visual attention systems, Proc. SPIE Human Vision and Electronic Imaging IV (HVEI’99), vol. 3644, SPIE Press, San Jose, CA, 1999, pp. 473–482.

[4] P. Viola, M. Jones, Robust real-time face detection, International Journal of Computer Vision 57 (2) (2004) 137–154.

[5] F. Liu, M. Gleicher, Video retargeting: automating pan and scan, in: MULTIMEDIA ’06: Proceedings of the 14th Annual ACM International Conference on Multimedia, ACM Press, New York, NY, USA, 2006, pp. 241–250.

[6] V. Setlur, S. Takagi, R. Raskar, M. Gleicher, B. Gooch, Automatic image retargeting, in: MUM ’05: Proceedings of the 4th International Conference on Mobile and Ubiquitous Multimedia, ACM Press, New York, NY, USA, 2005, pp. 59–68.

[7] F. Liu, M. Gleicher, Automatic image retargeting with fisheye-view warping, in: UIST ’05: Proceedings of the 18th Annual ACM Symposium on User Interface Software and Technology, ACM Press, New York, NY, USA, 2005, pp. 153–162.

[8] A. Shamir, O. Sorkine, Visual media retargeting, in: ACM SIGGRAPH Asia Courses, 2009.

[9] R. Gal, O. Sorkine, D. Cohen-Or, Feature-aware texturing, in: Proceedings of Eurographics Symposium on Rendering, 2006, pp. 297–303.

[10] O.S. Yu-Shuen Wang, Chiew-Lan Tai, T.-Y. Lee, Optimized scale-and-stretch for image resizing, ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH ASIA) 27 (5).

[11] Y.-F. Zhang, S.-M. Hu, R.R. Martin, Shrinkability maps for content-aware video resizing, Computer Graphics Forum 27 (7) (2008) 1797–1804.

[12] S. Avidan, A. Shamir, Seam carving for content-aware image resizing, ACM Transactions on Graphics (Proceedings SIGGRAPH) 26 (3) (2007).

[13] M. Rubinstein, A. Shamir, S. Avidan, Improved seam carving for video retargeting, ACM Transactions on Graphics (Proceedings SIGGRAPH) 27 (3) (2008).

[14] P. Krähenbühl, M. Lang, A. Hornung, M. Gross, A system for retargeting of streaming video, in: SIGGRAPH Asia ’09: ACM SIGGRAPH Asia 2009 papers, ACM, New York, NY, USA, 2009, pp. 1–10. <http://doi.acm.org/10.1145/1661412.1618472>.

[15] Y.-S. Wang, H. Fu, O. Sorkine, T.-Y. Lee, H.-P. Seidel, Motion-aware temporal coherence for video resizing, ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH Asia) 28 (5) (2009) (article no. 127).

[16] Y.-S. Wang, H.-C. Lin, O. Sorkine, T.-Y. Lee, Motion-based video retargeting with optimized crop-and-warp, ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH) 29 (4) (2010) (article no. 90).

[17] J. Meng, Y. Juan, S.-F. Chang, Scene change detection in an MPEG-compressed video sequence, in: A.A. Rodriguez, R.J. Safranek, E.J. Delp (Eds.), Proc. SPIE, vol. 2419, pp. 14–25, Digital Video Compression: Algorithms and Technologies 1995, Arturo A. Rodriguez; Robert J. Safranek; Edward J. Delp; Eds., 1995, pp. 14–25.

[18] J. Lu, M. Liou, A simple and efficient search algorithm for block-matching motion estimation, IEEE Transactions on Circuits and Systems for Video Technology 7 (2) (1997) 429–433.

[19] The various videos presented in this manuscript can be found at: <http://www.cs.tau.ac.il/guttm/video/>.

[20] S.-C. Liu, C.-W. Fu, S. Chang, Statistical change detection with moments under time-varying illumination, IEEE Transactions on Image Processing 7 (9) (1998) 1258–1268.

[21] T.A. Davis, A column pre-ordering strategy for the unsymmetric-pattern multifrontal method, ACM Transactions on Mathematical Software 30 (2) (2004) 165–195. <http://doi.acm.org/10.1145/992200.992205>.

[22] M. Irani, P. Anandan, J. Bergen, R. Kumar, S. Hsu, Efficient representations of video sequences and their applications (1996).

[23] C. Pal, N. Jojic, Interactive montages of sprites for indexing and summarizing security video, CVPR ’05: Proceedings of the IEEE Computer Society Conference

on Computer Vision and Pattern Recognition, vol. 2, IEEE Computer Society, Washington, DC, USA, p. 1192.

[24] A. Rav-Acha, Y. Pritch, S. Peleg, Making a long video short: dynamic video synopsis, in: CVPR '06: Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, IEEE Computer Society, Washington, DC, USA, 2006, pp. 435–441.

[25] C.D. Correa, K.-L. Ma, Dynamic video narratives, ACM Transactions on Graphics (Proceedings of SIGGRAPH) 29 (3).

[26] C. Barnes, D.B. Goldman, E. Shechtman, A. Finkelstein, Video tapestries with continuous temporal zoom, ACM Transactions on Graphics (Proceedings of SIGGRAPH) 29 (3).

[27] M. Guttmann, L. Wolf, D. Cohen-Or, Semi-automatic stereo extraction from video, in: Proceedings of the Twelfth IEEE International Conference on Computer Vision (ICCV-09), 2009.