Video Stabilization and Face Saliency-based Retargeting

Yinglan Ma[1], Qian Lin[2], Hongyu Xiong[2]
[1]Department of Computer Science, Stanford University [2]Department of Applied Physics, Stanford University

Abstract

Technology revolution has brought great convenience of daily life recording using cellphones and wearable devices nowadays. However, hand shake and human body movement is likely to happen during the capture period, which significantly degrades the video quality. In this work, we study and implement an algorithm that automatically stabilizes the shaky videos. We first calculate the video motion path using feature matching and then smooth out high frequency undesired jitters with L_1 optimization. The method ensures that the smoothed paths only compose of constant, linear and parabolic segments, mimicking the camera motions employed by professional cinematographers. Since the human face are of broad interest and appear in large amount of videos, we further incorporated face feature detection module for video retargeting purposes. The detected faces in the video also enables many potential applications, and we add decoration features in this work, e.g., glasses and hats on the faces.

1. Introduction

Nowadays nearly 2 billion people own smartphones worldwide, and an increasing number of videos are captured by mobile devices. However, videos captured by hand handled devices are always shaky and undirected due to the lack of stabilization equipment on the handhold devices. Even though there are commercial hardware components that could stabilize the device when we record, they are relatively redundant and not handy for daily use. Moreover, most hardware stabilization systems only removes high frequencies jitters but are unable to remove low frequency motions arise from panning shots or walking movements. Such slow motion is particular problematic in shots that intend to track prominent foreground object or person.

To overcome the above difficulties, we implement a post-processing video stabilization pipeline aiming to remove undesirable high and low frequency motions from casually captured videos. Similar to most post-processing video stabilization algorithms, our implementation involves three main steps: (1) estimate original shaky camera path from feature tracking in the video; (2) calculate a smoothed path, which is cast as an constraint optimization problem; (3) Synthesizing the stabilized video using the calculated smooth camera path. To reduce high frequency noise, we use the L_1 path optimization method described in [1] to produce purely constant, linear or parabolic segments of smoothed motion, which follows cinematographic rules. To reduce low frequency swanning in videos containing a person as the central object, we apply further restraint to the motion of the facial features. In order to make the solution approachable, our method uses automatic feature detection and do not require user interaction.

Our video stabilization method is a purely software approach, and can be applied to videos from any camera devices and sources. Another popular class of mobile video stabilization methods use the phone's build-in gyroscope to measure the camera path. Our method has the advantage of being applicable to any video from any sources, for example online video, without any prior knowledge of the capturing device or other physical parameters of the scene. Our approach also enables facial retargeting, which can be extent to other kinds of salient features.

2. Previous Work

2.1. Literatures

Video stabilization methods can be categorized into three major directions: 2D method, 3D method and motion estimation method.

2D methods estimate frame-to-frame 2D transformations, and smooth the transformations to create a more stable camera path. Early work by Matsushita et al. [5] applied low-path filters to smooth the camera trajectories. Gleicher and Liu [4] proposed to create a smooth camera path by inserting linearly interpolated frames. Liu et al.[6] later incorporated subspace constraints in smoothing camera trajectories, but it required longer feature tracks.

3D methods rely on feature tracking to stabilize shaky videos. Beuhler et al. [8] utilized projective 3D reconstruction to stabilize videos from uncalibrated cameras. Liu et al. [9] were the first to introduce content-preserving warping in video stabilization. However, 3D reconstruction is difficult and unrobust. Liu et al. [6] reduced the problem

to smoothing long feature trajectories, and achieved comparable results to 3D reconstruction based methods. Goldstein and Fattal[10] proposed an epipolar transfer method to avoid direct 3D reconstruction. Obtaining long feature tracks is often fragile in consumer videos due to occlusion, rapid camera motion and motion blur. Lee et al. [11] incorporated feature pruning to select more robust feature trajectories to resolve the occlusion issue.

Motion estimation methods calculate transitions between consecutive frames with view-overlap. To reduce the alignment error due to parallax, Shum and Szeliski[12] imposed local alignment, and Gao et al.[7] introduced a dual-homography model. Liu et al[13] proposed a mesh-based, spatially-variant homography model to represent the motion between video frames, but the smoothing strategy did not follow cinematographic rules.

Our implementation, based on [1], apply L_1-norm optimization to generate a camera path that consists of only constant, linear and parabolic segments, which follow cinematographic principles in producing professional videos.

2.2. Our Contribution

In this work, we re-implement the L_1-norm optimization algorithm [1] to automatically stabilize the videos captured, with a smoothed feature path containing only constant, linear and parabolic segments. Additionally, in order to enable the video to retarget on human faces, we use the facial landmark detection algorithm from OpenFace toolkit [3] to set facial saliency constraints for the path smoothing; the strength of the constraint could be tuned from 0 (no facial retargeting) to 1 (video fixing on facial features), and in this way we are able to combine both video path smoothing and facial retargeting according to specific user needs.

Beyond that, in order to make our work more fun, we also manage to attach interesting decorations such as hat, glasses, and tie above, on, or below the human faces detected, and their transformations are based on the movement of human face in the video.

3. Proposed Method

3.1. L_1-Norm Optimized Video Stablization

In this section, we describe the method of video stablization in this work.

3.1.1 Norms of smoothing

When applying path smoothing algorithm, we should always be careful to choose which regularization method we use, since different regularization methods works differently for different error distribution. [2]

For error distributions with sharply defined edge or extremes (typified by the uniform distribution) one should use Tchebycheff (L_∞) smoothing. For error distributions at the other end of the spectrum, which is with long tails, one should use L_1 smoothing. In between these extremes, which are short-tail spectra such as normal distribution, least squares or L_2 smoothing appears to be best.

3.1.2 L_1-Norm Optimization

In the perspective of a single feature point, the video motion can be viewed as a path of its coordinates (x, y) movement with respect to the frame number. Since it is difficult to avoid jitters with hand-held devices, we will observe that the path the is wiggling. Video stablization is to obtain the new coordinates (x, y) at each frame and thus a new path with enhanced smoothness. In the perspective of the frames, the task is to smooth the transformations between frames so that the feature points movement would be minimal. The frame transformation is generalized as affine transform, including translational and rotational motion, and scaling caused by object/camera distance change.

We estimate the camera path by first matching features between consecutive frames C_t and C_{t+1}, and then calculate the affine transformation F_{t+1} based on the matching. That is, the process can be formatted as $C_{t+1} = F_{t+1}C_t$. Then we estimate the affine transformation F_{t+1} using these two set of feature coordinates, C_t and C_{t+1}. In this work, we extract features of each frame (opencv function cv::goodFeaturesToTrack), and find the matching in the next frame using iterative Lucas-Kanade method with pyramids (cv::calcOpticalFlowPyrLK).

We denote the smoothed features as P_t, then we have a correlation between the original features in frame t and the smoothed ones, as $P_t = B_tC_t$, where B_t is is the stabilization/retargeting matrix, transforming the original features to the smoothed ones. Since we only want the smoothed path to contain constant, linear, and parabolic segments, we minimize the first, second, and third derivatives of the smoothed path with weights $c = (c^1, c^2, c^3)^T$:

$$O(P) = c^1|D(P)|_1 + c^2|D^2(P)|_1 + c^3|D^3(P)|_1, \quad (1)$$

where

$$\begin{aligned} |D(P)|_1 &= \sum_t |P_{t+1} - P_t|_1 = \sum_t |R_t|_1, \\ |D^2(P)|_1 &= \sum_t |R_{t+1} - R_t|_1, \\ |D^3(P)|_1 &= \sum_t |R_{t+2} - 2R_{t+1} + R_t|_1. \end{aligned} \quad (2)$$

Here the residual is $R_t = B_{t+1}F_{t+1} - B_t$.

For each affine transform:

$$B_t = \begin{bmatrix} b_{11} & b_{12} & t_x \\ b_{21} & b_{22} & t_y \end{bmatrix} \quad (3)$$

in 6 DOF, we vectorize it as $p_t = (b_{11}, b_{12}, b_{21}, b_{22}, t_x, t_y)^T$, which is the parametrization of B_t; correspondingly

$$|R_t(p)|_1 = |p_{t+1}^T M(F_{t+1}) - p_t|_1. \quad (4)$$

We make use of Linear Programming (LP) technique to solve this L_1-norm optimization problem. To minimize $|R_t(p)|_1$ in LP, we introduce slack variables $e^1 \geq 0$, so that $-e^1 \leq R_t(p) \leq e^1$; similarly there are e^2 and e^3 for $|R_{t+1}(p) - R_t(p)|_1$ and $|R_{t+2}(p) - 2R_{t+1}(p) + R_t(p)|_1$, respectively. For $e = (e^1, e^2, e^3)^T$, the objective function of the problem is to minimize $c^T e$.

In addition, we want to limit how much B_t (or p_t) could deviate from the original path, i.e. the actual shift should within the cropping window. Thus, we can add constraints on the parameters in LP, such as: $lb \leq Up_t \leq ub$, where U is the linear combination coefficient of p_t. The complete L_1 minimization LP for smoothed video path with constraints is summarized below:

Algorithm 1 Summarized LP for the smoothed video path

Input: Frame pair transform $F_t, t = 1, 2, ..., n$
Output: Update transform B_t
▷ B_t could be transformed to p_t
Minimize: $c^T e$
w.r.t $p = (p_1, p_2, ..., p_n)$
where $e = (e^1, e^2, e^3)^T$, $e^i = (e_1^i, e_2^i, ..., e_n^i)$, $c = (c^1, c^2, c^3)^T$
subject to:
1. $-e_t^1 \leq R_t(p) \leq e_t^1$
2. $-e_t^2 \leq R_{t+1}(p) - R_t(p) \leq e_t^2$
3. $-e_t^3 \leq R_{t+2}(p) - 2R_{t+1}(p) + R_t(p) \leq e_t^3$
4. $e_t^i \geq 0$
constraints:
$lb \leq Up_t \leq ub$

We use lpsolve library for modeling and solving our LP system.

3.2. Facial Features Detection and Retargeting

In many videos, a particular subject, usually a person, is featured. In this case it is not only important to remove fast, jittering camera motions, but also unintended slow panning or swanning that momentarily move the subject off-center and lead to distraction for the viewer. This can be posed as a constraint on the path optimization as requiring that salient features of the subject to be closed to the center region throughout the video.

The first step towards salient-point-preserving video stabilization is salient feature detection and tracking. In particular, it is desirable to have the algorithm automatically recognize and detect these salient features without user input. There are many face detectors available for such task. We use Constrained Local Neural Fields (CLNF) for facial landmark detection available on OpenFace. Detail of the algorithm can be found in [3]. The CLNF algorithm works robustly under varied illumination and are stabilized for video. It outputs a fixed number of facial landmarks, including the face silhouette, the lips, nose tip and eyes, as shown in Fig. 2c. These multiple landmarks allow a more stable and accurate estimate of the facial position. In contrary, other face detector, for example the opencv built-in ones, were observed to produce inaccurate bounding box and are not stable over video frames during our experiment. The detailed facial landmarks from CLNF also enable us to perform other post-processing on the video, for example the face decoration described in Section 3.4.

After detecting the facial landmarks in each frame t, we estimate the center of face $C_{f,t}$ by averaging all the landmarks. Let C_0 be the desired position of the center of face, for example the center of frame. Let P_t and S_t be the original and smoothed camera trajectory, then the saliency constraint can be posed as a additional term to the loss function

$$L_t = (1 - w_s)(S_t - \bar{P}_t)^2 + w_s(S_t - P_t + C_{f,t} - C_0)^2 \quad (5)$$

where $\bar{P}_t$ is average over a window of frames, and w_s is a parameter to adjust how much weight the saliecy constraint have on the optimization. Minimizing L_t then produce the desired smoothed trajectory S_t.

3.3. Metrics & Characterization

3.3.1 Evaluation of Smoothed Path

For the stabilizing problem we are concerning about, it would be inappropriate to simply regard the undesired shaking as short-tail normal distribution, so using the L_1 norm between each frame pair during minimization is more suitable. In addition, L_1 optimization has the property that the resulting solution is sparse, i.e. the computed path therefore has derivatives which are exactly zero for most segments. On the other hand, L_2 minimization (in a least-squared sense), tend to result in small but non-zero gradients. Qualitatively, the L_2 optimized camera path always has some small non-zero motion (most likely in the direction of the camera shake), while the L_1 optimized we used ($|D(P)|_1$, $|D^2(P)|_1$, and $|D^3(P)|_1$) will create path is only composed of segments resembling a static camera, (uniform) linear motion, and constant acceleration [1].

Therefore, we will compare the L_1 norm $|D(P)|_1$ between the original video feature path and the smoothed one, and use this comparison as metrics of our experiments described below. Specifically, we will calculate the average absolute shift between adjacent points on the video feature path, with respect to both x and y directions, and average absolute rotation angle increment. The same calculations will be done to the smoothed path.

3.3.2 Evaluation of Facial Retargeting

As for the part of facial retargeting, in addition to the comparison between the L_1 norm $|D(P)|_1$ of the original video feature path and the new one, where we can extract the information about smoothing, we are also interested to see how the facial features are targeted. So we will calculate the average position of the face features with respect to the center of frame, and simultaneously calculate the average absolute position deviation.

3.4. Face Decoration

With per frame face features detected, we can add fun face decorations to our videos, such as glasses, hat and mustache. By incorporating feature locations, we are able to translate, scale and rotate the decorations to place them appropriately onto human faces. Since our videos are stabilized and focus on faces, the transitions of the decorations are smoother. Here is an example of how we utilize the feature points in adding decorations.

Adding glasses: we extract left eye, right eye, left brow and right brow feature points to calculate a horizontal eye axis, and use it to estimate the orientation of the glasses. Scale is approximated from eye distance, and translation depends on the locations of the eye points.

Since face silhouette feature points are usually less stable, we avoid using those points in adding face decorations.

Screenshots of adding hat and glasses are shown in Fig. 4.

4. Experiments

Table 1 lists the algorithm run time on our laptop. The second column lists time for path smoothing without facial feature, and the third column lists time for path smoothing with facial feature as salient constraint. In the latter case, the CNFL facial landmark detection takes up the biggest chunk of time ($\sim$ 45ms per frame). [1] reported 20 fps on low-resolution video, and 10 fps with un-optimized saliency.

Table 1. Timing per frame of the algorithm. Video resolution 640×360.

	w.o. face	w. face
motion estimation (ms)	12.1	59.1
optimize camera path (μs)	0.15	0.40
render final result (μs)	2.7	2.7
face decoration (ms)	-	5.7
total (ms)	15	68
speed (fps)	67	15

4.1. Video Stabilization

We apply our path smoothing algorithm to shaky videos and observe significant reduction of jittering. An example output can be found on Youtube.

To visualize the effect of stabilization, we plot the estimated camera trajectory before and after our algorithm in Fig. 1. We also provide a quantitative measurement of the L_1 norm $|D(P)|_1$ before and after smoothing in Table. 1. As we can see the L_1 norm decreases a lot, which means the abrupt jitters are significantly decreased.

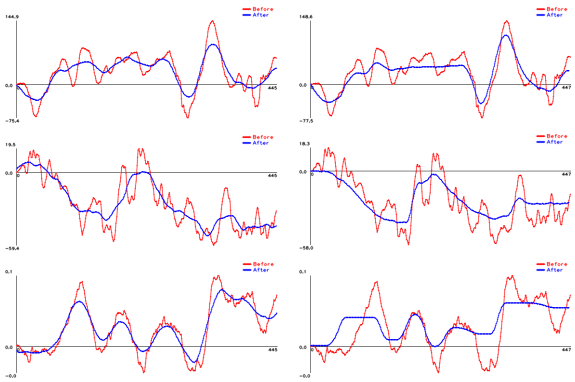

Figure 1. Path before and after (Left column) L_2-norm smoothing (Right column) L_1-norm smoothing. (Top)x-direction. (Middle)y-direction. (Bottom)rotational angle.

Table 2. L_1 norm $|D(P)|_1$ between the original video feature path and the smoothed one, in both x and y directions and the rotational angle.

path	$< \|\Delta x_t\| >$	$< \|\Delta y_t\| >$	$< \|\Delta a_t\| >$
original	1569	857	1.12
smoothed	705	234	0.44

4.2. Facial Retargeting

Our experiment with video stabilization using facial features are shown in Fig. 2. Fig. 2(a) is the original video, which contains slow swanning motion of both the camera and the subject person. Fig. 2(b) is the stabilized output using only camera path smoothing. The slow motion of the subject is still prominent. Fig. 2(c) is the stabilized output using camera path smoothing with a constraint of the motion of facial features. It leads to stabilization of the subject at the center over frames. Both result videos can be found on Youtube link 1 and link 2.

As expected, stabilization comes at a price of reduced resolution. The original image are cropped by 20% in Fig. 2(b) and (c) to remove black margins due to warpping. There are still residue margins in Fig. 2(c).

We also quantify the smoothing effect and the facial targeting, as we can see from Table. 2. With the increase of the facial saliency constraint ratio ω, both L_1 norm and the absolute position shift drops, which means, the larger ω is, the smoothier the video gets, and the more centered the human face is. The result is expected from our algorithm.

4.3. Comparison with State-of-the-art Systems

Due to no publicly available implementation of previous works, we obtain the original and output videos reported in Grundmann's paper [1], and calculate the evaluation metrics described in Section 3.3 on their output video and present alongside with our results. As we can see from the comparison below, our implemented algorithm is comparable to the state-of-the-art system.

4.4. Face Decoration

With per frame face features detected, we can add fun face decorations to our videos, such as glasses, hat and mustache. By incorporating feature locations, we are able to translate, scale and rotate the decorations to place them appropriately onto human faces. Since our videos are stabilized and focus on faces, the transitions of the decorations are smoother. Screenshots of adding hat and glasses are shown in Fig. 4.

5. Conclusion & Perspectives

All in all, video feature path is significantly smoothed using the L_1 optimization stabilization algorithm; the L_1 norm $|D(P)|_1$, which signifies the moving between frames, greatly drops after applying the stabilization.

If the facial retargeting method is included, the video would be more focused on human faces; the larger the saliency constraint ratio ω is, the more centered the human faces are with respect to the cropped video frame.

Decoration addition such as glasses, hat, or tie could also be attached to the faces in the video, with the same orientation as the faces. More fun stuffs will be applied to make this work fancier in the future.

Reference

[1] Matthias Grundmann, Vivek Kwatra, Irfan Essa. Auto-Directed Video Stabilization with Robust L1 Optimal Camera Paths. CVPR, 2011.

[2] JR Rice, JS White. Norms for smoothing and estimation. SIAM review, 1964.

[3] Tadas Baltruaitis, Peter Robinson, and Louis-Philippe Morency. Constrained Local Neural Fields for robust facial landmark detection in the wild. ICCVW, 2013.

[4] Michael L. Gleicher and Feng Liu. 2007. Re-cinematography: improving the camera dynamics of casual video. In Proceedings of the 15th ACM international conference on Multimedia (MM '07). ACM, New York, NY, USA, 27-36.

[5] Yasuyuki Matsushita, Eyal Ofek, Weina Ge, Xiaoou Tang, and Heung-Yeung Shum. 2006. Full-Frame Video Stabilization with Motion Inpainting. IEEE Trans. Pattern Anal. Mach. Intell. 28, 7 (July 2006), 1150-1163.

[6] F. Liu, M. Gleicher, J. Wang, H. Jin, and A. Agarwala. Subspace video stabilization. In ACM Transactions on Graphics, volume 30, 2011.

[7] Junhong Gao , Seon Joo Kim , M. S. Brown, Constructing image panoramas using dual-homography warping, Proceedings of the 2011 IEEE Conference on Computer Vision and Pattern Recognition, p.49-56, June 20-25, 2011

[8] Buehler, C., Bosse, M., and McMillan, L. 2001. Non-metric image-based rendering for video stabilization. In Proc. CVPR.

[9] Feng Liu , Michael Gleicher , Hailin Jin , Aseem Agarwala, Content-preserving warps for 3D video stabilization, ACM Transactions on Graphics (TOG), v.28 n.3, August 2009

[10] Amit Goldstein , Raanan Fattal, Video stabilization using epipolar geometry, ACM Transactions on Graphics (TOG), v.31 n.5, p.1-10, August 2012

[11] Chen, B.-Y., Lee, K.-Y., Huang, W.-T., and Lin, J.-S. 2008. Capturing intention-based full-frame video stabilization. Computer Graphics Forum 27, 7, 1805–1814.

[12] Heung-Yeung Shum , Richard Szeliski, Systems and Experiment Paper: Construction of Panoramic Image Mosaics with Global and Local Alignment, International Journal of Computer Vision, v.36 n.2, p.101-130, Feb. 2000

[13] Shuaicheng Liu, Lu Yuan, Ping Tan, and Jian Sun. 2013. Bundled camera paths for video stabilization. ACM Trans. Graph. 32, 4, Article 78 (July 2013), 10 pages.

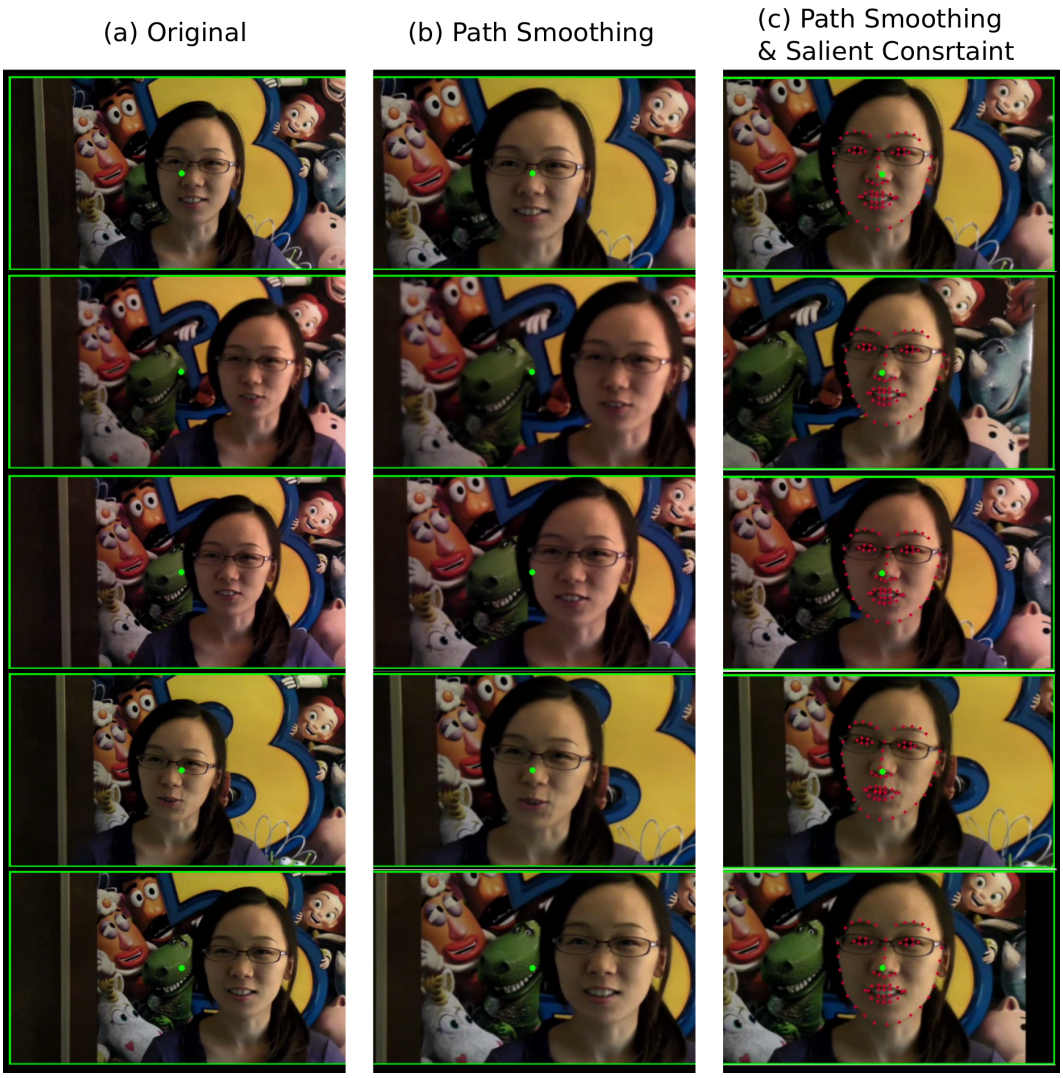

Figure 2. Demonstration of facial retargeting in video stabilization. The green dot indicates the center of frame. Green lines show boarder of frame. Red dots in (c) indicated detected facial landmarks from OpenFace [3]. They are intended as a guide to the eye. Both videos can be found on Youtube (b) and (c).

Table 3. L_1 norm $|D(P)|_1$ between the original video feature path and the smoothed one, in both x and y directions and the rotational angle.

ω	$< \lvert\Delta x_t\rvert >$	$< \lvert\Delta y_t\rvert >$	$< \lvert x - x_{center}\rvert >$	$< \lvert y - y_{center}\rvert >$
original	1392	496	32805	5882
0.2	1139	254	32583	4902
0.5	792	234	21568	3433
0.95	221	247	2695	1954

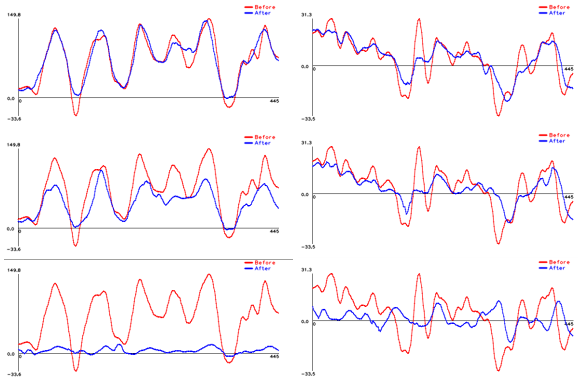

Figure 3. Path smoothing before and after with facial saliency constraints. (Left column) x-direction. (Right column) y-direction. From top to bottom, the facial constraint ratios ω are 0.2, 0.5, and 0.95, respectively.

Table 4. Comparison between our algorithm and the state-of-the-art one from [1]

method	$< \lvert\Delta x_t\rvert >$	$< \lvert\Delta y_t\rvert >$	$< \lvert\Delta a_t\rvert >$
state-of-the-art [1]	273	296	0.53693
our algorithm	705	234	0.44387

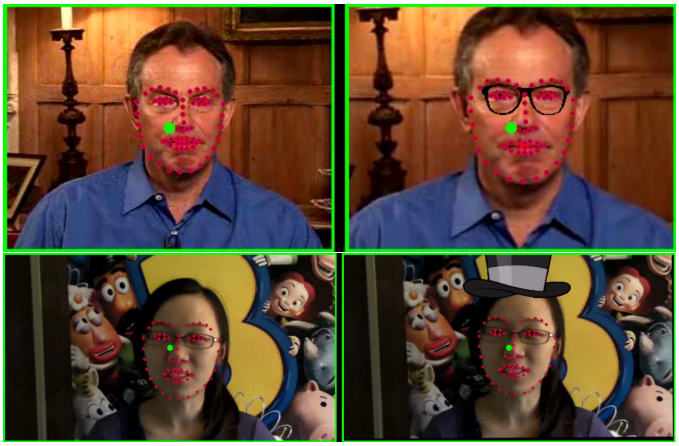

Figure 4. Face decoration with glasses and hat