Multimedia Tools and Applications manuscript No.
(will be inserted by the editor)

Content-and-disparity-aware Stereoscopic Video Stabilization

Shih-Syun Lin · Thi Ngoc Hanh Le · Pang-Yu Wu · Tong-Yee Lee

Received: date / Accepted: date

Abstract Filming stereoscopic videos has become easier with the development of science and technology, and such videos now proliferate on the Internet. Meanwhile, video stabilization is an important research topic. Thus, this study presents a method of stabilizing stereoscopic videos with preserving the disparities between objects in the frames. First, the feature points must be tracked and separated into many groups. We posit that the shaky motion is caused not only by translations but also by rotations. Thus, directly smoothing the path will not produce a similar trajectory so that we solve the shakiness of the turning before smoothing the path. To address such shakiness, we initially estimate the rotation angles between two adjacent frames. By determining the angle changes of all the frames, we can find out the preference of rotation in a video. Furthermore, the inconsistent angular velocity can be alleviated and the shakiness of the turning is solved by rotating the frame appropriately. Then, the Bézier curve is utilized to smooth the trajectories. We split a trajectory into a set of subtrajectories and subsequently smooth the latter independently. Unlike previous researches, we split the trajectory according to the feature tracking rate to obtain similar trajectories in the original video path. After making subtrajectories smooth, we merge them to attain a smoothed trajectory. The joint of the two subtrajectories is replaced by their interpolation. Finally, we optimize the smoothness and context preservation to stabilize videos without requiring extensive clipping.

Keywords Stabilization · Stereoscopic video · Trajectory · Bézier curve · Optimization

S.-S. Lin
National Taiwan Ocean University, E-mail: linss@mail.ntou.edu.tw

Thi Ngoc Hanh Le
National Cheng-Kung University, E-mail: lehanh.msc@gmail.com

P.-Y. Wu
National Cheng-Kung University, E-mail: bigben59456@gmail.com

T.-Y. Lee
National Cheng-Kung University, E-mail: tonylee@mail.ncku.edu.tw

1 Introduction

We often ignore hand shaking when shooting videos, but such this action may cause significant shaking in the final output. Many tips are available for preventing this situation, e.g., shooting a video slowly. However, these tips will restrict people from shooting certain types of videos, such as those that entail moving the camera quickly, and may have limited effects on stabilization. Nowadays, stereoscopic videos proliferate on the Internet. Removing the shaky motion in videos after shooting and maintaining the 3D quality become an important research topic. Many hardware-based stabilizers can help people stabilize videos spontaneously. However, it's inconvenient when stabilizers are always carried. Therefore, this research topic is worth exploring.

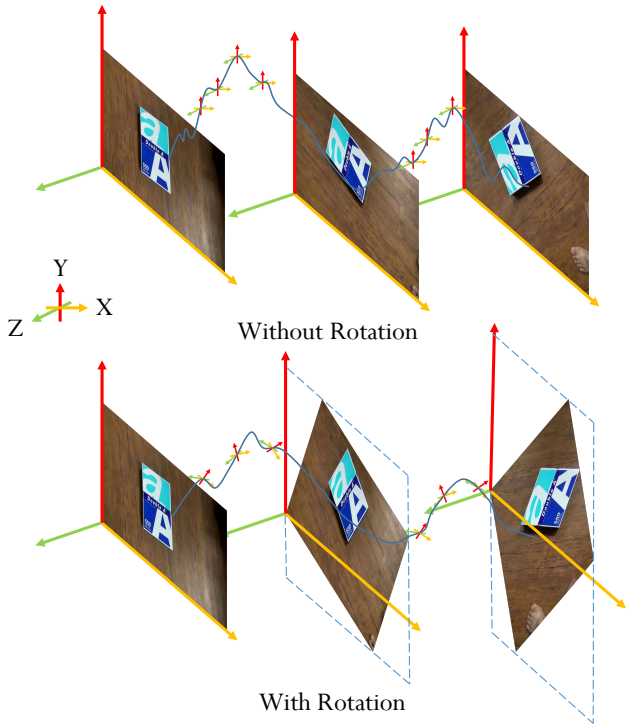

Fig. 1 Rotation before smoothing. We intend to reduce the shaking of features by rotating the frames before trajectory smoothing. The remainder of the shaking shall be caused by translation. The resulting smooth trajectory will be close to that desired by videographers in this step.

Shooting videos with hand-shaking may occur unnatural shaky videos. The phenomenon of shaky videos can be shaky translation or shaky rotation or both. Therefore, a suitable adjustment for translation and rotation is required. In this paper, we propose a useful video stabilization framework as follows. First, we identify and then weight the objects in the video. Second, we track the features in the video. Features are used to estimate relations between frames. Third, we remove turning shake and smooth the rotation motion. Fourth, we smooth the feature trajectories by using a Bézier curve after removing the turning shake

and then execute optimization to locate the features at the ideal position. Finally, we crop the video with a clipping window, and only valid information remains in the output video. Our system overview is explained in Section 3.

We separate the smoothing into rotation and translation, and they both exhibit order relation between frames. As rotation and translation influence the object position in the video, the camera in this study rotates and moves simultaneously. Although we can stabilize a video if we consider either process only, such approach may cause slightly inaccurate results. Turning shake is often small and can be reduced to a certain extent via the direct smoothing of the path. Relative to turning shake, translation shake is noticeable, so most existing researches aim to smooth only the translation motion. Only a few studies focus on the rotation adjustment. As far as we know, no work has separated the smoothing into two parts and considered them with order relation. We attempt to remove turning shake and render the path closer to what videographers desire (see Fig. 1). In the part of rotation, we remove turning shake and adjust the speed of rotation. In the part of translation, we provide a new method of trajectory cutting. In contrast to previous work, the main advantages of the proposed method are described as follows. The proposed method allows us to manage the stereoscopic video by keeping the disparity between videos for the left and right eyes. The disparity cannot be changed before the optimization of smoothness. Thus, the deformation in the result becomes remarkably small. We can easily control the disparity with an energy term to render the result of the video for the left eye similar to that for the right eye without the need to find the corresponding features in the two videos. Experimental results show that our method can effectively stabilize the stereoscopic video.

2 Related Work

Significance detection technique [3, 22, 2] can be applied to develop many types of research, such as image/video retargeting, thematic map generation, panorama warping, video stabilization, and so on. In our study, the goal aims to stabilize the shaky stereoscopic video while preserving the disparity. 2D video stabilization usually estimates the relation between consecutive frames initially by using this relation to represent the transformation in a video (e.g., affine or homography transformation). Similar to affine transformation, which includes translation, rotation, scaling, and shearing, our method considers rotation and translation. Scaling is caused by the camera yawing or pitching, and the scaling factor differs between objects. Shearing may change the distance between objects in the video and influence the disparity in a stereoscopic video. By contrast, homography transformation presumes that all objects are on the same plane and estimates the rotation and translation of the camera between two frames. Although the transformations usually provide good results, such an approach may incorrectly estimate the deformation (see Fig. 2), and the transformations may change the distances between objects. Therefore, we stabilize the videos by considering rigid transformation including rotation and translation.

In Fig. 2, the camera yaws at this moment, so the features of the cube move. However, the features of the cylinder remain stationary at this instant. If numerous objects similar to the cube are present, then the estimation of homography transformation is suitable for the features of the cube, but the features of the cylinder will be wrong with this transformation. For instance, the features of the cube on the right side can be obtained from those of the left side with homography transformation, but the features of the cylinder on the right side cannot be ascertained from those of the left side. In our proposed method, we avoid incorrect

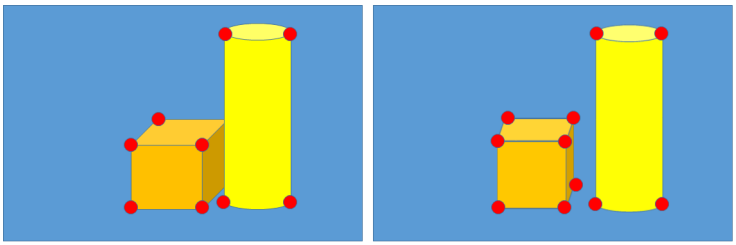

Fig. 2 Incorrect estimation of homography transformation.

estimation of homography transformation which is used in many previous works [23, 14, 15].

In 2D methods, the researches [18, 19, 12] smooth the transformations to stabilize videos and another approach [6] even uses L1 optimization. The aims of the study [7] include solving the rolling shutter effects. To stabilize videos with bundled-paths, the studies [17, 8, 23, 24] directly smooth the feature trajectories. We cannot directly stabilize stereoscopic videos with these methods because they do not maintain the disparity between the videos for the left and right eyes. Stabilizing the video for both eyes may eliminate the disparity and provide low visual quality of stereoscopic video. In contrast to previous works, our approach can maintain object contents and their disparities as possible as we could to stabilize stereoscopic video.

In 3D methods, the camera's motion in 3D space should be estimated. Liu et al. [14] recover the camera path by a structure-from-motion approach. Liu et al. [16] uses a depth camera to recover the camera path. Smith et al. [21] recover the camera path with a light field. Jia et al. [11] generalize 2D motion smoothing to 3D and aims at smoothing 3D rotation, and the rotation model is obtained with a gyroscope. Despite the good performance of 3D methods, using them to reconstruct a scene is difficult. 2.5D methods can obtain similar results as the 3D methods and stabilize additional kinds of videos [4, 15]. However, 2.5D and 3D methods require features tracked in a long term. Tracking features under such term are unsuitable for many videos. Thus, 2D methods are generally allowed to handle additional videos. We use a 2D method to stabilize videos in this study.

Wang et al. [23] smooth the path by a Bézier curve after tracking and optimization to stabilize videos. Similar to other research, Liu et al. [14] use other methods to smooth the path to resemble a line and parabola, and a low-pass filter is adopted to remove high-frequency motion from a video. Matsushita et al. [18] use a Gaussian filter to remove the shaky motion in a video. Zhang et al. [24] employ a bilateral filter to avoid a large cropping ratio. Guo et al. [9] estimate the camera motion with homography transformation, and then they smooth the video for one eye and warp it for the other eye to maintain parallax. These methods stabilize videos with homography transformation and may potentially change object depth. To maintain object depth in a video and prevent deformation from incorrect estimation as in Fig. 2, we propose an effective approach for stabilization in this paper.

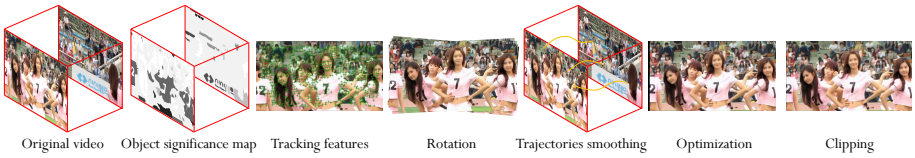

Fig. 3 System overview. First, we must generate a significance map. Then, we track the features in the video. We subsequently remove the shaky turning motion and smooth the trajectories of the features. Next, we perform optimization to obtain a stable video and crop the video with a clipping window.

3 Methodology

3.1 System Overview

The methods in [5, 3] are utilized to identify objects in a video and weight them. A frame is segmented into many patches. The set of patches in the tth frame is $P^t = \{p_1^t, p_2^t, p_3^t, ..., p_{n_p}^t\}$, and n_p is the number of patches. We track features in the video using the Voodoo camera tracker [1]. The set of features in the tth frame is $C^t = \{c_1^t, c_2^t, c_3^t, ..., c_{n_c}^t\}$, where n_c is the number of features. Many features will disappear in the video. Moreover, the ith feature c_i^t may not exist in the other frames. Similar to the approach [23], our method removes bad features by using an epipolar constraint [10]. We then estimate the rotation motion of the camera according to the good features. Frames in a video are rotated to render the rotation motion of the camera smoothly. Then, we smooth the feature trajectories after the rotation. Stabilization is optimized by making the features move as if in a smooth trajectory. Finally, we crop the video to show only valid information. The workflow of our system is illustrated in Fig. 3.

3.2 Significance Map Generation

Significance map is generated similarly to the study in [13]. The generated objects are then assigned feature weights. The stereo matching [20] and object tracking [5] play an important role in the stereoscopic video stabilization algorithm. In this study, we extract different objects in the video by following method [5] and estimate the weight of each pixel in a frame according to the study [3]. With the segmentation and the saliency map of a frame, we obtain two details: which pixel belongs to which object and each pixel's importance. Next, we ascertain the average saliency of all the pixels in an object. Average saliency is set as the object saliency in a frame. Then, the average saliency of the same object in different frames is set as the final object saliency in the video. The weights are used after global normalization.

3.3 Tracking and Weight Setting for Features

We utilize the Voodoo camera tracker [1] to track the features in the video and remove the bad features by using epipolar constraint [10], as performed in the study [23]. Furthermore, the features will be removed by tracking for a short time (i.e., less than 5 frames) and using the features with long distance movement (i.e., over 40 pixels between two consecutive frames). This approach allows us to ensure that the remaining features are highly reliable.

Given the demand to track features in the video and stabilize the video by smoothing the trajectories of these features, we utilize barycentric coordinates to express the feature position. Therefore, we can identify the grid through the location of its feature.

We can identify which object a feature belongs to by the segmentation of the video. By counting the feature that belongs to an object in the number of frames, the feature belongs to the object that it belongs to in most frames. If objects containing the features with the same number of frames are found, we categorize the features as belonging to the object in which the feature appears at the earliest. Therefore, every feature belongs to only one object in the video and the feature weight is the saliency value of the object.

3.4 Rotation Stabilization Process

Before smoothing the feature trajectories, we must remove the turning shake because the feature trajectories are more similar to the trajectories that videographers want. Turning shake influences the paths of smoothing the feature trajectories. Furthermore, camera rotation and translation lead to different feature movements. That is, the feature trajectories will be needed to be more stable. In addition, our method considers the camera rotation. If a meaningful rotation occurs in the video, then treating such rotation as shaky motion will be inappropriate. Consequently, the result differs excessively from the input video. Therefore, the turning shake of rotation is removed in this study while preserving the rotation tendency of the original video. This is our major difference or advantage in comparison with previous works [23, 14].

Fig. 4 Reference point. The red point is the center of the frame, and the closest feature is located in the yellow box. We set this point to be the reference point. Then, we rotate the frame around the red point. The position of the center is the same as the original position and the result is similar to the image on the right side.

3.4.1 Rotation Angle Estimation

First, we set the feature nearest to the center of the frame as the reference point. If the feature still can be tracked and remains close to the center of the frame, then we will retain the reference point. Otherwise, we set another feature point nearest to the center of the frame as new reference point when the original reference point has become far from the center of the frame or even disappeared. The process is depicted in Fig. 4. That is, to estimate the angle between the tth frame and the $(t+1)$th frame, we choose the feature nearest to the center of the tth frame as the reference point. Then, if the feature is far from the center of the frame or disappears at the $(t+n)$th frame, we will find a new reference point. With the reference point, we obtain vectors from the reference point to all the other remaining tracked features.

We calculate the dot product of the unit vector of the given vector and positive horizontal axis, and we obtain the angle by arccosine of the following dot product:

$$\theta_k^t = \arccos(\hat{u} \cdot \hat{c}_{m,k}^t), \tag{1}$$

where is $\hat{u}$ the unit vector of the positive horizontal axis, and $\hat{c}_{m,k}^t$ is the unit vector of the given vector from the reference point c_m^t to the tracked feature $(c_k^t \cdot c_m^t)$ is the reference point at the tth frame. As the unit vector has magnitude 1, the dot product of two unit vectors is the cosine value of the angle between two vectors. We can obtain the angle between two vectors by the arccosine function of the dot product (from 0° to 360°).

With the angle between the given vector and positive horizontal axis, we obtain the angle between the same given vector and positive horizontal axis in the next frame and the angle change by the subtraction of the two angles. Then, we use the average of angle changes to be the angle between two consecutive frames:

$$\Delta\theta^{t+1} = \frac{1}{n} \sum_{c_k \in C^t \cap C^{t+1}} (\theta_k^{t+1} - \theta_k^t), \tag{2}$$

where the angle change of the given vector $c_{m,k}^t$ from tth frame to $(t+1)$th frame is $(\theta_k^{t+1} - \theta_k^t)$, and we can use $\Delta\theta^{t+1}$ to represent it. n is the number of the tracked features. If n is extremely small, then we will set this angle change as 0°. In above equation (2), unreliable features usually lead to an incorrect angle estimation.

Next, in order to obtain the rotation angles of whole frames of the video, we add up all the rotation angles before the tth frame as follows:

$$\theta^t = \sum_{i=1}^{t} \Delta\theta^i, \tag{3}$$

The rotation motion of the input video is known. For instance, after adding up the rotation angles, the rotation angles between each two consecutive frames change from 0°, 1°, 3°, 2°, 3° to 0°, 1°, 4°, 6°, and 9°. That is, the angle is between a frame and the first frame and the trend of rotation is obtained.

3.4.2 Ideal Rotation Process

With the rotation angle between each frame and the first frame, we ascertain that the rotation in a period time is clockwise or counterclockwise. We also identify the moment when the direction of the rotation changes and the duration of rotation. To stabilize the rotation process through this information, we first determine the moments when the direction of rotation changes using the $(\theta^{t+1} - \theta^t) \times (\theta^t - \theta^{t-1})$, $\forall t \in T$, where a set T is defined as those moments. If the rotation angle increases in a period time, then both $(\theta^{t+1} - \theta^t)$ and $(\theta^t - \theta^{t-1})$ are positive. Otherwise, both are negative. In another words, if the rotation direction changes, their products will be negative.

Once the moments when rotation direction changes are identified, we check the time interval between two adjacent moments in T to distinguish between a jitter or a rotation turning point. If the time of a rotation direction change is very close to the last moment, then this rotation may be a jitter. Otherwise, it is a turning point between two meaningful rotation motions. Then, we choose two adjacent moments of the turning points. We posit that in a meaningful rotation motion, the rotation velocity is constant. In other words, the angles between each two adjacent frames are the same in this rotation motion and the extent of

rotation in this rotation motion does not change. Our goal is to make the rotation angles in a period follow an arithmetic progression. Assume that we want to change the original rotation angles 1°, 4°, 3°, 8°, and 9° into 1°, 3°, 5°, 7°, and 9°, respectively, after the stabilization. Notice that the 3 degrees of rotation in the original rotation angle is a jitter because the declining trend of the rotation angle is too short. We must remove the jitter in a meaningful rotation motion. Accordingly, we calculate the ideal angle for each frame with the identified meaningful rotation as follows:

$$\tilde{\theta}^{t_i} = \theta^{t_1} + \frac{i-1}{i+1}(\theta^{t_n} - \theta^{t_1}), \tag{4}$$

where t_1 is the beginning of the meaningful rotation and t_n is the end. A total of n frames are present in this period, and we can obtain the ith ideal rotation angle $\tilde{\theta}^{t_i}$ easily. In addition, θ^{t_1} and θ^{t_n} do not change, so the extent of the ideal rotation is similar to the original.

To rotate each frame with the ideal angle, using the angle between a frame and the first frame as well as the ideal angle between the frame and the first frame, we rotate the frame by a degree as follows:

$$\hat{\theta}^t = \tilde{\theta}^t - \theta^t. \tag{5}$$

For example, if the rotation angle at moment t is 9° counterclockwise but the ideal angle at that moment is 7° counterclockwise, then we should rotate the frame at that moment by 2° clockwise (or −2° counterclockwise). The angle −2° is calculated by subtracting the original from the ideal angle.

In order to retain video content and make the result stable, we maximally reduce the biggest rotation angle in the video by using the following equation:

$$\bar{\theta}^t = \hat{\theta}^t - \hat{\theta}^{t_\alpha} - \frac{\hat{\theta}^{t_\beta} - \hat{\theta}^{t_\alpha}}{2}, \tag{6}$$

where $t_\alpha = \min_{t \in frame} \hat{\theta}^t$ and $t_\beta = \max_{t \in frame} \hat{\theta}^t$. We obtain the moments with maximum and minimum rotation angles, and we can simplify the equation as follows:

$$\bar{\theta}^t = \hat{\theta}^t - \frac{\hat{\theta}^{t_\alpha}}{2} - \frac{\hat{\theta}^{t_\beta}}{2}. \tag{7}$$

In this step, we obtain the frames with the biggest and the smallest rotation angles. First, we set the smallest rotation angle to 0°. Therefore, the biggest rotation angle becomes $(\hat{\theta}^{t_\beta} - \hat{\theta}^{t_\alpha})$. For instance, if the original rotation angles are between −3° and 7°, after this step, it will be between 0° to 10°. Then, to minimize the rotation amplitude (that is the biggest clockwise rotation angle equal to the biggest counterclockwise rotation angle), we calculate the biggest rotation angle as $\frac{\hat{\theta}^{t_\beta} - \hat{\theta}^{t_\alpha}}{2}$ and the smallest rotation angle as $\frac{-(\hat{\theta}^{t_\beta} - \hat{\theta}^{t_\alpha})}{2}$. In the same way, the rotation angle between 0° and 10° will become −5° to 5°.

With the rotation angle of each frame, we obtain R^t, which is the rotation matrix R of the tth frame. Then, we rotate the frame around its center. Notice that if the video input is a stereoscopic video, we calculate the rotation angles of the video for the left and the right eyes separately, and we can obtain the average of the two rotation angles $\bar{\theta}^t = \frac{\bar{\theta}^t_L + \bar{\theta}^t_R}{2}$. Therefore, the rotation angles at tth frames are the same in the two videos, and the disparities between the two videos can be maintained. The rotation matrix of the tth frame is formulated as the following:

$$R^t = \begin{bmatrix} \cos\bar{\theta}^t & -\sin\bar{\theta}^t \\ \sin\bar{\theta}^t & \cos\bar{\theta}^t \end{bmatrix}. \tag{8}$$

This rotation matrix allows us to rotate a frame easily. We rotate all the feature points in the frame around the center of the frame by the rotation matrix as shown in Fig. 4. The features after rotation are expressed as:

$$\vec{c}_i^t = R^t \times (c_i^t - O) + O, \tag{9}$$

where O is the center of the frame and $\vec{c}_i^t$ is the new position of the ith feature at the tth frame. The new position of the feature point $\vec{c}_i^t$ is the original position of the feature c_i^t which rotates around the center of the frame by the matrix. Similarly, we rotate all the vertices of the frame as expressed by the following:

$$\vec{v}_i^t = R^t \times (v_i^t - O) + O, \tag{10}$$

where the ith vertex of the tth frame v_i^t rotates around the center of the frame like feature points. The set of vertices after rotation at the tth frame is $\bar{V}^t$. If the video input is a stereoscopic video, the vertex after the rotation of the video for the left eye $\vec{v}_i^{t,L}$ will be equal to that of the right one $\vec{v}_i^{t,R}$. Given that R^t is the same in the two videos, the relation of the two corresponding vertices will remain unchanged and the original vertices $v_i^{t,L}$ will be equal to $v_i^{t,R}$. This step will not affect the parallax between the two videos. Consequently, the problem of rotation is solved.

3.5 Trajectory Smoothing

After handling the shaky motion in rotation, we undertake the translation shake. We smooth trajectories by using the Bézier curve. We want to reduce the shake in the original video, but the smooth path should resemble the path of a videographer shooting. Thus, we must split the video before smoothing. As the videographer sometimes shoots and causes shakiness intentionally when moving the camera, we render the smooth and original paths similarly. Our goal is to remove the shaky motion and keep the camera path as smooth as possible. This step is necessary in most stabilization research. Many filters can handle high frequency shakiness, such as the low-pass filter. In the study [23], a degree 2 Bézier curve can be a smooth path of the features. We cut the trajectory according to the video content but not to a fixed length of time such as in the study [23]. Therefore, our smooth path is more similar to the original path. Furthermore, we use degree n Bézier curve to obtain the smooth path, where n is decided according to the number of frames in subtrajectories.

3.5.1 Trajectory Splitting by Video Content and Smoothing

In this phase, we first cut the feature trajectory into subtrajectories. The camera's tendency is not always similar to a video. If we directly smooth the whole trajectory, excessive difference from its original path may occur. Even though the camera's main motion does not change in a video, splitting the trajectory can still work. In this part, we find that frames with too few tracked features are cutting points. These frames may be the moments of camera motion change or scene change, and hence setting such frames as the cutting points is reasonable. When the cutting points are too close, we choose only one of them as the cutting point. This step prevents the subtrajectory from being too short and avoids unstable results. We cut the feature trajectories with these cutting points and then smooth the subtrajectories separately.

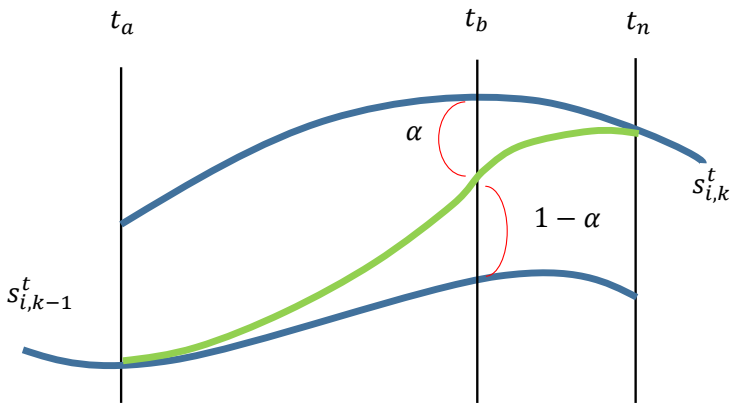

Fig. 5 Combined subtrajectories. The overlapping of two subtrajectories start from t_a to t_n. We control the ratio of the previous subtrajectory to the next subtrajectory by $\alpha = \frac{t_b - t_a}{t_n - t_a}$ when connecting the two subtrajectories at t_b ($t_a \leq t_b \leq t_n$). The blue line at the bottom of the figure is the previous subtrajectory and the other blue line is the next subtrajectory. The green line is the smooth path at the junction. The default length of the overlapping $(t_n - t_a)$ is 20 frames.

Finally, we obtain the smooth trajectory by connecting the smooth subtrajectories. For each subtrajectory, we can obtain its smooth path by using the following equation:

$$s^t_{i,x} = \sum_{m=t_1}^{t_n} \omega^m \cdot \bar{c}^m_i, \tag{11}$$

where $s^t_{i,x}$ is ideal position at the tth frame of the xth subtrajectory in the ith trajectory. We smooth the subtrajectory by using a Bézier curve. The feature that is generated by the aforementioned rotation is used as the control point, and $\bar{c}^m_i$ is the position of the ith trajectory after rotation at the mth frame. The subtrajectory begins at t_1 and ends at t_n. The weight of the control point $\bar{c}^m_i$ is calculated by $\omega^m = \binom{n}{m-t_1} \gamma^{m-t_1} (1-\gamma)^{n-m+t_1}$, where $\gamma = \frac{t-t_1}{n}$. The point close to $\bar{c}^t_i$ will have more influence on the ideal position at the tth frame. Therefore, we can obtain a smooth path, and this path is similar to the original camera's tendency.

We must connect the subtrajectories after smoothing. However, subtrajectories cannot be directly connected because two smooth paths differ at the junction. Connecting directly will suddenly change the moving direction of the feature. We split the trajectory to 20 frames which are overlapping between two consecutive subtrajectories, and we smooth the feature in the overlapping section to handle this problem easily. Interpolation is employed to smooth the features in the overlapping section. We obtain the complete smooth trajectory by connecting the end of the previous smooth subtrajectory and the beginning of the next smooth subtrajectory as follows:

$$s^{t_b}_i = (1-\alpha) \cdot s^{t_b}_{i,k-1} + \alpha \cdot s^{t_b}_{i,k}. \tag{12}$$

The overlapping section between two subtrajectories starts at t_a and ends at t_n. t_n is the end of the previous smooth subtrajectory $s^t_{i,k-1}$ as well as the cutting point, and t_a is the beginning of the next smooth subtrajectory $s^t_{i,k}$. We aim to obtain the ideal position at t_b ($t_a \leq t_b \leq t_n$). The ratio of $s^{t_b}_{i,k-1}$ to $s^{t_b}_{i,k}$ is controlled by $\alpha = \frac{t_b - t_a}{t_n - t_a}$. Therefore, both the first half of the junction and the end of the previous smooth path, as well as the second half of the junction and the beginning of the next smooth path would be alike. After this step, the ideal position of the ith trajectory at the t_bth frame $s^{t_b}_i$ is identified (see Fig. 5).

3.5.2 Subtrajectory Splitting by Displacement

Although we split the entire video into several clips, the camera may change its moving direction in an individual clip. In this situation, the smooth path would differ from the original path. We detect whether the ideal position is far from the original position and split the subtrajectory into two parts to prevent this situation. We split the trajectory at the frame with the less tracked feature, but we cut the subtrajectory in half here. We cut the subtrajectory in half instead of at the point far from the original position to reduce the distance between the ideal and original positions. Conversely, if we cut the subtrajectory at the point far from original position, then that point would remain in the same place. We check for any ideal position far from the origin in a subtrajectory and split the subtrajectory in half repeatedly. Therefore, the smooth path could be similar to the original path. These kinds of subtrajectories are smoothed by a Bézier curve as before. The combined subtrajectory can be obtained through the following:

$$s_{i,k}^{t_b} = (1-\alpha)\cdot s_{i,k_{j-1}}^{t_b} + \alpha \cdot s_{i,k_j}^{t_b}. \tag{13}$$

As the cutting here targets the subtrajectory, the ideal position of a subtrajectory at the t_bth frame $s_{i,k}^{t_b}$ is the interpolation of the position at t_b from the $(j-1)$th subtrajectory of the kth subtrajectory $s_{i,kj-1}^{t_b}$ and the next subtrajectory $s_{i,k_j}^{t_b}$, as shown in Fig. 5.

3.5.3 Frame Translation

With the ideal position of features, we translate all the frames according to the following equation:

$$\Delta d^t = \frac{1}{n}\sum_{i=1}^{n_c}(s_i^t - \bar{c}_i^t), \tag{14}$$

where the moving direction and distance at the tth frame is Δd^t. After obtaining the ideal position of features, we calculate Δd^t by averaging the distance between the ideal and original positions to all features at the tth frame. The weights of features are not considered in this step because the important features may be overfitting and the unimportant ones may be farther from the ideal position when we optimize in the next step. Although features moving far away from the ideal position are unimportant, the intense shaking of these features will make the result unstable. Therefore, most features become close to their ideal position before the optimization in this step.

With the distance and direction of the movement of the frame, we translate the frame and the translation will affect both vertices and features as follows:

$$\bar{v}_i^t = \vec{v}_i^t + \Delta d^t, \tag{15}$$

$$\bar{c}_i^t = \vec{c}_i^t + \Delta d^t, \tag{16}$$

where the vertices after rotation $\vec{v}_i^t$ and the features after rotation $\vec{c}_i^t$ can be moved to the new position $\bar{v}_i^t$ and $\bar{c}_i^t$ with this simple translation. If the input video is a stereoscopic video, then we must translate the two frames with the same translation to maintain the disparity between them. Therefore, Δd^t must equal $\frac{\Delta d_L^t + \Delta d_R^t}{2}$ when the input video is a stereoscopic video.

3.6 Optimization

In this section, we are going to perform optimization for the stabilization. To render the features after our rotation and translation close to the ideal position for producing a stable video, we must preserve the shape of the video content to prevent excessive deformation through the following equation:

$$\Omega_1 = \sum_{p \in P^t} \omega_p \sum_{e_i^t \in p} \|(Rot \times e_i^{'t} - e_1^{'t}) - (Rot \times \tilde{e}_i^t - \tilde{e}_1^t)\|^2, \tag{17}$$

where ω_p is the weight of a patch p as well as the weight of the object in the video. Deformation of important objects is more visually obvious than the deformation of unimportant ones. This weight ω_p allows us to preserve the shape of important objects with high priority. Then, we segment a frame into many patches by the initial objects in this frame. A patch is a set of many edges ($e^t = \begin{bmatrix} e_x^t \\ e_y^t \end{bmatrix}$). Suppose that the edge is between v_a^t and v_b^t, then $(e_x^t = v_{a_x}^t - v_{b_x}^t)$ and $(e_y^t = v_{a_y}^t - v_{b_y}^t)$. As the frame has been rotated and translated in the previous process, the edge $\tilde{e}$ here is also an edge after rotation and translation between the vertices and after rotation and translation $\tilde{v}^t$. *Rot* is a rotation matrix defined according to the relation between the ith edge $\tilde{e}_i$ and the first edge $\tilde{e}_1$. That is, the relation between two edges must be perpendicular or parallel because of quad mesh system. Thus, *Rot* must be $\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$ or $\begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}$. We only have to check whether $(e_i \cdot e_1)$ is 0 or not to obtain the matrix *Rot*, and then make $(Rot \times e_i^t)$ and e_1^t are parallel.

To achieve a stable result, the aforementioned features should be close to the ideal position as shown in the following equation:

$$\Omega_2 = \sum_{c_i \in C^t} \omega_{c_i} \|c_i^{'t} - s_i^t\|^2, \tag{18}$$

where ω_{c_i} is the weight of the feature c_i^t. We classify the features into groups at the beginning, and the importance of an object is a constant in the video. Therefore, the feature c_i has the same weight ω_{c_i} at any time. This energy term allows us to render the important features close to their ideal position. To make important features with high weights close to the ideal position is more significant than doing so for unimportant features. In other words, the shaky motion of an important object is more visually obvious, so stabilizing the important objects rather than the unimportant ones is more critical.

Although an energy term allows us to preserve the shape of objects, unimportant objects may be deformed and the features of such objects with low weights may be farther from their ideal position and original position when smoothing the features and retaining object shape. However, the heavy deformation of objects will lead to unsatisfactory results even for an unimportant object. Moreover, the distance of the features between the original and ideal position would differ even when they are found in the same object. For these reasons, we must render the features close to their original position by using the energy term as follows:

$$\Omega_3 = \sum_{c_i \in C^t} \omega_{c_i} \|c_i^{'t} - \tilde{c}_i^t\|^2. \tag{19}$$

Given the translation in the previous part, the video is more stable than the original. Features close to their original positions can preserve the shape of objects and the result is still stabilized. We solve this energy term with barycentric coordinates as well.

If a stereoscopic video must be stabilized, maintaining the disparity between the videos for the left and right eyes is necessary. The method in this study is formulated as the following equation:

$$\Omega_4 = \sum_{i=1}^{n_v} \|(v_i^{'t,R} - v_i^{'t,L}) - (\tilde{v}_i^{t,R} - \tilde{v}_i^{t,L})\|^2. \tag{20}$$

As the disparity is maintained in the previous parts of rotation and translation, we can preserve the disparity easily in this part by keeping the distance of the corresponding vertices between two frames. $v_i^{'t,R}$ is the ith vertex at the tth frame of video for the right eye, and $v_i^{'t,L}$ is the ith vertex at the tth frame of video for the left eye. The corresponding vertices after rotation and translation are $\tilde{v}_i^{t,R}$ and $\tilde{v}_i^{t,L}$. This energy term allows us to maintain the disparity which is similar to that of the input video.

The last constraint is a hard constraint. It aims to ensure that the meshes will not flip after optimization:

$$v_{b_x}^{'} - v_{a_x}^{'} > 0, \ \forall \ v \in g_i, \tag{21}$$

$$v_{c_x}^{'} - v_{d_x}^{'} > 0, \ \forall \ v \in g_i, \tag{22}$$

$$v_{d_y}^{'} - v_{a_y}^{'} > 0, \ \forall \ v \in g_i, \tag{23}$$

$$v_{c_y}^{'} - v_{b_y}^{'} > 0, \ \forall \ v \in g_i, \tag{24}$$

where the vertices in one grid g are a, b, c, and d in clockwise order from the top left corner to the bottom left corner. To make sure the top right corner is at the right side of the top left corner after optimization, we design the constraint $v_{b_x}^{'} - v_{a_x}^{'} > 0, \ \forall \ v \in g_i$. It also works on the other edges.

To minimize the objective function, we must determine which vertices can achieve it through the following equation:

$$\Omega = \omega_1 \Omega_1 + \omega_2 \Omega_2 + \omega_3 \Omega_3 + \omega_4 \Omega_4, \tag{25}$$

where ω represents the weights of energy terms. In this study, we set $\omega_1 = 10$, $\omega_2 = 0.1$, $\omega_3 = 1$, and $\omega_4 = 1$, and the result can minimize the objective function frame by frame.

3.7 Video Clipping

In order to make sure that valid information is obtained after the previous processes, we re-determine the boundary of the video with clipping window covers. The new boundary of the clipping window is determined as follows:

$$Left = \max_t \max_{v_i \in left\ boundary} V_{i_x}^t, \tag{26}$$

$$Rright = \min_t \max_{v_i \in rifht\ boundary} V_{i_x}^t, \tag{27}$$

$$Top = \max_t \max_{v_i \in top\ boundary} V_{i_y}^t, \tag{28}$$

$$Bottom = \min_t \max_{v_i \in bottom\ boundary} V_{i_y}^t. \tag{29}$$

That is, we set the rightmost point of the left boundary in all frames to the left boundary clipping window. Similarly, the right boundary clipping window is set by the leftmost point of the right boundary in all frames. We obtain the other boundaries in the same way. The clipping window allows us to ensure valid information in the video as shown in Fig. 6.

Fig. 6 Clipping window boundary. We identify the rightmost left boundary, the leftmost right boundary, the topmost bottom boundary, and the lowest top boundary of all frames as the boundaries of the clipping window.

4 Experimental Results and Discussion

Our algorithm was implemented and tested on a desktop PC with 1.05 GHz GPU and 4 GB RAM. For a (640×360) resolution stereoscopic video with 135 frames, the average time for preprocessing (video segmentation, saliency detection, feature tracking), ideal rotation estimation, optimization, and clipping are 1351.23, 5.06, 157.61, and 0.01 seconds, respectively. For a fair comparison, most shaky videos used in the related works were tested in our experiments. In addition, we also use our stereoscopic camera to obtain more shaky stereoscopic videos. Those various videos including strong occlusions (Cases 1 and 2 in Fig. 7), dynamic backgrounds (Cases 1 and 3-8 in Fig. 7) and parallax effects (Cases 5-8 in Fig. 7) were tested for representing the usefulness of the proposed method. Figs. 7-11 and supplemental videos show more results and comparisons.

Effectiveness. Several tests were conducted for discussing the effects of rotation adjustment, translation smoothing, and disparity maintenance. To demonstrate the feasibility of the rotation adjustment, stabilizing with and without rotation process was tested as shown in Fig. 8. The rotation angles between consecutive frames become more smooth and the rotation velocity is reduced. Fig. 9 reveals the usability of trajectory cutting or not in translation smoothing. The smooth path with trajectory cutting is more similar to the original path. In addition, if we smooth the videos for each eye separately, the disparity between left and right videos will be erroneous. Therefore, a quantitative analysis was conducted to evaluate the disparity preservation of our results, as follows:

$$D_{err} = \frac{1}{N_f} \sum_{i=1}^{N_f} \|(S_i^L - S_i^R) - (\tilde{S}_i^L - \tilde{S}_i^R)\|, \tag{30}$$

where D_{err} is the average difference of source feature pairs and that of stabilized feature pairs. $(S_i^L - S_i^R)$ denotes the ith feature pair in source frames, and $(\tilde{S}_i^L - \tilde{S}_i^R)$ denotes the ith feature pair in stabilized results. N_f represents the total number of feature pairs. From Fig. 10, the difference D_{err} between feature pairs of source and result in each frame is minimized.

Comparisons. In this part, we compare our results with those of other studies, mainly [23] and [15], as shown in Table 1. The execution time of our system is longer than other methods.

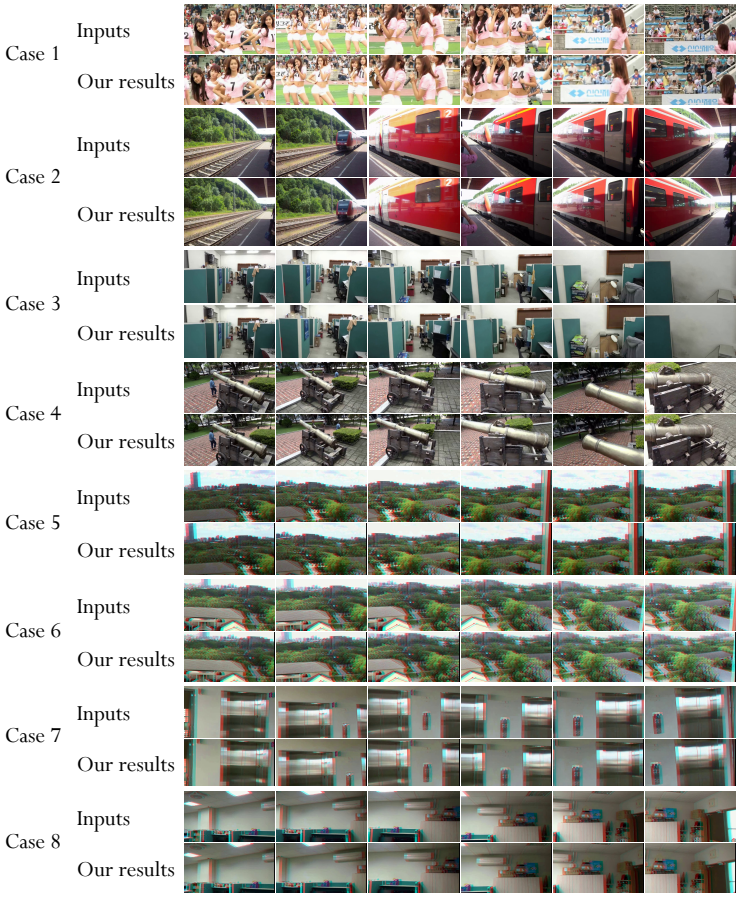

Fig. 7 Results of our approach. Each case includes the selected source frames and corresponding results.

The proposed method performs well in terms of shape preservation, disparity preservation, and clipping size. In [23] and [15], a frame is prewarped as an initial guess by using homography transformation. Although such approaches can obtain good results in most cases, the outcomes may have serious deformation given incorrect estimation of the homography transformation. Our method can handle more videos than that in [23], [15] and achieve stabilization. Video stabilization using homography transformation may occur unnatural deformation. Therefore, we adopt the rigid transformation to stabilize a video. We can stabilize a video in a simple manner, and preserve the shape of objects and more video content (see Fig. 11). For more comparisons, please refer to our supplementary video. Moreover, we calculate the average rate of retained content of the stabilized results by using Wang et al. [23] and our proposed method. Table 2 shows that our clipping rate is 21.3% and better than that of [23].

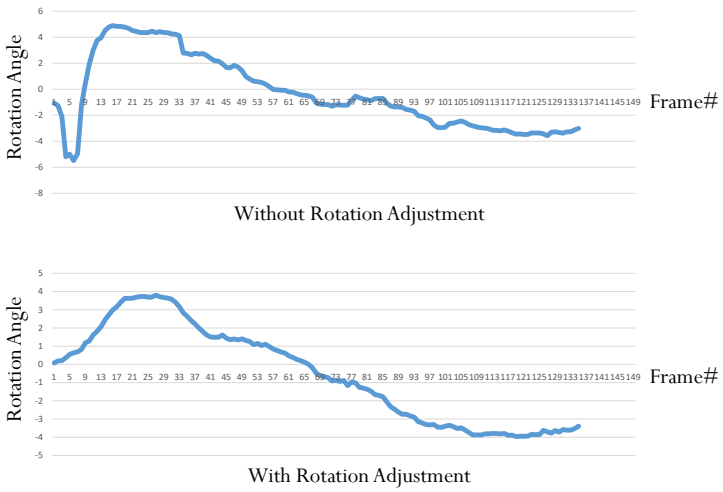

Fig. 8 Comparison between stabilizing without (top) and with (bottom) rotation adjustment.

Table 1 Comparisons with the related approaches including [23] and [15].

	Stabilization effect	Shape preservation	Disparity preservation	Clipping size	Computation time
[15]	Normal	Normal	n/a	Large	Medium
[23]	Normal	Normal	n/a	Medium	Fast
Ours	Better	Better	Yes	Small	Long

Table 2 Video content maintenance. We compare our results with the study [23] and show that our method can keep more video contents.

	Wang et al. (2013)	Ours
Averagerate of retained content for 7 videos	68.1%	78.7%

5 Conclusions, Limitation and Future Work

In this study, we propose a practical approach for video stabilization that facilitates handling normal and stereoscopic videos. For the rotation, we developed a path that would allow the trajectories to become similar to the path videographers want. We obtained different results by using various rotation methods. For example, we create the result with stable rotation which is a meaningful rotation. For the trajectory smoothing, we explored the cutting of the trajectories according to the video content. Such approach made the smoothing path become similar to the original path while the trajectory was smoothened. Given the energy term of shape preservation, we maintained the shape of objects after performing stabilization. We also maintained the disparity between the left and right eyes in processing the video, which led to flawless 3D quality.

Our method has the following limitation. If an input video has quick rotation or translation, our system will possibly produce unsatisfactory results as shown in Fig. 12. Comparison with the related method [23], our method is better than their method in terms of the shape preservation of salient objects. In addition, in the aspect of the cropping ratio, ours is also better than their method. Nevertheless, we think that the cropping region of ours is

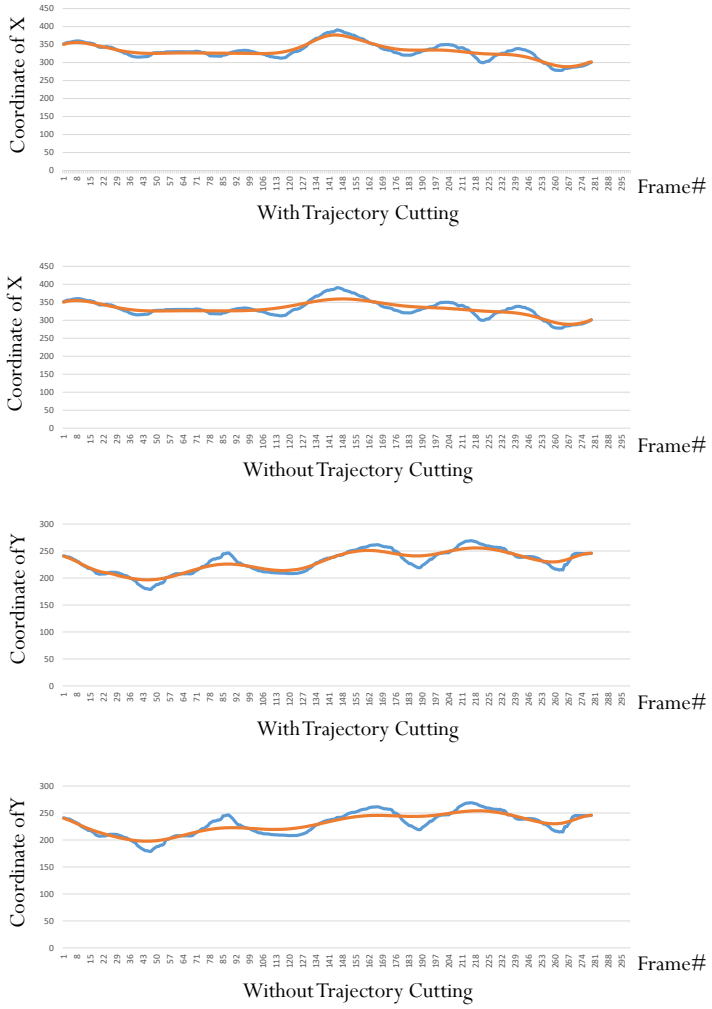

Fig. 9 Comparison between translation smoothing with and without trajectory cutting. The blue line is original video path and the orange line is the path after translation smoothing.

still large, so we treat this result as a bad result. On the other hand, the method with rotation solving we discussed entails letting the rotational speed become the fixed value, but another technique can produce a better effect. Moreover, the weights of the features are equal to the importance of objects only so far. We can consider the feature weight with the length or the stability of a feature trajectory in future research. In addition, the preprocessing methods of this study require more time for calculation, so we will use other methods techniques instead of them.

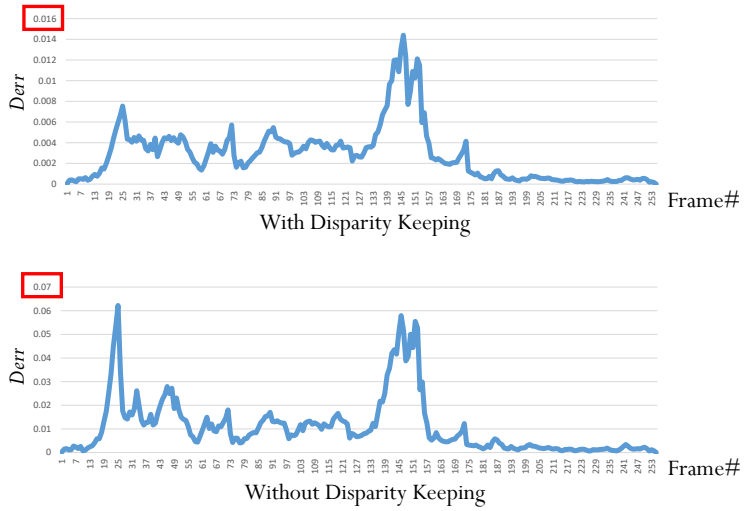

Fig. 10 Comparison between stabilizing with (top) and without (bottom) disparity energy term.

Fig. 11 Compared with the studies [23] and [15], we can maintain the shape of objects and keep more video content.

References

1. (Accessed date 30 June 2017) Voodoo camera tracker: A tool for the integration of virtual and real scenes. ftp://ftp.tnt.uni-hannover.de/pub/digilab/
2. Fan D, Wang W, Cheng M, Shen J (2019) Shifting more attention to video salient object detection. In: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp 8546–8556

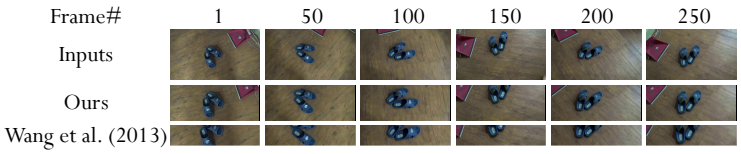

Fig. 12 Failure results. Our outcome is more smooth than the inputs, but the clipping region is too much. Fortunately, compared to the generated results by the study [23], our clipping content is less than that of [23].

3. Goferman S, Zelnik-Manor L, Tal A (2012) Context-aware saliency detection. IEEE Transactions on Pattern Analysis and Machine Intelligence 34(10):1915–1926
4. Goldstein A, Fattal R (2012) Video stabilization using epipolar geometry. ACM Transactions on Graphics 31(5):126:1–126:10
5. Grundmann M, Kwatra V, Han M, Essa I (2010) Efficient hierarchical graph-based video segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp 2141–2148
6. Grundmann M, Kwatra V, Essa I (2011) Auto-directed video stabilization with robust l1 optimal camera paths. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp 225–232
7. Grundmann M, Kwatra V, Castro D, Essa I (2012) Effective calibration free rolling shutter removal. pp 1–8
8. Guo H, Liu S, He T, Zhu S, Zeng B, Gabbouj M (2016) Joint video stitching and stabilization from moving cameras. IEEE Transactions on Image Processing 25(11):5491–5503
9. Guo H, Liu S, Zhu S, Zeng B (2016) Joint bundled camera paths for stereoscopic video stabilization. In: Proceedings of the IEEE International Conference on Image Processing, pp 1071–1075
10. Hartley R, Zisserman A (2003) Multiple View Geometry in Computer Vision. Cambridge University Press
11. Jia C, Sinno Z, Evans BL (2014) Real-time 3d rotation smoothing for video stabilization. In: Proceedings of the Asilomar Conference on Signals, Systems and Computers, pp 673–677
12. Lee KY, Chuang YY, Chen BY, Ouhyoung M (2009) Video stabilization using robust feature trajectories. In: Proceedings of the International Conference on Computer Vision, pp 1397–1404
13. Lin S, Lin C, Yeh I, Chang S, Yeh C, Lee T (2013) Content-aware video retargeting using object-preserving warping. IEEE Transactions on Visualization and Computer Graphics 19(10):1677–1686
14. Liu F, Gleicher M, Jin H, Agarwala A (2009) Content-preserving warps for 3d video stabilization. ACM Transactions on Graphics 28(3):44:1–44:9
15. Liu F, Gleicher M, Wang J, Jin H, Agarwala A (2011) Subspace video stabilization. ACM Transactions on Graphics 30(1):4:1–4:10
16. Liu S, Wang Y, Yuan L, Bu J, Tan P, Sun J (2012) Video stabilization with a depth camera. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp 89–95
17. Liu S, Yuan L, Tan P, Sun J (2013) Bundled camera paths for video stabilization. ACM Transactions on Graphics 32(4):78:1–78:10
18. Matsushita Y, Ofek E, Ge W, Tang X, Shum HY (2006) Full-frame video stabilization with motion inpainting. IEEE Transactions on Pattern Analysis and Machine Intelli-

gence 28(7):1150–1163
19. Morimoto C, Chellappa R (1998) Evaluation of image stabilization algorithms. In: Proceedings of the International Conference on Acoustics, Speech, and Signal Processing, vol 5, pp 2789–2792
20. Nie G, Cheng M, Liu Y, Liang Z, Fan D, Liu Y, Wang Y (2019) Multi-level context ultra-aggregation for stereo matching. In: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp 3278–3286
21. Smith BM, Zhang L, Jin H, Agarwala A (2009) Light field video stabilization. In: Proceedings of the IEEE International Conference on Computer Vision, pp 341–348
22. Wang W, Shen J, Xie J, Cheng M, Ling H, Borji A (2019) Revisiting video saliency prediction in the deep learning era. IEEE Transactions on Pattern Analysis and Machine Intelligence
23. Wang Y, Liu F, Hsu P, Lee T (2013) Spatially and temporally optimized video stabilization. IEEE Transactions on Visualization and Computer Graphics 19(8):1354–1361
24. Zhang L, Xu Q, Huang H (2017) A global approach to fast video stabilization. IEEE Transactions on Circuits and Systems for Video Technology 27(2):225–235