Turkish Journal of Electrical Engineering & Computer Sciences

http://journals.tubitak.gov.tr/elektrik/

Research Article

Turk J Elec Eng & Comp Sci
(2019) 27: 1260 – 1275

doi:10.3906/elk-1810-68

A novel accuracy assessment model for video stabilization approaches based on background motion

Md. Alamgir HOSSAIN, Tien-Dung NGUYEN, Eui-Nam HUH*
Department of Computer Science and Engineering, College of Electronics and Information, Kyung Hee University, Yongin-si, Republic of Korea

Received: 09.10.2018 • **Accepted/Published Online:** 21.12.2018 • **Final Version:** 22.03.2019

Abstract: In this paper, we propose a new accuracy measurement model for the video stabilization method based on background motion that can accurately measure the performance of the video stabilization algorithm. Undesired residual motion present in the video can quantitatively be measured by the pixel by pixel background motion displacement between two consecutive background frames. First of all, foregrounds are removed from a stabilized video, and then we find the two-dimensional flow vectors for each pixel separately between two consecutive background frames. After that, we calculate a Euclidean distance between these two flow vectors for each pixel one by one, which is regarded as a displacement of each pixel. Then a total Euclidean distance of each frame is averaged to get a mean displacement for each pixel, which is called mean displacement error, and finally we calculate the average mean displacement error. Our experimental results show the effectiveness of our proposed method.

Key words: Video stabilization assessment, video stability measurement, video quality assessment, video stabilization, mean displacement error

1. Introduction

The demand for video stabilization (VS) is increasing over time [1]. Now many video processing software programs and camcorders are using a video stabilization facility [2]. Most importantly, Apple uses gyroscopes on the iPhone and iPad, but the devices cannot discard an unexpected large translational motion [3]. Microsoft [4–6], Facebook [7], and Adobe [8] have been doing much research in this area, too. In addition, video stabilization has major applications in aerial video surveillance, geo-registration, autonomous vehicle navigation, model-based compression, structure from motion, motion analysis, and mosaicking [1]. The quality of a video depends on the accuracy of video stabilization. All are to satisfy the human eye. The human eye is very sensitive to motion frequency, amplitude, spatial image frequency, color, intensity, and context of a video [9–12]. Currently, a large number of video stabilization algorithms are using subjective measurements to validate the VS algorithm. Though subjective measurement has some usages in psycho-visual experiments, this evaluation strategy does not reflect much in scientific measurements. Unfortunately, the state-of-the-art video stabilization algorithms such as Subspace [13] and the L1 camera optimization path for rolling shutter removal [14] are evaluated based on optimal user experience. If spatial consistency is not achieved for a video, the video frame will suffer from noticeable unnatural seams [15]. Researchers observe that the nearest frame shows the smallest alignment error [15], whereas Liu et al.[16] showed that consecutive background pixels have similarity over 90%. Safdarnejad et

*Correspondence: johnhuh@khu.ac.kr

al. [17] mentioned that the predominant foreground and textureless region are the main cause of video stability failure. Most importantly, a foreground can moderately affect the accuracy of video stabilization [13, 15, 17]. Liu et al. [13] stated that a rolling shutter distortion is a noise and it cannot be modeled accurately. Grundmann et al. [18] mentioned that compensating the rolling shutter effect [3, 4, 19] caused by the complementary metal oxide semiconductor (CMOS) camera is a crucial task compared to a video captured a charge coupled device (CCD) camera in video stabilization.

Though the video stabilization algorithm tries to entirely remove unwanted motions present in a video, undesired motions always exist in the video. The sequences of video frame transformations in video stabilization algorithms follow a background of a reference frame until the reference frame is changed. If a video is perfectly stable, no residual motions will remain in the video except for the foreground motion [20], i.e. the difference between consecutive background frames will be zero. Any deviation from zero background motion will be due to inaccurate motion detection and estimation, transformation error, and accumulation error. Motivated by the warping process of video stabilization, in this paper, we estimate a background motion, which is the key to judging the stability of a video. Our hypothesis is that the background pixels in the coordinate (x_i, y_j, t) are equal to the background pixels in the coordinate $(x_i, y_j, t+dt)$ at time t and $t+dt$ respectively in the case of a nondynamic background. If the background pixels in the coordinate (x_i, y_j, t) are not equal to the background pixels in the coordinate $(x_i, y_j, t+dt)$ at time t and $t+dt$, respectively, we calculate each background pixel displacement between frames $I_t(x_i, y_j, t)$ and $I_{t+1}(x_i, y_j, t+dt)$. To compute background pixel displacement, the two-dimensional motion vectors in the horizontal direction and the vertical direction are calculated between background frames $I_t(x_i, y_j, t)$ and $I_{t+1}(x_i, y_j, t+dt)$, and then we calculate a Euclidean distance for the horizontal and vertical directional motion vectors of each pixel separately, which is the displacement of each pixel. Next, we compute a total sum of the L^2 norm distance of each pixel per frame. Finally, a total sum of the L^2 norm distance is averaged out to get a mean displacement error (MDE) that indicates an average motion of each background pixel present in the video. The overall architecture of our proposed algorithm is shown in Figure 1. The main contribution of this paper is twofold: the proposed method can quantitatively measure undesired motions present in the stabilized video, and we appraise the performance of three state-of-the-art VS algorithms using our proposed MDE or average mean displacement error (AMDE) algorithm. In addition to this, the accuracy of our proposed model is evaluated based on our synthetically created videos.

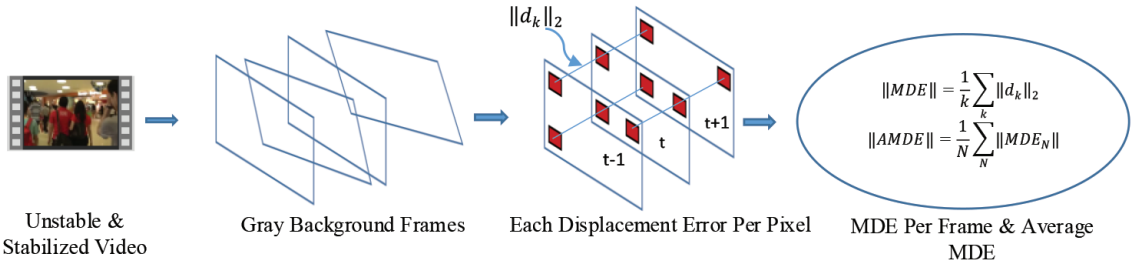

Figure 1. Overall structure of our algorithm.

2. Related works

Basically, the peak signal to noise ratio (PSNR) and the structural similarity (SSIM) index [21] are the two most important video quality assessment metrics that do not consider the jitters of the video [2, 21]. Mainly, the PSNR [20–23] measures the quality of a compressed video pixel by pixel. Morimoto et al. [20] calculated

interframe transform fidelity (ITF) and global transform fidelity (GTF) in the case of VS performance analysis. Here, fidelity refers to the fitting capability of a motion model to the actual motion, ITF is the amount of PSNR between two sequent frames, and GTF indicates PSNR between the ground truth frame and stabilized frame.

Mean opinion score (MOS) is calculated as a part of a subjective measurement [10, 11, 22, 23] to check the quality of a video. Niskanen et al. [24] considered divergence, jitter, and blur to measure VS accuracy. Here, jitter refers to a high-frequency motion in a video, whereas the divergence estimates latency between consecutive frames. Later, researchers considered the point spread function (PSF) to measure the blur introduced by the video stabilization method. In the end, the proposed method tests the accuracy of the two anonymous VS methods A and B. Mean square error (MSE) [2] between a reference path and a path after stabilization of a video is used to assess the stability of the VS method. Qu et al. [25] tested their proposed method based on synthetically shaking video to measure the accuracy of the three stabilization algorithms of Deshaker, Grundmann, and Qu. Zhang et al. [1] assessed the stability of intensity-based video stabilization and feature-based video stabilization. From the ground truth motion and the estimated motion, the researchers computed a mean square error (MSE) between consecutive frames and root mean square (RMS) error for the whole sequences. Zhai et al. [26] took a threshold to categorize a high-frequency component (jitter) and a low-frequency component (divergence) and decomposed these jitters and divergence utilizing a high-pass filter and low-pass filter. Furthermore, synthetic jitters were added to a stable video to get an unstable video, and finally the researchers assessed the stability of the video stabilization algorithm based on the MSE of the reference path and the processed path. A major limitation of these methods is that the synthetically created motion and the real undesired motion will not be the same. Zhang et al. [27] claimed that there are no methods that can measure the stability of a video. The researchers claimed that an unstable video will have more average motions than the average motions of a stable video. They computed the ratio of rotation and translation components of a stabilized video to rotation and translation components of a real video. They called this procedure a heuristic stability approach. Liu et al. [6] suggested an empirical process that assumes that if the more energy is in the low-frequency part of a video, the video will be more stable. We see that most existing proposed metrics [1, 2, 21, 25, 26] for VS approaches are either synthetically shaking video-based or heuristic approaches [2, 6, 24, 27].

3. Design methodology

In this section, we discuss our accuracy assessment model. The model has basic three steps: preprocessing and foreground mask generation, foreground removal and background, and MDE and AMDE model construction. Figure 2 depicts the details of our accuracy measurement model for the VS method.

1) Preprocessing and foreground mask generation: First, an unstable or stabilized video is input into our system. Then the unstable or stabilized video is converted into a gray-scale $I_t(x, y)$ video. We borrow the idea of foreground mask $F(x, y)$ generation from the SuBSENSE background subtraction algorithm, where St. Charles et al. [28] effectively modeled dynamic background. For each background pixel $BG(x, y)$, N number of recent samples for a background $BG(x, y)$ are stored as

$$BG(x, y) = \{BG_1(x, y), BG_2(x, y), BG_3(x, y), \ldots, BG_N(x, y)\}. \quad (1)$$

The foreground and background classification is shown in Eq. (2). Here, 1 and 0 represent the foreground

and background, respectively.

$$F_t(x,\ y) = \begin{cases} 1, & \text{if} \#\left\{dist(I_t(x,\ y),\ BG_n(x,\ y)) < R,\ \forall n\right\} < \#_{min} \\ 0, & \text{otherwise,} \end{cases} \tag{2}$$

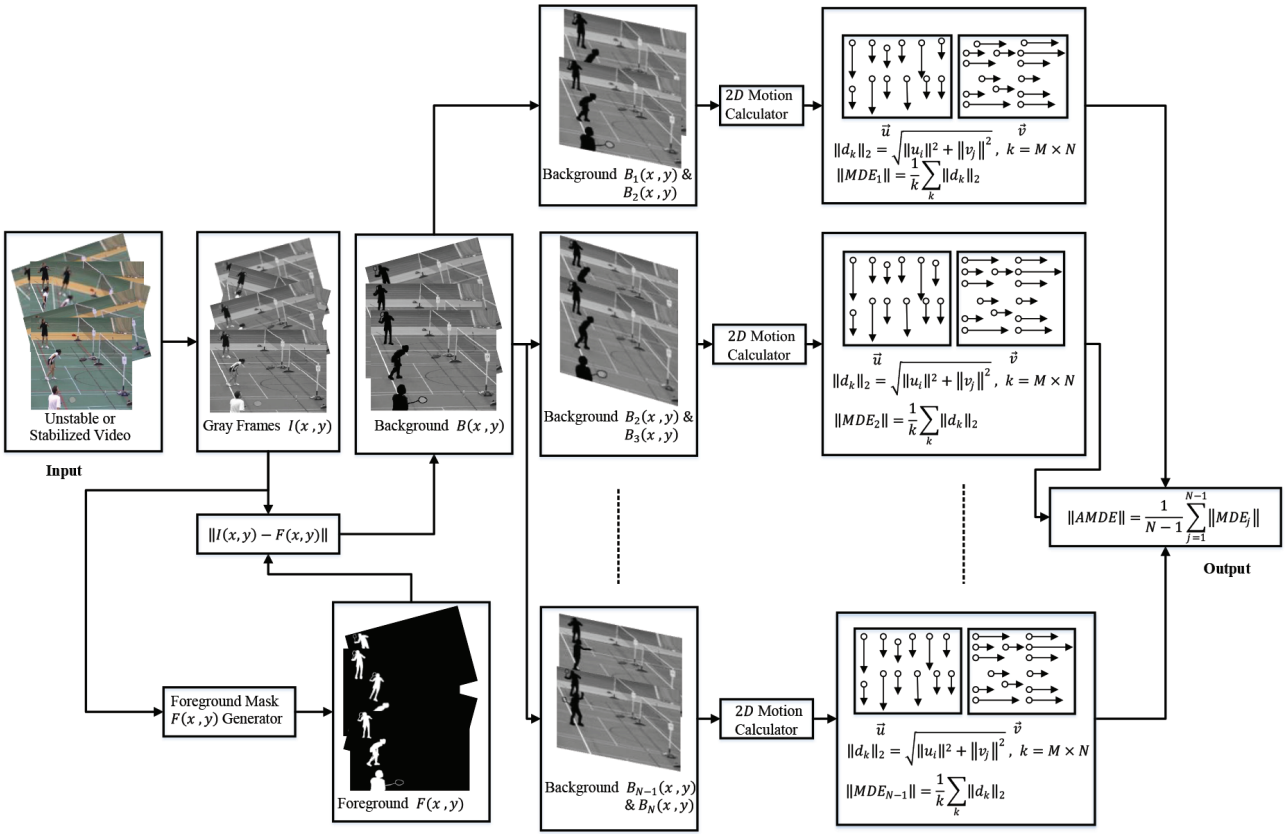

Figure 2. Accuracy assessment model for the VS algorithm.

where $F_t(x, y)$ is the foreground mask, $dist(I_t(x, y), BG_n(x, y))$ is the intensity and local binary similarity pattern $(LBSP)$ distance between the current observation $I_t(x,\ y)$ and $BG_n(x,\ y)$, R is the current updated decision threshold, and $\#_{min}$ denotes minimum matching requirement.

2) Foreground removal and background: We assume that $I(x,\ y)$, $B(x,\ y)$, and $F(x,\ y)$, are a gray-scale video frame, a background frame, and a foreground frame respectively at the coordinate $(x,\ y)$. At time t, we determine foreground mask $F_t(x,\ y)$ as in Eq. (2), and the background model $B_t(x,\ y)$ for the grayscale video $I_t(x,\ y)$ will be

$$B_t(x,\ y) = |I_t(x,\ y) - F_t(x,\ y)|. \tag{3}$$

At time $t+1$, we will get the following background:

$$B_{t+1}(x,\ y) = |I_{t+1}(x,\ y) - F_{t+1}(x,\ y)|. \tag{4}$$

Therefore, according to Eqs. (3) and (4), we will get the background frame sequence $B_1(x,\ y)$, $B_2(x,\ y)$, $B_3(x, y) \ldots$ $B_n(x, y)$ for the K number of gray-scale video sequences $I_k(x, y)$, where $\forall k \geq 2$ and $\forall n \geq 1$.

Then we calculate the vertical displacement vectors u and the horizontal displacement vectors v between the background $B_1(x,\ y)$ and $B_2(x,\ y)$, $B_2(x,\ y)$ and $B_3(x,\ y)$, $\ldots$, $B_{n-1}(x,\ y)$ and $B_n(x,\ y)$ for each pixel one by one at the coordinate $(x,\ y)$, where $\forall n \geq 2$.

3) MDE and AMDE model construction: Our proposed MDE or AMDE method models the undesired motion of the background pixels. We know that the optical flow algorithm takes the assumption that pixel intensities between consecutive frames do not change much and neighboring pixels have similar motion characteristics, so pixel intensities will be

$$I(x,\ y,\ t) = I(x + dx,\ y + dy,\ t + dt), \tag{5}$$

where dx and dy are the vertical and the horizontal distance in case of the pixel moving at the time dt (https://docs.opencv.org/3.3.1/d7/d8b/tutorial_py_lucas_kanade.html). Utilizing the Taylor series, discarding common elements, and dividing by dt in Eq. (5), we get the following equation:

$$f_x u + f_y v + f_t. \tag{6}$$

Here,

$$f_x = \frac{\partial f}{\partial x},\ f_y = \frac{\partial f}{\partial y},\ u = \frac{dx}{dt},\ v = \frac{dy}{dt}. \tag{7}$$

Therefore, we will get the optical flow vectors $\{(u_i,\ v_j) \mid (u_i,\ v_j) \in \{B_t(x,\ y),\ B_{t+1}(x,\ y)\},\ \forall t \geq 1\}$. We use Farnebäck's dense optical flow method [29] to calculate the vertical flow vector u and horizontal flow vector v for each pixel separately for the $M \times N$-pixel video frame. Now we estimate the displacement for the pixels between background $B_t(x,\ y)$ and background $B_{t+1}(x,\ y)$ by calculating the norm of the vectors u and v. The Manhattan distance (L^1 norm) is presented as

$$||d||_1 = |u_i| + |v_j|. \tag{8}$$

For the L^1 distance, the MDE is defined as

$$||MDE|| = \frac{1}{M \times N} \sum_{i=1}^{M} \sum_{j=1}^{N} ||u_i|| + ||v_j||. \tag{9}$$

We know that the Manhattan distance (L^1 norm) measures the shortest path to move horizontally and vertically, whereas the Euclidean (L^2 norm) distance computes the smallest distance in the plane. In our cases, we use the Euclidean (L^2 norm) distance. The L^2 norm, $||d_k||_2$, is calculated from the optical flow vectors $\vec{u} \in \{0, 1, 2, 3, \ldots, M\}$ and $\vec{v} \in \{0, 1, 2, 3, \ldots, N\}$, where $k = M \times N$:

$$||d_k||_2 = \sqrt{||u_i||^2 + ||v_j||^2}. \tag{10}$$

Therefore, the mean of the displacement of each pixel between the background frames $B_1(x,\ y)$ and $B_2(x,\ y)$ is

$$||MDE_1|| = \frac{1}{M \times N} \sum_{i=1}^{M} \sum_{j=1}^{N} \sqrt{||u_i||^2 + ||v_j||^2}. \tag{11}$$

The MDE between the background frames $B_2(x, y)$ and $B_3(x, y)$ is

$$||MDE_2|| = \frac{1}{M \times N} \sum_{i=1}^{M} \sum_{j=1}^{N} \sqrt{||u_i||^2 + ||v_j||^2}. \tag{12}$$

We continue the MDE calculation like in Eqs. (11) and (12), and finally we determine the MDE between background frames $B_{N-1}(x, y)$ and $B_N(x, y)$:

$$||MDE_{N-1}|| = \frac{1}{M \times N} \sum_{i=1}^{M} \sum_{j=1}^{N} \sqrt{||u_i||^2 + ||v_j||^2}. \tag{13}$$

The average of the MDEs is calculated for the N video frames as

$$\begin{aligned} ||AMDE|| &= \frac{1}{N-1}(||MDE_1|| + ||MDE_2|| + ||MDE_3|| + \cdots + ||MDE_{N-2}|| + ||MDE_{N-1}||) \\ ||AMDE|| &= \frac{1}{N-1} \sum_{j=1}^{N-1} ||MDE_{j-1}||. \end{aligned} \tag{14}$$

Algorithm 1. Video assessment model without the foreground.

Input : *Video was taken after stabilization*
Output: *MDE, AMDE*

read the *previousframe* from a video;
convert the *previousframe* into the gray scale;
remove foreground pixels from the *previousframe* using SubSENSE algorithm;
$amde \leftarrow 0$, $sum\_mde \leftarrow 0$, $count\_mde \leftarrow 0$;
while *video frame* $\neq 0$ **do**
    read the *nextframe*;
    if *nextframe empty* **then**
        break;
    else
        convert the *nextframe* into the gray scale;
        remove foreground pixels from the *nextframe* using the SubSENSE algorithm;
        calculate Gunnar Farnebäck's dense optical flow vectors u and v in the vertical and horizontal direction between the consecutive background *previousframe* and *nextframe*;
        $||d_k||_2 \leftarrow 0$, $sum\_d \leftarrow 0$, $mde \leftarrow 0$;
        for $i \leftarrow 1$ **to** N **do**
            for $j \leftarrow 1$ **to** M **do**
                $||d_k||_2 \leftarrow sqrt(u * u + v * v)$;
                $sum\_d \leftarrow sum\_d + ||d_k||_2$;
            end
        end
        $mde \leftarrow sum\_d/(N * M)$;
        $count\_mde++$;
        print mde;
        $sum\_mde \leftarrow sum\_mde + mde$;
        $previousframe \leftarrow nextframe$;
    end
end
$amde \leftarrow sum\_mde/count\_mde$;
print amde;

Finally, our video assessment algorithm is organized in order: Algorithm 1 displays the video assessment model without the foreground. In the case of video assessment with the foreground, it needs to escape lines 3 and 11 of Algorithm 1. The displacement is calculated in pixels (unit = pixel). The MDE as in Eq. (11) is the mean displacement of each pixel per frame. If the background pixels have d pixels displacement from the background $B_t(x, y)$ in Eq. (3) to the background $B_{t+1}(x, y)$ in Eq. (4), we can claim that the foreground pixels $F_t(x, y)$ in Eq. (2) will also displace the same distance d pixels from the video frame $I_t(x, y)$ in Eq. (3) to the video frame $I_{t+1}(x, y)$ in Eq. (4), since all the pixels for a video frame move at the same rate when a camera unintentionally moves. The displacement of each pixel in pixels indicates the amount of undesired motion for each pixel separately. In practice, all pixels will not move the same amount for both types of pixels (background pixel, foreground pixel) in the case of a CMOS camera. The camera will have rolling shutter effects. That is why we find the separate background pixel movements and calculate the average movement of the pixels for each frame, which we call MDE. Therefore, we can conclude that the foreground pixel movement also follows the background MDE or AMDE as in Eq. (14) on an average. In this way, our proposed MDE and AMDE models measure the performance of the different video stabilization algorithms. The proposed method is absolutely effective, as our experiment proves. The displacement is due to estimation, transformation, and accumulation errors that come from different steps of the video stabilization process. Our proposed MDE in Eqs. (11)–(13) and AMDE in Eq. (14) estimate the error amount numerically. MDEs of ≈ 0 and AMDEs of ≈ 0 mean that there are no errors in the processed video. A better video stabilization method will have lower MDEs and AMDEs. The main goal of the VS method is to remove the unwanted motion present in the input video while maintaining the other perceptual video quality.

3.1. Methodology for synthetically shaking video

To validate our proposed method, we generate synthetically shaking videos having the geometric transformation of the images. Our created artificial video types are based on 1) translation, 2) rotation, 3) translation and rotation (TR), 4) rotation and translation (RT), and 5) scaling, rotation, and translation (SRT). To create a shaking video, each pixel (x, y) of each frame is transformed by adding a translational vector $\langle h, k\rangle$. The transformed pixels (x', y') are $x' = x+h$, and $y' = y+k$. The transformation matrix as in Eq. (15) is obtained by adding a homogeneous coordinate. When a pixel $(x, y, 1)$ of a frame spins an angle θ about the origin, the transformed matrix for a rotation of each pixel $(x', y', 1)$ is represented as in Eq. (16). In the case of TR video, if a pixel $(x, y, 1)$ is translated first by a translational vector $\langle h, k\rangle$, followed by a rotation of an angle θ about the coordinate origin, the transformed matrix will be like Eq. (17). Regarding rotation translation, the matrix representation will be different as in Eq. (18). To create a more real shaking video, each pixel is transformed in terms of scaling, rotation, and translation. To generate a SRT video, the new pixel position (x', y') is transformed as $x' = sx\cos\theta - sy\sin\theta + h, \quad y' = sx\sin\theta + sy\cos\theta + k$, and its matrix is represented as in Eq. (19). We create all these types of videos to prove that our method can detect all kinds of the motion.

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & h \\ 0 & 1 & k \\ 0 & 0 & 1 \end{bmatrix} . \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} . \tag{15}$$

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} . \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} . \tag{16}$$

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & h\cos\theta - k\sin\theta \\ \sin\theta & \cos\theta & h\sin\theta + k\cos\theta \\ 0 & 0 & 1 \end{bmatrix} . \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} . \tag{17}$$

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & h \\ \sin\theta & \cos\theta & k \\ 0 & 0 & 1 \end{bmatrix} . \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} . \tag{18}$$

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} s\cos\theta & -s\sin\theta & h \\ s\sin\theta & s\cos\theta & k \\ 0 & 0 & 1 \end{bmatrix} . \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} . \tag{19}$$

4. Experimental results

This section explains experimental results of our proposed method, evaluates our proposed method based on synthetically created unstable videos and real videos, and appraises the performance of three of the state-of-the-art VS approaches using our proposed model. Finally, we compare our method with the PSNR and SSIM. We use Eqs. (11)–(13) and (14) to compute the MDE and AMDE, respectively, throughout our research.

4.1. Dataset

We use the changedetection dataset of 2014[1] to prove that the background motion between consecutive frames will be zero. Also, we create eight different types of videos of different frame dimensions using the images shown in Figures 3a–3h and based on the translation, rotation, composition of translation and rotation, composition of rotation and translation, and composition of scaling, rotation, and translation (SRT) transformation of Eqs. (15) to (19). Figures 3a–3d are images collected from the Internet, whereas images of 3a–3h are captured by a camera. All the images shown in Figure 3 are used only for creating the synthetically shaking videos.

The synthetically created shaking videos are only used to ensure our proposed method's accuracy. Afterward, we select five categories of videos (regular, quick rotation, crowd, parallax, and running) with the foreground objects from the video database.[2] All the categories of the videos are very complex except the regular category. The quick rotation category has very large motion among the other categories. We select three digital video stabilization software programs: VirtualDub-Deshaker,[3] Google YouTube Stabilizer,[4] and Adobe Premiere Pro Warp Stabilizer.[5] Adobe Premiere Pro generally applies the subspace video stabilization [13] approach and the YouTube Stabilizer uses the $L1$ optimal camera path algorithm [14]. The comparison of the three digital VS (DVS) approaches is performed by our method with the same dataset.[2] Finally, the comparison of our proposed approach with the PSNR and the SSIM is carried out by the MOS of the videos stabilized by the DVS algorithms.

[1] http://www.changedetection.net/
[2] http://liushuaicheng.org/SIGGRAPH2013/database.html
[3] https://www.guthspot.se/video/deshaker.htm
[4] http://www.youtube.com
[5] https://www.adobe.com/products/premiere.html

(a) left (b) right (c) lenna (d) schefflera

(e) flower (f) street flower (g) tree (h) indoor

Figure 3. Video created using the images.

4.2. Background motion and our method's robustness

To evaluate our algorithm, we executed the experiments extensively. Our hypothesis is that the nondynamic background motion of a video will be zero if a video is perfectly stable. In Table 1, background motions (AMDE) of all the videos are going to approach zero. The motions are not exactly zero because each BG[6] of the videos is not completely static. We find 0.00019, 0.00051, 0.00034, and 0.00016 MDEs between the two same video frames for four different cases. Therefore, the above experiments prove that nondynamic background motions between consecutive frames will be zero.

Table 1. Background motion.

Sl.	Video name	Only_background motion (average MDE)
1.	Baseline / highway	0.0730
2.	Baseline / office	0.0782
3.	Baseline / pedestrians	0.0943
4.	Shadow / busstation	0.0691
5.	Shadow / backdoor	0.0816

Based on this motionless background criterion, we can measure the stability of a video. We also analyze the unstable and stabilized videos of the four different categories provided in the above mentioned dataset. The analysis is tabulated in Table 2 and depicted in Figure 4. All the videos have undesired background motions. The background motion of the simple case video is less than the other cases. For the crowd and the running videos, the unwanted background motions are larger than the others. The observation is that it is easy to

[6]Background.

Table 2. BG motion of the unstable and stabilized videos.

Sl.	Video name	Unstable_BG_ motion (AMDE)	Stable_BG _ motion (AMDE)
1.	Simple / 0.avi	2.125	0.321
2.	Simple / 2.avi	2.423	0.207
3.	Simple / 4.avi	0.975	0.582
4.	Simple / 5.avi	1.451	0.255
5.	Simple / 8.avi	1.812	0.311
6.	Crowd / 1.avi	2.343	1.693
7.	Crowd / 2.avi	1.762	1.032
8.	Crowd / 4.avi	1.725	0.841
9.	Crowd / 14.avi	1.837	0.721
10.	Parallax / 5.avi	1.322	0.427
11.	Parallax / 13.avi	2.007	0.966
12.	Parallax / 17.avi	2.181	0.511
13.	Running / 3.avi	3.181	1.334
14.	Running / 15.avi	2.996	1.022
15.	Running / 16.avi	2.678	0.788

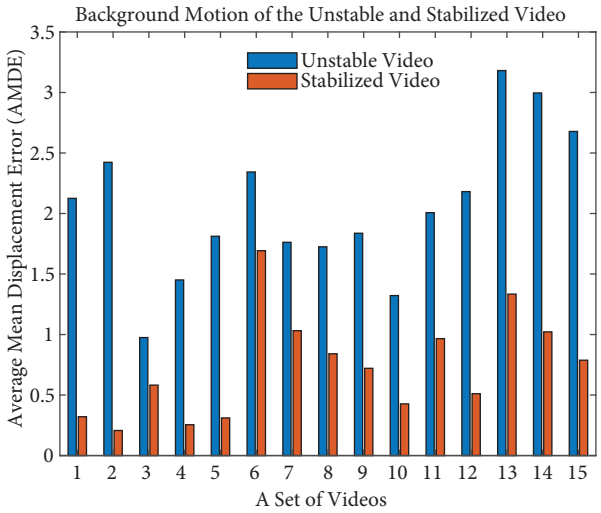

Figure 4. Unstable BG and stabilized BG for videos.

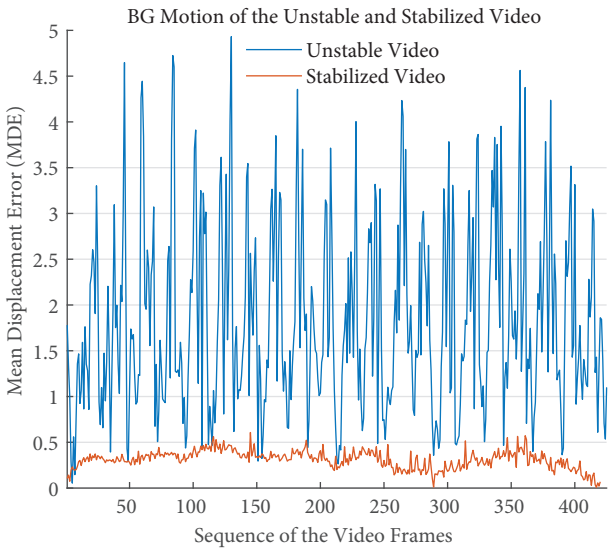

Figure 5. Unstable BG and stabilized BG for frames.

discard the simpler motion than complex motion. In Figure 5, we depict frame by frame MDEs of an unstable and stabilized video named 8.avi of the regular category provided online.[2] Clearly, the MDEs of the unstable video are higher than the MDEs of the stabilized video. Therefore, we can check the stability of a video based on the background motion.

Afterward, to test our method's accuracy, we compare the ground truth values with the measured values of the synthetically shaking videos. The ground truth values and the measured values are tabulated in the experiments in the case of translation, rotation, TR, RT, and SRT in Eqs. (15) to (19). The average errors in the case of a single translation and single rotation are 0.023 and 0.084 (Table 3 and Table 4), respectively,

whereas the average errors of both composite transformation TR and RT are 0.12 (Table 5 and Table 6). Though the composite translation and rotation (TR) and the composite rotation and translation (RT) are not commutative, our results (Table 5 and Table 6) have no noticeable differences. In the case of the SRT video, the average error is 0.13 (Table 7). Moreover, the error is increasing for the composite transformation compared to the single translation and the single rotation. In the above mentioned five cases, In the above mentioned five cases, the differences between the ground truth AMDEs and our measured AMDEs are very small, which are negligible. From Tables 3 to 7, it is concluded that the translational displacement is easyier to detect than the scaling and rotational displacement. Therefore, our proposed method is robust and it can detect the background motion, which will quantify the accuracy of the VS methods.

Table 3. Accuracy for translation.

Sl.	Video name	Ground truth (AMDE)	Our result (AMDE)	Diff. (AMDE)
1.	leftT	1.414	1.390	0.024
2.	rightT	1.414	1.406	0.008
3.	lenaT	1.414	1.329	0.085
4.	flowerT	1.414	1.415	0.001
5.	sflowerT	1.414	1.416	0.002
6.	treeT	1.414	1.420	0.006
7.	indoorT	1.414	1.377	0.037
			Average error =	**0.023**

Table 4. Accuracy for rotation.

Sl.	Video name	Ground truth (AMDE)	Our result (AMDE)	Diff. (AMDE)
8.	leftR	5.511	5.373	0.138
9.	rightR	5.511	5.391	0.120
10.	lenaR	2.993	2.862	0.131
11.	flowerR	5.133	5.134	0.001
12.	sflowerR	5.386	5.395	0.009
13.	treeR	5.040	5.079	0.039
14.	indoorR	5.199	5.044	0.155
			Average error =	**0.084**

Table 5. Accuracy for TR.

Sl.	Video name	Ground truth (AMDE)	Our result (AMDE)	Diff. (AMDE)
1.	leftTR	5.628	5.669	0.041
2.	rightTR	5.628	5.687	0.061
3.	lenaTR	3.330	3.122	0.208
4.	ScheffaTR	6.546	6.418	0.128
5.	sflowerTR	5.482	5.704	0.222
6.	indoorTR	5.272	5.369	0.097
			Average error =	**0.126**

Table 6. Accuracy for RT.

Sl.	Video name	Ground truth (AMDE)	Our result (AMDE)	Diff. (AMDE)
1.	leftRT	5.606	5.648	0.042
2.	rightRT	5.606	5.662	0.056
3.	lenaRT	3.310	3.104	0.206
4.	ScheffaRT	6.524	6.397	0.127
5.	sflowerRT	5.460	5.682	0.222
6.	indoorRT	5.250	5.349	0.099
			Average error =	**0.125**

Table 7. Accuracy for SRT.

Sl.	Video name	Ground truth (AMDE)	Our result (AMDE)	Diff. (AMDE)
7.	leftSRT	7.388	7.368	0.020
8.	rightSRT	7.388	7.396	0.008
9.	flowerSRT	6.770	7.037	0.267
10.	sflowerSRT	7.185	7.319	0.134
11.	indoorTR	6.885	7.108	0.223
			Average error =	**0.130**

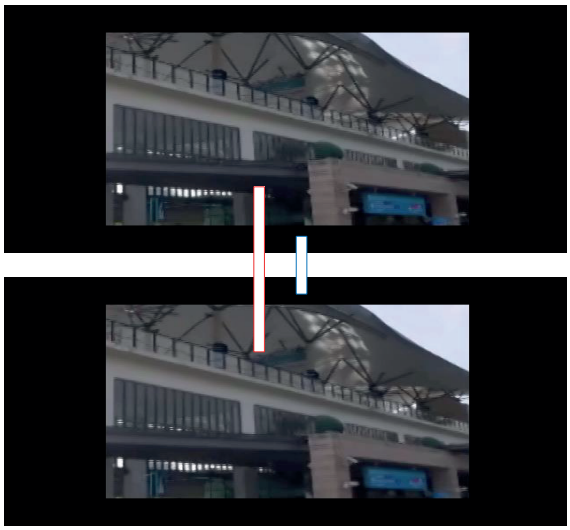

Figure 6. Stability assessment for unfilled case.

Remark 1 *When a video is stabilized, some methods crop the video like in Figure 6 due to the large motion. The red line shows the filled image area and the blue line indicates the unfilled area. Our proposed method only considers the filled image area in terms of stability of a video. The proposed MDE calculates the total motion pixel by pixel between the two consecutive frames, then divides by the frame size. In the case of Figure 6, the displacements of the black portion (the unfilled area) will be zero. Therefore, we only estimate the displacement of the filled image area and divide the total displacement by the filled image area size. Another important issue is that the frame rate should be the same when comparing the DVS algorithm.*

4.3. Comparison of three DVS methods

The performance of the three DVS methods (the VirtualDub-Deshaker video stabilizer, the YouTube stabilizer, and the Adobe warp stabilizer) is measured in terms of the background motion using our proposed model. The dataset is taken from a website.[2] The experiment is carried out on five different categories without foreground videos. The summary of the performance is shown in Tables 8 to 12. All the methods fail to discard the undesired motion entirely. The three DVS methods reduce more undesired motions for the regular case video than the other cases. The average background motion is less than 0.65 pixel AMDE for the regular case videos, whereas the other four cases of videos have more than 1.30 pixel AMDE background motion. On the average, the YouTube stabilizer shows the best performance for all cases while the VirtualDub-Deshaker stabilizer gains better performance and the the Adobe warp stabilizer scores well in all cases.

4.4. Our method compared with the other methods

We compare our proposed method with the PSNR[7] and SSIM [21] based on the MOS of the five different categories of videos stabilized by the three DVS algorithms. To the best of our knowledge, there is no dataset with the MOS for video stabilization. Liu only provides some unstable and stabilized videos of different categories at the url,[2] but the MOS is not included there. Therefore, for the MOS, we use the Double Stimulus Continuous

[7] https://docs.opencv.org/2.4/doc/tutorials/highgui/video-input-psnr-ssim/video-input-psnr-ssim.html

Quality Scale (DSCQS)[8] of the ITU-R standard to assess the perceptual quality in terms of the stability of a video. According to the DSCQS, when taking a test, an input and processed video are displayed in the consecutive windows. Then the observer directly assesses and puts a score according to the given scale. The rating scale of 1–5 is used, where we designate 1 as excellent (perfect stability quality) and 5 as bad (very difficult to understand the stability quality). The other scores of 2, 3, and 4 indicate good (very satisfactory), fair (requires more stability), and poor (hard to understand the stability), respectively. We took 22 observers, where two were female and rest of them were male. All the observers were students from five different labs who came from two different countries. Among all the observers, some observers had experience in video and image processing, computer vision, and video stabilization and rest of them were from artificial intelligence, cloud computing, and big data analysis labs. For a single unstable input video, three videos stabilized by the three DVS algorithms were displayed consecutively. The participants were asked to watch the videos carefully and rate the overall stability according to the above mentioned rating scale. Finally, the collected scores from the 22 participants were averaged to get the final assessment MOS value. The Spearman rank correlation coefficient (S-corr.) was used to estimate the correlation between our proposed method, the PSNR, and the SSIM with the MOS of the stabilized video separately. The S-corr. shows the correlation between the two variables. When the two variables increase, the S-corr. becomes positive. Alternatively, if one variable is increasing while the other variable is decreasing, S-corr. will be negative. S-corr. of zero value means that there is no correlation between the two variables. When the two variables are perfectly monotonically correlated, the S-corr. becomes one.

The comparisons of our proposed method with the PSNR and SSIM are tabulated in Tables 8 to 12 in terms of the correlation with the MOS. The correlation is computed for each video of each category separately. In addition to the averages of the MOS, our MDE, the PSNR, and the SSIM are computed, and then the S-corr. values for these average values are also estimated. Among the three methods, our method gains the best correlation results, whereas the SSIM shows fewer correlation coefficients and the PSNR is in the second position in the case of the five different categories where each category belongs to five different videos. Our proposed method has the perfect S-corr. value of 1 in all cases. The PSNR achieves the S-corr. value of 1 only for the two videos of the regular case and the quick rotation case separately, while the SSIM gives one S-corr. value of 1 for each case and both the PSNR and SSIM score the S-corr. value of 0.5 for the rest of the videos. The three methods obtain an average S-corr. value of 1 only for the quick rotation case. The PSNR is usually used to check the image reconstruction quality and the SSIM estimates the image structural degradation. These two methods can only measure the distortion of an image. When an unstable video is stabilized, most of the pixels change their coordinate positions. Therefore, the PSNR and SSIM fail to estimate the stability of a video. However, our proposed method can count the stability criterion. Our experimental results and MOS survey data can be found online.[9]

5. Conclusion and future work

We have proposed a background motion-based novel performance metric, which can measure pixel-wise motion errors of the stabilized video. Our hypothesis is that background pixels do not move too much in the consecutive frames after video stabilization if a video is nearly perfectly stabilized. Similarly, the unwanted motions of foreground pixels always follow the background pixels' movement. To get a more accurate result, we remove the foreground pixels from a stabilized video. Afterwards, 2-dimensional motions are detected, and then the L^2

[8] https://tech.ebu.ch/docs/techreview/trev_271-evain.pdf
[9] https://github.com/alamgir39/khuresearch

Table 8. Our method compared with the PSNR and SSIM based on the MOS of the videos stabilized by the three DVS algorithms (regular case).

Video	Deshaker				Youtube				Adobe Warp				S-corr.		
	MOS	OUR	PSNR	SSIM	MOS	OUR	PSNR	SSIM	MOS	OUR	PSNR	SSIM	OUR	PSNR	SSIM
#1	2.23	0.40	10.75	14.02	1.68	0.33	10.24	13.28	3.68	1.16	10.35	14.54	**1**	0.5	**1**
#2	2.59	0.34	13.65	28.55	1.82	0.24	13.42	28.55	2.41	0.31	13.61	29.94	**1**	**1**	0.5
#3	3.00	0.44	12.93	34.77	1.68	0.20	12.33	34.43	2.77	0.35	12.39	35.33	**1**	**1**	0.5
#4	2.50	0.14	15.84	56.01	1.50	0.13	14.11	51.88	3.50	0.24	14.94	54.44	**1**	0.5	0.5
#5	2.50	0.43	14.15	21.29	1.41	0.16	12.70	19.10	3.50	1.07	12.57	20.17	**1**	0.5	0.5
AVG.	2.56	**0.35**	13.46	30.93	1.62	**0.21**	12.56	29.45	3.17	**0.63**	12.77	30.88	**1**	0.5	0.5

Table 9. Our method compared with the PSNR and SSIM based on the MOS of the videos stabilized by the three DVS algorithms (quick rotation case).

Video	Deshaker				Youtube				Adobe Warp				S-corr.		
	MOS	OUR	PSNR	SSIM	MOS	OUR	PSNR	SSIM	MOS	OUR	PSNR	SSIM	OUR	PSNR	SSIM
#1	2.91	1.74	12.04	22.60	1.73	1.03	11.89	22.13	2.41	1.40	11.84	23.60	**1**	0.5	0.5
#2	3.09	2.10	11.16	19.28	2.18	1.30	11.49	20.31	2.82	1.83	9.77	17.37	**1**	0.5	0.5
#3	2.95	2.93	11.51	19.95	2.68	2.24	11.77	19.97	2.77	2.93	7.54	4.84	**1**	0.5	0.5
#4	2.95	2.66	10.41	35.29	2.41	1.72	10.77	36.22	3.41	3.37	6.63	12.20	**1**	**1**	**1**
#5	2.45	1.03	12.66	53.77	2.18	0.93	12.71	52.99	3.59	1.61	8.25	30.62	**1**	**1**	0.5
AVG.	2.87	**2.09**	11.56	25.66	2.24	**1.44**	11.72	30.32	3.00	**2.23**	8.81	17.72	**1**	**1**	**1**

Table 10. Our method compared with the PSNR and SSIM based on the MOS of the videos stabilized by the three DVS algorithms (crowd case).

Video	Deshaker				Youtube				Adobe Warp				S-corr.		
	MOS	OUR	PSNR	SSIM	MOS	OUR	PSNR	SSIM	MOS	OUR	PSNR	SSIM	OUR	PSNR	SSIM
#1	2.36	1.64	13.34	36.53	1.64	1.49	12.59	33.77	3.27	2.17	12.00	33.65	**1**	0.5	0.5
#2	2.36	2.50	11.50	32.98	1.86	2.17	12.17	34.03	3.50	2.85	10.81	30.45	**1**	**1**	**1**
#3	2.50	1.72	10.84	28.85	1.77	1.21	10.79	28.54	3.18	2.13	10.35	27.19	**1**	0.5	0.5
#4	2.59	2.13	11.50	34.68	1.91	2.03	10.57	31.85	3.45	2.58	10.82	32.55	**1**	0.5	0.5
#5	2.59	2.16	9.65	27.59	2.09	2.13	9.59	27.70	3.68	2.96	9.41	25.81	**1**	0.5	**1**
AVG.	2.48	**2.03**	11.40	32.13	1.85	**1.80**	11.14	31.18	3.42	**2.54**	10.68	29.93	**1**	0.5	0.5

Table 11. Our method compared with the PSNR and SSIM based on the MOS of the videos stabilized by the three DVS algorithms (parallax case).

Video	Deshaker				Youtube				Adobe Warp				S-corr.		
	MOS	OUR	PSNR	SSIM	MOS	OUR	PSNR	SSIM	MOS	OUR	PSNR	SSIM	OUR	PSNR	SSIM
#1	2.23	2.02	11.44	33.80	1.55	1.87	12.24	37.09	2.91	2.38	12.32	37.95	**1**	0.5	0.5
#2	2.86	0.91	16.34	45.82	1.68	0.45	14.12	41.22	3.64	1.99	13.56	41.28	**1**	0.5	0.5
#3	2.77	2.41	11.43	37.75	1.73	2.33	11.17	35.44	3.00	2.48	11.44	37.92	**1**	**1**	**1**
#4	3.18	0.61	13.41	51.96	1.64	0.36	10.76	43.53	4.14	1.47	11.01	44.85	**1**	0.5	0.5
#5	2.73	1.90	9.32	26.10	2.00	1.61	9.76	26.73	3.64	1.93	9.39	26.07	**1**	0.5	0.5
AVG.	2.75	**1.57**	12.39	39.08	1.72	**1.32**	11.61	36.80	3.46	**2.05**	11.54	37.61	**1**	0.5	0.5

Table 12. Our method compared with the PSNR and SSIM based on the MOS of the videos stabilized by the three DVS algorithms (running case).

Video	Deshaker				Youtube				Adobe Warp				S-corr.		
	MOS	OUR	PSNR	SSIM	MOS	OUR	PSNR	SSIM	MOS	OUR	PSNR	SSIM	OUR	PSNR	SSIM
#1	2.77	3.08	10.35	27.78	1.95	2.98	10.30	26.92	4.00	3.23	9.50	23.21	**1**	0.5	0.5
#2	2.45	1.46	14.12	48.34	1.45	1.04	12.45	44.97	3.50	1.99	13.48	46.20	**1**	0.5	0.5
#3	2.73	1.59	10.91	45.88	2.00	1.47	11.16	46.56	4.00	1.71	9.59	40.39	**1**	**1**	**1**
#4	2.77	2.50	9.34	26.40	1.73	1.99	9.15	26.20	3.95	2.69	9.08	24.15	**1**	0.5	0.5
#5	2.77	2.07	11.27	30.89	1.68	1.71	11.19	30.28	3.64	2.73	10.41	28.03	**1**	0.5	0.5
AVG.	2.70	**2.14**	11.20	35.86	1.76	**1.84**	10.85	34.99	3.82	**2.47**	10.41	32.40	**1**	0.5	0.5

norm distance displacement is computed. As a result, our proposed algorithm detects a temporal inconsistency effectively and distinguishes between a stable and unstable video clearly (Figure 5). Moreover, the small differences between the experimental results and the ground truths (Tables 3 to 7) show our proposed method's robustness. Our proposed model sufficiently differentiates the three best DVS methods (Tables 8 to 12). The distinguished results of the three latest algorithms in the case of real videos and our method's validation by the synthetically shaking videos show the effectiveness of our algorithm. Most importantly, our proposed MDE method shows the best correlation scores with the MOS compared to the PSNR and the SSIM (Tables 8 to 12). Our proposed algorithm neither depends on the synthetically shaking video nor assumes heuristic or empirical assumptions. In the future, we plan to consider the blurring effect introduced by the video stabilization and the unfilled spatial areas in addition to the stability criterion to distinguishably measure the accuracy of the DVS method.

Acknowledgment

This research was supported by the Ministry of Science and ICT, Korea, under the Information Technology Research Center support program (IITP-2018-2013-1-00717) supervised by the Institute for Information & Communications Technology Promotion.

References

[1] Zhang C, Chockalingam P, Kumar A, Burt P, Lakshmikumar A. Qualitative assessment of video stabilization and mosaicking systems. IEEE Work App Comp 2008; 1-6.

[2] Tanakian MJ, Rezaei M, Mohanna F. Camera motion modeling for video stabilization performance assessment. 7th Iran Conf Mach 2011; 1-4.

[3] Karpenko A, Jacobs D, Baek J, Levoy M. Digital video stabilization and rolling shutter correction using gyroscopes. CSTR 2011; 2.

[4] Baker S, Bennett E, Kang SB, Szeliski, R. Removing rolling shutter wobble. Proc CVPR IEEE 2010; 2392-2399.

[5] Liu S, Wang Y, Yuan L, Bu J, Tan P, Sun, J. Video stabilization with a depth camera. Proc CVPR IEEE 2012; 89-95

[6] Liu S, Yuan L, Tan P, Sun J. Bundled camera paths for video stabilization. ACM T Graphic 2013; 78.

[7] Kopf J. 360 video stabilization. ACM T Graphic 2016; 195.

[8] Liu F, Gleicher M, Jin H, Agarwala, A. Content-preserving warps for 3D video stabilization. ACM T Graphic 2009; 44.

[9] Oshima M, Hayashi T, Fujioka S, Inaji T, Mitani H, Kajino J, Ikeda K, Komoda K. VHS camcorder with electronic image stabilizer. IEEE T Consum Electr 1989; 749-758.

[10] Winkler S. Issues in vision modeling for perceptual video quality assessment. Signal Processing 1999; 231-252.

[11] Wang Z, Sheikh HR, Bovik AC. Objective video quality assessment. The Handbook of Video Databases: Design and Applications 2003; 1041-1078.

[12] Wang Y, Ostermann J, Zhang YQ. Digital Video Processing and Communications. Hoboken, NJ, USA: Prentice Hall, 2001.

[13] Liu F, Gleicher M, Wang J, Jin H, Agarwala A. Subspace video stabilization. ACM T Graphic 2011; 4.

[14] Grundmann M, Kwatra V, Essa I. Auto-directed video stabilization with robust l1 optimal camera paths. Proc CVPR IEEE 2011; 225-232.

[15] Matsushita Y, Ofek E, Ge W, Tang X, Shum HY. Full-frame video stabilization with motion inpainting. IEEE T Pattern Anal 2006; 1150-1163.

[16] Liu S, Yuan L, Tan P, Sun J. Steadyflow: Spatially smooth optical flow for video stabilization. Proc CVPR IEEE 2014; 4209-4216.

[17] Safdarnejad SM, Atoum Y, Liu X. Temporally robust global motion compensation by keypoint-based congealing. European Conference on Computer Vision 2016; 101-119.

[18] Grundmann M, Kwatra V, Castro D, Essa I. Calibration-free rolling shutter removal. IEEE Conference on Computational Photography 2012; 1-8.

[19] Forssén PE, Ringaby E. Rectifying rolling shutter video from hand-held devices. Proc CVPR IEEE 2010; 507-514.

[20] Morimoto C, Chellappa R. Evaluation of image stabilization algorithms. Int Conf Acoust SPEE 1998; 2789-2792.

[21] Wang Z, Bovik AC, Sheikh HR, Simoncelli EP. Image quality assessment: from error visibility to structural similarity. IEEE T Image Process 2004; 600-612.

[22] Kinape RM, Amorim MF. A study of the most important image quality measures. International Conference of the Engineering in Medicine and Biology Society 2003; 934-936.

[23] Eskicioglu AM, Fisher PS. Image quality measures and their performance. IEEE T Commun 1995; 2959-2965.

[24] Niskanen M, Silvén O, Tico M. Video stabilization performance assessment. IEEE Int Con Multi 2006; 405-408.

[25] Qu H, Song L, Xue G. Shaking video synthesis for video stabilization performance assessment. P Soc Photo-Opt Ins 2013; 1-6.

[26] Zhai B, Zheng J, Wang Y, Zhang C. A multi-scale evaluation method for motion filtering in digital image stabilization. Proc Int C Tools Art 2015; 682-688.

[27] Zhang L, Chen XQ, Kong XY, Huang, H. Geodesic video stabilization in transformation space. IEEE T Image Process 2017; 2219-2229.

[28] St. Charles PL, Bilodeau GA, Bergevin R. Subsense: A universal change detection method with local adaptive sensitivity. IEEE T Image Process 2015; 359-373.

[29] Farnebäck G. Two-frame motion estimation based on polynomial expansion. Scandinavian Conference on Image Analysis 2003; 363-370.