2016 23rd International Conference on Pattern Recognition (ICPR)
Cancún Center, Cancún, México, December 4-8, 2016

StableFlow: A Novel Real-Time Method for Digital Video Stabilization

Abdelrahman Ahmed, Mohamed S. Shehata
Department of Electrical and Computer Engineering
Memorial University of Newfoundland, St. John's, NL, Canada
ama727,mshehata@mun.ca

Abstract

Digital video stabilization is crucial in many applications such as object detection and tracking. It has been studied for decades yielding an extensive amount of literature in the field, however, current approaches suffer from either being computationally expensive or under-performing in terms of visual quality . In this paper, we present StableFlow, a novel real-time method that was inspired by the mass-spring-damper physical model. In StableFlow, a video frame is modelled as a mass suspended in each direction by a critically dampened spring and damper which can be fine-tuned to adapt with different shaking patterns. The proposed method is tested on video sequences that have different types of shakiness and diverse video contents. The obtained results are then compared to current state-of-the-art stabilization algorithms including Youtube stabilization and it is found that the proposed method significantly outperforms other algorithms in terms of visual quality while performing in real time.

I. Introduction

The small sized and low priced modern digital video cameras helped increase their deployment in mobile handheld devices and unmanned aerial vehicles (UAVs) to capture videos for civilian and military applications. However, since these cameras are mounted on moving platforms, recorded videos are often very shaky due to the influence of optical turbulence. Video stabilization aims to minimize the amount of shakiness in the video to enhance visual quality and remove undesirable motion. Video stabilization is usually done using one or more of the following three techniques: 1) Mechanical stabilization [19], which is usually achieved through sensors like Gyros, to compensate for the undesired motion, 2) Optical stabilization [19], which is another hardware-based solution that adjusts the camera lens based on the path of the light travelling through the lens system and 3) Digital stabilization [19], which is a post-processing software-based solution that is applied on the digital images after being produced. This paper focuses on the third technique (digital video stabilization) because software-based solutions are normally cheaper than the hardware-based ones and are more attractive in many applications.

Digital video stabilization approaches can be further subdivided into two categories based on the motion estimation model: 1) 2D Stabilization[15], [7], and 2) 3D Stabilization[14], [11]. Digital video stabilization usually undergo three main stages: 1) Estimating the camera motion, 2) Generating a smooth camera motion and 3) Refining the stabilized video using the estimated smoothed camera path. This paper falls in the 2D stabilization category aiming to provide robust and high quality stable sequences using 2D linear transformations.

In this paper, we present StableFlow, a novel real-time method inspired by the mass spring damper model. The method eliminates the undesired motion in the video based on the response from the mass spring model. The video frame is modelled as a mass suspended in each direction by a critically dampened spring and damper. Springs allow the mass (i.e., the frame) to move while controlling its range of movement in case of shakiness. On the other hand, the dampers help keep the system stable and prevent undesired oscillations. The system is parametric and can be fine-tuned to adapt with various motion types consisting of rotation, translation or combination of both. The remainder of this paper is organized as follows: related work is presented in section 2; a detailed description of the proposed method is presented in Section 3, followed by the evaluation criteria and results in Section 4. A discussion of the results is then presented in section 5. Finally, Section 6 concludes the paper and directs future work.

II. Literature Review

Despite the fact that digital video stabilization has been studied for many years, it still remains an active research field. The 2D digital video stabilization methods estimates linear transformations (affine or homography) between successive frames and smooth the trajectory over time to produce a stabilized video. Wang et al. [21] assumed the motion model can fit a polynomial curve, e.g. a cubic curve to smooth the parameters. Irani et al. [9] dealt with complex video sequences and attempted to estimate a homography transformation which stabilizes a dominant planar region in the video. Matsushita et al. [15] extended the stabilized frames to become full frames and applied low pass filter to smooth the parameters over time. Grundmann et al. [7] proposed the application of L1-norm optimization model to smooth the camera path and follow cinematography rules. This technique is considered to be the current state-of-the-art and is currently integrated into Googles YouTube.
On the other hand, 3D methods require the recovery of the structure from the video sequence including 3D camera poses and depth structures. These structures can be computed using structure from motion (SFM) techniques [2], [5], [10], [23]. Buehler et al. [3] computed SFM in a general un-calibrated camera setting using the bundle adjustment method [20]. Liu et al. [11] proposed a full 3D stabilization method by introducing content-preserving warps for the novel view synthesis. Liu et al. [14] used a depth camera to recover depth information and perform 3D digital video stabilization. To overcome the challenges associated with 3D reconstruction of a full video, Goldstein and Fattal [6] used the concepts of epipolar geometry to avoid 3D reconstruction. Wang et al.[22] proposed a new representation of each feature trajectory as a Bezier curve and then smoothed over time. Liu et al. [12] choosed to smooth the basis trajectories of the subspace [8] which are extracted from long feature tracks of 50 frames or more. This method achieved high quality stabilization that is similar to the full 3D methods, while avoiding the need of a full 3D reconstruction. This technique has been integrated in Adobe After Effects as a digital video stabilization function named Warp Stabilizer. Recently, Liu et al. [13] proposed an extension of the method in [12] to cope with stereoscopic videos.

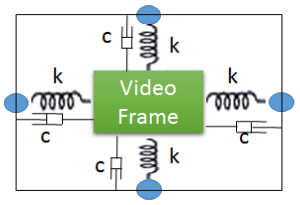

Fig. 1. Mass spring model for digital video stabilization.

However, the need of long feature trajectories is difficult to achieve especially in videos with quickly changing scenes or frequent occlusions.

The work in this paper was motivated by the hardware-based solution in [18] [1] where they built a system to adjust the LCD screen in vehicles or smart phones based on input from hardware sensors. In their work, vertical translation motion was only addressed and hardware accelerometers were used to obtain the acceleration and motion information.

To the best of the authors knowledge, the proposed method in this paper is the first digital video stabilization method to employ a physics inspired software model based on mass spring damper model to smooth the camera path and generate a video with a better visual quality without any a priori knowledge of the camera motion while achieving real time performance. The results outperforms current state-of-the-art methods.

III. Proposed Method

A. Physical Model

The proposed method can be described using the physics model shown in Figure 1. The video frame is modelled as a mass suspended with a spring and damper in each direction. In this model, there is no gravity as it is irrelevant in the context of digital video stabilization and hence is omitted. In the proposed model, the springs suppress the undesired motion but still allow the frame to move freely. On the other hand, the dampers prevent the frame from undesired counter productive oscillations. The dampers can also be fine-tuned to allow the system to converge to the steady state as early as possible.

The analysis of the system in 1-Dimension is shown in Figure 2. The first force acting on the frame is F_{spring} which is generated from stretching the spring

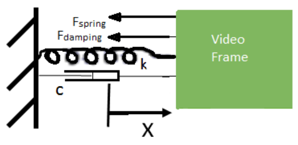

Fig. 2. Model in 1-Dimension.

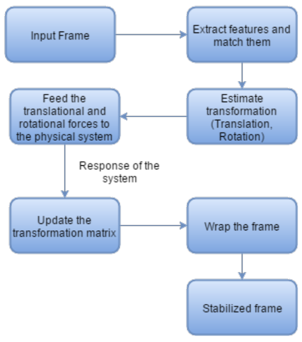
Input Frame
Extract features and match them
Estimate transformation (Translation, Rotation)
Feed the translational and rotational forces to the physical system
Response of the system
Update the transformation matrix
Wrap the frame
Stabilized frame

Fig. 3. Flowchart of the proposed algorithm

and acts in the opposite direction of the stretch, F_{spring} is defined by:

$$F_{spring} = -kx \tag{1}$$

where k is the spring constant, and x is the stretch length of the spring
In addition, there is a damping (friction) force $F_{damping}$ that resists the motion and is proportional to the velocity $\dot{x}$, $F_{damping}$ is defined by:

$$F_{damping} = -c\dot{x} \tag{2}$$

where c is the damping coefficient Therefore, the total force acting on the frame will be:

$$F = F_{spring} + F_{damping} = -kx - c\dot{x} \tag{3}$$

From Newton's law of motion we have:

$$\sum f = m\ddot{x} \tag{4}$$

where $\ddot{x}$ is the acceleration

$$m\ddot{x} + c\dot{x} + kx = 0 \tag{5}$$

Moreover, the motion of this frame also depends on what is called the damping ratio ζ which is given by:

$$\zeta = \frac{c}{2\sqrt{km}} \tag{6}$$

$$\zeta \begin{cases} < 1, & \text{Under-damped system and will keep oscillating.} \\ > 1, & \text{Over-damped, no compensation for shakiness.} \\ = 1, & \text{Optimally damped.} \end{cases} \tag{7}$$

To make the formulation of the model simpler, we assume the mass of the frame to be a unit mass $m = 1$ because setting the mass to a certain value will only lead to a scale in the response of the physical system. Also, as illustrated above in Equation 7 to obtain optimal damping $\zeta = 1$, hence equation 7 becomes:

$$c = 2\sqrt{k} \tag{8}$$

Substituting in Equation 5, we get:

$$\ddot{x} + 2\sqrt{k}\dot{x} + kx = 0 \tag{9}$$

It is worth mentioning that the analysis for rotational forces is similar to the mentioned analysis but with a rotational spring.

B. Numerical Solution

To solve Equation 9 numerically (i.e., using computer program), we use the Runge-Kutta method [4] which requires the conversion of the second order differential equation in Equation 9 into a set of first order differential equations. Since the acceleration can be written as the first derivative of velocity: $\ddot{x} = \dot{v}$, equation 9 can be expressed as a system of two first order differential equations:

$$\dot{x} = v \tag{10}$$

$$\dot{v} = -kx - 2\sqrt{k}\dot{x} \tag{11}$$

Equations 10 and 11 represent the form needed in order to use the Runge-Kutta method to numerically solve the differential equation in Equation 9. To solve using Runge-Kutta, four variables k_1, k_2, k_3 and k_4 has to be evaluated as follows:

$$k_1 = f(t, y) \tag{12}$$

where t is time and y is the function to be approximated.

$$k_2 = f(t + \Delta T/2, y + \Delta T/2k_1) \tag{13}$$

$$k_3 = f(t + \Delta T/2, y + \Delta T/2k_2) \tag{14}$$

$$k_4 = f(t + \Delta T, y + \Delta T k_3) \tag{15}$$

where ΔT is the time interval between two successive frames. After that the integration will be used to update the new value of the function being approximated:

$$y_{n+1} = y_n + \Delta T/6(k_1 + 2k_2 + 2k_3 + k_4) \tag{16}$$

C. Path Smoothing

let P_t denotes the original shaky frame path, which is formed by the concatenation of the frame affine transformations over time, as defined by:

$$P_t = \prod_{i=0}^{i} H_i. \tag{17}$$

The stabilized camera path $\acute{P_t}$ can be easily computed through the convolution of the path with the response of the physical model as shown in Equation 18:

$$\acute{P_t} = P_t * G(t) \tag{18}$$

where $G(t)$ is the physical response of the system. Once $\acute{P_t}$ is calculated, the final transformation parameters for the video frame can be easily extracted which are then used to warp the video frame and produce a stabilized frame. It is worth mentioning that the algorithm is highly parallelizable as the calculation for each node in the physical model can run in parallel.

D. Implementation of the Proposed Method

The implementation of the proposed method is described in details in Algorithm 2. The motion between frames is considered as affine transformation consisting of translation and rotation. The affine transformation is calculated based on feature matching between successive frames. The proposed method then converts the calculated affine transformation into forces and inputs it to the physical model. So, the input to the physical model is the jittery estimated transformation parameters i.e. rotation θ and translation in the vertical and horizontal direction T_y, T_x. After that, the Runge-kutta method is used to solve the model and estimate the position and velocity of the frame. Finally, the final stabilized frame is computed through warping the input frame with the estimated trajectory from the physical model.

Algorithm 1 RK4 Method

```
1: Input: $\Delta T$.
2: for each mass do
3:     Compute $k_1$ using equation 12 .
4:     Compute $k_2$ using equation 13 .
5:     Compute $k_3$ using equation 14 .
6:     Compute $k_4$ using equation 15 .
7:     Compute velocity and position using equation 16.
8: end for
9: Output: New position of the system particles.
```

Algorithm 2 The Proposed Method

```
1: Input: Input Frame.
2: Extract good features to track.
3: Track the features into previous frame.
4: Evaluate good matches.
5: Estimate affine transformation.
6: Convert transformation to forces and feed them to the mass spring model
7: Calculate the position and velocity using Runge-Kutta as shown in Algorithm 1
8: Calculate a smoothed trajectory and calculate its transformation.
9: Generate the final stabilized frame by warping the input frame with the transformation estimated from the smoothed trajectory.
10: Output: Stabilized Frame.
```

TABLE I. Comparison between the fidelity of the original sequence (left) ,using our method (middle) and Stabilized Fidelity from Google's Youtube(right).

Sequence	Original Fidelity	Our Stabilized Fidelity	Google's Youtube
Seq. 1	13.089 dB	14.3 dB	13.6 dB
Seq. 2	13.9 dB	14.735 dB	14.5 dB
Seq. 3	12.38 dB	13.725 dB	13.75 dB
Seq. 4	9.4 dB	11.45 dB	10.2 dB

IV. Results

The proposed algorithm has been tested on the datasets provided by [17] [7] and [11]. The evaluation criteria are based on: 1) the mean of the sequence to assess the visual quality improvement, 2) the average fidelity [16] (using Equation 19), and 3) the camera trajectory in x, y, and θ.

$$PSNR_{dB}(I_1, I_0) = 10 \log \frac{(255)^2}{MSE(I_1, I_0)} \tag{19}$$

where MSE is the mean square error measuring the error per pixel from the optimal stabilized result, and the 255 represents the maximum intensity a pixel may have.

Figure 4 shows a comparison of the mean for the stabilized sequences using Google's YouTube stabilizer [7] and the proposed method.Table I compares the fidelity values from the original video, the proposed method and Google's Youtube method. Figures 5 - 7, shows the trajectories comparison in vertical, horizontal and rotational angle respectively for the test sequence number 4.

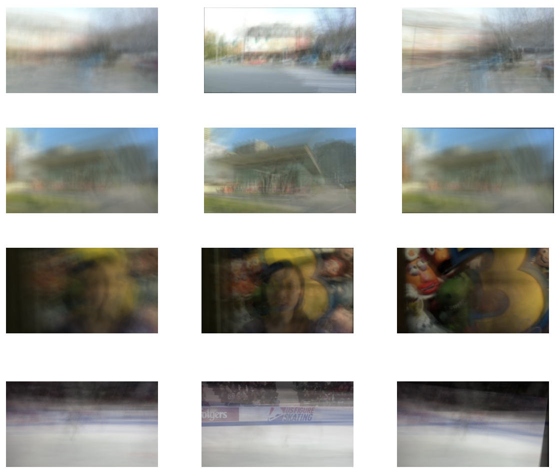
(a) Original Mean of Sequence (b) Our Stabilized Sequence Mean (c) Google's Youtube Stabilized Sequence Mean

Fig. 4. Sequence Mean Comparison

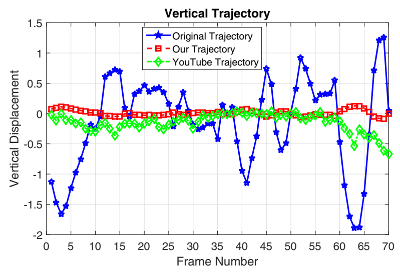

Fig. 5. Vertical Trajectory

V. Discussion and Performance Analysis

A. Discussion

The proposed method outperformed the current state-of-the-art method used on Google's Youtube in the video sequences being tested. However, as shown in Table I in Sequence number 3, Youtube performed slightly better than the proposed method and that is mainly because youtube method operates on the whole video in advance while the proposed method performs in real-time with no information about future frames in the video. Hence, the proposed method will not optimally perform in instances of extreme transit motion (i.e, extreme motion in less than one second) after which the motion path returns to normal shaky pattern as before. In general, the proposed method produces videos with high visual quality and takes benefit of the robustness and simplicity of 2D methods and takes advantage of the physical system properties to generate a stable camera path.

As shown in section III, the system depends on the spring constant k. Selecting a very small value for k will soften the resistance of the spring to motion and hence the system will not be able to suppress much of the jitter motion which eventually will lead to inefficient stabilization. On the other hand, selecting a very high value for k will cause the spring force F_{spring} to be very high which in turns makes it harder for the system to converge to a steady state. Moreover, this very high spring force F_{spring} may, in some cases, lead to more jitter in the produced video because it will enlarge the response of the system in the opposite direction of the motion. In the proposed method, the spring constant must be carefully chosen for each test sequence to produce the best possible stabilization result. In this paper, the authors manually selected this constant empirically through experiments with values that range from 0.5 to 1.25. However, in the future work, the system will adjust the constant k to adapt with the movement on a per frames-batch basis. This can be achieved by allowing the system to automatically select the proper constant for the spring using an iterative method.

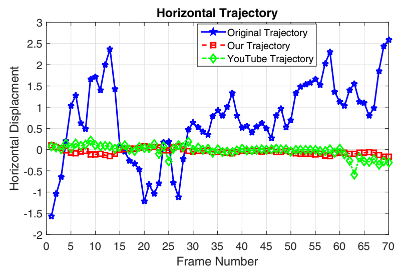

Fig. 6. Horizontal Trajectory

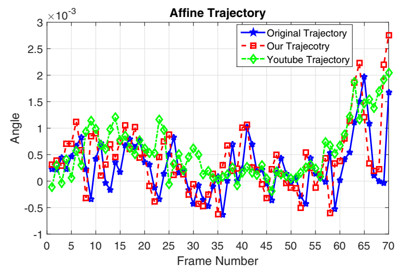

Fig. 7. Rotational Trajectory

TABLE II. Running time in Frames per second for each of the tested sequences

Sequence	Running Time (fps)
Seq. 1	20.7
Seq. 2	18.7
Seq. 3	20.27
Seq. 4	19.2

B. Runtime Analysis

The proposed algorithm has been tested on the following configurations: Intel Quad Core processor @2.20 GHz. The average running time for the whole algorithm is around 20 fps, which meets real-time applications constraints. The total running time of the proposed method on each of the test sequence is given in Table II. On the other hand, Youtube running time did not exceed 10 fps for the tested video sequences.

VI. Conclusion and Future Work

This paper presented a novel stabilization method based on the mass spring damper model with 2D linear transformations that meets real time requirement. The proposed method outperforms current state-of-the-art methods, such as Youtube stabilization, while achieving real time performance. Future work include modifying the method to automatically select the spring constant to be able to adapt to extreme shakiness. Future work also include performing more tests on various datasets and comparing them to other current state of the art methods.

References

[1] B. Abali, H. Franke, and M. E. Giampapa. Method and apparatus for image stabilization in display device, Nov. 13 2001. US Patent 6,317,114.

[2] S. Agarwal, Y. Furukawa, N. Snavely, I. Simon, B. Curless, S. M. Seitz, and R. Szeliski. Building rome in a day. *Communications of the ACM*, 54(10):105–112, 2011.

[3] C. Buehler, M. Bosse, and L. McMillan. Non-metric image-based rendering for video stabilization. In *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, volume 2, pages II–609. IEEE, 2001.

[4] E. Fehlberg. Low-order classical runge-kutta formulas with stepsize control and their application to some heat transfer problems. 1969.

[5] Y. Furukawa, B. Curless, S. M. Seitz, and R. Szeliski. Towards internet-scale multi-view stereo. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 1434–1441. IEEE, 2010.

[6] A. Goldstein and R. Fattal. Video stabilization using epipolar geometry. *ACM Transactions on Graphics (TOG)*, 31(5):126, 2012.

[7] M. Grundmann, V. Kwatra, and I. Essa. Auto-directed video stabilization with robust l1 optimal camera paths. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 225–232. IEEE, 2011.

[8] M. Irani. Multi-frame correspondence estimation using subspace constraints. *International Journal of Computer Vision*, 48(3):173–194, 2002.

[9] M. Irani, B. Rousso, and S. Peleg. Recovery of ego-motion using image stabilization. In *Computer Vision and Pattern Recognition, 1994. Proceedings CVPR'94., 1994 IEEE Computer Society Conference on*, pages 454–460. IEEE, 1994.

[10] N. Jiang, P. Tan, and L.-F. Cheong. Seeing double without confusion: Structure-from-motion in highly ambiguous scenes. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 1458–1465. IEEE, 2012.

[11] F. Liu, M. Gleicher, H. Jin, and A. Agarwala. Content-preserving warps for 3d video stabilization. In *ACM Transactions on Graphics (TOG)*, volume 28, page 44. ACM, 2009.

[12] F. Liu, M. Gleicher, J. Wang, H. Jin, and A. Agarwala. Subspace video stabilization. *ACM Transactions on Graphics (TOG)*, 30(1):4, 2011.

[13] F. Liu, Y. Niu, and H. Jin. Joint subspace stabilization for stereoscopic video. In *Computer Vision (ICCV), 2013 IEEE International Conference on*, pages 73–80. IEEE, 2013.

[14] S. Liu, Y. Wang, L. Yuan, J. Bu, P. Tan, and J. Sun. Video stabilization with a depth camera. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 89–95. IEEE, 2012.

[15] Y. Matsushita, E. Ofek, W. Ge, X. Tang, and H.-Y. Shum. Full-frame video stabilization with motion inpainting. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 28(7):1150–1163, 2006.

[16] C. Morimoto and R. Chellappa. Evaluation of image stabilization algorithms. In *Acoustics, Speech and Signal Processing, 1998. Proceedings of the 1998 IEEE International Conference on*, volume 5, pages 2789–2792. IEEE, 1998.

[17] U. of Central Florida. Ucf aerial action data set. http://crcv.ucf.edu/data/UCF_Aerial_Action.php.

[18] A. Rahmati, C. Shepard, and L. Zhong. Noshake: Content stabilization for shaking screens of mobile devices. In *Pervasive Computing and Communications, 2009. PerCom 2009. IEEE International Conference on*, pages 1–6, March 2009.

[19] P. Rawat and J. Singhai. Review of motion estimation and video stabilization techniques for hand held mobile video. *Signal & Image Processing: An International Journal (SIPIJ) Vol*, 2, 2011.

[20] B. Triggs, P. F. McLauchlan, R. I. Hartley, and A. W. Fitzgibbon. Bundle adjustmenta modern synthesis. In *Vision algorithms: theory and practice*, pages 298–372. Springer, 2000.

[21] Y. Wang, Z. Hou, K. Leman, and R. Chang. Real-time video stabilization for unmanned aerial vehicles. In *MVA*, pages 336–339, 2011.

[22] Y.-S. Wang, F. Liu, P.-S. Hsu, and T.-Y. Lee. Spatially and temporally optimized video stabilization. *Visualization and Computer Graphics, IEEE Transactions on*, 19(8):1354–1361, 2013.

[23] C. Wu. Towards linear-time incremental structure from motion. In *3D Vision-3DV 2013, 2013 International Conference on*, pages 127–134. IEEE, 2013.