StableFlow: A Physics Inspired Digital Video Stabilization

by

©Abdelrahman Ahmed

A dissertation submitted to the School of Graduate Studies
in partial fulfillment of the requirements for the degree of

Master of Electrical and Computer Engineering

Faculty of Engineering and Applied Science
Memorial University of Newfoundland

October 2016

St. John's, Newfoundland

Abstract

This thesis addresses the problem of digital video stabilization. With the widespread use of handheld devices and unmanned aerial vehicles (UAVs) that has the ability to record videos, digital video stabilization becomes more important as the videos are often shaky undermining the visual quality of the video. Digital video stabilization 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 thesis, we firstly introduce a novel study of the effect of image denoising on feature-based digital video stabilization. Then, we introduce SteadyFlow, a novel technique for real-time stabilization inspired by the mass spring damper model. 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 technique is tested on video sequences that have different types of shakiness and diverse video contents. The obtained results significantly outperforms state-of-the art stabilization techniques in terms of visual quality while performing in real time.

To the soul of my father ...

To my mom and my awesome fiancee.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction and Overview

1.1 Introduction

The widespread of hand-held devices (cell phones, portable camcorders) made recording videos easier. However, these amateur videos are often so shaky with a lot of unwanted motion jitter [5, 19]. On the other hand, a professional stable video requires professional, sophisticated and expensive devices (track , special body mount) as seen in Figure 1.1. Video stabilization aims to improve the quality of the recorded video from ordinary cameras to be similar to those recorded with specialized equipments.Video stabilization can be achieved using one of the following methods:

1. Optical Image Stabilization (OIS)
2. Mechanical Image Stabilization (MIS)
3. Digital Video Stabilization (DIS)

Optical Image Stabilization (OIS)

Optical image stabilization systems , deal with the video during the acquisition phase

Fig. 1.1: Body mount for camera stabilization. [1]

before the image is transformed into the digital representation. These systems can be further categorized into

- Lens- based often abbreviated IS or OS
- Sensor-based abbreviated IBIS

In general, Sensors are used to detect vibrations in pitch and yaw to calculate both the angle and the speed of the movement. Then in IS, by the help of an actuator the lens group can move in both directions horizontally and vertically to counteract the vibration and maintain a stable image as seen in Figure 1.2, as distinct in IBIS it moves the image sensor itself as the final element in the optical path.

Mechanical Image Stabilization (MIS)

Mechanical image stabilization is another hardware-based system for image stabilization. This system uses gyroscopes and shock absorbers to compensate for vibrations in any direction. Steadicam [1] shown in Figure 1.3 shows an example for such systems.

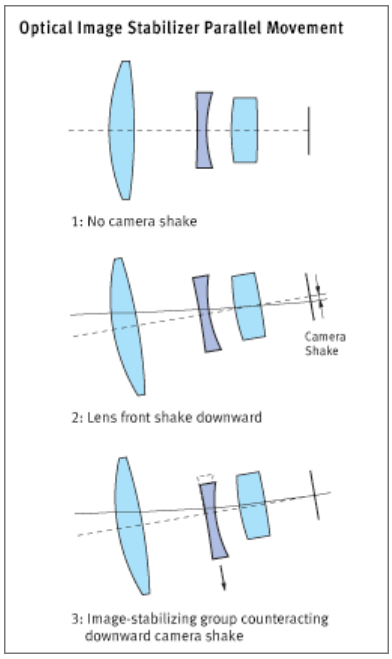


Fig. 1.2: Optical image stabilization system. [2]

Fig. 1.3: Steadicam equipment [1]

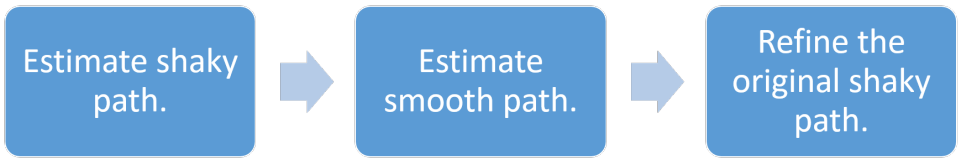


Fig. 1.4: Digital Image Stabilization work flow.

Digital Image Stabilization (DIS)

Digital image stabilization is normally applied after images have been converted into digital images as a post-processing technique. DIS aims to minimize the amount of unwanted jitter in the movement by shifting the image to compensate for the jitter. Some systems use sensors to help correcting the image, however, that can not be done for all videos, urging the need for an automated computer vision approach. A typical DIS system consists of three main steps as shown in Figure 1.4

1. Estimate original camera path (Shaky path).
2. Estimate new smooth camera path.
3. Correct the original path based on the smoothed one.

However, DIS systems face a lot of challenges:

1. **Noise :** Image noise present a huge challenge for DIS as they can severely damage the image quality and make the process of estimating the camera motion (path) a very challenging task. The types of noise vary from additive, impulsive and frequency noise. This topic will be further discussed in detail in Chapter 3.

2. **Quick camera motion :** Quick rotations and zooming represents a challenge for digital video stabilization, since most methods rely on long feature trajectories. In this case, it becomes nearly impossible to maintain long feature trajectory which in turn leads to severely damaging the performance of the digital video stabilization algorithms.

3. **Real-time Performance Vs Video Quality :** Most video stabilization algorithms suffer from being either computationally expensive to produce high quality videos or trying to achieve real-time performance sacrificing video quality. The main challenges therefore is to achieve real-time performance while maintaining high video quality.

Efficient video stabilization system try to accommodate and overcome most of the aforementioned challenges. The types of jitter in the video can be categorized into either:

- Low frequency jitter: originating from the movement of the platform holding the camera.

- High frequency jitter : originating from sudden camera movements.

To obtain a high quality video, both forms of jitter have to be suppressed without the introduction of any kind of artifacts (e.g. wobble, excessive cropping) to the video. In summary, the objectives of the video stabilization are :

1. Good stability.

2. No geometrical distortion, or artifacts.

3. Reasonable cropping size.

1.2 Thesis Contributions

This section provides a brief description of the contributions in this thesis:

1. **Denoising effect on Stabilization** Since noise is considered a challenge to most of the stabilization algorithms, adding a denoising step as a pre-processing stage for stabilization is very common in surveillance systems. However, no study has been done before to evaluate the impact of the denoising stage on the quality of the final stabilized video. Also, the known measures of image processing and video stabilization in the literature is not sufficient to provide this quantitative analysis, so we propose in the study a new quantitative measure to assess the impact of the denoising on the final video. This work is to be submitted to Elseiver Pattern Recognition Journal.

2. **StableFlow** which is a novel method based on mass-spring-damper 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 Adobe After Effects and it is found that the proposed method significantly outperforms other algorithms in terms of visual quality while performing in real time. This work presents a novel method and has been accepted for publication in the International Conference of Pattern Recognition (ICPR 2016). Also an extended version is to be submitted to the IEEE Transactions on Image Processing Journal.

1.2.1 Publications

- Abdelrahman Ahmed and Moahmed S Shehata, "Towards High-quality parallel stabilization" in VISAPP 2016. International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications
- Abdelrahman Ahmed and Moahmed S Shehata, "StableFlow: A Novel Real-Time Physics Inspired Method for Digital Video Stabilization" in ICPR 2016. International Conference of Pattern Recognition (ICPR 2016).
- Abdelrahman Ahmed and Moahmed S Shehata, "Effect of Denoising on Video stabilization" submitted to IET Image Processing.
- Abdelrahman Ahmed and Moahmed S Shehata, "Video stabilization with Mass-spring model" is to be submitted to the IEEE Transactions on Image Processing Journal.

1.3 Thesis Organization

The remainder of this thesis is organized as follows. Chapter 2 provides the literature review on video stabilization. Chapter 3 presents the work of studying the effect of denoising on stabilization. Chapter 4 discusses a novel proposed algorithm for video stabilization. Chapter 5 concludes this thesis with a discussion of future research directions.

Chapter 2

Literature Review

2.1 Introduction

Digital video stabilization can be categorized into three main categories based on the motion estimation model: 1) 2D Methods, 2) 3D Methods, and 3) 2.5D Methods.

2D digital video stabilization methods estimates linear transformations (affine or homography) between successive video frames, then, by concatenating these transformations, a camera path in 2D space can be obtained. Let the video be a sequence of images $I_1, I_2, ...I_n$, where each frame pair (I_{t-1}, I_t) is associated with a linear motion model F_t. The camera path C_t is defined as:

$$C_t = C_{t-1}F_t \tag{2.1}$$

$$C_{t-1} = \prod_{i=1}^{i=t-1} F_i. \tag{2.2}$$

The 2D camera path is then smoothed over time to produce a stabilized video.In the literature most researches tries to enhance both the motion estimation [5] and path planning of the camera [6]. The advantage of 2D methods is its robustness as it only requires feature

correspondences between neighbouring frames. The 2D model fitting is much more robust compared to 3D methods. However, When a scene contains large depth variations, the 2D model is not suitable and artifacts, such as content distortions, would be introduced in the stabilized results. Our work "Towards High-quality parallel stabilization" [7]and "StableFlow: A Novel Physics inspired real time stabiliztion method" belong to this category. Our proposed work can produce results with the same quality as 3D methods while having the robustness of 2D methods.

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 [8–11]. Buehler et al. [12] computed SFM in a general un-calibrated camera setting using the bundle adjustment method [13]. Liu et al. [14] proposed a full 3D stabilization method by introducing content-preserving warps for the novel view synthesis. Liu et al. [15] used a depth camera to recover depth information and perform 3D digital video stabilization.

2.5D Methods relax the requirement of full 3D reconstruction to some partial 3D information such as epipolar geometry [16] . The 2.5D methods argue that the 3D reconstruction is an overshoot for video stabilization purposes. The 2.5D methods can produce comparable results to full 3D methods while reducing the computational complexity. However, the requirement of long feature tracks, for 30 frames or more, is still a bottleneck for the robustness in these algorithms.

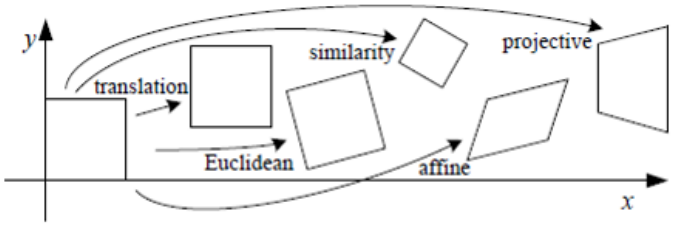


Fig. 2.1: Graphical representation of basic set of 2D Transformations [3]

2.2 2D Video Stabilization

2D stabilization methods use 2D transformations (affine, homography, similarity, .. etc), as seen in Fig. 2.1, to represent the camera motion then smooth the parameters of these matrices to stabilize the video. In the next section, these transformation matrices are discussed.

2.2.1 Transformation Matrices

Transformation matrices form the mathematical parameters that map pixels from one image to another. A number of motion models are possible such as :

- Dolly : Forward and backward motion
- Boom: Up and down motion
- Pan: Y-axis rotation.
- Tilt: X-axis rotation.
- Roll: View-axis rotation.

Table 2.1 summarizes various transformation matrices sorted by the number of degrees of freedom and showing the different parameters for each matrix.

Transform	Degrees Of Freedom	Matrix
Translation	2	$M = \begin{Bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \end{Bmatrix}$
Similarity	4	$M = \begin{Bmatrix} 1+a & -b & t_x \\ b & 1+a & t_y \end{Bmatrix}$
Affine	6	$M = \begin{Bmatrix} 1+a_{00} & a_{01} & t_x \\ a_{10} & 1+a_{11} & t_y \end{Bmatrix}$
Homography	8	$M = \begin{Bmatrix} 1+h_{00} & h_{01} & h_{02} \\ h_{10} & 1+h_{11} & h_{12} \\ h_{20} & h_{21} & 1 \end{Bmatrix}$

Table 2.1: Parametric Transformation Matrices

2.2.2 2D Video Stabilization Methods

Video stabilization algorithms development can be traced back to the development in the field of motion estimation. Different algorithms have been proposed to reduce the computational complexity and to improve the accuracy of the motion estimation. Hence,tThe efficiency of video stabilization depends on the accuracy of the motion estimation and optical flow methods. For example, Horn and Schunck [17] is a widely used optical flow method, however, the method solely computes the slow motion and provides the motion vectors in one direction only. Global motion estimation can either be calculated by feature-based methods [18–20] or pixel-based methods [5, 21, 22].

Early methods of video stabilization [23, 24] proposed the estimation of a series of affine or homography transformation between successive video frames over time and the application of a Gaussian low pass filter to smooth the transformation parameters and reduce the high frequency jitter in the video. Hansen et al. [25] proposed a video stabilization system based on a mosaic-based registration technique. Rong Hu et al. [19] proposed an algorithm to estimate the global camera motion using Scale-Invariant Feature Transfrom (SIFT) features. Derek Pang et al. [20] proposed using Dual-Tree complex wavelet transform (DT-CWT). This method uses the relationship between the phase changes of

DT-CWT and the shift invariant feature displacement in spatial domain to perform the motion estimation, then, an optimal Gaussian kernel filtering is used to suppress the motion jitters. Wang et al. [26] assumed the motion model can fit a polynomial curve, e.g. a cubic curve to smooth the parameters. Litvin et al. [27] proposed the usage of Kalman filtering to estimate the parameters of a the affine transformation. Irani et al. [28] attempted to estimate a homography transformation which stabilizes a dominant planar region in the video. Matsushita et al. [5] extended the stabilized frames to become full frames and applied low pass filter to smooth the parameters over time. Recently, Liu et al. [29] proposed modeling the camera path into multiple camera path. The model is based on mesh-based, spatially-variant motion representation and an adaptive, space-time path optimization. Liu et. al [30] proposed smoothing the pixel trajectories, which are motion vectors collected at the same pixel location over time. Grundmann et al. [31] proposed the automatic application of constrainable, L1-optimal camera paths to generate stabilized videos and compute camera paths that are composed of constant, linear and parabolic segments mimicking the camera motions employed by professional cinematographers. This technique is considered to be the current state-of-the-art and is currently integrated into Google's YouTube. This method will be discussed in more detail in the upcoming subsection.

2.2.3 L1 Optimal Camera Path

From a cinematographic perspective, the most pleasant steady viewing experience is conveyed by the use of static cameras, panning cameras mounted on tripods and cameras placed onto a dolly and avoiding any sudden change in acceleration. Therefore, the computed camera path needs to adhere to these characteristics as much as possible. To mimic professional footage, the optimized camera path should be composed by the following path

segments:

- A constant path, representing a static camera $|D(P)|_t$.
- A path of constant velocity, representing a panning or a dolly shot $|D^2(P)|_t$.
- A path of constant acceleration, representing the ease-in and out transition between static and panning cameras $|D^3(P)|_t$.

Grundmann et al. [31] formulated the smoothed path estimation based on L1-norm optimization as follows; Given the original path C_t ,the desired smooth path will be:

$$P_t = C_t B_t \tag{2.3}$$

where $B_t = C_t^{-1}$,P_t is the update transform which brings the original frame to its stabilized position. The objective function will be formulated as:

$$O(P) = w_1|D(P)|_1 + w_2|D^2(P)|_1 + w_3|D^3(P)|_1 \tag{2.4}$$

where w_1,w_2 and w_3 are different weights that can be fine-tuned to improve the accuracy of the video stabilization. The objective function can be minimized by linear programming using forward differencing method for the three terms $|D(P)|_1$, $|D^2(P)|_1$ and $|D^3(P)|_1$.

2.3 3D Video Stabilization

3D methods estimate the 3D camera motion to perform stabilization. Beuhler et al. [12] proposed a 3D video stabilization method based on a projective reconstruction of the scene with an uncalibrated camera, when euclidean reconstruction is feasible. Liu et al. [14] proposed the content-preserving warp for the novel view synthesis which is inspired by as-rigid-as-possible shape manipulation [32]. The proposed method consists of three stages:

- Recovery of 3D camera motion and a sparse set of 3D static scene points.
- Desired camera path is selected interactively by the user
- Least square optimization computing a spatially- varying warp from the input video frame into an output frame. The wrap is computed to follow the displacements suggested by the recovered 3D structure and to minimize the distortion of local shape and content in the video frames.

However, 3D reconstruction is still a limitation for all these methods despite the significant progress in 3D reconstruction [8–11].

2.4 2.5D Video Stabilization Methods

2.5D Methods try to overcome the challenges associated with 3D reconstruction of a full video, Goldstein and Fattal [16] used the concepts of epipolar geometry to avoid 3D reconstruction. The method consists of tracking feature points in the scene and using them to compute fundamental matrices that model the stabilized camera motion. Then, the tracked points are projected onto the novel stabilized frames using epipolar point transfer and the new frames are synthesized using image-based frame warping. Wang et al. [33] proposed a new representation of each feature trajectory as a Bezier curve

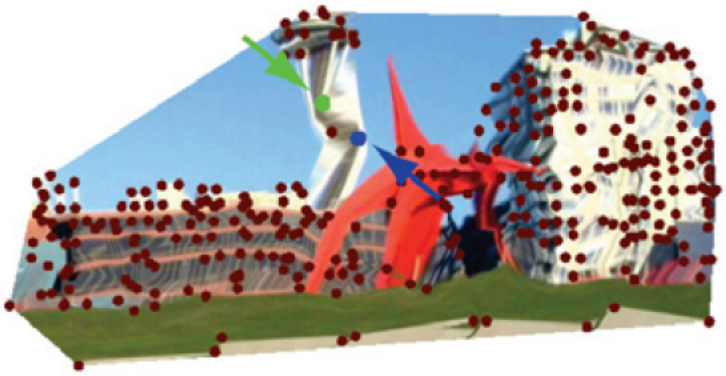


Fig. 2.2: Low-pass filter for each trajectory independently [34].

and then smoothed over time. Liu et al. [34] proposed the method of subspace video stabilization. The method is based on the choice to smooth the basis trajectories of the subspace [35] which are extracted from long feature tracks of 30 frames or more. The subspace method also achieved high quality stabilization that is similar to the full 3D methods, while avoiding the need of a full 3D reconstruction. Recently, Liu et al. [36] proposed an extension of the method in [34] to adapt with stereoscopic videos. However, the need of long feature trajectories is difficult to achieve especially in videos with quickly changing scenes or frequent occlusions. In the following, we briefly discuss the method of Liu et al. [34] subspace video stabilization. This technique has been integrated in Adobe After Effects as a digital video stabilization function named "Warp Stabilizer".

2.4.1 Subspace Video Stabilization

Given a set of 2D point trajectories, their concatenation into a trajectory matrix will be:

$$M = \begin{Bmatrix} x_1^1 & x_2^1 & \dots & x_F^1 \\ y_1^1 & y_2^1 & \dots & y_F^1 \\ \vdots & & & \\ x_1^N & x_2^N & \dots & x_F^N \\ y_1^N & y_2^N & \dots & y_F^N \end{Bmatrix}$$

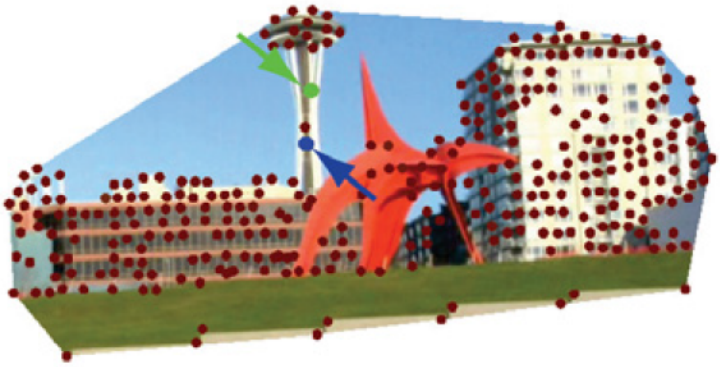


Fig. 2.3: Eigen-trajectory smoothing [34].

where N are the features per frame and F is the number of frame. According to Liu [34] attempting to apply low pass-filter directly on this trajectory matrix will lead to distortions in the image as the independent smoothing of the feature trajectory will not respect the relationship between the points as shown in Figure 2.2.

The subspace approach to video stabilization consists of four steps:

- 2D point tracking and assembly of the 2D trajectory sparse matrix.
- Moving factorization representing the trajectories as the product of basis vectors called "eigen-trajectories".
- Smoothing the eigen-trajectories.
- Rendering the final stable frame.

Figure 2.3, shows the output of the smoothing of the eigen-trajectories.

Chapter 3

Denoising Effect on Digital Video Stabilization

3.1 Introduction

Noise can degrade the image during the acquistion or transmission of the image and their removal algorithms depend on the type of noise present in the image. Image denoising usually refers to the process of applying specific filters according to the type of noise (e.g, additive noise, impulsive noise, etc...) to produce higher visual quality images with higher signal to noise ratio (SNR) [37].

Noise can be categorized into three main categories:

- Additive noise
- Impulsive noise
- Frequency noise

Additive Noise is modelled as the sum of the input signal and a random variable, for example, if the random variable is Gaussian, then this noise is called Additive White

Fig. 3.1: Image corrupted with AWGN. [4]

Fig. 3.2: Image corrupted with salt and pepper. [4]

Gaussian Noise (AWGN) as shown in Fig. 3.1

Impulsive Noise does not follow a specific probabilistic distribution. Impulsive noise is modelled as random occurrences of spikes in the input signal with random amplitudes. A special case of impulsive noise is Salt-and- pepper noise in which the energy spikes alter some of the input image pixels into either white (salt) or black (pepper) as shown in Fig. 3.2.

Frequency Noise is commonly modelled as a multiplication of the input signal and a random variable. Motion blur is considered as an example of frequency noise which is common in videos captured using moving camera platforms as shown in Fig. 3.3.

Fig. 3.3: Image corrupted with blurring noise. [4]

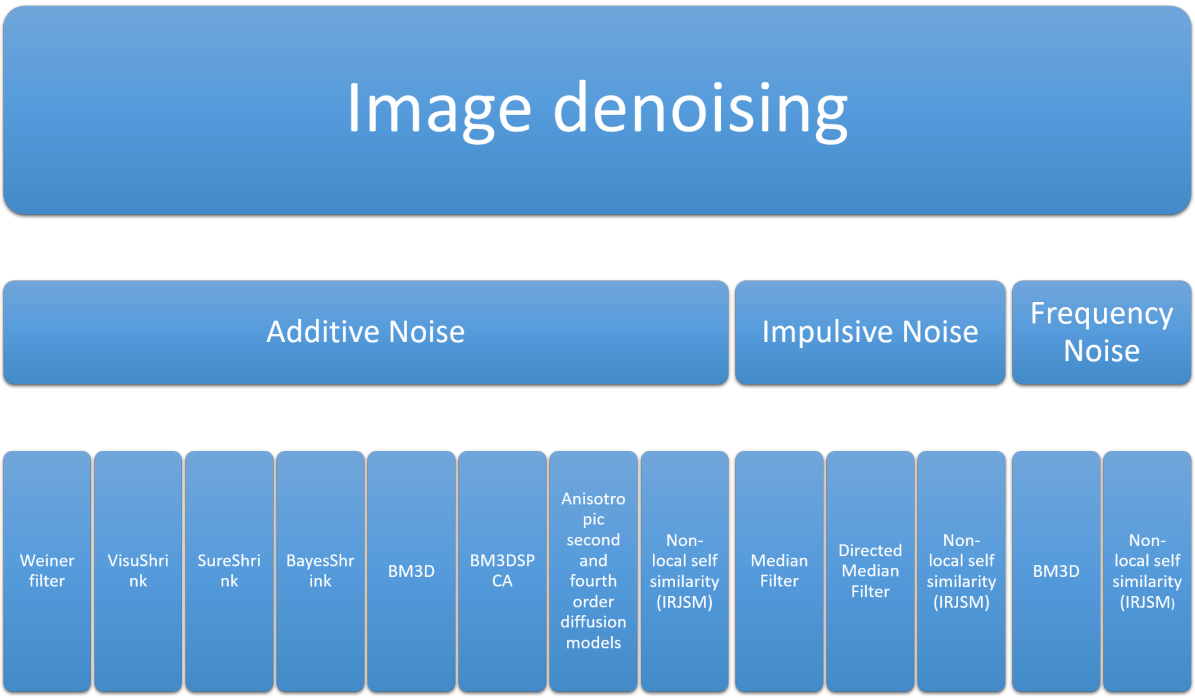


Fig. 3.4: Different Denoising Algorithms

3.2 Image Denoising

Different techniques have been proposed in the literature for each of the three denoising categories. In the case of the AWGN, Weiner filter represents one of the earliest denoising techniques for this type of noise [38] along with its variations [39,40]. Other popular techniques are based on wavelets were introduced such as VisuShrink [41], BayesShrink [42] and SureShrink [43]. The VisuShrink threshold is a function of noise variance and the number of samples. The SureShrink threshold is considered the optimal in terms of Stein's Unbiased Risk Estimator (SURE). Modern algorithms such as BM3D [44] and its variations BM3DSPCA [45] have been used as a benchmark for most state-of-the-art techniques. BM3D utilizes the block-matching concept and filtering in 3D transform domain with a sliding window. Wang et. al [46] introduced a novel technique using second order (CONVEFF Second) and fourth order (CONVEFF fourth) diffusion model for image denoising. Zhang et. al. [47] proposed a novel technique (IRJSM) for denoising based on

both local smoothness and non-local self-similarity of natural images in a unified statistical manner.

For denoising of the salt and pepper noise, median filter [4] is one of the earliest techniques used with low salt and pepper noise densities. However, modern state-of-the-art techniques addressed that drawback; for example, Zhang et al. [48] proposed a switching median filter with a pre-defined threshold to detect noisy pixels. Dong et al. [49] proposed a technique based on directional weighted median filter (DWM), then Ching-Ta et al. [50] further extended this filter and proposed the modified directional weighted median filter (MDWM) which can produce high quality results with images corrupted up to 70% noise density. Zhang et al. [47] non-local self-similarity technique can also be used with images corrupted by salt and pepper.

In case of blurring noise, the deblurring techniques can be categorized into two main categories based on the information needed for the estimation of the blur kernel. These categories are: 1) Single image, and 2) Multi-image. To be consistent with other types of noise categories presented in this paper (e.g. Additive noise and impulse noise), only the single image techniques are evaluated. Image blurring is commonly modelled as a convolution of the image and a blur kernel as follows:

$$y = k \otimes x \tag{3.1}$$

where y represents the blurred image, k is the blur kernel, x is the original image and $\otimes$ represents the convolution operation. Fergus et al. [51] introduced a Bayesian-based approach to estimate the kernel. Also, BM3D [44], BM3DSPCA [45] and IRJSM [47] can be used for deblurring images.

Table 3.1: Experiment Summary

Noise type	Algorithms considered
AWGN	BM3D, IRJSM, Conveff Second
Salt & Pepper	MDWM, IRJSM
Blur	BM3D, IRJSM

3.3 Study Desgin

In this study, current state-of-the-art denoising techniques will be examined to evaluate their effect on feature-based stabilization of a shaky video from different datasets. In the following subsections, the details of each component of this study is provided. It is worth mentioning that due to the high processing demand for some of the techniques used in the study, some of the experiments have been run on the ACENet cluster [52] on both parallel and sequential jobs.

3.3.1 Datasets

The study was conducted on three different datasets provided by [53] , [31] and [14]. The datasets were chosen carefully to evaluate the denoising techniques effect on feature-based stabilization under different and diverse image contents, such as: 1)Large flat texture regions, 2) Repeated patterns, and 3) Large number of edges and corners. The frames of each video sequence being tested are extracted and the noise is then synthesized into the frames, then, the frames undergo denoising followed by stabilization.

3.4 Evaluation Criteria

The evaluation of the effect of denoising techniques on featre-based stabilization is based on [23] using the following measures: 1) Mean of the stabilized sequence, 2) Fidelity, and 3) Proposed background detail preserving measure.

3.4.1 Mean of Stabilized Sequence

The mean of the stabilized video sequence is the average of all the frames as follows:

$$Mean(I_1, I_2, ..., I_n) = \frac{1}{n}\sum_{i=1}^{n} I_i. \tag{3.2}$$

In this paper, we have chosen to use the mean of the stabilized sequence because it shows the visual quality enhancement achieved by the stabilization after the denoising technique under evaluation is applied.

3.4.2 Fidelity

In [23], the fidelity has been introduced as a measure to evaluate any stabilization technique based on peak signal-to-noise ratio (PSNR) which measures how much the stabilized frames have departed from the optimal case. As defined in [23], the PSNR between two frames I_1 and I_0 is defined as:

$$PSNR_{dB}(I_1, I_0) = 10 \log \frac{(255)^2}{MSE(I_1, I_0)} \tag{3.3}$$

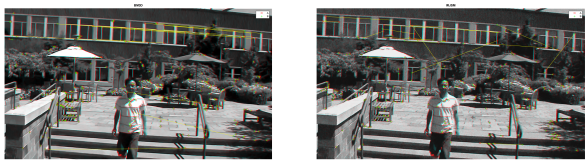
where MSE is the mean squared error measuring the error per pixel from the optimal stabilized result, and the 255 represents the maximum intensity a pixel can have. In this paper, we have chosen fidelity as a measure to evaluate the stabilized sequence after denoising because the direct relation in which a high fidelity means better stabilized sequence.

3.4.3 Background Detail Preserving (DP)

This is a novel measure that been proposed by the author to quantitively assess the effect of denoising algorithms on feature-based digital video stabilization. SNR has commonly been

(a) BM3D (b) IRJSM

Fig. 3.5: Sequence mean of AWGN with 25% noise



(a) BM3D SNR=47.8dB, DP=58.34% (b) IRJSM SNR=42.1dB, DP=66.3%

Fig. 3.6: SNR VS Proposed measure Comparison of AWGN with 25% noise

used to indicate the quality of the denoised images using a specific denoising technique [37]. In Figure 3.6, the image in (a) represents the AWGN denoising result using $BM3D$ and it has a SNR of 47.8 dB while the image in (b) represents the AWGN denoising result using $IRJSM$ and it has a SNR of 42.1 dB. Based on this information only, one can expect that applying a stabilization technique on the sequence containing image (a) will result in better stabilization than the sequence containing image (b) because of the higher SNR in the former one and hence using $BM3D$ is better than $IRJSM$. However, after applying the stabilization technique, the fidelity of sequence that contains image (a) is 56.3 dB and its mean is shown in Figure 3.5 a; while the fidelity sequence that contains image(b) is 60.7 dB and its mean is shown in Figure3.5 b. This shows that the stabilization of sequence (b) is better than the stabilization of sequence (a) and hence $IRJSM$ is better than $BM3D$, in contradict to SNR-based the initial assumption. Therefore, we propose a new measure to evaluate the impact of the denoising algorithm on stabilization using background features that can be recovered after denoising. Hence, by comparing the average number of features which belong to the background extracted and matched

between frames before adding any type of noise and the average number of matches that has been recovered after denoising, we can have a percentage defining the impact of a specific desnoising algorithm on the stabilization. A higher recovery percentage means the motion of the video is better preserved leading to better estimation for transformation matrices and hence better stabilization. Using this proposed measure (abbreviated DP) on the images in Figure 3.6, we find that indeed image b has high DP than image a which is in-line with the other two measures (e.g. fidelity and mean). Although the fidelity measure can give a indication on the effect of the denoising technique, however, using it alone is insufficient to have an accurate quantitative assessment for the effect of the denoising technique. Fidelity measures how a stabilized video frame deviated from the optimal alignment case after warping, hence the measure will be affected by any rounding errors in the motion transformation parameters estimation process, unlike the proposed DP measure. More importantly, the proposed DP measure can be used as an early indication as it does not require the warping to be calculated.

3.5 Results from the study

3.5.1 Additive White Gaussian Noise

Traditional Algorithms

Gaussian white noise with different variances has been added to each frame of the test sequence with variances ranging from 0.01% to 0.1% with 0.01% step. Figures 3.7 - 3.12 shows the results of the effect of the Wiener filters, VISUSHRINK, and SURESHRINK denoising techniques on stabilization as applied to three different dataset contents, namely: 1) dataset with large flat areas, 2) dataset with rich features, and 3) dataset with repeated patterns.

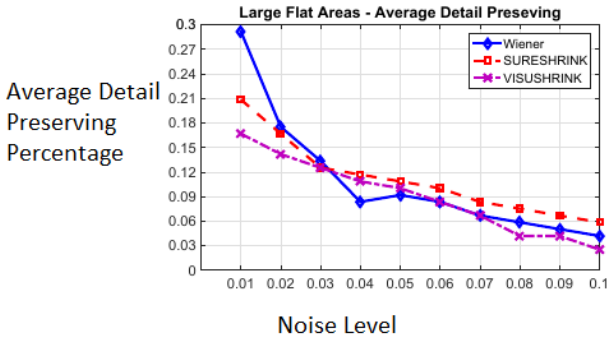


Fig. 3.7: AWGN - Percentage of matches recovered after using each Traditional denoising algorithm in datasets with large flat areas.

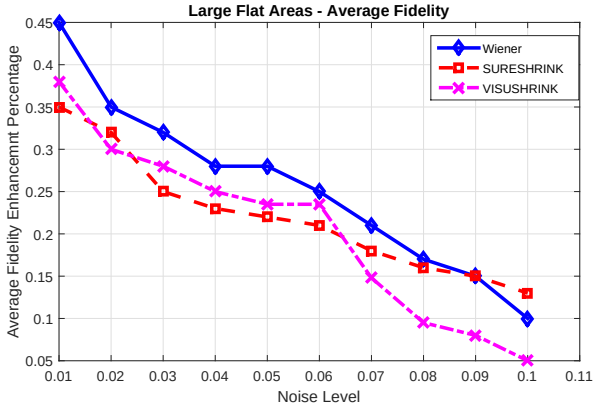


Fig. 3.8: AWGN - Fidelity of the resulting stabilized sequence after Traditional denoising in datasets with large flat areas.

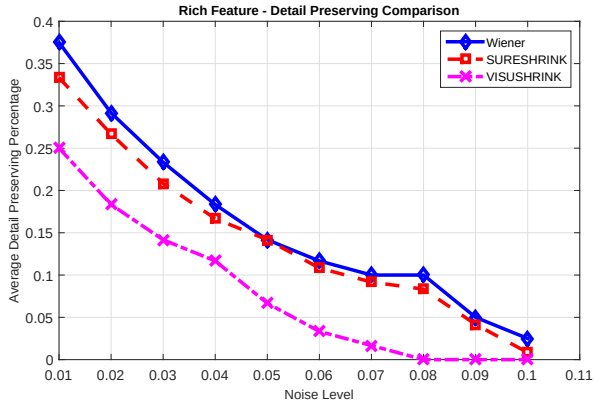


Fig. 3.9: AWGN - Percentage of matches recovered after using each Traditional denoising algorithm in datasets with rich features.

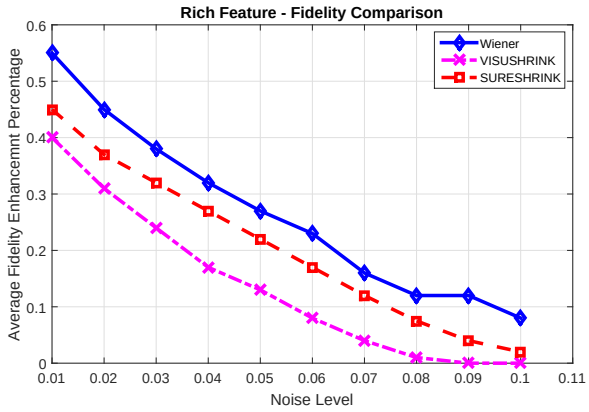


Fig. 3.10: AWGN - Fidelity of the resulting stabilized sequence after Traditional denoising in datasets with rich features.

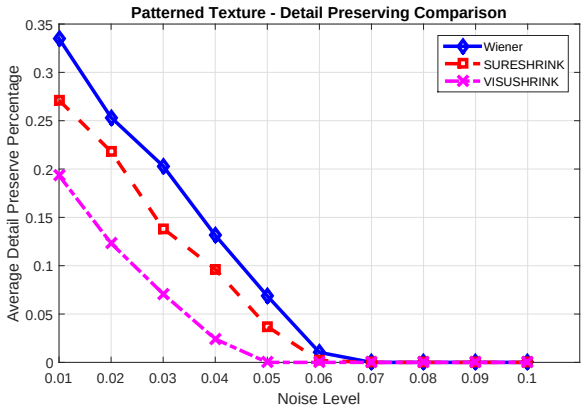


Fig. 3.11: AWGN - Percentage of matches recovered after using each Traditional denoising algorithm in datasets with Patterned texture.

Discussion

The adaptive wiener filter in general performs better than the two wavelet filters (SureShrink,VisuShrink). However, the processing time for the wiener filter was found to be greater than the other two wavelet filters. The wiener filter had an overhead processing time of 28% compared to the VisuShrink filter and 23.6% compared to the SureShrink wavelet. Both wavelet filters did not preserve the details of the image resulting in more loss of features unlike the wiener filter which preserved the edges and other high frequency parts of the image. VisuShrink performed poorly on all the experiments generating images

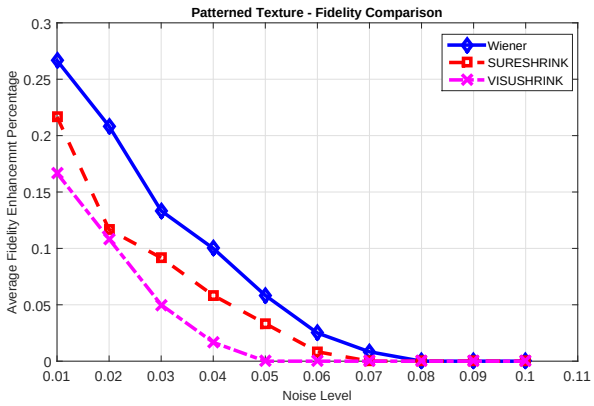


Fig. 3.12: AWGN - Fidelity of the resulting stabilized sequence after Traditional denoising in datasets with Patterned texture.

that are overly smoothed. This is because VisuShrink has the limitation that it employs a global threshold that is applied to all wavelet coefficients unlike, SureShrink that employs an adaptive threshold to each resolution level generating images with less smoothing.

State-of-the-art

Gaussian white noise with different variances has been added to each frame of the test sequences. The variances of the noise added range from 25% to 70% with 5% step. Figures 3.13 - 3.19 shows the results of the effect of the BM3D, IRJSM, and Conveff Second denoising techniques on stabilization as applied to three different dataset contents, namely: 1) dataset with large flat areas, 2) dataset with rich features, and 3) dataset with repeated patterns.

Discussion

As summarized in Tables 3.2 - 3.4, BM3D in general performs better with significantly lower running time compared to the two other algorithms, Conveff Second and IRJSM. Although BM3D had generated smooth images, yet the edges of the image were well preserved even at high noise levels. BM3D scores the highest percentage of recovered

Table 3.2: BM3D Performance , DP denotes Detail Preserving Percentage and Fid denotes Fidelity Enhancement Percentage

Dataset Type	Noise Percentage									
	25%		35%		45%		55%		65%	
	DP	Fid.	DP	Fid.	DP	Fid.	DP	Fid.	DP	Fid.
Feature Rich	63.7%	54.8%	50.8%	47.67%	41.8%	40.41%	30.15%	35.65%	20.7%	28.42%
Large Flat Areas	50.32%	37.7%	35.98%	28.3%	20.47%	16.82%	12.0%	10.95%	6.8%	2.3%
Repeated Patterns	50.1%	41.78%	39.12%	32.0%	19.2%	19.01%	8.99%	5.61%	5.23%	0.34%

matches even when increasing the noise level up to 70% over different datasets. It also scores the highest SNR and Fidelity and generated the most stable sequence as seen in Figure 3.13.

IRJSM performed slightly worse than BM3D in sequences with large flat areas and in sequences with high levels of noise. In other sequences which had repeated textures, IRJSM outperformed BM3D. Unlike BM3D, IRJSM did not over smooth the image allowing it to preserve the features even when producing lower SNR images as seen in Figure 3.6 and in Tables 3.2 - 3.4 over different datasets. However, the main drawback in IRJSM is that it is very computationally expensive taking around 10 minutes per frame, as its denoising technique is based on an iterative technique that takes time to converge.

Conveff Second performed poorly at high noise levels which can be seen in the terms of fidelity or the percentage of matches that can be recovered.

(a) Noisy Mean (b) BM3D (c) Conveff Second (d) IRJSM

Fig. 3.13: AWGN - Mean of the sequence at 25% noise level

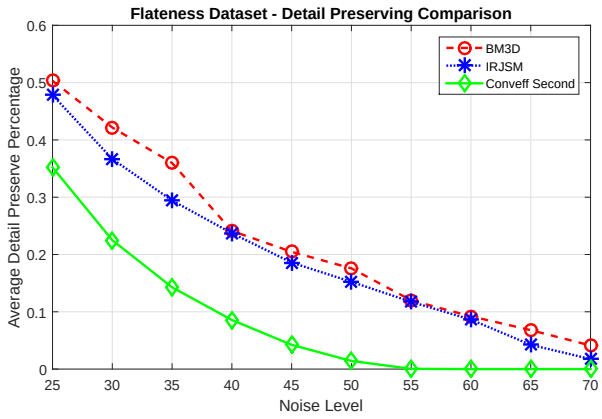


Fig. 3.14: AWGN - Percentage of matches recovered after using each denoising algorithm in datasets with large flat areas.

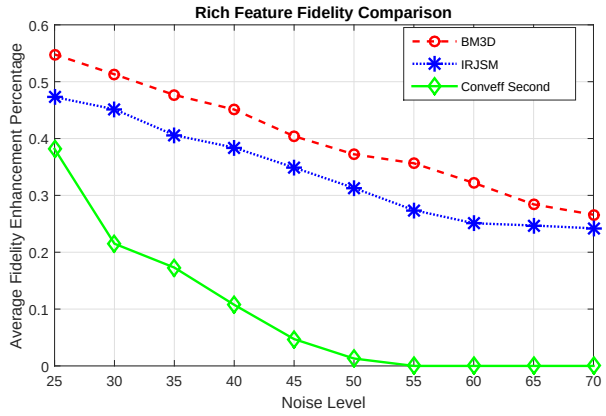


Fig. 3.15: AWGN - Fidelity of the resulting stabilized sequence after denoising in datasets with large flat areas.

Table 3.3: Conveff Second Performance , DP denotes Detail Preserving Percentage and Fid denotes Fidelity Enhancement Percentage

Dataset Type	Noise Percentage									
	25%		35%		45%		55%		65%	
	DP	Fid.	DP	Fid.	DP	Fid.	DP	Fid.	DP	Fid.
Feature Rich	42.15%	38.17%	23.7%	17.3%	11.3%	4.72%	5.15%	0%	1.5%	0%
Large Flat Areas	35.21%	25.41%	14.28%	10.20%	4.23%	1.07%	0.04%	0%	0%	0%
Repeated Patterns	18.5%	13.72%	4.74%	1.65%	0%	0%	0%	0%	0%	0%

3.5.2 Salt-and-Pepper Noise

In this study, Salt-and-pepper noise with different densities has been added to each frame of the test sequence, then the frames undergo denoising followed by stabilization. The

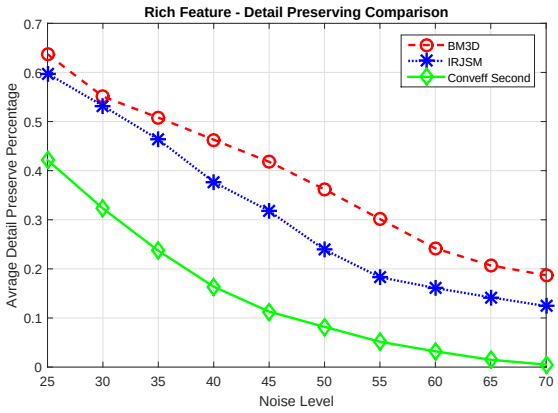


Fig. 3.16: AWGN - Percentage of matches recovered after using each denoising algorithm in features rich datasets

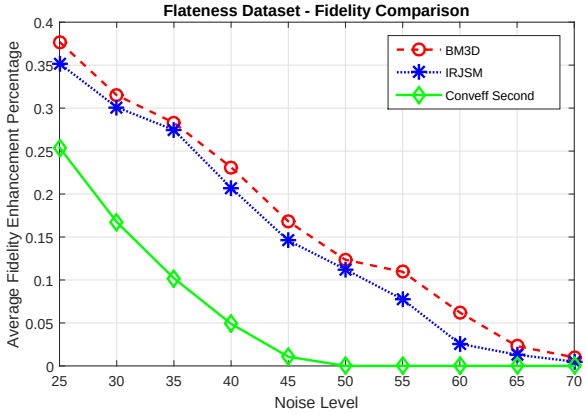


Fig. 3.17: AWGN - Fidelity of the resulting stabilized sequence after denoising in features rich dataset

Table 3.4: IRJSM Performance , DP denotes Detail Preserving Percentage and Fid denotes Fidelity Enhancement Percentage

Dataset Type	Noise Percentage									
	25%		35%		45%		55%		65%	
	DP	Fid.	DP	Fid.	DP	Fid.	DP	Fid.	DP	Fid.
Feature Rich	59.75%	47.3%	46.35%	40.6%	31.8%	34.89%	18.3%	27.4%	14.2%	24.7%
Large Flat Areas	47.82%	35.2%	29.42%	27.45%	18.59%	14.59%	11.74%	7.74%	4.23%	1.3%
Repeated Patterns	51.68%	43.21%	38.74%	31.44%	18.3%	17.78%	9.89%	3.89%	4.2%	0.2%

variances of the noise added range from 25% to 70% with 5% step. Figures 3.21 - 3.26 shows the results of the experiment on the diverse used datasets. Figure 3.20 shows the mean of the sequence (after adding salt and pepper noise) and the resultant mean after

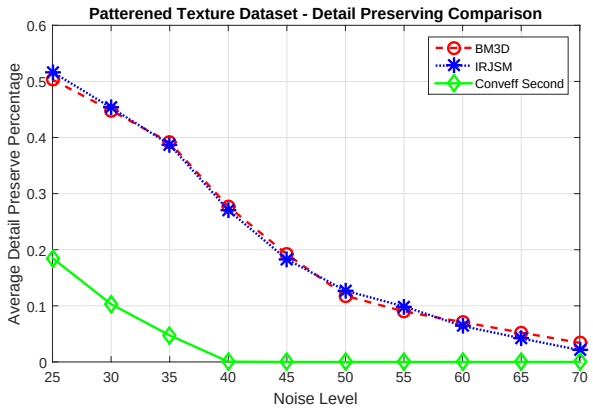


Fig. 3.18: AWGN - Percentage of matches recovered after using each denoising algorithm in patterned texture datasets

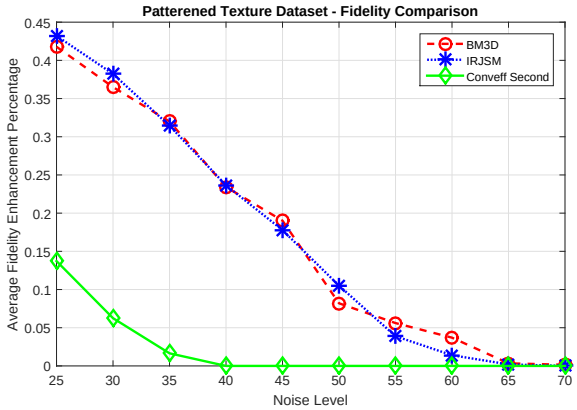
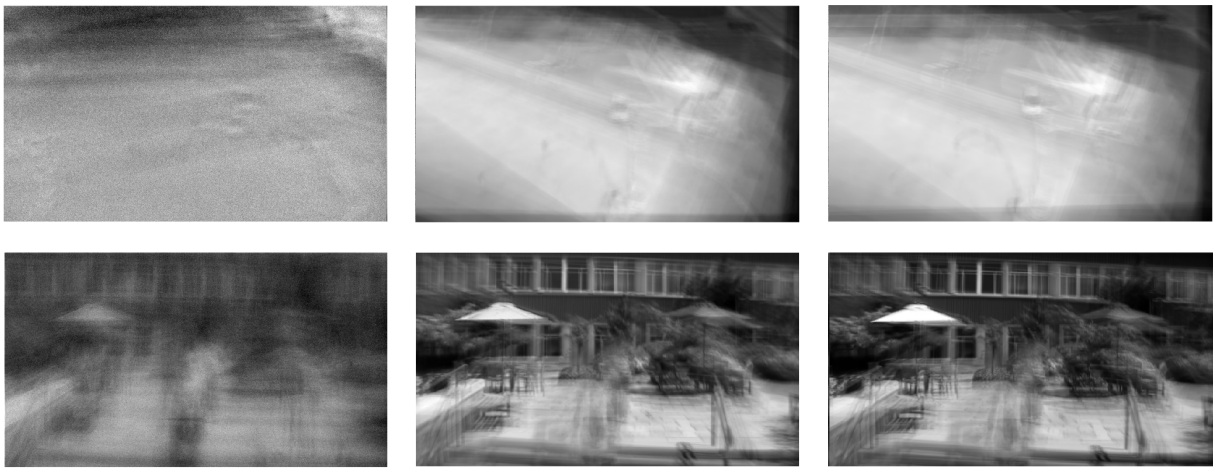


Fig. 3.19: AWGN - Fidelity of the resulting stabilized sequence after denoising algorithm in patterned texture datasets.

denoising using the two different techniques at 25% noise level.

Discussion

As summarized in Tables 3.5 - 3.6 , IRJSM outperformed the MDWM over all the datasets, preserving most of the image features and generating better stabilized sequences. However in cases where the amount of noise is not high, generally lower than 30%.



(a) Noisy Mean (b) MDWM (c) IRJSM

Fig. 3.20: Salt & Pepper - Mean of the sequence.

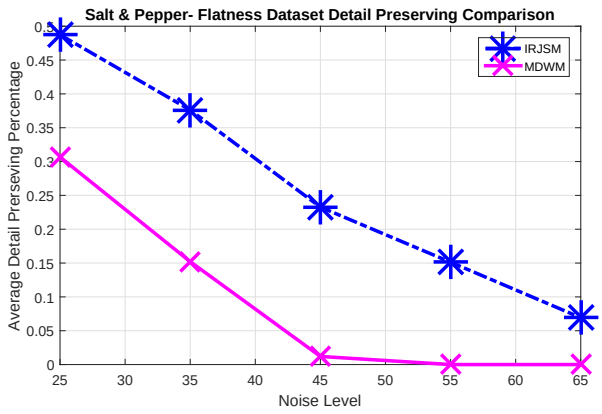


Fig. 3.21: Salt & Pepper - Percentage of matches recovered after using each denoising algorithm in datasets with large flat areas.

Table 3.5: Salt & Pepper ,MDWM Performance , DP denotes Detail Preserving Percentage and Fid denotes Fidelity Enhancement Percentage

Dataset Type	Noise Percentage									
	25%		35%		45%		55%		65%	
	DP	Fid.	DP	Fid.	DP	Fid.	DP	Fid.	DP	Fid.
Feature Rich	40.81%	45.3%	25.31%	19.52%	8.13%	3.65%	0.15%	0%	0%	0%
Large Flat Areas	30.67%	24.47%	15.21%	12.14%	1.2%	0.5%	0%	0%	0%	0%
Repeated Patterns	25.24%	19.68%	10.57%	6.89%	2.11%	0.35%	0%	0%	0%	0%

3.5.3 Blurring

In this study, Gaussian Blur kernel with different variances has been multiplied by each frame of the test sequence, then the frames undergo deblurring followed by stabilization.

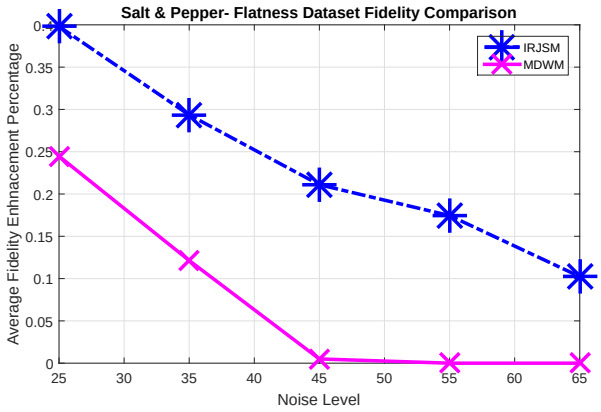


Fig. 3.22: Salt & Pepper - Fidelity of the resulting stabilized sequence after denoising in datasets with large flat areas.

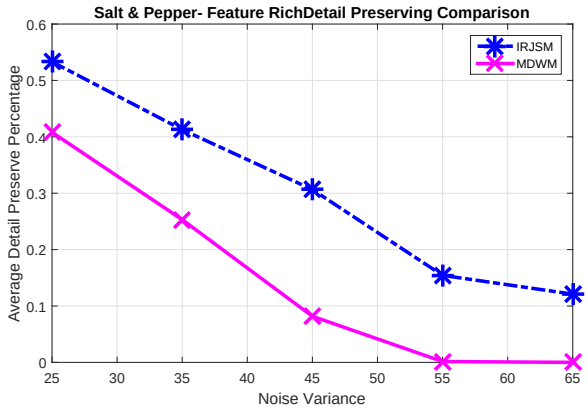


Fig. 3.23: Salt & Pepper - Percentage of matches recovered after using each denoising algorithm in features rich datasets

Table 3.6: Salt & Pepper, IRJSM Performance , DP denotes Detail Preserving Percentage and Fid denotes Fidelity Enhancement Percentage

Dataset Type	Noise Percentage									
	25%		35%		45%		55%		65%	
	DP	Fid.	DP	Fid.	DP	Fid.	DP	Fid.	DP	Fid.
Feature Rich	53.3%	65.7%	41.27%	50.5%	30.65%	32.12%	15.43%	29.8%	12.1%	22.42%
Large Flat Areas	48.74%	39.81%	37.62%	29.35%	23.27%	21.11%	15.13%	17.45%	6.93%	10.24%
Repeated Patterns	50.81%	48.38%	39.15%	33.74%	27.23%	25.4%	13.16%	15.21%	5.5%	7.1%

The variances of the blur kernel used range from 1% to 5% with 1% step.

Figures 3.27 - 3.32 shows the results of the experiment on the used datasets.

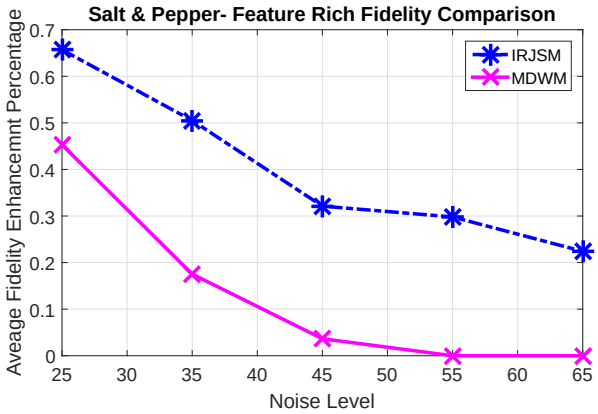


Fig. 3.24: Salt & Pepper - Fidelity of the resulting stabilized sequence after denoising in features rich dataset

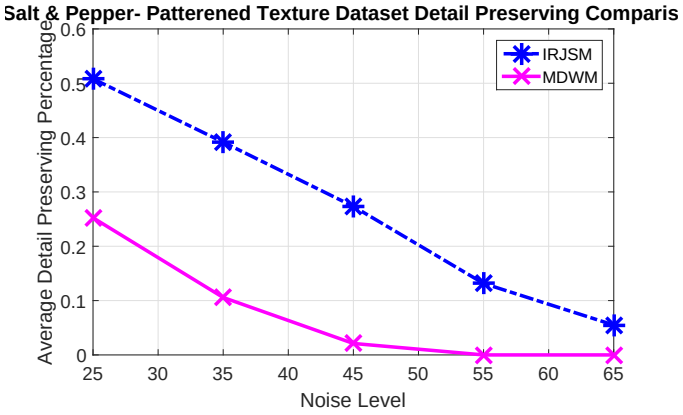


Fig. 3.25: Salt & Pepper - Percentage of matches recovered after using each denoising algorithm in patterned texture datasets

Discussion

As shown in Tables 3.7 - 3.8, the added blur was challenging for both techniques in the different datasets. On low noise levels both techniques produce comparable results. However, BM3D outperforms IRJSM on higher noise levels. Also, the computational complexity of BM3D is significantly lower than IRJSM.

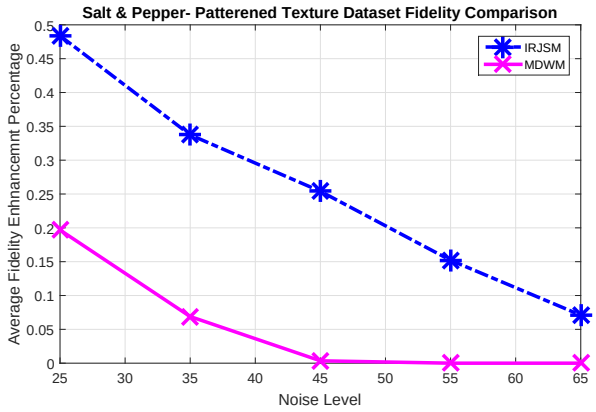


Fig. 3.26: Salt & Pepper - Fidelity of the resulting stabilized sequence after denoising algorithm in patterned texture datasets.

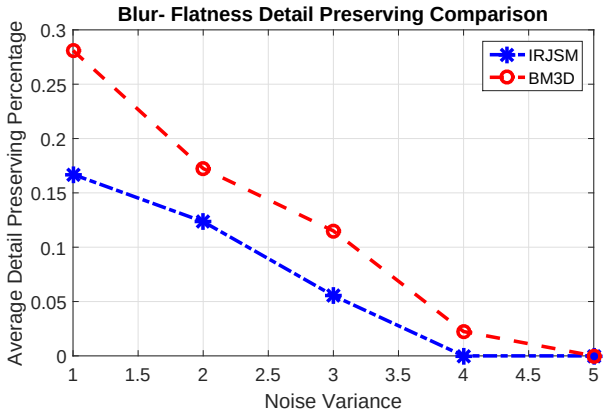


Fig. 3.27: Percentage of matches recovered after using each denoising algorithm in datasets with large flat areas.

Table 3.7: Blurring ,BM3D Performance , DP denotes Detail Preserving Percentage and Fid denotes Fidelity Enhancement Percentage

Dataset Type	Noise Percentage									
	1%		2%		3%		4%		5%	
	DP	Fid.	DP	Fid.	DP	Fid.	DP	Fid.	DP	Fid.
Feature Rich	36.71%	40.3%	23.25%	20.17%	14.03%	9.85%	6.91%	2.13%	1.05%	0%
Large Flat Areas	28.1%	32.25%	17.24%	20.12%	11.5%	6.3%	2.24%	0.15%	0%	0%
Repeated Patterns	18.6%	20.43%	9.27%	6.1%	2.57%	1.01%	0%	0%	0%	0%

3.6 Summary Of Findings

The most important findings from the study can be summarized in:

- Using the recent algorithm does not always guarantee the best stabilization results

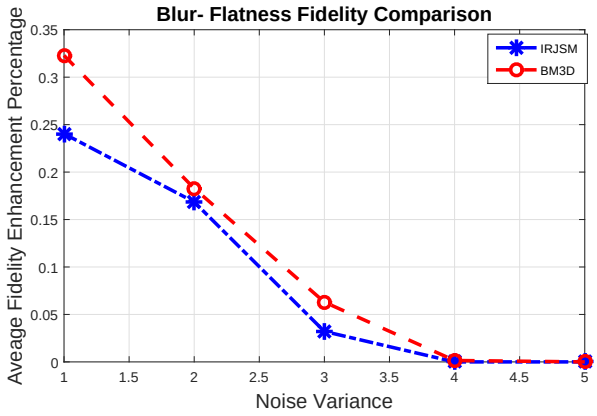


Fig. 3.28: Blur - Fidelity of the resulting stabilized sequence after denoising in datasets with large flat areas.

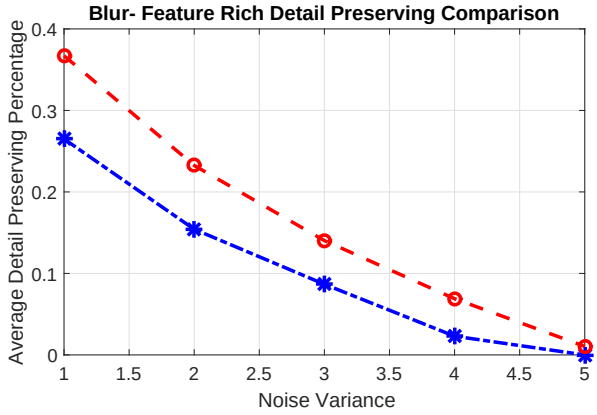


Fig. 3.29: Blur - Percentage of matches recovered after using each denoising algorithm in features rich datasets

Table 3.8: Blur, IRJSM Performance , DP denotes Detail Preserving Percentage and Fid denotes Fidelity Enhancement Percentage

Dataset Type	Noise Percentage									
	1%		2%		3%		4%		5%	
	DP	Fid.	DP	Fid.	DP	Fid.	DP	Fid.	DP	Fid.
Feature Rich	26.5%	17.38%	15.42%	6.3%	8.01%	1.1%	2.3%	0.05%	0%	0%
Large Flat Areas	16.7%	24.04%	12.35%	16.85%	5.5%	3.21%	0%	0%	0%	0%
Repeated Patterns	13.21%	15.73%	5.8%	3.21%	1.23%	0.5%	0%	0%	0%	0%

as seen in Figure 3.6

- The content of the dataset can also affect the choice of the denoising algorithm. For example in case of videos with patterned texture with AWGN, IRJSM generated

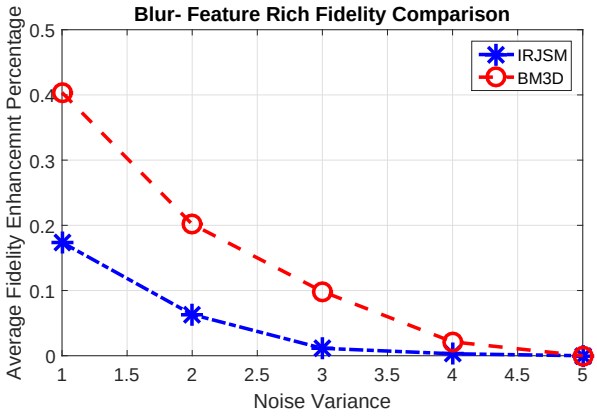


Fig. 3.30: Blur - Fidelity of the resulting stabilized sequence after denoising in features rich dataset

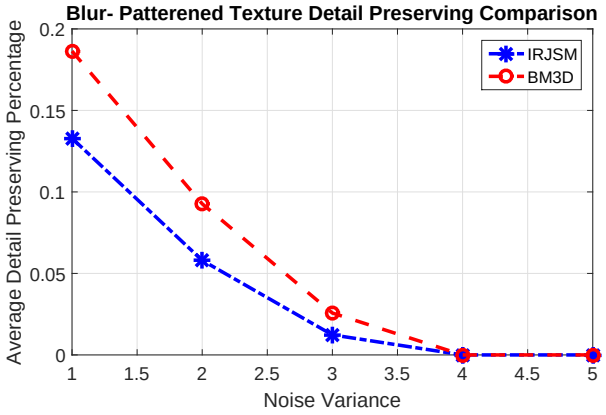


Fig. 3.31: Blur - Percentage of matches recovered after using each denoising algorithm in patterned texture datasets

the best stabilization results. Though for videos rich in features or with large flat areas BM3D proved to outperform other algorithms.

- For Salt and pepper, IRSJM preserved more features hence producing better videos form the stabilization process.

- In case of blurring, BM3D outperformed other algorithms preserving important image features for video stabilization.

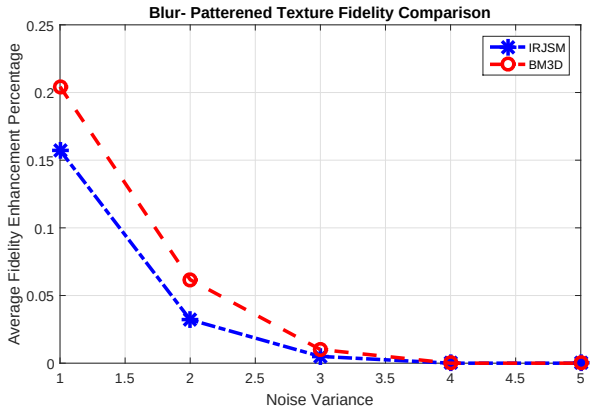


Fig. 3.32: Blur - Fidelity of the resulting stabilized sequence after denoising algorithm in patterned texture datasets.

3.7 Conclusion

This chapter highlights the importance of the choice of the denoising algorithm on the outcome of the stabilization. As an inappropriate choice of denoising algorithm may lead to the loss of important information. It also shows that the most commonly used measures does not always reflect the best denoising choice. Hence, a new feature-matching based evaluation measure has been presented to evaluate the relation between the denoising algorithm and the stabilization quality. This measure depends on the accuracy of the matching algorithm used. Not only, we also highlight that choosing the latest denoising algorithm will not always guarantee the best results. But also,we highlight that the video content may greatly affect the choice of the technique.

Chapter 4

StableFlow: A Novel Video Stabilization Method

4.1 Introduction

This chapter introduces StableFlow a novel method for video stabilization. StableFlow falls in the 2D stabilization method category, and is based on a novel physics model to estimate the 2D transformations and generate a stable video. A detailed description of StableFlow, implementation details and obtained results is introduced in this chapter.

4.2 StableFlow

In this section, we introduce StableFlow, a novel two pass digital video stabilization real-time method inspired by the mass spring damper model. The method eliminates the undesired motion in the video based on the impulse response from the mass spring model in the first pass, then eliminates the high frequency jitter using a Gaussian low-pass filter in the second pass as shown in figure 4.1. In the first pass, the video frame is modelled as a mass suspended in each direction by critically dampened spring and damper. The

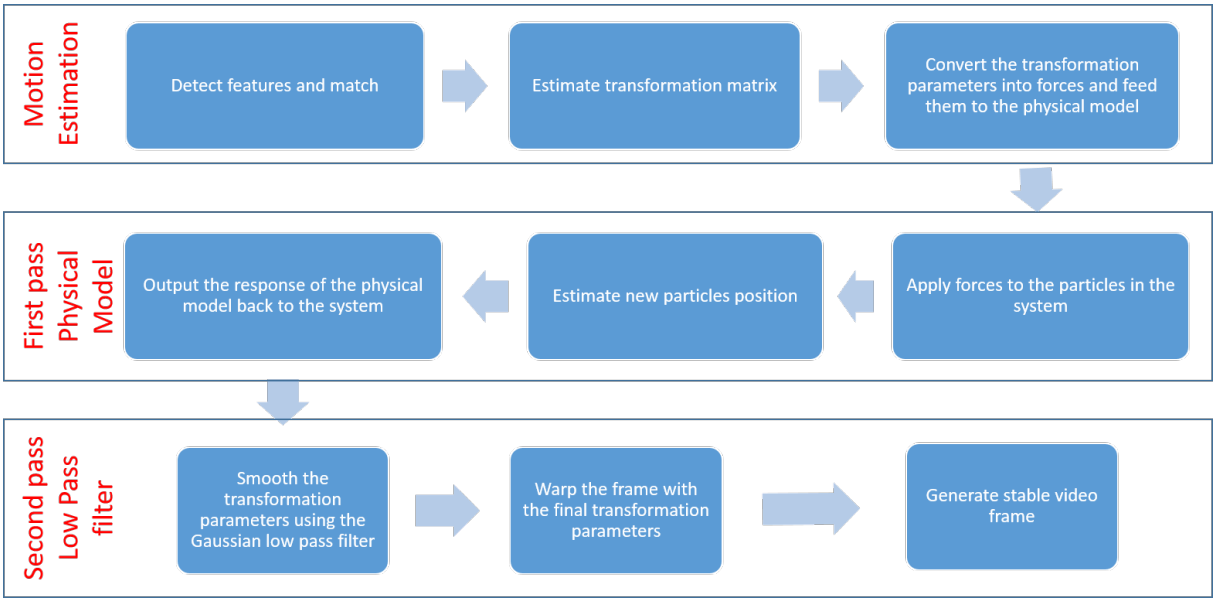


Fig. 4.1: Flowchart of the proposed algorithm

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 prevent undesired oscillations back and forth. The system is parametric and can be fine-tuned to adapt with various types of motion including rotation, translation or combination of both.

The proposed method is novel and unique in the following aspects:

1. It 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.

2. No a priori knowledge of the camera motion is required

3. Meets real time requirements.

4. It is highly parallelizable as the calculation for each node in the physical model can run in parallel.

5. The results outperforms current state-of-the-art methods.

4.2.1 Motion Estimation

The first block in the proposed algorithm is the Motion estimation as seen in Figure 4.1. Motion estimation is the process of estimating transformation parameters between successive frames in a video, hence recovering the camera path (trajectory). Given a set of frames T_i, $i = 0, 1, 2, ...$, assuming the transformation between frames T_i and T_{i+1} as affine transforms H_i, the camera path will be the product of all the preceding inter-frame transforms as follows:

$$H_{cummlative,i} = \prod_{i=0}^{i-1} H_i \tag{4.1}$$

As shown in Figure 4.1, the first step in motion estimation is feature detection and matching to identify key unique image features in the video frame. In the literature, a lot of different methods is found as Harris corner detector [54], SIFT [55], SURF [56], ORB [57], FREAK [58]. After that the features are matched over successive frames. In SteadyFlow, feature detection and matching is done according to the paper proposed by [59] to detect corners and match them; the main advantage of this implementation is its robustness and low computational complexity. The second step in motion estimation is to estimate the transformation parameters; in this step an affine transformation H is estimated between the matches found between successive frames, then, RASNSAC is used to filter the outliers in the matches and estimate the best transformation between the frames. The last step is transforming the estimated translation and rotation in the affine matrix H into forces to be fed to the physical system, which is done be multiplying the translation ($H_{Translation}$) and the rotation ($H_Rotation$) with the delta time between frames Δt as follows:

$$F_t = H_{Translation} * \Delta t \tag{4.2}$$

$$F_{theta} = H_{Rotation} * \Delta t \tag{4.3}$$

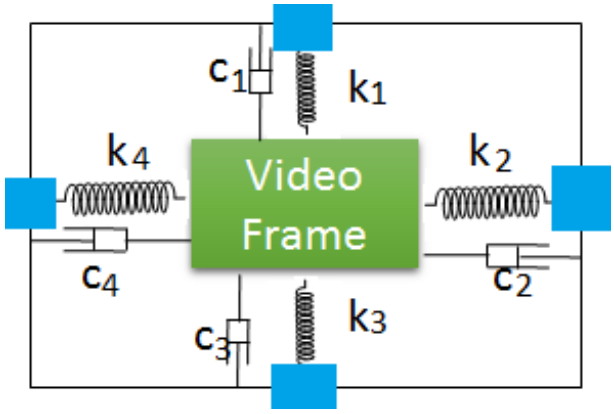


Fig. 4.2: Mass spring model for digital video stabilization.

where $F_t and F_{theta}$ are the translation and rotational forces respectively.

4.2.2 First Pass - Physical Model

Apply forces to the particles in the system

The proposed method can be described using the physics model shown in Figure 4.2. 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 the forces due to gravity are not considered in the formulation of the system equations $mg = 1$ as seen in Figure 4.3. In the proposed model, the frame can move freely, but the springs suppress the undesired motion. Moreover, the dampers, which can be fine-tuned to allow the system to converge to the steady state quickly, prevent the frame from undesired oscillations. From the video stabilization point of view, there are two types of forces that lead to shaky videos: (1) translation and (2) rotation; which are described next.

Translational Forces

The analysis of the system forces in 1-Dimension is shown in Figure 4.3. The first force acting on the frame is F_{spring} which is generated from stretching the spring and acts in

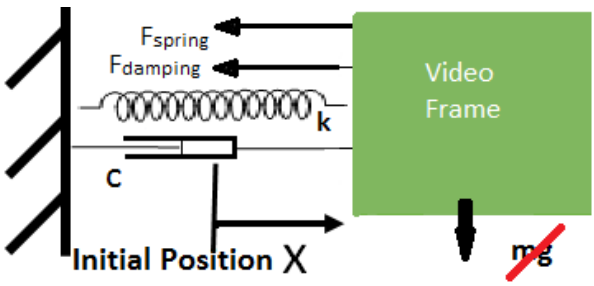


Fig. 4.3: Model in 1-Dimension.

the opposite direction of the stretch; F_{spring} is defined by:

$$F_{spring} = -kx \tag{4.4}$$

where k is the spring constant, and x is the stretch length of the spring

In addition, there is a damping force $F_{damping}$ that resists the motion and is proportional to the velocity $\dot{x}$, $F_{damping}$ is defined by:

$$F_{damping} = -c\dot{x} \tag{4.5}$$

where c is the damping coefficient

Therefore, the total force acting on the frame will be:

$$F = F_{spring} + F_{damping} = -kx - c\dot{x} \tag{4.6}$$

From Newton's law of motion we have:

$$\sum f = m\ddot{x} \tag{4.7}$$

where $\ddot{x}$ is the acceleration

$$m\ddot{x} + c\dot{x} + kx = 0 \tag{4.8}$$

Moreover, the motion of this frame also depends on what is called the damping ratio ζ which is given by [60]:

$$\zeta = \frac{c}{2\sqrt{km}} \tag{4.9}$$

$$\zeta \begin{cases} < 1, \text{ Under-damped system and will keep oscillating.} \\ > 1, \text{ Over-damped, system will be shaky.} \\ = 1, \text{ Optimally damped.} \end{cases} \tag{4.10}$$

To make the formulation of the model simpler, we assume the mass of the frame to be a unit mass $m = 1$ as the mass of video frame has no physical meaning and setting the mass to a certain value will only lead to a scale in the response of the physical system. As illustrated above in Equation 4.10, to obtain an optimal damping $\zeta = 1$, equation 4.9 becomes:

$$c = 2\sqrt{k} \tag{4.11}$$

Substituting in Equation 4.8, we get:

$$\ddot{x} + 2\sqrt{k}\dot{x} + kx = 0 \tag{4.12}$$

Rotational Forces

The analysis for rotational forces is similar to the analysis of translation forces but with a rotational spring as shown in Figure 4.4. Similar to equation 4.12, the equation for the rotation will be:

$$\ddot{\theta} + 2\sqrt{k_r}\dot{\theta} + k_r\theta = 0 \tag{4.13}$$

where $\ddot{\theta}$ represents the angular acceleration, $\dot{\theta}$ is the angular velocity and θ is the rotation angle, k_r is the rotational spring constant.

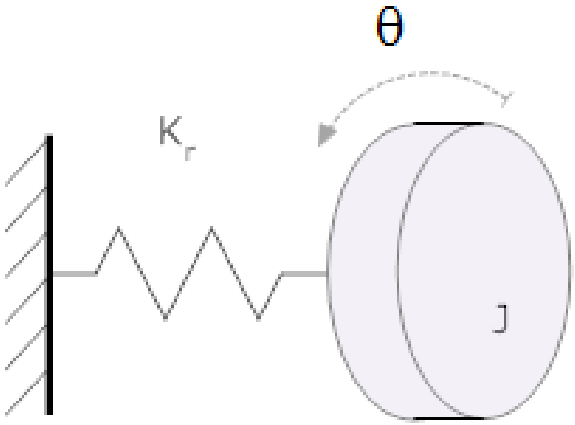


Fig. 4.4: Rotational Model.

Estimate new particles position

To estimate new postions of the particles, we need to solve Equation 4.12 numerically (i.e., using computer program) using the Runge-Kutta method [61] Therefore, we must convert the second order differential equations 4.12 and 4.13 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 4.12 can be expressed as a system of two first order differential equations:

$$\dot{x} = v \tag{4.14}$$

$$\dot{v} = -kx - 2\sqrt{k}\dot{x} \tag{4.15}$$

Equations 4.14 and 4.15 represent the form needed in order to use the Runge-Kutta method to numerically solve the differential equation in Equation 4.12. To solve using Runge-Kutta, four variables k_1, k_2, k_3 and k_4 have to be evaluated using:

$$k_1 = f(t, y) \tag{4.16}$$

where t is time and y is the function to be approximated and f is the integration function

$$k_2 = f(t + \Delta T/2, y + \Delta T/2k_1) \tag{4.17}$$

$$k_3 = f(t + \Delta T/2, y + \Delta T/2k_2) \tag{4.18}$$

$$k_4 = f(t + \Delta T, y + \Delta T k_3) \tag{4.19}$$

where ΔT is the time interval between two successive frames.

Next, the integration will be used to update the new value of the function being approximated:

$$y_{n+1} = y_n + \frac{\Delta T}{6}(k_1 + 2k_2 + 2k_3 + k_4) \tag{4.20}$$

In the case above, the procedure will be executed for both translation and rotation as in the following equations:

$$T_{n+1} = T_n + \Delta T/6(k_1 + 2k_2 + 2k_3 + k_4) \tag{4.21}$$

$$\theta_{n+1} = \theta_n + \Delta T/6(k_1 + 2k_2 + 2k_3 + k_4) \tag{4.22}$$

Algorithm 1 RK4 Method

1: **Input:** ΔT.
2: **for** each mass **do**
3: Compute k_1 using equation 4.16 .
4: Compute k_2 using equation 4.17 .
5: Compute k_3 using equation 4.18 .
6: Compute k_4 using equation 4.19 .
7: Compute velocity and position using equation 4.20.
8: **end for**
9: **Output:** New position of the system particles.

As seen in the equations 4.21 and 4.22, the equations is based on an iterative solution

which keeps on running until convergence i.e. the difference in value between iterations becomes negligible less than 0.0001 or until it reaches the maximum number of iterations which has been set to 30 in our implementation.

Output the response from the Physical System

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

$$P_t = \prod_{i=0}^{i} H_i. \tag{4.23}$$

The stabilized camera path $\acute{P}_t$ can be computed through the convolution of the path with the response of the physical model $A(t)$ as shown in Equation 4.24:

$$\acute{P}_t = P_t * A(t) \tag{4.24}$$

where $A(t)$ is an affine matrix that represents the physical response of the system.
And the difference in the particle position represent the translation part of matrix $A(t)$, i.e. the horizontal translation will equal the current horizontal position subtracted by the previous position, similarly the vertical translation is calculated. Also, the difference in the angle represents the rotation angle used for the matrix $A(t)$.

4.2.3 Second Pass - Gaussian Low pass filter

Smooth the transformation parameters using the Gaussian low pass filter

The second pass of the proposed algorithm is a standard Gaussian low pass filter with zero mean and a default standard deviation $\sigma = 0.5$. Given the smoothed path produced from the first pass as in Equation 4.24. The low pass filter is applied to the smooth

camera path over a window in time for N frames through convolution to suppress any high frequency jitter. The smoothed parameters are calculated as follows:

$$\hat{P}_t = \acute{P}_t * \acute{G}_t \tag{4.25}$$

where $\acute{G}_t$ is the low pass Gaussian filter. Using the gaussian low pass filter, gives the algorithm the advantage of suppressing any sudden jitter in the transformation parameters over time. Then, the next step is using the produced transformation matrix to re-align the frame using the built-in OpenCV warping function. The final step is rendering the video frame and writing it to the video file.

4.2.4 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 and is implemented based on the paper GoodFeaturesToTrack [59]. In the first pass, the proposed method then converts the calculated affine transformation into forces using equation 4.2 and4.3 and inputs it to the physical model. After that, the Runge-kutta method is used to solve the model and estimate the position and velocity of the frame given the set of springs and dampers as shown in Figures 4.2 and 4.3. Finally, in the second pass, the Gaussian filter is applied on the transformation parameters over a window of 15 frames to compute the final stabilized frame is computed through warping the input frame with the estimated trajectory from the second pass of the algorithm.

Algorithm 2 The Proposed Method

1: **Input:** Input Frame.
2: Extract good features to track [59].
3: Track the features into previous frame [59].
4: Evaluate good matches [59].
5: Estimate affine transformation.
6: Convert transformation to forces and feed them to the mass spring model.
7: Calculate the damping coefficient using Equation 4.11.
8: Calculate the position and velocity using Runge-Kutta as shown in Algorithm 1
9: Calculate a smoothed trajectory based on the physical model response .
10: For each frame over the window of the N frames, the trajectory is smoothed using a Gaussian low pass filter.
11: Calculate the final smoothed trajectory and calculate its transformation matrix.
12: Generate the final stabilized frame by warping the input frame with the transformation matrix estimated from the smoothed trajectory.
13: **Output:** Stabilized Frame.

4.3 Results

The proposed algorithm has been tested on different videos from the datasets provided by [53] [31] and [14] and other online videos. Some of the tested videos can be found on our webpage (http://stableflow.weebly.com/). The evaluation criteria are based on: 1) the mean of the sequence to assess the visual quality improvement, 2) the average fidelity [23] (using Equation 4.26), 3) the camera trajectory in x, y, and θ, and 4) cropping percentage

$$PSNR_{dB}(I_1, I_0) = 10 \log \frac{(255)^2}{MSE(I_1, I_0)} \tag{4.26}$$

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 4.5 shows a comparison of the mean for the stabilized sequences using Google's YouTube stabilizer [31], Adobe After Effects [34] and the proposed method.Table 4.1 compares the fidelity values from the original video, the proposed method , Google's Youtube

(a) Original Mean of Sequence

(b) Our Stabilized Sequence Mean

(c) Google's Youtube Stabilized Sequence Mean

(d) Adobe After Effects Stabilized Sequence Mean

Fig. 4.5: Sequence Mean Comparison

and Adobe After Effects method. Figures 4.7 - 4.15, shows the trajectories comparison in vertical, horizontal and rotational angle respectively for some of the sequences. For a good cropping the value should be close to zero. Figure4.6 compares the cropping for the different methods. It is clear that the proposed method preserves more of the original frame and has the least cropping percentage.

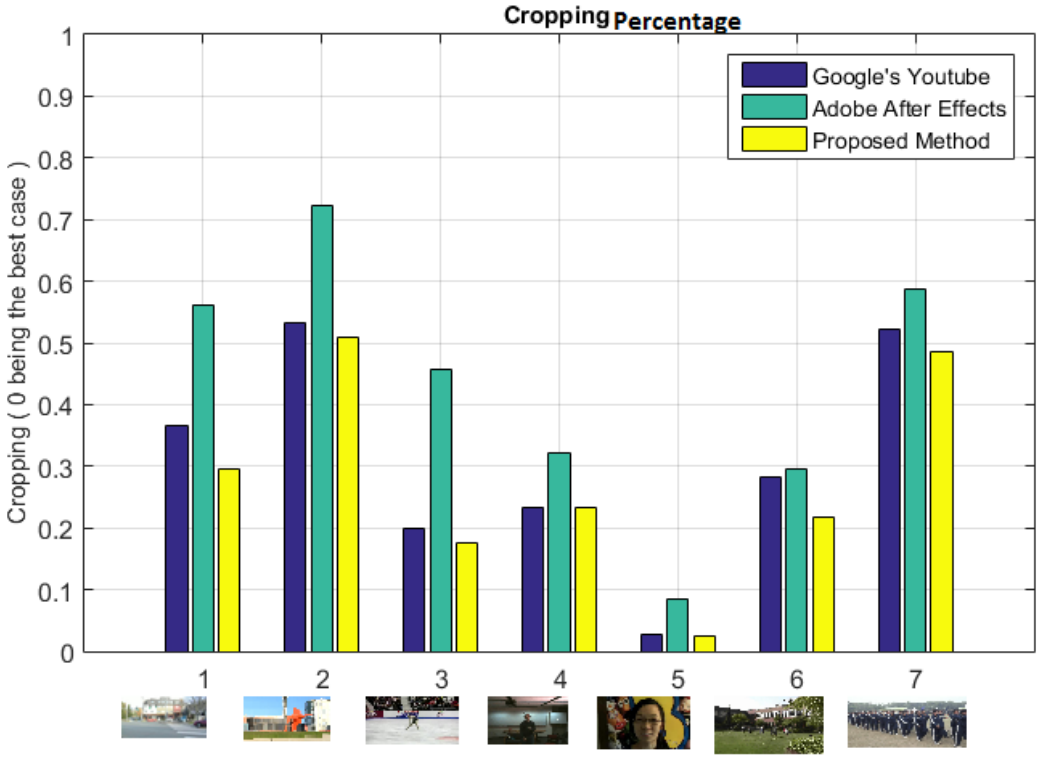


Fig. 4.6: Cropping Percentage comparison.

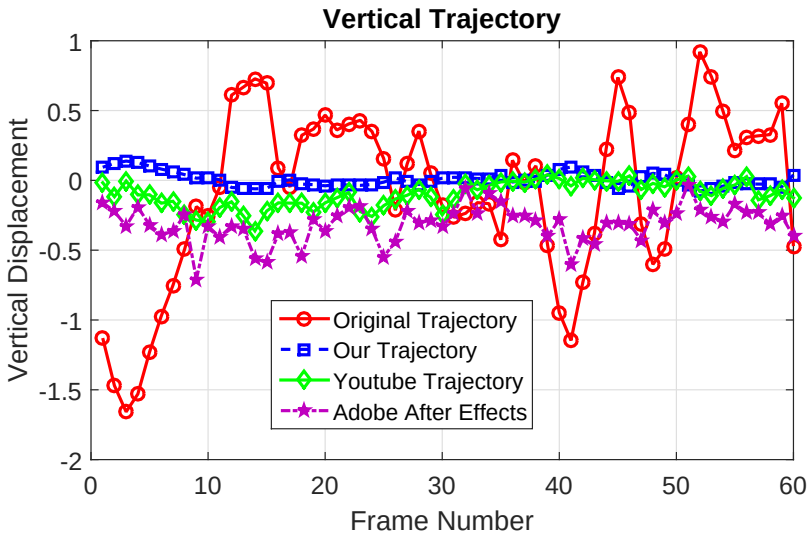


Fig. 4.7: Sequence 4 - Vertical Trajectory

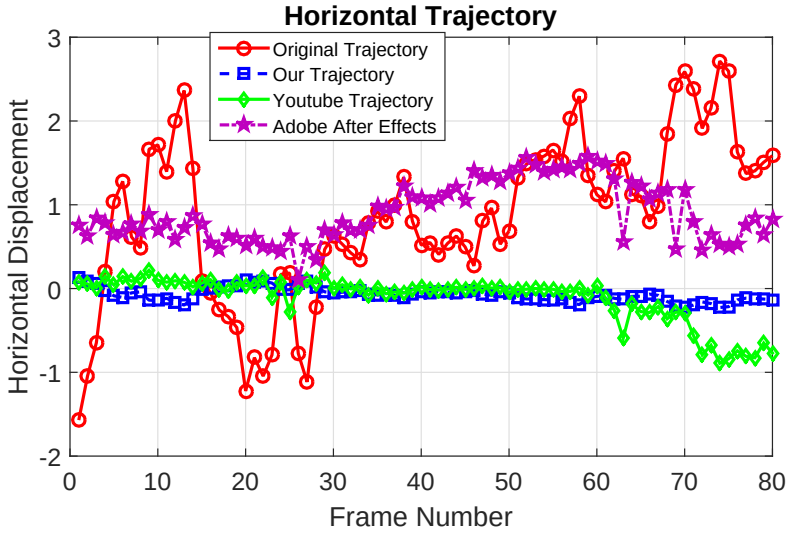


Fig. 4.8: Sequence 4 - Horizontal Trajectory

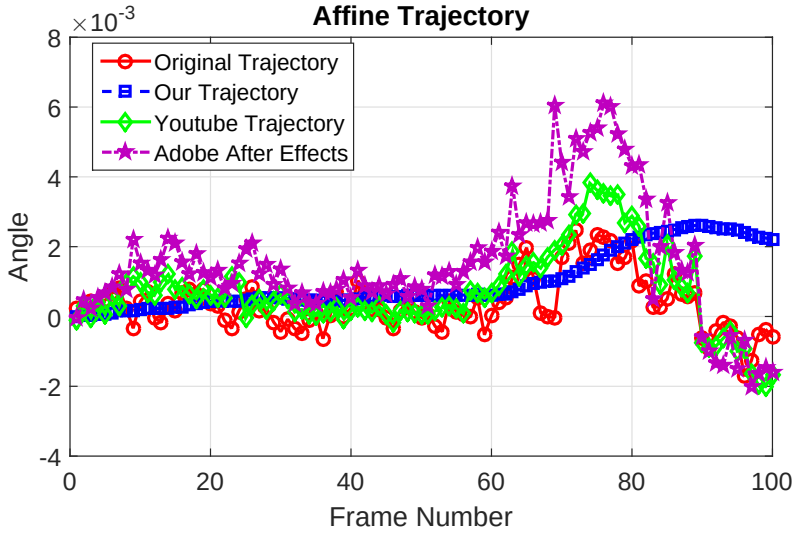


Fig. 4.9: Sequence 4 - Rotational Trajectory

Table 4.1: 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	Adobe After Effects
Seq. 1	42 dB	62 dB	59 dB	56
Seq. 2	34 dB	51 dB	50 dB	47
Seq. 3	12.38 dB	14.2 dB	13.75 dB	13.4
Seq. 4	9.4 dB	13.4 dB	10.2 dB	9.8

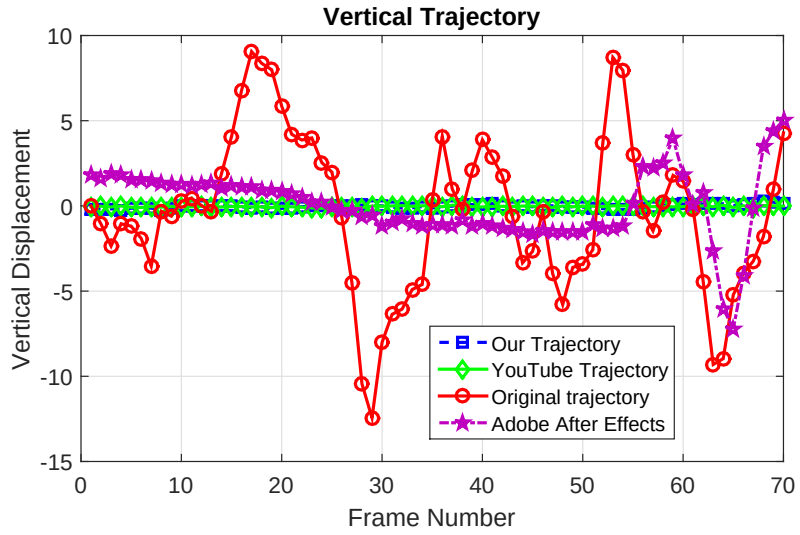


Fig. 4.10: Sequence 2 - Vertical Trajectory

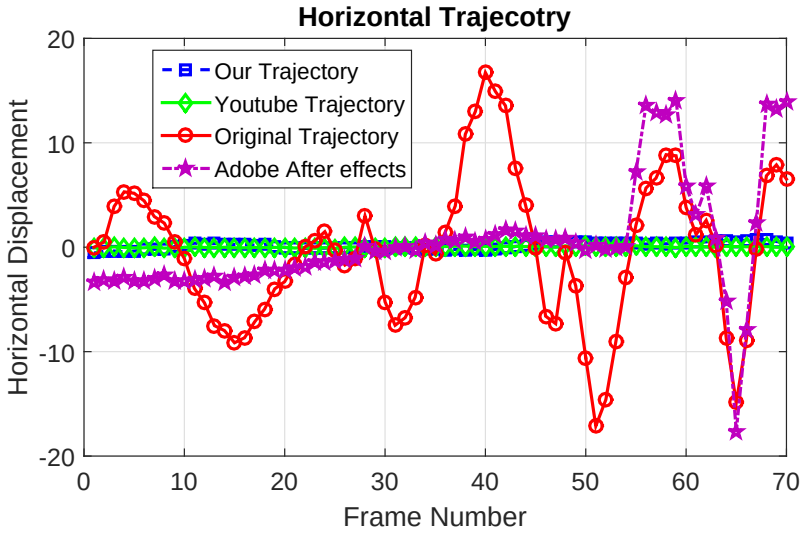


Fig. 4.11: Sequence 2 -Horizontal Trajectory

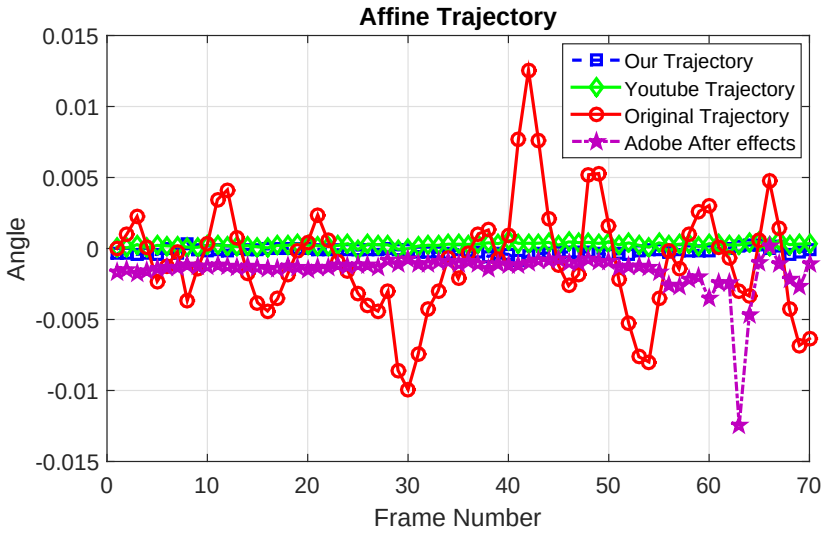


Fig. 4.12: Sequence 2 -Rotational Trajectory

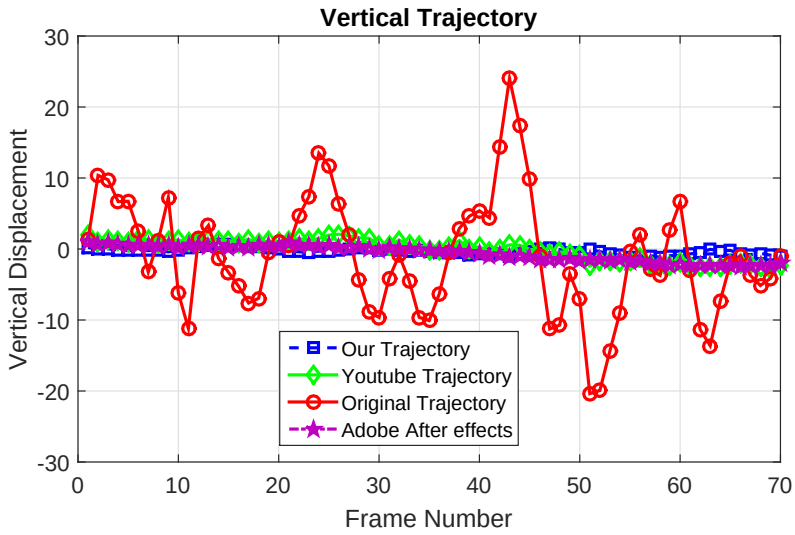


Fig. 4.13: Sequence 3 -Vertical Trajectory

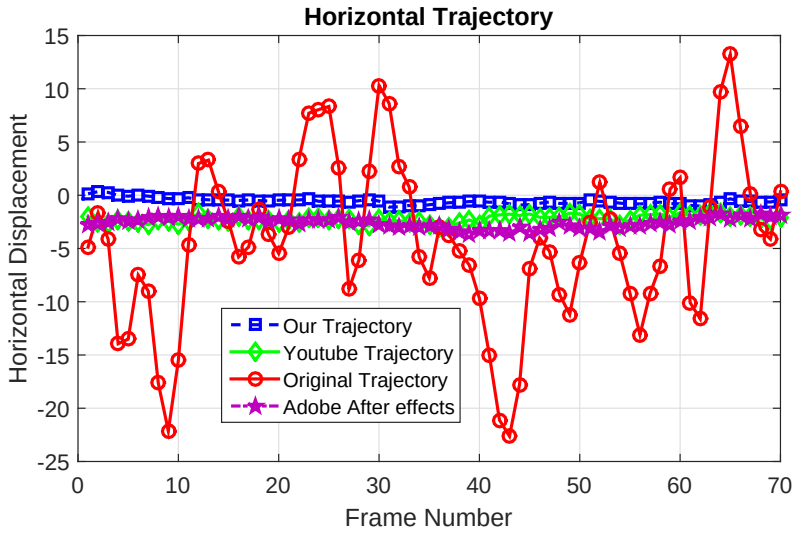


Fig. 4.14: Sequence 3 -Horizontal Trajectory

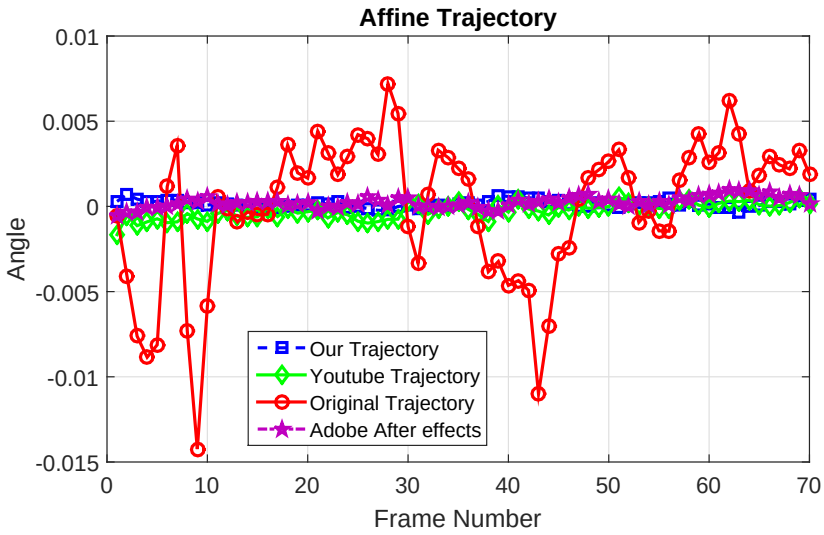


Fig. 4.15: Sequence 3 -Rotational Trajectory

4.4 Discussion and Performance Analysis

4.4.1 Discussion

The proposed method outperformed the current state-of-the-art method used on Google's Youtube and Adobe After Effects in the video sequences being tested. As seen in Figures 4.7 - 4.15, the proposed method generates a stable camera path that is either better or comparable to the state-of-the-art methods. Also the proposed method produced videos with higher fidelity as shown in Table4.1. 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.

Choice of the Spring Constant

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.

Since the gaussian low pass filter in the second pass of the algorithm suppress any sudden jitter, therefore , instead of choosing a value for each video, the problem can be optimized to choose the spring constant that produces the best quality after the application of the second pass over the different datasets. Generally, the test videos used in this thesis are classified into two categories 1) Videos with high frequency jitter and 2) Videos

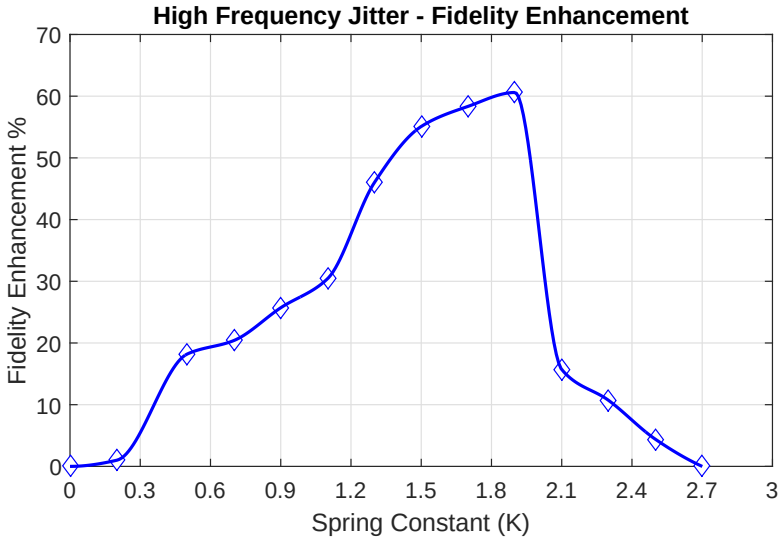


Fig. 4.16: Relation between Spring Constant and Video Fidelity for videos with high frequency jitter

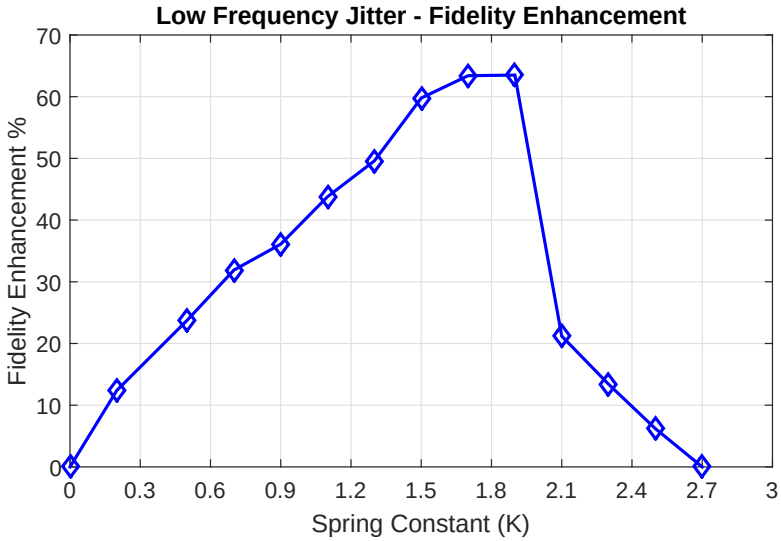


Fig. 4.17: Relation between Spring Constant and Video Fidelity for videos with low frequency jitter

Table 4.2: Running time in Frames per second for each of the tested sequences

Sequence	Motion Estimation (fps)	First Pass Running Time (fps)	Second Pass Running Time (fps)	Overall
Seq. 1	20.7	59.2	140	20.7
Seq. 2	18.7	57.6	153	18.7
Seq. 3	20.27	56.8	165	20.27
Seq. 4	19.2	58.7	133	19.2

with low frequency jitter. The algorithm is applied with spring constant ranging from 0.2 to 2.7 with a step of 0.1. As shown in Figures 4.16 and 4.17, the spring constant $k = 1.9$ gives the best results in both cases. Hence, it was used in all the results produced in the previous subsections.

4.4.2 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 4.2. It is worth mentioning that the first step of the motion estimation block which is feature detection and matching is the most computationally expensive step for the whole algorithm.

4.5 Conclusion

In this chapter, we presented a novel stabilization method based on 2D linear transformations that meets real time requirement. The proposed method outperforms current state-of-the-art methods, such as Youtube and Adobe After Effects "WrapStabilizer" function, while achieving real time performance. The proposed method novelty can be summarized in the following

1. It 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.

2. No a priori knowledge of the camera motion.

3. Real time performance.

4. Highly parallelizable

Future work include modifying the method to be used on feature-less scenes using pixel trajectory. Also, deployment on various automated systems.

Chapter 5

Conclusion and Future Work

5.1 Summary

In this thesis, we have proposed a novel solution to the problem of video stabilization. In chapter 1, we introduced the problem of video stabilization and discussed the challenges related to this topic. These challenges can be summarized as follows:

1. Noise
2. Quick Camera motion
3. High Computational Complexity

According to the adopted motion models, video stabilization methods can be categorized into 2D, 3D and 2.5D . In chapter 2, we reviewed most related video stabilization approaches based on these categories.

The contribution of this thesis consists of :

1. Introducing a novel study of the effect of denoising algorithms on Video Stabilization.

2. Introducing a novel physics inspired video stabilization method.

Chapter 3 presented the study of the effect of denoising on video stabilization. We studied the effect of different types of noise along with the state-of-the-art denoising techniques on a diverse dataset, from the obtained results we showed that choosing the recent method of denoising does not always guarantee the best results. In case of AWGN, it is clear from the obtained results that BM3D outperformed other algorithms, however, in case of videos with patterned texture, IRJSM outperformed BM3D and generated better images. In case of Salt and pepper, IRJSM outperformed the other algorithm over all the video types. In case of Blurring, BM3D outperformed BM3D on all the datasets.

Chapter 4 presented a two pass novel physics inspired 2D video stabilization method "StableFlow". The proposed method outperforms state-of-the-art methods while keeping real time performance. We proposed to use mass spring damper model to suppress the low frequency jitter in the video. Then in the second pass we further smoothed the transformation parameters using a gussian low pass filter. We demonstrated the advantages of the StableFlow by stabilizing challenging videos with extreme shakiness.

5.2 Future Work

Future work include introducing more measures to assess the effect of denoising on stabilization on feature-less scenes. Also, extending the StableFlow method to handle feature-less scenes and employ pixel trajectories.

References

[1] SteadiCam, "Steadicam," http://tiffen.com/steadicam/.

[2] C. Inc., "Cannon inc." https://www.usa.canon.com/internet/portal/us/home/products/.

[3] R. Szeliski, *Computer Vision: Algorithms and Applications*, 1st ed. New York, NY, USA: Springer-Verlag New York, Inc., 2010.

[4] R. C. Gonzalez and R. E. Woods, *Digital Image Processing (3rd Edition)*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 2006.

[5] Y. Matsushita, E. Ofek, W. Ge, X. Tang, and H.-Y. Shum, "Full-frame video stabilization with motion inpainting," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 7, pp. 1150–1163, 2006.

[6] M. L. Gleicher and F. Liu, "Re-cinematography: Improving the camerawork of casual video," *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, vol. 5, no. 1, p. 2, 2008.

[7] A. Ahmed and M. S. Shehata, "Towards high-quality parallel stabilization," in *Proceedings of the 11th Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications*, 2016, pp. 665–670.

[8] 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*, vol. 54, no. 10, pp. 105–112, 2011.

[9] 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*. IEEE, 2010, pp. 1434–1441.

[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*. IEEE, 2012, pp. 1458–1465.

[11] C. Wu, "Towards linear-time incremental structure from motion," in *3D Vision-3DV 2013, 2013 International Conference on*. IEEE, 2013, pp. 127–134.

[12] 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*, vol. 2. IEEE, 2001, pp. II–609.

[13] B. Triggs, P. F. McLauchlan, R. I. Hartley, and A. W. Fitzgibbon, "Bundle adjustment—a modern synthesis," in *Vision algorithms: theory and practice*. Springer, 2000, pp. 298–372.

[14] F. Liu, M. Gleicher, H. Jin, and A. Agarwala, "Content-preserving warps for 3d video stabilization," in *ACM Transactions on Graphics (TOG)*, vol. 28, no. 3. ACM, 2009, p. 44.

[15] 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*. IEEE, 2012, pp. 89–95.

[16] A. Goldstein and R. Fattal, "Video stabilization using epipolar geometry," *ACM Transactions on Graphics (TOG)*, vol. 31, no. 5, p. 126, 2012.

[17] B. K. Horn and B. G. Schunck, "Determining optical flow," *Artificial intelligence*, vol. 17, no. 1-3, pp. 185–203, 1981.

[18] H.-C. Chang, S.-H. Lai, and K.-R. Lu, "A robust and efficient video stabilization algorithm," in *Multimedia and Expo, 2004. ICME'04. 2004 IEEE International Conference on*, vol. 1. IEEE, 2004, pp. 29–32.

[19] R. Hu, R. Shi, I.-f. Shen, and W. Chen, "Video stabilization using scale-invariant features," in *Information Visualization, 2007. IV'07. 11th International Conference*. IEEE, 2007, pp. 871–877.

[20] D. Pang, H. Chen, and S. Halawa, "Efficient video stabilization with dual-tree complex wavelet transform," *EE368 Project Report, Spring*, 2010.

[21] H. Farid and J. B. Woodward, "Video stabilization and enhancement," *TR2007-605, Dartmouth College, Computer Science*, 1997.

[22] O. Adda, N. Cottineau, and M. Kadoura, "A tool for global motion estimation and compensation for video processing," *LEC/COEN*, vol. 490, 2003.

[23] 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*, vol. 5. IEEE, 1998, pp. 2789–2792.

[24] A. Karpenko, D. Jacobs, J. Baek, and M. Levoy, "Digital video stabilization and rolling shutter correction using gyroscopes," *CSTR*, vol. 1, p. 2, 2011.

[25] M. Hansen, P. Anandan, K. Dana, G. Van der Wal, and P. Burt, "Real-time scene stabilization and mosaic construction," in *Applications of Computer Vision, 1994., Proceedings of the Second IEEE Workshop on.* IEEE, 1994, pp. 54–62.

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

[27] A. Litvin, J. Konrad, and W. C. Karl, "Probabilistic video stabilization using kalman filtering and mosaicing," in *Electronic Imaging 2003.* International Society for Optics and Photonics, 2003, pp. 663–674.

[28] 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.* IEEE, 1994, pp. 454–460.

[29] S. Liu, L. Yuan, P. Tan, and J. Sun, "Bundled camera paths for video stabilization," *ACM Transactions on Graphics (TOG) (Proceedings of SIGGRAPH 2013)*, vol. 32, no. 4, 2013.

[30] ——, "Steadyflow: Spatially smooth optical flow for video stabilization," in *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on.* IEEE, 2014, pp. 4209–4216.

[31] 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.* IEEE, 2011, pp. 225–232.

[32] T. Igarashi, T. Moscovich, and J. F. Hughes, "As-rigid-as-possible shape manipulation," in *ACM transactions on Graphics (TOG)*, vol. 24, no. 3. ACM, 2005, pp. 1134–1141.

[33] 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*, vol. 19, no. 8, pp. 1354–1361, 2013.

[34] F. Liu, M. Gleicher, J. Wang, H. Jin, and A. Agarwala, "Subspace video stabilization," *ACM Transactions on Graphics (TOG)*, vol. 30, no. 1, p. 4, 2011.

[35] M. Irani, "Multi-frame correspondence estimation using subspace constraints," *International Journal of Computer Vision*, vol. 48, no. 3, pp. 173–194, 2002.

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

[37] A. Buades, B. Coll, and J.-M. Morel, "A review of image denoising algorithms, with a new one," *Multiscale Modeling & Simulation*, vol. 4, no. 2, pp. 490–530, 2005.

[38] N. Wiener, *Extrapolation, interpolation, and smoothing of stationary time series*. MIT press Cambridge, MA, 1949, vol. 2.

[39] F. Jin, P. Fieguth, L. Winger, and E. Jernigan, "Adaptive wiener filtering of noisy images and image sequences," in *Image Processing, 2003. ICIP 2003. Proceedings. 2003 International Conference on*, vol. 3. IEEE, 2003, pp. III–349.

[40] J. Chen, J. Benesty, Y. Huang, and S. Doclo, "New insights into the noise reduction wiener filter," *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 14, no. 4, pp. 1218–1234, 2006.

[41] D. L. Donoho and J. M. Johnstone, "Ideal spatial adaptation by wavelet shrinkage," *Biometrika*, vol. 81, no. 3, pp. 425–455, 1994.

[42] S. G. Chang, B. Yu, and M. Vetterli, "Adaptive wavelet thresholding for image denoising and compression," *Image Processing, IEEE Transactions on*, vol. 9, no. 9, pp. 1532–1546, 2000.

[43] D. L. Donoho and I. M. Johnstone, "Adapting to unknown smoothness via wavelet shrinkage," *Journal of the american statistical association*, vol. 90, no. 432, pp. 1200–1224, 1995.

[44] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian, "Image denoising by sparse 3-d transform-domain collaborative filtering," *Image Processing, IEEE Transactions on*, vol. 16, no. 8, pp. 2080–2095, 2007.

[45] ——, "Bm3d image denoising with shape-adaptive principal component analysis," in *SPARS'09-Signal Processing with Adaptive Sparse Structured Representations*, 2009.

[46] Y. Wang, W. Ren, and H. Wang, "Anisotropic second and fourth order diffusion models based on convolutional virtual electric field for image denoising," *Computers & Mathematics with Applications*, vol. 66, no. 10, pp. 1729–1742, 2013.

[47] J. Zhang, D. Zhao, R. Xiong, S. Ma, and W. Gao, "Image restoration using joint statistical modeling in a space-transform domain," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 24, no. 6, pp. 915–928, 2014.

[48] S. Zhang and M. A. Karim, "A new impulse detector for switching median filters," *IEEE Signal processing letters*, vol. 9, no. 11, pp. 360–363, 2002.

[49] Y. Dong and S. Xu, "A new directional weighted median filter for removal of random-valued impulse noise," *Signal Processing Letters, IEEE*, vol. 14, no. 3, pp. 193–196, 2007.

[50] C.-T. Lu and T.-C. Chou, "Denoising of salt-and-pepper noise corrupted image using modified directional-weighted-median filter," *Pattern Recognition Letters*, vol. 33, no. 10, pp. 1287–1295, 2012.

[51] R. Fergus, B. Singh, A. Hertzmann, S. T. Roweis, and W. T. Freeman, "Removing camera shake from a single photograph," *ACM Transactions on Graphics (TOG)*, vol. 25, no. 3, pp. 787–794, 2006.

[52] ACENET, "Acenet computing cluster," http://www.ace-net.ca/, 2003.

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

[54] C. Harris and M. Stephens, "A combined corner and edge detector," in *Proceedings of the 4th Alvey Vision Conference*, 1988, pp. 147–151.

[55] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vision*, vol. 60, no. 2, pp. 91–110, Nov. 2004. [Online]. Available: http://dx.doi.org/10.1023/B:VISI.0000029664.99615.94

[56] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, "Speeded-up robust features (surf)," *Comput. Vis. Image Underst.*, vol. 110, no. 3, pp. 346–359, Jun. 2008. [Online]. Available: http://dx.doi.org/10.1016/j.cviu.2007.09.014

[57] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "Orb: An efficient alternative to sift or surf," in *Proceedings of the 2011 International Conference on Computer Vision*, ser. ICCV '11. Washington, DC, USA: IEEE Computer Society, 2011, pp. 2564–2571. [Online]. Available: http://dx.doi.org/10.1109/ICCV.2011.6126544

[58] R. Ortiz, "Freak: Fast retina keypoint," in *Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, ser. CVPR