Universidade Estadual de Campinas
Instituto de Computação

Marcos Roberto e Souza

Digital Video Stabilization: Algorithms and Evaluation

Estabilização Digital de Vídeos: Algoritmos e Avaliação

CAMPINAS
2018

Marcos Roberto e Souza

Digital Video Stabilization: Algorithms and Evaluation

Estabilização Digital de Vídeos: Algoritmos e Avaliação

Thesis presented to the Institute of Computing of the University of Campinas in partial fulfillment of the requirements for the degree of Master in Computer Science.

Dissertação apresentada ao Instituto de Computação da Universidade Estadual de Campinas como parte dos requisitos para a obtenção do título de Mestre em Ciência da Computação.

Supervisor/Orientador: Prof. Dr. Hélio Pedrini

Este exemplar corresponde à versão final da Dissertação defendida por Marcos Roberto e Souza e orientada pelo Prof. Dr. Hélio Pedrini.

CAMPINAS
2018

Agência(s) de fomento e nº(s) de processo(s): CAPES

Ficha catalográfica
Universidade Estadual de Campinas
Biblioteca do Instituto de Matemática, Estatística e Computação Científica
Ana Regina Machado - CRB 8/5467

So89d

Souza, Marcos Roberto e, 1994-
Digital video stabilization : algorithms and evaluation / Marcos Roberto e Souza. – Campinas, SP : [s.n.], 2018.

Orientador: Hélio Pedrini.
Dissertação (mestrado) – Universidade Estadual de Campinas, Instituto de Computação.

1. Processamento de imagens. 2. Otimização matemática. 3. Vídeo digital. 4. Sistemas multimídia. 5. Estimativa de parâmetro. 6. Visão por computador. I. Pedrini, Hélio, 1963-. II. Universidade Estadual de Campinas. Instituto de Computação. III. Título.

Informações para Biblioteca Digital

Título em outro idioma: Estabilização digital de vídeos : algoritmos e avaliação
Palavras-chave em inglês:
Image processing
Mathematical optimization
Digital video
Multimedia systems
Parameter estimation
Computer vision
Área de concentração: Ciência da Computação
Titulação: Mestre em Ciência da Computação
Banca examinadora:
Hélio Pedrini [Orientador]
Fabio Augusto Faria
André Santanchè
Data de defesa: 23-02-2018
Programa de Pós-Graduação: Ciência da Computação

Universidade Estadual de Campinas
Instituto de Computação

Marcos Roberto e Souza

Digital Video Stabilization: Algorithms and Evaluation

Estabilização Digital de Vídeos: Algoritmos e Avaliação

Banca Examinadora:

- Prof. Dr. Hélio Pedrini
  IC/UNICAMP
- Prof. Dr. Fabio Augusto Faria
  ICT/UNIFESP
- Prof. Dr. André Santanchè
  IC/UNICAMP

A ata da defesa com as respectivas assinaturas dos membros da banca encontra-se no processo de vida acadêmica do aluno.

Campinas, 23 de fevereiro de 2018

Acknowledgements

I am thankful to

- My parents, Silvana and Edson, and my brother, Marcelo, for their encouragement and unconditional help.
- My advisor, Prof. Dr. Hélio Pedrini, for his tutorship, patience, and support.
- IC/UNICAMP for the infrastructure.
- CAPES for the financial support.

Resumo

O desenvolvimento de equipamentos multimídia permitiu um crescimento significativo na produção de vídeos por meio de câmeras, celulares e outros dispositivos móveis. No entanto, os vídeos capturados por esses dispositivos estão sujeitos a movimentos indesejados devido à vibração da câmera. Para superar esse problema, a estabilização digital visa remover o movimento indesejado dos vídeos pela aplicação de ferramentas computacionais, sem o uso de hardware específico, para melhorar a qualidade visual das cenas de forma a melhorar aspectos do vídeo segundo a percepção humana ou facilitar aplicações finais, como detecção e rastreamento de objetos. O processo de estabilização digital de vídeos bidimensional geralmente é dividido em três etapas principais: estimativa de movimento da câmera, remoção do movimento indesejado e geração do vídeo corrigido. Neste trabalho, investigamos e avaliamos métodos de estabilização digital de vídeos para corrigir vibrações e instabilidades que ocorrem durante o processo de aquisição. Na etapa de estimativa de movimento, desenvolvemos e analisamos um método consensual para combinar um conjunto de técnicas de características locais para estimativa do movimento global. Também apresentamos e testamos uma nova abordagem que identifica falhas na estimativa do movimento da câmera por meio de técnicas de otimização e calcula uma estimativa corrigida. Na etapa de remoção do movimento indesejável, propomos e avaliamos uma nova abordagem para estabilização de vídeos com base em um filtro Gaussiano adaptativo para suavizar a trajetória da câmera. Devido a incoerências existentes nas medidas de avaliação disponíveis na literatura em relação à percepção humana, duas representações são propostas para avaliar qualitativamente os métodos de estabilização de vídeos: a primeira baseia-se em ritmos visuais e representa o comportamento do movimento do vídeo, enquanto que a segunda é baseada na imagem da energia do movimento e representa a quantidade de movimento presente no vídeo. Experimentos foram realizados em três bases de dados. A primeira consiste em onze vídeos disponíveis na base de dados GaTech VideoStab e outros três vídeos coletados separadamente. A segunda, proposta por Liu et al., consiste em 139 vídeos divididos em diferentes categorias. Finalmente, propomos uma base de dados complementar às demais, composta a partir de quatro vídeos coletados separadamente. Trechos dos vídeos originais com presença de objetos em movimento e com fundo pouco representativo foram extraídos, gerando-se um total de oito vídeos. Resultados experimentais demonstraram a eficácia das representações visuais como medida qualitativa para avaliar a estabilidade dos vídeos, bem como o método de combinação de características locais. O método proposto baseado em otimização foi capaz de detectar e corrigir falhas de estimativa de movimento, obtendo resultados significativamente superiores em relação à não aplicação dessa correção. O filtro Gaussiano adaptativo permitiu gerar vídeos com equilíbrio adequado entre a taxa de estabilização e a quantidade de pixels preservados nos quadros dos vídeos. Os resultados alcançados como o nosso método de otimização nos vídeos da base de dados proposta foram superiores aos obtidos pelo método implementado no `YouTube`.

Abstract

The development of multimedia equipments has allowed a significant growth in the production of videos through professional and amateur cameras, smartphones and other mobile devices. However, videos captured by these devices are subject to unwanted vibrations due to camera shaking. To overcome such problem, digital stabilization aims to remove undesired motion from videos through software techniques, without the use of specific hardware, to enhance visual quality either with the intention of enhancing human perception or improving final applications, such as detection and tracking of objects. The two-dimensional digital video stabilization process is usually divided into three main steps: camera motion estimation, removal of unwanted motion, and generation of the corrected video. In this work, we investigate and evaluate digital video stabilization methods for correcting disturbances and instabilities that occur during the process of video acquisition. In the motion estimation step, we develop and analyzed a consensual method for combining a set of local feature techniques for global motion estimation. We also introduce and test a novel approach that identifies failures in the global motion estimation of the camera through optimization and computes a new estimate of the corrected motion. In the removal of unwanted motion step, we propose and evaluate a novel approach to video stabilization based on an adaptive Gaussian filter to smooth the camera path. Due to the incoherence of assessment measures available in the literature regarding human perception, two novel representations are proposed for qualitative evaluation of video stabilization methods: the first is based on the visual rhythms and represents the behavior of the video motion, whereas the second is based on the motion energy image and represents the amount of motion present in the video. Experiments are conducted on three video databases. The first consists of eleven videos available from the GaTech VideoStab database, and three other videos collected separately. The second, proposed by Liu et al., consists of 139 videos divided into different categories. Finally, we propose a database that is complementary to the others, composed from four videos collected separately, which are excerpts from the original videos with moving objects in the foreground and with little representative background extracted, resulting in eight final videos. Experimental results demonstrated the effectiveness of the visual representations as qualitative measure for evaluating video stability, as well as the combination method over individual local feature approaches. The proposed method based on optimization was able to detect and correct the motion estimation failures, achieving considerably superior results compared to when this correction is not applied. The adaptive Gaussian filter allowed to generate videos with adequate trade-off between stabilization rate and amount of frame pixels. The results reached with our optimization method for the videos of the proposed database were superior to those obtained with `YouTube`'s state-of-the-art method.

List of Figures

List of Tables

List of Abbreviations

2D	Two-Dimensional
3D	Three-Dimensional
AMM	Average Amount of Motion
AVC	Advanced Video Coding
BRIEF	Binary Robust Independent Elementary Features
BRISK	Binary Robust Invariant Scalable Keypoints
CenSurE	Center Surround Extremas
CLAHE	Contrast Limited Adaptive Histogram Equalization
CMOS	Complementary Metal-Oxide-Semiconductor
CV	Coefficient of Variation
dB	Decibel
FAST	Features from Accelerated Segment Test
FPS	Frames per Second
FREAK	Fast Retina Keypoints
GFTT	Good Features to Track
GLCM	Grey Level Co-occurrence Matrix
HSV	Hue-Saturation-Value
ITF	Interframe Transformation Fidelity
MEI	Motion Energy Image
MHI	Motion History Image
MSER	Maximally Stable Extremal Regions
ORB	Oriented FAST and Rotated BRIEF
PSNR	Peak Signal to Noise Ratio
RANSAC	Random Sample Consensus
SFM	Structure-from-Motion
SIFT	Scale-Invariant Feature Transform
SSIM	Structural Similarity
SURF	Speeded up Robust Features
VR	Visual Rhythm

Contents

Chapter 1

Introduction

Digital image and video processing refers to the set of techniques applied to the input data in order to produce enhanced images or videos, extract meaningful information, perform geometric transformations, detect regions of interest, measure object properties, among many other different operations [33, 85, 114]. A variety of knowledge domains can be benefitted from image and video processing techniques, for instance, medicine, biology, remote sensing, surveillance, robotics, photography, astronomy, microscopy.

The availability of new digital technologies and the reduction of equipment costs have facilitated the generation of large volumes of videos in high resolutions. Several devices have allowed the acquisition and editing of videos in various circumstances, such as digital cameras, smartphones and other mobile devices. However, the use of cameras under adverse conditions usually results in non-precise motion and occurrence of shaking, which may compromise the stability of the obtained videos.

1.1 Problem and Motivation

Video stabilization [4, 16, 27, 44, 55, 57, 62, 77, 80, 94] aims to correct camera motion oscillations that occur in the acquisition process, particularly when the cameras are mobile and handled in adverse conditions. Different categories of stabilization approaches have been developed to improve the quality of videos, which can be broadly classified as mechanical stabilization, optical stabilization and digital stabilization.

Unlike other categories, the digital stabilization is done in software, without the aid of any specific device. This type of stabilization is important because the costs in terms of hardware requirements are low. In addition, it is indispensable for videos already recorded. From the definitions presented previously, digital video stabilization can be thought as a video processing problem.

The digital stabilization problem can be categorized into online stabilization [36, 112] and offline stabilization [31, 59]. In the first category, the calculation of the transformation to be applied only on a frame employs information from previous frames. On the other hand, information from all video frames can be used in the second category and, therefore, the entire video must be available a priori. To limit the scope of this work, we do not deal with the development of strictly online techniques.

Due to the large amount of video that are captured, stored and transmitted, it is fundamental to investigate and develop efficient multimedia processing and analysis techniques. In the case of video stabilization, efficient methods are important to improve their quality according to human perception or to facilitate certain tasks, such as multimedia indexing and retrieval [23, 24, 41]. Applications include: video enhancement in general, robotics, unmanned aerial vehicle cameras, among others.

Methods found in the literature for digitally stabilizing videos are usually classified into two-dimensional (2D) or three-dimensional (3D) categories. We are particularly interested in investigating 2D methods, in which geometric transformations are employed to represent frame-to-frame motion and stabilize the videos. The reason for this interest is that even though 3D methods allow higher quality stabilization, 2D methods have a lower computational cost and are more robust to a variety of situations [65, 117], which causes them to be constantly preferred in practice [59]. The 2D digital video stabilization process is usually divided into three main steps: camera motion estimation, removal of unwanted motion, and generation of the corrected video.

In the first step, the motions made by the camera are estimated, constructing a path that corresponds to the one traveled by the camera, in an unstable way. The presence of moving objects, non-rigid motions and the parallax effect are some of the aspects that increase the challenges in the research and development of techniques for motion estimation.

The estimation step of the camera motion can be divided into two main approaches: (i) global estimation, when only rigid transformations are applied to the frames and (ii) local estimation, when non-rigid transformations are applied to correct frames with spatially distinct movements [69]. Due to the time available to carry out this work, we aim to improve only the global motion estimation. This step is typically performed through local features [36, 69]. Among the possible failure scenarios in the methods based on local features, we highlight:

- presence of moving objects in scenes where the background is not very representative;
- situations where objects in the foreground cover most of the scene;
- scenes that are not very representative in general.

From the path made by the camera, it is calculated what is unwanted motion. Because most videos have motions that are intentional, just the unwanted motion must be removed. In this step, there is usually a trade-off between the number of pixels held in the video frames and the quality of the stabilization. If we wrongly consider the intentional motion as unwanted, we may have a large loss of pixels in the frames of the stabilized video. After removing unwanted motion, the video frames are transformed according to the remaining motion and a smoother video is obtained.

Techniques and metrics for quality evaluation must be well established so that video stabilization approaches can be developed, refined and compared in a consistent manner. Therefore, ineffective evaluation measures may lead to the development of inadequate

techniques, compromising the advance of state-of-the-art video stabilization approaches. However, most of the quantitative techniques for the evaluation of video stabilization available in the literature appear to be inaccurate and, in some cases, incompatible with human visual perception. Moreover, the techniques used to evaluate and report the results subjectively are little explored.

1.2 Objectives

This work aims to investigate and evaluate digital video stabilization methods for correcting disturbances and instabilities that occur during the process of video capture. It also proposes novel methods for digital video stabilization and for qualitative evaluation of the video stabilization process. Experiments are performed on several video sequences. A comparative analysis of the results obtained with the proposed method and with other approaches of the literature are presented and discussed.

This general objective is correlated with the following specific objectives:

- carrying out a wide bibliographic survey, with the study of the main methods that constitute the state-of-the-art in video stabilization;
- evaluation of the performance of different local features methods in motion estimation;
- proposition and rating of a new technique for motion estimation;
- evaluation of the use of different filtering techniques for the removal of unwanted motion;
- presentation and evaluation of a new filter to removal of unwanted motion step;
- verification and documentation of the behavior of the stabilization evaluations present in the literature in different cases;
- proposition of methods for subjective evaluation of video stabilization;
- conducting comparative experiments with other available approaches.

1.3 Research Questions

By considering the presented objectives, the questions that we intend to answer through the development of this research work include:

- Can the use of different local features combined improve the motion estimation?
- Can the information from motion estimation of adjacent frames be used to detect and correct failures in the motion estimation?
- Can an adaptive filter generate videos with a higher amount of information maintaining the quality of the stabilization?

- Are the stabilization evaluation metrics available in the literature coherent with visual perception?
- Can the visual rhythm and the motion energy image characterize the stability of a video and be used for its evaluation?

1.4 Contributions

The main contributions of this work are:

- **A consensual approach to combining different methods of local features in motion estimation.** The motivation for this step is the hypothesis that different local features methods do not perform the same in different contexts. That is, a method that obtains good results in a particular situation may not obtain good results in others. This scenario can be determined by factors such as lighting, blurring or even content of the images. We have experimentally shown that the results of individual methods can be improved by combining different methods.
- **An approach that detects failures in the global motion estimation obtained through local features and proposes an optimization technique to calculate a new estimate of the corrected motion.** Experiments show that estimation of the optimization method is considerably superior when compared to the individual use of local features. In addition, it is evident that this stage is crucial to the success of the video stabilization process. The state-of-the-art stabilization method used in `YouTube` [36] is also used for comparison, which presents typical flaws when using local features to motion estimation, obtaining in these cases a worse result than the method proposed.
- **A new technique for removing unwanted motion based on the Gaussian filter to smooth the camera path.** Applying simple and direct low-pass filters to smooth the camera path is not enough in cases where movements are made intentionally. Thus, we propose the application of low-pass filters, more specifically the Gaussian filter, adaptively, in which the intensity of the filter is different in each part of the path. Experiments demonstrate the effectiveness of the method, which generates videos with proper stabilization rate while maintaining a reasonable amount of frame pixels.
- **New techniques for the qualitative evaluation of video stabilization through visual representations based on visual rhythms and motion energy image.** We believe that the information about the movement present in the video can characterize and be used to evaluate the quality of the stabilization. Thus, we propose a visualization scheme based on visual rhythms to represent the behavior of the motion present in a video. In addition, a visualization based on motion energy image is used to represent the amount of motion present in a video. Both proposed evaluation approaches are intended for human beings to assess the quality

of the stabilization. Experimental results demonstrate that the both visual representations are effective to evaluate the stability of camera motion by differentiating stable and unstable videos. Furthermore, the visual rhythm allows to determine how and when a given motion occurs. More complex types of motion, such as zoom and quick shifts, can also be identified.

1.5 Text Organization

This text is organized as follows. Relevant concepts and work related to the topic of digital video stabilization are presented and described in Chapter 2. The proposed approaches, tested databases, evaluation measures and computational resources used in the development of the project are described in Chapter 3. Experimental results are presented and discussed in Chapter 4. Some final remarks and directions for future work are included in Chapter 5. Appendix A presents some additional experiments explored in this work. Bibliographic references associated with the subject under investigation are presented at the end of this work.

Chapter 2

Background

This chapter is divided into two parts. The first presents some fundamental theoretical concepts for the understanding of this work. The second part describes some works related to the research theme.

2.1 Theoretical Concepts

This section presents some topics associated with the problem of digital video stabilization. The topics presented here are employed throughout the entire process of digital stabilization. Local features and optimization methods are used to calculate the geometric transformations between each pair of frames in order to estimate the camera motion. Then, this motion is filtered to generate a smoother motion. Finally, visualizations are used to carry out a qualitative evaluation of the stabilized video.

2.1.1 Local Features

Local features can be used to find a sparse set of corresponding locations in different images [105]. The advantage of using features to find such matches is due to the robustness of the recent methods that have been proposed over recent decades. Among them, we highlight: Harris and Stephens [40], Scale Invariant Feature Transform (SIFT) [71, 72], Maximally Stable Extremal Regions (MSER) [78], Speeded Up Robust Features (SURF) [7], Center Surround Extremas (CenSurE - also referred to as STAR) [1], Binary Robust Invariant Scalable Keypoints (BRISK) [60], Oriented FAST and Rotated BRIEF (ORB) [93], Fast Retina Keypoint (FREAK) [3].

The process of extracting local features can be divided into three stages: detection, description and matching [72, 105]. In the detection stage, a search is performed on the image in order to find locations that can be used to evaluate the correspondence in the other images. We can categorize local feature methods according to the type of structure they detect: corners, blobs and regions [106]. The techniques used to detect potential local features differ among methods, such as difference of the Gaussian filter [72] and determinant of the Hessian matrix [7]. Then, orientation of the detected features is determined through techniques such as wavelet transformation [7] and image gradient [72].

The region around the local feature is then described so that it is preferably invariant to scaling and rotation, among other variations. Such invariance is only possible due to the calculation of the scale and orientation of each local feature. As a result of the description, we have a vector describing the characteristics of each local feature, computed by considering the scale and orientation. The matching step determines the corresponding regions between the local feature sets of two images. Alternatively, the search between equivalent local features can be done by considering only a small neighborhood, a process called tracking. For more information about local features, references [105, 106] can be consulted.

The combination and complementarity of different methods for the detection of local features have been studied in recent works [9,30]. The main motivations for them are: (i) with little prior knowledge of the image contents, it is recommended to combine different types of detectors [106]; (ii) methods that detect a same type of structure still differ in their theoretical basis and can be used together [105]. Among recent works, it is possible to mention a study of the complementarity on eleven local features detectors based on measures of repeatability, spatial distribution, contribution [30], as well as an adaptive combination of local features for content-based image retrieval [9], where a regressor is trained to find the best pair of local features based on measures of spatial distribution, contribution, and local structures of a cluster.

Local Feature Matching

In order to find the equivalent local features between two images I and I', the correspondence between the previously described local features should be computed. A strategy would be: for each local feature in I, find the closest feature in I', considering some measure of distance between the characteristics of the local features.

A simple way to find local feature correspondences would be to determine a threshold and consider only matches that are below that threshold. Another technique would be brute force, so that $x_i \in I$ corresponds to $x'_j \in I'$ if and only if the characteristics of x_i are the closest to x'_j among all local features. In addition, we can use the cross-validation strategy, in which the characteristics of x'_j also need to be the closest to x_i.

2.1.2 Geometric Transformations

Given two sets of points P and P', which refer to two images I and I', such that $x_i \in P$ e $x'_i \in P'$ and x_i and x'_i are matching, a transformation can be defined by matrix M, of size 3×3 and arranged in homogeneous coordinates, so that $x'_i = M \cdot x_i$.

Among the different types of transformation matrices possible, we will define five: translational, Euclidean, similarity, affine and projective [29]. These types differ, in the context of this work, regarding the type of unwanted motion to be corrected and the minimum number of points required to calculate the estimation. The translational transformation corrects only horizontal and vertical shifts and is defined by the matrix shown in Equation 2.1, where T_x and T_y define the translations. To estimate this transformation,

only one point is required in each set.

$$M_T = \begin{bmatrix} 1 & 0 & T_x \\ 0 & 1 & T_y \\ 0 & 0 & 1 \end{bmatrix} \tag{2.1}$$

From at least two points in each set, the Euclidean transformation also corrects the rotational motion between two frames. Equation 2.2 shows the transformation matrix, where T_x and T_y define the translations and θ represents the angle of rotation.

$$M_E = \begin{bmatrix} \cos\theta & -\sin\theta & T_x \\ \sin\theta & \cos\theta & T_y \\ 0 & 0 & 1 \end{bmatrix} \tag{2.2}$$

The affine transformation also corrects scales, in which it requires at least three points for its estimation. This transformation is given in Equation 2.3a, where the unknowns a_{ij} represent a system of four equations that can be decomposed into the multiplication of Euclidean matrices 2.2 and scaling 2.3b. If $\lambda_1 = \lambda_2$ we call this similarity transformation.

$$M_A = \begin{bmatrix} a_{11} & a_{12} & T_x \\ a_{21} & a_{22} & T_y \\ 0 & 0 & 1 \end{bmatrix} \tag{2.3a}$$

$$\begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} \tag{2.3b}$$

$$\tag{2.3c}$$

Finally, in the projective transformation (homography), defined in Equation 2.4a, four points are needed to correct, in addition to all previous parameters, horizontal and vertical perspectives in addition to compression and shear. As in the related transformation, variables h_{ij} of Equation 2.4a are described as a system of nine equations, which can be decomposed as the multiplication of the affine matrix 2.3a by the perspective matrix 2.4c and compression and shear matrix 2.4b.

$$H_P = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \tag{2.4a}$$

$$\begin{bmatrix} \zeta & \alpha \\ 0 & \frac{1}{\zeta} \end{bmatrix} \tag{2.4b}$$

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ p_x & p_y & 1 \end{bmatrix} \tag{2.4c}$$

Robust Estimation

When estimating a transformation matrix using local features, problematic correspondences can affect the estimation, generating an estimate of low quality, which compromises the performance of the entire stabilization system. Two main problems can be addressed:

1. incorrect matches, where one or more local features of the first image may have been mismatched with a local feature of the other image. This problem is quite common, although it is simple to solve with an outlier detection method.

2. matches of moving objects, where local features may be present in moving objects and not in the static background. This problem is more difficult to solve than the previous one since the correspondences follow a certain pattern, and in extreme cases, when in greater number, they are determined as inliers. Either way, an outlier detection method solves most cases.

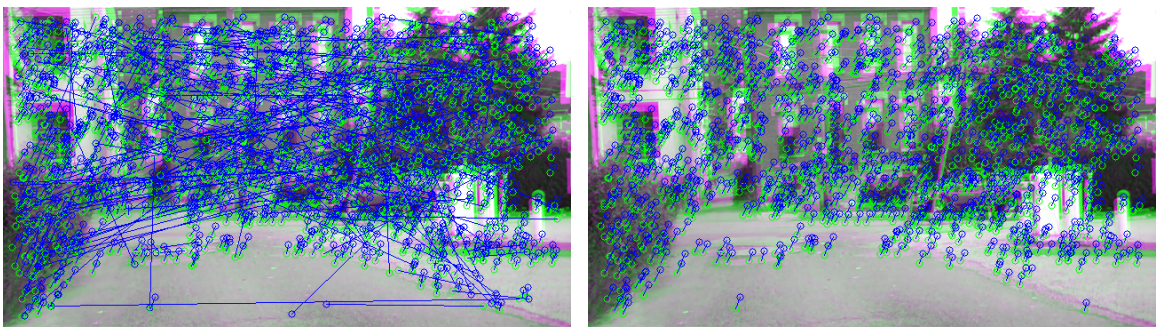
These problems are typically solved with the Random Sample Consensus (RANSAC) [28] method, an iterative technique for estimating mathematical model parameters from a data set containing outliers. At each iteration of the algorithm, the following tasks are performed:

1. a model is estimated from a random subset of the data.

2. each instance of the data is classified as inlier or outlier by means of the residues obtained when applying the previously estimated model.

3. the model is saved with the maximum number of inliers in all iterations.

After executing all iterations, the final model is generated from the set of maximum inliers. Figure 2.1 shows frames with correct and incorrect matches, before and after the execution of RANSAC.



(a) all matches (b) only correct matches

Figure 2.1: Frames with incorrect matches.

2.1.3 Optimization

Optimization can be defined as a search for the best solution, given a set of possible solutions. Typically, an optimization problem consists of minimizing or maximizing a given function. We can also define an optimization problem as [21]

$$\begin{aligned} \text{minimize} \quad & f(x) \\ \text{subject to} \quad & x \in \Omega \end{aligned} \tag{2.5}$$

where f is the function in which it is desired to minimize, called cost function or objective function. The vector x is a vector of dimension n of independent variables, called decision variables. The set Ω is a subset of $\mathbb{R}^n$, called feasible set or constraints set. When $\Omega = \mathbb{R}^n$, we determine that this is an unconstraint optimization problem. To maximize f, just minimize $-f$ [52].

It is possible to have multiple values for x that minimize the function f [21]. The minima of a function can be limited by an interval; in this case, we call them the local minima. Alternatively, they can still be considered throughout the domain of the function, which we call them the global minima. An optimization problem can be categorized depending on the type of the decision variable, objective function, and constraints [25]. Among them, we highlight:

- linear programming: the objective function and constraints are linear.
- nonlinear programming: the objective function and/or constraints are non-linear.
- integer programming: decision variables are integers.
- multi-objective optimization: the problem involves more than one objective.

Optimization has been widely explored in the context of image registration [5, 34, 42, 53,54,75,76,109,111]. Typically, initial parameters are calculated in order to approximate the alignment of the images, then an optimization guided by a measure of similarity leads to the initial solution to the final [34]. That is, the measure of similarity is the objective function, which must be maximized. Or minimized, in the case of a dissimilarity measure. In problems with images of the same modality, the cross-correlation coefficient can be used as a similarity measure [5], whereas mutual information can be used in problems with images of different modalities [75]. Several optimization methods have been proposed in literature, for instance, gradient descent method [53], evolutionary algorithms [111] and simulated annealing [51].

Powell's Method

In this work, we use Powell's method [87] in the motion estimation step. Powell's method is a local optimization method, which uses only function values without performing derivative calculations. Thus, it is categorized as a zero-order method [52]. Algorithm 1 describes the main steps of Powell's method with d directions [52, 87].

Algorithm 1 Powell's Method [52]

```
1:  procedure POWELL
2:      Initialize the initial guess x_0
3:      Initialize the direction vectors v as the unit vectors of each direction i ∈ d
4:      repeat
5:          for each direction v_i do
6:              x_{i-1} ← the minima along the line through x_{i-1} in the direction v_i
7:          v_{n+1} ← x_0 − x_n
8:          x_{n+1} ← the minima along the line through x_0 in the direction v_{d+1}
9:          for each direction v_i do
10:             v_i ← v_{i+1}
11:         x_0 ← x_{d+1}
12:     until |x_{d+1} − x_0| < ϵ
```

The search for minimum points, shown in lines 6 and 8 of the Algorithm 1, is typically performed by Brent's method [12]. A possible alternative would be the Golden-section search [49] method. For further details on optimization concepts and Powell's method, the references cited in this subsection can be consulted.

2.1.4 Filtering

In this section, we present two filtering techniques that had their use investigated in the removal unwanted motion step. These are: Kalman filter and Gaussian filter.

Kalman Filter

The Kalman filter [48] is an instantaneous "state" estimator of a linear system disturbed by a Gaussian white noise [35]. This is one of the most commonly used methods for tracking and estimation in linear systems due to their simplicity, optimality and robustness [47]. Subsequently, the filter was extended to nonlinear systems [47]. The filter performs the estimation based only on the value of the previous state and the current measurement. Normally, it is described in two phases: prediction and update, described in Equations 2.6 and 2.7, respectively.

In the prediction phase, an a priori estimate $\widehat{x}_{t|t-1}$ is generated, which considers only the previous value $\widehat{x}_{t-1|t-1}$. In Equations 2.6, F_t is the state transition matrix, u_t defines the vector of input controls, B_t represents the input control matrix, whereas Q_t is the covariance matrix associated with the input noise control, $P_{t|t-1}$ refers to the a priori covariance matrix and $P_{t-1|t-1}$ a covariance matrix of the previous iteration.

$$\begin{aligned} \widehat{x}_{t|t-1} &= F_t \widehat{x}_{t-1|t-1} + B_t u_t \\ P_{t|t-1} &= F_t P_{t-1|t-1} F_t^T + Q_t \end{aligned} \tag{2.6}$$

In the update phase, $\widehat{x}_{t|t-1}$ is combined with the current observation z_t, thus obtaining an estimate $\widehat{x}_{t|t}$. In Equations 2.7, K_t is the Kalman gain, H_t is the transformation matrix that maps the parameters to the observation domain and R_t is the uncertainty

matrix associated with the noise of observation.

$$\begin{aligned} \widehat{x}_{t|t} &= \widehat{x}_{t|t-1} + K_t\left(z_t - H_t\widehat{x}_{t|t-1}\right) \\ P_{t|t} &= P_{t|t-1} - K_t H_t P_{t|t-1} \\ K_t &= P_{t|t-1}H_t^T\left(H_t P_{t|t-1}H_t^T + R_t\right)^{-1} \end{aligned} \tag{2.7}$$

Gaussian Filter

The Gaussian filter is a linear low-pass filter and therefore attenuates the high frequencies of a signal. The Gaussian filter modifies the input through a convolution considering a Gaussian function in a window of size W_g. Thus, the Gaussian function is used as impulse response in the Gaussian filter and can be defined as

$$G(x) = ae^{-\frac{(x-\mu)^2}{2\sigma^2}} \tag{2.8}$$

where a is a constant considered as 1 so that $G(x)$ has values between 0 and 1. The constant μ is the expected value, considered as 0, whereas σ^2 represents the variance.

The Gaussian filter is commonly used in the field of digital communication. In two dimensions, the Gaussian filter is used mainly in the smoothing of images [33, 85].

2.1.5 Video Visualization

Video visualization is concerned with the creation of a new visual representation, obtained from an input video, capable of indicating its characteristics and important events [10]. Video visualization techniques can generate different types of output data, such as another video, a collection of images or a single image. Borgo et al. [10] reported a review of several video visualization techniques proposed over the last years.

In this work, we use as base two concepts of the literature to create representations that allow us to evaluate methods of stabilization of videos: Visual Rhythms and Motion Energy Image.

Visual Rhythms

Visual rhythm [22] (VR) is a summary of the video temporal information in a single image. This is done by concatenating information from each frame of the video. Visual rhythms have been generally applied in the context of video identification and classification, for instance, location of video subtitles, detection of shot boundaries, detection of face spoofing, among others [86, 97].

In this work, two different paths for constructing the visual rhythms are considered for each video: horizontal and vertical. These rhythms differ according to the information that is extracted from the video frames. The vertical rhythm extracts the information from the columns of each frame, whereas the horizontal rhythm is constructed from the rows of each frame.

Typically, a single column or row (or a small set of them) of each frame is used to construct the visual rhythm. Figure 2.2 represents the construction of a horizontal

visual rhythm typically presented in the literature. In this work, however, we adopted the average of the columns for the vertical rhythm and the average of the rows for the horizontal rhythm.

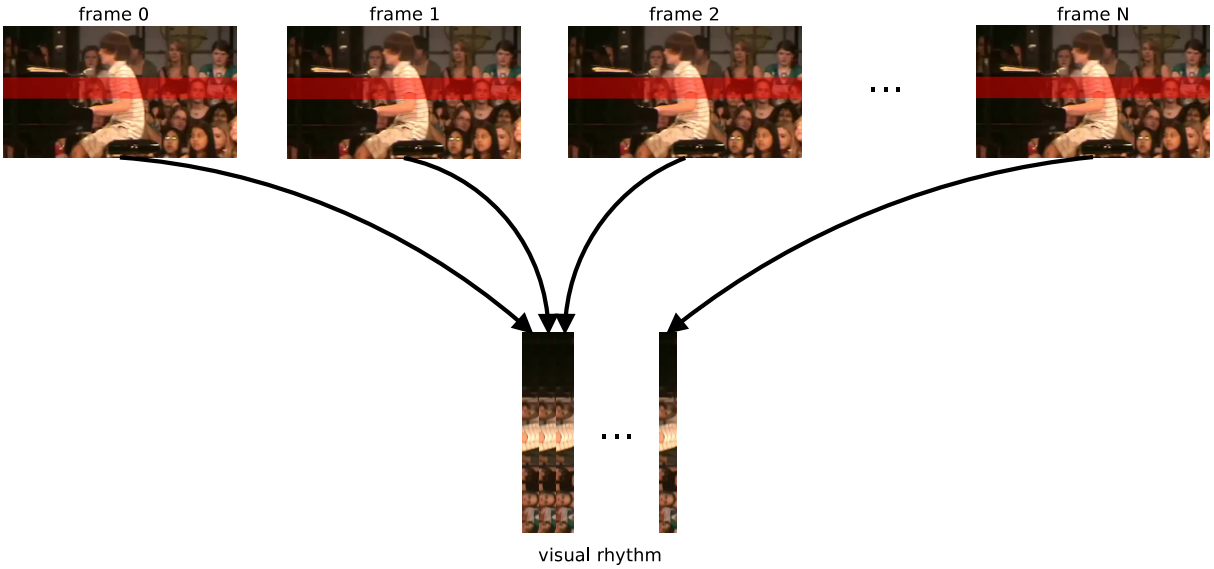


Figure 2.2: Construction of a horizontal visual rhythm - traditional mode.

Motion Energy Image

Motion energy image (MEI) is a binary image that represents the occurrence of video motion in a given region. This occurrence is determined by the difference in the gray level intensities of the video frames. The white pixels denote the occurrence of motion, whereas the black pixels denote the absence of motion [2].

In conjunction with the motion history image (MHI), MEI is generally used in the context of recognizing human actions in videos [2]. In this work, we consider the average of the motion energy images obtained throughout the video to assess the amount of motion and to characterize its stability. Figure 2.3 shows examples of motion energy images.

2.2 Related Work

In this section, we briefly describes some relevant work related to 2D and 3D video stabilization. Then, we present some objective and subjective evaluation measures used in previous approaches.

2.2.1 Digital Video Stabilization

Video stabilization is usually categorized into three main classes: mechanical stabilization, optical stabilization, and digital stabilization.

Mechanical stabilization typically uses sensors to detect camera shifts and compensate for unwanted motion. A common way is to use gyroscopes to detect motion and send signals to motors connected to small wheels so that the camera can move in the opposite

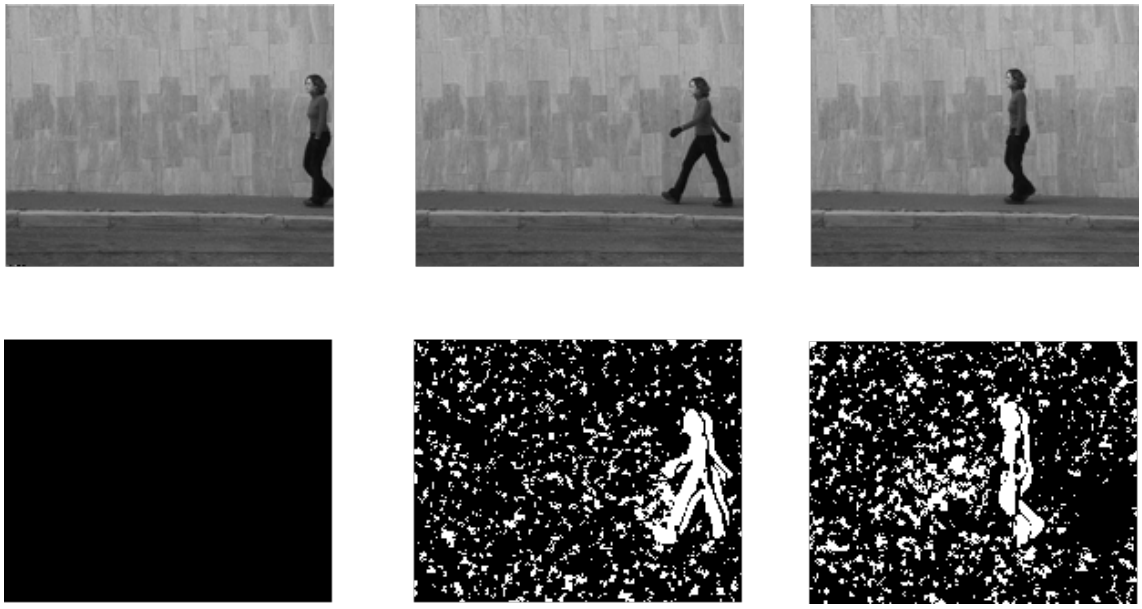


Figure 2.3: Example of motion energy images for a frame sequence. Original frames on top, and their MEI on the bottom.

direction of motion. The camera is usually positioned on a tripod. Despite the efficiency usually obtained with this type of system, there are disadvantages in relation to the resources required, such as device weight and battery consumption.

Optical stabilization [14] is widely used in photographic cameras and consists of a mechanism to compensate for the angular and translational motion of the cameras, stabilizing the image before it is recorded on the sensor. A mechanism for optical stabilization introduces a gyroscope to measure velocity differences at distinct instants in order to distinguish between normal and unwanted motion. Other systems employ a set of lenses and sensors to detect angle and speed of motion for video stabilization.

Digital stabilization of videos is implemented without the use of special devices. In general, unwanted camera motion is estimated, and then compensated by applying transformations to the frames. These techniques are typically slower when compared to optical techniques; nevertheless, they can achieve adequate results in terms of quality and speed, depending on the algorithms used.

Digital video stabilization methods are commonly categorized into two-dimensional (2D) and three-dimensional (3D). 2D techniques estimate camera motion from two consecutive frames, and apply two-dimensional transformations to stabilize the video. On the other hand, 3D techniques attempt to reconstruct the camera path from three-dimensional transformations, such as scaling, translation and rotation.

2D Methods

Approaches that use 2D transformations focus on contributing to specific steps in their stabilization process [59]. By considering the estimation of camera motion, 2D methods can be further subdivided into two categories [18]: (i) intensity-based approaches [15, 89], which directly use the texture of the images as motion vector and (ii) local feature-based approaches [6, 95], which locate a set of corresponding local features in adjacent frames.

Since local feature-based approaches have a lower computational cost, they are most commonly used [79].

Techniques such as the extraction of regions of interest can be used in this step, in order to avoid cutting certain objects or regions that are supposed to be important to the observer [19]. The use of a depth sensor was proposed in order to handle with scenes that have depth variations [68].

Recently, the combination of local feature detection methods using the maximally stable extremal regions (MSER) [78] and features from accelerated segment test (FAST) [92] was employed for frame-to-frame motion estimation by performing keypoint detection with FAST only within regions detected by MSER, which demonstrated to be very effective [118]. Line segments and keypoints were combined to estimate a warping-based motion model estimation [61].

In the removal of unwanted motion step, approaches have employed the Kalman filter [26, 63], regularization [15], optimization [36, 69], among other methods. Such mechanisms aim to remove instability from camera motion, normally located in the high frequency of the camera path [59]. Other works focus on improving the quality of the videos, often lost in the stabilization process. The most commonly used techniques include: inpainting to fill missing frame parts [19,31,79], deconvolution to improve the video focus [19, 79], and weighting of stabilization metrics and video quality aspects [31, 59].

Recent improvements in 2D methods have made them comparable to 3D methods in terms of quality. In the following subsections, we will present two of these methods in more detail, which are considered state-of-the-art in 2D digital stabilization.

Auto-Directed Video Stabilization with Robust L1 Optimal Camera Paths

In the work developed by Grundmann et al. [36], the motion estimation step was performed using pyramidal Lucas-Kanade [73]. A local outlier rejection was employed by discretizing features into a grid of 50×50 pixels, applying RANSAC within each grid cell, and discarding matches with distance was greater than two pixels.

A L1-norm optimization was proposed in order to generate a camera path that follows cinematographic rules. The algorithm is based on a linear programming to minimize the first, second and third derivatives of the resulting camera path. Additional constraints are incorporated on the path of the camera.

To mimic professional footage, the paths are optimized to be composed of three path segments:

- constant path, where $DP(t) = 0$;
- constant velocity path, where $D^2P(t) = 0$;
- constant acceleration path, where $D^3P(t) = 0$;

The goal is to find a camera path minimizing these objectives while satisfying some constraints. Three constraints are explored: the cropping window transformed by the new path should be contained within the frame rectangle transformed by the old path; the new path should preserve the original intent of the original path; salient points should

be included within all or a specific part of the cropping window transformed by the new path.

The optimization can be expressed as

$$O(P) = w_1|D(P)|_1 + w_2|D^2(P)|_1 + w_3|D^3(P)|_1 \tag{2.9}$$

where $D(P)$, $D^2(P)$, and $D^3(P)$ are the first, second and third derivatives of the smoothed path, respectively, whereas w_1, w_2, and w_3 are the weights assigned to each of these terms.

Each weight w_1, w_2, and w_3 can be tuned for a better stabilization. The optimization can be done using forward differencing method for the three terms. Without constraints, the optimal path is constant.

Motion models with higher degree of freedom than similarities are needed for complete stabilization. However, even though they can achieve better performance for a few frames, they start to suffer from excessive skew and perspective when applied to more frames. Thus, in order to deal with wobble and rolling shutter, a hybrid approach is proposed, applying the rigid camera path (similarity matrix) only for every k key-frames. For intermediate frames, the homographies are used to account for misalignments.

The difference between two optimal and rigid adjacent camera transforms is decomposed into the know estimated similarity part and a smooth residual motion. The low-dimensional similarity is replaced with the higher-dimensional homography. For each intermediate frame, these replacements are concatenated starting from its previous and next key-frames. This results in two sample location per pixel. Thus, a linear blending between these two locations is applied in order to determine a per-pixel warp for the frame. This is called a wobble suppression method.

Bundled Camera Paths for Video Stabilization

In the work developed by Liu [69], the global motion estimation was performed using the SURF method. A global outlier rejection was applied by RANSAC with greater threshold, followed by a local outlier rejection in 4×4 sub-images with a lower threshold.

A mesh-based model, in which multiple paths are calculated at different locations of the video, proved to be efficient in dealing with parallax and removing rolling shutter effects while stabilizing the video without the use of 3D methods.

At each frame, a uniform grid mesh is defined. At the i-th grid cell, the warping from frame t to frame $t + 1$ introduces a homography $F_i(t)$, which can be determined from the motion of the four vertices. This mesh is defined between global homography and per-pixel optical flow. However, estimating this model is very risky, because we may not have sufficient features in every cell. Thus, a shape-preserving constraint is proposed. The motion is estimated by minimizing two energy terms: data term for matching features, and a shape-preserving term.

Suppose $\{p, q\}$ is the p-th matched feature pair from frame t to frame $t + 1$. The feature p can be represented by a 2D bilinear interpolation of the four vertices V_p, where w_p are interpolation weights that sum to 1. Is expected that the corresponding feature q can be represented by the same weights of the warped grid vertices $\hat{V}_q$. The data term E_d

is defined as

$$E_d(\hat{V}) = \sum_p \|V_q w_p - q\|^2 \tag{2.10}$$

where V_q is the set of vertices after the transformation (to be found), vector w_p corresponds to the weights found by the interpolation, whereas p and q are matched local features in the frames t and $t+1$.

The shape-preserving term E_s requires the triangle of neighboring vertices v, v_0, v_1 to follow a similarity transformation.

$$E_s(\hat{V}) = \sum_{\hat{v}} \|\hat{v} - \hat{v}_1 - sR_{90}(\hat{v}_0 - \hat{v}_1)\|^2, R_{90} = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \tag{2.11}$$

where $\hat{v}$, $\hat{v}_0$ and $\hat{v}_1$ are the neighbor vertices after the transformation, and $s = \frac{\|v - v_1\|}{\|v_0 - v_1\|}$.

The data term is the quadratic error between the bilinear interpolation and the feature q. The shape-preserving is the quadratic error between the estimated vector and the vector that would make the triangle be a rectangle triangle. The final energy is the linear combination of the two terms with an alpha factor to control the amount of regularization. Since the final energy is quadratic, the warped mesh V can be solved by a sparse linear system solver.

The alpha factor was adaptively set per frame, based on two errors: (i) fitting error, which is the average residual of the feature matching under the estimated homographies; (ii) smoothness error, which measures the similarity between neighboring local cells homographies. The final error is the sum of this two factors. Alpha is discretized into 10 values between 0.3 and 3. Then, the model is estimated using every discretized value and the model with minimum error.

After having a new mesh, each local homography in the grid cell can be estimated by solving a linear equation involving the four vertices before and after the warping. To facilitate the warping estimation, a global homography was used to let matching features closer.

The camera path smoothing applied consider multiple competing factors: remove jitters, avoid excessive cropping, and minimize various geometrical distortions. Given an original path $C = C(t)$, we seek an optimized path $P = P(t)$ by minimizing the following function:

$$O(\{P(t)\}) = \sum_t \left(\|P(t) - C(t)\|^2 + \lambda_t \sum_{r \in \Omega_t} \omega_{t,r}(C) \cdot \|P(t) - P(r)\|^2 \right) \tag{2.12}$$

where Ω_t are the neighborhood at frame t; the data term $||P(t) - C(t)||^2$ enforcing the new camera path to be close to the original one; the smoothness term $||P(t) - P(r)||^2$ stabilizing the path; the weight $w_t, r(C)$ to preserve motion discontinuities under fast panning/rotation or scene transition; and parameter λ_t to balance the above two terms.

The adaptive weight is defined as:

$$\omega_{t,r} = G_t(\|r - t\|) \cdot G_m(\|C(r) - C(t)\|) \tag{2.13}$$

where G_t gives larger weight to nearby frames and G_m measures the changes of two camera poses. If the bundled paths are optimized independently, neighboring paths could be less consistent. Hence, the paths are optimized together minimizing the following function.

$$\sum_i O(\{P_i(t)\}) + \sum_t \sum_{j \in N(i)} \|P_i(t) - P_j(t)\|^2 \tag{2.14}$$

where $N(i)$ includes eight neighbors of the grid cell i.

Both optimizations were be solved by a Jacobi-based iteration. The term G_m is evaluated at individual cells. The λ_t value is determined from the global path, which are empirically set to 5 and check the cropping ration and distortion for every frame. For any frame that does not satisfy the threshold, its parameter is decreased by a step $(1/\lambda_t)$.

3D Methods

In turn, 3D methods usually employ structure-from-motion (SFM) techniques [64, 65]. Typically, the stabilization quality of the 3D methods is superior compared to 2D methods, however, with a higher computational cost [65,117] and with less robustness for the various possible situations present in a video [59].

Although 3D methods can generate good results in static scenes using image-based rendering techniques [8, 13], they usually do not handle dynamic scenes correctly, causing motion blur [64]. Thus, the concept of content preservation was introduced, restricting each output frame to be generated from a single input frame [64]. Other approaches address this problem through a geometric approximation by abdicating to be robust with respect to the parallax [116]. Other difficulties found in 3D methods appear in amateur videos, such as lack of parallax, zoom, use of complementary metal-oxide-semiconductor (CMOS) sensors, among others [65].

Although not common, 3D methods can fill missing parts of a frame by using information from several other frames [8]. More recently, 2D and 3D methods have been extended to deal with stereoscopic videos [37, 66]. Hybrid approaches have emerged to obtain the efficiency and robustness of 2D methods in addition to the high quality of 3D methods. Some of them are based on concepts such as trajectories subspace [65] and epipolar transfer [32].

Other Approaches

Several recent approaches to digital video stabilization have been proposed in the literature. Some of them are briefly described as follows.

The use of depth sensors has been proposed to deal with scenes with depth variations [68]. One method assigns spatio-temporal weights to local patches, which emphasizes regions with motion similar to that of the camera [50]. A multi-camera panoramic video stabilization technique considers independent oscillations in each camera [38]. A hybrid algorithm has been proposed for 360 degree video stabilization using a deformable rotational motion model [56].

More recently, a real-time smoothing method based on a linear estimation of the

Kalman filter with constant velocity has been proposed. The projection, estimated to guarantee certain restrictions, is combined with a model that smooths the camera path probabilistically [43]. In addition, a method has been proposed to stabilize encoded videos constructing the motion made by the camera through motion vectors of the coding itself [67].

2.2.2 Video Stabilization Evaluation

In the context of image and video processing, results can be typically analyzed through two categories: (i) objective evaluation, when obtained through functions applied between two images [33] or video frames, and (ii) subjective evaluation, when the analysis is performed by human observers. In both cases, a desired goal is to assess stabilization based on criteria in agreement with the perception of the human visual system.

Objective Evaluation

Criteria for measuring the amount and nature of the camera displacement have been proposed to evaluate the quality of video stabilization in an objective manner [82]. Unintentional motion is decomposed into divergence and jitter through low-pass and high-pass filters, respectively. The amount of jitter from the stabilized and original video is compared. The divergence is also verified, which indicates the amount of expected displacement. For an overall assessment, the blurring caused by the stabilization process is considered.

Most of the video stabilization works found in the literature have adopted the Inter-frame Transformation Fidelity (ITF) [6, 18, 20, 89, 96], which can be expressed as

$$\text{ITF} = \frac{1}{N-1}\sum_{t=1}^{N-1}\text{PSNR}(t) \tag{2.15}$$

where N is the number of frames in the videos. Typically, the stabilized sequence has a higher ITF value than the original sequence. The Peak Signal to Noise Ratio (PSNR) is used to evaluate the overall difference between two frames of the video, expressed as

$$\text{PSNR}(f_t, f_{t+1}) = 10\log_{10}\frac{WHL_{\max}^2}{\sum_{x=1}^{W}\sum_{y=1}^{H}[f_t(x,y) - f_{t+1}(x,y)]^2} \tag{2.16}$$

where f_t and f_{t+1} are two consecutive frames of the video, W and H are the width and height of each frame, respectively, L_{max} is the maximum value intensity of the image. The PSNR metric is expressed in decibel (dB), a unit originally defined to measure sound intensity on a logarithmic scale. Typical PSNR values range from 20 to 40. The PSNR value should increase from the initial video sequence to the stabilized sequence, since frames after transformation will tend to be more similar.

More recent works have considered the Structural Similarity (SSIM) [110] as an alternative to PSNR [18]. The SSIM was originally developed to measure qualitative differences

between two images. Calculated in several image windows, the SSIM between the window x and window y of the frames f_i and f_{i+1} can be expressed as

$$\text{SSIM}(x,y)=\frac{(2\mu_x\mu_y+(k_1+L_{\max})^2)(2\sigma_{xy}+(k_2+L_{\max})^2)}{(\mu_x^2+\mu_y^2+(k_1+L_{\max})^2)(\sigma_x^2+\sigma_y^2+(k_2+L_{\max})^2)} \tag{2.17}$$

where μ_x and μ_y are the means of x and y, σ_x^2 and σ_y^2 are the variances of x and y. Variable σ_{xy} is the covariance between x and y, whereas k_1 and k_2 are constant.

For each pixel (x, y) of the input frames f_t and f_{t+1}, we will have an SSIM value in the range $[-1, 1]$, where the higher its value, the greater their similarity. These values will compose another image, denoted S_t. The mean of these values is typically used as a measure of similarity.

We will refer to the value of SSIM and PSNR as their respective abbreviation. ITF is used as in Equation 2.15, whereas ITF_{SSIM} refers to ITF calculation considering SSIM in place of PSNR.

Liu et al. [69] employed the amount of energy present in the low-frequency portion of the 2D motion estimated as a stability metric. The rate of frame cropping and distortion are used to assess the stabilization process more generally.

Synthesizing unstable videos from stable videos has been proposed for the evaluation of video stabilization [90] in order to provide the ground-truth of the stable videos. The methods are evaluated according to two aspects: (i) the distance between the stabilized frame and the reference frame and (ii) the average of the SSIM between each pair of consecutive frames.

Due to the weaknesses of ITF in motion videos, an evaluation method based on the variation of the intersection of angles between the global motion vectors, calculated from the scale-invariant feature transform (SIFT) keypoints [71], was proposed to evaluate the video stabilization process [17]. In fixed-camera videos, the ITF is considered, however, only for overlapping the frame background, instead of the entire frame.

Subjective Evaluation

Several methods found in the literature briefly describe and analyze review the path made by the camera and the path of the stabilized video [15, 19, 63, 79, 91]. These paths are usually related to the different factors that compose the estimated 2D motion. For instance, the works present the camera path for horizontal and vertical translations and rotations. Figure 3.9 (in the next chapter) shows examples of path for horizontal translation estimated from the original (green) and smoothed (blue) path.

From the path, it is possible to identify when a motion occurs and its intensity in the original video, as well as such motion after its smoothing. This type of visualization can be very useful to analyze the behavior of the motion smoothing step used in a certain method. However, its result depends on the technique used in motion estimation, so that the path does not reliably represent the video motion. Thus, the path visualization may not be a good alternative to the evaluation of the stabilization quality, as well as not an adequate visualization for videos with spatially distinct motion.

Some works in the literature deal with frame sequences usually superimposed by horizontal and vertical lines [15, 18, 20, 63, 79, 95, 113]. Thus, it is possible to check the

alignment of a small set of consecutive frames. Figure 2.4 illustrates an example of such type of visualization, where objects intercepted by lines are more aligned in the stabilized video.

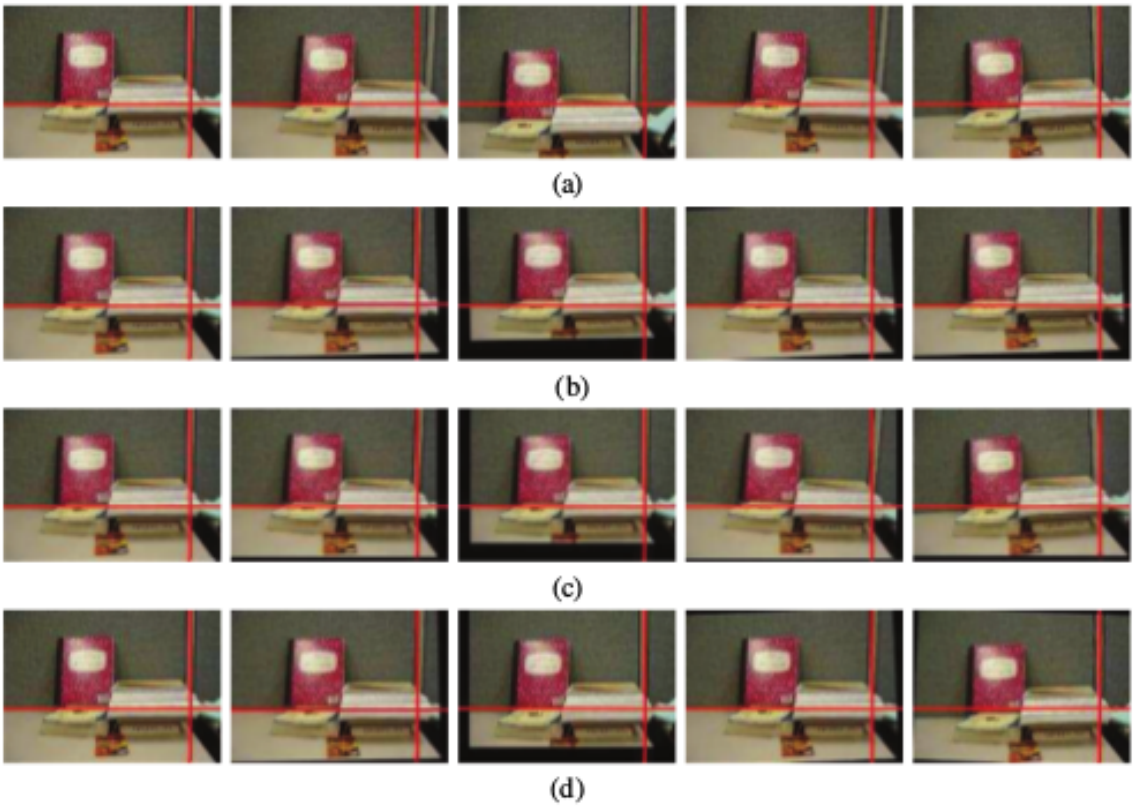
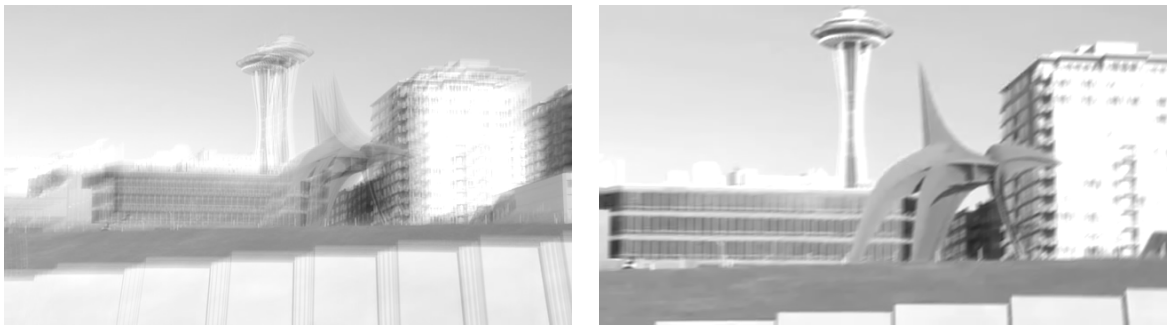


Figure 2.4: Sequence of video frames. (a) original video; (b-d) different versions of the stable video. Extracted from [113].

From the sequence of frames, the displacement of each frame is noticeable, in addition to the amount of pixels lost due to the transformation applied to each frame. However, this technique becomes impractical when a large number of frames is considered, compromising the analysis of the entire video.

Furthermore, there are approaches that summarize a video in a single image calculated through the average gray levels of the frames [46, 118], as shown in Figure 2.5. Better-defined images are expected for more stable videos. From this representation, it is possible to check if is exists motion in the video, but it is difficult to determine the amount and nature of the motion.



(a) (b)

Figure 2.5: The average gray levels for the first ten frames. (a) original video; (b) stabilized video.

Chapter 3

Video Stabilization Methods

In this chapter, we present the proposed methods for digital stabilization of videos. Figure 3.1 illustrates the main stages of the general stabilization process, where the most relevant contributions of this work are indicated in each step. As input to the process, we have an unstable video that will go through all the stages of the method. As output, we have the generated stabilized version of the video as well as quantitative and qualitative outcomes derived from the stabilization evaluation.

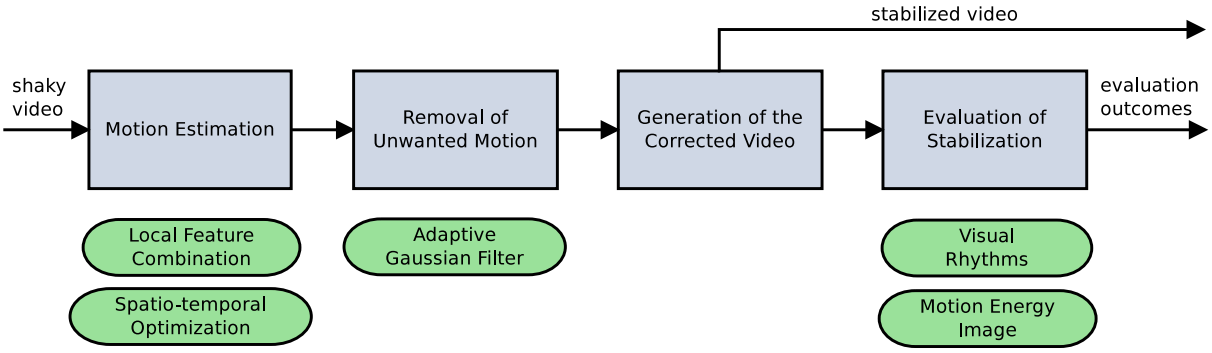


Figure 3.1: Main steps of the proposed digital video stabilization method.

Initially, the camera motion is estimated. In this step, we propose two approaches: (i) the combination of local features [100], which fuses different local feature methods in a consensual way in order to estimate the global camera motion, and (ii) an optimization strategy based on the structural similarity index [99] that considers spatial and temporal information to correct cases of failure in the global motion estimation.

A smoother motion is then calculated from the camera motion. For this, we propose an adaptive Gaussian filter [101], in which the intensity of the filter is changed adaptively along the video according to the behavior of the estimated motion. Subsequently, the frames of the video are transformed in order to follow the smoothed motion. In addition, frame borders are cropped to keep only the useful information.

Finally, the final video is submitted to a qualitative assessment. In the evaluation stage, we propose two approaches: (i) evaluation of the behavior of motion present in the video based on visual rhythms [103], and (ii) evaluation of the amount of motion present in the video based the motion energy image [102].

3.1 Motion Estimation

This section describes the developed methods associated with motion estimation. Initially, we present the motion estimation approach with only a single local feature. Next, we show the method for estimating motion with combined local features. Finally, we present the method for correction of the estimation based on optimization.

The process of motion estimation with a single local feature starts with the detection and description of local features in the video frames. After extracting the local features between two adjacent frames, their correspondence is performed using the brute-force method with cross-checking [104], where the Euclidean distance (or Hamming distance when the descriptor is binary) between the feature vectors for each pair of local features $x_i \in f_t$ and $x'_j \in f_{t+1}$ is calculated for two adjacent frames f_t and f_{t+1}. Thus, x_i corresponds to x'_j if and only if x_i is the closest local feature to x'_j, and x'_j the closest to x_i.

Figures 3.2 and 3.3 show the detection of local features in a frame and the correspondence between the local features of two adjacent overlapped frames, respectively.

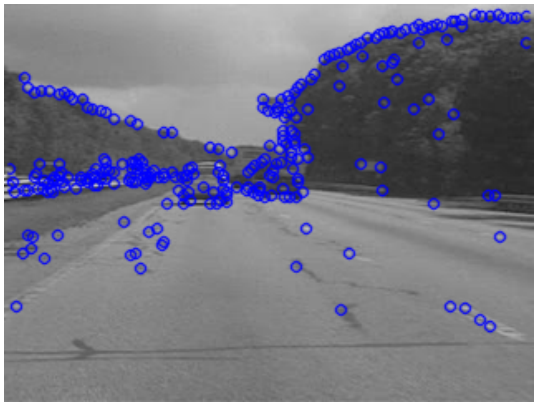


Figure 3.2: Detection of local features between adjacent frames.

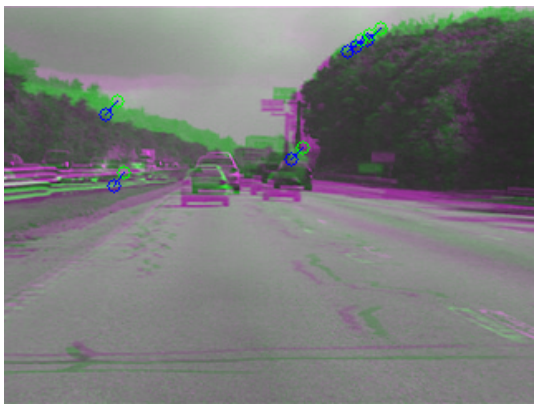


Figure 3.3: Matching of local features between adjacent frames.

After determining the matches between local features, it is necessary to estimate the transformation matrix that maps the set of local features in a frame f_t to the set of local features in a frame f_{t+1}. We consider the similarity matrix, therefore, the parameters

of the matrix transformation take into account camera shifts (translation), distortion (scaling) and undesirable motion (rotation) for the construction of a stabilization model. We decide to choose the similarity matrix because matrices with a greater degree of freedom can generate excessive distortions, as discussed by Grundmann et al. [36] and presented in Chapter 2. Such problems were confirmed in preliminary experiments.

In the process of digital video stabilization, oscillations of the camera that occurred at the time of recording must be compensated. The transformation matrix should take into account only the correspondences that are, in fact, between two equivalent local features. In addition, it should not consider the movement of objects present in the scene.

The Random Sample Consensus (RANSAC) method [28] is applied to estimate a transformation matrix that considers only inliers in order to disregard the incorrect correspondences and those that describe the movement of objects. In the application of this method, the value of the residual threshold parameter, which determines the maximum error for a match to be considered as inlier, is calculated for each pair of frames. Algorithm 2 presents the calculation to determine the final similarity matrix.

Algorithm 2 Similarity Matrix Computation

1: **procedure** FINALMATRIX
2: Generate the similarity matrix M by considering all matches.
3: Let MSE(M) be the mean square error of matrix M.
4: Apply the RANSAC considering the MSE(M) as residual threshold value.
5: Generate the similarity matrix M' considering only the inliers obtained previously.
6: Let MSE(M') be the mean square error of matrix M'.
7: Apply the RANSAC considering the MSE(M') as residual threshold value.
8: Generate the similarity matrix M_{final} considering only the inliers obtained for the second execution of RANSAC.

In cases of pairs of frames with spatially variant motion, the correct matches also tend to have certain variation. Thus, the residual threshold is calculated so that its value is low enough to eliminate undesired matches and high enough such that the correct matches are maintained.

In the following subsections, we present the proposed method that combines local features for motion estimation. Next, we describe the optimization method to correct problems in the global estimation.

3.1.1 Motion Estimation with Local Combined Features

The diagram shown in Figure 3.4 presents the main components of the motion estimation with combined local features.

Initially, we consider a set of methods M_f to be combined. For each method $m \in M_f$, its respective detection and description are applied for each pair of frames. Then, the local features are matched considering the local features of each method m separately. The correspondence is performed using the brute-force method with cross-checking [104]. Given the sets of local features that belong to the adjacent frames f_t and f_{t+1}, such that $x_i^m \in f_t$ and $y_j^m \in f_{t+1}$, we calculate the Euclidean distance (or Hamming distance when

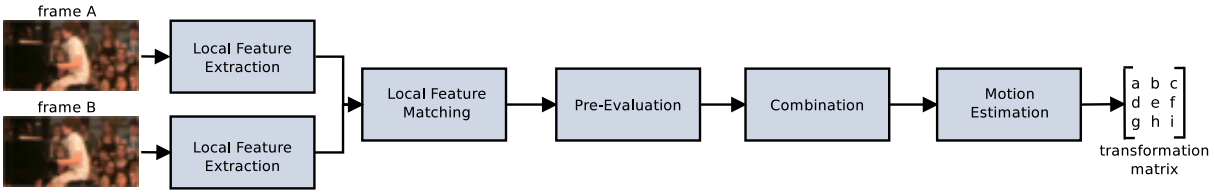


Figure 3.4: Main components of method for combining local features.

the descriptor is binary) between the feature vectors for each pair of local features x_i^m and y_j^m. Thus, x_i^m corresponds to y_j^m if and only if x_i^m is the closest local feature to y_j^m, and y_j^m the closest to x_i^m.

Before combining the matches of each method, a pre-evaluation is necessary to discard those that would perform poorly or could contribute negatively to the final combination. After having computed the correspondences between the local features of each method in the previous step, a transformation matrix for each method m is calculated taking into account all its local features. The mean squared error e^m for each transformation matrix is then computed. We conjecture that a transformation matrix with a very high mean square error indicates a large number of outliers. Thus, we consider the harmonic mean of the quadratic errors e_{hmean}, and all methods that do not satisfy Equation 3.1 are disregarded.

$$e^m \leq \sigma e_{hmean} \tag{3.1}$$

where σ is a constant. In this work, σ was assigned as 1.5.

After the pre-evaluation of the methods, a consensual combination is applied in the remaining methods, such that only local features that are consistent with their transformation are considered as final local features. This combination can be seen as a method based on RANSAC [28], which makes use of different sources of information rather than considering random samples from the same source. The combination step is shown in the diagram in Figure 3.5.

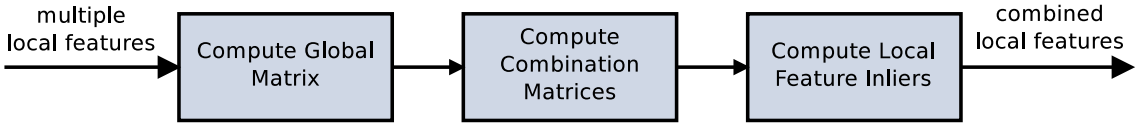


Figure 3.5: Main stages of the combination of local features.

Let the sets of local features be $F_{all} = F_0 \cup F_1 ... \cup ... Fi$, and $F'_{all} = F'_0 \cup F'_1 ... \cup ... F'_i$, where F_i are the local features of the frame f_t, F'_i the features f_{t+1}, and i the methods that passed in the previous step. The transformation matrix of F_{all} to F'_{all}, called the global transformation matrix, is estimated. This is done in order for the mean square error of that transformation matrix to be used as threshold in the following steps.

Next, a transformation matrix is calculated for each possible combination of methods. Figure 3.6 illustrates this calculation for three different methods, in which each geometric object represents a distinct method.

Subsequently, each previously calculated transformation matrix H^c is applied to the set of local features F_{all}, obtaining F''_{all}. For each local feature of F''_{all}, the quadratic error

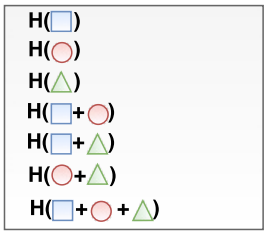
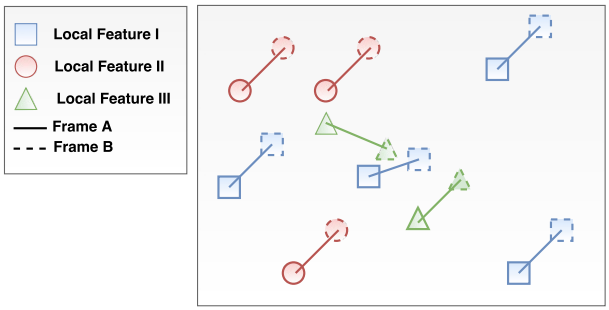


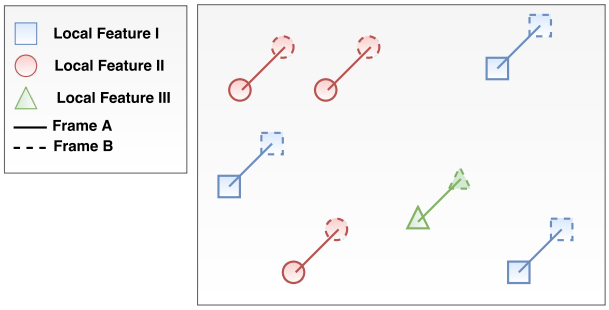
Figure 3.6: Transformation matrices for the combination of local features.

with its equivalent local feature in F'_{all} is calculated. If this quadratic error is less than or equal to the mean square error of the global transformation matrix, such local feature is considered inlier of H^c, and otherwise, outlier of H^c.

Finally, the combined local features are taken as the inliers of the transformation matrix that has the largest number of inliers. Figures 3.7(a) and 3.7(b) illustrate the expected result before and after applying the process of combining local features. The transformation matrix that represents the motion of the frame is then calculated from the combined local features. Whenever it is not possible to estimate a transformation matrix, the identity matrix is considered.



(a) matching of local features



(b) matching of combined local features

Figure 3.7: Combination of local features.

The motivation of this combination is to use different methods and types of local features to confirm the motion between two frames. Typically, the RANSAC method for the removal of outliers is applied. This is important because in addition to removing possible incorrect matches, it also removes matches describing the motion of objects present in the scene, which we are not interested in compensating. As our combination approach uses a consensus method, the subsequent application of RANSAC may be dispensable. The computational performance of the proposed method is highly related to the number of

methods under consideration and their respective costs.

3.1.2 Motion Estimation with Spatio-Temporal Optimization

The diagram shown in Figure 3.8 presents the main steps of the motion estimation proposed with spatio-temporal optimization. Initially, the motion estimation between two consecutive frames is made through local features methods, as described in the previous subsections. Next, the spatio-temporal consistency of the estimation is verified. If there is inconsistency, the motion estimation is done again using the proposed optimization method.

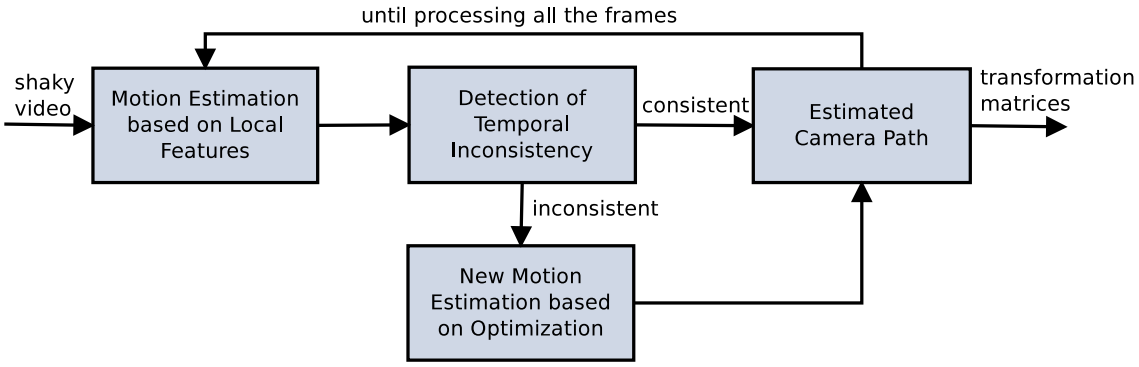


Figure 3.8: Main steps of the proposed motion estimation with spatio-temporal optimization.

Detection of Spatio-Temporal Inconsistency

We applied a consistency check on the estimated matrix, comparing it with the estimated (and final) in the previous frame pair. As premise, we consider that the previous frames have a correct motion estimation. Moreover, difference in the camera motion and by the objects between two frames is small. Even though sudden movements may occur, the video sequences have typically several frames per second, which makes the difference between adjacent frames more gradual.

The structural similarity index (SSIM) [110] is the basic evaluation metric for the detection and optimization steps. In the SSIM calculation, we consider only the previously cropped regions of the frames. We calculate the SSIM image S_t for the frame pair f'_t and f_{t+1}. Then, we calculate the absolute difference of the image pixels of S_t, with the image S_{t-1}. Thus, we obtain the difference image $D_t = S_t - S_{t-1}$, which indicates how similar the image of similarity is in relation to the previous image. Since the images S_t and S_{t-1} consider the transformed frames, the regions of them with high values indicate the presence of remaining movements. If the estimation is correct, such motion corresponds to objects, not to the camera. Thus, if the movements are similar and spatially close in both images (S_t and S_{t+1}), D_t tends to have lower values.

For an estimate to be considered potentially inconsistent, at least one of the following inequalities must be satisfied

$$\text{mean}(S_t) - \text{SSIM}_m * c_S < \text{tol}_S \tag{3.2}$$

where mean(S_t) is the mean of values of the image S_t (also called SSIM value), SSIM$_m$ is the mean of the SSIM values of the last k estimations of previous frames, whereas c_S is a constant that defines the percentage of the mean that will be considered as the threshold and tol$_S$ defines the tolerance.

$$\text{mean}(D_t) - \text{DIFF}_m * c_D > \text{tol}_D \tag{3.3}$$

where mean(D_t) is the mean difference image D_t (also called DIFF value), and DIFF$_m$ is the average of the DIFF values of the last k estimations of previous frames, whereas c_D is a constant that defines the percentage of the mean that will be considered as the threshold and tol$_D$ defines the tolerance.

Inequality 3.2 checks the variation between (i) the similarity between the reference frame f_{t+1} and the transformed frame f'_t and (ii) the similarity obtained in the previous frame pairs. On the other hand, Inequality 3.3 checks the variation between (i) the difference of the similarity of the current and previous frame and (ii) the difference between the pairs of previous frames.

As the estimation of the previous frame pair is correct and the movements of consecutive pairs are comparable, both similarity and difference values should have little variation. If the inconsistency is detected, the motion is again estimated using the optimization method described as follows. Otherwise, the motion estimated by the local features will compose the estimation of the path made by the camera.

Although we use the local feature method as basis, our optimization method can be applied to any other approach that estimates the motion between pairs of frames.

Improved Motion Estimation Based on Optimization

After detecting an inconsistency, the new motion estimation is calculated. This new estimate refers to the transformation matrix that minimizes an objective function based on the similarity value and the difference value. This objective function can be expressed as

$$f = \begin{cases} (\alpha)(1 - \text{mean}(S_{new})) + (1 - \alpha)(\text{mean}(D_{new})) & \text{if conditions are satisfied} \\ 1 + \text{per} + \text{dist} & \text{otherwise} \end{cases}$$

where S_{new} is the SSIM image between the reference frame f_{t+1} and the frame transformed by the matrix being minimized f''_t, considering the cropping area obtained by the same matrix, whereas D_{new} is the difference image between S_{new} and S_{t-1}. The factor α is responsible for the proportion between the two terms. These values are considered only if three conditions are met: cropping condition, coherence condition, and boundary condition.

- cropping condition: in this condition, the cropping percentage (per) should be have lower and higher bounds defined as constants. In this work, we consider $0.8 \leq per \leq 1$. This condition is applied to prevent optimization from finding the minimum considering only a small amount of pixels in the frames. After transformation, a

small region of the frame may be very similar to the region of the reference frame, but this does not fit the entire frame.

- coherence condition: this condition is complementary to the previous one, such that the cropping may be within the established limits, but the corners of the cropping rectangle may be incoherent. For this, four inequalities must be satisfied, which can be expressed as

$$x'_2 > x'_1 \tag{3.4}$$
$$y'_2 > y'_1 \tag{3.5}$$
$$x'_2 \leq \text{rows} \tag{3.6}$$
$$y'_2 \leq \text{columns} \tag{3.7}$$

  where x'_1 is the coordinate x of the left vertex of the cropping rectangle, y'_2 the coordinate y of the lower vertex, whereas rows and columns refer to the dimensions of the original frame.

- limit condition: this condition is used to limit the values of the components of the transformation matrix to be found. In this work, we limit only the value of the scale component, so that it has an absolute value less than or equal to 1.5.

If one of the above conditions is not satisfied, the second term of Equation 3.1.2 is considered. This term is used only as a penalty, whose values are always greater than the first. This is guaranteed by the constant 1. To guide the optimization method, the cropping percentage is also added to the equation, as well as a variable called *dist*. This variable refers to the distance $L1$ between the transformation matrix applied in the previous frame and the matrix to be estimated. Thus, smaller values are obtained if closer to the previous matrix, helping the method to return to a feasible solution.

In this work, the Powell method [88] is used to minimize the objective function presented in Equation 3.1.2. As an initial estimation, we consider the transformation matrix of the previous frame, also based on the premise that the movement has little variation from one frame to another. The Powell method obtained the best results in preliminary experiments. Due to this, the conditions were maintained as presented and not modeled as constraints.

3.2 Removal of Unwanted Motion

After estimating the final similarity matrices for each pair of adjacent frames of the video, a trajectory is calculated for each of the factors. In this work, we consider a vertical translation factor, a horizontal translation factor, a rotation factor and a scaling factor. Each factor f of the matrix is decomposed and the trajectory of each of them is calculated in order to accumulate its previous values, expressed as

$$t_i^f = t_{i-1}^f + \Delta_i^f \tag{3.8}$$

where t_i is the value of a given trajectory in the i-th position, and Δ_i^f is the value of factor f for the i-th similarity matrix previously estimated. The trajectories are then smoothed. The equations presented in the remainder of the text will be always applied to the trajectories of each factor separately. Thus, the factor index f will be omitted in order to not overload the notation.

In contrast to the recent work of the literature [36, 69], we decompose matrices into factors so that it is possible to filter each factor with different intensities.

Assuming that only the camera motion is present in the transformation matrices, the calculated trajectory refers to the path made by the camera during the video recording. To obtain a stabilized video, it is necessary to remove the oscillations from this path, keeping only the desired motion. For this, we apply and analyze the filtering methods presented in Chapter 2.

For the Gaussian filter, the parameter W_g indicates the number of points of the output window, whose value is expressed as

$$W_g = \frac{N}{3} - 1 \tag{3.9}$$

where N is the total number of frames in the video.

The best results in terms of stability were achieved with $\sigma = 40$ in our preliminary experiments. For the Kalman filter, covariance matrix $Q_t = 4e^{-4}$ and uncertainty matrix $R_t = 0.25$ were the values that obtained intensity closer to the Gaussian filter. However, the direct application of the filter is not enough and different instants of the video have distinct oscillations. Therefore, we propose an adaptive Gaussian filter to remove only the unwanted camera motion.

3.2.1 Adaptive Gaussian Filter

The smoothing of an intense motion may result in videos with a low amount of pixels. Moreover, this type of motion is typically a desired camera motion, which should not be smoothed. Therefore, the parameter σ is computed in such a way that it has smaller values in these regions. Thus, the trajectory will be smoothed by considering a distinct value for σ_i at each point i. To determine the value of σ_i, a sliding window of size twice as large as the frame-rate measure is applied, so that the window information lasts for two video seconds. The ratio r_i is expressed as

$$r_i = \left(1 - \frac{\mu_i}{max\_value}\right)^2 \tag{3.10}$$

where max_value corresponds to either width in the horizontal translation trajectory or height in the vertical translation trajectory. In this work, we consider $\theta = \frac{\pi}{6}$ as the angle (in radians) in the rotation trajectory. Thus, the motion will be considered large based mainly on the video resolution. Value μ_i is calculated in such a way to give higher weights

to points closer to i, where μ_i is expressed as

$$\mu_i = \frac{\sum_{j \in W_i,\, j \neq i} G(|j-i|, \sigma_\mu)\Delta_j}{\sum_{j \in W_i} G(|j-i|, \sigma_\mu)} \tag{3.11}$$

where j is the index of each point in the window of i, whereas $G()$ is a Gaussian function with σ calculated as

$$\sigma_\mu = (\text{FPS})(1 - \text{CV}) \tag{3.12}$$

where FPS is the video frames per second, and CV is the coefficient of variation of the absolute values of the trajectory that are inside the window. Since the value of CV lies between 0 and 1, its final value is limited to 0.9 in order for σ_μ not to have null values. Therefore, σ_μ makes the actual size of the window adaptive, such that the higher the variation of motion inside the window, the higher the weight given to the central points.

The coefficient of variation can be expressed as

$$CV = \frac{std(\forall t_i \mid i \in W_i)}{avg(\forall t_i \mid i \in W_i)} \tag{3.13}$$

where W_i is the same window as in Equation 3.11 and t_i the trajectory value. Therefore, the coefficient of variation corresponds to the standard deviation *std* to the average *avg*.

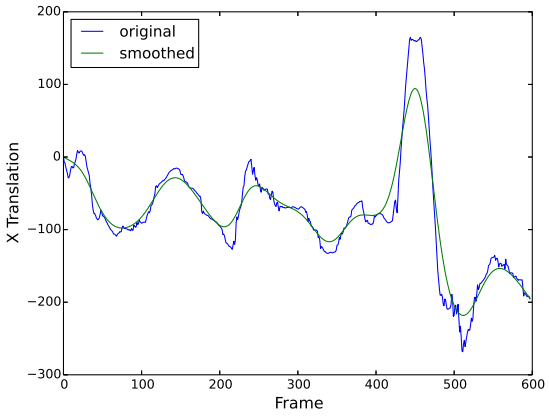
Assuming that r_i ranges between 0 and 1, a linear transformation is applied to obtain a proper interval for the Gaussian filter. This transformation is given as

$$\sigma_i = \frac{\sigma_{\max} - \sigma_{\min}}{r_{\max} - r_{\min}} (r_i - r_{\min}) + \sigma_{\min} \tag{3.14}$$

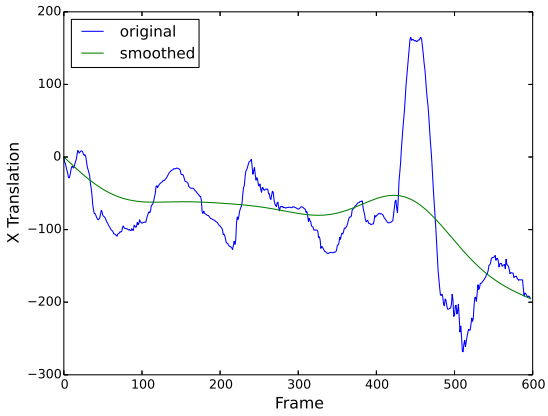
where $\sigma_{\min}$ and $\sigma_{\max}$ are the minimum and maximum values of the new interval (after linear transformation), respectively. In this work, these values are defined as 0.5 and 40, respectively. Values $r_{\min}$ and $r_{\max}$ are the minimum and maximum values of the old interval (before linear transformation). In this work, value $r_{\max}$ is always set to 1. To control whether a motion is unwanted, a value in the interval between 0 and 1 is set to $r_{\min}$. The same $r_{\min}$ is used as a lower limit to r_i, before applying the linear transformation.

An exponential transformation is then applied to σ_i values to amplify their magnitude. After calculating σ_i for each point of the trajectory, its values are lightly smoothed by a Gaussian filter with $\sigma = 5$, chosen empirically. This is done to avoid abrupt changes in the value of σ_i along the trajectory. Finally, the Gaussian filter is applied n times (once for each point in the trajectory), generating a smoothed trajectory (indexed by k) for each σ_i previously calculated. The final smoothed trajectory corresponds to the concatenation of points for each of the generated trajectories, and the k-th trajectory contributes with its k-th point. Thus, an adaptive smoothed path is obtained.

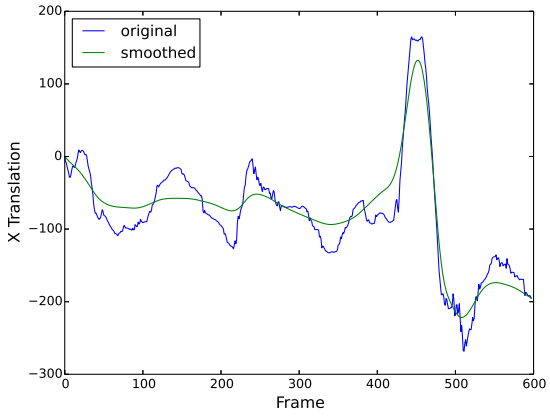
Figures 3.9a, 3.9b and 3.9c show the trajectory generated by considering the horizontal translational factor (blue) and the obtained smoothing (green), respectively, using the Gaussian filter with $\sigma = 20$ and $\sigma = 40$, besides the adaptive version proposed in this work. It is possible to observe that the smoothing is applied at different degrees along the trajectory.



(a) Gaussian filter with σ=10



(b) Gaussian filter with σ=40



(c) Adaptive Gaussian filter

Figure 3.9: Smoothing of camera motion trajectories.

3.3 Generation of the Corrected Video

After applying the motion filtering, it is necessary to recalculate the value of each factor for each transformation matrix. In order to do that, the transformation matrix value of a given factor is calculated by the difference between each point of its smoothed trajectory and its predecessor. With the transformation matrices of each pair of frames updated, the transformation matrix is applied to the first frame of the pair to take it to the coordinates of the second.

Applying the geometric transformation in the frame causes information to be lost in certain pixels of the frame boundary. Figure 3.10 presents a transformed frame, where it is possible to observe the loss of information at the borders. They are then cropped so that no frames in the stabilized video hold pixels without information. To determine the frame boundaries, each transformation matrix is applied to the original coordinates of the four vertices, thus generating the transformed coordinates for the respective frame. Finally, the innermost coordinates of all frames are considered final. Figure 3.11, extracted from [79], illustrates the cropping process applied to the transformed frame.

Figure 3.10: Frame after application of geometric transformation.

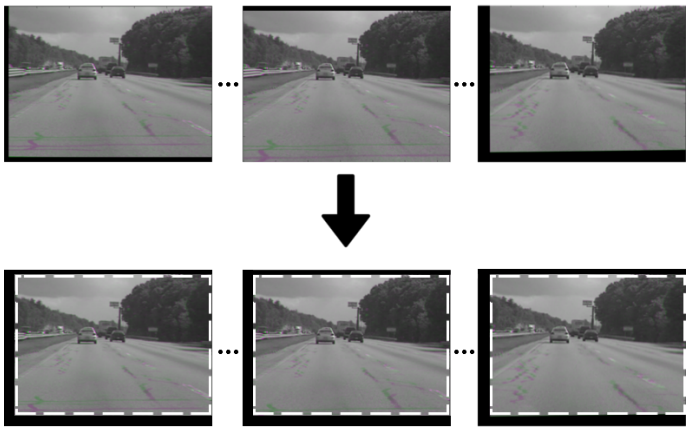


Figure 3.11: Frame after boundary cropping.

3.4 Evaluation of Stabilization

After the stabilization process, we performed an evaluation of the stability of the video, in order to verify and analyze the performance of different methods in different scenarios. Although it is not very common in the literature, we believe that evaluating the motion estimation step separately is essential in the evaluation of stabilization. Since the motion estimation has a great influence on the final stabilization result, and can be difficult to detect the stabilization failure with a single and final evaluation.

One way to evaluate the motion estimate is with similarity measures, such as the PSNR or SSIM. However, as discussed later in Chapter 4, such measures may not represent the motion estimation in a coherent way. The overlapping of the frames, as shown in Figure 3.10, can be used for this evaluation. For this, we generate a new video from the motion estimation. In which, each frame g_t of the new video is composed of the overlap of the transformed frame f'_t and the original frame f_{t+1}. Thus, we can evaluate the motion estimation by the overlapping of each frames pair and by the temporal information contained in the video. So, the background must be correctly overlapped, and in addition, the overlapping region must have some continuity.

In the evaluation of the entire stabilization, we believe that two main attributes should be considered: (i) the rate of preserved pixels and (ii) the stability of the video. The rate of preserved pixels is reported by few works in the literature. However, we believe that

a stabilization that maintains very little information is unsatisfactory, regardless of the stability of the video generated. Therefore, such that measure is indispensable.

The rate of preserved pixels in a video can be expressed as

$$\text{Rate of preserved pixels} = 100\frac{W_s H_s}{WH} \tag{3.15}$$

where W and H correspond to the width and height of the frames in the original video, W_s and H_s correspond to the width and height of the frames in the video generated by the stabilization process, respectively.

To evaluate the stability of the video in a qualitative way, we propose two approaches: an approach based on visual rhythms, presented in the subsection 3.4.1. And another based on the motion energy image, exposed in the subsection 3.4.2.

3.4.1 Visual Rhythms

In the evaluation based on visual rhythms, two different path directions are considered: horizontal and vertical. The vertical rhythm extracts the information from the columns of each frame, while the horizontal rhythm takes the information from the lines of each frame.

For both path directions, the rhythm is obtained from the sequential concatenation of the information, so that the j-th column of the visual rhythm image corresponds to the information in the j-th frame. In the horizontal rhythm, a rotation is performed on the rows in order to obtain the columns in the final image. The width of a visual rhythm corresponds to the number of frames of the video, whereas its height corresponds to the height or width of the frames for the vertical or horizontal rhythm, respectively.

Figure 3.12 shows the relations between the pixels of the neighborhood in a visual rhythm image, from which we can see that the visual rhythm maintains the temporal and spatial information of the video. Thus, the temporal behavior of the gray levels in a certain region can be easily visualized. This provides information on how and when movements occur in the video, that is, in addition to being able to distinguish the direction, the intensity, and the form that the movements are spatially arranged, we can verify the frequency of certain type of movement and determine the moments of its occurrence. Stable video is expected to have a more uniform visual rhythm, with fewer twitches and better defined curves.

Figure 3.13 shows the construction of a horizontal rhythm for two frames 3×3. At the transition between frames A and B, the camera moves from right to left, causing the pixels to be to the right of their original position. Thus, when obtaining the horizontal rhythm, the pixels of the column corresponding to frame B are below the equivalent pixels of frame A, thereby forming a declination.

The separation of the vertical and horizontal visual rhythms is important to detect and evaluate problems in the video stabilization process more thoroughly. From the vertical rhythm, we can analyze the characteristics of the motion in the y axis. Thus, inclined rhythm lines indicate camera movements from the bottom to top, whereas declined lines indicate camera movements from top to bottom. From the horizontal rhythm, in turn, we

Neighbor i of the previous frame	Neighbor i	Neighbor i of the next frame
Same pixel of the previous frame	Pixel	Same pixel of the next frame
Neighbor $i+1$ of the previous frame	Neighbor $i+1$	Neighbor $i+1$ of the next frame

Figure 3.12: Patterns for pixel neighborhood in the visual rhythm.

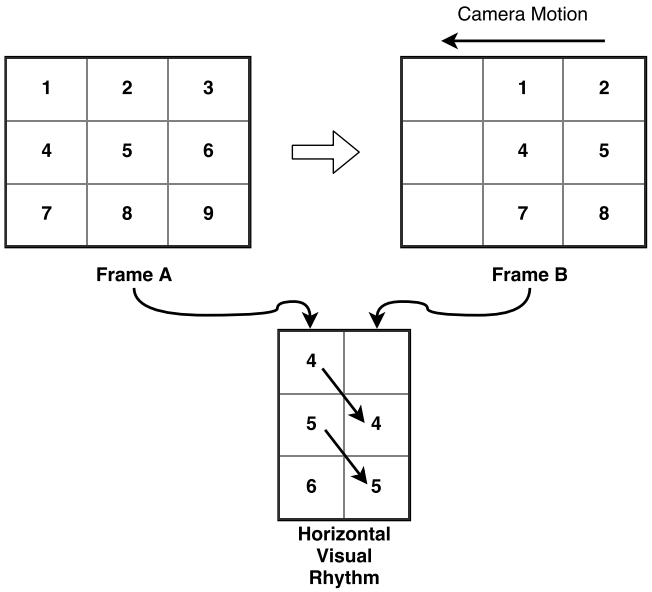


Figure 3.13: Direction of horizontal visual rhythm.

have the characteristics of the motion in the x axis. Thus, sloped lines indicate camera movements from left to right, whereas declined lines indicate camera movement from right to left.

The use of only one column or row in the extraction of information from each frame may be inadequate since it considers little information of the frame. In addition, it makes horizontal and vertical separation less accurate. This problem can be seen in Figure 3.14, where a vertical movement of the camera occurs, which can influence the horizontal rhythm, depending on the difference of the pixels between the rows. Thus, the average of the columns or rows is adopted in our work to compensate for this difference, making the horizontal rhythm less sensitive to vertical movements, and the vertical rhythm less sensitive to horizontal movements.

In Figure 3.14, both columns of the horizontal rhythm should have either the same values or values very close. However, with a single row in each frame, the direction of the rhythm is uncertain.

As post-processing, we apply an adaptive histogram equalization technique through

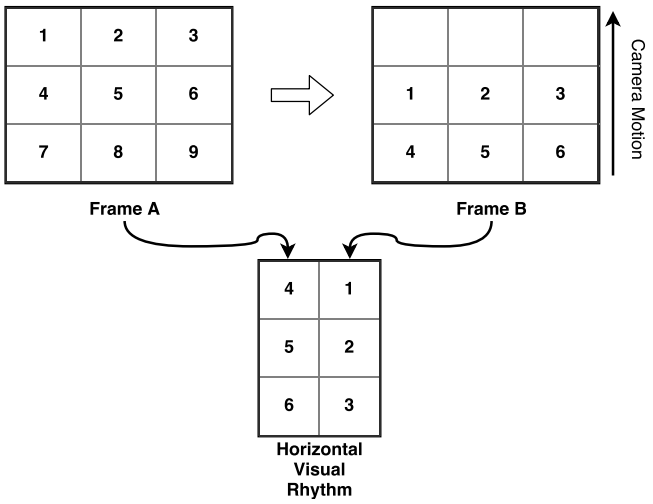


Figure 3.14: Direction of horizontal visual rhythm with a single row.

the Contrast Limited Adaptive Histogram Equalization (CLAHE) [119]. This is done to improve the contrast of the visual rhythm, facilitating human perception.

The construction of the visual rhythms is not based on motion estimation, as occurs in other visualizations, shown in Chapter 2. Therefore, their performance is not dependent on any motion estimation technique, which makes the representation of the video motion more reliable. In the context of video stabilization, such independence of methods for motion estimation is crucial to allow a more unbiased assessment of the results.

Among the good practices in the construction of visual rhythms for the evaluation of video stabilization results, we recommend to:

- crop the frames of the stabilized video so that there are no pixels with null information (since null information may imply inadequate row or column averages);
- preserve the frame rate of the video in order to not change its number of frames or generate visual rhythms of different sizes;
- rescale the video frames to the original size in order for the visual rhythms to have the same size.

3.4.2 Motion Energy Image

We conjecture that the stabilization evaluation can be done through the amount of motion present in the video, which complements the analysis of the motion behavior. Thus, we propose a stabilization evaluation method based on the motion energy image. Figure 3.15 presents the main stages of our method.

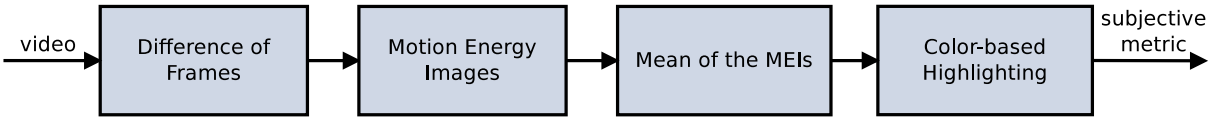


Figure 3.15: Main steps of the proposed stabilization evaluation method.

For each video frame i, the difference of the gray level intensities of each pixel is calculated. This is done by considering the pre-processed frames through a Gaussian filter with kernel experimentally set as $\sigma = 5$, which is applied to smooth the frames, so that the difference is calculated without disregarding unnecessary details. In this step, a binary image is obtained, in which 1 is assigned to the pixel with difference greater than a certain threshold, and 0 otherwise. This calculation can be seen as a sub-step of the MEI construction, expressed as

$$\text{Diff}_{i,j}(x,y) = \begin{cases} 1 & \text{if } \texttt{med}(|f_i(x,y) - f_j(x,y)|) \geq T \\ 0 & \text{otherwise} \end{cases} \tag{3.16}$$

where (x, y) denotes a given pixel and f is the already smoothed frame. In turn, i and j correspond to the i-th and j-th frame indices, respectively. T corresponds to the threshold, experimentally chosen as 10. Finally, `med()` is a median filter with kernel of size 5, applied to decrease the discontinuities of the differences.

We consider an MEI for each frame i, which is obtained through the differences of the frames within a sliding window of size W_{MEI}, centered in i. The MEI calculation can be expressed as

$$\text{MEI}_i = \frac{\sum\limits_{j \in \Omega_i, j \neq i} G(|i-j|, \sigma)\text{Diff}_{i,j}}{\sum\limits_{j \in \Omega_i, j \neq i} G(|i-j|, \sigma)} \tag{3.17}$$

where $G()$ is a Gaussian function that assigns larger weights to the differences of the nearest frames. Ω_i is the neighborhood of i determined by the sliding window.

In contrast to the MEI calculation typically performed in the literature, we consider the differences from the central window frame. This is done so that motion that occurs more gradually can be captured by MEI.

The window size W_{MEI} is based on the number of frames per second (FPS), in order to always consider the same time interval, expressed as

$$W_{MEI} = \frac{\text{FPS}}{5} \tag{3.18}$$

where 5 was chosen empirically.

The use of a Gaussian function to provide larger weights for the frames closer to the central frame is premised on the fact that oscillations present in unstable videos usually occur more suddenly than a desired motion.

By taking the MEI of each frame, the average image of the MEIs is calculated, where each pixel (x, y) is taken as the arithmetic mean of the pixels (x, y) of all the MEIs of the video. Thus, from the gray level image obtained, it is possible to verify the amount of motion present in the video, its location and spatial distribution in the frames.

The human visual system can distinguish thousands of tones and intensities of color, however, only a few tens of shades of gray [33, 85]. Thus, a pseudocolor transformation is applied, so that high gray-level intensity values are mapped to red, whereas lower intensities to blue. Figure 3.16 shows the color mapping used. A more stable video is

expected to have less motion and, therefore, a view with colors that are closer to blue than an unstable video is obtained.

Figure 3.16: Pseudocolor transformation applied to the images of the average MEIs.

Chapter 4

Experiments

Results obtained in the experiments are described in this chapter. Three databases are used to evaluate the effectiveness of the proposed video stabilization methods. The first consists of eleven videos available in the GaTech VideoStab [36] dataset and three others collected separately. The second, available by Liu et al. [69], consists of 139 videos divided into categories. Finally, we create a dataset that is complementary to the others, in which four videos are collected separately. From the original videos, excerpts with moving objects in the foreground and with little representative backgrounds are extracted, generating a total of eight videos.

4.1 Datasets

Table 4.1 reports a summary of the first database with videos in alphabetical order. This dataset is composed of videos with more general situations, where some videos can be stabilized successfully even using simpler stabilization methods, while others require more complex methods. From now on, we will refer to the videos in this database through the identifiers assigned to each of them. Videos `#1`, `#2`, `#3`, `#4`, `#6` and `#8` present simpler motion, not requiring complex stabilization methods.

Video `#5` contains intense zoom. This type of video may cause certain difficulties in the removal of the unwanted motion step, making simpler methods keep few video frame pixels. Video `#7` contains objects in constant motion, which may cause difficulties in the stability of the final video if the motion estimation step is not able to deal with such objects.

Video `#10` presents non-rigid motion, that is, methods based on a global motion estimation may not yield good results. Video `#11` contains an object very close to the camera, occupying much of the scene. This type of situation may also compromise the quality of final stabilization if the motion estimation is not done correctly and does not disregard the object in all frames.

Video `#12` presents the rolling shutter effect. In this case, a wobble suppression method or a more local estimation is necessary to achieve higher quality stabilization. Videos `#10` and `#14` contain a large number of translational movements made intentionally during the video acquisition process. Just as in the case of zooming, a simpler unwanted motion

removal method can lead to excessive frame cropping and hold only few pixels.

Table 4.1: Video sequences from the first dataset.

#	**Video**	**Source**	**Features**
1	gleicher1	GaTech VideoStab	Regular Video
2	gleicher2	GaTech VideoStab	Regular Video
3	gleicher3	GaTech VideoStab	Regular Video
4	gleicher4	GaTech VideoStab	Regular Video
5	greyson_chance	GaTech VideoStab	Zooming
6	hippo	nghiaho.com/uploads/hippo.mp4	Regular Video
7	lf_juggle	GaTech VideoStab	Moving Objects
8	new_gleicher	GaTech VideoStab	Regular Video
9	sam_1	GaTech VideoStab	Moving Objects
10	sam and cocoa	youtu.be/627MqC6E5Yo	Non-rigid Camera Motion
11	sany0025	GaTech VideoStab	Near Objects
12	shake_pgh_1	GaTech VideoStab	Rolling Shutter and Scene Change
13	shaky_car	MatLab	Abrupt Motion
14	yuna_long	GaTech VideoStab	Intentional Motion

Table 4.2 presents the database proposed by Liu et al. [69], which is divided into six categories, containing a total of 139 videos. The `Crowd` category has videos in the presence of crowds. Videos in the `Parallax` category have large parallax. The `Quick Rotation` category has videos with abrupt translations. The videos in the `Regular` category are more general. In the `Running` category, the camera operator is always running, which produces videos with a lot of translational movements. In the `Zooming` category, videos are characterized by the presence of zoom. Subsequently, we will refer to the videos in this dataset by the name of the category followed by the identifier of each video, attributed by the authors.

Table 4.2: Categories and amount of videos present in the second dataset.

Category	# **Videos**
Crowd	22
Parallax	17
Quick Rotation	28
Regular	22
Running	21
Zooming	29
Total	139

Table 4.3 presents a summary of the videos from the database created in this work. We will refer to the videos in this database through the term $\texttt{ours}_{\texttt{ID}}$, where `ID` is the identifier assigned to each video. Video $\texttt{our}_1$ consists of an artificial video created by Liu

et al. [70] in which an object was added to an already unstable video. This object moves forward and backward, occupying different proportions of the scene along the video. This scenario presents difficulties for the motion estimation step due to possible interference of the object, especially when it occupies most of the scene. Videos $\texttt{ours}_2$, $\texttt{ours}_{3-5}$ and $\texttt{ours}_{6-8}$ are collected from `YouTube` and consists of drones flying in indoor scenarios. They also present difficulties for the motion estimation step, since closed scenarios usually have little representative background and the drones are constantly moving.

Table 4.3: Video sequences created in this work.

#	**Video**	**Source**	**Range**
1	artificial	Liu et al. [70]	entire video
2	drone	youtu.be/z5GQJq2esw0	00:00:06 to 00:00:26
3	witch_1	youtu.be/LsaVrWCma9k	00:00:00 to 00:00:30
4	witch_2		00:00:15 to 00:00:30
5	witch_3		00:00:30 to 00:00:42
6	witch2_1	youtu.be/a8_nko2MeCE	00:00:00 to 00:00:15
7	witch2_2		00:00:15 to 00:00:30
8	witch2_3		00:00:30 to 00:00:43

In the following sections, the videos present in the described datasets are used to conduct the experiments required to evaluate the methods proposed in this work. All experiments were implemented in Python programming language [74] version 2.7.6 with the following libraries: SciPy [45], NumPy [107], scikit-learn [84], scikit-image [108] and OpenCV [11].

4.2 Evaluation Metrics

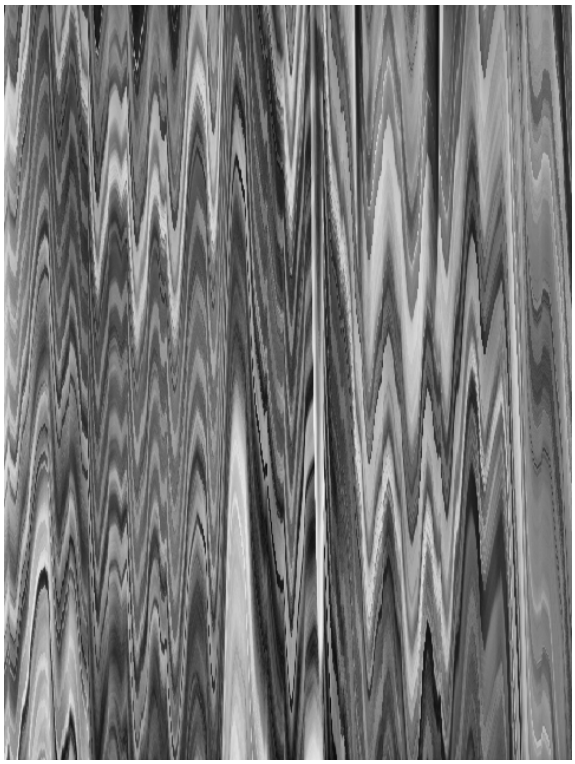
The results obtained with the evaluation metrics are reported in this section, which is divided into two subsections. The first presents the results obtained with the visualization based on visual rhythms, whereas the second shows the results of the evaluation based on the motion energy image. These results are presented first because they will serve as a basis for reporting the results obtained with the other stages of our method. The inconsistencies of objective metrics will be presented throughout this chapter. Despite these problems, objective metrics are used in this work for the comparison of large datasets.

4.2.1 Visual Rhythms

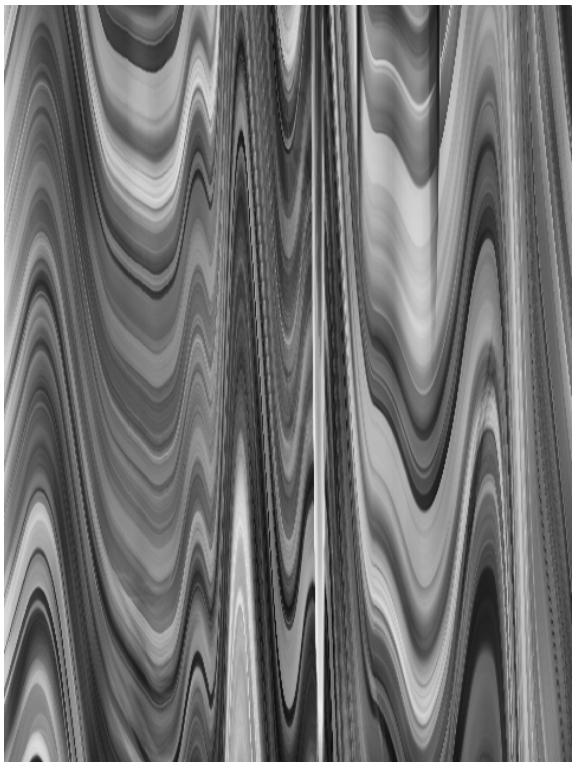
In this subsection, we present the results obtained through the visualization based on visual rhythms. Vertical and horizontal rhythms, extracted from some videos of the evaluated datasets, are shown.

Figure 4.1 presents the visual rhythms generated for the video `#12` before and after the video stabilization process. This experiment was done to verify if an unstable video could be differentiated from a stabilized video. In order to obtain the stabilized version

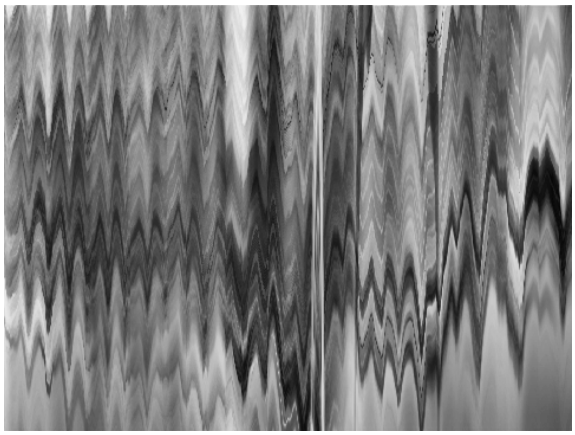
of the video, we submit it to YouTube, which applies one of the state-of-the-art digital video stabilization approaches [36]. The width of all the images presented in this section was considered constant for a better organization.



(a) horizontal visual rhythm - original video



(b) horizontal visual rhythm - stabilized video



(c) vertical visual rhythm - original video

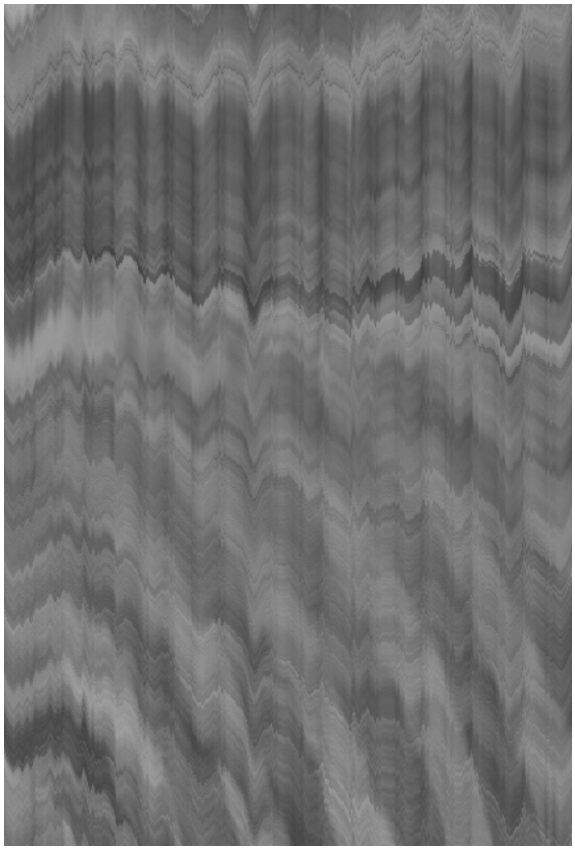
(d) vertical visual rhythm - stabilized video

Figure 4.1: Visual rhythms for video #12.

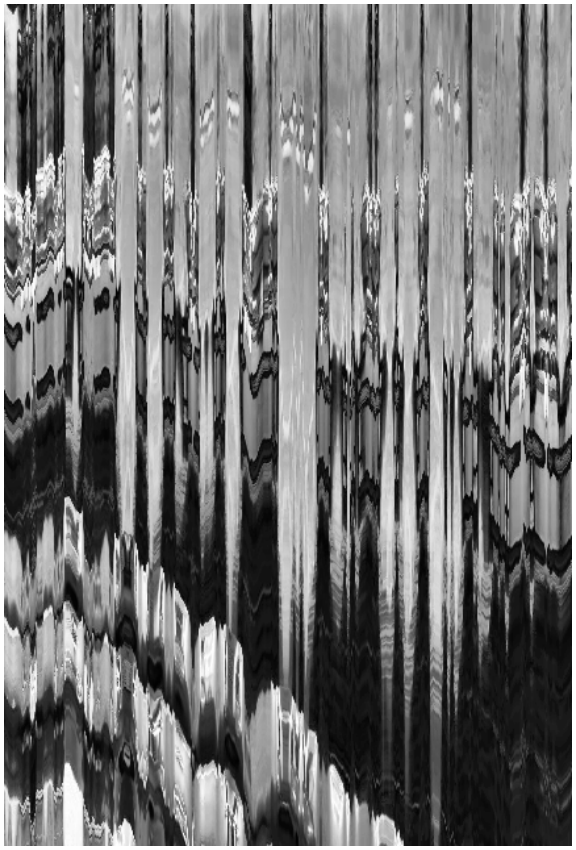
From the horizontal visual rhythm of the unstable video, shown in Figure 4.1a, we can notice the twitches and irregularities present in the lines. On the other hand, in the horizontal visual rhythm of the stabilized video, shown in Figure 4.1b, there are more continuous, well defined and softer lines. Analogously, the vertical visual rhythm of the unstable video, shown in Figure 4.1c, has twitches and irregularities that are eliminated in the visual rhythm of the stabilized video, shown in Figure 4.1d. We can also observe that vertical and horizontal rhythms are not influenced by each other, where certain motion

regions occur in one but not in the other.

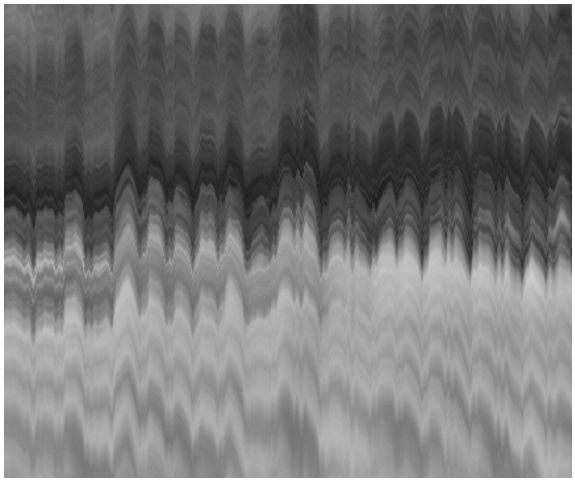
For the video $\texttt{Regular}_8$, we present a comparison of the visual rhythms obtained through the average of the rows or columns, and through the column or central row. In this case, we present the horizontal and vertical visual rhythms only for the unstable video. We present this figure to show the superiority of our strategy, which uses the mean rows and columns in order to have a better separation of horizontal and vertical movement.



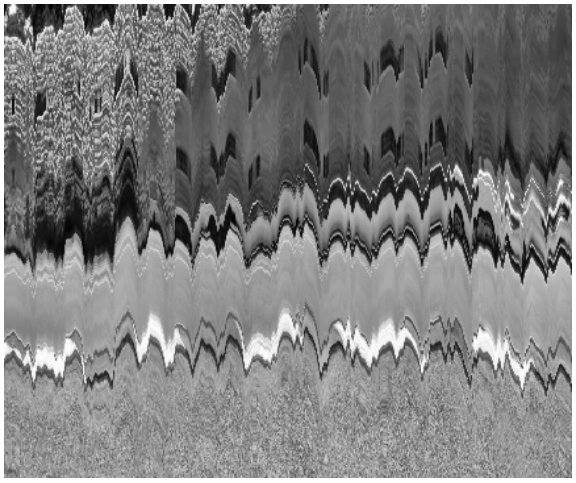
(a) horizontal visual rhythm - mean row



(b) horizontal visual rhythm - central row



(c) vertical visual rhythm - mean column

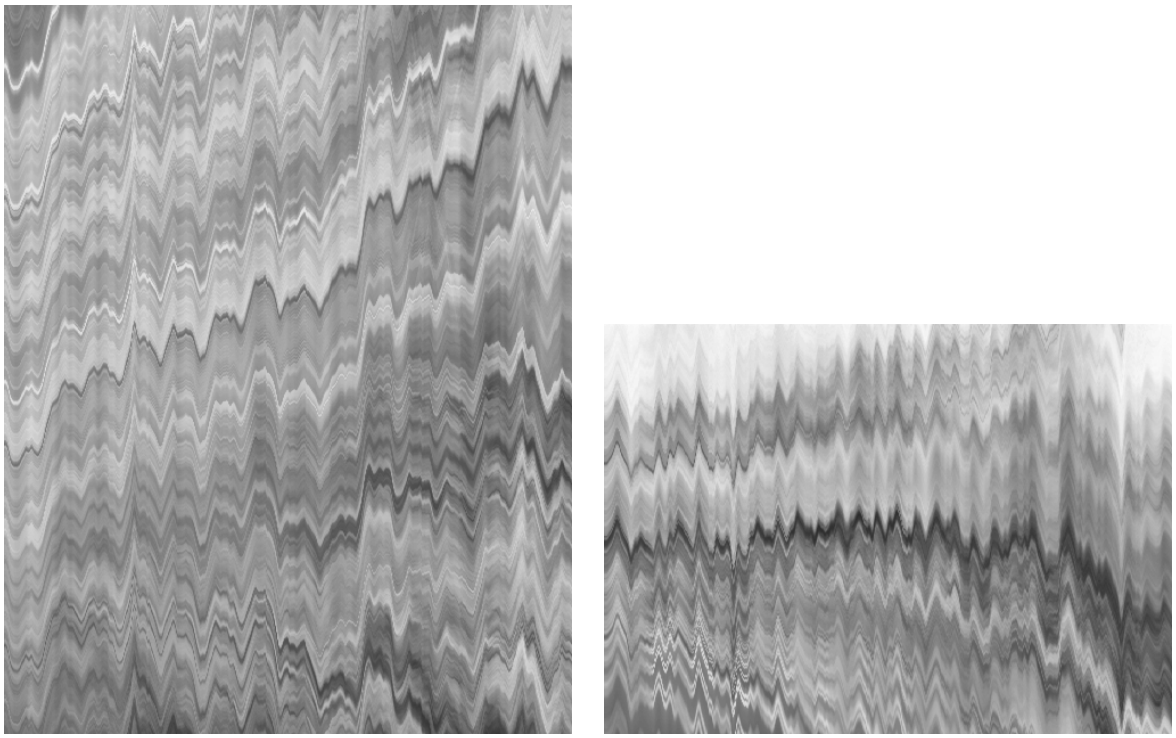


(d) vertical visual rhythm - central column

Figure 4.2: Visual rhythms for original video $\texttt{Regular}_8$.

It can be seen from Figure 4.2a that the visual rhythm with only one row can be negatively influenced by the vertical motion of the video, with artifacts that do not correspond to the horizontal motion, such as the discontinuities present in the rhythm, whereas the visual rhythms presented by their average are more consistent with the motion present in the video. An analogous behavior can be seen in the vertical rhythm shown in Figure 4.2c.

Figure 4.3 presents the visual rhythms of the unstable video `#1`. For this video, we present the rhythms obtained after the stabilization of `YouTube`, in addition to a stabilization with inferior performance. Figure 4.4 shows the horizontal and vertical rhythms for both versions of the stabilized video. The purpose of this experiment is to demonstrate that the visualization can also distinguish videos stabilized by different methods.

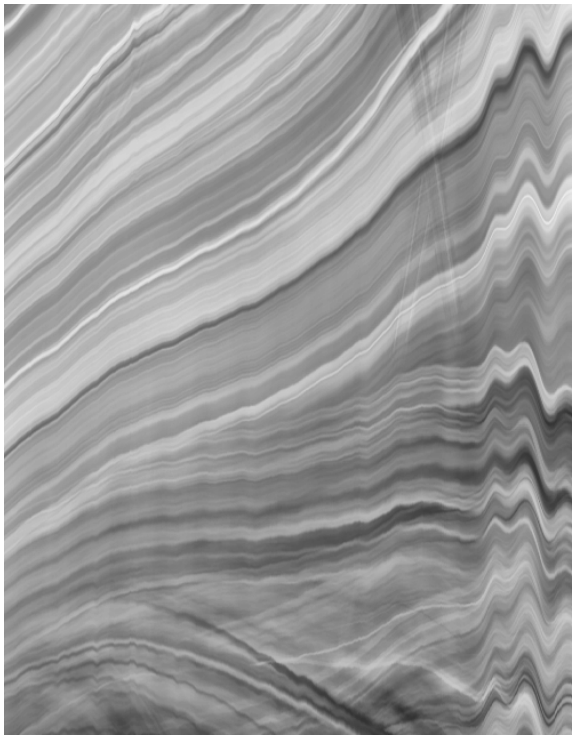


(a) horizontal visual rhythm (b) vertical visual rhythm

Figure 4.3: Visual rhythms for original video `#1`.

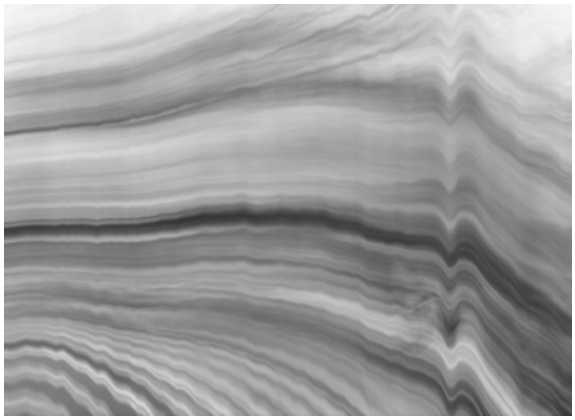
By comparing the visual rhythms for the unstable video and the rhythms for the stabilized videos, it is possible to confirm the validity of using visual rhythms to compare versions of stable and unstable videos. In addition, from the visual rhythms of the two different methods, illustrated in Figure 4.4, we can observe the occurrence of less twitches and smoother lines throughout the entire rhythm, both for the horizontal and vertical rhythm. This shows that the visual rhythm can be used in the comparison of two different video stabilization methods.

The horizontal and vertical rhythms for the original and stabilized video `Quick Rotation`$_0$ are shown in Figure 4.5. In this case, the video was stabilized with the method proposed by Liu et al. [69]. The version of the video `#4` stabilized with `YouTube` was not shown here since the method modified its frame rate, reducing the number of frames and making the visualization of the stabilized video considerably smaller

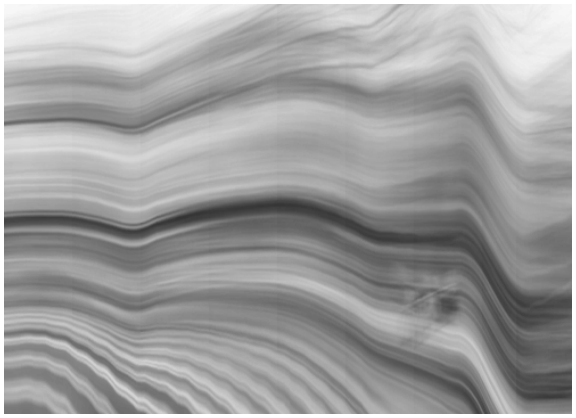


(a) horizontal visual rhythm - stabilization with an inferior result

(b) horizontal visual rhythm - `YouTube` stabilization



(c) vertical visual rhythm - stabilization with an inferior result



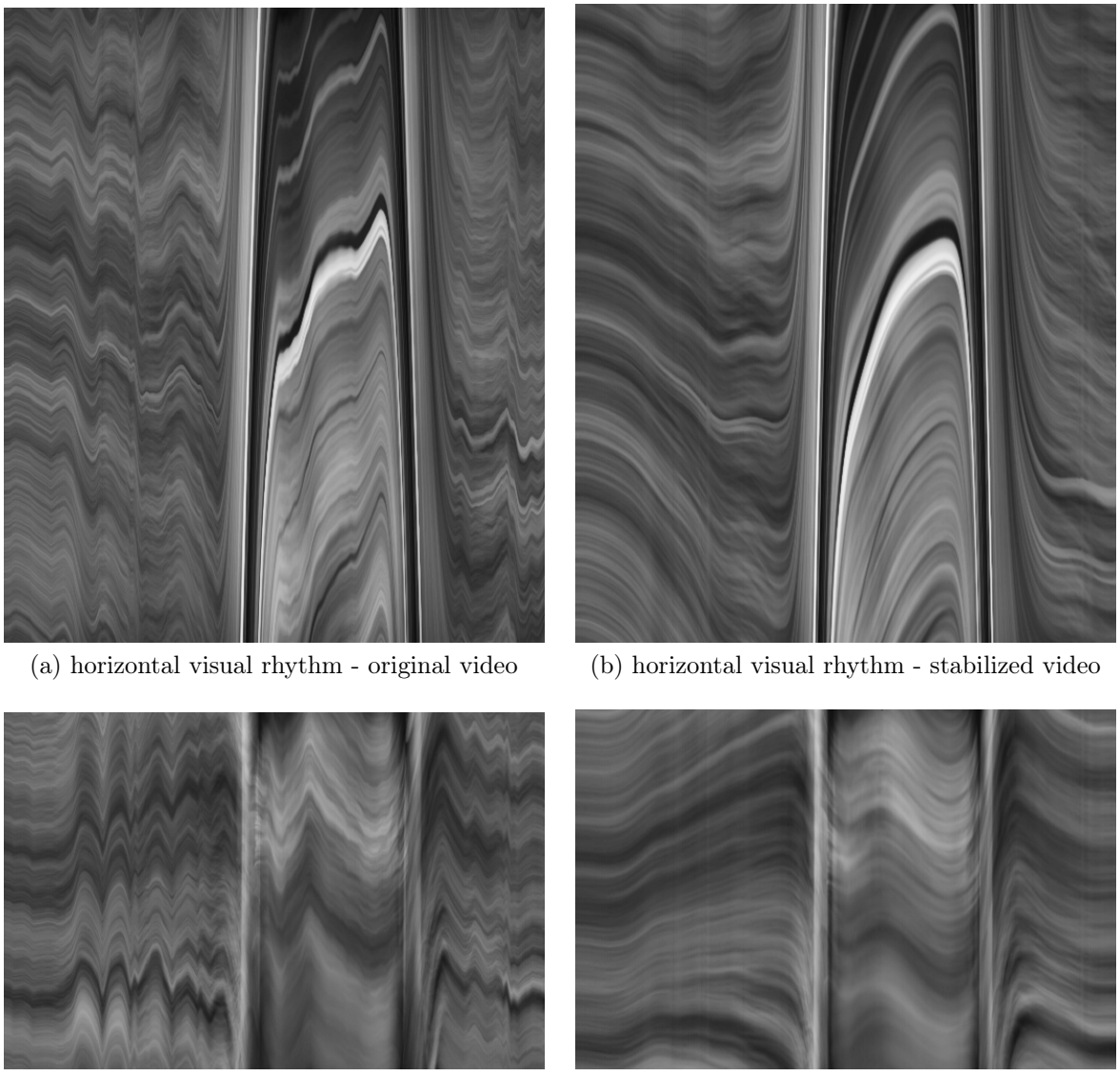
(d) vertical visual rhythm - `YouTube` stabilization

Figure 4.4: Visual rhythms for stabilized video `#1`.

than the original video. The objective of this experiment is to show that it is possible to notice the presence of abrupt translations in the video from the visualization.

Besides confirming that the smoother lines obtained with the visual rhythm for the stabilized video, it is possible to observe the occurrence of totally vertical lines in the horizontal visual rhythms, which indicates a very fast horizontal movement of the camera. It is also possible to notice that the horizontal lines are inclined in their origin, which indicates that the displacement is from left to right.

In Figure 4.6, we present the horizontal and vertical visual rhythms for the original and stabilized video $\texttt{Zooming}_0$. The video was stabilized through the method proposed



(a) horizontal visual rhythm - original video

(b) horizontal visual rhythm - stabilized video

(c) vertical visual rhythm - original video

(d) vertical visual rhythm - stabilized video

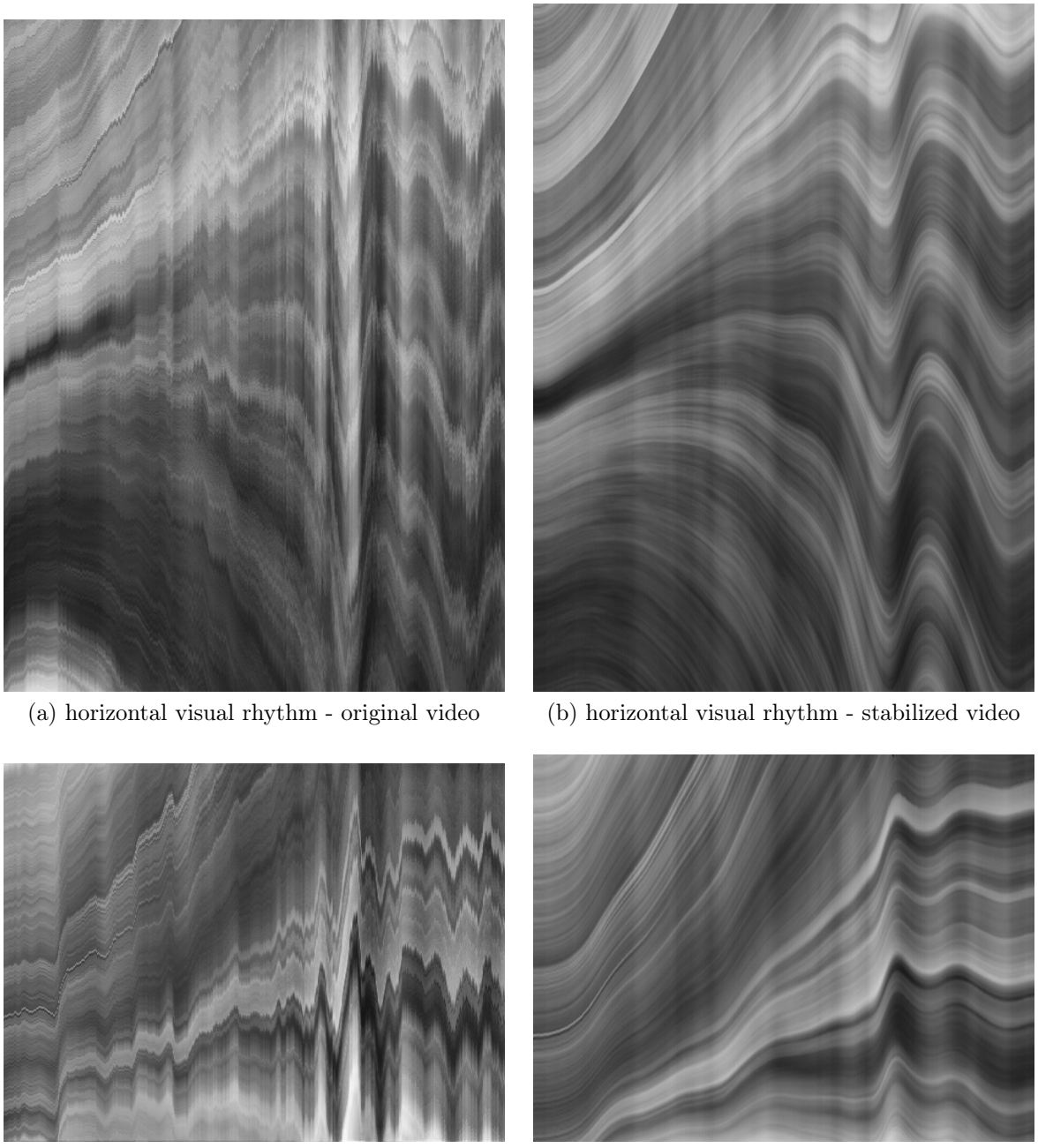
Figure 4.5: Visual rhythms for video $\texttt{Quick Rotation}_0$.

by Liu et al. [69]. The aim of this experiment is to show that it is possible to notice the presence of zoom in the video from the visualization.

In the visual rhythms for video $\texttt{Zooming}_0$, it is also possible to see the presence of well defined, regular lines in the visual rhythm of the stabilized video. In addition, it is possible to observe inclined and declined lines in the horizontal visual rhythms present simultaneously in the beginning of the video, which indicates the existence of zoom.

4.2.2 Motion Energy Image

In this subsection, we present the results obtained in our experiments with the evaluation based on the motion energy image. Figure 4.7 presents the difference images by considering several indices within a sliding window for the originally unstable video. Figure 4.8 illustrates the images corresponding to the videos obtained after the stabilization process



(a) horizontal visual rhythm - original video

(b) horizontal visual rhythm - stabilized video

(c) vertical visual rhythm - original video

(d) vertical visual rhythm - stabilized video

Figure 4.6: Visual rhythms for video $\texttt{Zooming}_0$.

through the `YouTube` approach [36].

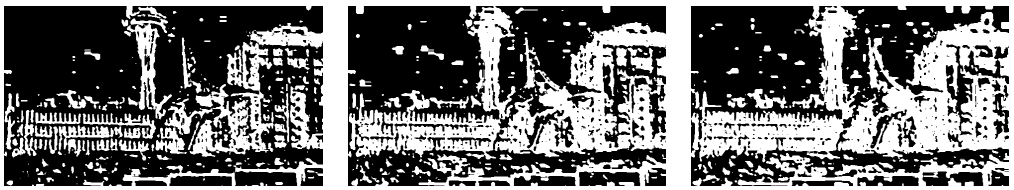
From Figures 4.7 and 4.8, we can notice the presence of more white pixels in the images of the difference of the unstable video, indicating a larger amount of motion. This is even more visible with the increase in frame distance. These results confirm that the use of the difference between frames with a certain distance can capture motion that is not perceived by comparison of adjacent frames.

Figure 4.9 shows the MEI for the same frame obtained for both the unstable video and the stabilized video. It can be verified that the MEI summarizes well the images of the differences and that the version of the stabilized video has darker pixels compared to

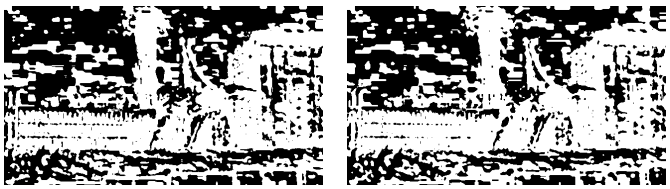
(a) $j = 3$ (b) $j = 4$ (c) $j = 5$

(d) $j = 6$ (e) $j = 7$

Figure 4.7: Difference images for unstable video #4 with $i = 2$.



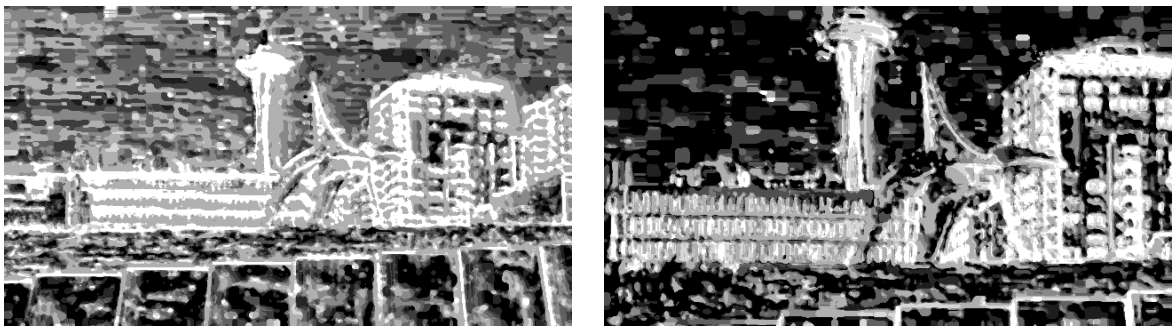
(a) $j = 3$ (b) $j = 4$ (c) $j = 5$



(d) $j = 6$ (e) $j = 7$

Figure 4.8: Difference images for video #4 after stabilization with $i = 2$.

that of the unstable video, which indicates the presence of less motion.

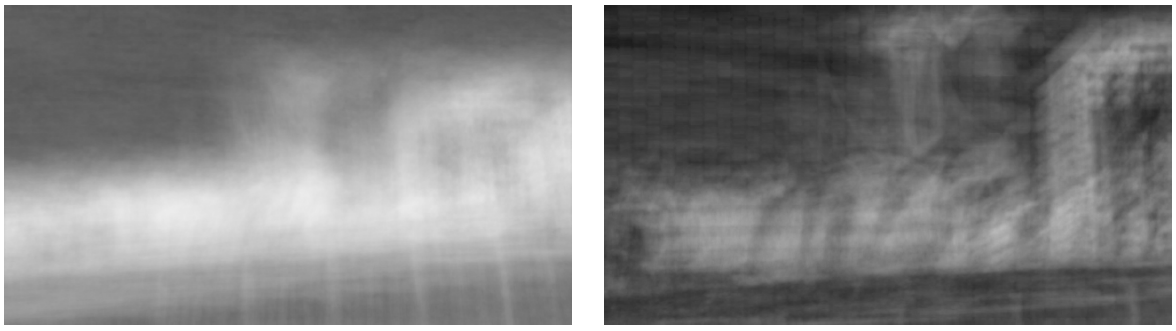


(a) original video (b) stabilized video

Figure 4.9: MEI for video #4 with $i = 2$.

Figure 4.10 illustrates the gray level image of the average of the MEIs for the unstable and stabilized video. From the figure, it is possible to observe an image with darker gray

levels and with more defined shapes in the image corresponding to thee stabilized video. Similar results were observed in all videos in the database under consideration.



(a) original video (b) stabilized video

Figure 4.10: Image of average of the MEIs for video #4.

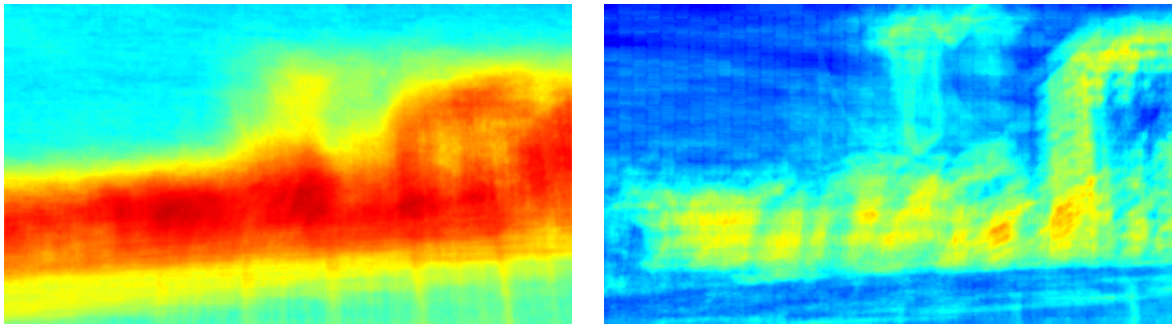
In the following results, we present the images obtained with the proposed method and compare them with the average grayscale of the video frames, as shown in Figure 2.5 described in Section 2. The purpose of these experiments is to demonstrate that our visualization can distinguish unstably and stabilized videos, as well as show that it is superior to the simple average of the gray tones of the frames.

Figure 4.12 shows a color image of the average of the MEIs for the unstable and stabilized video #4. It is possible to observe a greater visual distinction when compared to the gray level image. For the unstable video, the image contains red regions, which indicates the occurrence of a large amount of motion throughout the video. On the other hand, the image is predominantly blue and green for the stabilized video. Figure 4.11 shows the result obtained with the average grayscale for the same video. It is possible to notice that the stabilized version is better defined, whereas the unstable video image is more blurred. However, it is difficult to infer how much motion is present in the video from the image.

(a) original video (b) stabilized video

Figure 4.11: Average grayscale image for video #4.

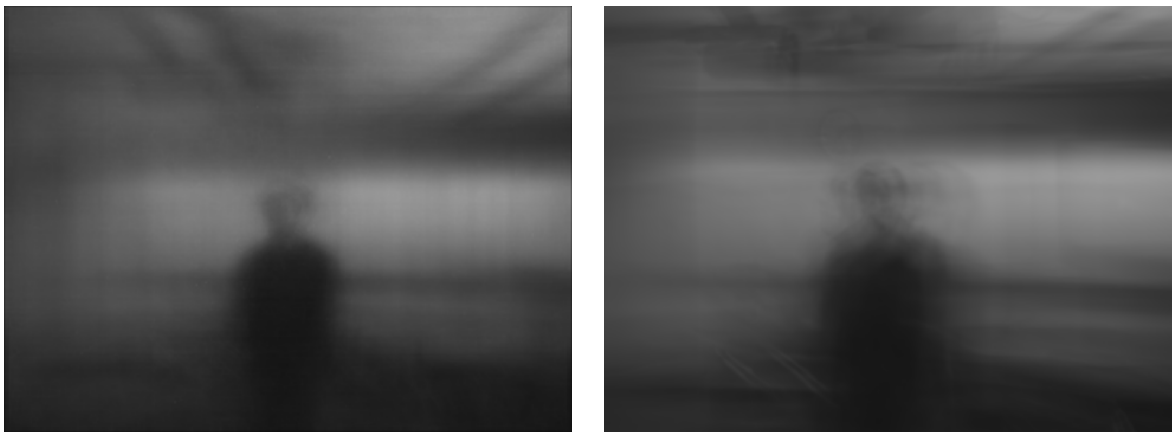
The drawback of the average grayscale image becomes even clearer in the comparison of the results obtained for the video #7. Figures 4.13 and 4.14 show the results of the average grayscale and the average of the MEIs for video #7. From the gray level image,



(a) original video (b) stabilized video

Figure 4.12: Average image of the colored MEIs for video #4.

it is not so easy to differentiate the unstable video from the stabilized one. In fact, the stabilized video seems to have more motion. On the other hand, the stabilized video presents an average MEI image with bluer tones, correctly indicating a smaller amount of motion.



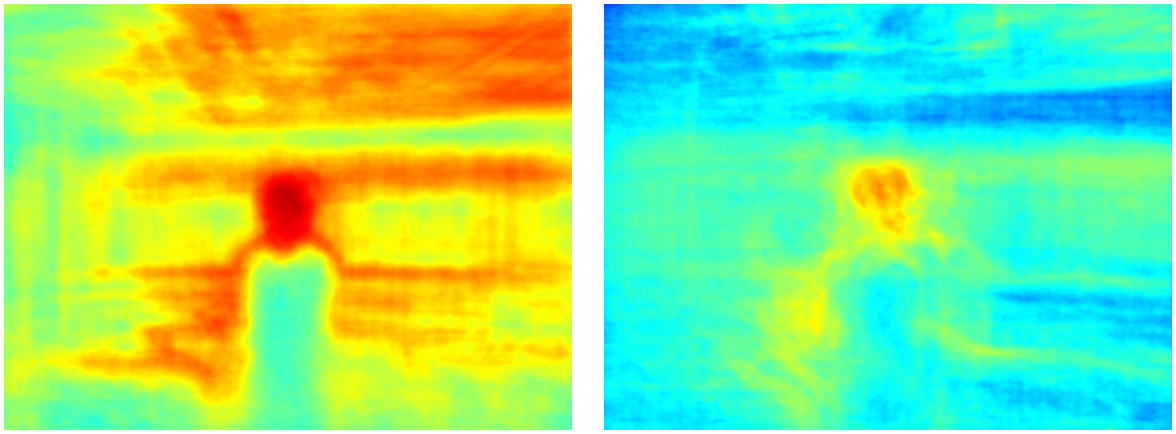
(a) original video (b) stabilized video

Figure 4.13: Average grayscale image for video #7.

The visual representation proposed in this work is efficient to show the amount of motion present in a video, making possible the evaluation and comparison of different stabilization methods. Our technique is more effective than the simple average of the gray levels of the video frames, which can generate inaccurate results when considering the intentional motion of the camera and small changes in the scene.

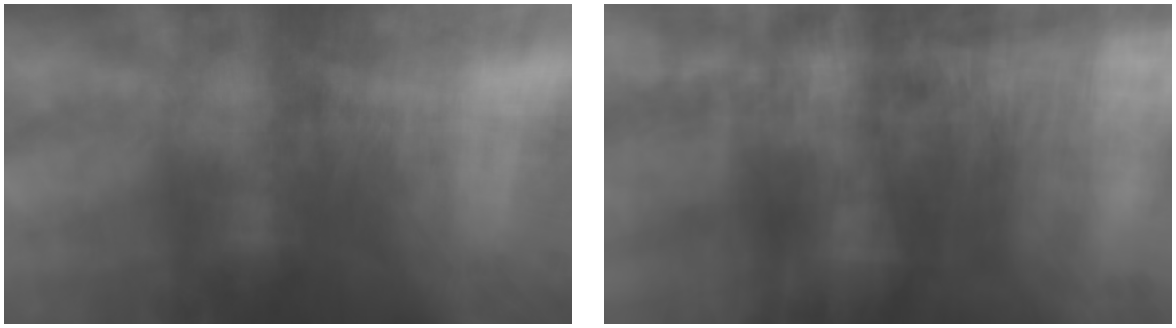
Figures 4.15 and 4.16 show the results obtained for the average grayscale image and the proposed visual representation in a video for a crowded scene.

From Figures 4.15 and 4.16, we can see the differences between the image versions in the proposed visual representation, before and after the video stabilization. Even after stabilization, we can notice red color in the result, which is probably due to the presence of moving people in the scene. However, stronger tones of red are featured in the unstable version of the video, which characterizes a video in the presence of much motion. The images of the average grayscale, however, show little difference, demonstrating the



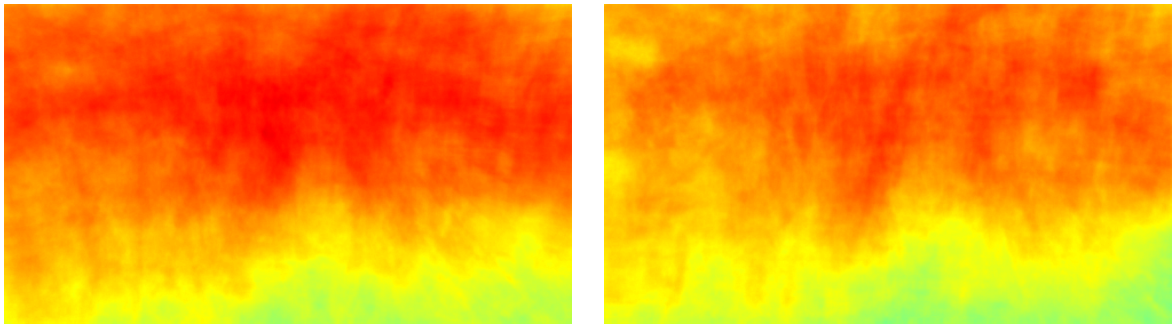
(a) original video (b) stabilized video

Figure 4.14: Average image of the colored MEIs for video `#7`.



(a) original video (b) stabilized video

Figure 4.15: Average grayscale image for video $\texttt{Crowd}_0$.



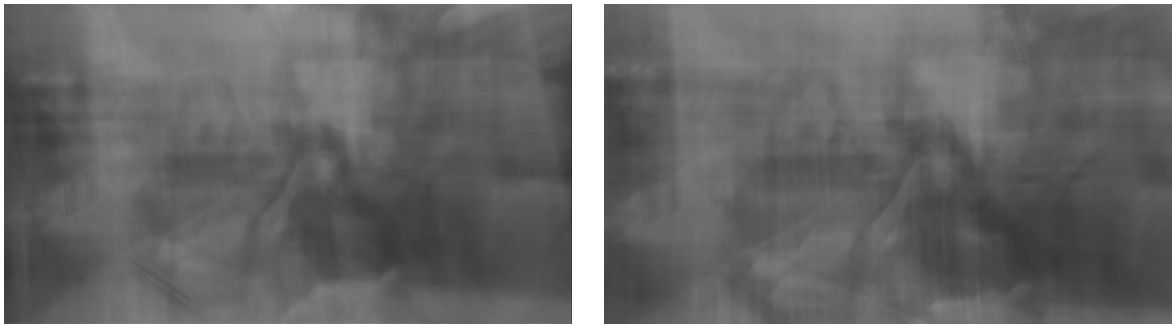
(a) original video (b) stabilized video

Figure 4.16: Average image of the colored MEIs for video $\texttt{Crowd}_2$.

superiority of our visual representation.

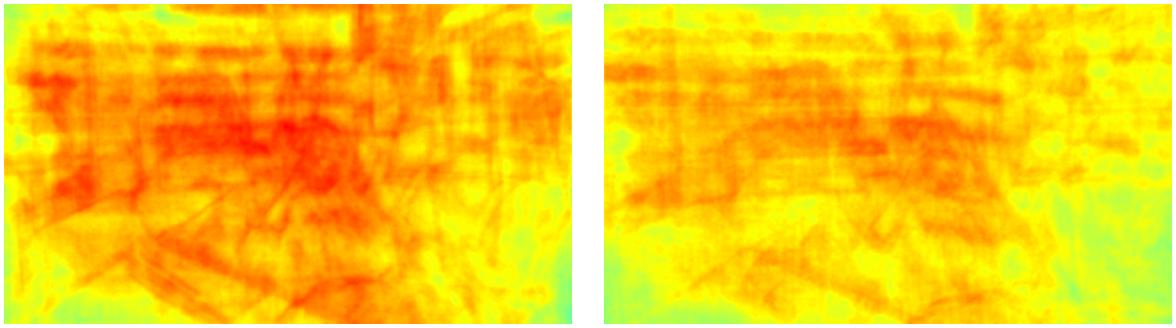
Figures 4.17 and 4.18 show the results obtained for the average grayscale image and the proposed visual representation in a video that contains parallax effect.

From Figures 4.17 and 4.18, we can observe that redder tones were obtained in the unstable video version, whereas the image of the average grayscale presents little distinction between the two versions of the video.



(a) original video (b) stabilized video

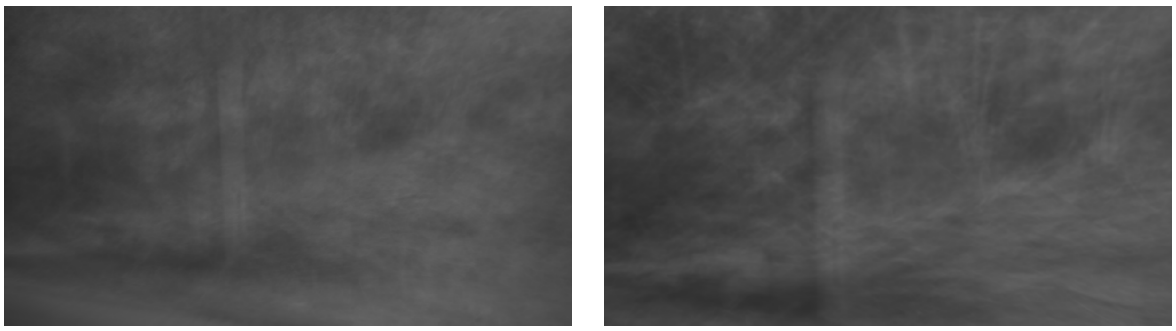
Figure 4.17: Average grayscale image for video $\texttt{Parallax}_0$.



(a) original video (b) stabilized video

Figure 4.18: Average image of the colored MEIs for video $\texttt{Parallax}_0$.

Figures 4.19 and 4.20 illustrate the results obtained for the average grayscale image and the proposed visual representation in a video with fast translations.

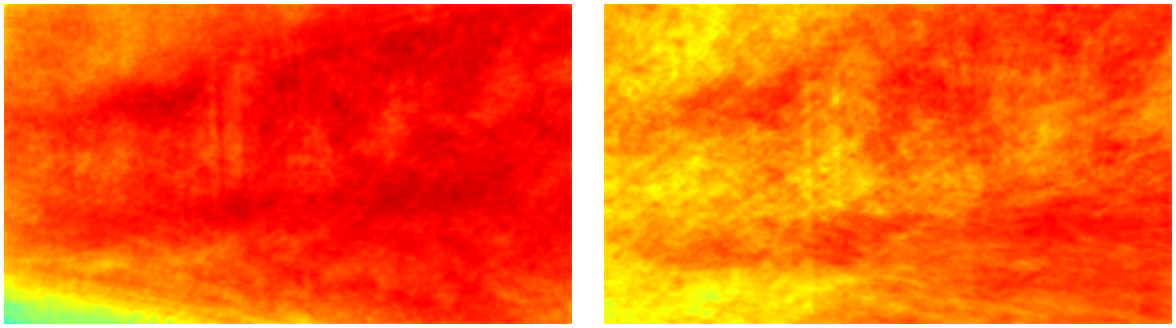


(a) original video (b) stabilized video

Figure 4.19: Average grayscale image of the average for video $\texttt{QuickRotation}_0$.

From Figures 4.19 and 4.20, we can notice that a video in the presence of fast translations tends to have very red tones. Similarly to other cases, lighter tones are obtained in the stabilized version. After stabilization, the visual representation continues with red tones, since there is still a certain amount of motion desired in the video. Again, the visualization of the average grayscale image is not very effective.

Figures 4.21 and 4.22 present the results obtained for the average grayscale image and



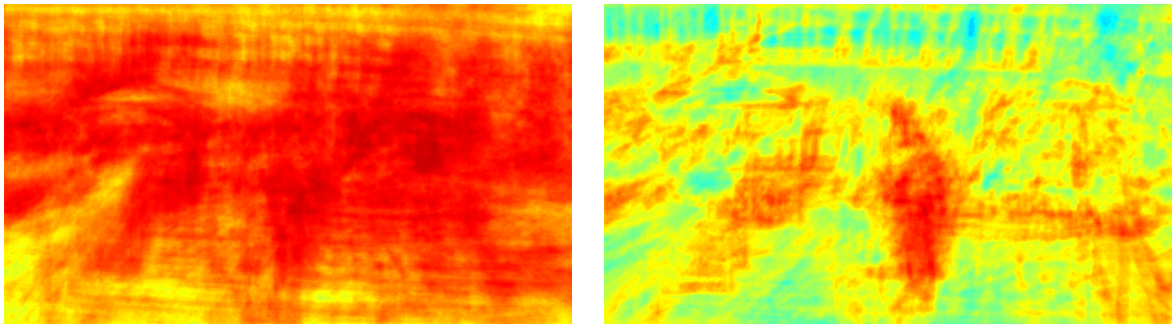
(a) original video (b) stabilized video

Figure 4.20: Average image of the colored MEIs for video $\texttt{QuickRotation}_0$.

the proposed visual representation in a video with regular scene.

(a) original video (b) stabilized video

Figure 4.21: Average grayscale image for video $\texttt{Regular}_0$.

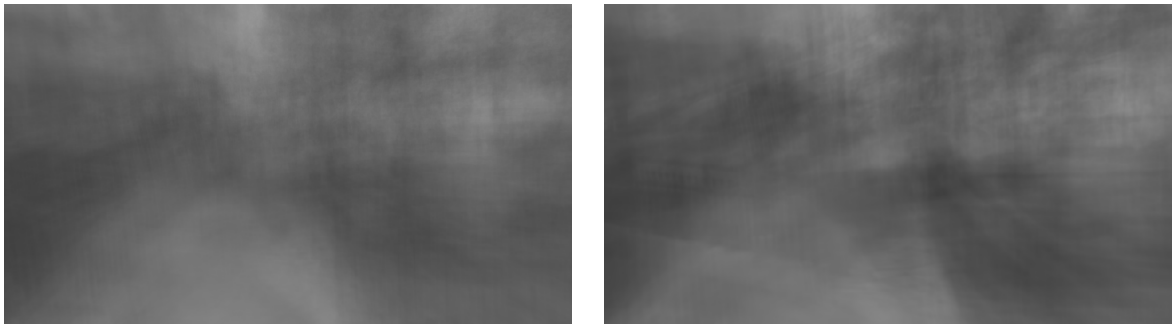


(a) original video (b) stabilized video

Figure 4.22: Average image of the colored MEIs for video $\texttt{Regular}_0$.

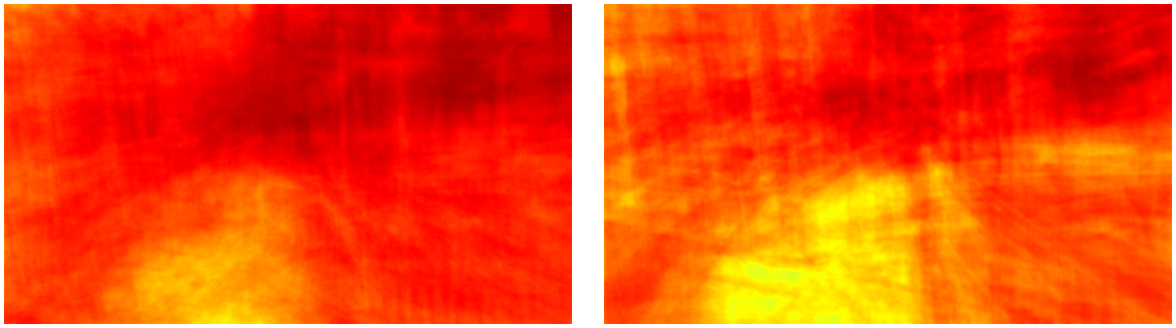
From Figures 4.21 and 4.22, the image for the stabilized version has considerably lighter colors, once this a simple scene and has less movement. We can also notice that redder tones are present in the region where a person is moving.

Figures 4.23 and 4.24 show the results obtained for the average grayscale image and the proposed visual representation, where the person shooting the video was running at the time of scene acquisition.



(a) original video (b) stabilized video

Figure 4.23: Average grayscale image for video $\texttt{Running}_0$.

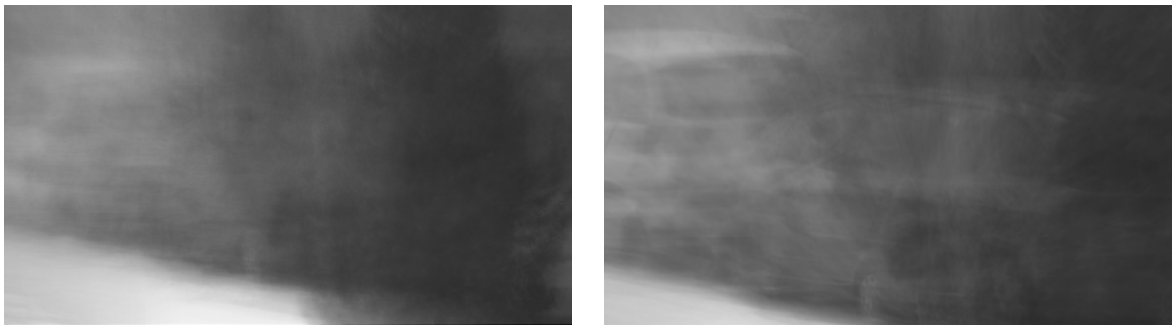


(a) original video (b) stabilized video

Figure 4.24: Average image of the colored MEIs for video $\texttt{Running}_0$.

From Figures 4.23 and 4.24, we can observe that the image tones are very reddish in both versions. This occurs due to the substantial change in the scene and to the motion caused by the person who shoots the video. Notwithstanding, we can notice lighter tones in the stabilized version.

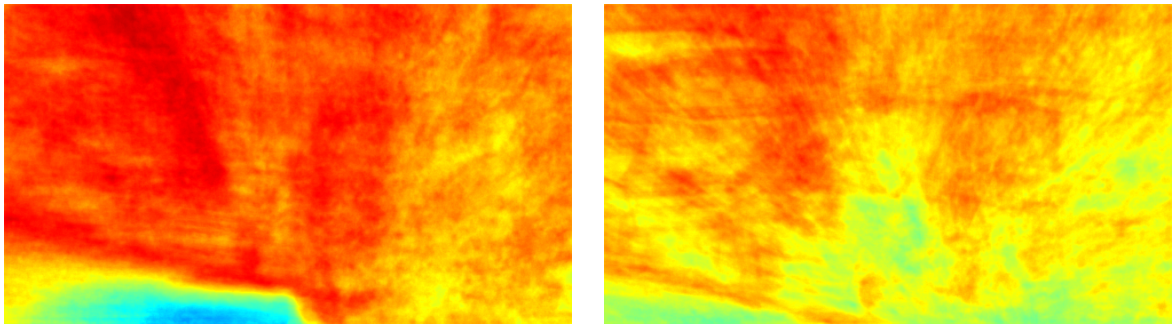
Figures 4.25 and 4.26 present the results obtained for the average grayscale image and the proposed visual representation in a video in the presence of zoom.



(a) original video (b) stabilized video

Figure 4.25: Average grayscale image for video $\texttt{Zooming}_0$.

From Figures 4.25 and 4.26, it is possible to observe that the stabilized version has lighter tones, which demonstrates the advantages of our method.



(a) original video (b) stabilized video

Figure 4.26: Average image of the colored MEIs for video $\texttt{Zooming}_0$.

4.3 Motion Estimation

In this section, we describe the results obtained in the motion estimation step. Tables 4.4 and 4.5 show the PSNR and SSIM average values obtained through different local features for the videos presented in Table 4.1. In these experiments, we consider only a single application of RANSAC. SIFT, SURF and ORB features are described by their own methods, whereas the other ones are described by Fast Retina Keypoint (FREAK) [3]. Tables 4.6 and 4.7 present the results obtained with the dual application of RANSAC, as described in Chapter 3. This experiment was performed to verify the performance of different local features methods in the motion estimation step. In addition, it checks the impact of using RANSAC with one and two passes.

Table 4.4: PSNR for different local features in motion estimation with one-pass RANSAC.

Video	Original	BRISK	FAST	GFTT	HARRIS	MSER	ORB	SIFT	STAR	SURF
1	18.792	16.304	29.961	30.349	30.418	28.961	30.133	29.787	29.306	30.358
2	20.390	19.532	29.571	29.805	29.815	27.359	29.819	29.571	28.876	30.275
3	16.186	14.587	24.654	25.629	25.756	22.580	25.164	22.902	24.096	25.651
4	19.965	21.728	32.160	32.620	32.684	27.389	33.073	32.245	30.893	33.214
5	23.276	18.713	31.564	32.769	30.705	32.563	33.304	33.207	32.445	33.609
6	19.680	16.044	29.313	29.615	29.761	23.513	28.478	29.751	25.760	29.373
7	24.108	22.958	29.392	29.627	29.231	26.153	29.196	29.256	25.270	30.082
8	17.880	17.202	28.180	28.469	28.514	26.912	28.234	27.830	27.532	28.351
9	19.248	18.070	24.281	24.840	24.676	23.222	25.244	24.790	23.522	25.859
10	12.971	14.444	18.861	19.057	19.098	18.287	18.945	18.856	18.791	18.872
11	21.487	19.599	28.466	28.623	28.768	27.878	28.682	28.526	27.492	28.325
12	15.081	17.439	23.676	24.052	24.034	22.962	23.933	23.467	22.958	23.934
13	23.840	24.059	27.746	27.673	30.278	28.048	30.412	26.378	28.639	28.394
14	18.064	18.395	26.288	26.836	26.926	25.839	26.113	27.075	24.410	26.210
Average	19.355	18.505	27.437	27.855	27.905	25.833	27.909	27.403	26.428	**28.036**

From the results, we can see that the SURF method obtains the best results for most of the cases, always presenting the highest average values. In addition, the two-pass RANSAC strategy was effective, obtaining higher values and evidencing the superiority of the SURF method in comparison to the others. In addition to the excellent results

Table 4.5: SSIM for different local features in motion estimation with one-pass RANSAC.

Video	Original	BRISK	FAST	GFTT	HARRIS	MSER	ORB	SIFT	STAR	SURF
1	0.467	0.333	0.944	0.948	0.949	0.928	0.946	0.941	0.934	0.949
2	0.570	0.548	0.925	0.926	0.927	0.872	0.928	0.925	0.911	0.939
3	0.333	0.231	0.828	0.875	0.878	0.765	0.858	0.760	0.818	0.875
4	0.673	0.734	0.957	0.961	0.962	0.880	0.965	0.957	0.942	0.965
5	0.714	0.539	0.923	0.939	0.893	0.944	0.954	0.953	0.943	0.960
6	0.554	0.307	0.946	0.953	0.956	0.753	0.929	0.950	0.841	0.950
7	0.718	0.687	0.896	0.904	0.888	0.787	0.882	0.894	0.752	0.921
8	0.485	0.461	0.922	0.923	0.922	0.889	0.915	0.914	0.908	0.923
9	0.441	0.413	0.753	0.780	0.767	0.687	0.785	0.777	0.692	0.834
10	0.294	0.358	0.649	0.653	0.653	0.606	0.633	0.649	0.641	0.650
11	0.688	0.617	0.886	0.888	0.893	0.875	0.889	0.886	0.872	0.879
12	0.317	0.425	0.755	0.760	0.755	0.718	0.747	0.738	0.722	0.760
13	0.642	0.658	0.773	0.769	0.815	0.770	0.818	0.737	0.784	0.784
14	0.657	0.640	0.917	0.929	0.930	0.899	0.907	0.934	0.862	0.921
Average	0.540	0.496	0.862	0.871	0.870	0.812	0.868	0.858	0.830	0.879

Table 4.6: PSNR for different local features in motion estimation with two-pass RANSAC.

Video	Original	BRISK	FAST	GFTT	HARRIS	MSER	ORB	SIFT	STAR	SURF
1	18.792	28.334	30.303	29.773	29.637	29.537	29.696	30.378	29.783	30.593
2	20.390	28.941	30.246	30.163	30.095	28.198	29.771	29.863	29.300	30.418
3	16.186	24.689	26.104	25.979	25.973	23.382	25.644	24.853	24.925	26.143
4	19.965	30.783	33.411	33.163	33.170	27.984	32.667	32.678	31.377	33.486
5	23.276	29.779	33.554	33.562	32.775	32.797	33.338	33.531	32.787	33.706
6	19.680	26.153	29.592	29.349	29.179	25.279	28.711	29.893	27.136	30.014
7	24.108	27.928	30.269	30.280	30.047	26.287	29.951	29.520	29.520	30.311
8	17.880	27.482	28.302	28.226	28.101	27.356	27.654	27.806	27.806	28.499
9	19.248	23.257	26.005	26.021	25.788	23.587	25.329	26.156	23.730	26.164
10	12.971	18.966	19.020	18.966	18.959	18.560	18.937	18.972	18.885	19.011
11	21.487	27.847	28.613	28.501	28.418	27.971	28.488	28.621	28.621	28.635
12	15.081	23.793	24.064	24.132	24.106	23.281	23.952	23.761	23.279	24.165
13	23.840	28.397	30.234	30.302	30.029	27.963	30.229	26.558	28.638	30.401
14	18.064	25.897	27.818	27.714	27.689	26.168	27.438	27.205	24.731	27.861
Average	19.355	26.589	28.395	28.295	28.140	26.311	28.022	27.803	26.819	28.529

obtained with SURF, we highlight the considerable gain of the BRISK method by adopting the two-pass strategy. The results obtained are consistent between the two measures used.

4.3.1 Local Combined Features

In this subsection, we present the results obtained with the consensual local feature combination method. In our experiments, we consider three different local features: Maximally Stable Extremal Regions (MSER) [78], Scale-Invariant Feature Transform (SIFT) [71] and STAR [1]. The SIFT features are described in the method itself, whereas the other ones are described with Fast Retina Keypoint (FREAK) [3]. Each method is also evaluated

Table 4.7: SSIM for different local features in motion estimation with two-pass RANSAC.

Video	Original	BRISK	FAST	GFTT	HARRIS	MSER	ORB	SIFT	STAR	SURF
1	0.467	0.913	0.950	0.946	0.945	0.938	0.943	0.948	0.941	0.952
2	0.570	0.914	0.939	0.938	0.938	0.895	0.932	0.929	0.920	0.942
3	0.333	0.845	0.891	0.890	0.890	0.803	0.879	0.843	0.850	0.892
4	0.673	0.945	0.967	0.966	0.966	0.892	0.962	0.961	0.949	0.967
5	0.714	0.899	0.958	0.960	0.944	0.947	0.961	0.959	0.948	0.964
6	0.554	0.859	0.958	0.955	0.953	0.836	0.944	0.955	0.893	0.960
7	0.718	0.853	0.927	0.927	0.916	0.792	0.912	0.901	0.760	0.930
8	0.485	0.909	0.924	0.920	0.918	0.900	0.903	0.920	0.911	0.925
9	0.441	0.686	0.850	0.848	0.834	0.705	0.837	0.805	0.701	0.864
10	0.294	0.651	0.660	0.651	0.648	0.624	0.627	0.654	0.647	0.660
11	0.688	0.876	0.895	0.894	0.893	0.877	0.894	0.887	0.874	0.894
12	0.317	0.753	0.779	0.770	0.765	0.733	0.753	0.750	0.737	0.779
13	0.642	0.780	0.815	0.818	0.816	0.766	0.815	0.741	0.782	0.819
14	0.657	0.910	0.950	0.949	0.949	0.907	0.946	0.936	0.870	0.951
Average	0.540	0.842	0.890	0.887	0.883	0.830	0.879	0.871	0.842	**0.892**

individually with the step of removing outliers performed by RANSAC. These methods were used because they obtained good results in preliminary experiments. The aim of this experiment is to show that the combined use of local features methods can improve the motion estimation of isolated methods when these methods do not make a correct estimation.

Initially, we evaluated the results only for the motion estimation step. This is done in order to remove the influence of the later steps. In this evaluation, we computed the mean of the PSNR and SSIM values frame by frame, considering the similarity matrix obtained shortly after the estimation step. In order for the extracted metrics to be coherent, the transformed frames are cropped by taking into account only the similarity matrix of that frame.

Tables 4.8 and 4.9 present the results obtained, in which we can see a better combination performance for most videos for both PSNR and SSIM. In average values where the combination is not the best result, it tends to be close to the best. For example, in video `#7` the combination has a better result than the MSER and STAR methods, whereas in video `#13` the result is greater than MSER and SIFT. These results show that the application of the combination strategy obtains a greater robustness in the motion estimation.

Although the results obtained after the smoothing confirm the results obtained in the assessment of motion estimation, this evaluation is important because it provides us with more precise information about the motion estimation step, once applying different methods in the smoothing step may change what is considered as unwanted motion.

Tables 4.10 and 4.11 present the results obtained for the dataset presented in Table 4.2. In Table 4.10, it can be noted that the combination always provides higher values in terms of average PSNR in comparison to the other detection methods. From Table 4.11, it is possible to observe that the PSNR results are consistent with the SSIM results, confirming the superiority in motion estimation with the use of the combination.

Table 4.8: PSNR for different local features in motion estimation.

Video	Original	MSER	SIFT	STAR	Combination
1	18.792	29.537	30.378	29.783	30.513
2	20.390	28.198	29.863	29.300	29.993
3	16.186	23.382	24.853	24.925	25.772
4	19.965	27.984	32.678	31.377	32.985
5	23.276	32.797	33.531	32.787	33.726
6	19.680	25.279	29.893	27.136	29.988
7	24.108	26.287	29.520	25.486	28.801
8	17.880	27.356	28.115	27.806	28.249
9	19.248	23.587	25.329	23.730	25.463
10	12.971	18.560	18.937	18.885	19.016
11	21.487	27.971	28.621	27.600	28.424
12	15.081	23.281	23.761	23.279	23.922
13	23.840	27.963	26.558	28.638	28.516
14	18.064	26.168	27.205	24.731	26.987
Average	19.355	26.311	27.803	26.819	28.025

Table 4.9: SSIM for different local features in motion estimation.

Video	Original	MSER	SIFT	STAR	Combination
1	0.467	0.938	0.948	0.941	0.950
2	0.570	0.895	0.929	0.920	0.931
3	0.333	0.803	0.843	0.850	0.880
4	0.673	0.892	0.961	0.949	0.964
5	0.714	0.947	0.959	0.948	0.963
6	0.554	0.836	0.955	0.893	0.959
7	0.718	0.792	0.901	0.760	0.874
8	0.485	0.900	0.920	0.911	0.920
9	0.441	0.705	0.805	0.701	0.807
10	0.294	0.624	0.654	0.647	0.658
11	0.688	0.877	0.887	0.874	0.885
12	0.317	0.733	0.750	0.737	0.759
13	0.642	0.766	0.741	0.782	0.783
14	0.657	0.907	0.936	0.870	0.930
Average	0.540	0.830	0.871	0.842	0.876

Figures 4.27 and 4.28 present the percentage gain for PSNR and SSIM metrics. It is possible to notice that the combination performs better in the `Regular` and `Zooming` categories. In addition, it can be seen that the percentage gain can vary according to the metrics adopted.

Table 4.10: PSNR for different local features in motion estimation.

Category	Original	MSER	SIFT	STAR	Combination
Crowd	19.479	25.337	25.870	25.781	26.256
Parallax	18.746	22.566	23.217	23.388	23.657
Quick Rotation	19.963	25.805	27.465	26.694	28.020
Regular	19.468	24.162	26.605	26.290	27.259
Running	17.326	23.272	24.475	24.399	24.641
Zooming	20.113	25.625	27.916	27.503	28.776
Average	19.180	24.461	25.925	25.675	26.435

Table 4.11: SSIM for different local features in motion estimation.

Category	Original	MSER	SIFT	STAR	Combination
Crowd	0.602	0.842	0.860	0.857	0.872
Parallax	0.562	0.737	0.762	0.764	0.776
Quick Rotation	0.568	0.789	0.852	0.825	0.866
Regular	0.500	0.695	0.813	0.809	0.846
Running	0.434	0.734	0.786	0.782	0.788
Zooming	0.535	0.779	0.860	0.847	0.889
Average	0.534	0.763	0.822	0.814	0.840

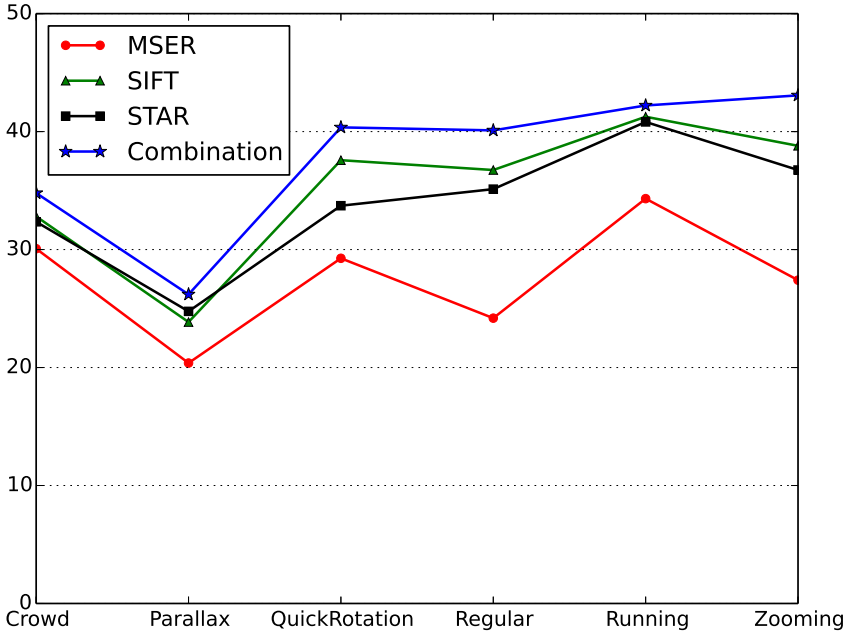


Figure 4.27: Average PSNR gain. (%).

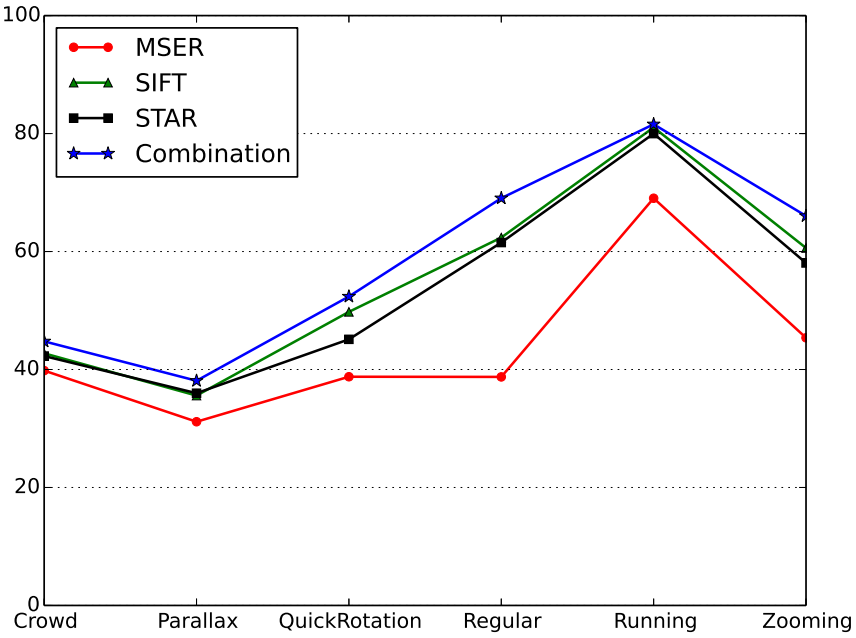


Figure 4.28: Average SSIM gain. (%).

The gains obtained in each video using the combination were relatively low. However, there are some videos that the gain obtained is above average. We highlight one of these videos, the video `#11` of the `Zooming` category, called Zooming_{11}. Table 4.12 presents the PSNR and SSIM for three frames of the Zooming_{11}, in addition to the average of all frames.

Table 4.12: PSNR and SSIM for frames of video Zooming_{11}.

# Frame	MSER		SIFT		STAR		Combination	
	PSNR	**SSIM**	**PSNR**	**SSIM**	**PSNR**	**SSIM**	**PSNR**	**SSIM**
73	18.812	0.501	17.152	0.358	16.716	0.302	18.663	0.481
364	13.719	0.049	18.265	0.525	13.250	0.034	22.135	0.803
399	13.435	0.092	14.740	0.134	15.371	0.231	20.000	0.700
Overall	15.598	0.230	18.646	0.464	19.054	0.494	20.831	0.642

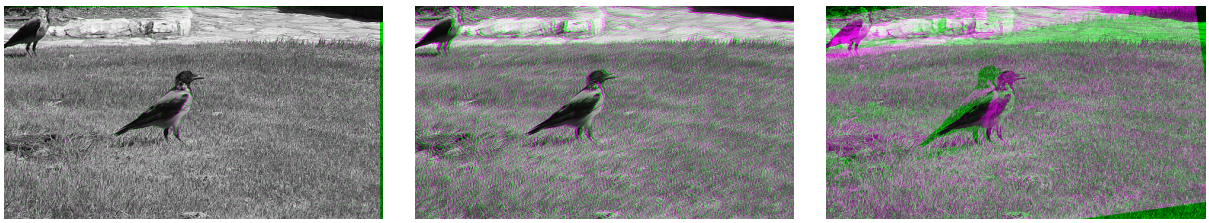
We can see in Table 4.12 that the performance of the combination is related to the best individual method, and in some cases may perform considerably better than all the methods used. Due to the success of the combination, the mean value for both metrics is considerably higher in the Zooming_{11}.

Figures 4.29, 4.30 and 4.31 present the three frames in which the frames are transformed and overlapped to the next frame. It is possible to note, in all cases, that the overlap obtained in the combination is more accurate.

We present only this subset of local features, which is the one that obtained the largest

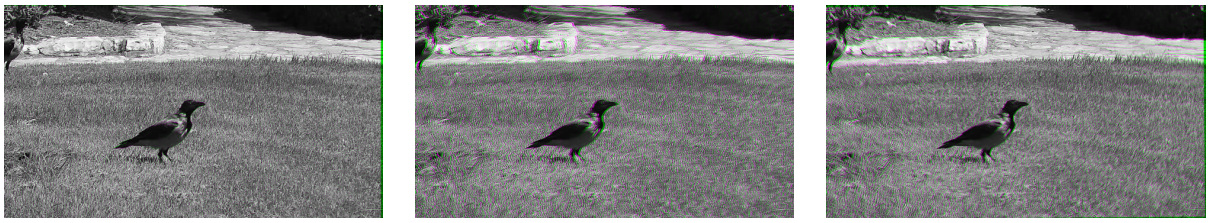
(a) Combination (b) SIFT (c) STAR

Figure 4.29: Comparison among the methods for frame #73.



(a) Combination (b) MSER (c) STAR

Figure 4.30: Comparison among the methods for frame #364.



(a) Combination (b) MSER (c) SIFT

Figure 4.31: Comparison among the methods for frame #399.

gain in comparison to the individual methods. We believe that this improvement occurs because we are combining methods with inferior results. Thus, the combination of weaker methods produces superior results. In addition, we notice that the results obtained with the combination are lower than previously shown, for instance, with the SURF method. This occurs because the results achieved with SURF for these two datasets are already very good, correctly estimating the global motion between two frames in practically all cases.

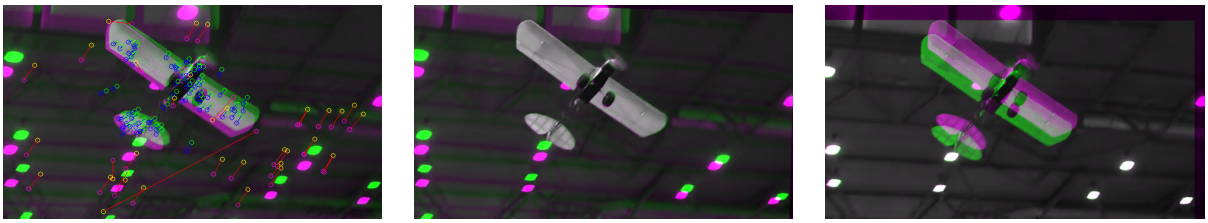
4.3.2 Spatio-Temporal Optimization

In order to avoid an exhaustive analysis, we report the results obtained only for some video portions, since they illustrate the cases that are repeated in the others. Figures 4.32, 4.33 and 4.34 present some failure situations of local features for different videos, as well as the correction performed with our method. Matches considered as inliers by the RANSAC method are drawn in blue and green, whereas the outlier matches are drawn in pink and yellow. This experiment was done to demonstrate that the spatio-temporal optimization method was able to correct the motion estimation between pairs of frames that had an

incorrect estimation.

(a) matches of local features (b) warped frame based on local features (c) our result

Figure 4.32: Motion estimation for the 11th frame of video $\texttt{ours}_1$.



(a) matches of local features (b) warped frame based on local features (c) our result

Figure 4.33: Motion estimation for the 481th frame of video $\texttt{ours}_2$.

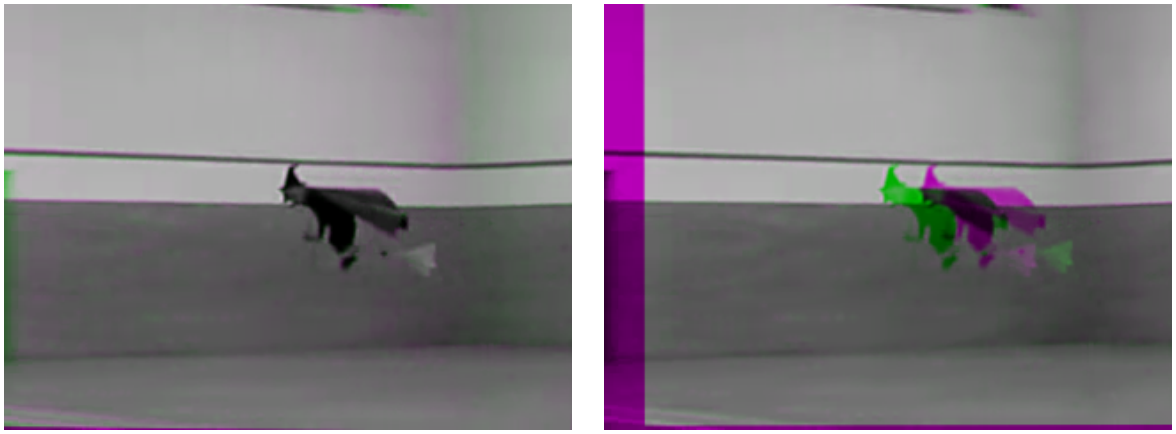
(a) matches of local features (b) warped frame based on local features (c) our result

Figure 4.34: Motion estimation for the 129th frame of video $\texttt{ours}_6$.

In the three cases presented, we can see that the matches of the objects were considered as inliers, which made the movements of the object, not the camera, compensated. On the other hand, our optimization-based method obtained excellent results, finding the transformation matrix that matches the motion performed by the camera.

Higher values of similarity measures, such as PSNR or SSIM, may indicate a better quality in the motion estimation for most cases. However, there are cases where such measures do not indicate the correct estimate and, therefore, a simple optimization that takes the measures into account would not be efficient. Figure 4.35 presents different matches for the same pair of frames, where different values of PSNR and SSIM are obtained. It can be observed that higher values are obtained in incorrect cases. Since background is

unrepresentative, higher similarity is obtained if object matching is done. However, this is semantically incorrect since the object is in motion.



(a) PSNR = 30.576 and SSIM = 0.932 (b) PSNR = 23.491 and SSIM = 0.896

Figure 4.35: Different matches for the 40th frame of video $\texttt{ours}_7$.

4.4 Removal of the Unwanted Motion

In this section, we describe the results obtained in the removal of the unwanted motion. In the experiments performed, we compared the values of the ITF metric, as well as the amount of pixels held for different versions of the trajectory smoothing. In addition, we present the visual rhythm for some cases.

In the first version, we used the Gaussian filter considering $\sigma = 40$. In another version, the Gaussian filter is used in a slightly more adaptive way, choosing different values of σ for each trajectory according to the size of the trajectory range with respect to the size of the video frame. Higher values of σ are assigned to paths with smaller intervals; we denote this version as semi-adaptive. The adaptive version of the Gaussian filter proposed in this work is presented. For all these versions, we use the SURF method with the two-pass RANSAC in the motion estimation. The SURF was used in this experiment because it obtained the motion estimation correctly for all videos we considered. In this way, the motion estimation step does not interfere with the evaluation of this step. In addition, the videos were submitted to the `YouTube` stabilization method [36] in order to compare its results against ours.

The following experiments were performed to verify and compare the performance of the Gaussian filter and the Kalman filter in the removal of unwanted motion. The ITF and ITF_{SSIM} metric are calculated for the video sequence before and after the stabilization process. Tables 4.13, 4.14 and 4.15 shows the results obtained with the Kalman filter and the Gaussian filter with $\sigma = 40$.

We can observe a certain superiority in the use of the Gaussian filter, which achieves a higher ITF and ITF_{SSIM} value for all videos with basically the same amount of pixels kept for most videos. Videos #5, #9, #10, #12 and #14 keep a lower amount of pixels compared to the other videos. This is due to the presence of desired camera motion, which

Table 4.13: Comparison of ITF values between Gaussian filter and Kalman filter.

Video	**Original**	**Gaussian Filter**	**Kalman Filter**
1	18.793	27.738	25.888
2	20.390	29.331	27.201
3	16.186	22.559	22.122
4	19.965	33.380	26.298
5	23.277	28.660	25.991
6	19.681	29.804	25.576
7	24.109	28.510	28.063
8	17.881	25.448	24.081
9	19.248	23.251	21.426
10	12.972	18.453	16.680
11	21.487	26.826	25.704
12	15.081	-	-
13	23.841	30.621	28.200
14	18.065	20.265	-
Average	19.355	26.526	24.769

Table 4.14: Comparison of ITF_{SSIM} values between Gaussian filter and Kalman filter.

Video	**Original**	**Gaussian Filter**	**Kalman Filter**
1	0.468	0.903	0.846
2	0.571	0.937	0.894
3	0.334	0.819	0.773
4	0.673	0.962	0.844
5	0.715	0.862	0.794
6	0.555	0.950	0.864
7	0.719	0.908	0.895
8	0.486	0.884	0.840
9	0.442	0.652	0.605
10	0.295	0.557	0.528
11	0.689	0.847	0.827
12	0.317	-	-
13	0.643	0.844	0.800
14	0.657	0.777	-
Average	0.540	0.839	0.793

is erroneously considered as oscillations by the Gaussian filter, if the value of σ used is high enough. However, smaller values may not remove the oscillations from the videos efficiently, since each video has oscillations of different proportions.

When we analyze the visual rhythms of the videos, we notice that the Gaussian filter

Table 4.15: Comparison of hold pixels (%) between Gaussian filter and Kalman filter.

Video	**Gaussian Filter**	**Kalman Filter**
1	69.276	71.000
2	71.750	74.771
3	72.972	73.003
4	48.958	54.903
5	2.540	4.958
6	67.891	73.507
7	60.495	57.167
8	70.648	72.287
9	25.797	33.818
10	17.519	27.204
11	43.599	52.875
12	0	0
13	70.312	71.875
14	7.448	0
Average	44.943	47.669

obtains softer rhythms than the Kalman filter for all fourteen videos, confirming the results obtained with the measures. However, in addition to not giving us more detailed information about the movement of each video, the values (ITF mainly) do not correctly reflect the difference between the two versions.

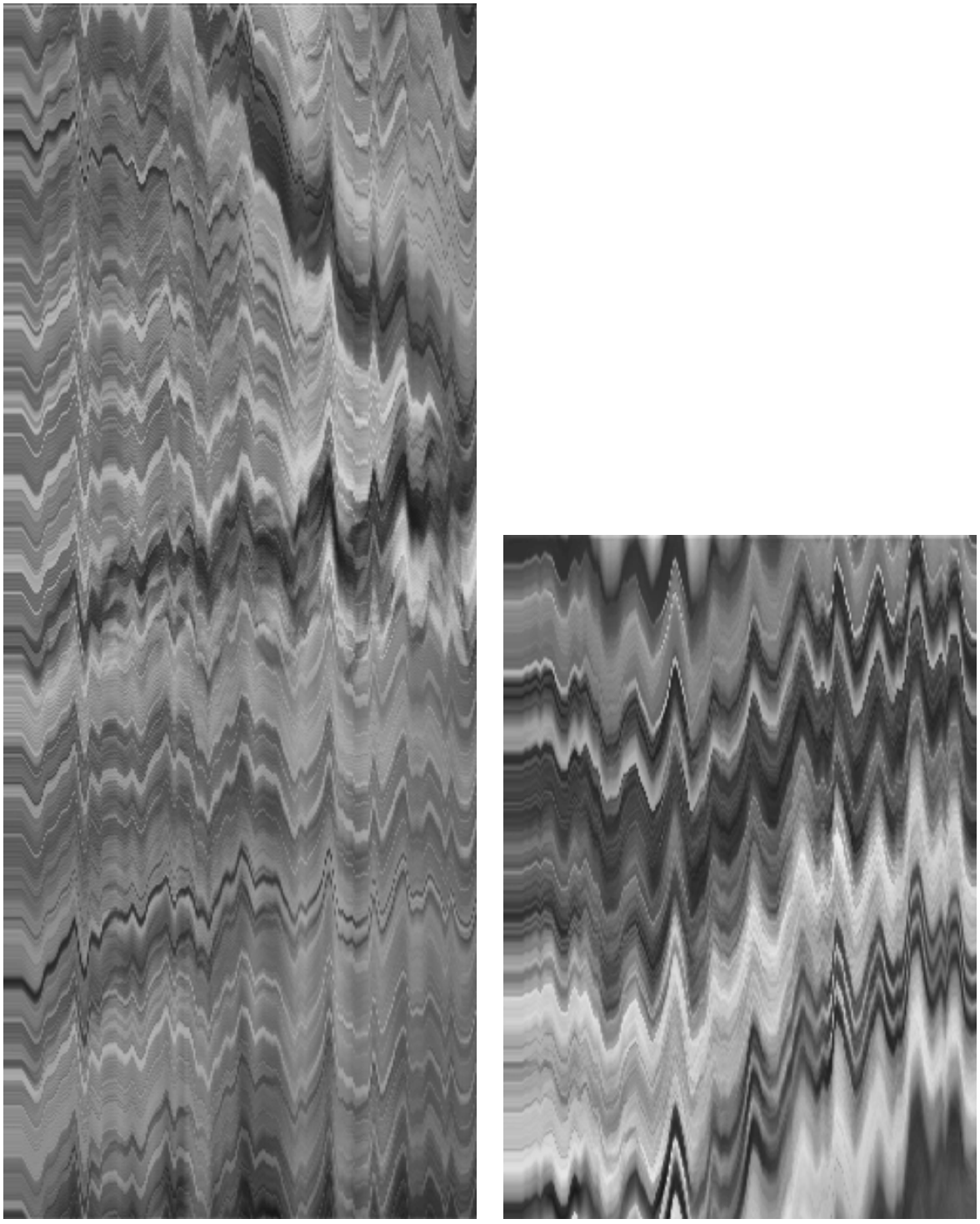
Figures 4.36, 4.37 and 4.38 show the visual rhythms obtained for the video #3 in the original version and after filter stabilization Kalman and Gaussian filter with $\sigma = 40$. Even if both versions obtain similar measure values, we can note that the visual rhythms obtained by the Gaussian filter are considerably smoother than the rhythms obtained by the Kalman filter, which indicates the superiority of the Gaussian filter.

Tables 4.16, 4.17 and 4.18 presents a comparison of the mean of the ITF and ITF_{SSIM} values, besides the mean of the percentage of pixels held in each category, considering the Kalman filter and the Gaussian filter. For all categories, the Gaussian filter obtained higher values. Both versions have pixel percentages kept close by, alternating between the highest category-by-category value.

We conjecture that the Gaussian filter obtained results superior to the Kalman filter because the former does not use only previous values for the estimation, but more complete information.

4.4.1 Adaptive Gaussian Filter

In order to improve the quality of the stabilization for cases where there is a small amount of pixels held. Tables 4.19, 4.20 and 4.21 present the results obtained with the version of the semi-adaptive Gaussian filter, where trajectories with greater difference between the minimum and maximum values will have a lower value for σ. We used $\sigma = 40$ for

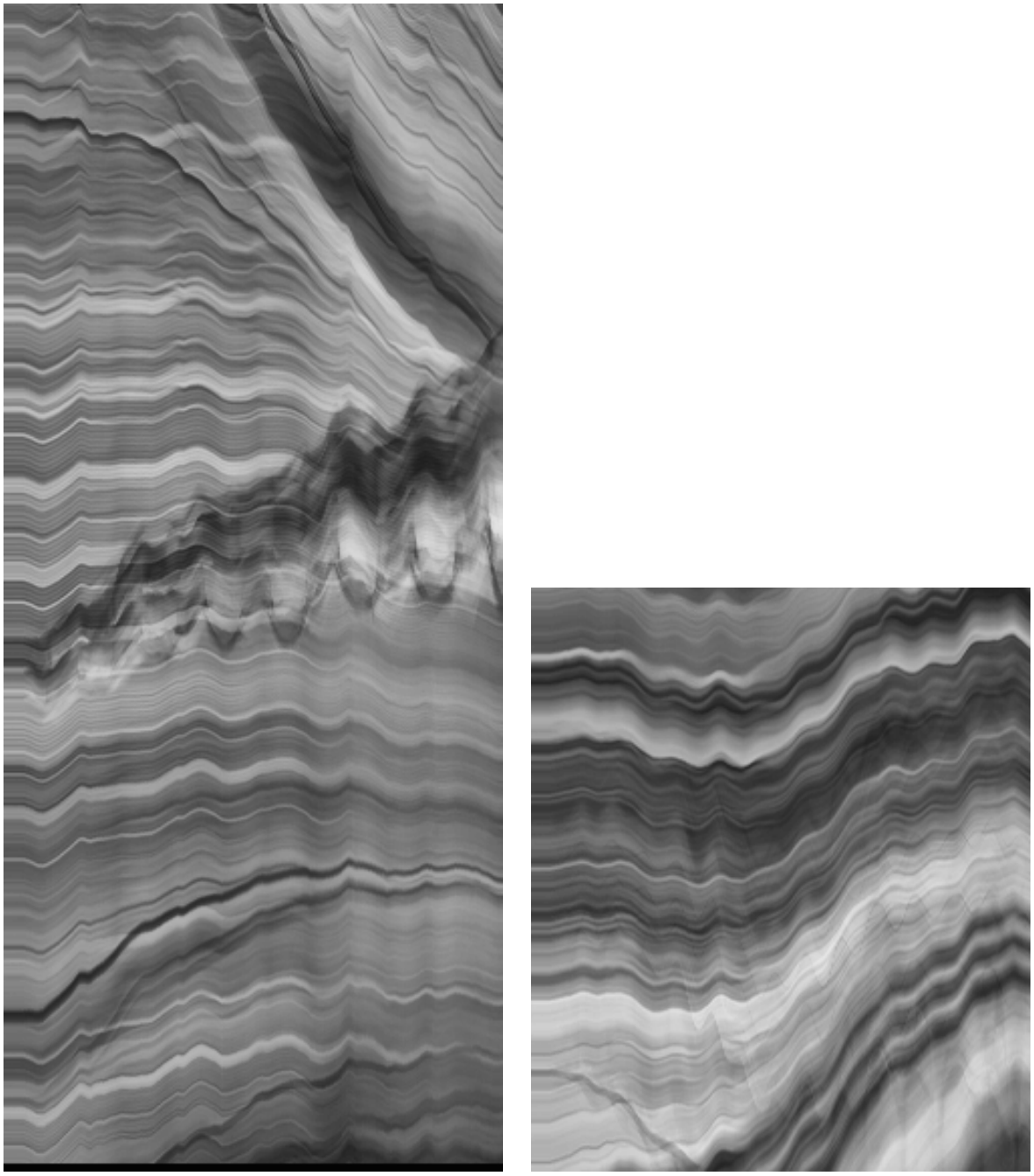


(a) horizontal visual rhythm (b) vertical visual rhythm

Figure 4.36: Visual rhythms for original video #3.

trajectories with intervals smaller than 80% of the respective frame size, whereas $\sigma = 20$ otherwise. For the adaptive version proposed in this work, we experimentally set $r_{\min}$ as 0.4. The purpose of the following experiments is to analyze the performance of semi-adaptive and adaptive Gaussian filters in the removal of unwanted motion and to compare them with the results obtained previously.

The semi-adaptive version maintains more pixels in the videos in which the original

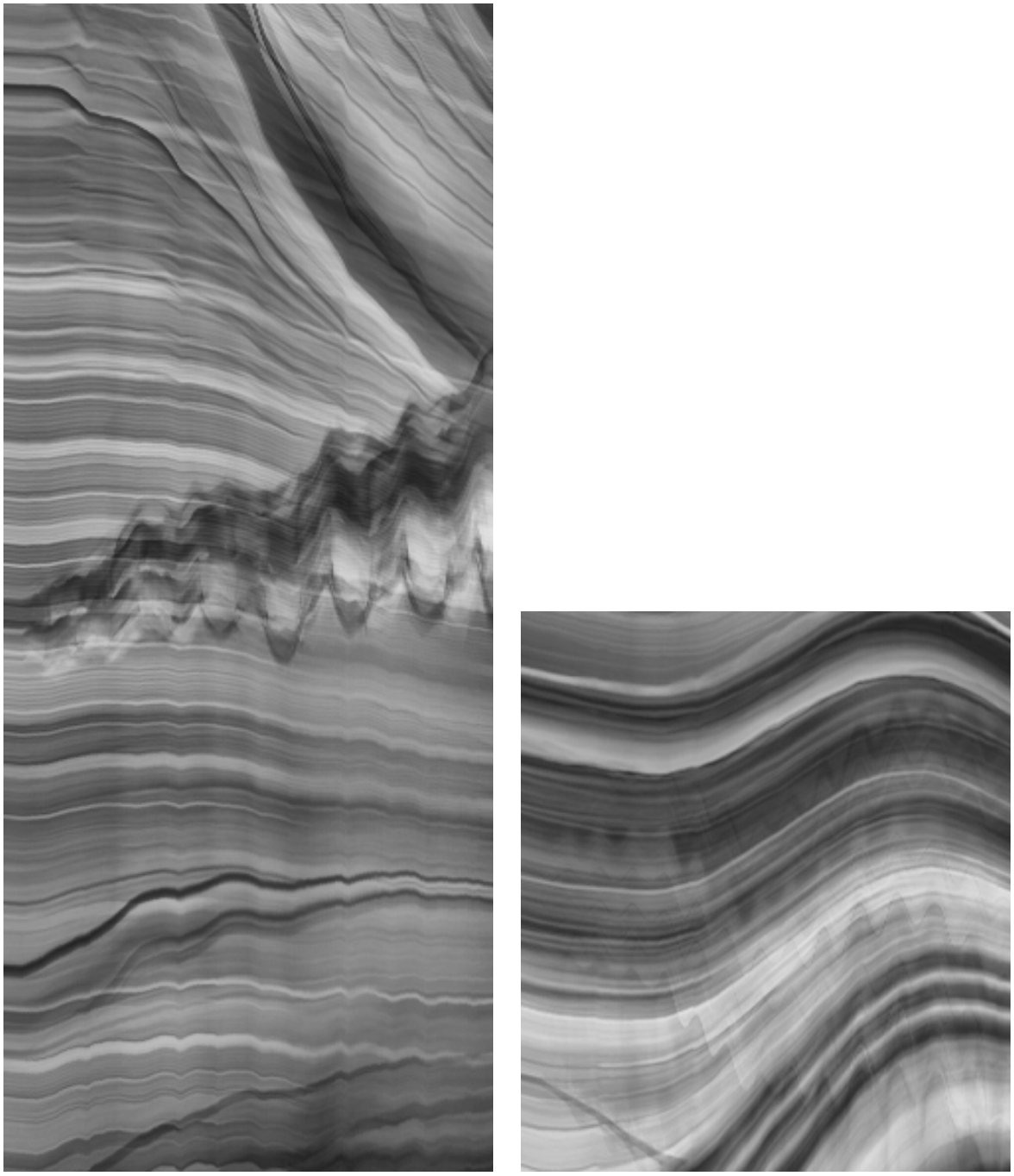


(a) horizontal visual rhythm (b) vertical visual rhythm

Figure 4.37: Visual rhythms for video #3 stabilized by Kalman filter.

Gaussian filter had problems, since $\sigma = 20$ was applied to them. However, the amount of pixels held in the frames is lower than the other videos. This is because, in many cases, $\sigma = 20$ is still a very high value. On the other hand, smaller values for σ can ignore the oscillations that are present in other instants of the video, thus generating videos not stabilized enough and consequently with a lower measure value. Therefore, as shown in the adaptive version, whose smoothing intensity is changed along the trajectory, achieved measure values comparable to the original and semi-adaptive version, maintaining considerably more pixels.

Although the ITF and ITF_{SSIM} values has decreased in the adaptive version, this does



(a) horizontal visual rhythm (b) vertical visual rhythm

Figure 4.38: Visual rhythms for video #3 stabilized by Gaussian filter.

not necessarily represent that the videos are more unstable. When we analyze the visual rhythms for the fourteen videos, we notice that the rhythms have smooth lines, even if slightly different from those obtained by the version of the Gaussian filter with $\sigma = 40$.

Figure 4.39 shows the visual rhythms extracted from the video, from which we can verify the smoothness of the rhythms. Even with video #3 having similar values in the adaptive version, compared to the Kalman filter version. The visual rhythms obtained in the adaptive version are considerably smoother.

Tables 4.22, 4.23 and 4.24 presents the averages of the ITF and ITF_{SSIM} values and the

Table 4.16: Comparison of ITF between Gaussian filter and Kalman filter.

Video	Original	Gaussian Filter	Kalman Filter
Crowd	19.479	23.621	22.218
Parallax	18.746	21.136	20.888
QuickRotation	19.964	23.115	-
Regular	19.457	25.135	23.958
Running	17.327	23.123	21.942
Zooming	20.113	22.076	21.964
Average	19.181	23.034	22.194

Table 4.17: Comparison of ITF_{SSIM} between Gaussian filter and Kalman filter.

Video	Original	Gaussian Filter	Kalman Filter
Crowd	0.603	0.780	0.727
Parallax	0.562	0.667	0.656
Quick Rotation	0.569	0.519	-
Regular	0.501	0.782	0.745
Running	0.435	0.726	0.666
Zooming	0.536	0.580	0.507
Average	0.534	0.676	0.660

Table 4.18: Comparison of hold pixels (%) between Gaussian filter and Kalman filter.

Video	Gaussian Filter	Kalman Filter
Crowd	37.200	41.702
Parallax	44.609	39.523
QuickRotation	1.829	0.000
Regular	64.023	64.370
Running	31.044	32.392
Zooming	17.911	15.470
Average	32.769	32.243

percentage of pixels held with the semi-adaptive Gaussian filter and the adaptive Gaussian filter. It can be noticed that the semi-adaptive version can maintain a higher percentage of pixels when compared to the Gaussian filter with $\sigma = 40$. However, a significant increase is obtained only with the adaptive version. The gain in the percentage of pixels held can be seen in all categories and is even more significant in the `QuickRotation`, `Zooming` and `Running` categories.

Tables 4.25 and 4.26 present a comparison of the results between our method and YouTube approach [36]. The percentage of pixels held was not reported since the YouTube method resizes the stabilized videos to their original size. Thus, a qualitative analysis is

Table 4.19: Comparison of ITF values between semi-adaptive Gaussian filter and adaptive Gaussian filter.

# Video	Original	Semi-Adaptive Gaussian	Adaptive Gaussian
1	18.793	27.620	27.455
2	20.390	29.331	28.914
3	16.186	22.559	22.090
4	19.965	33.380	27.931
5	23.277	27.814	27.360
6	19.681	29.804	29.077
7	24.109	28.510	28.876
8	17.881	25.448	25.182
9	19.248	21.845	21.435
10	12.972	17.465	16.381
11	21.487	26.826	25.659
12	15.081	19.827	17.895
13	23.841	30.621	29.987
14	18.065	19.759	19.773
Average	19.355	25.772	24.858

Table 4.20: Comparison of ITF_{SSIM} between semi-adaptive Gaussian filter and adaptive Gaussian filter.

# Video	Original	Semi-Adaptive Gaussian	Adaptive Gaussian
1	0.468	0.899	0.896
2	0.571	0.937	0.931
3	0.334	0.819	0.789
4	0.673	0.962	0.884
5	0.715	0.857	0.842
6	0.555	0.950	0.941
7	0.719	0.908	0.908
8	0.486	0.884	0.877
9	0.442	0.603	0.587
10	0.295	0.537	0.470
11	0.689	0.847	0.835
12	0.317	0.515	0.465
13	0.643	0.844	0.834
14	0.657	0.782	0.778
Average	0.540	0.810	0.788

done through the first frame of each video, whose results are classified into three categories: superior (when our method maintains more pixels), inferior (when the YouTube method

Table 4.21: Comparison of hold pixels (%) between semi-adaptive Gaussian filter and adaptive Gaussian filter.

# Video	Semi-Adaptive Gaussian	Adaptive Gaussian
1	70.745	74.500
2	71.750	75.781
3	72.972	76.056
4	48.958	62.465
5	8.312	53.385
6	67.891	70.838
7	60.495	73.667
8	70.648	73.284
9	35.750	57.139
10	27.907	70.296
11	43.559	57.260
12	16.611	59.847
13	70.312	71.719
14	39.045	54.146
Average	50.353	66.455

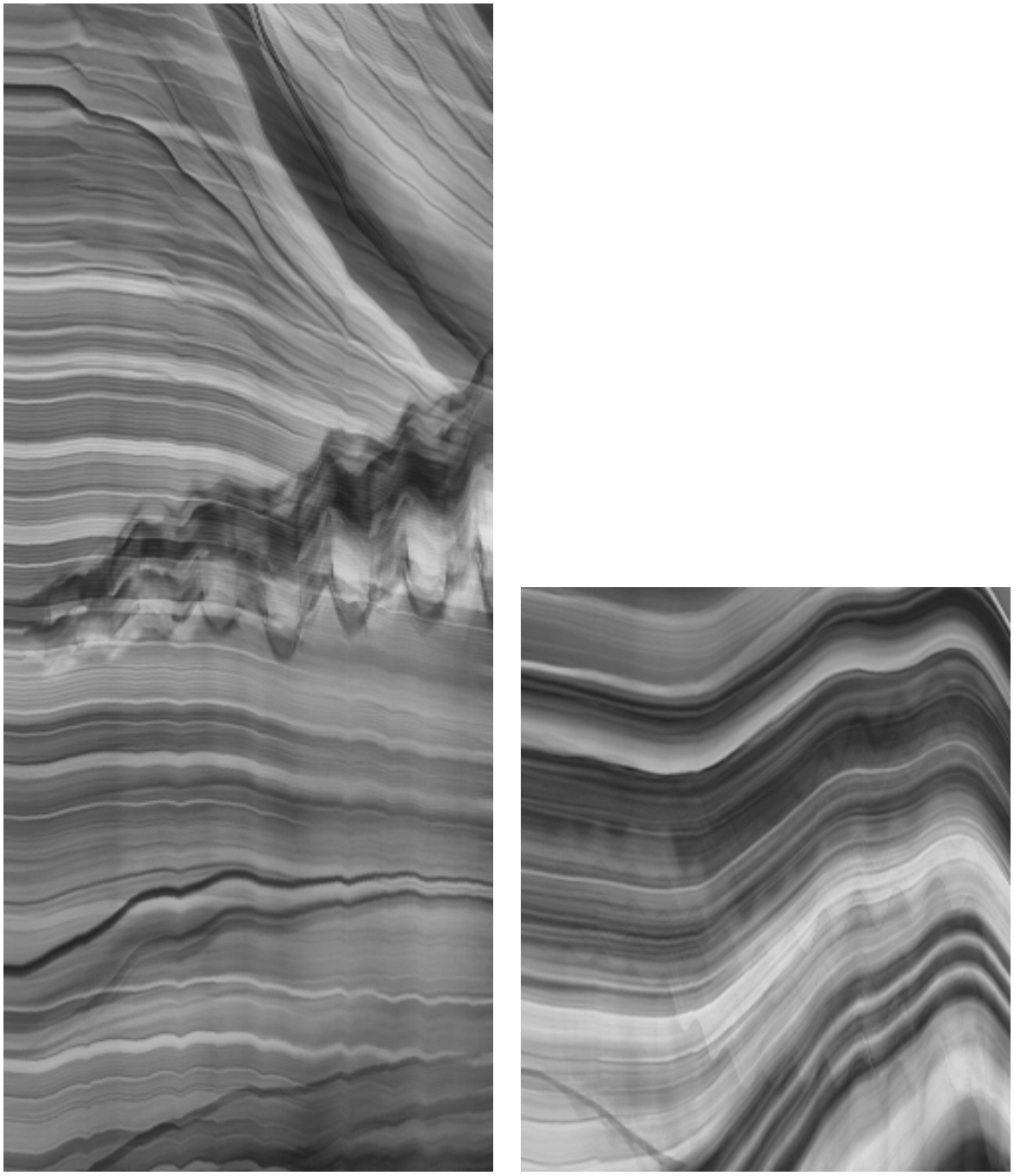
Table 4.22: Comparison of ITF between semi-adaptive Gaussian filter and adaptive Gaussian filter.

# Video	Original	Semi-Adaptive Gaussian	Adaptive Gaussian
Crowd	19.479	23.016	22.964
Parallax	18.746	20.903	20.808
Quick Rotation	19.964	22.612	21.686
Regular	19.457	25.007	24.732
Running	17.327	22.938	21.413
Zooming	20.113	21.413	20.868
Average	19.181	22.648	22.079

holds more pixels) and comparable (when both methods hold basically the same amount of pixels). Figures 4.40, 4.41 and 4.42 illustrate the analysis performed. This experiment verifies the performance of the YouTube's method and compare the performance achieved with the adaptive Gaussian filter.

We can observe a certain parity for both methods in terms of ITF and ITF_{SSIM} metrics, with a slight advantage of the YouTube method [36], while the maintained pixels are in general comparable and, when lower, they do not differ much. This demonstrates that the proposed method is competitive with one of the methods considered as current state-of-the-art.

From Figure 4.40, it is possible to observe that more information is maintained on the



(a) horizontal visual rhythm (b) vertical visual rhythm

Figure 4.39: Visual rhythms for video #3 stabilized by adaptive Gaussian filter.

top, left and right sides of the video obtained with our method. The difference is not considerably large and the advantage or disadvantage obtained follows these proportions in most videos.

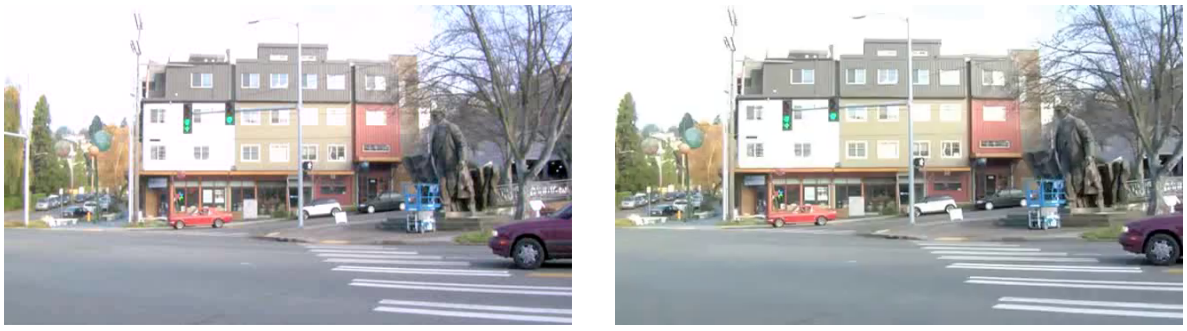
In Figure 4.41, there is less information maintained on the top and bottom sides in the use of the adaptive Gaussian filter. On the other hand, there is a larger amount of information held on the left and right sides. Figure 4.42 illustrates a situation where our method maintains less pixels. Lower amount of information is held on each side with our method.

Table 4.23: Comparison of ITF_{SSIM} between semi-adaptive Gaussian filter and adaptive Gaussian filter.

# Video	Original	Semi-Adaptive Gaussian	Adaptive Gaussian
Crowd	0.603	0.762	0.763
Parallax	0.562	0.658	0.658
Quick Rotation	0.569	0.563	0.638
Regular	0.501	0.779	0.771
Running	0.435	0.716	0.646
Zooming	0.536	0.604	0.598
Average	0.534	0.680	0.679

Table 4.24: Comparison of hold pixels (%) between semi-adaptive Gaussian filter and adaptive Gaussian filter.

# Video	Semi-Adaptive Gaussian	Adaptive Gaussian
Crowd	48.658	59.434
Parallax	54.102	65.675
QuickRotation	7.465	47.217
Regular	67.959	72.046
Running	34.157	57.597
Zooming	26.973	52.140
Average	39.886	59.018



(a) Adaptive Gaussian filter (b) YouTube [36]

Figure 4.40: Video `#1`: amount of pixels hold through our method is superior than state-of-the-art approach.

The advantages of `YouTube` stabilization are a bit sharper when looking at videos and visual rhythms. Figure 4.43 shows the visual rhythms of the video `#3`. The horizontal rhythm presents more significant differences, with smoother lines and without the trepidations presented in the rhythm derived from the adaptive Gaussian filter version.

Some even more pronounced trepidations can be seen in the video `#10`. Figures 4.44, 4.45 and 4.46 present visual rhythms of the original video `#10` and after stabilization with

Table 4.25: Comparison of ITF between adaptive Gaussian filter and YouTube method [36].

# **Video**	**Original**	**Adaptive Gaussian Filter**	**YouTube** [36]	**Hold Pixels**
1	18.793	27.455	27.890	Superior
2	20.390	28.914	28.604	Superior
3	16.186	22.090	23.030	Comparable
4	19.965	27.931	33.711	Superior
5	23.277	27.360	27.599	Inferior
6	19.681	29.077	29.390	Superior
7	24.109	28.876	29.252	Comparable
8	17.881	25.182	25.908	Superior
9	19.248	21.435	20.922	Inferior
10	12.972	16.381	20.495	Superior
11	21.487	25.659	26.672	Comparable
12	15.081	17.895	19.283	Comparable
13	23.841	29.987	28.845	Comparable
14	18.065	19.773	20.128	Inferior
Average	19.355	24.858	25.837	-

Table 4.26: Comparison of ITF_{SSIM} between adaptive Gaussian filter and YouTube method [36].

# **Video**	**Adaptive Gaussian Filter**	**YouTube** [36]
1	0.896	0.907
2	0.931	0.931
3	0.789	0.832
4	0.884	0.968
5	0.842	0.849
6	0.941	0.940
7	0.908	0.931
8	0.877	0.900
9	0.587	0.621
10	0.470	0.709
11	0.835	0.871
12	0.465	0.580
13	0.834	0.871
14	0.778	0.768
Average	0.788	0.834

Gaussian filter and `YouTube` method. We can notice that the motion intent obtained with the adaptive Gaussian filter is as smooth as than that obtained with `YouTube`. However, there are several remaining twitches. This is due to the more local nature of the movement

(a) Adaptive Gaussian filter

(b) YouTube [36]

Figure 4.41: Video #3: amount of pixels hold through our method is comparable to state-of-the-art approach.

(a) Adaptive Gaussian filter

(b) YouTube [36]

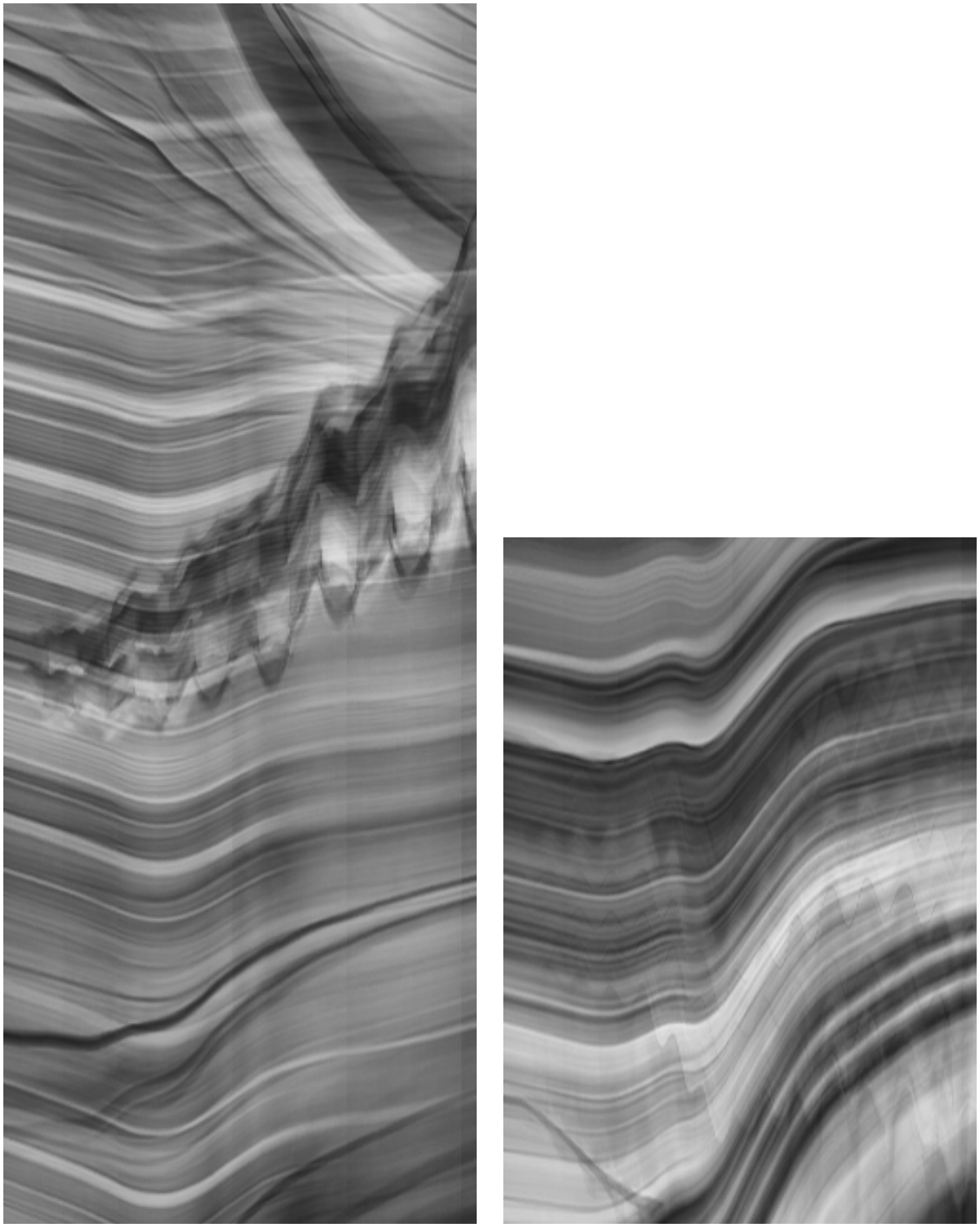
Figure 4.42: Video #12: amount of pixels hold through our method is inferior than state-of-the-art approach.

present in the video. Our method does not produce results with the same quality since it does not deal with local motion, such as the mesh-based method [69], and does not use the wobble suppression method applied by the YouTube method [36].

4.4.2 Local Combined Features

In this subsection, we describe the results obtained with the combination of local features after the removal of unwanted motion. The mean values of ITF and ITF_{SSIM} for the videos obtained in the end of the process are presented in Tables 4.27 and 4.28, in addition to the percentage of pixels held reported in Table 4.29. Thus, we can observe the impact of the estimation of the motion on the final result of the stabilization. For smoothing the trajectory, the adaptive Gaussian filter was used since the videos to be considered have intentional camera motion, and the direct application of the Gaussian filter would result in a video with few or no pixels.

As in the motion estimation evaluation, we can see that the combination obtained better results for most videos with respect to ITF and the ITF_{SSIM}. The percentage of video frame pixels held is smaller for the combination compared to SIFT. This is reasonable, since the higher the correction applied by the transformation matrices, the larger the amount of pixels lost. However, it is possible to notice that, in some cases,

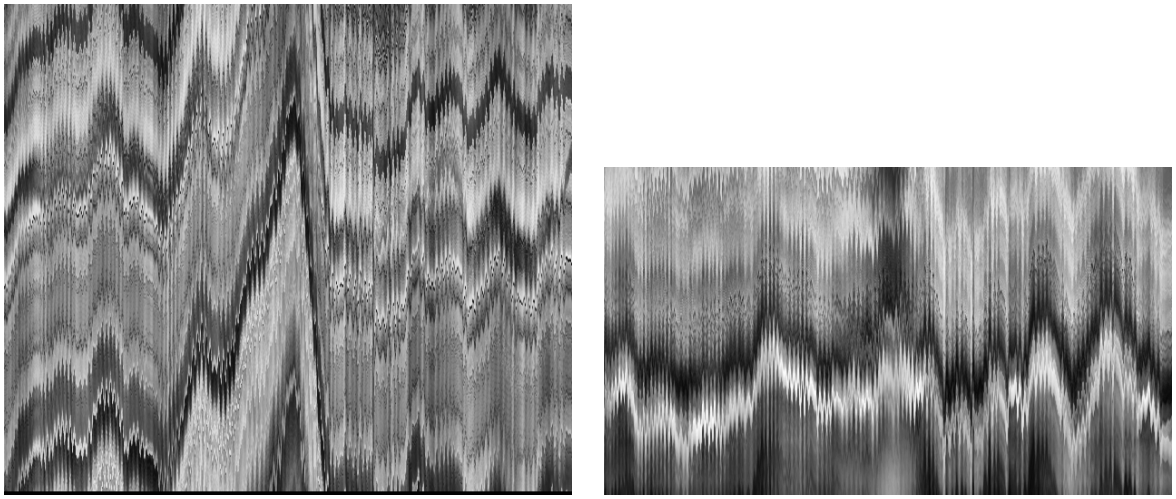


(a) horizontal visual rhythm (b) vertical visual rhythm

Figure 4.43: Visual rhythms for video #3 stabilized by YouTube.

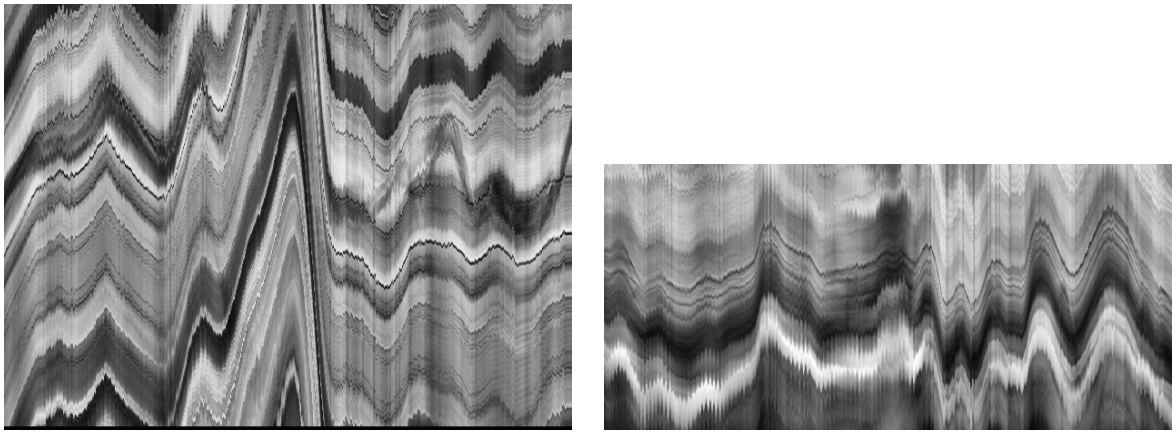
the percentage of pixels held in videos with poor quality estimation can also cause many pixels to be lost. This occurs, for example, in the video #7 for the MSER and STAR detectors.

In Table 4.30, we have the average ITF values for each category of dataset present in Table 4.2. In Table 4.31, we have the ITF_{SSIM} values. Finally, Table 4.32 reports the



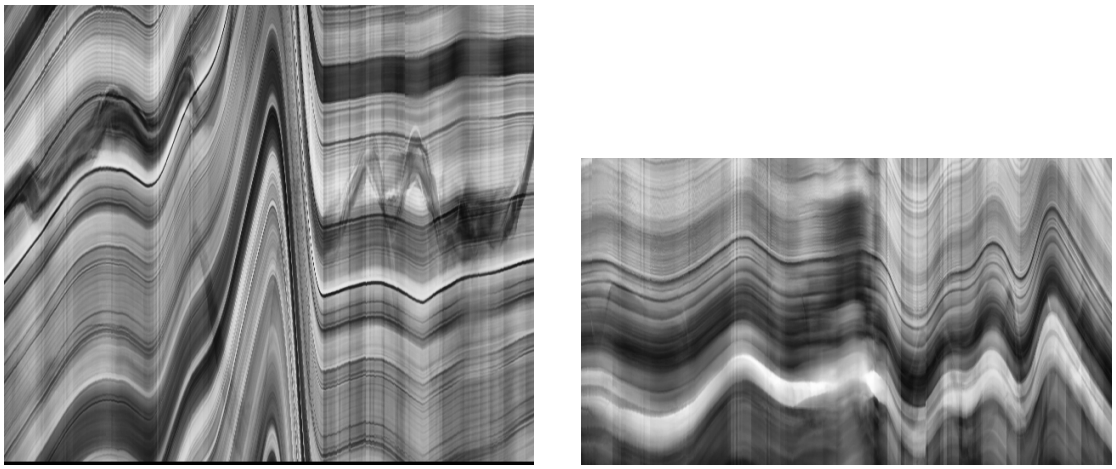
(a) horizontal visual rhythm (b) vertical visual rhythm

Figure 4.44: Visual rhythms for original video `#10`.



(a) horizontal visual rhythm (b) vertical visual rhythm

Figure 4.45: Visual rhythms for video `#3` stabilized by adaptive Gaussian filter.



(a) horizontal visual rhythm (b) vertical visual rhythm

Figure 4.46: Visual rhythms for video `#3` stabilized by YouTube.

Table 4.27: ITF for the stabilized videos.

Video	Original	MSER	SIFT	STAR	Combination
1	18.793	26.824	27.451	27.009	27.540
2	20.390	26.153	28.349	27.760	28.528
3	16.186	20.060	21.041	21.190	21.912
4	19.965	23.765	27.666	26.334	27.676
5	23.277	25.625	27.343	26.385	27.347
6	19.681	23.307	29.089	25.298	29.139
7	24.109	-	28.147	21.383	26.555
8	17.881	24.215	24.702	24.877	25.226
9	19.248	20.369	20.998	20.254	21.454
10	12.972	16.021	16.360	16.359	16.475
11	21.487	25.231	25.708	24.870	25.474
12	15.081	17.785	17.667	18.142	17.834
13	23.841	25.109	25.474	26.903	26.406
14	18.065	19.466	19.725	19.389	19.674
Average	19.355	22.610	24.266	23.297	24.374

Table 4.28: ITF_{SSIM} for the stabilized videos.

Video	Original	MSER	SIFT	STAR	Combination
1	0.468	0.878	0.895	0.883	0.897
2	0.571	0.853	0.918	0.903	0.920
3	0.334	0.692	0.735	0.745	0.781
4	0.673	0.779	0.881	0.855	0.882
5	0.715	0.799	0.838	0.817	0.839
6	0.555	0.780	0.934	0.849	0.941
7	0.719	-	0.881	0.649	0.824
8	0.486	0.839	0.863	0.865	0.873
9	0.442	0.480	0.560	0.459	0.579
10	0.295	0.441	0.466	0.464	0.475
11	0.689	0.818	0.838	0.805	0.828
12	0.317	0.420	0.453	0.421	0.457
13	0.643	0.720	0.729	0.773	0.762
14	0.657	0.751	0.775	0.749	0.773
Average	0.540	0.712	0.769	0.731	0.774

percentage of pixels held in the video frames.

In the evaluation of the videos obtained after the entire stabilization process, it is possible to observe from Tables 4.30, 4.31 and 4.32 that the results confirm the best performance when the combination is used.

Table 4.29: Pixels kept in the final videos.

Video	MSER	SIFT	STAR	Combination
1	73.245	73.500	74.500	73.500
2	76.024	75.762	76.286	76.286
3	76.311	76.823	76.312	76.569
4	60.375	62.465	62.465	62.465
5	23.042	54.253	47.931	54.031
6	71.333	70.838	71.829	70.838
7	0	74.861	4.278	65.333
8	73.241	73.284	73.241	73.590
9	13.512	57.740	7.913	53.389
10	70.296	69.440	71.383	69.146
11	54.771	57.083	54.542	55.958
12	12.092	59.297	10.220	55.660
13	64.500	72.781	68.203	69.094
14	52.576	53.672	50.750	52.877
Average	51.523	66.557	53.561	64.910

Table 4.30: Average ITF for the stabilized videos.

Category	Original	MSER	SIFT	STAR	Combination
Crowd	19.479	22.200	22.657	22.587	22.930
Parallax	18.746	19.403	20.463	20.528	21.096
Quick Rotation	19.964	20.050	21.514	21.242	21.314
Regular	19.457	23.869	21.354	23.622	24.468
Running	17.327	20.269	20.917	20.664	21.153
Zooming	20.113	19.384	20.658	20.137	20.845
Average	19.181	20.862	21.261	21.463	21.968

Table 4.31: ITF_{SSIM} for the stabilized videos.

Category	Original	MSER	SIFT	STAR	Combination
Crowd	0.603	0.731	0.751	0.748	0.762
Parallax	0.562	0.603	0.642	0.642	0.671
Quick Rotation	0.569	0.571	0.634	0.508	0.605
Regular	0.501	0.592	0.726	0.722	0.758
Running	0.435	0.581	0.620	0.579	0.628
Zooming	0.536	0.524	0.590	0.565	0.598
Average	0.534	0.600	0.661	0.627	0.670

Table 4.32: Average of the pixels maintained for the stabilized videos.

Category	MSER	SIFT	STAR	Combination
Crowd	56.194	59.555	56.110	58.874
Parallax	55.971	66.283	66.184	62.174
Quick Rotation	35.332	48.669	24.902	37.151
Regular	62.831	71.981	70.941	72.019
Running	47.572	57.332	49.558	56.217
Zooming	45.588	52.042	45.200	49.494
Average	50.581	59.310	52.149	55.988

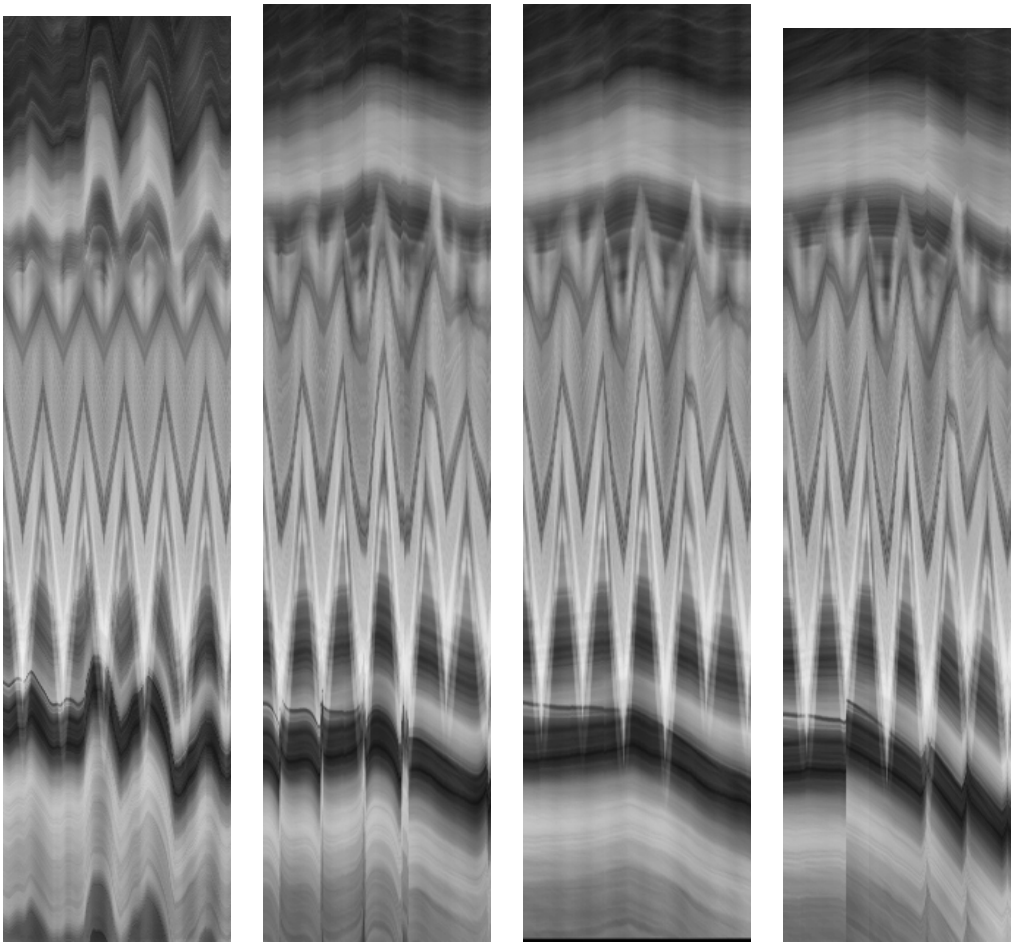
4.4.3 Spatio-Temporal Optimization

In this subsection, we present the final results of the stabilization of the videos considering the process of spatio-temporal optimization, after the process of removal of unwanted motion. For the stabilization of videos, we apply an adaptive Gaussian filter in the motion estimation obtained only with local features, as in the results obtained with our method. In addition, we compared our results against those achieved with the state-of-the-art `YouTube` stabilization method [36]. To do this, we submitted the unstable videos on `YouTube` and retrieved the video generated after the stabilization process.

The step of smoothing the camera path applied by the `YouTube` method is more robust and can yield superior results. Moreover, we apply neither a local estimate nor a wobble suppression technique, such as that applied by the `YouTube` method. In frames where there is no critical problem in motion estimation, the `YouTube` method tends to perform better. This is not emphasized in our experiments, since the scope of this work is to present the correction in the global motion estimation. In this subsection, our focus is to verify the impact of these problems and their correction on the final video, after the entire stabilization process.

Figure 4.47 shows the horizontal visual rhythms obtained both for the original and stabilized videos. We can notice that the rhythms obtained in the stabilization through estimation based on local features and in the version obtained with the `YouTube` method have several discontinuities, which represent abrupt movements in the videos, both due to the problem in the motion estimation presented in Figure 4.32. On the other hand, our method does not have these discontinuities and achieves a considerably superior result, as illustrated through its visual rhythm.

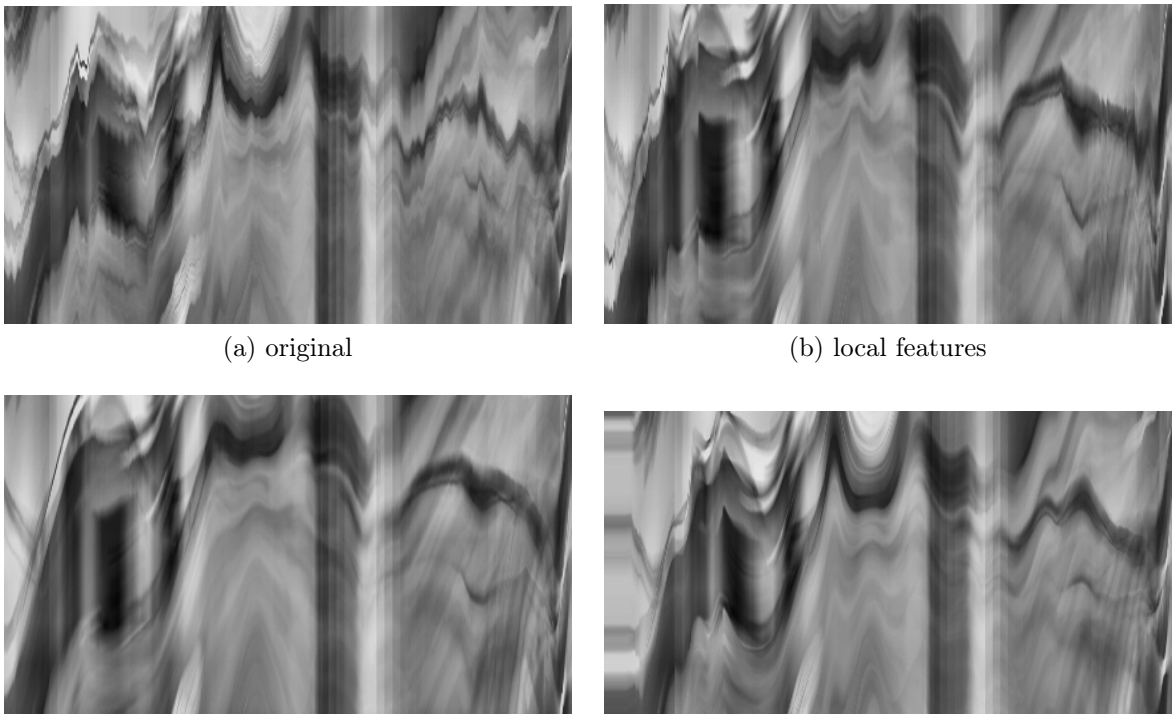
Figure 4.48 shows the vertical visual rhythms obtained both from unstable video `#13` and videos stabilized with the three versions. We can verify that the beginning of the `YouTube` visual rhythm has vertical lines, unlike the original rhythm and the other stabilized versions. This indicates that the `YouTube` method has added an artificial motion to the video, which does not correspond to the desired purpose. In this case, the image remains static in the first frames, without the occurrence of motion. This occurred because the motion estimation computed by the `YouTube` method takes into account the movement of the object, which is consequently compensated in the stabilization.



(a) original (b) local features (c) our result (d) YouTube

Figure 4.47: Horizontal visual rhythms for video $\texttt{ours}_1$.

The visual rhythms, illustrated in Figure 4.48, show that our method corrects several instabilities or discontinuities that occur in the estimation based on local features. This is especially noticeable at the beginning and end of the visual rhythms. Compared to the visual rhythm obtained with `YouTube`, the rhythm generated by the proposed method is significantly more regular, representing a better quality in the video stabilization process.



(a) original (b) local features

(c) our result (d) YouTube

Figure 4.48: Vertical visual rhythms for video $\texttt{ours}_7$.

Chapter 5

Conclusions

The main objective of this work was to investigate the problem of video stabilization. We then developed and evaluated 2D methods for digital stabilization of videos. The 2D video stabilization process is usually divided into three main steps: estimation of camera motion, removal of unwanted motion, and generation of the corrected video.

This work presented five novel methods related to digital video stabilization. Experiments were conducted on three distinct sets of videos. The first consisted of fourteen video sequences. The second consisted of one hundred and thirty-nine videos divided into six categories. The third, constructed in this work, consisted of eight videos.

For the step of camera motion estimation, this work presented two new approaches. The first technique combined local features in the context of video stabilization, where different local feature detectors were applied to each pair of adjacent frames to obtain a consensual estimation of the frame-by-frame motion of the methods under consideration. The results obtained with the combination were compared to the methods employed individually. In general, the combined method was relatively close to the best detector and, in some videos, was considerably superior. The second technique improved the global motion estimation through a spatio-temporal optimization. The proposed method was based on the structural similarity index and used temporal information to detect and correct the motion estimation obtained with local features. Results achieved with the stabilization process were compared to the state-of-the-art `YouTube` method. The obtained results showed that the proposed method was capable of properly detecting and correcting problems in the global motion estimation step. Since the motion estimation in the `YouTube` method is based on local features, these problems were visible in the stabilized video. Our method, in turn, dealt well with such cases and generated videos that are considerably more stable. In cases where video frames have neither very representative background nor moving objects, our method overcomes the `YouTube` method.

For the step of unwanted motion removal, we presented a technique for video stabilization based on an adaptive Gaussian filter to smooth the camera trajectory in order to remove oscillations. The proposed filter assigned distinct values to σ along the camera trajectory by considering that the intensity of the oscillations changes throughout the video. The results obtained in the experiments were compared to different versions for the smoothing of the trajectory: Kalman filter, Gaussian filter with $\sigma = 40$, and a semi-adaptive Gaussian filter. The new approach achieved comparable values for stabilization

quality while maintaining a significantly higher amount of pixels. A comparison against the stabilization method used on YouTube demonstrated that our approach produced competitive results.

In this work, we also investigated and proposed two techniques for the subjective evaluation of stabilization. The first was a representation based on visual rhythms for the subjective evaluation of video stabilization. The vertical visual rhythm was constructed from the average of the columns of each frame, whereas the horizontal visual rhythm was constructed from the average of the rows of each frame. We were able to characterize and separate the horizontal and vertical movements of the video, determining how and when they occur. The stability of a video could be determined from the regularity and smoothness of the curves for each visual rhythm. Furthermore, the presence of more complex movements, such as zoom, could be verified in the visual rhythm. The second was a representation based on the motion energy image (MEI) for the subjective evaluation of video stabilization. The visualization was constructed from the mean of MEIs calculated for all the video frames and then highlighted with a pseudocolor transformation. We were able to characterize the amount of spatial motion, as well as its location, present in the video. By considering that an unstable video has a greater amount of motion than its stabilized version, we could use this technique to evaluate the stabilization of videos. The results showed that the proposed visualization was adequate and represented well both the amount and location of spatial motion.

Based on the results presented and discussed, we answered the research questions presented in the Chapter 1 as:

- Can the use of different local features combined improve the motion estimation? Answer: Motion estimation can be improved through the combination of local features, that is, if the use of a single local feature method has not been sufficient for a correct motion estimation.

- Can the information from motion estimation of adjacent frames be used to detect and correct failures in the motion estimation? Answer: The information from previous frames can be used to detect and correct motion estimation through an optimization method.

- Can an adaptive filter generate videos with a higher amount of information maintaining the quality of the stabilization? Answer: An adaptive Gaussian filter achieved comparable results for stabilization quality in comparison with its direct version while maintaining a significantly higher amount of pixels.

- Are the stabilization evaluation metrics available in the literature coherent with visual perception? Answer: In many cases, the ITF and ITF_{SSIM} values have been inconsistent with the visual perception of video stability.

- Can the visual rhythm and the motion energy image characterize the stability of a video and be used for its evaluation? Answer: The two proposed visual representations, based respectively on visual rhythms and motion energy image, were able to characterize the stability of a video.

5.1 Future Work

From the investigation conducted on this work, we have identified some directions that can be explored in future work. They are briefly described in the following subsections.

5.1.1 Estimation of Camera Motion

There are other local feature methods than those considered in our current experiments. The performance of these methods could be studied in a future work. Furthermore, only few combinations of local features were evaluated, such that several other combinations could be further investigated.

In the optimization method, the impact of the parameters involved in the process was only superficially studied. Thus, a deeper investigation could be carried out on this subject. Moreover, additional optimization strategies, replacing Powell's method, could be considered and analyzed.

The current methods for estimating local motion are based on the use of local features [69], which make them incompatible with the optimization method proposed in this work. Thus, we could elaborate a method for the local motion estimation for the frames in which the optimization is applied, extending our method to deal with local motion.

5.1.2 Removal of Unwanted Motion

The adaptive Gaussian filter, proposed in this work, has several parameters that can also be investigated in a more systematic way. Furthermore, methods that consider more local motion use their own optimization to remove the unwanted motion. Thus, we could extend the adaptive Gaussian filter to deal with more local motion estimates.

We could also propose a new method for removing unwanted motion through a constrained optimization, which obtained the smoothest camera path possible considering a certain minimum amount of frame pixels to be held.

5.1.3 Evaluation of Stabilization

The construction of a database with a large number of videos containing ground-truth of global motion estimation would allow for a more precise assessment of the video stabilization process.

We could develop objective metrics calculated from the visual representations (visual rhythm and motion energy image) proposed in this paper and use them for the characterization and evaluation of video stabilization.

We could also use the visual representations to train a classifier to recognize unstable and stable videos on a dataset to be built. From a trained classifier, we could use it to estimate membership [58, 81, 115] of an input video, relative to the class of stable videos. Thus, we would have a percentage of video stability, which could be used as an objective measure for the stabilization evaluation.

Additionally, we could train a classifier to determine which types of motion are present in a particular video. Thus, we would able to develop an adaptive method for the stabilization that used the information regarding the motion of the input video to its stabilization.

The determination of what is considered as unwanted motion is usually subjective. Therefore, we could propose to build a data set with stable videos and perturb them of how to obtain unstable videos. Thus, we would have the ground-truth of the stable path made by the camera, which could be used to evaluate the path obtained with the stabilized video, complementing other measurements.

Finally, we intend to create a data set with several videos divided into different categories of movements and make it available in order for other researchers to evaluate and compare their methods.

Bibliography

[1] M. Agrawal, K. Konolige, and M. R. Blas. Censure: Center Surround Extremas for Realtime Feature Detection and Matching. In *European Conference on Computer Vision*, pages 102–115. Springer, 2008.

[2] M. A. R. Ahad. *Motion History Images for Action Recognition and Understanding*. Springer Science & Business Media, 2012.

[3] A. Alahi, R. Ortiz, and P. Vandergheynst. FREAK: Fast Retina Keypoint. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 510–517. IEEE, 2012.

[4] A. A. Amanatiadis and I. Andreadis. Digital Image Stabilization by Independent Component Analysis. *IEEE Transactions on Instrumentation and Measurement*, 59(7):1755–1763, 2010.

[5] R. Bajcsy and S. Kovačič. Multiresolution Elastic Matching. *Computer Vision, Graphics, and Image Processing*, 46(1):1–21, 1989.

[6] S. Battiato, G. Gallo, G. Puglisi, and S. Scellato. SIFT Features Tracking for Video Stabilization. In *14th International Conference on Image Analysis and Processing*, pages 825–830. IEEE, 2007.

[7] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool. Speeded-up Robust Features (SURF). *Computer Vision and Image Understanding*, 110(3):346–359, 2008.

[8] P. Bhat, C. L. Zitnick, N. Snavely, A. Agarwala, M. Agrawala, M. Cohen, B. Curless, and S. B. Kang. Using Photographs to Enhance Videos of a Static Scene. In *18th Eurographics Conference on Rendering Techniques*, pages 327–338. Eurographics Association, 2007.

[9] N. Bhowmik, V. Gouet-Brunet, L. Wei, and G. Bloch. Adaptive and Optimal Combination of Local Features for Image Retrieval. In *International Conference on Multimedia Modeling*, pages 76–88. Springer, 2017.

[10] R. Borgo, M. Chen, B. Daubney, E. Grundy, G. Heidemann, B. Höferlin, M. Höferlin, H. Leitte, D. Weiskopf, and X. Xie. State of the Art Report on Video-Based Graphics and Video Visualization. *Computer Graphics Forum*, 31(8):2450–2477, 2012.

[11] G. Bradski and A. Kaehler. *Learning OpenCV: Computer Vision with the OpenCV Library*. O'Reilly Media, Inc., 2008.

[12] R. P. Brent. An Algorithm with Guaranteed Convergence for Finding a Zero of a Function. *The Computer Journal*, 14(4):422–425, 1971.

[13] C. Buehler, M. Bosse, and L. McMillan. Non-Metric Image-based Rendering for Video Stabilization. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2. IEEE, 2001.

[14] B. Cardani. Optical Image Stabilization for Digital Cameras. *IEEE Control Systems*, 26(2):21–22, 2006.

[15] H.-C. Chang, S.-H. Lai, and K.-R. Lu. A Robust and Efficient Video Stabilization Algorithm. In *IEEE International Conference on Multimedia and Expo*, volume 1, pages 29–32. IEEE, 2004.

[16] J.-Y. Chang, W.-F. Hu, M.-H. Cheng, and B.-S. Chang. Digital Image Translational and Rotational Motion Stabilization using Optical Flow Technique. *IEEE Transactions on Consumer Electronics*, 48(1):108–115, 2002.

[17] B. Chen, J. Zhao, and Y. Wang. Research on Evaluation Method of Video Stabilization. In *International Conference on Advanced Material Science and Environmental Engineering*, pages 253–258. Atlantis Press, 2016.

[18] B.-H. Chen, A. Kopylov, S.-C. Huang, O. Seredin, R. Karpov, S.-Y. Kuo, K. R. Lai, T.-H. Tan, M. Gochoo, and D. Bayanduuren. Improved Global Motion Estimation via Motion Vector Clustering for Video Stabilization. *Engineering Applications of Artificial Intelligence*, 54:39–48, 2016.

[19] B.-Y. Chen, K.-Y. Lee, W.-T. Huang, and J.-S. Lin. Capturing Intention-based Full-Frame Video Stabilization. *Computer Graphics Forum*, 27(7):1805–1814, 2008.

[20] S. Choi, T. Kim, and W. Yu. Robust Video Stabilization to Outlier Motion using Adaptive RANSAC. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1897–1902. IEEE, 2009.

[21] E. K. Chong and S. H. Zak. *An Introduction to Optimization*, volume 76. John Wiley & Sons, 2013.

[22] M. Chung, J. Lee, H. Kim, S. Song, and W. Kim. Automatic Video Segmentation based on Spatio-temporal Features. *Korea Telecom Journal*, 4(1):4–14, 1999.

[23] M. V. M. Cirne and H. Pedrini. A Video Summarization Method Based on Spectral Clustering. In *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications*, pages 479–486. Springer, 2013.

[24] M. V. M. Cirne and H. Pedrini. Summarization of Videos by Image Quality Assessment. In *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications*, pages 901–908. Springer, 2014.

[25] U. Diwekar. *Introduction to Applied Optimization*, volume 22. Springer Science & Business Media, 2008.

[26] S. Ertürk. Image Sequence Stabilisation based on Kalman Filtering of Frame Positions. *Electronics Letters*, 37(20):1, 2001.

[27] S. Ertürk. Real-Time Digital Image Stabilization using Kalman Filters. *Real-Time Imaging*, 8(4):317–328, 2002.

[28] M. A. Fischler and R. C. Bolles. Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[29] T. A. E. S. Freitas. *Estabilização de Vídeos com Base em Descritores H. 264*. PhD thesis, Faculdade de Ciências e Tecnologia, Universidade Nova de Lisboa, Portugal, Mar. 2013.

[30] G. Gales, A. Crouzil, and S. Chambon. Complementarity of Feature Point Detectors. In *International Conference on Computer Vision Theory and Applications*, pages 334–339, Angers, France, May 2010.

[31] M. L. Gleicher and F. Liu. Re-cinematography: Improving the Camerawork of Casual Video. *ACM Transactions on Multimedia Computing, Communications, and Applications*, 5(1):2, 2008.

[32] A. Goldstein and R. Fattal. Video Stabilization using Epipolar Geometry. *ACM Transactions on Graphics*, 31(5):1–10, Aug. 2012.

[33] R. C. Gonzalez and R. E. Woods. *Digital Image Processing*. Prentice Hall, Upper Saddle River, NJ, 2002.

[34] A. A. Goshtasby. *Image Registration: Principles, Tools and Methods*. Springer Science & Business Media, 2012.

[35] M. S. Grewal. *Kalman Filtering*. Springer, 2011.

[36] M. Grundmann, V. Kwatra, and I. Essa. Auto-Directed Video Stabilization with Robust L1 Optimal Camera Paths. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 225–232. IEEE, June 2011.

[37] H. Guo, S. Liu, S. Zhu, and B. Zeng. Joint Bundled Camera Paths for Stereoscopic Video Stabilization. In *IEEE International Conference on Image Processing*, pages 1071–1075. IEEE, 2016.

[38] A. Hamza, R. Hafiz, M. M. Khan, Y. Cho, and J. Cha. Stabilization of Panoramic Videos from Mobile Multi-camera Platforms. *Image and Vision Computing*, 37:20–30, 2015.

[39] R. M. Haralick and K. Shanmugam. Textural Features for Image Classification. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-3(6):610–621, 1973.

[40] C. Harris and M. Stephens. A Combined Corner and Edge Detector. In *Alvey Vision Conference*, volume 15, page 50. Citeseer, 1988.

[41] T. S. Huang. *Image Sequence Analysis*, volume 5. Springer Science & Business Media, 2013.

[42] M. Jenkinson, P. Bannister, M. Brady, and S. Smith. Improved Optimization for the Robust and Accurate Linear Registration and Motion Correction of Brain Images. *Neuroimage*, 17(2):825–841, 2002.

[43] C. Jia and B. L. Evans. Online Motion Smoothing for Video Stabilization via Constrained Multiple-model Estimation. *EURASIP Journal on Image and Video Processing*, 2017(1):25, 2017.

[44] R. Jia, H. Zhang, L. Wang, and J. Li. Digital Image Stabilization based on Phase Correlation. In *International Conference on Artificial Intelligence and Computational Intelligence*, volume 3, pages 485–489. IEEE, 2009.

[45] E. Jones, T. Oliphant, and P. Peterson. SciPy: Open Source Scientific Tools for Python. *http://www.scipy.org*, 4, 2014.

[46] N. Joshi, W. Kienzle, M. Toelle, M. Uyttendaele, and M. F. Cohen. Real-time Hyperlapse Creation via Optimal Frame Selection. *ACM Transactions on Graphics*, 34(4):63, 2015.

[47] S. J. Julier and J. K. Uhlmann. New Extension of the Kalman Filter to Nonlinear Systems. In *AeroSense*, pages 182–193. International Society for Optics and Photonics, 1997.

[48] R. E. Kalman. A New Approach to Linear Filtering and Prediction Problems. *Journal of Basic Engineering*, 82(1):35–45, 1960.

[49] J. Kiefer. Sequential Minimax Search for a Maximum. *American mathematical society*, 4(3):502–506, 1953.

[50] S. W. Kim, S. Yin, K. Yun, and J. Y. Choi. Spatio-temporal Weighting in Local Patches for Direct Estimation of Camera Motion in Video Stabilization. *Computer Vision and Image Understanding*, 118:71–83, 2014.

[51] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by Simulated Annealing. *Science*, 220(4598):671–680, 1983.

[52] J. Kiusalaas. *Numerical Methods in Engineering with Python 3*. Cambridge University Press, 2013.

[53] S. Klein, J. P. Pluim, M. Staring, and M. A. Viergever. Adaptive Stochastic Gradient Descent Optimisation for Image Registration. *International Journal of Computer Vision*, 81(3):227, 2009.

[54] S. Klein, M. Staring, and J. P. Pluim. Evaluation of Optimization Methods for Nonrigid Medical Image Registration using Mutual Information and B-splines. *IEEE Transactions on Image Processing*, 16(12):2879–2890, 2007.

[55] S.-J. Ko, S.-H. Lee, and K.-H. Lee. Digital Image Stabilizing Algorithms based on Bit-Plane Matching. *IEEE Transactions on Consumer Electronics*, 44(3):617–622, 1998.

[56] J. Kopf. 360 Video Stabilization. *ACM Transactions on Graphics*, 35(6):195, 2016.

[57] S. Kumar, H. Azartash, M. Biswas, and T. Nguyen. Real-Time Affine Global Motion Estimation using Phase Correlation and its Application for Digital Image Stabilization. *IEEE Transactions on Image Processing*, 20(12):3406–3418, 2011.

[58] J. Langford and B. Zadrozny. Estimating Class Membership Probabilities using Classifier Learners. In *Tenth International Conference on Artificial Intelligence and Statistics*, Barbados, Jan. 2005.

[59] K.-Y. Lee, Y.-Y. Chuang, B.-Y. Chen, and M. Ouhyoung. Video Stabilization using Robust Feature Trajectories. In *IEEE 12th International Conference on Computer Vision*, pages 1397–1404. IEEE, 2009.

[60] S. Leutenegger, M. Chli, and R. Y. Siegwart. BRISK: Binary Robust Invariant Scalable Keypoints. In *International Conference on Computer Vision*, pages 2548–2555. IEEE, 2011.

[61] S. Li, L. Yuan, J. Sun, and L. Quan. Dual-feature Warping-based Motion Model Estimation. In *IEEE International Conference on Computer Vision*, pages 4283–4291, 2015.

[62] C.-T. Lin, C.-T. Hong, and C.-T. Yang. Real-Time Digital Image Stabilization System using Modified Proportional Integrated Controller. *IEEE Transactions on Circuits and Systems for Video Technology*, 19(3):427–431, 2009.

[63] A. Litvin, J. Konrad, and W. C. Karl. Probabilistic Video Stabilization using Kalman Filtering and Mosaicing. In *Proceedings of SPIE*, volume 5022, pages 663–674. International Society for Optics and Photonics, May 2003.

[64] F. Liu, M. Gleicher, H. Jin, and A. Agarwala. Content-preserving Warps for 3D Video Stabilization. *ACM Transactions on Graphics*, 28(3):44, 2009.

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

[66] F. Liu, Y. Niu, and H. Jin. Joint Subspace Stabilization for Stereoscopic Video. In *IEEE International Conference on Computer Vision*, pages 73–80, 2013.

[67] S. Liu, M. Li, S. Zhu, and B. Zeng. CodingFlow: Enable Video Coding for Video Stabilization. *IEEE Transactions on Image Processing*, 26(7):3291–3302, 2017.

[68] S. Liu, Y. Wang, L. Yuan, J. Bu, P. Tan, and J. Sun. Video Stabilization with a Depth Camera. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 89–95. IEEE, 2012.

[69] S. Liu, L. Yuan, P. Tan, and J. Sun. Bundled Camera Paths for Video Stabilization. *ACM Transactions on Graphics*, 32(4):78, 2013.

[70] S. Liu, L. Yuan, P. Tan, and J. Sun. Steadyflow: Spatially Smooth Optical Flow for Video Stabilization. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 4209–4216, 2014.

[71] D. G. Lowe. Object Recognition from Local Scale-Invariant Features. In *Seventh IEEE International Conference on Computer Vision*, volume 2, pages 1150–1157. IEEE, 1999.

[72] D. G. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

[73] B. D. Lucas and T. Kanade. An Iterative Image Registration Technique with an Application to Stereo Vision. In *International Joint Conference on Aritifical Intelligence*, volume 81, pages 674–679, 1981.

[74] M. Lutz. *Programming Python*, volume 8. O'Reilly Media, Inc., 1996.

[75] F. Maes, A. Collignon, D. Vandermeulen, G. Marchal, and P. Suetens. Multimodality Image Registration by Maximization of Mutual Information. *IEEE Transactions on Medical Imaging*, 16(2):187–198, 1997.

[76] F. Maes, D. Vandermeulen, and P. Suetens. Comparative Evaluation of Multiresolution Optimization Strategies for Multimodality Image Registration by Maximization of Mutual Information. *Medical Image Analysis*, 3(4):373–386, 1999.

[77] L. Marcenaro, G. Vernazza, and C. S. Regazzoni. Image Stabilization Algorithms for Video-Surveillance Applications. In *International Conference on Image Processing*, volume 1, pages 349–352. IEEE, 2001.

[78] J. Matas, O. Chum, M. Urban, and T. Pajdla. Robust Wide-baseline Stereo from Maximally Stable Extremal Regions. *Image and vision computing*, 22(10):761–767, 2004.

[79] 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*, 28(7):1150–1163, 2006.

[80] C. Morimoto and R. Chellappa. Fast Electronic Digital Image Stabilization. In *13th International Conference on Pattern Recognition*, volume 3, pages 284–288. IEEE, 1996.

[81] A. Niculescu-Mizil and R. Caruana. Predicting Good Probabilities with Supervised Learning. In *22nd International Conference on Machine Learning*, pages 625–632. ACM, 2005.

[82] M. Niskanen, O. Silvén, and M. Tico. Video Stabilization Performance Assessment. In *IEEE International Conference on Multimedia and Expo*, pages 405–408. IEEE, 2006.

[83] N. Otsu. A Threshold Selection Method from Gray-Level Histograms. *IEEE Transactions on Systems, Man, and Cybernetics*, 9(1):62–66, 1979.

[84] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, and V. Dubourg. Scikit-Learn: Machine Learning in Python. *The Journal of Machine Learning Research*, 12:2825–2830, 2011.

[85] H. Pedrini and W. Schwartz. *Análise de Imagens Digitais - Princípios, Algoritmos e Aplicações*. Editora Thomson Learning, 2007.

[86] A. Pinto, W. R. Schwartz, H. Pedrini, and A. R. Rocha. Using Visual Rhythms for Detecting Video-based Facial Spoof Attacks. *IEEE Transactions on Information Forensics and Security*, 10(5):1025–1038, 2015.

[87] M. J. Powell. An Efficient Method for Finding the Minimum of a Function of Several Variables without Calculating Derivatives. *The computer journal*, 7(2):155–162, 1964.

[88] M. J. Powell. An Efficient Method for Finding the Minimum of a Function of Several Variables Without Calculating Derivatives. *The Computer Journal*, 7(2):155–162, 1964.

[89] G. Puglisi and S. Battiato. A Robust Image Alignment Algorithm for Video Stabilization Purposes. *IEEE Transactions on Circuits and Systems for Video Technology*, 21(10):1390–1400, 2011.

[90] H. Qu, L. Song, and G. Xue. Shaking Video Synthesis for Video Stabilization Performance Assessment. In *Visual Communications and Image Processing*, pages 1–6. IEEE, 2013.

[91] K. Ratakonda. Real-time Digital Video Stabilization for Multi-media Applications. In *IEEE International Symposium on Circuits and Systems*, volume 4, pages 69–72. IEEE, 1998.

[92] E. Rosten and T. Drummond. Machine Learning for High-speed Corner Detection. *European Conference on Computer Vision*, pages 430–443, 2006.

[93] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski. ORB: An Efficient Alternative to SIFT or SURF. In *IEEE International Conference on Computer Vision*, pages 2564–2571. IEEE, 2011.

[94] Y. G. Ryu and M. J. Chung. Robust Online Digital Image Stabilization based on Point-Feature Trajectory without Accumulative Global Motion Estimation. *IEEE Signal Processing Letters*, 19(4):223–226, 2012.

[95] Y. Shen, P. Guturu, T. Damarla, B. P. Buckles, and K. R. Namuduri. Video Stabilization using Principal Component Analysis and Scale Invariant Feature Transform in Particle Filter Framework. *IEEE Transactions on Consumer Electronics*, 55(3):1714–1721, 2009.

[96] D. Shukla and R. K. Jha. A Robust Video Stabilization Technique using Integral Frame Projection Warping. *Signal, Image and Video Processing*, 9(6):1287–1297, 2015.

[97] A. Silva Pinto, H. Pedrini, W. Schwartz, and A. Rocha. Video-based Face Spoofing Detection through Visual Rhythm Analysis. In *25th Conference on Graphics, Patterns and Images*, pages 221–228. IEEE, 2012.

[98] I. Sobel and G. Feldman. A 3x3 Isotropic Gradient Operator for Image Processing. *A Talk at the Stanford Artificial Project in*, pages 271–272, 1968.

[99] M. R. Souza, L. F. R. Fonseca, and H. Pedrini. Improvement of Global Motion Estimation in Two-Dimensional Digital Video Stabilization Methods. *IET Image Processing*, 2017. (submitted).

[100] M. R. Souza and H. Pedrini. Combination of Local Feature Detection Methods for Digital Video Stabilization. *Signal, Image and Video Processing*, 2017. (submitted).

[101] M. R. Souza and H. Pedrini. Digital Video Stabilization Based on Adaptive Camera Trajectory Smoothing. *EURASIP Journal on Image and Video Processing*, 2017. (submitted).

[102] M. R. Souza and H. Pedrini. Motion Energy Image for Evaluation of Video Stabilization. *The Visual Computer*, 2017. (submitted).

[103] M. R. Souza and H. Pedrini. Visual Rhythms for Qualitative Evaluation of Video Stabilization. *Journal of Signal Processing Systems*, 2017. (submitted).

[104] R. Stephens. *Essential Algorithms: A Practical Approach to Computer Algorithms*. John Wiley & Sons, 2013.

[105] R. Szeliski. *Computer Vision: Algorithms and Applications*. Springer Science & Business Media, 2010.

[106] T. Tuytelaars, K. Mikolajczyk, et al. Local Invariant Feature Detectors: A Survey. *Foundations and Trends® in Computer Graphics and Vision*, 3(3):177–280, 2008.

[107] S. Van der Walt, S. C. Colbert, and G. Varoquaux. The NumPy Array: A Structure for Efficient Numerical Computation. *Computing in Science & Engineering*, 13(2):22–30, 2011.

[108] S. Van der Walt, J. L. Schönberger, J. Nunez-Iglesias, F. Boulogne, J. D. Warner, N. Yager, E. Gouillart, and T. Yu. Scikit-Image: Image Processing in Python. *PeerJ*, 2:e453, 2014.

[109] M. P. Wachowiak, R. Smolíková, Y. Zheng, J. M. Zurada, and A. S. Elmaghraby. An Approach to Multimodal Biomedical Image Registration utilizing Particle Swarm Optimization. *IEEE Transactions on Evolutionary Computation*, 8(3):289–301, 2004.

[110] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image Quality Assessment: From Error Visibility to Structural Similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004.

[111] S. Winter, B. Brendel, I. Pechlivanis, K. Schmieder, and C. Igel. Registration of CT and Intraoperative 3-D Ultrasound Images of the Spine using Evolutionary and Gradient-based Methods. *IEEE Transactions on Evolutionary Computation*, 12(3):284–296, 2008.

[112] J. Yang, D. Schonfeld, C. Chen, and M. Mohamed. Online Video Stabilization based on Particle Filters. In *International Conference on Image Processing*, pages 1545–1548. IEEE, 2006.

[113] J. Yang, D. Schonfeld, and M. Mohamed. Robust Video Stabilization based on Particle Filter Tracking of Projected Camera Motion. *IEEE Transactions on Circuits and Systems for Video Technology*, 19(7):945–954, 2009.

[114] I. T. Young, J. J. Gerbrands, and L. J. Van Vliet. *Fundamentals of Image Processinga*. Delft University of Technology Delft, 1998.

[115] B. Zadrozny and C. Elkan. Transforming Classifier Scores into Accurate Multiclass Probability Estimates. In *Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 694–699. ACM, 2002.

[116] G. Zhang, W. Hua, X. Qin, Y. Shao, and H. Bao. Video Stabilization based on a 3D Perspective Camera Model. *The Visual Computer*, 25(11):997–1008, 2009.

[117] Z. Zhao and X. Ma. Video Stabilization based on Local Trajectories and Robust Mesh Transformation. In *IEEE International Conference on Image Processing*, pages 4092–4096. IEEE, 2016.

[118] Q. Zheng and M. Yang. A Video Stabilization Method Based on Inter-frame Image Matching Score. *Global Journal of Computer Science and Technology*, 17(1-F), 2017.

[119] K. Zuiderveld. Contrast Limited Adaptive Histogram Equalization. In *Graphics Gems IV*, pages 474–485. Academic Press Professional, Inc., 1994.

Appendix A

Additional Experiments

This appendix presents some additional topics explored in this work related to the video stabilization problem. We intend to conduct a more detailed investigation on each of these tasks to extend the methods and improve their results.

The following subsections address each topic: (i) parallelization of digital video stabilization, (ii) objective evaluation from metrics extracted from visual representations, and (iii) local motion estimation.

A.1 Parallelization of Digital Video Stabilization

Due to the increase in the amount of videos available and their spatial resolution, it is fundamental to develop efficient methods for video processing. The structure of the stabilization problem presents some steps that can be performed independently. Thus, the use of parallelization techniques can significantly reduce the execution time required in the stabilization process, favoring applications with time restrictions. The diagram of Figure 3.8 presents the proposed parallel video stabilization architecture.

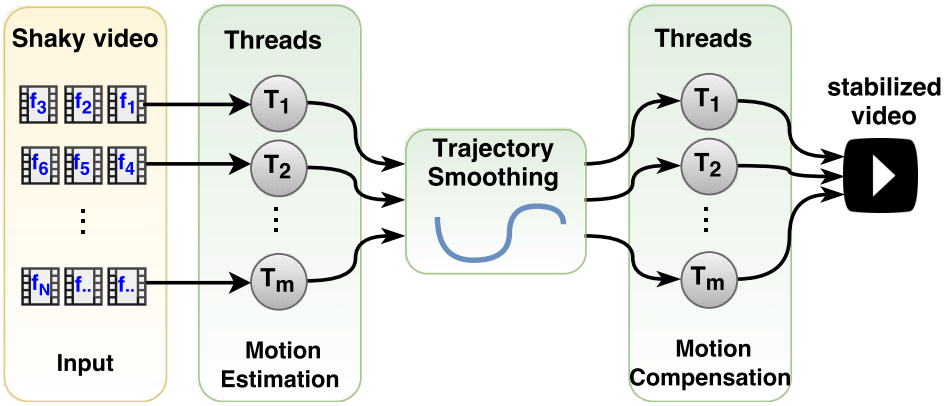


Figure A.1: Main steps of the proposed parallel video stabilization architecture.

In the initial step, k contiguous frames are assigned to i-th thread, where k can be

expressed as

$$k = \begin{cases} \frac{N}{m} & \text{if } t_i = 0 \text{ and } b_j = 0 \\ \frac{N}{m} + 1 + (N \bmod m) & \text{if } t_i = m - 1 \\ \frac{N}{m} + 1 & \text{otherwise} \end{cases} \tag{A.1}$$

where N is the number of frames in the j-th batch b loaded into memory and m the number of threads. The batches are used as a device to make it possible to process videos larger than available memory.

The i-th thread, $1 \leq i \leq m$, has as its first frame, the last of the previous thread. The first thread of each batch considers the last frame of the previous batch, for every batch b_j such that $j \neq 0$. This is done so that every pair of intermediate frames has its transformation matrix properly calculated. Thus, each thread is responsible for computing the same number of frames except the first thread of the first batch and the last of each batch.

We used the Lucas-Kanade method [73] to estimate the motion, and the mean filter for the removal of unwanted motion. Then, each thread is again responsible for the same set of frames in the video. From the original and the smoothed path, it is necessary to recalculate the value of the transformation matrix factors of each frame. At this point, each thread considers the arrays of their respective frames. With the updated matrix, this is applied to the first frame of the pair, so as to take it to the coordinates of the posterior frame. When applying the transformation to all frames, a stabilized video is obtained.

We are interested in evaluating the improvements and effects of parallelization on the video stabilization process, but not the quality of the video obtained. For this, we use two metrics: speedup and efficiency.

Speedup measures how fast an algorithm is relative to its sequential version. Ideally, speedup should be equal to the number of threads used. However, this is rare because of the overhead present in parallelism. Speedup can be expressed as

$$\text{Speedup} = \frac{T_s}{T_p} \tag{A.2}$$

where T_s is the sequential time and T_p is the time of the parallel version.

The efficiency evaluates how much the threads are used in the execution of a parallel algorithm, which can be seen as the average percentage of use of each thread, and can be expressed as

$$\text{Efficiency} = \frac{T_s}{m * T_p} \tag{A.3}$$

where m is the number of threads.

Table A.1 presents the videos considered in these experiments. The videos were taken from `YouTube` and cropped in order for them not to have abrupt frame transitions. Three different video resolutions were considered: 854×480 pixels (HQ), 1920×1080 pixels (HD), and 4096×2160 pixels (4K).

All experiments were executed on a 2.8 GHz Intel(R) Core(TM) i5 CPU with 4GB RAM using Linux 4.4.0-38 and `C++` language. The memory used to store the frames was

Table A.1: Videos considered in the experiment and their respective sources.

# Video	Source
1	youtu.be/95GlwrqE44U
2	youtu.be/W5ZxDYFMLFo
3	youtu.be/1I2nbdg0Yuw

limited to 1GB. The method was implemented in two versions: with the `pThreads` library, and with the `OpenMP` framework. Three executions were performed for each video, and the metrics were computed from the average time.

Tables A.2 and A.3 present the speedup and efficiency obtained in all videos, respectively, considering the version implemented with `pThreads`. Tables A.4 and A.5 present the results of the version implemented with `OpenMP`.

Table A.2: Speedup values for the implementation in `pThreads`.

# Video	2 Threads			4 Threads		
	HQ	**HD**	**4K**	**HQ**	**HD**	**4K**
toquio	1.986	2.072	2.113	2.743	2.746	2.863
animals	2.038	2.063	2.093	2.833	2.748	2.853
birds	2.013	2.086	2.107	2.769	2.766	2.866

Table A.3: Efficiency values for the implementation in `pThreads`.

# Video	2 Threads			4 Threads		
	HQ	**HD**	**4K**	**HQ**	**HD**	**4K**
toquio	0.993	1.036	1.056	0.686	0.686	0.716
animals	1.019	1.032	1.047	0.708	0.687	0.713
birds	1.006	1.043	1.053	0.692	0.691	0.716

Table A.4: Speedup values for the implementation in `OpenMP`.

# Video	2 Threads			4 Threads		
	HQ	**HD**	**4K**	**HQ**	**HD**	**4K**
toquio	1.898	1.903	1.975	3.024	2.911	2.923
animals	1.903	1.922	1.962	3.054	2.924	2.920
birds	1.908	1.910	1.972	3.033	2.916	2.916

From the results, we can observe that the parallelization mechanism was effective, obtaining linear speedup with two threads. There was a little change in the results for the two versions. It was expected that videos with higher resolutions would show

Table A.5: Efficiency values for the implementation in `OpenMP`.

# Video	2 Threads			4 Threads		
	HQ	**HD**	**4K**	**HQ**	**HD**	**4K**
toquio	0.949	0.952	0.987	0.756	0.728	0.731
animals	0.952	0.961	0.981	0.763	0.731	0.730
birds	0.954	0.955	0.986	0.758	0.729	0.729

better results, although this did not occur due to the hardware used. Nevertheless, the parallelization method seems to be promising.

In future experiments, we intend to consider a larger number of video sequences and a superior hardware configuration to allow us to better evaluate the results for a larger number of threads.

A.2 Objective Evaluation

Although subjective evaluation is important to observe the effectiveness of the video stabilization methods, it is indispensable to develop objective metrics that are consistent with visual perception, especially when we need to comparte large data sets.

The following subsections describe the metrics obtained with both the motion energy image and visual rhythm representations.

A.2.1 Motion Energy Image

Figure A.2 presents the histograms of the images shown in Figure 4.10, where we can easily distinguish the image from the stabilized and the non-stabilized video.

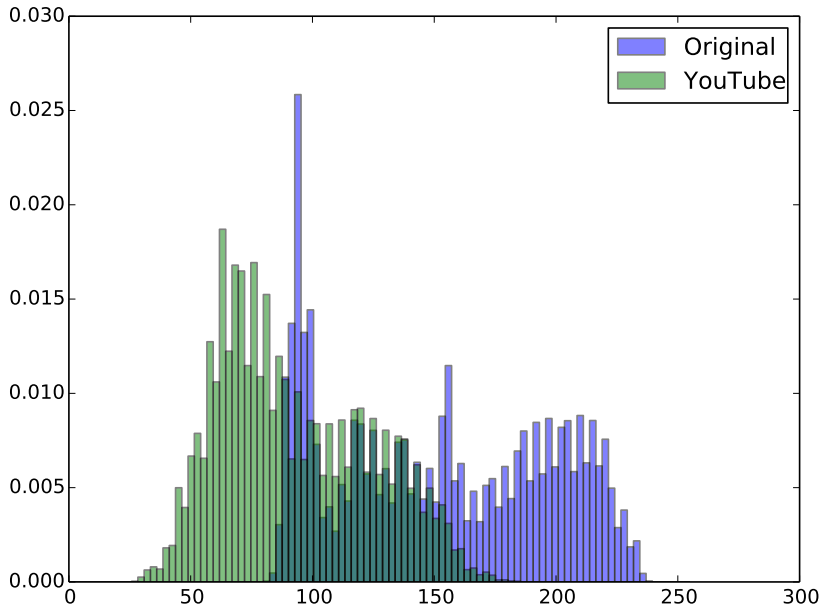


Figure A.2: Histogram of average image of MEIs for video `#4`.

We extracted statistical measurements from the graylevel image in order to obtain an objective metric that characterizes the average amount of motion (AAM) present in the video and that can be used to determine the quality of the stabilization process. For this, we consider the normalized average of the gray-level intensities, which can be expressed as

$$\text{AAM} = \frac{\sum_{x}^{W}\sum_{y}^{H} I(x,y)}{WHL_{\max}} \tag{A.4}$$

where W and H correspond to the width and height of the image, respectively, whereas $L_{\max}$ is the maximum intensity that a pixel can assume.

The AAM value is normalized between 0 and 1. Higher values indicate a greater amount of motion. Typically, a more stable video should generate a lower AAM value than its unstable version. For visualization purpose, we used the AAM to compare videos before and after the stabilization process. Therefore, we need not be concerned with the interference of moving objects, since this will occur in both videos.

Table A.6 displays the values of AAM, as well as the ITF values for the original videos and after the `YouTube` stabilization method [36].

Table A.6: AAM and ITF values for the videos from the first dataset.

# Video	Original		YouTube	
	ITF	**AAM**	**ITF**	**AAM**
1	18.793	0.758	27.890	0.557
2	20.390	0.671	28.604	0.456
3	16.186	0.815	23.030	0.646
4	19.965	0.596	33.711	0.364
5	23.277	0.683	27.599	0.574
6	19.681	0.828	29.390	0.547
7	24.109	0.627	29.252	0.417
8	17.881	0.775	25.908	0.578
9	19.248	0.800	20.922	0.746
10	12.972	0.847	20.495	0.718
11	21.487	0.671	26.672	0.550
12	15.081	0.840	19.283	0.756
13	23.841	0.598	28.845	0.462
14	18.065	0.650	20.128	0.590
Mean	19.355	0.726	25.837	0.569

Table A.7 shows the AAM and ITF values for the tested videos before and after the stabilization process. Both versions are available in the database. The stabilized version was originally obtained with the method proposed by Liu at al. [69].

From Tables A.6 and A.7, it can be noticed that the proposed metric has consistent results with the ITF metric in the evaluated videos, which demonstrates that it can be used as an alternative to the ITF. From Table A.7, we can see that the mean value of

Table A.7: Mean AAM and ITF values for the videos from the second dataset.

Category	Original		Stabilized	
	ITF	**AAM**	**ITF**	**AAM**
Crowd	19.479	0.787	23.699	0.731
Parallax	18.746	0.741	22.499	0.684
Quick Rotation	19.963	0.758	24.166	0.705
Regular	19.468	0.713	28.248	0.567
Running	17.326	0.828	22.765	0.750
Zooming	20.113	0.746	24.630	0.684
Mean	19.183	0.762	24.334	0.687

AAM is lower in `Regular`, `Zooming` and `Parallax` categories, which have a lower amount of movement in their videos.

Table A.8 presents the values of AAM and ITF metrics for video `#4` stabilized through a simple method, where a Gaussian smoothing filter is applied with different values of σ.

Table A.8: AAM and ITF values for video `#4`.

Gaussian $\sigma = 10$		**Gaussian** $\sigma = 40$		**Gaussian** $\sigma = 890$	
ITF	**AAM**	**ITF**	**AAM**	**ITF**	**AAM**
26.840	0.488	33.380	0.385	33.865	0.375

From Table A.8, it can be seen that the ITF and AAM values decrease with the increase of σ. This occurs because the method considered the motion as undesired and corrected most of the motion with increasing σ. It is possible to observe that, with $\sigma = 890$, the ITF obtained with the Gaussian filter is superior to that obtained with `YouTube` method. However, the video generated with the Gaussian filter is visually more unstable, containing several distortions.

The AAM values, also reported in Table A.7, are not smaller than the value obtained with the `YouTube` method and, therefore, more consistent with the visual result of the video.

A.2.2 Visual Rhythm

In the visual rhythm, the behavior of the movement present in the video is represented by the shapes of the curves. A more stable video has rhythms with smoother curves. As shown in Figure 3.13, the directions of the visual rhythm can be checked in each column pair of pixels. We believe that a softer visual rhythm has more regular directions, with less abrupt changes in the directions that are near. Thus, to get a new objective metric from the visual rhythm, the directions and then calculate their changes must be computed. Figure A.3 illustrates the method for calculating the metric.

Initially, we calculate the visual rhythm gradients, in order to obtain the directions

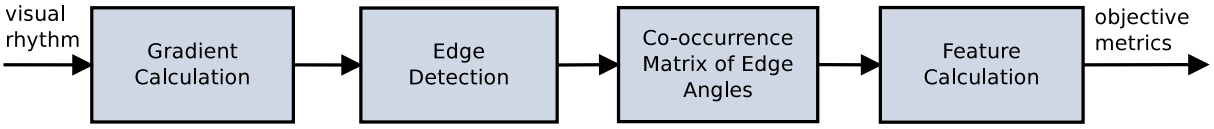


Figure A.3: Main steps of the objective metric method.

of each pixel of the rhythm. This was implemented through the Sobel filter [98]. The gradients are decomposed into magnitude and angle.

A thresholding with the Otsu algorithm [83] is applied to the magnitudes in order to determine the edges of the visual rhythm. This is done in order to consider only the edge angles in the following calculations. Then, a co-occurrence matrix is calculated. This matrix is based on the grey level co-occurrence matrix (GLCM) [39]. However, it considers the co-occurrence of the angles of the edges, in the direction of the angles themselves.

Initially, we eliminate the sign from the angles, leaving them in the range of 0 to 180 degrees. For the calculation of the co-occurrence matrix M, we consider n directions $D = \{d_0, d_1, ..., d_n\}$, resulting in a matrix of size $n \times n$. The angles are then quantized in possible directions. For each pixel i belonging to the edge, we have its angle $\theta_i \in D$, from which we calculate the nearest pixel j in the direction of θ_i itself. Then, it counts as a co-occurrence at the position M_{θ_i,θ_j} increasing this position of the matrix. For cases where θ_i are different from the important angles, we have two pixels j_1 and j_2. Thus, the two positions of the matrix are incremented proportionally to the distances of the angles.

Finally, the matrix is normalized by the sum of its elements. Thus, the value of the matrix in the position M_{θ_i,θ_j} indicates the probability that θ_j is the next direction of the visual rhythm, since the previous one was θ_i. From the generated co-occurrence matrix, we can calculate features in order to obtain objective metrics. Among the textural features defined by Haralick and Shanmugam [39], homogeneity can be expressed as

$$\text{homogeneity} = \sum_{i=0}^{n}\sum_{j=0}^{n} \frac{1}{1+(i-j)^2} M_{i,j} \tag{A.5}$$

The homogeneity feature, when calculated from the co-occurrence matrix of the edge angles, will assume greater values as closer the angles of consecutive directions. Table A.9 reports the results of the homogeneity for the visual rhythms (horizontal and vertical) for the video sequences shown in Table 4.1 for the original videos stabilized by the `YouTube` method [36]. We can notice that the obtained results can distinguish original and stabilized videos. However, we must further investigate the extraction of other features from the co-occurrence matrix, which may be complementary to the homogeneity feature.

A.3 Local Motion

Local motion estimation is important for both non-rigid oscillations and parallax effects. The following subsections present methods for both local motion estimation through optimization mechanisms and frame transformation based on the local motion.

Table A.9: Results of homogeneity for video sequences.

# Video	Original		YouTube	
	horizontal	vertical	horizontal	vertical
1	0.446	0.449	0.737	0.773
2	0.404	0.478	0.685	0.741
3	0.392	0.473	0.647	0.708
4	0.409	0.425	0.665	0.742
5	0.511	0.575	0.715	0.649
6	0.481	0.627	0.678	0.734
7	0.423	0.446	0.652	0.615
8	0.409	0.508	0.697	0.672
9	0.499	0.523	0.549	0.625
10	0.387	0.441	0.588	0.620
11	0.609	0.557	0.743	0.636
12	0.515	0.474	0.535	0.678
13	0.499	0.441	0.665	0.614
14	0.479	0.618	0.521	0.750
Average	0.461	0.502	0.648	0.682

A.3.1 Local Motion Estimation

When using the optimization method proposed in this work for the correction of global motion estimation, there is no longer a set of local features to allow the construction of meshes proposed by Liu et al. [69]. Figure A.4 illustrates the method for constructing the meshes in these cases based on optimization.

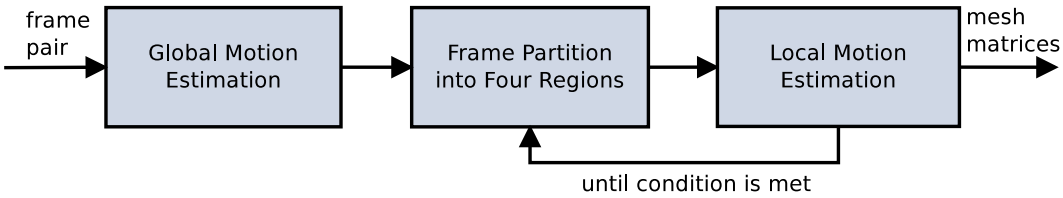


Figure A.4: Main steps of the local motion estimation method.

Initially, the overall estimate of the pair of frames is calculated by the optimization method, as presented in Chapter 3. Then the frame is divided into four regions of same size and the motion is estimated for each of them separately. This new estimation is also calculated through optimization, considering the global motion estimation as the initial guess. This is done recursively until the stop criterion is satisfied. A minimum size of the frame region is used as stopping criterion, such that it is not further divided.

The estimation must be calculated in such a way that (i) the consecutive frames are similar, (ii) that the local motion is not very different from the global motion, and (iii) that the motion of neighbor cells are similar. For this, we define the objective function as

$$f = (\alpha)(1 - \text{mean}(S)) + (1 - \alpha)(d(M_0, M_{new})) \tag{A.6}$$

where mean$_(S)$ is the mean value of SSIM between the reference frame and the transformed frame, considering only the determined region of the frame, whereas $d(M_0, M_{new})$ is the Euclidean distance between the initial guess matrix M_0 and the estimated matrix M_{new}. The SSIM value guarantees the similarity of the frames, whereas the use of the distance between the matrices is intended to meet the two other requirements. In addition, the transformation presented later tends to force even more than the motions of cells are close.

A.3.2 Frame Transformation

After estimating the local motion, it is necessary to transform the frame in a non-linear way, however, without losing its shape. For this, Liu et al. [69] used an optimization approach in the unwanted motion removal step, that approximates the smoothed paths of each cell, as presented in Equation 2.14. Then, the matrices of each cell are applied directly to the frame. However, this does not allow the use of techniques such as the adaptive Gaussian filter proposed in our work. Furthermore, it did not present good results in our experiments. In our method, each pixel will have a distinct transformation matrix, which can be defined as

$$H(i) = \frac{\sum_{c \in \Omega_i} \frac{1}{\lambda_{i,c}} M_c}{\sum_{c \in \Omega_i} \frac{1}{\lambda_{i,c}}} \tag{A.7}$$

where $H(i)$ is the transformation matrix of the pixel i. The set Ω_i comprises the neighboring cells of the cell in which the pixel i is present, in addition to the cell of the pixel itself. The matrix M_c is the transformation matrix of cell c, whereas $\lambda_{i,c}$ is the Euclidean distance between the pixel i and the central pixel of cell c.

Applying such transformation allows the frames to be transformed non-linearly without deforming it. Figure A.5 illustrates overlapping original frames both after the global and local transformations. From the frames presented, it is possible to notice that the method was able to improve the global motion estimation, correcting some of the transformations in certain points of the image without undoing the match in points that were correct.

As future investigation, we intend to refine the local motion estimation method, as well as the frame transformation approach. In addition, several experiments will be conducted to demonstrate the effectiveness of the proposed motion estimation strategies.

(a) original (b) global motion (c) local motion

Figure A.5: Comparison between global and local transformations for video #10.