Motion Estimation Reliability and the Restoration of Degraded Archived Film

A dissertation submitted to the University of Dublin
for the degree of Doctor of Philosophy

David Corrigan
University of Dublin, Trinity College, July 2007

Signal Processing and Media Applications
Department of Electronic and Electrical Engineering
Trinity College Dublin

To my family.

Abstract

The motivation for this thesis has been to improve the robustness of image processing applications to motion estimation failure and in particular applications for the restoration of archived film. The thesis has been divided into two parts.

The first part is concerned with the development of an missing data detection algorithm that is robust to Pathological Motion (PM). PM can cause clean image data to be misdiagnosed as missing data. The proposed algorithm uses a probabilistic framework to jointly detect PM and missing data. A five frame window is employed to detect missing data instead of the typical three frame window. This allows the temporally impulsive intensity profile of blotches to be distinguished from the quasi-periodic profile of PM. A second diagnostic for PM is defined on the local motion fields of the five frame window. This follows the observation that Pathological Motion causes the *Smooth Local Flow Assumption* of motion estimators to be violated. A ground truth comparison with standard missing data detectors shows that the proposed algorithm dramatically reduces the number of falsely detected missing data regions but also results in an increased missed detection rate.

The second part is concerned with the applications of Global Motion Estimation. The first contribution of this part is to introduce a robust and precise algorithm for Global Motion Estimation (GME) using motion information from an MPEG-2 stream, which uses a hybrid of two existing GME algorithms. The first algorithm estimates the parameters from the block based motion field obtained from an MPEG-2 stream and also generates a coarse motion-based background segmentation of the frame. The second is a gradient-based technique which estimates the parameters from the image data directly. In this work, an initial estimate for the global motion is obtained using the first approach. The parameters are then refined using the gradient based technique, producing a final global motion estimate in the form of a six-parameter affine model. Unlike existing gradient-based techniques, the motion-based segmentation is used to weight out local motion rather than the size of the image residuals. The performance of the hybrid algorithm is compared to an existing gradient-based technique over a range of challenging sequences and is shown to be more robust to local motion than the existing approaches. The hybrid algorithm is then applied to the problem of mosaicking in sports sequences in which the global motion parameters can be used to make a panorama of an entire shot.

The final contribution of the thesis is a new algorithm to segment frames affected by film tear. Film Tear is a form of degradation in archived film and is the physical ripping of the film material. Tear causes displacement of a region of the degraded frame and the loss of image data along the boundary of tear. In [21], a restoration algorithm was proposed to correct the displacement in the frame introduced by the tear by estimating the global motion of the 2 regions either side of the tear. However, the algorithm depended on a user-defined segmentation to divide the frame. This thesis presents a new fully-automated segmentation algorithm which divides affected frames along the tear. The algorithm employs the graph cuts optimisation technique and uses temporal intensity differences, rather than spatial gradient, to describe the boundary properties of the segmentation. Segmentations produced with the proposed algorithm agree well with the perceived correct segmentation. A strength of the algorithm is that, although the segementation can proceed automatically, a user initialised segmentation is still possible in cases where the automatic segmentation fails.

Declaration

I hereby declare that this thesis has not been submitted as an exercise for a degree at this or any other University and that it is entirely my own work.

I agree that the Library may lend or copy this thesis upon request.

Signed,

David Corrigan

July 9, 2007.

Acknowledgments

As the seemingly never ending endeavour of writing my thesis draws to a close, I would like to thank all the people I have had the pleasure of working with over the years. In particular, I would like to thank my supervisor, Dr. Anil Kokaram for his time, advice and patience. Gratitude must also go to my co-supervisor, Dr. Naomi Harte, for all the help and advice she has given, in particular when reviewing my publications and thesis. Thanks must also go to all past and present members of the SIGMEDIA group who have assisted me in my work, especially to Dr. Hugh Denman, Dr. Francis Kelly and Dr. Francois Pitie. A special mention to Dan, Daire, Dee, Gary, Akash, Ric, John, Gavin, Rozenn, Phil, Guillaume, Agnes, Robbie, Conor, Nora, Tim, Linda, Brian and Frank and everyone else in the lab and department for making such a friendly and enlightening working environment.

This work was funded in part by the Embark Initiative, operated by the Irish Research Council for Science, Engineering and Technology and funded by the state under the national development plan. Thanks to everyone involved for their foresight.

Finally, I would like to thank my family and friends. In particular to my parents, for all their encouragement, understanding and support they given me.

Thank you all very much.

Contents

List of Acronyms

1D 1 Dimensional

2D 2 Dimensional

3D 3 Dimensional

AR AutoRegessive

BM Block Matching

BBC British Broadcasting Corporation

DFT Discrete Fourier Transform

DFD Displaced Frame Differences

PM Pathological Motion

GME Global Motion Estimation

GOP Group Of Pictures

IR Infra-Red

ICM Iterated Conditional Modes

IRLS Iteratively Re-weighted Least Squares

LMedS Least Median of Squares

MRF Markov Random Field

MAP Maximum A Posteriori

ML Maximum Likelihood

MDD Missing Data Detection

MDI Missing Data Interpolation

MDT Missing Data Treatment

MPEG Moving Pictures Expert Group

Pel Pixel or Picture element

PDF Probability Density Function

ROD Rank Order Difference

ROC Receiver Operating Characteristic

RGB Red Green Blue

RAGMOD Robust Accumulation for Global Motion field mODelling

sROD simplified Rank Order Difference

SDI Spike Detection Index

TDF Temporal Discontinuity Field

1
Introduction

The rise of Digital Visual Transmission in last decade of the 20^{th} century dramatically changed the landscape of the broadcasting industry for content providers, broadcasters and viewers. It allowed for a dramatic increases in the number of available broadcast channels available, increasing opportunities for providers and promised higher visual quality for viewers, free from the noise associated with the old analogue systems. With the introduction of Digital TV in particular, broadcasters realised that the value of archives had greatly increased as they were required to provide the content for all the extra broadcast channels.

However, as broadcasters turned to archives to fill the extra channels two things became apparent. As archived media had been subjected over the years to wear and tear, the quality of the material had been reduced. Consequently, archived footage often fails to live up to the expectations of visual quality in the digital age. The second consideration is that degraded footage consumes more bandwidth than higher quality material, making it more expensive to broadcast. Hence, a requirement for digital restoration of archived media was born.

A wide variety of visual impairments have been observed in archived media[1]. These include film shake caused by unintentional camera motion or problems during scanning. This can be observed in image sequences as a high frequency global motion. Another degradation type is film flicker which manifests as brightness fluctuations in image sequences. Dirt & Sparkle are frequently occurring degradation artefacts. It occurs when material adheres to the film surface or when the film is abraded. It also is commonly referred to as blotches. The visual impact is that of temporally impulsive dark and bright spots distributed randomly across an image

[1]See [57] for a more extensive taxonomy of degradations found in archived media

sequence. Essentially blotches are a missing data problem as the spots are hiding patches of the true image.

Digital restoration of these artefacts often depends on the estimation of object motion in sequences to achieve a reliable and high quality restoration. For instance, knowledge of the global motion of a sequence is required to stabilise image sequences effected by shake. However, these restoration algorithms assume that all the motion in the image sequence can be reliably estimated. Consequently, the quality of motion estimation is a key factor affecting the quality of restoration. Poor quality restoration is often caused by failures in motion estimation.

The motivation behind this work has been to develop an understanding of the causes of motion estimation failure and to improve the robustness of applications dependent on motion estimation, especially those in the restoration domain. This can be achieved by either developing more robust motion estimation algorithms or, where that is not possible, by detecting regions of motion estimation failure and by treating such regions as special cases.

1.1 Pathological Motion Detection in Missing Data Treatment

Missing Data Treatment is a restoration application which refers to the removal of temporally impulsive blotches from image sequences. In the 1980's, a BBC research team led by Richard Storey [98] was the first to consider that blotches could be digitally restored. However, this work focused on blotch removal without motion compensation. Kokaram [52] recognised that this approach could be enhanced by considering motion estimation as an integral part of the process, especially for the detection of larger blotches.

However, like all applications dependent on motion estimation, missing data treatment algorithms are susceptible to motion estimation failures. In particular, the certain motions of objects cannot be estimated accurately no matter how sophisticated the motion estimation algorithm is. Such motion is an issue as it often causes false detections of blotches, resulting in destruction of true image data. This motion is known as Pathological Motion(PM) [8, 84].

The first contribution of this thesis is to develop a new missing data treatment algorithm which is robust to Pathological Motion. The strategy employed is to detect regions of Pathological Motion and to reject these regions as possibilities for blotch detection, thereby eliminating false positives caused by motion estimation failure. This distinction is made possible by considering more sequence data, an approach first adopted by Bornard [8]. The detection is augmented by diagnosis of PM on the generated motion information. Detection is made using a probabilistic framework derived using Bayes Law which allows both the observed data and prior knowledge of blotches and PM to be considered.

1.2 Global Motion Estimation

Global Motion Estimation (GME) is the process by which the frame-to-frame camera motion is computed from the image sequence data. It has been heavily used in the restoration world to mitigate against the effects of shake. However, with the advent of the MPEG standards in video compression, it has gained prominence as a means of improving the compression ratios of sequences.

This thesis addresses some of the issues which affect the robustness of Global Motion Estimation algorithms. The main concern is the impact of local motion (*i.e.* motion that is independent of the global or camera motion) which often acts as a bias in the result. A popular method of diagnosis of local motion is based on the computation of camera-motion compensated intensity differences. However, this approach assumes knowledge of an approximation to the global motion. When this approximation is erroneous, the diagnosis of local motion breaks down, resulting in further inaccuracies in the estimate. The contribution of this thesis is to adopt a diagnosis technique which is not so reliant on knowledge an accurate approximation to the camera motion, ensuring that this kind of failure is avoided. The resulting algorithm is a hybrid approach to global motion estimation, for which the chosen application is the computation of mosaics from image sequences.

1.2.1 Film Tear Restoration

Film tear is a degradation artefact in archived film caused by the physical ripping of the film material, dividing the film in two. Principally, film tear restoration is a problem of Missing Data Treatment since the tearing results in the destruction of image data. However, tear restoration cannot be considered purely as a missing data problem, as it also results in a regional displacement induced by the separation of the two halves of the film. This issue must be addressed before any Missing Data Treatment process is employed. Displacement correction is in essence a problem of global motion estimation since knowledge of the global motion of each region is sufficient for correction.

Correction of this regional displacement was a problem first addressed in [21]. It outlines a framework to detect film tear, to divide torn frames and to estimate and correct the induced displacement. The contribution of this thesis is to propose a novel frame segmentation algorithm to isolate the regions of the frame separated by the tear. The algorithm is derived from the interactive segmentation technique outlined by Boykov *et al.* in [11] which requires a description of the region and boundary properties of the desired segmentation. The boundary conditions of the segmentation are characterised using knowledge of the image intensities of the tear region. The segmentation also draws upon some of the local motion diagnosis techniques used in the hybrid GME algorithm to describe the region properties of the segmentation.

1.3 Thesis outline

The thesis is divided into two parts along the lines of the above sections. Part I (chapters 2 & 3) deals with the issue of Pathological Motion, while part II (chapters 4, 5 and 6) outlines the work on Global Motion Estimation. The novel contributions of the thesis are outlined in chapters 3, 5 and 6. The following is a summary of each chapter of the thesis.

Chapter 2: An Introduction to the Problem of Pathological Motion in Missing Data Treatment

This chapter provides a description of the phenomenon of Pathological Motion and how it affects Missing Data Detection algorithms. First of all, an brief outline of popular motion estimation techniques are outlined. This is followed by a description of the causes of motion estimation failure in general and of Pathological Motion in particular. Finally the chapter outlines the state of the art in missing data detection, describing both the classic techniques that assume perfect knowledge of the motion and the newer techniques which allow for the presence of Pathological Motion.

Chapter 3: Pathological Motion Detection for Robust Missing Data Treatment

This chapter outlines the novel blotch detection algorithm. It provides a description of the algorithm as well as description of the experiments performed to evaluate it. The performance of the algorithm is compared to a number of existing blotch detectors. A ground truth for blotches was available and each algorithm is compared in terms of its correct detection and false alarm rates.

Chapter 4: A Brief Review of the Global Motion Estimation State of the Art

This chapter introduces some of the Global Motion Estimation algorithms described in the literature. It also discusses the limitations of the existing techniques and introduces the concept of a hybrid approach to Global Motion Estimation.

Chapter 5: A Hybrid Algorithm for Robust and Precise Global Motion Estimation from MPEG Streams

The novel hybrid GME algorithm is introduced in this chapter. It gives an in-dept description of the algorithm and how it is applied to the process of mosaicking from MPEG-2 streams. A ground truth evaluation of the algorithm is performed and it is also compared to an existing prominent global motion estimation technique.

Chapter 6: Global Motion Estimation in Restoration: Automated Film Tear Restoration

The novel torn frame segmentation algorithm is described here. An outline of the tear restoration framework is presented and is followed by a detailed description of the segmentation algorithm. Summaries of the results of the segmentation and the overall restoration framework are outlined. The final sections of the chapter outline a further possible application of the algorithm, namely to the detection of dirty splices in iamges.

Chapter 7: Conclusions

The final chapter assesses the contributions of this thesis and outlines some directions for future work.

1.4 Publications

Portions of the work described in this thesis have appeared in the following publications:

[19] D. Corrigan, N. Harte and A. Kokaram. Pathological Motion Detection for Robust Missing Data Treatment. *In the IEEE International Conference on Image Processing*, pages 621-624, 2006.

[23] D. Corrigan, A. Kokaram, R. Coudray and B. Besserer. Robust Global Motion Estimation From Mpeg Streams with a Gradient Based Refinement. *In the IEEE International Conference on Acoustics, Speech and Signal Processing*, volume 2, pages II-285 - II-288, 2006.

[20] D. Corrigan and N. Harte and A. Kokaram. Automated Segmentation of Torn Frames using the Graph Cuts Technique. *To appear in the IEEE International Conference on Image Processing*, 2007.

Other Publications

[21] D. Corrigan and A. Kokaram. Automated Tear Treatment in Degraded Archived Media. *In the IEEE International Conference on Image Processing*, volume 3, pages 1823-1826, 2004.

[22] D. Corrigan and A. Kokaram. Detection and Treatment of Film Tear in Degraded Archived Media. *In the IEEE International Conference on Image Processing*, volume 4, pages 779-782, 2004.

Part I

Pathological Motion Detection

2 An Introduction to the Problem of Pathological Motion in Missing Data Treatment

The treatment of missing image data is one of the most common problems in the restoration of archived film. Degradation artefacts known as blotches (also called dirt and sparkle) appear at random positions in the frame as regions of high spatial contrast to the neighbouring pixels and replace the true image data at their location (Fig. 2.1). Blotches can be of varying shapes and sizes, although the majority of blotches take up less than 10×10 pel in size. Usually, the intensities of the blotches are at the extrema of the intensity range. The intensities for dirt, which is the result of accumulation of particles on the film surface, are typically low, while sparkle, caused by abrasion of the film emulsion, is normally associated with high intensities. Another important characteristic of blotches is that they rarely occur in the same location in consecutive frames. Consequently, blotches cause temporal discontinuities in the image intensity function in both the forward and backward directions and are often considered as being temporal impulses. The goal of Missing Data Treatment (MDT) algorithms in restoration is to remove these blotches from a sequence. Typically, this problem is broken down in to 2 stages,

1. to detect the locations of the blotches which is referred to as **Missing Data Detection** (MDD). Missing data detectors commonly exploit the temporal characteristic of blotches, specifically that blotches cause temporal intensity discontinuities [15, 61, 75, 89, 92, 98, 104]. Due to the variable size, shape and intensity statistics of blotches, the use of purely spatial methods is not suitable for the detection of blotches.

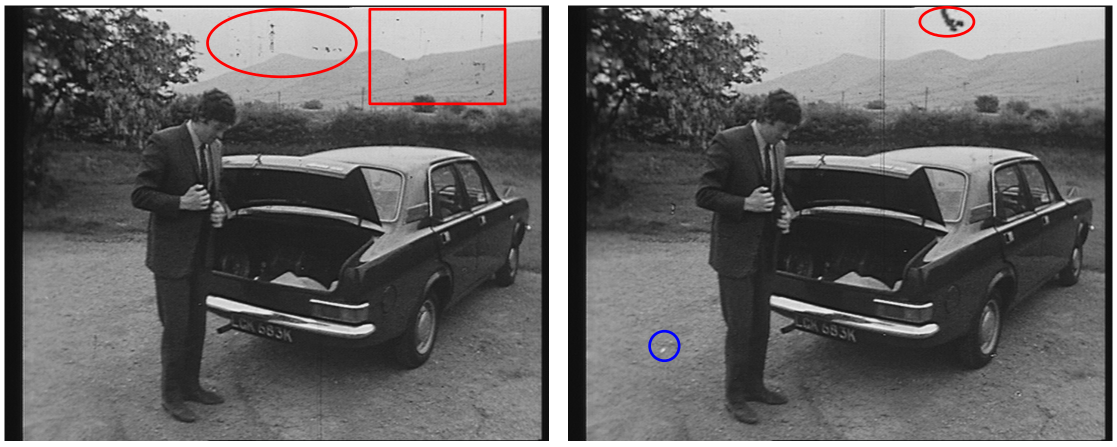


Figure 2.1: *Each image highlights a frame with blotches present. The red shapes highlight regions with dirt which are the dark spots in the highlighted regions. The blue circle highlights an area with sparkle present which is the bright spot in the circle.*

2. to find the true image data at these locations (also known as **Missing Data Interpolation** (MDI)). While purely spatial techniques for interpolation do exist (*e.g.* image inpainting [87] and texture synthesis [34,36]), most MDI algorithms exploit the temporal redundancy of image sequences and also consider data from neighbouring frames [57,60,62].

Motion Estimation is a vital component in MDT algorithms. It is employed in detection to compensate for object or camera motion, ensuring that false alarms do not occur due to their motion. Accurate Motion Estimation also ensures that the correct temporal data is used to "fill in" blotches in the interpolation stage. However, any improvement in the robustness of the algorithm is mitigated if the estimated motion is not accurate. Bad motion estimates can cause clean areas of the image to be flagged as a blotch and will cause blotches being "filled in" with the wrong data (Fig. 2.2).

For the purposes of this work, the causes of motion estimation failures are broken down into three main categories.

1. Motion estimation failure can be caused by the artefacts themselves. As blotches are impulsive defects, they obviously cannot be tracked and so will cause motion estimation failure. Large blotches, especially those larger than the block size of the motion estimator, significantly bias the value of the estimated motion. This problem was taken into account by Kokaram *et al.* by designing integrated MDT algorithms which refines the motion field, as well as detecting and recovering the missing data, in an iterative fashion [53,57,60].

2. Failures can occur when the quality of the image data is not suitable for accurate estimation of the motion. There are certain spatial textures of the image data, from which an unambiguous estimate of the motion can not be estimated. Such data is referred to as

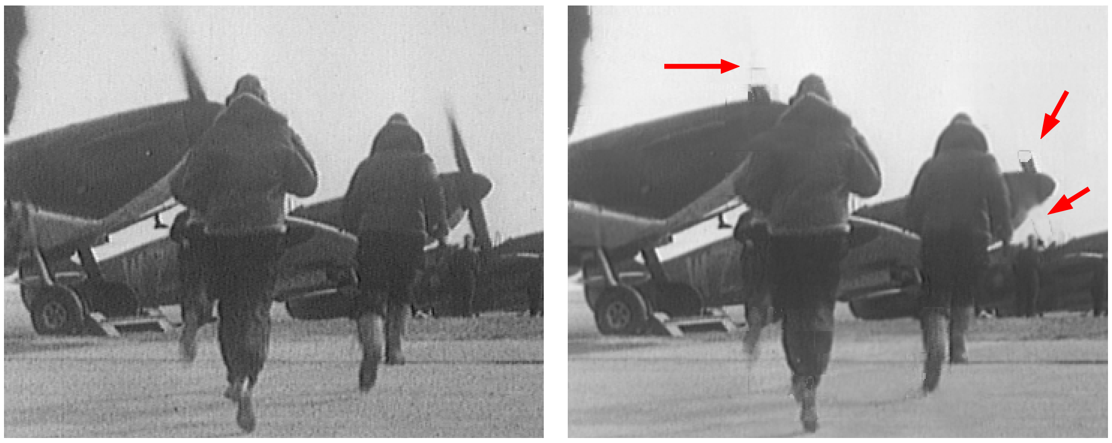


Figure 2.2: *Left: A frame of a sequence where the the propeller blades is not accurately estimated; Right: The image after MDT using [60]. The arrows indicate the removal of the blades due to the blades being misdiagnosed as missing data.*

ill-conditioned image data [48, 54]. Examples of these textures include flat homogeneous regions of an image (*e.g.* a clear sky), straight object edges and periodic textures. The aperture effect (Fig. 2.3) is a well-known example of ill-conditioning in motion estimation and is due to the block size of the motion estimator[1] (*i.e.* the aperture) is too small to allow an unambiguous estimate of the motion to be found.

3. Failure can be caused by a phenomenon referred to as Pathological Motion (PM). PM is associated with erratic motion patterns which cannot be estimated accurately by a motion estimator (Section 2.2). It also causes false alarms in MDD, resulting in damage to clean image data during MDT (Fig. 2.2).

This thesis is concerned with improving the robustness of MDD algorithms to Pathological Motion. In this chapter the notion of Pathological Motion and how, in the past, PM has been addressed in the context of blotch detection. First of all, a brief overview of motion estimation is presented which describes the most popular motion estimation techniques. This is followed by an introduction to Pathological Motion which intoduces some of the motion patterns which are commonly pathological. In section 2.3 a review is presented of some existing MDD algorithms which do not include models for PM. Finally, the state of the art of PM detection for Missing Data Detection is presented.

2.1 An Introduction to Motion Estimation

Motion estimation is a vital tool in many digital video processing applications and has been a prominent research here. This section gives an overview of the most prominent motion estimation

[1]See Section 2.1

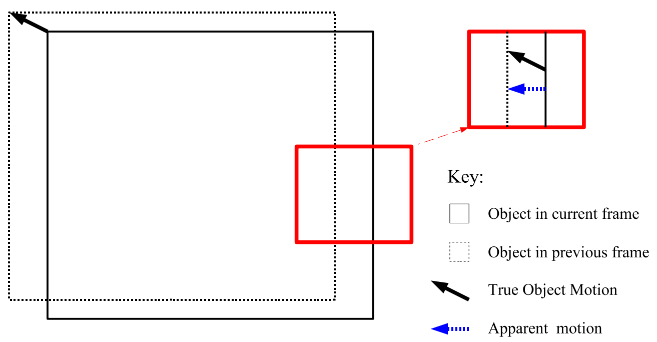


Figure 2.3: *This figure demonstrates the aperture effect which is an example of motion estimation failure due to ill-conditioned data. Because the aperture (the red box), in which the motion of the square is being estimated, is too small, the true motion of the square cannot be estimated. It appears from the aperture that the line is moving horizontally. However, the vertical motion of the line cannot be determined. The motion could be estimated accurately by estimating motion with the aperture centred on one of the corners or by increasing the aperture size.*

---

strategies and discusses their common traits. A more comprehensive review of motion estimation algorithms can be found in one of the many detailed review publications in the literature (*e.g.* [49, 54, 70, 97, 100]).

In order to estimate motion in image sequences, a model must be defined which describes the relationship between neighbouring frames. Such an image sequence model is given by

$$\begin{aligned} I_n(\mathbf{x}) &= I_{n-1}(f(\mathbf{x})) + e(\mathbf{x}) \\ &= I_{n+1}(g(\mathbf{x})) + e(\mathbf{x}) \end{aligned} \tag{2.1}$$

where I_n is the intensity function for frame n of the sequence, where $f(\mathbf{x})$ and $g(\mathbf{x})$ are functions describing the motion between frame n and frames $n-1$ and $n+1$ respectively at a pixel location $\mathbf{x}$ and $e(\mathbf{x})$ is the gaussian distributed error of the model. In other words, the intensity every pixel in the current image is the same intensity as a pixel at some location in both the current and previous frames plus some error.

Estimators fall into one of two categories. The first is Global Motion Estimation (GME), which attempts to find a single set of motion parameters that describes the motion of the entire frame. GME is not discussed further in this chapter, but Part II of this thesis discusses this topic in more detail. The second class of motion estimator, known as local motion estimation or optic flow, allows each pixel to have different motion parameters. In this way, the motion of individual objects in a sequence can be measured. The motion model for each pixel is usually chosen to be purely translational which results in an image sequence model given by

$$\begin{aligned} I_n(\mathbf{x}) &= I_{n-1}(\mathbf{x} + \mathbf{d}_{n,n-1}(\mathbf{x})) + e(\mathbf{x}) \\ &= I_{n+1}(\mathbf{x} + \mathbf{d}_{n,n+1}(\mathbf{x})) + e(\mathbf{x}) \end{aligned} \tag{2.2}$$

where $\mathbf{d}_{n,n-1}(\mathbf{x})$ and $\mathbf{d}_{n,n+1}(\mathbf{x})$ are the translational displacements between frames $n-1$ and n and between $n+1$ and n respectively. $\mathbf{d}_{n,n-1}(\mathbf{x})$ and $\mathbf{d}_{n,n+1}(\mathbf{x})$ describe the backward and forward motion fields for the frame n and are vector valued functions of the form $\mathbf{d}(\mathbf{x}) = [\mathbf{d}_x(\mathbf{x}),\ \mathbf{d}_y(\mathbf{x})]$. By defining the Displaced Frame Difference (DFD) as follows

$$\Delta_{n,n-1}(\mathbf{x}) = I_n(\mathbf{x}) - I_{n-1}(\mathbf{x} + \mathbf{d}_{n,n-1}(\mathbf{x})), \tag{2.3}$$

equation 2.2 can be expressed in the form

$$\begin{aligned} \Delta_{n,n-1}(\mathbf{x}) &= e(\mathbf{x}) \\ \Delta_{n,n+1}(\mathbf{x}) &= e(\mathbf{x}). \end{aligned} \tag{2.4}$$

The size of the DFD describes the error in the motion model. A small DFD value at a pixel site suggests that the estimated motion vector accurately describes the motion of the pixel, larger values indicate that the estimated vector is not accurate.

Independently estimating a motion vector for every single pixel is an ill-posed problem and another piece of information on the nature of motion fields needs to taken into account. This is provided from the observation that in a neighbourhood the motion field is smooth, an observation known as the *Smooth Local Flow Assumption*. Instead of estimating a vector for each pixel, the image is divided into blocks of $N \times N$ pixels and a single vector is estimated which describes the motion of a block. The choice of block size is significant the performance of the estimator. Choosing a larger block size makes the estimator more robust to image noise and to ill-conditioned data. However, if more than one object is moving within the block, then the estimated motion is not likely to match the motion of either object.

There are many classes of motion estimation algorithms, the most popular of which are

- Block Matching
- Gradient-Based Motion Estimation
- Phase Correlation

Block Matching

Block Matching (BM) algorithms estimate motion of each block in the current frame by finding the most similar block in a reference frame. The estimated motion vector is the relative position between the block in the current frame and the block in the reference frame (Fig. 2.4). Suitable similarity measures include the Mean Square Error (MSE) or Mean Absolute Difference (MAD)

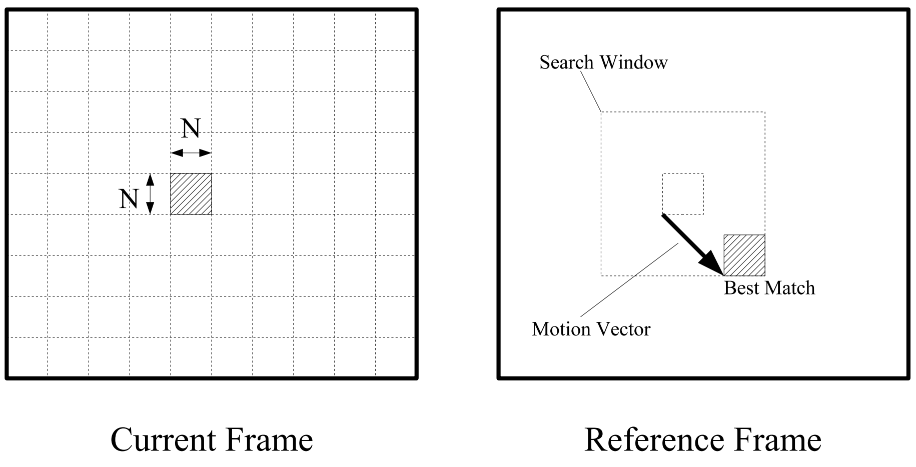


Figure 2.4: *This diagram represents the BM technique using a full motion search. The motion of each block in the current frame is estimated by finding the most similar block in the reference frame within a user-defined search window.*

of the DFD of the block, and the most similar block is the reference block that minimises the similarity measure. An obvious strategy for performing Block Matching is to test every possible vector in a user-defined range. Known as a Full Motion Search, this strategy, selects the motion vector in the range with the optimum similarity measure. The main disadvantage of the Full Motion Search is the heavy computational load involved in taking the similarity measures for every possible motion vector. Computational complexity increases with increasing motion vector range and depends on the motion vector precision, which is also user defined. In an effort to reduce the computational load, algorithms have been proposed which use alternative search strategies to the Full Motion Search [38, 67], searching only a subset of the candidate motion vectors in the given range. Such methods are no longer guaranteed to find the vector with the optimum similarity measure.

Gradient-Based Methods

An alternative strategy to block matching is gradient-based motion estimation which solves for the motion $\mathbf{d}$ of each block[2] by expressing the right hand side of equation 2.2 using a Taylor Series expansion as follows

$$I_n(\mathbf{x}) = I_{n-1}(\mathbf{x}) + \mathbf{d}^T \; \nabla I_{n-1}(\mathbf{x}) + \epsilon(\mathbf{x}) \tag{2.5}$$

where $\nabla I_{n-1}(\mathbf{x})$ is the spatial gradient of $I_{n-1}(\mathbf{x})$. The Taylor Series expansion allows direct access to $\mathbf{d}$. By applying the expansion to each pixel in the block, an over-determined system of equations is obtained and the motion vector $\mathbf{d}$ can be solved directly using a pseudo-inverse approach. The main advantage of gradient-based methods over BM is that they are much less

[2]The subscript $_{n,n-1}$ has been dropped for sake of brevity of notation.

computationally intensive and can, in theory, estimate vectors to an infinitely fine precision. However, the Taylor Series expansion is only valid over very small distances and consequently, only small motions may be estimated with this method. In order to overcome this limitation, a series of recursive methods have been proposed [4, 54, 76] which iteratively solve for $\mathbf{d}$, by expanding the Taylor Series about the current estimate of the motion.

Phase Correlation

Phase Correlation techniques [46, 66] exploit the properties of the Discrete Fourier Transform (DFT) to solve for the motion of each block. Since a block in the current frame is assumed to be a shifted version of a block in the reference frame ($I_n(\mathbf{x}) = I_{n-1}(\mathbf{x}+\mathbf{d})$), the 2D DFT of the block, $F_n(\mathbf{w})$, is given by

$$F_k(\mathbf{w}) = e^{j2\pi(\mathbf{d}.\mathbf{w})} F_{n-1}(\mathbf{w}) \tag{2.6}$$

Like the Taylor Series expansion in the gradient-based method, Eq. 2.6 allows direct access to the motion vector $\mathbf{d}$. By manipulating this equation, it can be shown that the inverse DFT of the normalised cross-power spectrum of I_n and I_{n-1} yields a two-dimensional delta function with the impulse offset by the vector $\mathbf{d}$. Consequently, the motion can be found by finding the position of the impulse. The main advantage of the phase correlation technique is its relatively small computational complexity. However, unless the phase correlation function is interpolated [101], the vectors cannot be estimated to sub-pixel precision.

2.1.1 Motion Estimation Failure and Temporal Discontinuities

Motion Estimation Failure occurs when the estimated motion vector does not match the true motion of the pixels. In the introduction to this chapter, these failures were broken into 3 categories; failures due to degradation, ill-conditioned image data and PM. Other contributory factors include the wrong choice of block size and the complexity of the motion estimator.

Obviously, in most cases, prior knowledge of the true motion does not exist. This calls for another method of quantifying failures. A convenient measure of failure is the size of DFD defined in equation 2.3. It has already been stated that, according to the motion model (Eq. 2.2), small DFD values correspond to a good motion match and that a large DFD value corresponds to bad motion match. In effect, the sequence intensity function is considered to be temporally smooth along motion trajectories and this smoothness is violated when the motion estimator fails. Therefore, failures are associated with temporal discontinuities in the intensity function.

Temporally impulsive degradation artefacts (*i.e.* blotches) cause temporal discontinuities (Fig. 2.5), as the intensity of the artefact is significantly different from the true intensity, and this is a characteristic exploited by most missing data detectors (See Section 2.3). On the other hand, errors due to ill-conditioned image data do not typically result in temporal discontinuities (Fig. 2.6). The problem is that there are many candidate vectors that result in low DFD values. The algorithm of Kokaram [52, 54] proposes a modified gradient-based motion estimator which

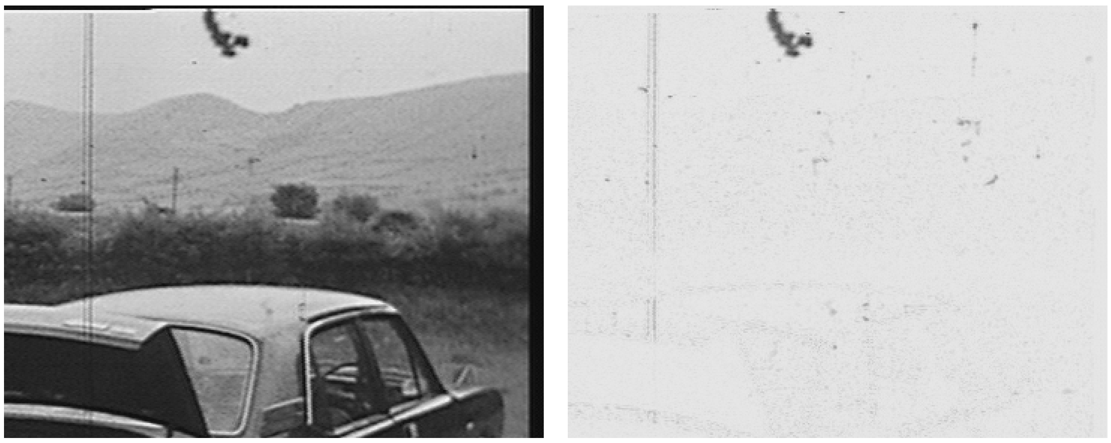


Figure 2.5: *This figure demonstrates that blotches cause temporal discontinuities. The right image shows the DFD between the frame depicted in left image and the previous frame in the sequence. The dark region in the DFD indicates high values caused by the blotch. High DFD values correspond to temporal discontinuities in the image sequence.*

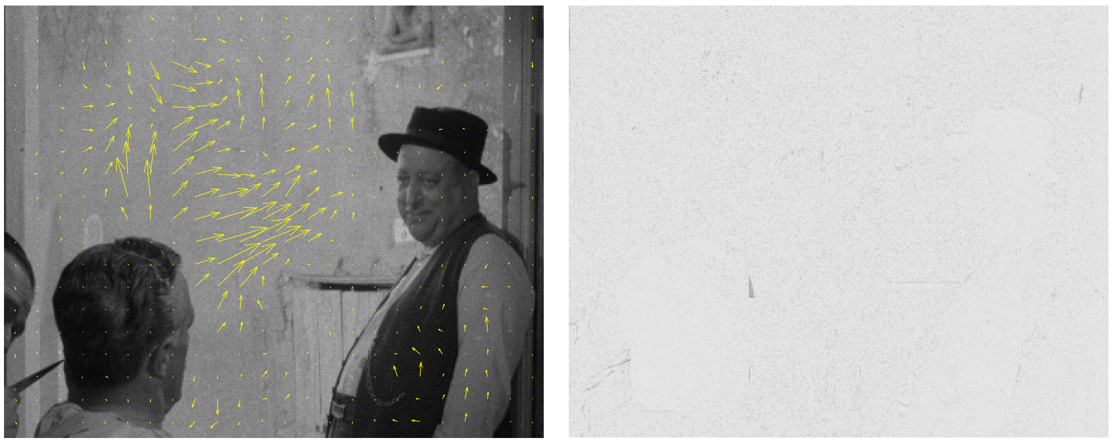


Figure 2.6: *This figure shows shows that motion estimation failures due to ill-conditioning do not normally result in temporal discontinuities. The arrows in the left image indicate the motion of each image block. The image blocks in the flat region in the middle of the image are ill-conditioned and the vector field in this region contains large motion vectors eventhough this region is not moving. In the DFD for the left frame (right image), there is no temporal discontinuity in this region. Although the vectors are clearly wrong, they still match with a similar block in the reference frame.*

---

incorporates a model for ill-conditioned data. It expands on earlier work by Martinez [70] and Driessen *et al.* [31] and constrains the motion estimate to be in the direction of maximum spatial contrast if the data is ill-conditioned.

The third category of motion estimation failure is Pathological Motion and this motion

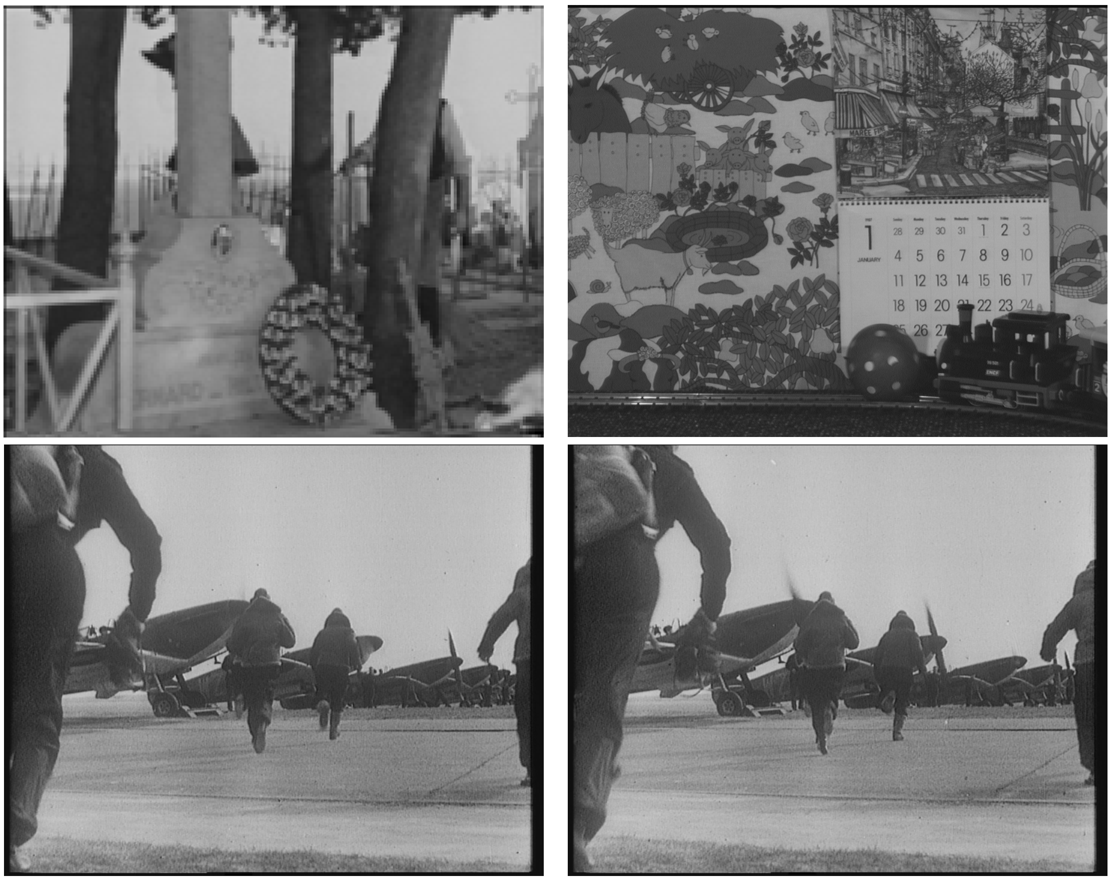


Figure 2.7: *This figure shows and example of occlusion and uncovering. The top row shows 2 examples of isolated occlusion. On the left the actors are walking behind foreground objects and on the right the train is occluding the calender. The bottom row shows an example of intermittent motion. In these two consecutive frames the aircrafts' propellors are appearing and disappearing. This pattern is repeated over the sequence.*

phenomenon is described in the following section. Like impulsive artefacts, PM can cause temporal discontinuities and can consequently be confused with missing data (Section 2.3). This is followed in section 2.4 by a review of MDD algorithms which include models for Pathological Motion.

2.2 Pathological Motion

State of the Art motion estimators contain models for a wide variety of motions including translation, affine motion and perspective distortions. However, there are motion patterns which fall outside these models and so can not be estimated properly. These motion patterns are considered to be pathological.

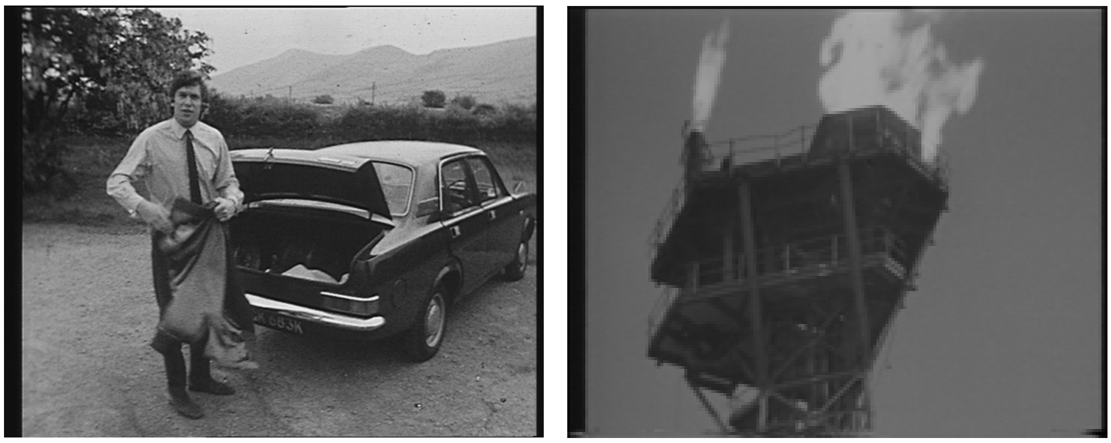


Figure 2.8: *The 2 images are examples of non-rigid motion. In the left image the shape of the jacket changes in an erratic manner. It is an-example of self-occluding motion as the sleeves are momentarily hidden behind the body of the jacket. The right shows an example of flames which also move in an erratic manner.*

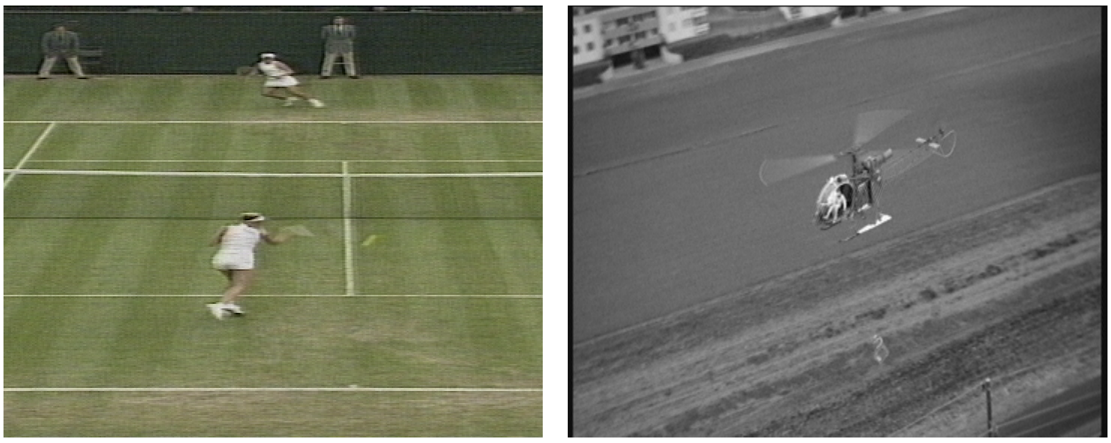


Figure 2.9: *This figure shows two images with fast motion which results in motion blur. Blurred objects appear to be elongated along their motion trajectories. In the left image the forearm, racket and ball are all blurred, as are the rotor blades in the right image.*

For the purposes of this thesis, *Pathological Motion is defined as the motion of objects which cause motion estimation failure.* This definition distinguishes PM from motion estimation failures caused by artefacts and by ill-conditioned data. PM is specific to the indicator of motion estimation failure used, which in this work is the presence of temporal intensity discontinuities. Furthermore, PM is motion estimator specific. Some motion patterns may cause failure in one estimator and not in another, although a given motion pattern is likely to be pathological for the majority of estimators.

Both Rares *et al.* [84,86] and Bornard [8] have presented taxonomies on common PM patterns.

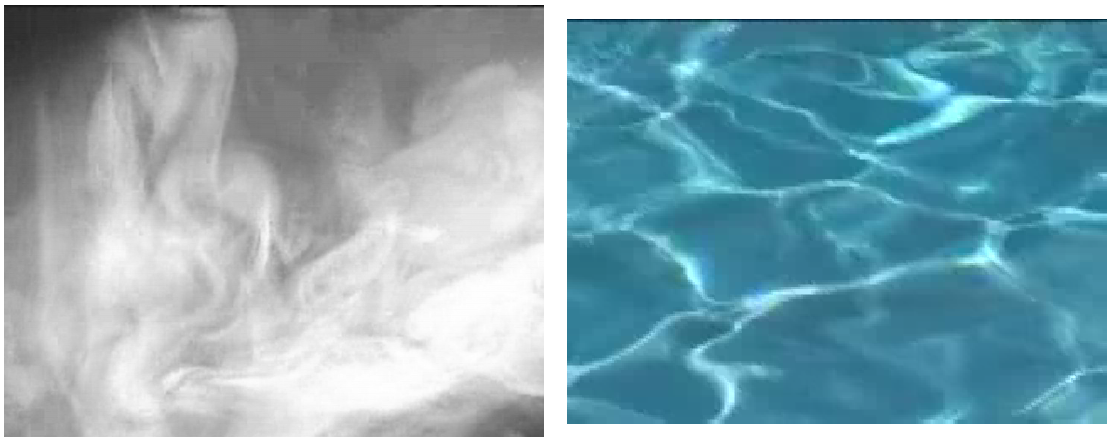


Figure 2.10: *Examples of motion superposition are shown in this figure. Left: A sequence with semi-transparent smoke. Right: Water with light reflecting on the surface.*

The following list of PM types further evaluates the taxonomies of Rares and Bornard and discusses patterns in terms of their underlying causes and how they cause motion estimation failures.

- *Occlusions and Uncovering* (Fig. 2.7) - This motion pattern can either isolated or repetitive. The isolated occlusion and uncovering typically occurs when the a foreground object is moving over the background. As the object moves, regions of the background near the boundary of the object are occluded and uncovered. Repetitive occlusion and uncovering occurs when an object is periodically appearing or disappearing. Examples include the flapping of a bird's wing where one side of the wing is visible in one frame and the other side is visible in the next frame. This repetitive occlusion can be considered a temporal aliasing of the moving object and is referred to as intermittent motion by both Rares and Bornard.

- *Non-Rigid Motion or Erratic Motion* (Fig. 2.8) - This category deals with the motion of non-rigid bodies whose shape is changing rapidly with time. This pattern typically causes motion estimation failure because the resolution of the estimated motion field is coarser than the true resolution of the motion field. This motion pattern can also be self-occluding, as part of the object may be occluded by the object itself.

- *Fast Motion* (Fig. 2.9) - Most motion estimators are restricted in the size of motion they can estimate. BM algorithms have an explicit limit on the size of vector, while gradient-based methods lose accuracy with large motions. If the size of the motion exceeds that limit, the estimator will fail. Fast motion also may also result in motion blur. Motion blur occurs when there is significant motion over the the exposure time of the camera which causes the moving object to be smeared over the background. As a result, the visible intensity profile and shape of the moving object is changed.

- *Motion Superposition* (Fig. 2.10) - Motion Superposition occurs when two or more motions are associated with a particular site. This is typically associated with the motion of transparent and semi-transparent objects but may also be caused by reflections on the surface of an object.

2.3 A Brief Review of Missing Data Detection Algorithms

The most effective Missing Data Detection algorithms exploit the temporal characteristics of blotches. Blotches are detected by searching for temporal discontinuities in the sequence. If a blotch is present at a location in a frame, then temporal discontinuities exist at that location between the current frame and the previous frame and also between the current frame and the next frame. Therefore, in order to detect blotches, it is necessary to compare the intensities of a three frame window centred on the frame under test. This section outlines some of the more well-known MDD algorithms. It considers both deterministic and probabilistic approaches.

2.3.1 Deterministic Blotch Detection

The simplest method of detecting blotches is to look for sites which have temporal discontinuities in both the forward and backwards motion vectors. In section 2.1.1 the link between high DFD values and temporal discontinuities was introduced. By defining the backward (Δ_b) and forward (Δ_f) DFDs as follows

$$\begin{aligned}\Delta_b(\mathbf{x}) &= I_n(\mathbf{x}) - I_{n-1}(\mathbf{x} + \mathbf{d}_{n,n-1}(\mathbf{x}))\\ \Delta_f(\mathbf{x}) &= I_n(\mathbf{x}) - I_{n+1}(\mathbf{x} + \mathbf{d}_{n,n+1}(\mathbf{x})),\end{aligned} \tag{2.7}$$

temporal discontinuities can be detected by applying a threshold to the DFDs. This results in a detector known as the SDIa (Spike Detection Index - a), outlined by Kokaram *et al.* in [61], which is defined by the following equation

$$b_{\text{SDIa}}(\mathbf{x}) = \begin{cases} 1 & \text{If } |\Delta_b(\mathbf{x})| > \delta_t \text{ AND } |\Delta_f(\mathbf{x})| > \delta_t \\ 0 & \text{otherwise} \end{cases} \tag{2.8}$$

where δ_t is the value of the threshold. A blotch is detected at pixel site $\mathbf{x}$ if $b_{\text{SDIa}}(\mathbf{x}) = 1$.

Although the SDIa algorithm detects temporal discontinuities, it makes no distinction about the sign of the DFD value. Given that the intensity of blotches is either much higher or lower than the image data they replace, and are by extension much higher or lower than the local intensities in the surrounding frames, this implies that the temporal intensity profile of a blotch is an impulse. Therefore, not only should the size of the DFDs be large, they should also either be both positive or negative (*i.e.* have the same sign). This observation was first made

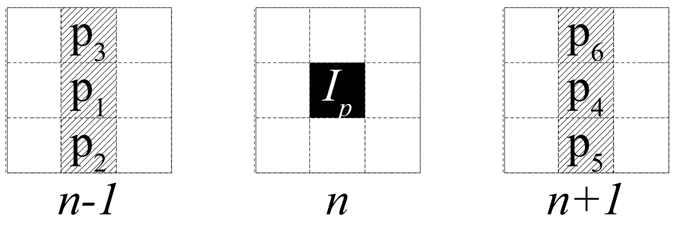


Figure 2.11: *This diagram illustrates the pixels used in ROD detector.*

by Storey [98] who incorporated it in a detector that did not use motion compensation. The motion compensated extension of this algorithm, called the SDIp [55], is given by

$$b_{\text{SDIp}}(\mathbf{x}) = \begin{cases} 1 & \text{If } |\Delta_b(\mathbf{x})| > \delta_t \text{ AND } |\Delta_f(\mathbf{x})| > \delta_t \text{ AND } (\text{sign}(\Delta_f) = \text{sign}(\Delta_b)) \\ 0 & \text{otherwise} \end{cases} \tag{2.9}$$

Obviously, the condition for a blotch in SDIp is stricter than in SDIa. It employs a more complete blotch model and in theory should have a lower false alarm rate than SDIa.

A similar detector to the SDIp is outlined in [92], in which Schallauer *et al.* enforce the impulse condition in a different manner. Instead of constraining the sign of the DFDs to be the same, the intensities of the motion compensated backward and forward frames are compared. Even if a blotch is present in the current frame, the intensities in the backward forward frames should be similar. The detector in [92] proceeds on this basis by applying a second smaller threshold to the difference between the forward and backward intensities. The detector now becomes

$$b_{\text{sch}}(\mathbf{x}) = \begin{cases} 1 & \text{If } |\Delta_b(\mathbf{x})| > \delta_t \text{ AND } |\Delta_f(\mathbf{x})| > \delta_t \\ & \quad \text{AND } |I_{n+1}(\mathbf{x} + \mathbf{d}_{n,n+1}(\mathbf{x})) - I_{n-1}(\mathbf{x} + \mathbf{d}_{n,n-1}(\mathbf{x}))| < \delta_l \\ 0 & \text{otherwise} \end{cases} \tag{2.10}$$

where δ_l is a user-defined threshold such that $\delta_l < \delta_t$. In practice, this constraint is stricter than the sign constraint in the SDIp technique. Depending on the size of δ_l, the detection rate for this algorithm will be lower than for the SDIp.

The ROD Detectors

Nadenau and Mitra [75] introduced another blotch detector in image sequences known as the Rank Order Difference (ROD) Detector. It incorporates some spatial information into the detection by considering a neighbourhood of pixels in the forward and backward motion compensated frames.

Considering the following list of pixels, reperesented visually in Fig. 2.11,

$$\begin{aligned}
p_1 &= I_{n-1}(\mathbf{x} + \mathbf{d}_{n,n-1}(\mathbf{x})) \\
p_2 &= I_{n-1}(\mathbf{x} + \mathbf{d}_{n,n-1}(\mathbf{x}) + [0\ 1]) \\
p_3 &= I_{n-1}(\mathbf{x} + \mathbf{d}_{n,n-1}(\mathbf{x}) + [0\ \ -1]) \\
p_4 &= I_{n+1}(\mathbf{x} + \mathbf{d}_{n,n+1}(\mathbf{x})) \\
p_5 &= I_{n+1}(\mathbf{x} + \mathbf{d}_{n,n+1}(\mathbf{x}) + [0\ 1]) \\
p_6 &= I_{n+1}(\mathbf{x} + \mathbf{d}_{n,n+1}(\mathbf{x}) + [0\ \ -1]) \\
I_p &= I_n(\mathbf{x}),
\end{aligned} \tag{2.11}$$

the ROD detector proceeds by arranging p_1 to p_6 in ascending order forming a list $[r_1,\ r_2,\ r_3,\ r_4,\ r_5,\ r_6]$. The median of the pixels is then calculated as $M = (r_3 + r_4)/2$. From this three motion compensated difference values are calculated according to

$$\begin{aligned}
&\text{If } I_p > M \\
&\qquad e_1 = I_p - r_6 \\
&\qquad e_2 = I_p - r_5 \\
&\qquad e_3 = I_p - r_4 \\
&\text{If } I_p \leq M \\
&\qquad e_1 = r_1 - I_p \\
&\qquad e_2 = r_2 - I_p \\
&\qquad e_3 = r_3 - I_p
\end{aligned} \tag{2.12}$$

Blotches are detected by examining the values of the motion compensated differences. Three thresholds are selected and detection proceeds according to

$$b_{\text{ROD}}(\mathbf{x}) = \begin{cases} 1 & \text{If } (e_1 > \delta_1) \text{ OR } (e_2 > \delta_2) \text{ OR } (e_1 > \delta_3) \\ 0 & \text{otherwise} \end{cases} \tag{2.13}$$

where δ_1, δ_2 and δ_3 are the thresholds such that $\delta_1 \leq \delta_2 \leq \delta_3$. The choice of δ_1 is the most critical as the $e_1 > t_1$ condition is typically the first to be satisfied. In [5,104] a simplified ROD (sROD) detector was outlined. By setting both δ_2 and δ_3 to infinity, the detector becomes

$$b_{\text{sROD}}(\mathbf{x}) = \begin{cases} 1 & \text{If } (e_1 > \delta_1) \\ 0 & \text{otherwise} \end{cases} \tag{2.14}$$

The inclusion of the extra spatial information in the ROD detectors gives them improved robustness over the SDI detectors, especially to image noise. If there is a large variance in the values p_1 to p_6, then the range of intensities for which the pixel under test can be flagged as a blotch is restricted to the top or bottom of the intensity range (Fig. 2.12).

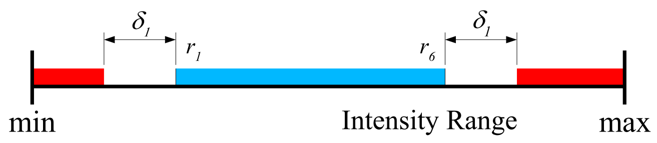


Figure 2.12: *The red bands indicate the range of intensities for which a blotch would be detected. As the difference between r_1 and r_6 increases, the range of blotch intensities reduces, making detection of blotches less likely.*

The 3D AR Detector

In [61], Kokaram *et al.* outlines another blotch detector that, like the ROD detectors, includes spatial information from the motion compensated neighbouring frames in the decision process. It is also used in the integrated MDT frameworks outlined in [53,60]. A 3D AR prediction model predicts the intensity of the current pixel, $I_n(\mathbf{x})$, which is modelled as a linear combination of a neighbourhood of pixels in the neighbouring frames. The model for $I_n(\mathbf{x})$ is given by

$$\begin{aligned} I_n(\mathbf{x}) &= \sum_{k=1}^{N} a_k I_{n-1}(\mathbf{x} + \mathbf{d}_{n,n-1}(\mathbf{x}) + \mathbf{q}_k) + e_b(\mathbf{x}) \\ &= \sum_{k=1}^{N} b_k I_{n+1}(\mathbf{x} + \mathbf{d}_{n,n+1}(\mathbf{x}) + \mathbf{p}_k) + e_f(\mathbf{x}) \end{aligned} \tag{2.15}$$

where a_k and b_k are the model coefficients, where $\mathbf{q}_k$ and $\mathbf{p}_k$ are spatial offsets and where e_b and e_f are the model errors for the backward and forward frames. The model coefficients are estimated by finding the set of coefficients which minimise the model error for the given set of spatial offsets (Fig. 2.13). The size of this model error is used as an indicator for blotches. If temporal discontinuities exist then the intensity of the current pixel does not behave in accordance with this model and consequently the model error will be large. Hence, a blotch detector can be defined as follows

$$b_{3\text{DAR}}(\mathbf{x}) = \begin{cases} 1 & \text{If } |e_b(\mathbf{x})| > \delta_t \text{ AND } |e_f(\mathbf{x})| > \delta_t \\ 0 & \text{otherwise} \end{cases} \tag{2.16}$$

When estimating the model coefficients, it is assumed that they are constant over a spatial window. The choice of window size is important. If a blotch exits which is significant with respect to the size of the window, then the blotch will bias the coefficient values. The coefficients model the blotch rather than the true image data and therefore the model error will be low. Therefore this detector is only suitable for detecting small blotches.

2.3.2 Probabilistic Approaches

All the MDD algorithms discussed so far have used hard thresholds to detect temporal discontinuities. No method has attempted to include any prior knowledge of temporal discontinuities

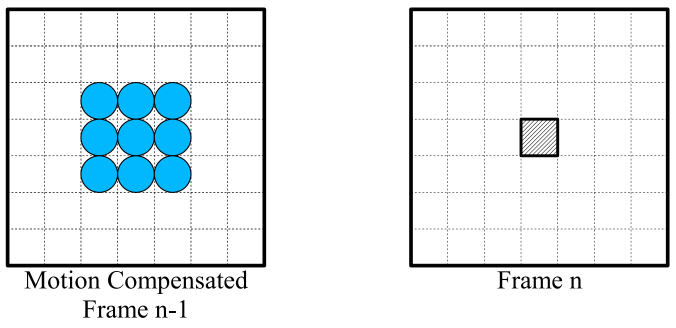


Figure 2.13: *The shaded pixel in frame n is assumed to be a linear combination of the blue pixels in the previous frame. The spatial offsets in this case are* $[-1\ -1]$, $[0\ -1]$, $[1\ -1]$, $[-1\ 0]$, $[0\ 0]$, $[1\ 0]$, $[-1\ 1]$, $[0\ 1]$ *and* $[1\ 1]$.

in the decision process. Temporal discontinuities tend to form as a regions and are not spatially impulsive events. In other words, Temporal Discontinuity Fields (TDF) are spatially smooth.

This information can be introduced into the decision process using a probabilistic framework. In probabilistic frameworks, the field to be estimated is modelled as a Markov Random Field (MRF). This implies that the value of the field at any site somehow depends on the values of the field in its neighbourhood, allowing the spatial properties of temporal discontinuity fields to be modelled.

Framework - Taking the detection of the backward temporal discontinuity field for the current frame $t_b(\mathbf{x})$ as an example, the problem can be expressed in probabilistic terms as finding the field that maximises the PDF given by

$$P(t_b(\mathbf{x})|\Delta_b(\mathbf{x}), T_b) \tag{2.17}$$

where T_b is the value of the neighbourhood of field values about site $\mathbf{x}$. Using Bayes Law this posterior distribution can be factorised as follows

$$P(t_b(\mathbf{x})|\Delta_b(\mathbf{x}), T_b) \propto P_l(\Delta_b(\mathbf{x})|t_b(\mathbf{x})) \times P_s(t_b(\mathbf{x})|T_b) \tag{2.18}$$

where $P_l(.)$ is known as the data likelihood and $P_s(.)$ is known as a prior.

Data Likelihood - The likelihood expresses the expected distribution of the observed data (*i.e.* Δ_b) in both the presence and absence of temporal discontinuity. When temporal discontinuities do not exist, the DFD values are expected to be small. A suitable expression of the likelihood is given by

$$P_l(\Delta_b(\mathbf{x})|s_b(\mathbf{x})) \propto \exp \begin{cases} -\frac{\Delta_b(\mathbf{x})^2}{2\sigma_e^2} & t_b(\mathbf{x}) = 0 \\ -\frac{\alpha^2}{2} & t_b(\mathbf{x}) = 1 \end{cases} \tag{2.19}$$

When a temporal discontinuity does not exist the distribution of the DFD assumed to be gaussian with a variance σ_e^2 given by the variance of the motion model error (*i.e.* Equation 2.2 is valid). The α value is effectively a soft threshold on the size of the DFD. If $\Delta_b(\mathbf{x})^2 > \alpha^2\sigma_e^2$, the Maximum Likelihood (ML) solution for $t_b(\mathbf{x})$ is $t_b(\mathbf{x}) = 1$.

Prior - The second term in the expansion, $P_s(t_b(\mathbf{x})|T_b)$, is a probability prior on t_b which enforces the spatial smoothness constraint. An MRF model is used for t_b and, from the Hammersley-Clifford Theorem [2], it follows that the prior can be described using a Gibbs Distribution. A suitable expression that articulates the spatial smoothness constraint is given by

$$p(t_b(\mathbf{x})|T_b) \propto \exp -\left\{\Lambda \sum_{\mathbf{y}\in\mathcal{N}_s(\mathbf{x})} \lambda_{\mathbf{y}} |t_b(\mathbf{x}) \neq t_b(\mathbf{y})|\right\} \tag{2.20}$$

where $\mathcal{N}_s(\mathbf{x})$ is the spatial neighbourhood[3] of $\mathbf{x}$, where Λ is the scaling factor between the prior and the likelihood terms and where $\lambda_{\mathbf{y}}$ is a weight inversely proportional to $\|\mathbf{x} - \mathbf{y}\|$.

Solving for $t_b(\mathbf{x})$ - The optimum solution for $t_b(\mathbf{x})$ is the Maximum a Posteriori (MAP) configuration according to equation 2.18. Given that every pixel can have two values, there are 2^n possible configurations where n is the number of pixels in the image. Consequently, it is computationally intractable to estimate the posterior probability for every configuration of the field. There are a number of different optimisation strategies for finding a solution. These include *Gibbs Sampling* coupled with *Simulated Annealing* [37, 51], *Iterated Conditional Modes* (ICM) [3] and *Graph Cuts* [11, 39]. The Gibbs Sampling technique converges to the globally optimum solution irrespective of the the initial configuration of the field but typically slow. ICM is a less computationally intensive method. However, it only finds the a local maximum of the posterior expression and so the result is heavily dependent on the initial configuration of the result. The Graph Cuts technique is guaranteed to find the optimum technique and is also less computationally intensive, however it only finds the optimum solution in the two state case.

The Morris Algorithm

In [73], Morris proposes a blotch detection algorithm which uses a probabilistic framework, similar to that described above, to evaluate both forward and backward temporal discontinuity fields. The blotch detector is defined by

$$b_{\text{Morris}}(\mathbf{x}) = \begin{cases} 1 & \text{If } t_b = 1 \text{ AND } t_f = 1 \\ 0 & \text{otherwise} \end{cases} \tag{2.21}$$

where t_b and t_f are the binary backward and forward temporal discontinuities respectively. This detector is similar to the SDIa technique. Instead of using a threshold, δ_t, on the DFDs to detect temporal discontinuities, a probabilistic framework is used.

[3]The four or 8 pixel first-order neighbourhoods are commonly used

$s(\mathbf{x})$	$t_b(\mathbf{x})$	$t_f(\mathbf{x})$	$b(\mathbf{x})$
0	0	0	0
1	0	1	0
2	1	0	0
3	1	1	1

Table 2.1: *The Blotch Model.*

In [15], Chong *et al.* introduced a modified Morris detector. The authors make the observation that a major reason for false alarms in MDD are motion estimation failures associated with moving edges. A moving edge detector is introduced and is included in the temporal discontinuity detection frame work. The likelihood expression now becomes

$$P_l(\Delta_b(\mathbf{x})|t_b(\mathbf{x})) \propto \exp \begin{cases} -\frac{\Delta_b(\mathbf{x})^2}{2\sigma_e^2} & t_b(\mathbf{x}) = 0 \\ -(\frac{\alpha^2}{2} + \phi(\mathbf{x})) & t_b(\mathbf{x}) = 1 \end{cases} \tag{2.22}$$

and the prior expression becomes

$$p(t_b(\mathbf{x})|T_b) \propto \exp -\left\{ \Lambda \sum_{\mathbf{y} \in \mathcal{N}_s(\mathbf{x})} (\lambda_{\mathbf{y}} + \phi(\mathbf{x}))|t_b(\mathbf{x}) \neq t_b(\mathbf{y})| \right\} \tag{2.23}$$

where $\phi(\mathbf{x})$ is an energy penalty associated with the output of the moving edge detector. $\phi(\mathbf{x})$ contains non-zero values only at moving edge sites and its size is proportional to the gradient magnitude at the edge. Effectively temporal discontinuity detection has been made more conservative. In this detector, the Maximum Likelihood (ML) solution for $t_b(\mathbf{x})$ is $t_b(\mathbf{x}) = 1$ when $\Delta_b^2 > \sigma_e^2(\alpha^2 + 2\phi(\mathbf{x}))$.

2.3.3 Direct Modelling of Blotches

In the probabilistic approach outlined above, the blotch detection occurred in two stages. The first stage estimated forward and backward temporal discontinuity fields using a probabilistic framework. The second setects blotches by interpreting the discontinuity information according to Eq. 2.21. A logical extension to this approach would be to merge the two stages into a single probabilistic framework. Detecting blotches in this manner, would allow a spatial smoothness prior to be introduced on the blotch field itself. The problem is one of finding the MAP estimate of the binary blotch field, $b(\mathbf{x})$, given both forward and backward DFD fields (*i.e.* $P(b(\mathbf{x})|\Delta_b, \Delta_f)$).

Table 2.1 gives an example of an appropriate blotch model. For each site $\mathbf{x}$ there are two binary temporal discontinuity values $t_b(\mathbf{x})$ and $t_f(\mathbf{x})$ which represent the values of the backward and forward temporal discontinuity fields respectively. Consequently, there are four possible configurations of these values for each $\mathbf{x}$. To represent this, a new state variable $s(\mathbf{x})$ is defined for the site, which can have four values corresponding to each configuration of t_b and t_f. If and only if both t_b and t_f are high, then a blotch is considered to exist (*i.e.* $b(\mathbf{x}) = 1$). This corresponds to $s(\mathbf{x}) = 3$. According to table 2.1, $s(\mathbf{x}) = 1$ and $s(\mathbf{x}) = 2$ correspond to the case where only one temporal discontinuity exists. Including these two states in the framework prevents false alarms when only one DFD is large.

Framework - There are now two variables to estimate, $s(\mathbf{x})$ and $b(\mathbf{x})$, and the relationship between these variables is governed by table 2.1. The problem can now be expressed in bayesian terms by

$$P(s(\mathbf{x})|\Delta_b, \Delta_f) \propto P_l(\Delta_b(\mathbf{x}), \Delta_f(\mathbf{x})|s(\mathbf{x})) \times P_r(s(\mathbf{x})) \tag{2.24}$$

This expression consists of one likelihood and a prior on the state field $s(\mathbf{x})$.

Likelihood - The likelihood expression is given by

$$P_l(\Delta_b(\mathbf{x}), \Delta_f(\mathbf{x})|s(\mathbf{x})) \propto \exp \begin{cases} -\frac{\Delta_b(\mathbf{x})^2}{2\sigma_e^2} - \frac{\Delta_f(\mathbf{x})^2}{2\sigma_e^2} & \text{if } s(\mathbf{x}) = 0 \\ -\frac{\Delta_b(\mathbf{x})^2}{2\sigma_e^2} - \frac{\alpha^2}{2} & \text{if } s(\mathbf{x}) = 1 \\ -\frac{\alpha^2}{2} - \frac{\Delta_f(\mathbf{x})^2}{2\sigma_e^2} & \text{if } s(\mathbf{x}) = 2 \\ -\frac{\alpha^2}{2} - \frac{\alpha^2}{2} & \text{if } s(\mathbf{x}) = 3. \end{cases} \tag{2.25}$$

Using this expression prevents falsely detected blotches when only one DFD is large. If, for example, $\Delta_b(\mathbf{x})$ is large and $\Delta_f(\mathbf{x})$ is small, then the ML state will be $s(\mathbf{x}) = 1$ and hence $b(\mathbf{x}) = 0$.

Prior - The second term in the expansion is the prior on $s(\mathbf{x})$. Like the probabilistic temporal discontinuity detector, the result should be spatially smooth. A suitable expression for $P_r(s(\mathbf{x}))$ is

$$P_r(s(\mathbf{x})) = P_s(s(\mathbf{x})|S) \times P_s(b(\mathbf{x})|B) \tag{2.26}$$

where $P_s(s(\mathbf{x})|S)$ and $P_s(b(\mathbf{x})|B)$ are spatial smoothness priors on the state field $s(\mathbf{x})$ and blotch field $b(\mathbf{x})$. The same spatial smoothness priors are used for $P_s(s(\mathbf{x})|S)$ and $P_s(b(\mathbf{x})|B)$, and are given respectively by

$$\begin{aligned} p(s(\mathbf{x})|S) &\propto \exp -\left\{\Lambda_s \sum_{\mathbf{y}\in\mathcal{N}_s(\mathbf{x})} \lambda_{\mathbf{y}} |s(\mathbf{x}) \neq s(\mathbf{y})|\right\} \\ p(b(\mathbf{x})|T_b) &\propto \exp -\left\{\Lambda_b \sum_{\mathbf{y}\in\mathcal{N}_s(\mathbf{x})} \lambda_{\mathbf{y}} |b(\mathbf{x}) \neq b(\mathbf{y})|\right\} \end{aligned} \tag{2.27}$$

Solving for s and b - As the relationship between $s(\mathbf{x})$ and $b(\mathbf{x})$ is fixed according to table 2.1, the value of $b(\mathbf{x})$ can be derived directly from the value of $s(\mathbf{x})$. For each site $\mathbf{x}$ there are four possible values of $s(\mathbf{x})$ (or states). The posterior probability for the four possible states of $\mathbf{x}$ is evaluated according to

$$\begin{aligned}
P(s=0|\Delta_b,\Delta_f) &\propto P_l(\Delta_b,\Delta_f|s=0)\times P_s(s=0|S)\times P_s(b=0|B)\\
P(s=1|\Delta_b,\Delta_f) &\propto P_l(\Delta_b,\Delta_f|s=1)\times P_s(s=1|S)\times P_s(b=0|B)\\
P(s=2|\Delta_b,\Delta_f) &\propto P_l(\Delta_b,\Delta_f|s=2)\times P_s(s=2|S)\times P_s(b=0|B)\\
P(s=3|\Delta_b,\Delta_f) &\propto P_l(\Delta_b,\Delta_f|s=3)\times P_s(s=3|S)\times P_s(b=1|B) \qquad (2.28)
\end{aligned}$$

where the site index $\mathbf{x}$ has been dropped for sake of brevity.

This model for blotch detection is incorporated into the JONDI missing data treatment framework [57]. This framework has been used in a variety of other applications including tear delineation [21], video matting [58] and foreground/background segmentation [50].

2.4 The Pathological Motion Detection State of the Art

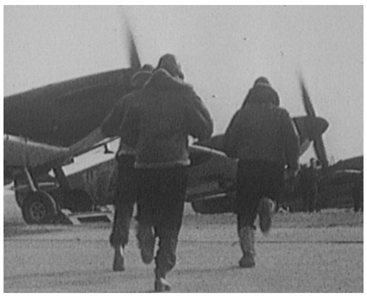
The missing data detectors outlined in the previous section suffer from false alarms in the presence of Pathological Motion. This occurs when PM causes forward and backward temporal discontinuities. Although these techniques are robust to isolated occlusions (when only one discontinuity exists), PM regions which persist over a longer temporal window will result in discontinuities in a number of consecutive frames. In effect, the temporal intensity profile is quasi-periodic as opposed to the impulse profile of the blotches. Since the window over which the MDD algorithms operate is restricted to three frames, PM can not be distinguished from missing data (See Fig. 2.14).

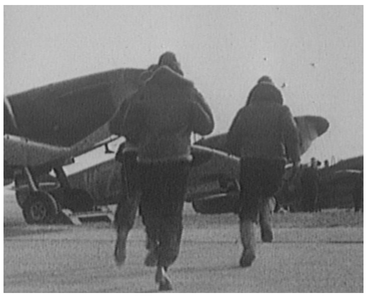
This section introduces the Missing Data Detection techniques in the literature which have attempted to model for the presence of Pathological Motion in image sequences. The algorithms outlined fall into two broad categories. The first category attempts to detect regions of motion estimation failure and then classifies them into regions of failure due to artefacts such as blotches and due to PM [8, 84]. The second category attempts to find possible PM regions in an image and to make MDD more conservative in these areas [50].

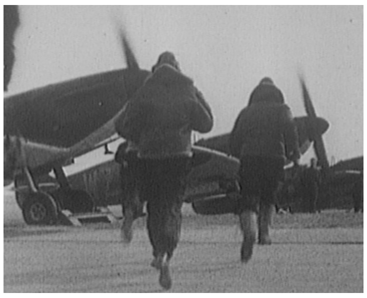
2.4.1 Long-Term Pathological Motion Detection [8]

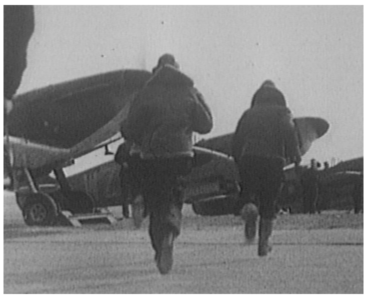
One method of making Missing Data Detection robust is to extend the temporal window over which blotches are detected, allowing long-term PM to be distinguished from missing data. This approach was first suggested by van Roosmalen [104], who extended the sROD detector (sRODex) to include three extra reference pixels from both frames $n-2$ and $n+2$, resulting in lower false alarm rate than the standard sROD detector(Fig. 2.15).

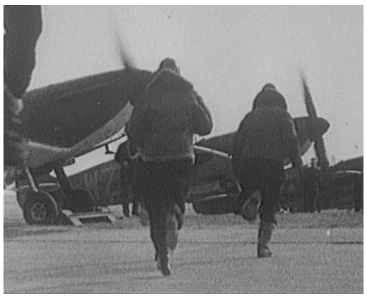
In his thesis [8], Bornard outlines a probabilistic MDD technique, inspired by the Morris detector, that also operates over a larger temporal window. The idea is to make MDD more conservative by preventing any suspected PM sites being detected as a blotch. It extends the


(a) Frame $n - 2$


(b) Frame $n - 1$


(c) Frame n


(d) Frame $n + 1$


(e) Frame $n + 2$

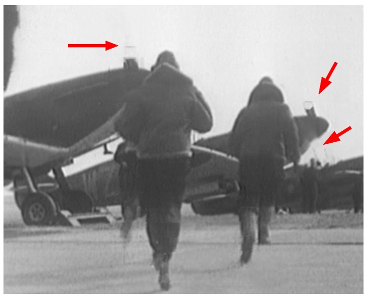
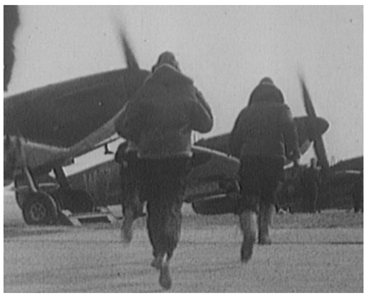
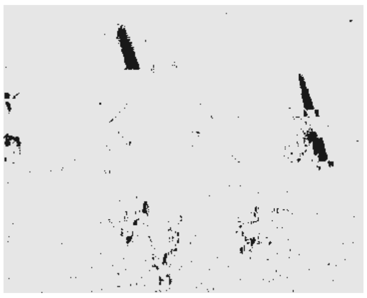

(f) Frame n after MDT

Figure 2.14: *This figure shows 5 consecutive frames (a-e) from the "propellors" sequence. The motion of the propellors is intermittent and the blades appear and disappear repeatedly. If the MDD window is three frames about frame n, the blades will be falsely detected as a blotch and will be removed after interpolation (f).*


(a) Frame n


(b) sROD mask

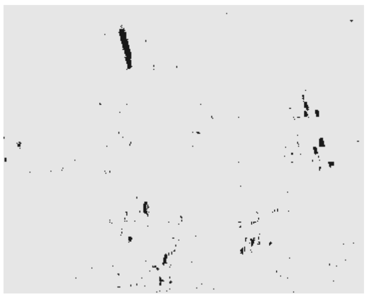

(c) sRODex mask

Figure 2.15: *This figure illustrates the lower false alarm rate of the sRODex detector. In the sROD detector, the blades are a major source of false alarms. In sRODex the majority of these sites are no longer detected as a blotch*

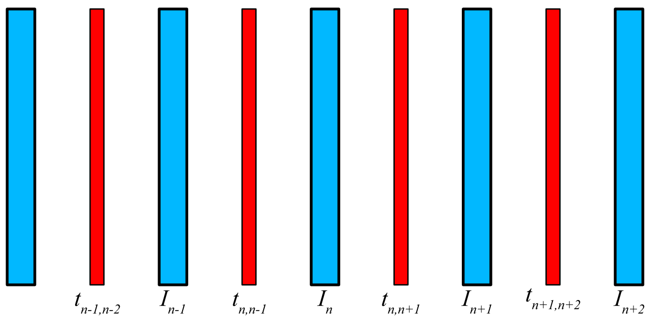


Figure 2.16: *This figure illustrates the arrangement of the five frames (blue) and four temporal discontinuity fields (red).*

blotch detection window from three to five frames centred on the frame to be tested. Between these frames, four temporal discontinuity fields are estimated for each pair of consecutive frames (Fig. 2.16). As blotches are characterised by an impulsive profile, there should be discontinuities in both of the two fields closest to the central frame ($t_{n,n-1}$ and $t_{n,n+1}$) and no discontinuities in either of outer fields ($t_{n-1,n-2}$ and $t_{n+1,n+2}$). If discontinuities also exist in the outer field, then the site is considered to be a Pathological Motion site and not a Missing Data site.

Like the Morris detector, Bornard outlines an algorithm which operates in two stages. In the first stage the four temporal discontinuity fields are estimated using a probabilistic framework similar to that in Section 2.3.2. In this framework, a further temporal prior is introduced that discriminates against temporal discontinuities occurring at the same sites in multiple frames. The second stage is a deterministic interpretation stage, which labels pixels as being blotches, PM sites or unaffected sites.

Temporal Discontinuity Detection

Bornard's method calculates temporal discontinuities over a five frame window. For each frame in the sequence the window is centred on the current frame and the remaining frames are motion compensated with respect to the current frame. Fig. 2.17 depicts the motion compensation scheme over the window. For a five frame window defined by the frame intensity functions, $[I_{n-2}(\mathbf{x}), I_{n-1}(\mathbf{x}), I_n(\mathbf{x}), I_{n+1}(\mathbf{x}), I_{n+2}]$, a set of 5 motion compensated frames are defined as

$$\begin{aligned}
I'_{n-2}\Big(\mathbf{x}\Big) &= I_{n-2}\Big(\mathbf{x} + \mathbf{d}_{n,n-1}(\mathbf{x}) + \mathbf{d}_{n-1,n-2}\big(\mathbf{x} + \mathbf{d}_{n,n-1}(\mathbf{x})\big)\Big)\\
I'_{n-1}\Big(\mathbf{x}\Big) &= I_{n-1}\Big(\mathbf{x} + \mathbf{d}_{n,n-1}(\mathbf{x})\Big)\\
I'_{n}\Big(\mathbf{x}\Big) &= I_{n}\Big(\mathbf{x}\Big)\\
I'_{n+1}\Big(\mathbf{x}\Big) &= I_{n+1}\Big(\mathbf{x} + \mathbf{d}_{n,n+1}(\mathbf{x})\Big)\\
I'_{n+2}\Big(\mathbf{x}\Big) &= I_{n+2}\Big(\mathbf{x} + \mathbf{d}_{n,n+1}(\mathbf{x}) + \mathbf{d}_{n+1,n+2}\big(\mathbf{x} + \mathbf{d}_{n,n+1}(\mathbf{x})\big)\Big) \qquad (2.29)
\end{aligned}$$

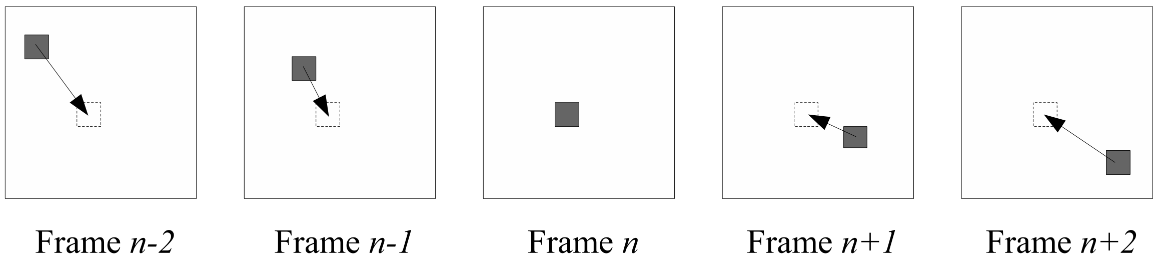


Figure 2.17: *This figure shows the motion compensation scheme over the 5 frame window. Each frame is motion compensated with respect to the central frame.*

Framework - A Temporal Discontinuity Field is calculated between each set of two consecutive frames resulting in four Temporal Discontinuity Fields per five frame window (Fig. 2.16). Taking the estimation of $t_{n,n+1}$ as an example, the posterior, $P(t_{n,n+1}|.)$, is factorised into a likelihood and priors using Bayes Law as follows:

$$P(t_{n,n+1}|I'_n, I'_{n+1}, T_{n,n+1}, t_{n,n-1}, t_{n+1,n+2}) \propto \\ P_l(I'_n, I'_{n+1}|t_{n,n+1}, T_{n,n+1}) \times P_s(t_{n,n+1}|T_{n,n+1}) \times P_t(t_{n,n+1}|t_{n,n-1}, t_{n+1,n+2}) \quad (2.30)$$

where $P_l(.)$ is the likelihood, $P_s(.)$ is the spatial smoothness prior and $P_t(.)$ is the temporal prior.

Likelihood - The expression for the likelihood is given as follows

$$P_l(I'_n, I'_{n+1}|t_{n,n+1}, T_{n,n+1}) = \exp\left[-\left(\left(I'_n(\mathbf{x}) - I'_{n+1}(\mathbf{x})\right)^2\left(1 - t_{n,n+1}(\mathbf{x})\right) + \beta_1 t_{n,n+1}(\mathbf{x})\right)\right] \quad (2.31)$$

Like the framework in Section 2.3.2, the ML solution for $t_{n,n+1}(\mathbf{x}) = 0$ occurs when the DFD, $I'_n(\mathbf{x}) - I'_{n+1}(\mathbf{x})$, is low. In this case, the condition for the ML solution of $t_{n,n+1} = 1$ is $\left(I'_n(\mathbf{x}) - I'_{n+1}(\mathbf{x})\right)^2 > \beta_1$.

Priors - There are two priors associated with the framework. The first is a spatial smoothness prior and is expressed as follows

$$p(t_{n,n+1}|T_{n,n+1}) = \exp\left[-\left(-\beta_2 \sum_{\mathbf{y} \in \mathcal{N}_S(\mathbf{x})} \left(t_{n,n+1}(\mathbf{x}) - \frac{1}{2}\right)\left(t_{n,n+1}(\mathbf{y}) - \frac{1}{2}\right)\left(1 - u_k(\mathbf{x}, \mathbf{y})\right)\right)\right] \quad (2.32)$$

where $\mathcal{N}_S(\mathbf{x})$ is an 8-pixel first-order spatial neighbourhood. The β_2 parameter controls the weighting of the spatial smoothness energy. $u_k(\mathbf{x}, \mathbf{y})$ is a binary field which is equal to 1 when an image edge (*i.e.* spatial discontinuity) lies between $\mathbf{x}$ and $\mathbf{y}$. This term results in the spatial smoothness prior being turned off across image edges. It is derived from the assumption that the boundaries of temporal discontinuities often coincide with edges.

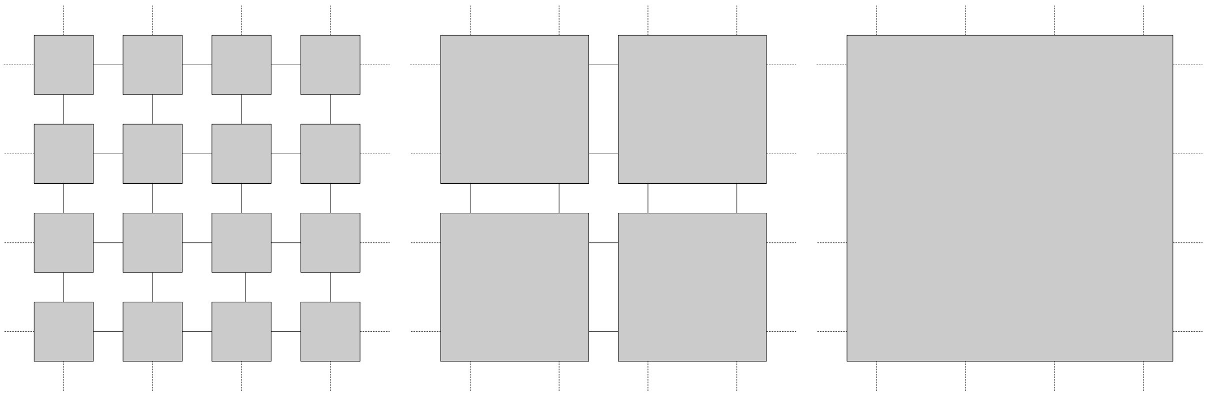


Figure 2.18: *This Figure shows the layout of a section of the fields on 3 consecutive levels. On the left is the finest level and contains a 4×4 block of field sites. The layout of the next coarsest level is shown in the middle. Each site corresponds to a 2×2 block of sites at the finer level. On the right is the coarsest level of the 3 and a site on this level corresponds to a 2×2 block on the middle level and a 4×4 block on the lowest level.*

The second prior is a temporal prior which penalises temporal discontinuities being detected at sites in consecutive fields. The prior is defined by the equation

$$P_t(t_{n,n+1}|t_{n,n-1}, t_{n+1,n+2}) \exp -\beta_3 t_{n,n+1}(\mathbf{x}) \left(t_{n-1,n}(\mathbf{x}) + t_{n+1,n+2}(\mathbf{x})\right) \tag{2.33}$$

where β_3 is a weighting parameter. If the neighbouring TDFs $t_{n-1,n}$ and $t_{n+1,n+2}$ are 0, then there is a 0 penalty. If both are equal to one, then the penalty is $2\beta_3$.

A multiresolution framework [41] is used to improve the efficiency of the discontinuity detection. A set of discontinuity fields of coarser resolution is organised into a hierarchical set of levels, with the full resolution field at the bottom and the coarsest resolution at the top level. Each field site at a level, i, consists of a 2×2 block of sites at the level below, $i - 1$. Fig. 2.18 illustrates how the sites are grouped into blocks. If the full resolution of the field is 720×576 the resolution of the next level will be 360×288 and so on. The MAP estimate for the discontinuity field is calculated at the coarser resolution and that result is then used to initialise the estimation at the level below, until the result is estimated at the full-resolution. Figure 2.19 shows an example four TDFs estimated from a five frame window.

Interpretation Step

After the discontinuity fields have been calculated an interpretation step is performed to classify the temporal discontinuities in terms of Pathological Motion and Missing Data. Firstly the discontinuity fields between the current and previous and current and next frames are examined. There are 3 possible cases for each site:

1. No temporal discontinuities exist in either field - No PM or Missing Data present. No action is necessary.

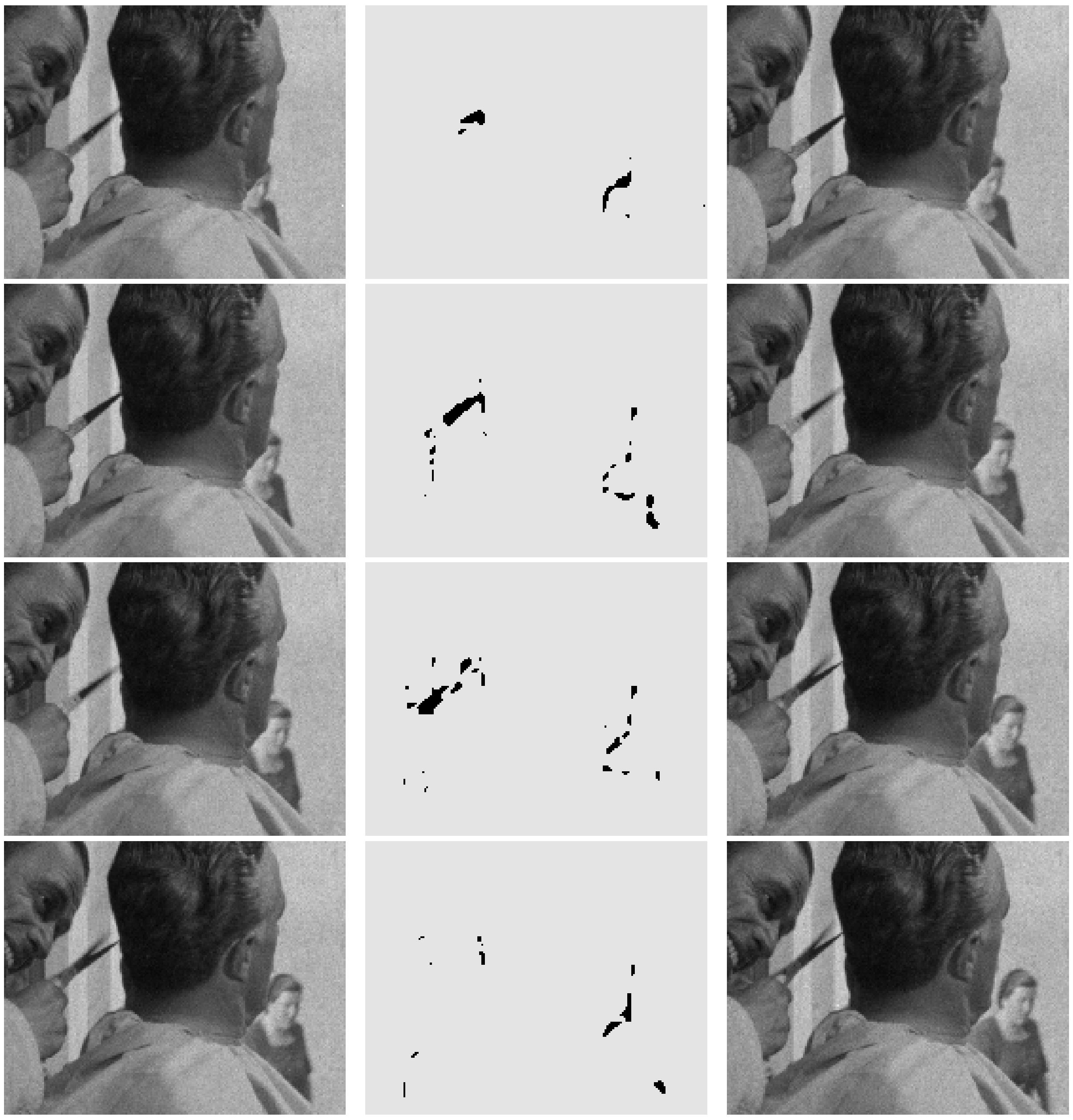


Figure 2.19: *This figure shows 4 consecutive discontinuity fields (middle column) from a sequence exhibiting Pathological Motion. The frames in the left and right columns show the frames between which the discontinuity fields were estimated. In this case the discontinuities are caused by the intermittent motion of the scissors and the uncovering of the woman in the background. The interpretation step examines sites where discontinuities occur in both of the central fields and tries to classify them in terms of Pathological Motion and Missing Data. In this examples, all the detected discontinuities are due to long term Pathological Motion.*

2. A temporal discontinuity exists in one of the fields - This is caused by an occlusion or uncovering. No action is necessary as this form of PM is not usually confused with Missing Data.

3. Temporal discontinuities exist in both fields - This case is equivalent to Missing Data detected using a three frame window. However it is possible that it could also be a form of PM. To distinguish between them the other discontinuity fields are examined. A search for temporal discontinuities is conducted in a radius about the pixel under consideration under test and if the number discontinuities exceeds a threshold the pixel is flagged as a PM rather than a Missing Data site.

2.4.2 Classification of Motion Blur [84, 85]

In his thesis [84], Rares outlines another PM detection algorithm which classifies motion estimation failures as either PM or blotch regions, based on their colour statistics. The form of PM detected by the algorithm is motion blur which is associated with fast moving objects. Areas of motion estimation failure are detected using a Complex Event Detector, developed by van Roosmalen [103], and classification between Motion Blur and blotches is performed by segmenting the image and examining the colour characteristics of segments corresponding to detected complex motion regions.

Colour Characteristics of Motion Blur

In an earlier paper [86], the author performs an analysis of the RGB colour histograms of images effected by Motion Blur and artefacts. The observation is made that artefacts are usually characterised by extreme pixel intensities and, therefore, one or more of the RGB channels will have very low or high values. Motion Blur is observed to cause a blending of the foreground object colour and the background colour. It is noted that the weighting of the average is not uniform and is dependent on the degree of blur. However, it is also noted that the degree of blurring is approximately equal in each of the RGB channels. This observation is expressed as follows

$$\frac{\mu^r_{br} - \mu^r_c}{\mu^r_{bg} - \mu^r_c} \approx \frac{\mu^g_{br} - \mu^g_c}{\mu^g_{bg} - \mu^g_c} \approx \frac{\mu^b_{br} - \mu^b_c}{\mu^b_{bg} - \mu^b_c} \tag{2.34}$$

where μ^r_{br}, μ^g_{br} and μ^b_{br} are the average RGB intensities of the blurred object in the current frame; where μ^r_c, μ^g_c and μ^b_c are the average RGB intensities of the unblurred object in a neighbouring frame and where μ^r_{bg}, μ^g_{bg} and μ^b_{bg} are the RGB intensities of the background in a neighbouring frame.

Algorithm Outline

The first stage of the algorithm involves detecting areas of motion estimation failure using a complex event detector outlined by van Roosmalen [103]. The image is then segmented using

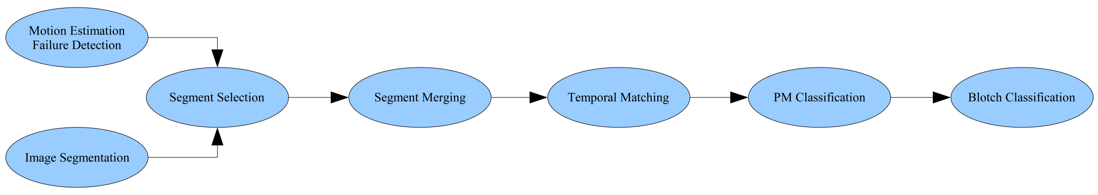


Figure 2.20: *A flow chart for the Rares algorithm.*

a watershed segmentation [105]. As the algorithm is only interested in regions where motion estimation failure occurs, segments corresponding to detected motion estimation failure regions are isolated. An overlap of 80% between each segment and detected motion estimation failure regions is suggested as a valid criterion for isolating the relevant segments. This is followed by a segment merging stage, in which neighbouring segments whose mean sum-squared-difference RGB value is within a threshold are merged.

At this stage classification between PM and blotches can begin. First of all, segments are classified, by temporally matching the classification in previously processed frames. Each candidate segment in the current frame is compared to a list of segments in the previously processed frame. If the candidate matches a segment in the previous frame (*i.e.* their colour similarity falls within a threshold), then the candidate segment is assigned the classification of the matched segment.

All remaining unclassified segments are then checked to see if they are regions of motion blur. By applying equation 2.34 to each candidate segment, where μ^r, μ^g and μ^b are the average RGB intensities of the candidate segment, the segment can by classified as PM by any finding two segments from the motion estimation failure region of the previously processed frame that satisfy the condition

$$\operatorname{var}\left(\frac{\mu^r-\mu^r_{s1}}{\mu^r_{s2}-\mu^r_{s1}},\ \frac{\mu^g-\mu^g_{s1}}{\mu^g_{s2}-\mu^g_{s1}},\ \frac{\mu^b-\mu^b_{s1}}{\mu^b_{s2}-\mu^b_{s1}}\right) < T_{\text{PM}} \tag{2.35}$$

where $[\mu^r_{s1},\ \mu^g_{s1},\ \mu^b_{s1}]$ and $[\mu^r_{s2},\ \mu^g_{s2},\ \mu^b_{s2}]$ are the average RGB values of the two segments from the previously processed frame and where T_{PM} is some small threshold.

Segments that are not classified as PM are now checked to see if they are blotches. As blotches are either dark or bright regions, it follows that the RGB values of blotches will also be high or low. A segment is classified as a blotch if the following condition is satisfied

$$\Big(\min(\mu^r,\ \mu^g,\ \mu^b) < T_{\text{Blotch}}\Big)\ \text{OR}\ \Big(\max(\mu^r,\ \mu^g,\ \mu^b) > (1-T_{\text{Blotch}})\Big) \tag{2.36}$$

where T_{Blotch} is a threshold and where the intensities are normalised to the scale from 0 to 1. The flowchart in Fig. 2.20 describes the relationship between each stage of the algorithm.

2.4.3 Two Level Segmentation for Handling Pathological Motion [50]

Pathological Motion is generally associated with the local motion of foreground objects in a sequence. Therefore by segmenting each frame into foreground objects and background the frame can be discriminated into regions where PM is likely to occur (*i.e.* foreground areas) and extremely unlikely to occur (*i.e.* The image background). In the paper [50], Kent *et al.* propose a method to perform a foreground/background segmentation and then to make Missing Data Detection more conservative in foreground areas.

Foreground/Background Segmentation

In order to detect local motion in a frame n, frames $n-1$ and $n+1$ are compensated for global affine motion with respect to frame n such that the respective backward and forward global DFDs are given by

$$\begin{aligned}\Delta_b^g &= I_n(\mathbf{x}) - I_{n-1}(A^g_{n,n-1}\mathbf{x} + \mathbf{d}^g_{n,n-1})\\ \Delta_f^g &= I_n(\mathbf{x}) - I_{n+1}(A^g_{n,n+1}\mathbf{x} + \mathbf{d}^g_{n,n+1})\end{aligned} \tag{2.37}$$

where $A^g_{n,n-1}$ and $\mathbf{d}^g_{n,n-1}$ and where $A^g_{n,n+1}$ and $\mathbf{d}^g_{n,n+1}$ describe the global affine motion between frames n and $n-1$ and frames n and $n+1$ respectively. The motion of the background areas should be the same as the estimated global motion and therefore the global DFDs should be low in these regions. Alternatively in foreground objects where local motion is present relative to the global motion the DFDs should be large.

Framework - A probabilistic framework is used to perform the segmentation. A state model similar to the Blotch model in Section 2.3.2 (See Table 2.1) is adopted. The factorisation for the posterior is given as follows

$$P(s(\mathbf{x})|\Delta_b^g(\mathbf{x}),\Delta_f^g(\mathbf{x}),S) \;\propto\; P_l(\Delta_b^g(\mathbf{x}),\Delta_f^g(\mathbf{x})|\, s(\mathbf{x})) \;\times\; P_s(s(\mathbf{x})|S) \tag{2.38}$$

where s is the state variable. States $s=0$, $s=1$ and $s=2$ correspond to background states while state $s=3$ corresponds to the foreground state.

Likelihood - The likelihood expression is given by

$$P_l(\Delta_b^g(\mathbf{x}),\Delta_f^g(\mathbf{x}) \mid s(\mathbf{x})) \;\propto\; \begin{cases} \exp-\left(\frac{\left(\Delta_b^g(\mathbf{x})\right)^2+\left(\Delta_f^g(\mathbf{x})\right)^2}{2\sigma_e^2}\right) & s=0\\ \exp-\left(\frac{\alpha^2}{2}+\frac{\left(\Delta_f^g(\mathbf{x})\right)^2}{2\sigma_e^2}\right) & s=1\\ \exp-\left(\frac{\left(\Delta_b^g(\mathbf{x})\right)^2}{2\sigma_e^2}+\frac{\alpha^2}{2}\right) & s=2\\ \exp-\left(\alpha^2\right) & s=3 \end{cases} \tag{2.39}$$

where α is a bias against the foreground state. A value of $\alpha = 2.76$ as this corresponds to the 99% confidence level given that a gaussian distribution is assumed for the DFDs.

Prior - The prior expresses a spatial smoothness constraint on s and is given by

$$P_s(s(\mathbf{x})|\ S) \ \propto\ \exp -\left(\Lambda_s \sum_{\mathbf{y}\in\mathbf{N}_S(\mathbf{x})} \lambda_y \big|s(\mathbf{x}) \neq s(\mathbf{y})\big|\right) \tag{2.40}$$

The value of s that maximises the posterior is chosen for each pixel and the ICM algorithm [3] is used to iterate the result. A multiresolution framework [41] is employed to improve robustness.

In the proposed algorithm, the MDD framework described in Section 2.3.3 is employed to detect blotches. In detected foreground areas (*i.e.* $s(\mathbf{x}) = 3$), the *alpha* penalty in the framework is increased. Furthermore, a post-process is applied to reject detected blotches that do not have high spatial contrast. In effect, MDD is made more conservative in the foreground regions.

2.5 Scope for a New Algorithm

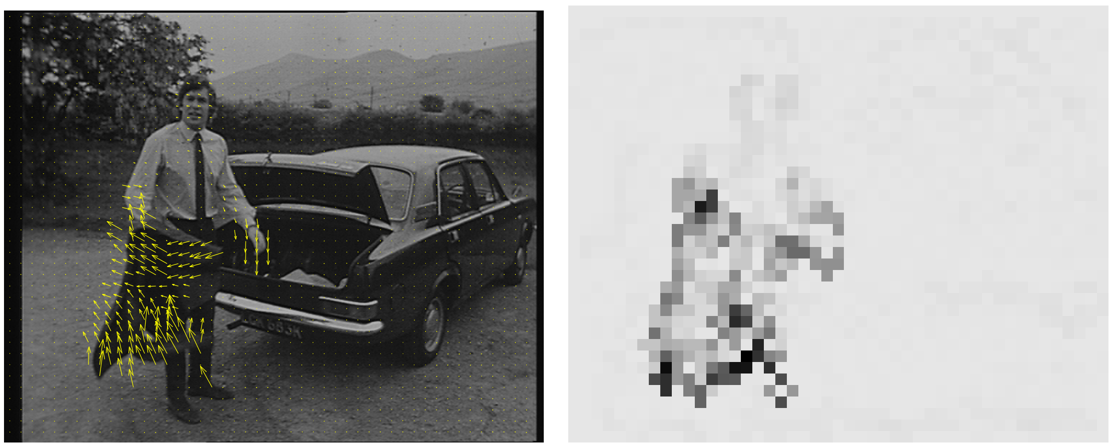
In [50], the authors discuss the limitations of Bornard's algorithm [8]. They point out that PM regions are assumed to be slow moving, as the size of any movement is restricted by the size of the neighbourhood used to search for temporal discontinuities in the interpretation step. The difficulties in dealing with blotches in PM regions are also discussed. If PM regions are excluded from blotch detection, these blotches will be ignored. On the other hand, if the PM regions are treated differently in blotch detection, and hence blotches have to be detected in a second pass, they point out the costly nature of implementing such a scheme on a large scale. As the authors are interested in developing a robust MDD algorithm for a movie post-production application, an algorithm which is easy to implement on a large scale[4] and which has the lowest possible false alarm rate is required. Therefore, a more cautious and compact technique is adopted whereby likely PM regions are treated differently in Missing Data Treatment.

There is some potential for improvement in Bornard's technique. In the algorithm, motion estimation failures are detected by looking for high DFD values. However, diagnosis of motion estimation failure could also be performed on the motion field itself. Motion Estimators make the assumption that motion vectors vary little over a spatial neighbourhood[5]. When motion estimation failure occurs, this assumption may no longer be valid. Therefore PM could be detected by finding areas of the motion vector field which are not smooth. A suitable measure of smoothness of a vector field is its divergence and so high divergence values are indicators of motion estimation failure (See Fig. 2.21).

Bornard's algorithm is the extension of the Morris algorithm [73] for blotch detection to a five frame temporal window. Like the Morris algorithm, it does not apply the probabilistic framework to detecting blotches or PM directly, but rather the temporal discontinuities are estimated from the probabilistic framework and are then interpreted in a separate stage. In section 2.3.3, it was discussed how a probabilistic framework could be used to detect missing data over a three

[4]A feature length film will contain tens of thousands of frames

[5]The *Smooth Local Flow Assumption*



(a) Frame with superimposed motion (b) Vector Field Divergence

Figure 2.21: *The jacket in the left image is an example of non-rigid motion and is pathological. The motion field in this region is not smooth and as a result the divergence of the motion field in this region has a high absolute value (right). The dark colours represent high divergence values.*

frame window. It follows logically, that there is a possibility for a new robust MDD algorithm algorithm along the lines of Bornard's technique which integrates the interpretation stage into a single probabilistic framework that classifies between regions of PM and missing data directly. Such an algorithm, would allow more prior information on blotches and PM into the framework, leading to a better performing algorithm.

Motivated by the potential for improvement in the Bornard technique, a new robust Missing Data Detection algorithm along these lines was developed as part of this work. The algorithm exploits the *Smooth Local Flow Assumption* to generate a new additional measure of motion estimation and also exploits the unified probabilistic framework to introduce new priors which improve the performance of the algorithm. The new algorithm is presented in the following chapter.

3 Pathological Motion Detection for Robust Missing Data Treatment

In this chapter a new robust Missing Data Detection algorithm is introduced which models for Pathological Motion. The algorithm employs an approach similar to the Bornard algorithm [8] but, inspired by the probabilistic model for blotch detection (Section 2.3.3), introduces a unified probabilistic framework to solve for the blotch detection problem. The unified framework models the blotch field as an MRF and, consequently, allows priors to be introduced on the blotch field. Another novel feature of the algorithm is the introduction of a new measure of motion estimation failure, given by the size of the motion field divergence, which considers areas of motion that are not smooth as Pathological.

Like Bornard's algorithm, the new algorithm includes extra temporal information to classify regions of motion estimation failure as being caused by PM or blotches. In effect, the aim is to distinguish between the impulsive temporal intensity profile of botches and the quasi-periodic profile of persistent or long-term PM. If a five frame window centred on the current frame is used, then four TDFs can be computed between consecutive frames over the window. When a blotch exists in the current frame, temporal discontinuities will occur in the two TDFs neighbouring the current frame. However, temporal discontinuities should not occur in the two other fields.

The algorithms differ in the method of classification between PM and missing data. Bornard does not directly detect PM and blotches using a probabilistic model. Instead the probabilistic

This work has been published in part in [19].

framework is used to detect temporal discontinuities and this information is subsequently interpreted in a non-probabilistic manner. The goal of the new algorithm is to develop an integrated probabilistic framework to directly detect blotches and PM. This is achieved by considering all the possible combinations of temporal discontinuity in the five frame window. The configurations of the temporal discontinuities at a pixel directly determines the result at that site. Each pixel can be either a part of a blotch, be in a region of PM and a normal well-behaved site. The integrated framework gives more control over the final result by allowing prior knowledge of the PM and blotch fields to be incorporated into the estimation, something which is not possible with the Bornard algorithm.

Apart from the temporal discontinuity feature, there are many other features that are also indicators of PM. Recognising this, the new algorithm augments the temporal discontinuity-based detection outlined above by incorporating another feature into the probabilistic framework. The chosen feature is the smoothness of the motion vector fields of the frame under consideration and also of the motion fields of the neighbouring frames. This stems from the observation that PM often results in the smoothness assumption of the motion field being violated. This is especially true with non-rigid body motion, where the motion of the object is varying rapidly along its contour. As the resolutions of motion fields are typically lower than the image resolution, diagnosis of PM from motion fields gives a coarser diagnosis and so is more suited for finding large regions of PM.

The new algorithm incorporates both measures of motion estimation failure into the probabilistic framework. The smoothness information is used as a guide for a more traditional temporal discontinuity based detector. Including the smoothness information improves the robustness of the algorithm by indicating blocks of the image which are likely to be PM regions. In these blocks, a bias is introduced toward the detection of PM. The next section of the chapter introduces the new algorithm. Before the probabilistic framework is introduced in Section 3.3, the blotch model is introduced. The outline of the framework is followed by a description of the implementation details of the algorithm. Section 3.4 evaluates the algorithm, comparing it with both the Bornard algorithm and a selection of 3-frame blotch detectors (*e.g.* SDIp, ROD). Finally comments are made on the algorithm and directions for further research and development in this area are discussed.

3.1 Algorithm Overview

The goal of the algorithm is to segment the current frame into regions of PM, missing data and uncorrupted sites. To achieve this, a label field, $l(\mathbf{x})$, is defined as follows

$$l(\mathbf{x}) = \begin{cases} 0 & \text{No missing data or Pathological Motion} \\ 1 & \text{Missing Data} \\ 2 & \text{Pathological Motion.} \end{cases} \tag{3.1}$$

Every pixel in the image is assigned one of the three labels. Like the Bornard algorithm, a pixel can not be both a missing data site and PM site. Although there is no reason why a blotch could not occur in a PM region, it would not be possible to reliably detect a blotch using a purely temporal discontinuity based detector. Furthermore, a PM detection implies that the data at any blotch cannot be interpolated temporally. Consequently, interpolation must be performed using spatial rather than temporal information. In short, the blotch model outlined here aims to reliably detect as much missing data as possible, while also highlighting the regions of the image where missing data cannot be robustly detected. These regions must be processed separately using an alternative MDT strategy. Two features are used to estimate the label field, the configuration of temporal discontinuities and the smoothness of the motion field.

3.1.1 Temporal Discontinuity Based Detection

The Displaced Frame Difference between a pair of neighbouring frames is used as the measure of temporal discontinuity. A temporal discontinuity is said to exist at a given site if the absolute value of the DFD at the site is sufficiently large. In the five frame window there are four pairs of neighbouring frames where the frames are denoted by $I_{n-2}(\mathbf{x})$, $I_{n-1}(\mathbf{x})$, $I_n(\mathbf{x})$, $I_{n+1}(\mathbf{x})$ and $I_{n+2}(\mathbf{x})$. A DFD can be measured between each pair, with a binary Temporal Discontinuity Field (TDF) associated with each DFD. Consequently, four DFDs ($\Delta_{n-2}(\mathbf{x})$, $\Delta_{n-1}(\mathbf{x})$, $\Delta_{n+1}(\mathbf{x})$ and $\Delta_{n+2}(\mathbf{x})$) and TDFs ($t_{n-2}(\mathbf{x})$, $t_{n-1}(\mathbf{x})$, $t_{n+1}(\mathbf{x})$ and $t_{n+2}(\mathbf{x})$) are estimated over the five-frame window. There are sixteen possible configurations of the four TDFs and a state field $s(\mathbf{x})$ is defined which describes the configuration of a site. Each configuration is directly mapped to a value of $l(\mathbf{x})$ (Table 3.1).

Missing data regions are characterised by an impulsive temporal intensity profile. Therefore, if a blotch exists in frame n, the absolute values of the DFDs between both frames n and $n-1$ ($\Delta_{n-1}(\mathbf{x})$) and frames n and $n+1$ ($\Delta_{n+1}(\mathbf{x})$) will be large but the other two DFDs will have small values. This results in a predictable temporal discontinuity profile described by $s(\mathbf{x}) = 6$ in table 3.1. On the other hand, the quasi-periodic profile of long-term PM will result high absolute values in more than two DFDs. Consequently, any temporal discontinuity configuration with two or more temporal discontinuities present, apart from the missing data case, is considered to be associated with PM (Table 3.1).

3.2 Motion Field Smoothness Based Detection

The second feature used to detect PM is the smoothness of the motion field. The vector field divergence is used the as the basis of the smoothness measure. Smooth vector fields have low divergences while high divergences result from vector fields that are not spatially smooth. By estimating the divergences of all the motion fields in the five-frame window, a smoothness energy term, $E_{\text{div}}(\mathbf{x})$, can be defined which describes the smoothness of the motion fields. This

State, $s(\mathbf{x})$	**Configuration, $\mathbf{t}(\mathbf{x})$**	$l(\mathbf{x})$	**State, $s(\mathbf{x})$**	**Configuration, $\mathbf{t}(\mathbf{x})$**	$l(\mathbf{x})$
0	$0,0,0,0$	0	8	$1,0,0,0$	0
1	$0,0,0,1$	0	9	$1,0,0,1$	2
2	$0,0,1,0$	0	10	$1,0,1,0$	2
3	$\mathbf{0,0,1,1}$	**2***	11	$1,0,1,1$	2
4	$0,1,0,0$	0	**12**	$\mathbf{1,1,0,0}$	**2***
5	$0,1,0,1$	2	13	$1,1,0,1$	2
6	$\mathbf{0,1,1,0}$	***1***	14	$1,1,1,0$	2
7	$0,1,1,1$	2	15	$1,1,1,1$	2

Table 3.1: *The temporal discontinuity state model for the probabilistic framework. $s(\mathbf{x}) = 6$ represents the missing data case. Long-term PM is given by any configuration or state where at least two discontinuities exist except for the missind data configuration, $s(\mathbf{x}) = 6$. *States 3 and 12 can correspond to missing data in the neighbouring frames. It is considered here as PM as it is not caused by missing data in the current frame.*

Divergence State, $v(\mathbf{x})$	$l(\mathbf{x})$
0	0,1
1	2

Table 3.2: *The divergence state model for the probabilistic framework.*

measure of divergence will be large in the presence of long-term PM and low in other cases. The divergence measure is associated with a binary field $v(\mathbf{x})$, where a value of 0 corresponds to a low divergence measure and a value of 1 corresponds to a high divergence measure. A high divergence measure corresponds to Pathological Motion and so $l(\mathbf{x}) = 2$. A low value indicates a well-behaved or missing data site, and consequently $l(\mathbf{x}) = 0$ or 1. This relationship is described in Table 3.2.

The fields s and v act as auxiliary variables which dictate the final value of $l(\mathbf{x})$ in the probabilistic framework. For a given value of s or v the value of l is directly defined through tables 3.1 and 3.2. Before the description of the probabilistic framework can proceed, measures for temporal discontinuity and divergence need to be introduced.

3.3 Probabilistic Framework

The framework derives an estimate of $s(\mathbf{x})$ from the posterior $P(s(\mathbf{x})|\mathbf{\Delta}_n(\mathbf{x}), E_{\text{div}}(\mathbf{x}))$. For convenience of notation, the four DFDs have been grouped into a vector valued function $\mathbf{\Delta}_n$ where $\mathbf{\Delta_n}(\mathbf{x}) = \left[\Delta_{n-2}(\mathbf{x}),\, \Delta_{n-1}(\mathbf{x}),\, \Delta_{n+1}(\mathbf{x}),\, \Delta_{n+2}(\mathbf{x})\right]$.

The posterior is factorised in a Bayesian fashion as follows

$$\begin{aligned} P(s|\mathbf{\Delta}_n, E_{\text{div}}) &\propto P(s, \mathbf{\Delta}_n, E_{\text{div}}) \\ &\propto P_t(\mathbf{\Delta}_n|s) \times P_d(E_{\text{div}}|s) \times P_r(s) \end{aligned} \tag{3.2}$$

where the index $\mathbf{x}$ has been dropped for clarity. There are two likelihoods associated with the framework, $P_t(.)$ associated with the DFDs of the window and $P_d(.)$ associated with the divergence measure. The final term is a prior on the state field $s(\mathbf{x})$. Mathematically, the correct form of this expansion is $P_t(\mathbf{\Delta}_n|E_{\text{div}}, s)$ $\times P_d(E_{\text{div}}|s)$ $\times P_s(s)$. Although it seems likely that some relationship between the DFDs and divergence measure exists, for the purposes of this framework it assumed that they are statistically independent.

Rather than considering the unknown variables $l(\mathbf{x})$, $v(\mathbf{x})$ and $s(\mathbf{x})$ as separate random variables, only one random variable $s(\mathbf{x})$ is considered. The values of $l(\mathbf{x})$ and $v(\mathbf{x})$ can then be determined from the estimate of $s(\mathbf{x})$ according to tables 3.1 and 3.2. This is a reinterpretation of the probabilistic frameworks proposed by Kokaram (eg [58]) in which each variable is treated as a separate random variable. In effect, the framework attempts to maximise the joint posterior distribution of all the variables (*i.e.* $P(l, s, v|.)$). However, the factorisation employed in that framework considers the variables to be statistically independent (*i.e.* $P(l, s, v) = P(l)P(s)P(v)$) which is clearly not the case as these variables share a deterministic relationship according to tables 3.1 and 3.2.

3.3.1 Temporal Discontinuity Likelihood

Before the temporal discontinuity likelihood can be introduced, the method for calculating the DFDs must be outlined. As has been outlined in Section 3.1.1 there are four DFDs to be estimated described by the four dimensional DFD vector $\mathbf{\Delta}_n(\mathbf{x})$. The DFDs are calculated by compensating the motion of each frame of window relative to the central frame. If the image sequence model is the local translation model given by

$$\begin{aligned} I_n(\mathbf{x}) &= I_{n-1}(\mathbf{x} + \mathbf{d}_{n,n-1}(\mathbf{x})) + e(\mathbf{x}) \\ &= I_{n+1}(\mathbf{x} + \mathbf{d}_{n,n+1}(\mathbf{x})) + e(\mathbf{x}), \end{aligned} \tag{3.3}$$

then the five motion compensated frames (I'_{n-2}, I'_{n-1}, I'_n, I'_{n+1}, I'_{n+2}) are given by

$$\begin{aligned} I'_{n-2}(\mathbf{x}) &= I_{n-2}\Big(\mathbf{x} + \mathbf{d}_{n,n-1}(\mathbf{x}) + \mathbf{d}_{n-1,n-2}\big(\mathbf{x} + \mathbf{d}_{n,n-1}(\mathbf{x})\big)\Big) \\ I'_{n-1}(\mathbf{x}) &= I_{n-1}\Big(\mathbf{x} + \mathbf{d}_{n,n-1}(\mathbf{x})\Big) \\ I'_n(\mathbf{x}) &= I_n(\mathbf{x}) \\ I'_{n+1}(\mathbf{x}) &= I_{n+1}\Big(\mathbf{x} + \mathbf{d}_{n,n+1}(\mathbf{x})\Big) \\ I'_{n+2}(\mathbf{x}) &= I_{n+2}\Big(\mathbf{x} + \mathbf{d}_{n,n+1}(\mathbf{x}) + \mathbf{d}_{n+1,n+2}\big(\mathbf{x} + \mathbf{d}_{n,n+1}(\mathbf{x})\big)\Big). \end{aligned} \tag{3.4}$$

Consequently, the four DFDs of the window are given by

$$\begin{aligned}\Delta_{n-2}(\mathbf{x}) &= I'_{n-1}(\mathbf{x}) - I'_{n-2}(\mathbf{x})\\ \Delta_{n-1}(\mathbf{x}) &= I'_{n}(\mathbf{x}) - I'_{n-1}(\mathbf{x})\\ \Delta_{n+1}(\mathbf{x}) &= I'_{n}(\mathbf{x}) - I'_{n+1}(\mathbf{x})\\ \Delta_{n+2}(\mathbf{x}) &= I'_{n+1}(\mathbf{x}) - I'_{n+2}(\mathbf{x}).\end{aligned} \tag{3.5}$$

Motion Estimation

The motion fields used to perform the motion compensation above are performed as a pre-process to the algorithm. For every frame of the sequence, a backward and forward motion field must be estimated (*i.e.* for motion in the current frame relative to the previous and next frames of the sequence respectively). Any motion estimator can be used to generate the motion fields. Local or Global estimators can be used. In fact, the algorithm can be performed without any motion estimation. The choice or motion estimator affects the correct detection and false alarm rate for missing data and also the PM detection rate. In the tests described in Section 3.4 the gradient-based algorithm outlined in [54] is used.

The Likelihood Expression

The data likelihood $P_t(\mathbf{\Delta}_n|s)$ constrains each DFD to be low when a temporal discontinuity does not exist. The expression for the likelihood can be expressed in vector form as

$$P_t(\mathbf{\Delta}_n|s) \;\propto\; \exp - \sum_{k=0}^{3} \left\{ \frac{\mathbf{\Delta_n}[k]^2}{2\sigma_e^2}\big(1 - \mathbf{t}[k]\big) + \frac{\alpha_k^2}{2}\mathbf{t}[k] \right\} \tag{3.6}$$

where σ_e^2 is the variance of the model error $e(\mathbf{x})$.

The overall likelihood energy is the sum of four components, where each component is equivalent to the likelihood energy of a probabilistic temporal discontinuity detector[1]. For each component the ML condition for a temporal discontinuity is

$$\big(\mathbf{\Delta_n}[k](x)\big)^2 > \sigma_e^2\alpha_k^2. \tag{3.7}$$

The temporal discontinuity vector $\mathbf{t}(\mathbf{x})$ is equivalent to a binary representation of $s(\mathbf{x})$. The

[1]See Equation 2.19 in Chapter 2, Section 2.3.2

likelihood energy for each value of $s(\mathbf{x})$ is given by

$$-\log\big(P_t(\mathbf{\Delta}_n|s)\big) \propto \begin{cases}
\frac{\Delta_{n-2}^2}{2\sigma_e^2} + \frac{\Delta_{n-1}^2}{2\sigma_e^2} + \frac{\Delta_{n+1}^2}{2\sigma_e^2} + \frac{\Delta_{n+2}^2}{2\sigma_e^2} & \text{if } s(\mathbf{x}) = 0 \\
\frac{\Delta_{n-2}^2}{2\sigma_e^2} + \frac{\Delta_{n-1}^2}{2\sigma_e^2} + \frac{\Delta_{n+1}^2}{2\sigma_e^2} + \frac{\alpha_4^2}{2} & \text{if } s(\mathbf{x}) = 1 \\
\frac{\Delta_{n-2}^2}{2\sigma_e^2} + \frac{\Delta_{n-1}^2}{2\sigma_e^2} + \frac{\alpha_3^2}{2} + \frac{\Delta_{n+2}^2}{2\sigma_e^2} & \text{if } s(\mathbf{x}) = 2 \\
\frac{\Delta_{n-2}^2}{2\sigma_e^2} + \frac{\Delta_{n-1}^2}{2\sigma_e^2} + \frac{\alpha_3^2}{2} + \frac{\alpha_4^2}{2} & \text{if } s(\mathbf{x}) = 3 \\
\frac{\Delta_{n-2}^2}{2\sigma_e^2} + \frac{\alpha_2^2}{2} + \frac{\Delta_{n+1}^2}{2\sigma_e^2} + \frac{\Delta_{n+2}^2}{2\sigma_e^2} & \text{if } s(\mathbf{x}) = 4 \\
\frac{\Delta_{n-2}^2}{2\sigma_e^2} + \frac{\alpha_2^2}{2} + \frac{\Delta_{n+1}^2}{2\sigma_e^2} + \frac{\alpha_4^2}{2} & \text{if } s(\mathbf{x}) = 5 \\
\frac{\Delta_{n-2}^2}{2\sigma_e^2} + \frac{\alpha_2^2}{2} + \frac{\alpha_3^2}{2} + \frac{\Delta_{n+2}^2}{2\sigma_e^2} & \text{if } s(\mathbf{x}) = 6 \\
\frac{\Delta_{n-2}^2}{2\sigma_e^2} + \frac{\alpha_2^2}{2} + \frac{\alpha_3^2}{2} + \frac{\alpha_4^2}{2} & \text{if } s(\mathbf{x}) = 7 \\
\frac{\alpha_1^2}{2} + \frac{\Delta_{n-1}^2}{2\sigma_e^2} + \frac{\Delta_{n+1}^2}{2\sigma_e^2} + \frac{\Delta_{n+2}^2}{2\sigma_e^2} & \text{if } s(\mathbf{x}) = 8 \\
\frac{\alpha_1^2}{2} + \frac{\Delta_{n-1}^2}{2\sigma_e^2} + \frac{\Delta_{n+1}^2}{2\sigma_e^2} + \frac{\alpha_4^2}{2} & \text{if } s(\mathbf{x}) = 9 \\
\frac{\alpha_1^2}{2} + \frac{\Delta_{n-1}^2}{2\sigma_e^2} + \frac{\alpha_3^2}{2} + \frac{\Delta_{n+2}^2}{2\sigma_e^2} & \text{if } s(\mathbf{x}) = 10 \\
\frac{\alpha_1^2}{2} + \frac{\Delta_{n-1}^2}{2\sigma_e^2} + \frac{\alpha_3^2}{2} + \frac{\alpha_4^2}{2} & \text{if } s(\mathbf{x}) = 11 \\
\frac{\alpha_1^2}{2} + \frac{\alpha_2^2}{2} + \frac{\Delta_{n+1}^2}{2\sigma_e^2} + \frac{\Delta_{n+2}^2}{2\sigma_e^2} & \text{if } s(\mathbf{x}) = 12 \\
\frac{\alpha_1^2}{2} + \frac{\alpha_2^2}{2} + \frac{\Delta_{n+1}^2}{2\sigma_e^2} + \frac{\alpha_4^2}{2} & \text{if } s(\mathbf{x}) = 13 \\
\frac{\alpha_1^2}{2} + \frac{\alpha_2^2}{2} + \frac{\alpha_3^2}{2} + \frac{\Delta_{n+2}^2}{2\sigma_e^2} & \text{if } s(\mathbf{x}) = 14 \\
\frac{\alpha_1^2}{2} + \frac{\alpha_2^2}{2} + \frac{\alpha_3^2}{2} + \frac{\alpha_4^2}{2} & \text{if } s(\mathbf{x}) = 15
\end{cases} \tag{3.8}$$

The value of the model error variance, σ_e^2, is determined by estimating the variance of the DFDs when $s(\mathbf{x}) = 0$. The threshold α is allowed to vary for each component of the likelihood. The α_2 and α_3 thresholds for the central DFDs (Δ_{n-2} and Δ_{n+2}) are linked to the 99% confidence values and have a value of 2.76. Alternatively, the value can be linked to a deterministic threshold on the DFD, δ_t. The equivalent α threshold to δ_t can be derived from equation 3.7 and is given by

$$\alpha_2 = \alpha_3 = \sqrt{\frac{\delta_t^2}{2\sigma_e^2}} \tag{3.9}$$

Detection of missing data is made more conservative by using a lower α (*i.e.* for α_1 and α_4) on the outer DFDs (*i.e.* Δ_{n-2} and Δ_{n+2}) than on the central DFDs. The outer threshold ratio, α_r, is defined as the ratio of outer thresholds α_1 and α_4 to the inner thresholds α_2 and α_3 as follows

$$\alpha_r = \frac{\alpha_1}{\alpha_2} = \frac{\alpha_4}{\alpha_3}. \tag{3.10}$$

A typical value of α_r is 0.5.

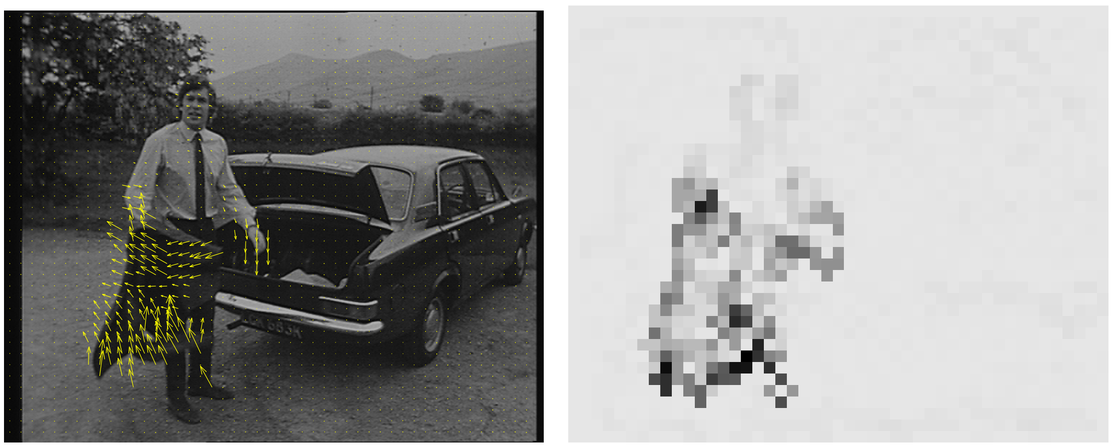

(a) Frame with superimposed motion (b) Vector Field Divergence

Figure 3.1: *The motion of the jacket in the left image is an example of PM. The motion field in this region is not smooth and as a result the divergence of the motion field in this region has a high absolute value. The dark colours represent high divergence values.*

3.3.2 Divergence Likelihood

The divergence of a vector or flow field measures the rate at which flow exits a given point of the flow field. The divergence of a 2D motion field, $\mathbf{d}(\mathbf{x})$, is defined as

$$\begin{aligned}\mathrm{div}\big(\mathbf{d}(\mathbf{x})\big) &= \nabla.\mathbf{d}(\mathbf{x}) \\ &= \Big(\frac{\partial}{\partial x}\mathbf{i} + \frac{\partial}{\partial y}\mathbf{j}\Big).\mathbf{d}(\mathbf{x}) \\ &= \frac{\partial \mathbf{d}_x(\mathbf{x})}{\partial x} + \frac{\partial \mathbf{d}_y(\mathbf{x})}{\partial y} \qquad (3.11)\end{aligned}$$

where $\mathbf{d}_x$ and $\mathbf{d}_y$ are the horizontal and vertical components of the motion field respectively. When PM occurs, the smoothness of the motion field is violated and the absolute value of the divergence is large (Fig. 3.1). When PM persists in a number of frames, then the divergences of a series of motion fields will be large. By finding the divergence of the motion fields involved in the motion-compensation of the 5-frame window, a divergence measure $E_{\mathrm{div}}(\mathbf{x})$ can be derived as follows

$$E_{\mathrm{div}}(\mathbf{x}) = \sum_{k=n-1}^{n} \left|\mathrm{div}\big(\mathbf{d}_{\mathbf{k},\mathbf{k-1}}(\mathbf{x})\big)\right| + \sum_{k=n}^{n+1}\left|\mathrm{div}\big(\mathbf{d}_{\mathbf{k},\mathbf{k+1}}(\mathbf{x})\big)\right|. \qquad (3.12)$$

In other words, $E_{\mathrm{div}}(\mathbf{x})$ is the sum of the absolute divergences of the four motion fields used for motion compensation.

The divergence likelihood constrains E_{div} to be low for either missing data or unaffected

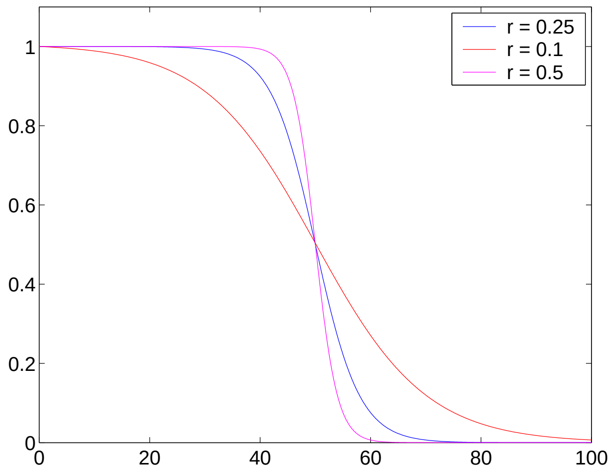


Figure 3.2: *This figure shows three plots of $\phi(a)$ for $a \in [0, 100]$. In this example $c = 50$. As r increases the drop in energy about e_t becomes the more abrupt.*

states. The expression used for the likelihood energy in the the framework is

$$-\log\big(P_d(E_{\text{div}}(\mathbf{x})|s(\mathbf{x}))\big) \propto \begin{cases} \Lambda_d\Big(1 - \phi\big(E_{\text{div}}(\mathbf{x})\big)\Big) & \text{if } s = \{0,1,2,4,6,8\} \\ 0 & \text{if } s = \{3,5,7,9,10, \\ & 11,12,13,14,15\} \end{cases} \tag{3.13}$$

where Λ_d is the weight of the divergence likelihood[2] in the framework and where $s = \{3,\ 5,\ 7,\ 9,\ 10,\ 11,\ 12,\ 13,\ 14,\ 15\})$ is the set of PM or missing data states (*i.e.* $l = \{1, 2\}$, $v = 1$) and $s = \{0,\ 1,\ 2,\ 4,\ 6,\ 8\})$ is the set of unaffected sites (*i.e.* $l = 0$, $v = 0$). $\phi\big(E_{\text{div}}(\mathbf{x})\big)$ is a sigmoid function defined on the divergence measure, E_{div}, and is given by

$$\phi(a) = \frac{1 + e^{r.e_t}}{e^{r.e_t}} \times \frac{e^{-r(a-e_t)}}{1 + e^{-r(a-e_t)}} \tag{3.14}$$

where r and e_t are positive real numbers (See Fig. 3.2). Effectively, the divergence likelihood acts as a bias towards the detection of PM when the divergence measure is large. e_t acts as a threshold for detecting high divergences.

3.3.3 Priors

The prior expression used in the framework is

$$P_r(s(\mathbf{x})) = P_s(l(\mathbf{x})|L) \tag{3.15}$$

which is a spatial smoothness prior on $l(\mathbf{x})$. A spatial smoothness prior on $s(\mathbf{x})$ is not included in the framework. The main reason for excluding the prior is to save on computational complexity.

[2]A value of $\Lambda_d = 1$ is typically used in the experiments outlined in Section 3.4.

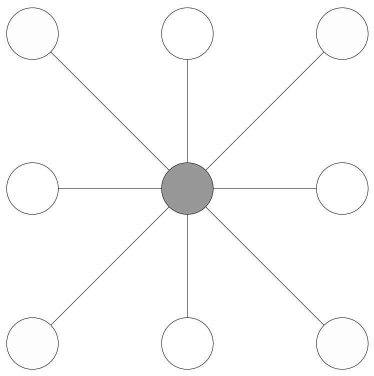


Figure 3.3: *This figure shows the 8-pixel first order neighbourhood of the grey pixel* $\mathbf{x}$*. The value of* $\lambda_{\mathbf{y}}$ *is assigned a value corresponding of the pixel* $\mathbf{y}$*'s distance from* $\mathbf{x}$*. For the pixels aligned either horizontally or vertically with* $\mathbf{x}$*,* $\lambda_{\mathbf{y}} = 1$*. For the diagonal neighbours,* $\lambda_{\mathbf{y}} = 1/\sqrt{2}$*.*

The complexity of evaluating the prior on a field is proportional to the number of possible values. As there are 16 possible values for $s(\mathbf{x})$ and 3 values for $l(\mathbf{x})$, it follows that evaluating the prior on l is less computationally intensive. As $l(\mathbf{x})$ is the field of interest the spatial smoothness of the result is not compromised.

The prior on l is given by

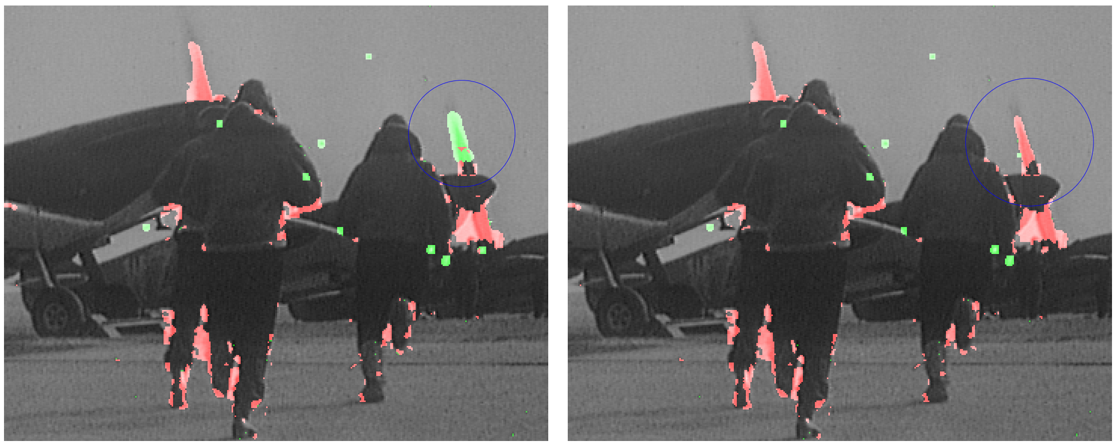
$$P(l(\mathbf{x})|L) \propto \exp -\left\{ \Lambda_l \sum_{\mathbf{y} \in \mathcal{N}_s(\mathbf{x})} \lambda_{\mathbf{y}} \Big(1 - u(\mathbf{x}, \mathbf{y})\Big) \rho\Big(l(\mathbf{x}), l(\mathbf{y})\Big) \right\} \tag{3.16}$$

where Λ_l is the weight of the prior in the framework (a value of i is commonly used), where $\mathcal{N}_s(\mathbf{x})$ is a spatial neighbourhood of $\mathbf{x}$ and where $\lambda_{\mathbf{y}}$ is a weight inversely proportional to $\|\mathbf{x} - \mathbf{y}\|$ (See Fig. 3.3). $\rho\Big(l(\mathbf{x}), l(\mathbf{y})\Big)$ is the energy penalty for neighbours $\mathbf{x}$ and $\mathbf{y}$ having the labels $l(\mathbf{x})$ and $l(\mathbf{y})$.

$u(\mathbf{x}, \mathbf{y})$ is a binary edge field which is 1 when an image edge exists between $\mathbf{x}$ and $\mathbf{y}$ and is derived from the central frame $I_n(\mathbf{x})$. The effect of $u(\mathbf{x}, \mathbf{y})$ is to turn off the smoothness across image edges, allowing sharp transitions in $l(\mathbf{x})$ across edges [65]. This is a particularly appropriate model for blotches, as the boundary of a blotch typically coincides with an image edge.

The PM Bias

Usually the value of ρ is 0 for all $l(\mathbf{x}) = l(\mathbf{y})$ and 1 for all $l(\mathbf{x}) \neq l(\mathbf{y})$. However, in this framework a bias is introduced into ρ to prevent a part of an "object" being classified as PM and another as missing data (Fig. 3.4). This is achieved by increasing the penalty for $\rho(1, 2)$ (*i.e.* increasing the penalty for assigning a site which has a neighbouring PM site as a missing data site). The effect is to make blotch detection more conservative by favouring PM detection at such sites.



(a) Result without the Missing Data Bias (b) Result with the Missing Data Bias

Figure 3.4: *This figure shows the effect of including the missing data bias in the framework. In the images above a propellor blade undergoing PM is highlighted by the blue circle. Detected PM is highlighted in red and detected missing data is highlighted in green. When the standard smoothness prior is used (i.e. $K = 1$) , the top part of the blade is incorrectly detected as missing data. Introducing the PM bias penalises against such a configuration by favouring the detection of PM over missing data. Consequently, the entire blade has been classified as PM when the bias is introduced.*

The expression used for ρ is

$$\rho\Big(l(\mathbf{x}), l(\mathbf{y})\Big) = \begin{cases} 0 & \text{if } l(\mathbf{x}) = l(\mathbf{y}) \\ K & \text{if } l(\mathbf{x}) = 1 \text{ AND } l(\mathbf{y}) = 2 \\ 1 & \text{otherwise} \end{cases} \tag{3.17}$$

where $K > 1$. A value of 5 is used for K in the experiments performed in Section 3.4.

3.3.4 Solving for $l(\mathbf{x})$

An estimate for $l(\mathbf{x})$ is found by finding an estimate of $s(\mathbf{x})$. The ICM optimisation strategy [3] is used to find estimates of $s(\mathbf{x})$. For every pixel in the image, $\mathbf{x}$, the MAP state is chosen as the new estimate for $s(\mathbf{x})$. As a new value of $s(\mathbf{x})$ is estimated for each pixel, the updated $s(\mathbf{x})$ is used to generate the new estimate of the next pixel. This process is iterated until the result converges, with a maximum of 20 iterations. The posterior for each value of s is evaluated according to

$$
\begin{aligned}
P(s=0|\mathbf{\Delta_n}) &\propto P_t(\mathbf{\Delta_n}|s=0) \times P_d(E_{\text{div}}|s=0) \times P_s(l=0|L)\\
P(s=1|\mathbf{\Delta_n}) &\propto P_t(\mathbf{\Delta_n}|s=1) \times P_d(E_{\text{div}}|s=1) \times P_s(l=0|L)\\
P(s=2|\mathbf{\Delta_n}) &\propto P_t(\mathbf{\Delta_n}|s=2) \times P_d(E_{\text{div}}|s=2) \times P_s(l=0|L)\\
P(s=3|\mathbf{\Delta_n}) &\propto P_t(\mathbf{\Delta_n}|s=3) \times P_d(E_{\text{div}}|s=3) \times P_s(l=2|L)\\
\\
P(s=4|\mathbf{\Delta_n}) &\propto P_t(\mathbf{\Delta_n}|s=4) \times P_d(E_{\text{div}}|s=4) \times P_s(l=0|L)\\
P(s=5|\mathbf{\Delta_n}) &\propto P_t(\mathbf{\Delta_n}|s=5) \times P_d(E_{\text{div}}|s=5) \times P_s(l=2|L)\\
P(s=6|\mathbf{\Delta_n}) &\propto P_t(\mathbf{\Delta_n}|s=6) \times P_d(E_{\text{div}}|s=6) \times P_s(l=1|L)\\
P(s=7|\mathbf{\Delta_n}) &\propto P_t(\mathbf{\Delta_n}|s=7) \times P_d(E_{\text{div}}|s=7) \times P_s(l=2|L)\\
\\
P(s=8|\mathbf{\Delta_n}) &\propto P_t(\mathbf{\Delta_n}|s=8) \times P_d(E_{\text{div}}|s=8) \times P_s(l=0|L)\\
P(s=9|\mathbf{\Delta_n}) &\propto P_t(\mathbf{\Delta_n}|s=9) \times P_d(E_{\text{div}}|s=9) \times P_s(l=2|L)\\
P(s=10|\mathbf{\Delta_n}) &\propto P_t(\mathbf{\Delta_n}|s=10) \times P_d(E_{\text{div}}|s=10) \times P_s(l=2|L)\\
P(s=11|\mathbf{\Delta_n}) &\propto P_t(\mathbf{\Delta_n}|s=11) \times P_d(E_{\text{div}}|s=11) \times P_s(l=2|L)\\
\\
P(s=12|\mathbf{\Delta_n}) &\propto P_t(\mathbf{\Delta_n}|s=12) \times P_d(E_{\text{div}}|s=12) \times P_s(l=2|L)\\
P(s=13|\mathbf{\Delta_n}) &\propto P_l(\mathbf{\Delta_n}|s=13) \times P_d(E_{\text{div}}|s=13) \times P_s(l=2|L)\\
P(s=14|\mathbf{\Delta_n}) &\propto P_t(\mathbf{\Delta_n}|s=14) \times P_d(E_{\text{div}}|s=14) \times P_s(l=2|L)\\
P(s=15|\mathbf{\Delta_n}) &\propto P_t(\mathbf{\Delta_n}|s=15) \times P_d(E_{\text{div}}|s=15) \times P_s(l=2|L).
\end{aligned}
\tag{3.18}
$$

The above expression outlines exactly how the posterior probability for each possible value of s. In particular, it describes how the prior probability is estimated for each value of s given that the spatial smoothness prior is enforced on the label field l rather than s.

ICM gives a sub-optimal estimate of s. The converged estimate represents a local maximum in the posterior PDF. Consequently, a good initialisation of the s is necessary to ensure that the converged result is close to the global maximum. A deterministic temporal discontinuity detector is used to estimate $\mathbf{t}(\mathbf{x})$ and is described by

$$
\begin{aligned}
t_{n-2}(\mathbf{x}) &= \begin{cases} 1 & \text{if } \Delta_{n-2}(\mathbf{x}) > \delta_t \\ 0 & \text{otherwise} \end{cases}, \quad & t_{n-1}(\mathbf{x}) &= \begin{cases} 1 & \text{if } \Delta_{n-1}(\mathbf{x}) > \delta_t \\ 0 & \text{otherwise} \end{cases},\\
t_{n+1}(\mathbf{x}) &= \begin{cases} 1 & \text{if } \Delta_{n+1}(\mathbf{x}) > \delta_t \\ 0 & \text{otherwise} \end{cases}, \quad & t_{n+2}(\mathbf{x}) &= \begin{cases} 1 & \text{if } \Delta_{n+2}(\mathbf{x}) > \delta_t \\ 0 & \text{otherwise} \end{cases}
\end{aligned}
\tag{3.19}
$$

where

$$\mathbf{t}(\mathbf{x}) = [t_{n-2}(\mathbf{x}), t_{n-1}(\mathbf{x}), t_{n+1}(\mathbf{x}), t_{n+2}(\mathbf{x})].$$

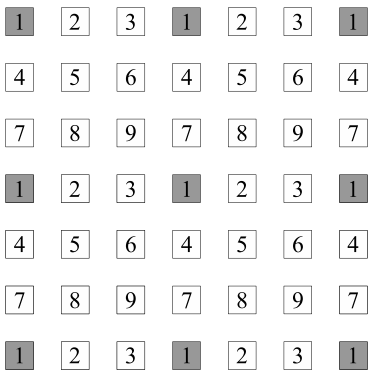


Figure 3.5: *This figure illustrates the operation of the checkerboard scan. Each square in the above image represents a pixel. The checkerboard scan simultaneously updates every third pixel along each row and column (the shaded pixels). In the first pass the pixels marked* 1 *are updated concurrently and in the second the pixels marked* 2 *are updated etc. . Accordingly, it takes 9 passes of the checkerboard scan to update every pixel.*

From the initial estimate of $\mathbf{t}(\mathbf{x})$, estimates of $s(\mathbf{x})$ and $l(\mathbf{x})$ can be found.

The order in which the pixels are updated at each iteration can bias the estimate. For example, a simple top-left to bottom-right raster scan can lead to a propagation of errors in the direction of the scan. In this algorithm, a "checkerboard" scan is used (See Fig. 3.5). The main advantage of the checkerboard scan is that error propagation at each iteration is limited to a 3×3 grid at each iteration. Furthermore, using the checkerboard scan can lead to significant computational efficiencies, due to the concurrent estimation of the new pixel values.

Multiresolution

A multiresolution scheme [41] is incorporated into the algorithm. Using the mulitresolution scheme results in faster convergence and the state field, $s(\mathbf{x})$, is more likely to converge to the global maximum of the posterior PDF.

A hierarchical pyramid of state fields of differing resolution is constructed, with the full resolution field at the bottom level of the pyramid (level 0) and with fields of successively coarser resolution at the higher levels of the pyramid. The horizontal and vertical resolution of the field at each level is half the resolution of the field at the level below. A similar pyramid is constructed for each of the DFDs and for the divergence measure, with each being filtered with a gaussian filter of variance 2.5 and then subsampled by a factor of two both horizontally and vertically. The coarser levels of these pyramids are generated by filtering the relevant DFD or divergence measure with a low pass filter.

The algorithm proceeds by initialising the state field, $s(\mathbf{x})$, at the coarsest level of the pyramid using equation 3.19. A new estimate of $s(\mathbf{x})$ at the coarsest level is obtained from the probabilistic framework and the new estimate is then used to initialising the framework at the

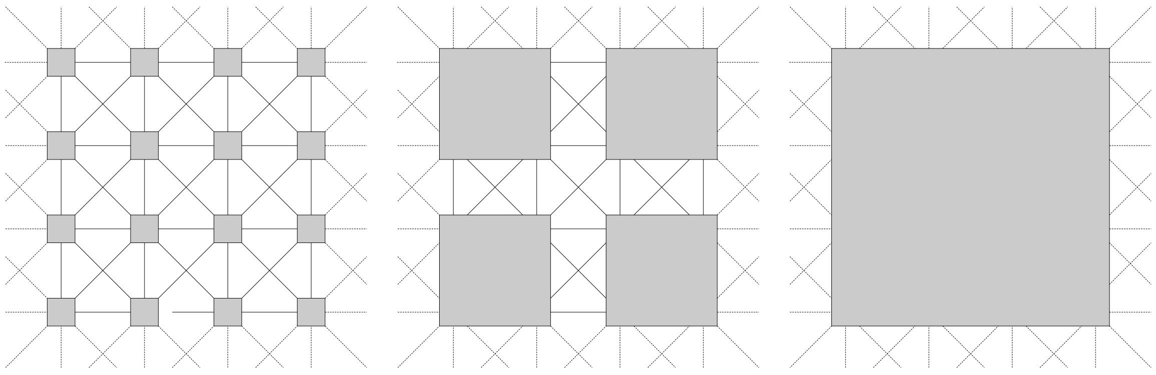

(a) Level 0 (b) Level 1 (c) Level 2

Figure 3.6: *Each image is a representation of a section of the state field at 3 lowest levels of the multiresolution framework. Each square represents a pixel in the field and the lines represent the links connecting neighbouring pixels. At the full resolution (level 0) each pixel as a single link to every pixel in its neighbourhood. However, at higher levels of the pyramid, the number of links between neighbouring vertical and horizontal pixels is greater than one, owing to the neighbourhood links at the full resolution. As a result, the spatial correlation between each pixel and its horizontal and vertical neighbours is increased relative to its diagonal neighbours.*

level below. This process continues until $s(\mathbf{x})$ has been estimated at full resolution.

An adjustment is made to the prior energy expression when estimating $s(\mathbf{x})$ at the coarser resolution. The weight $\lambda_{\mathbf{y}}$ is modified to reflect the increased spatial correlation between a pixel and horizontal and vertical neighbours relative its diagonal neighbours (See Fig. 3.6). At a level i of the pyramid , the value of $\lambda_{\mathbf{y}}$ is given by

$$\lambda_y = \begin{cases} 2^i + \sqrt{2}(2^i - 1) & \text{for vertical and horizontal neighbours} \\ \frac{1}{\sqrt{2}} & \text{for diagonal neighbours} \end{cases} \tag{3.20}$$

3.4 Results

The main contribution of the proposed algorithm is the introduction of a joint MRF model for blotches and PM (the label field $l(\mathbf{x})$). Pixels can either be clean, a blotch or PM. Even though blotches can exist at PM sites, such sites cannot be detected as blotches. Effectively, the correct detection rate for blotches is compromised in order to prevent the false detection of blotches due to PM. Another feature of the algorithm is the application a smoothness prior to the label field $l(\mathbf{x})$ instead of the full state field $s(\mathbf{x})$. This allows smoothness to be maximised on the field of interest (*i.e.* $l(\mathbf{x})$) while at the same time minimising the computational cost of estimating the spatial smoothness energy. This facilitates the introduction of a PM bias, which penalises the detection of neighbouring blotch and PM sites by adding an energy penalty K to the assignment of the site as a blotch for every PM site in its neighbourhood (See Fig. 3.4). The other significant features of the algorithm are the divergence likelihood and the outer threshold ratio α_r. The inclusion of the divergence likelihood introduces a new metric for measuring motion estimation failure while the outer threshold makes the detection of PM more likely by allowing a lower threshold for the detection of temporal discontinuities in the "outer" DFDs.

This section presents an evaluation of the proposed algorithm. It attempts to assess the contribution of the individual features to the overall performance of the algorithm, in particular the value of the divergence weight Λ_d, the PM bias K and the outer threshold ratio α_r. It also compares the performance of the algorithm to other missing data detectors, including conventional missing data detectors such as SDIp [55] and also the Bornard missing data/PM detection algorithm. This is achieved by comparing the results for $l(\mathbf{x})$ generated by each algorithm against a ground truth for blotches. Ground truths for dirt are generated using both Infra-Red scans and by manual detection of blotches. The correct detection and false alarm rates for missing data in the sequence are measured and are arranged in the form of Receiver Operating Characteristics (ROCs), with the control parameter being the value of the threshold δ_t.

However, ROCs cannot give a clear picture of the performance of the detector. They merely show the overall correct detection and false alarm rates for a frame or a sequence, giving no description of their distributions in the frame. Blotch detectors that model Pathological Motion will perform poorly in terms of correct detections as they are typically designed to prevent detection of blotches in regions of motion estimation failure. ROCs also cannot completely describe how effective the algorithm is at reducing false alarms, since they would not distinguish between PM and non-PM regions in a frame. Ideally, it would be desirable to ignore ground truth blotches in PM regions when estimating the correct detection rates. This could be achieved by using the masks for PM generated by the proposed algorithm as a ground truth for PM. However, not only would this bias the results in favour of the proposed algorithm, but it would also exclude blotches wrongly classified as PM (*e.g.* when blotches are located in similar positions in consecutive frames). In order to gain a complete understanding of the performance of the algorithm it is necessary to augment the ROC plots with a visual evaluation of the masks generated by the algorithm and also examine the detection rates of PM.

The remainder of this section outlines the experiments used to evaluate the performance of the algorithm. The algorithm is tested on two sequences with PM for which a ground truth for blotches was available and also on sequences for which no ground truth exists. Initially, the evaluation focuses on determining the contribution of the divergence likelihood, PM bias and the value of the outer threshold ratio to the performance of the algorithm (Section 3.4.3). The performance of the proposed algorithm is then compared to other blotch detectors in Section 3.4.4, including conventional detection algorithms as well as the Bornard algorithm. Finally a discussion of the results is presented. But before the evaluation begins, the two ground truth

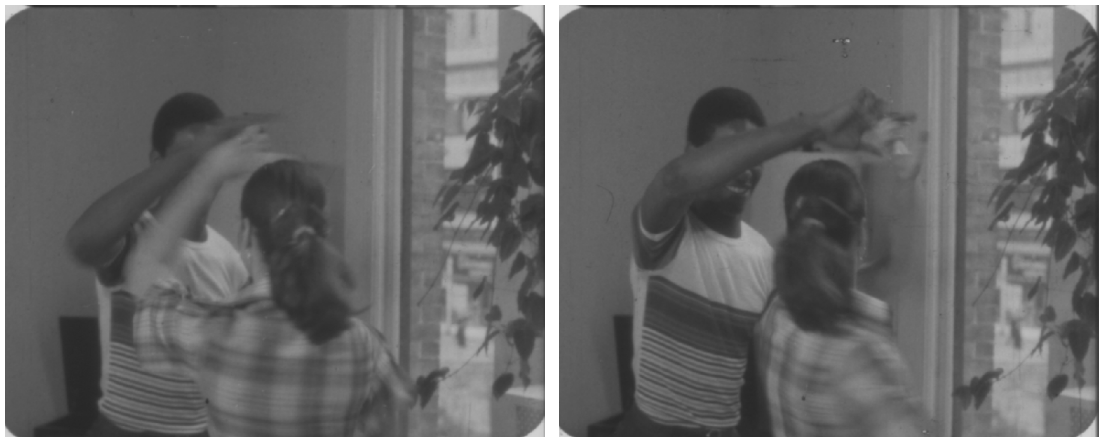


Figure 3.7: *The "dance" sequence is the first of the ground truth sequences. The movement of the dancer in the foreground is pathological. Its motion is self-occluding which also causes occlusion and uncovering of the background.*

sequences are introduced and a description of the experimental procedure is given.

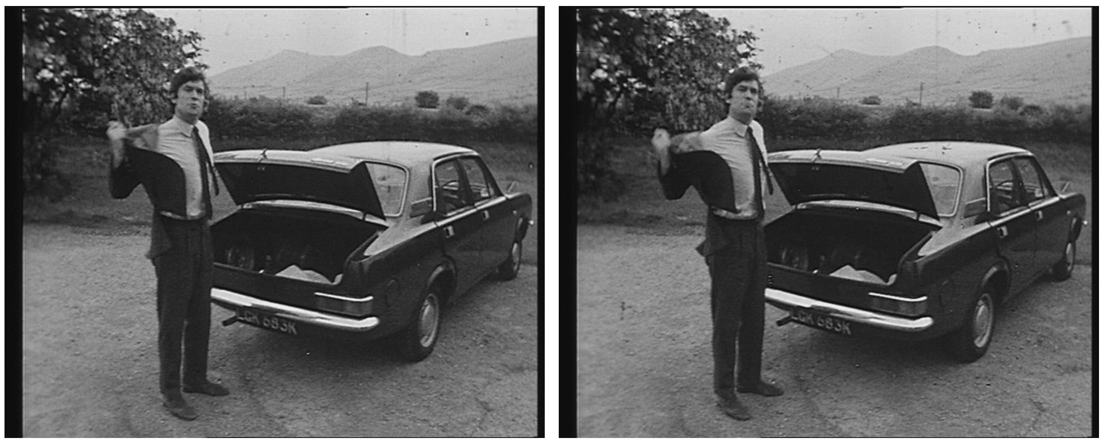
3.4.1 Ground Truth Acquisition

The first ground truth sequence is known as the "dance" sequence (See Fig. 3.7). The motion of the dancer in the foreground is pathological. As the dancer turns, her arms become occluded behind the rest of her body. There is also occlusion of the background.

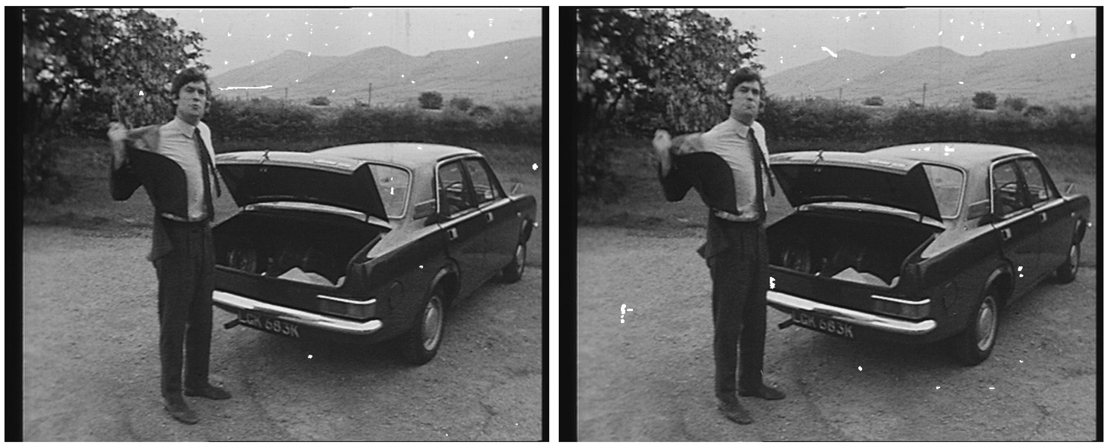
The blotch ground truth for this sequence is acquired using an Infra-Red IR scanner. The Infra-Red scans were provided by INA [44] as part of the EU Prestospace project [83]. IR scanners are used to produce a transparency map of each frame, where the intensity of each pixel in the map is proportional to the degree of transparency of the pixel. A useful property of dirt is that it is opaque to IR and as a result is associated with low intensities in the transparency map. By applying a threshold to the map, a binary ground truth for dirt can be obtained. In the "dance" sequence a ground truth is estimated for 125 frames of 720×576 pixels. The ground truth is obtained by applying a threshold of 150 to each 8-bit transparency map. Pixels with transparency intensities below this threshold are considered to be dirt.

Using IR scans is a good way to automatically generate a ground truth for dirt. However there are a number of drawbacks to using IR scans. IR scans detect dirt even when the dirt intensities are similar to the true image intensity. Such dirt can not be detected by image-based missing data detectors. IR scans will also detect non-impulsive dirt such as line scratches which are vertical lines of dirt in an image which can persist in the same location in a number of frames. As a result of these factors, the reported correct detection rate will be lower than the effective rate.

The second method used to generate a ground truth is to mark blotches in each frame by



(a) 2 frames from the jacket sequence



(b) Frames with blotches highlighted in white

Figure 3.8: *The second ground truth sequence is the "jacket" sequence. As the person in the sequence removes his jacket, the motion of the jacket is pathological and is an example of non-rigid object motion. This sequence has a higher blotch frequency than the "dance" sequence.*

hand. Acquiring a ground truth in this manner ensures that only perceivable dirt is marked, allowing for a more realistic ground truth. The obvious drawback of manually marking blotches, is the tediousness of the task given that the ground truth needs to be defined on a pixel scale. A ground truth for 22 frames of the "jacket" sequence (See Fig. 3.8) was generated in this manner. The resolution of this sequence is also 720×576. The motion of the jacket is an example of the erratic motion of non-rigid bodies. This sequence also has a higher frequency of blotches than the "dance" sequence.

3.4.2 Experimental Procedure

Motion Estimation - The gradient based motion estimator described by Kokaram in [54] is used to generate motion vectors for all the test sequences. The block size of vector field is 17×17 pixels with a 2 pixel horizontal and vertical overlap.

Standard parameter values of the proposed algorithm - The standard parameter configuration for the proposed algorithm is as follows

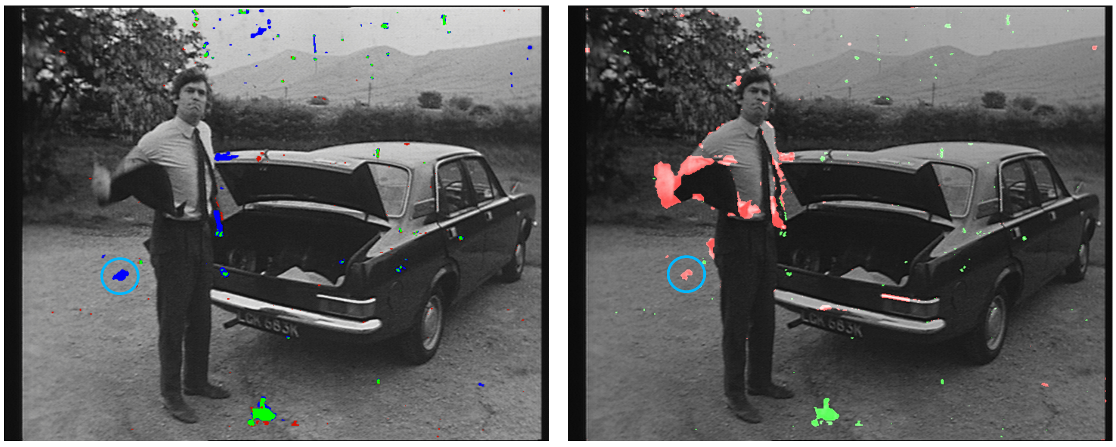
- The deterministic threshold δ_t is used as the control parameter for the ROC plots. The chosen values of δ_t are $\{5, 7, 9, 11, 13, 15, 17.5, 20, 25, 30, 35, 40\}$. These values are also used for the equivalent thresholds in the other missing data detectors used in the comparison (Section 3.4.4), ensuring that each detector is detected over an equivalent range.
- Four levels are used in the multiresolution framework. There is a maximum of twenty iterations per level.
- **Temporal Discontinuity Likelihood** - The value of α_2 and α_3 is directly related to the threshold δ_t on the DFDs used in the initialisation stage according to equation 3.9. The model error σ_e^2 is estimated before the first stage at each multiresolution level by finding the variance of the four DFDs when $s(\mathbf{x}) = 0$ (*i.e.* when no temporal discontinuity exists). The values of α_1 and α_4 are derived from the value of the outer threshold ratio α_r. The standard value of α_r is 0.5.
- **Divergence Likelihood** - The standard value of the divergence weight Λ_d is 1, the value of the divergence measure threshold e_t is 50 and the roll off rate r is 0.5.
- **Spatial Smoothness Prior** - The value of the smoothness weight Λ_l is 1 and the standard value of the PM bias K is 5.

ROC plots - An ROC curve is generated as follows

1. For a given set of parameters, the algorithm under test is applied to the ground truth sequences for every value of δ_t.
2. The missing data correct detection rate, r_c, and false alarm rate, r_f, are estimated for each frame (and for each δ_t), and are given by

$$\begin{aligned} r_c &= \frac{\sum_{\mathbf{x}} b_{gt}(\mathbf{x}) b_{det}(\mathbf{x})}{\sum_{x} b_{gt}(\mathbf{x})} \\ r_f &= \frac{\sum_{\mathbf{x}} (1 - b_{gt}(\mathbf{x})) b_{det}(\mathbf{x})}{\sum_{\mathbf{x}} (1 - b_{gt}(\mathbf{x}))} \end{aligned} \tag{3.21}$$

   where $b_{gt}(\mathbf{x})$ is the binary ground truth blotch mask for the frame and $b_{det}(\mathbf{x})$ is the binary output of the blotch detector under test.



(a) Validation image (b) PM/blotch detection

Figure 3.9: *This figures shows a validation image and a PM/blotch detection for the "jacket" sequence (correct detections - green, missed detections - blue, false alarms - red). The PM/blotch (red/green) images can be used to indicate whether or not a missed blotch detection occurs to the pixel being detected as PM (blue circle).*

3. The average correct detection and false alarm rates are then calculated for the ground truth sequences, giving a separate value for each δ_t.

4. The ROC plot for the test sequence is then generated from the average correct detection and false alarm rates.

PM rate plots - Plots of PM rate, r_p, against the DFD threshold δ_t are evaluated by finding the average relative frequency of PM in a sequence at each threshold. The PM rate is given by

$$r_p = \frac{\sum_{\mathbf{x}} |l(\mathbf{x}) == 2|}{\sum_{\mathbf{x}} 1} \tag{3.22}$$

where PM occurs when $l(\mathbf{x}) = 2$.

Visual Evaluation - Two types of image are used for the visual evaluation. They are validation images and PM/blotch detections (See Fig. 3.9). Validation images compare the estimated blotch masks to the ground truth and highlight correct detections (green), missed detections (blue) and false alarms (red). PM/blotch detections display the masks generated by the proposed algorithm and the Bornard algorithm with PM regions highlighted red and blotches highlighted green.

Implementation of the Algorithm

The tests outlined on the proposed algorithm in this chapter utilised a C++ implementation of the algorithm, incorporating the IPP libraries [45] where appropriate. The algorithm was tested on a PC with a 1.4 GHz Pentium M processor with 1 GB of RAM. The Morris, Bornard and Blotch MRF algorithms were also implemented in C++, while the SDIp and sROD algorithms were implemented in Matlab.

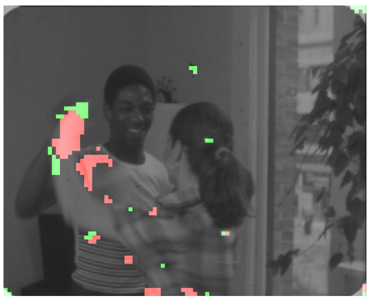
3.4.3 Algorithm Evaluation

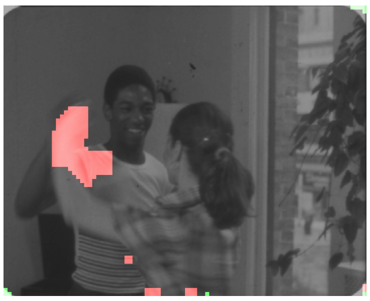
Fig. 3.10 gives an example of the evolution of the label field configuration for a frame of the “dance” sequence. Detection of missing data and PM at the coarser resolutions allows large PM and blotch features to be detected. As the resolution at which the optimisation occurs increases, the precision of the result increases allowing small features to be detected. In general convergence at the coarser levels is fast as spatial information propagates quickly through the field, but as the resolution increases the speed of convergence decreases. In the example shown in Fig. 3.10 the convergence occurs after six iterations at both Level 3 and Level 2. At the finer resolutions of Level 1 and Level 0 the maximum iteration limit of twenty iterations is reached. Computation time for this example is approximately ten seconds.

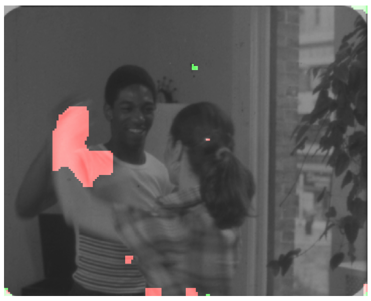
Applying the algorithm to all the frames of the test sequences, gives the ROC and PM rate curves shown in Fig. 3.11. Overall the maximum correct detection rates, r_c, for both sequences are quite low (approximately 25% for the “dance” sequence and 40% for the “jacket” sequence). However there is also a low false alarm rate, r_f, which does not increase above 0.1% (roughly 400 pixels in a 720×576 image) even at low threshold values. The PM rate plots for both test sequences (Fig. 3.11 (b) and (d)) show that the rate of PM detection increases exponentially as δ_t decreases.

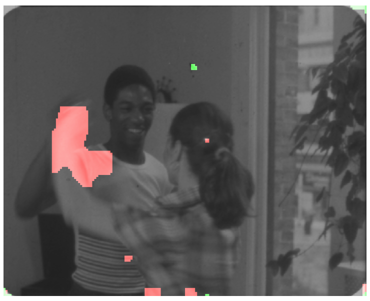
An unusual feature of the ROC plots for the test sequences is the apparent ceiling on the correct detection rates at lower values of the DFD threshold δ_t. The expected trend in an ROC plot of a blotch detector is to have low correct detection and false alarm rates at higher threshold values and for the rates to increase as the value of the threshold drops, tending to a notional correct detection and false alarm rate of 100% at a zero threshold when every pixel in the image is detected as a blotch. The lowest value of the threshold used therefore gives the highest false alarm and correct detection rate. However the ROC curves show that this pattern does not apply to the proposed algorithm. While the expected pattern is followed at high values of the threshold δ_t, at lower values of the threshold a value is reached, below which the correct detection rate no longer increases and continues to drop off. Although the rate at which temporal discontinuities are detected in the five frame window continues to increase, an increasing number of pixels are classified as PM (See Fig. 3.12) since the likelihood of temporal discontinuities at the same location between multiple frames greatly increases. This is confirmed by the exponentially increasing PM rate at lower threshold values. Consequently, less pixels are

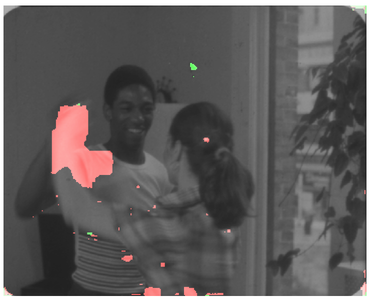
(a) A frame from the "dance" sequence

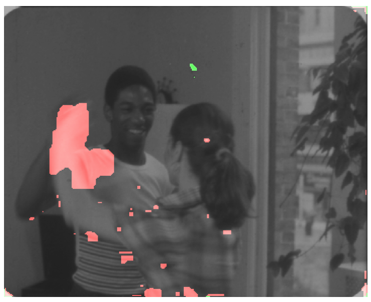

(b) Configuration after Deterministic Initialisation

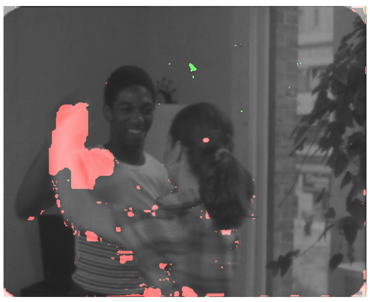

(c) Final Field at level 3 (6 iterations in total)


(d) Field after 1st iteration at Level 2 (7 iterations in total)


(e) Final Field at Level 2 (12 iterations)


(f) Field after 1st iteration at Level 1 (13 iterations)


(g) Final Field at Level 1 (32 iterations)


(h) Field after 1st iteration at full resolution (33 iterations)

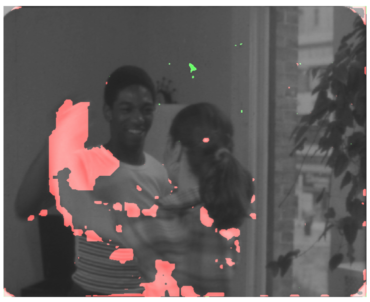
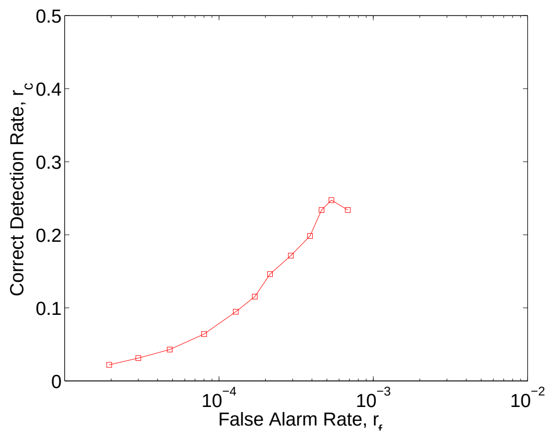
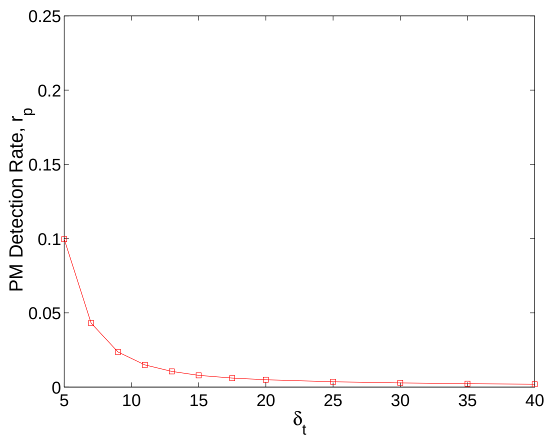

(i) Field after final iteration (52 iterations)

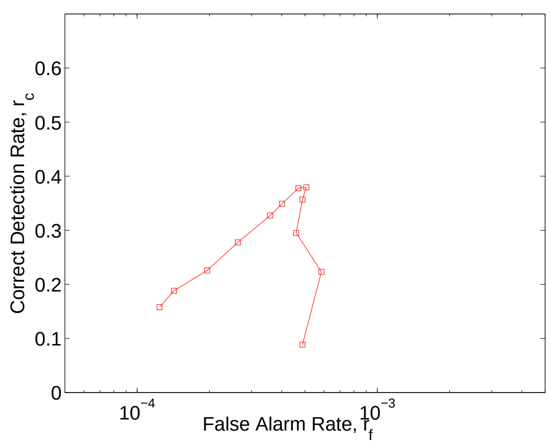
Figure 3.10: *This figure shows intermediate values of the label field (PM - Red, Blotches - Green) at various iterations of the ICM optimisation process for a frame from the "dance" sequence. The standard parameter set is used (Section 3.4.2) and the value of δ_t is 7. Level 0 corresponds to the full resolution; the resolution at level 1 is* 360 × 288 *and so on. At the coarsest resolution (Level 3) the resolution is* 90 × 72 *pixels.*



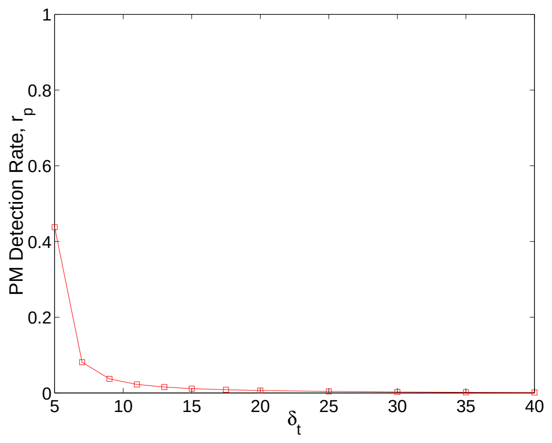
(a) ROC for the “dance” sequence



(b) PM rate curve for the “dance” sequence



(c) ROC for the “jacket” sequence



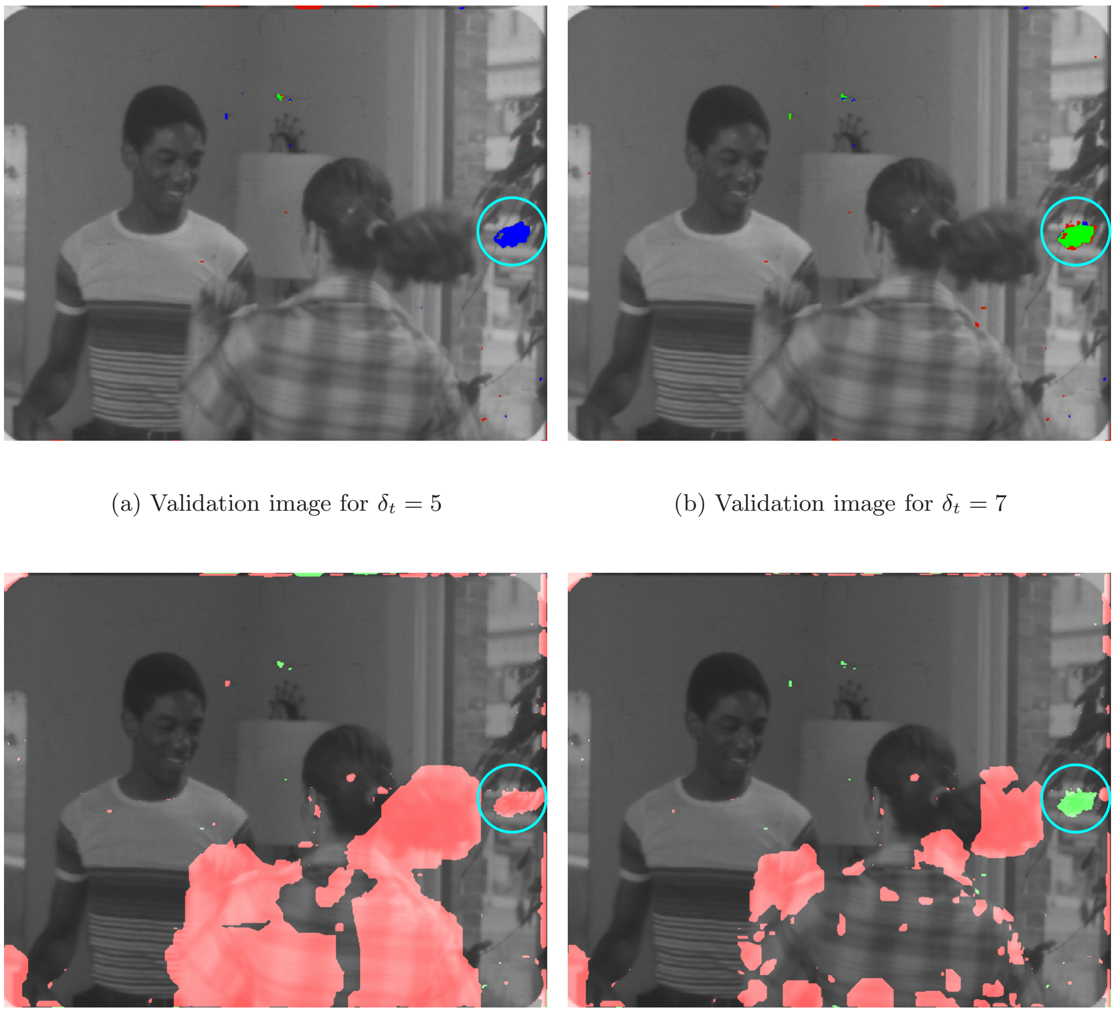
(d) PM rate curve for the “jacket” sequence

Figure 3.11: *This figure shows ROC and PM rate plots (r_p v δ_t) for both test sequences obtained from the proposed algorithm using the standard set of parameters.*

detected as missing data and the correct detection rate decreases. For this detector, the notional false alarm rate and correct detection rate at the zero threshold level is 0% as all pixels will be detected as Pathological Motion. The following sections describe how the Divergence Likelihood, the PM Bias and the Outer Threshold Ratio influence the performance of the algorithm.

The Divergence Likelihood

An ROC curve and PM rate curve is plotted for four different values of Λ_d (Fig. 3.13). The values chosen are $\Lambda_d = \{0,\ 1,\ 2,\ 5\}$. The other parameter values are the standard parameter



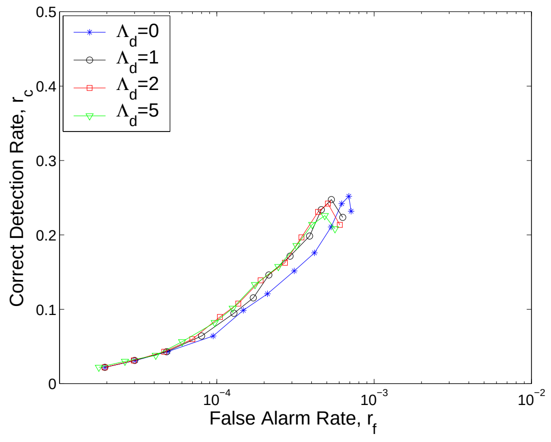
(a) Validation image for $\delta_t = 5$ (b) Validation image for $\delta_t = 7$

(c) PM/blotch detection for $\delta_t = 5$ (d) PM/blotch detection for $\delta_t = 7$

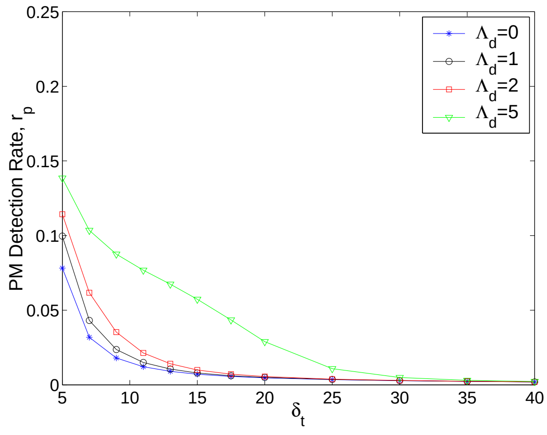
Figure 3.12: *This figure illustrates how lowering the DFD threshold δ_t results in a lower correct detection rate. The blue circle highlights a large blotch which is not detected at the lower threshold ((a) and (b) - correct detections in green, missed detections in blue). At the lower threshold this blotch is classified as PM ((c) and (d) - red for PM, green for blotches).*

values defined in Section 3.4.2. The value 0 is equivalent to the divergence likelihood being excluded from the framework. As the value of the divergence weight increases the influence of the divergence likelihood in the framework increases.

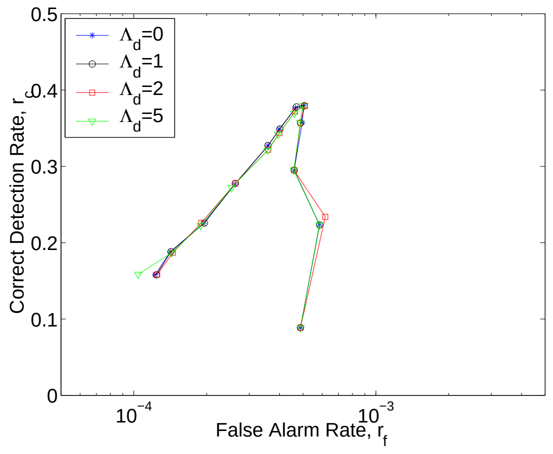
The ROC and PM rate curves highlight the mixed influence of the Divergence Likelihood (Fig. 3.13) on the performance of the algorithm for the two test sequences. The introduction of the divergence likelihood (when $\Lambda_d = 1$) improves the performance of the algorithm on the the “dance” sequence (Fig. 3.13 (a), Fig. 3.14). There is a notable reduction in the false alarm



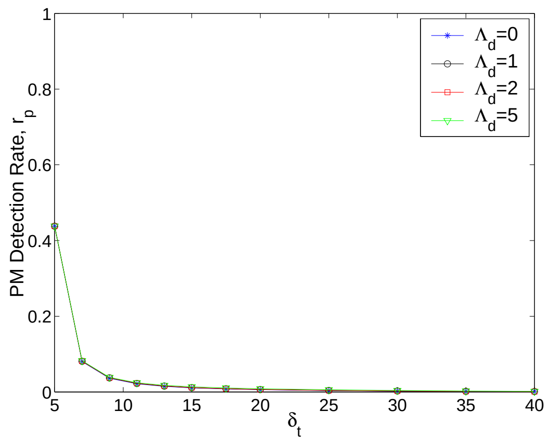
(a) ROC for the "dance" sequence



(b) PM rate curve for the "dance" sequence



(c) ROC for the "jacket" sequence

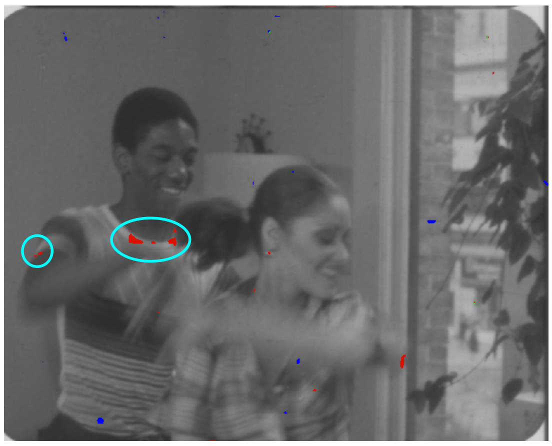


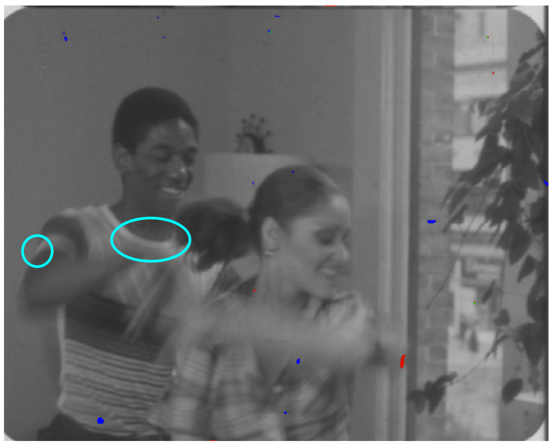
(d) PM rate curve for the "jacket" sequence

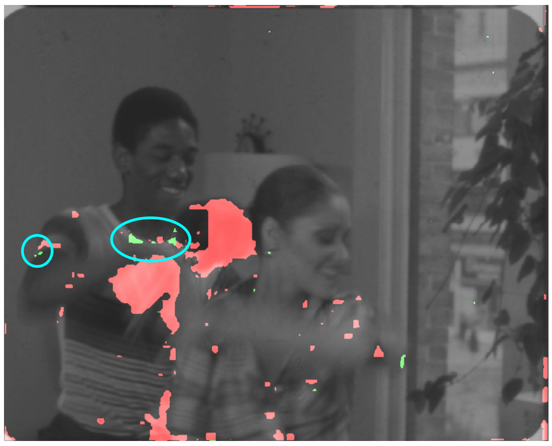
Figure 3.13: *This figure shows the ROC and PM rate curves (r_p v δ_t) for the four chosen values of the divergence likelihood weight Λ_d (i.e.* 0, 1, 2 *and* 5*).*

rate (r_f) for a negligible drop in the correct detection rate (r_c). Increasing the weight of the Divergence Likelihood further does not reduce the false alarm rate further. However, there is a drop-off in the maximum correct detection rate. Increasing the weight of the likelihood also increases the rate of PM detection (Fig. 3.13 (b)).

On the other hand, the weight of the divergence likelihood does not significantly alter the performance of the algorithm on the "jacket" sequence (Fig. 3.13 (c)). The PM rate is not significantly affected either (Fig. 3.13 (d)). This is further highlighted in Fig. 3.15 where the introduction of the likelihood does not change the blotch detection rate and only slightly


(a) Validation image without Divergence Likelihood ($\Lambda_d = 0$) (red - false alarms, blue - missed detections, green - correct detections)


(b) Validation image with Divergence Likelihood ($\Lambda_d = 1$)


(c) PM/blotch detection without Divergence Likelihood ($\Lambda_d = 0$) (red - detected pm, green - detected blotches)

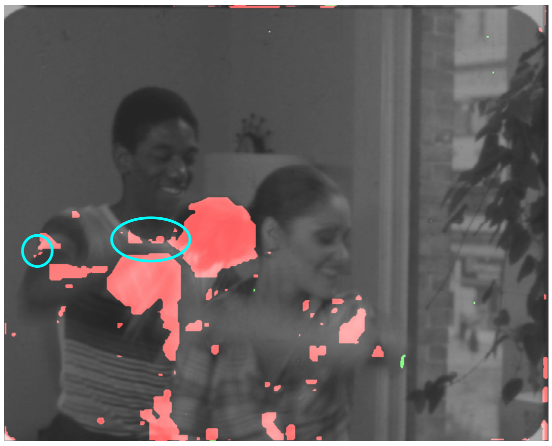
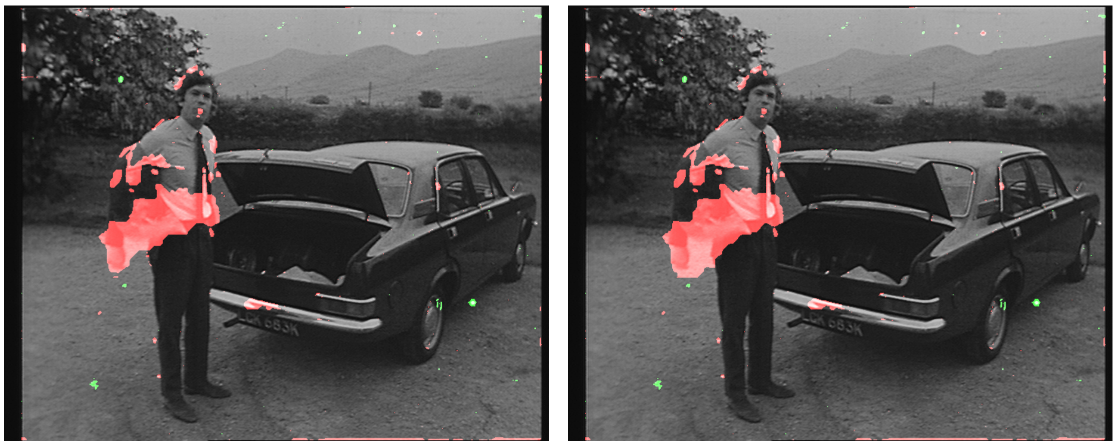

(d) PM/blotch detection with Divergence Likelihood ($\Lambda_d = 1$)

Figure 3.14: *This figure highlights a reduction in the number of incorrectly detected blotches caused by the introduction of the Divergence Likelihood. The blue circles highlight the false alarms that are removed when $\Lambda_d = 1$. Introduction of the likelihood also results in more Pathological Motion being detected ((c) and (d))*



(a) PM/blotch detection without Divergence Likelihood ($\Lambda_d = 0$)

(b) PM/blotch detection without Divergence Likelihood ($\Lambda_d = 1$)

Figure 3.15: *This figure shows detected PM (red) and missing data (green) on a frame of the "jacket" sequence.*

increases the amount of PM that is detected.

The PM bias

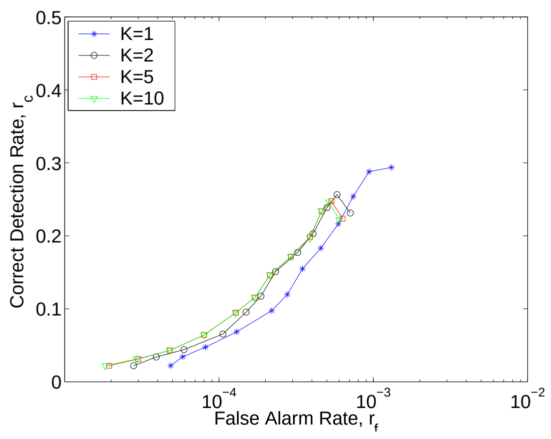
ROC and PM rate curves are plotted for four values of the missing data bias K (Fig. 3.16). The values chosen are $K = \{1, 2, 5, 10\}$ where $K = 1$ corresponds to the case where no bias exists and where the bias towards detecting PM increases with K.

The results of the quantitative evaluation show that introducing the PM bias generally aids the performance of the algorithm. Although there is a small drop off in the correct detection rates, especially at low values of the threshold δ_t, the false alarm rate is reduced as the strength of the bias is increased. The value of K has no significant difference on the rate of detected PM in either test sequence.

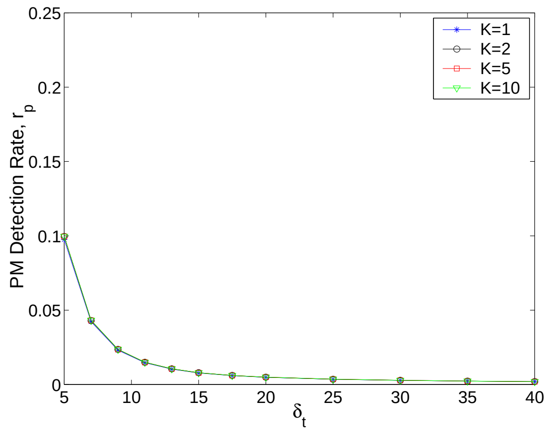
The effect of the bias on the results can be seen more clearly in Fig. 3.17. The large false alarm highlighted in the blue circle is removed by the bias, as the label of the neighbouring PM sites is interpolated into the false alarm region. However, the bias also causes blotches to be detected as PM. In this example, it is preferable to use the higher value of K, as the visual damaged caused by interpolating the large false alarm is significant (Fig. 3.18).

The Outer Threshold Ratio

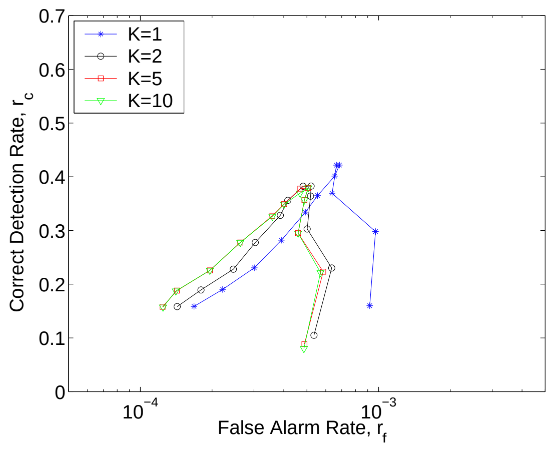
Once more ROC and PM rate curves are generated for four different values of α_r. The values are $\alpha_r = \{1, \frac{1}{\sqrt{2}}, \frac{1}{2}, \frac{1}{2\sqrt{2}}\}$. When $\alpha_r = 1$, the values of α_1 and α_4 are the same as α_2 and α_3.



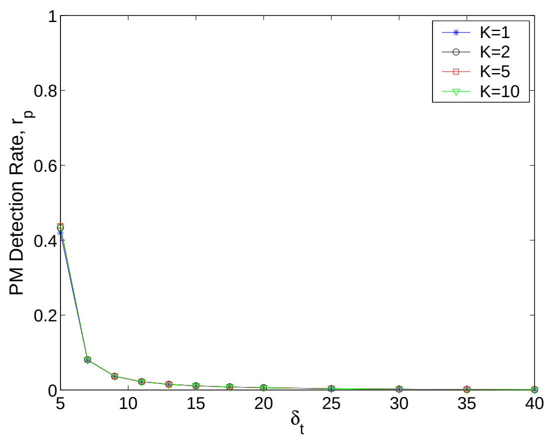
(a) ROC for the “dance” sequence



(b) PM rate curve for the “dance” sequence



(c) ROC for the “jacket” sequence



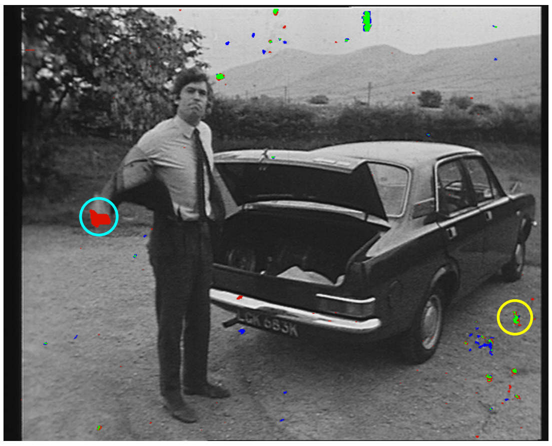
(d) PM rate curve for the “jacket” sequence

Figure 3.16: *This figure shows the quantitive evaluation for the four chosen values of K.*

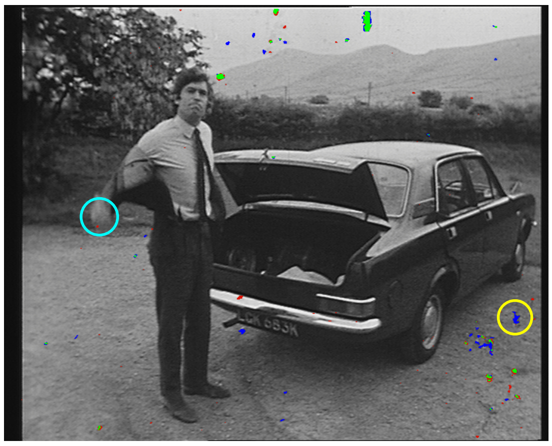
As α_r decreases the effective threshold on the outer DFDs decreases and so the likelihood of the detecting PM increases.

Reducing the threshold on the “outer” DFDs is intended to bias detection toward Pathological Motion over Missing Data and this is evident from the quantitive evaluation of the test sequences (Fig. 3.19). Decreasing the outer threshold ratio decreases the false alarm rate and the correct detection rate and increases the rate of detected PM. Reduction of the threshold δ_t at which the maximum correct detection rate occurs is another consequence of using a outer threshold ratio smaller than 1 (See Table 3.3).

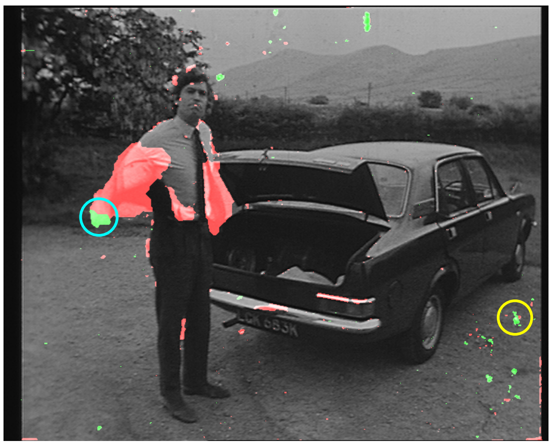
The example shown in Fig. 3.20 is indicative of the effect of using an outer threshold ratio less than one. When the ratio is one (*i.e.* the same α threshold is used for all DFDs), there



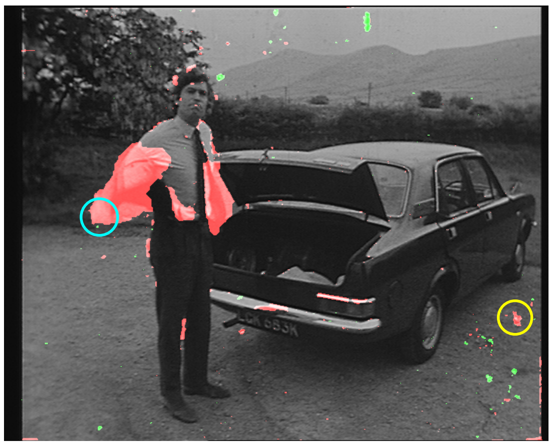
(a) Validation image for $K = 1$ (red - false alarms, green - correct detections, blue - missed detections)



(b) Validation image for $K = 5$

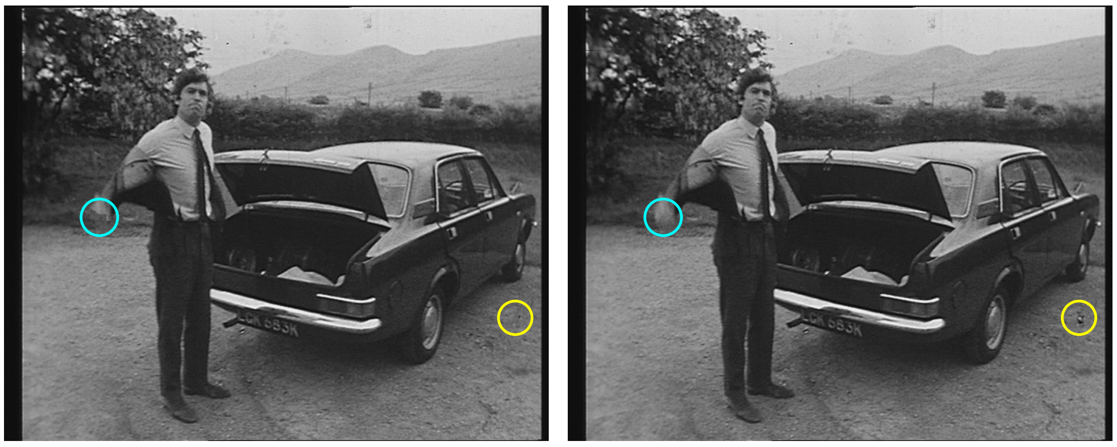


(c) PM/blotch (red/green) detection for $K = 1$



(d) PM/blotch detection for $K = 5$

Figure 3.17: *This figure shows validation images and PM/blotch detections for a frame of the jacket sequence for $K = 1$ (no bias) and $K = 5$ (suggested bias). The blue circles indicate an area where a false alarm is removed by the introduction of the bias. The yellow circle highlights where the bias has caused blotches to be detected as PM.*



(a) Restored Image for $K = 1$ blotch mask (b) Restored Image for $K = 5$ blotch mask

Figure 3.18: *The false alarm highlighted in the blue circle causes the hand to be removed in the restored image(left). The bias prevents this from happening in the right image, although the circle highlights a blotch that has not been removed.*

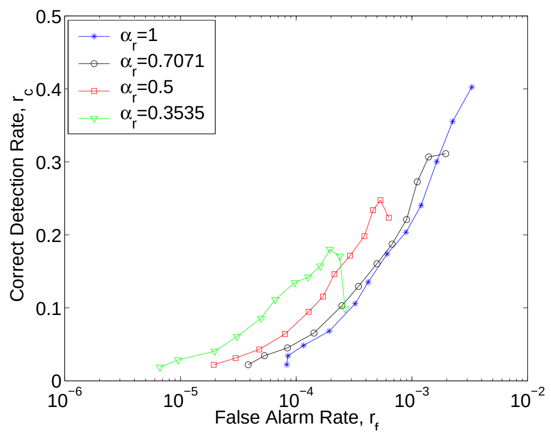
Outer Threshold Ratio, α_r	"Dance" Sequence	"Jacket" Sequence
1	5	7
$\frac{1}{\sqrt{2}}$	5	11
$\frac{1}{2}$	7	13
$\frac{1}{2\sqrt{2}}$	9	20

Table 3.3: *This table outlines the value of δ_t at which the maximum correct detection rate occurs in the ROC curves in Fig. 3.19. As the value of α_r decreases the value of δ_t increases.*

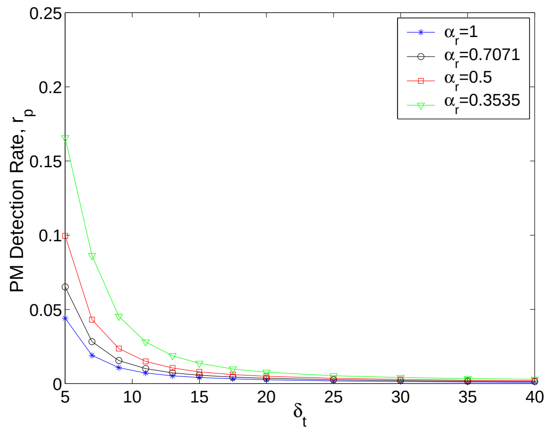
are more false alarms present. Using a smaller ratio causes most of the falsely detected blotches to be detected as PM, resulting in a reduced false alarm rate. The reduction in the blotch detection rate, r_c, can also result in an increased frequency of missed detectections (See Fig. 3.21). However, the ROC curves in Fig. 3.19 demonstrate that the reduction in the false alarm rate is more significant than the reduction in the correct detection rate.

3.4.4 Comparison with other Missing Data Detectors

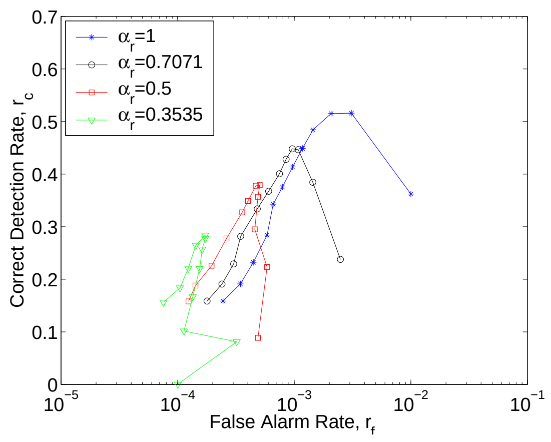
The proposed algorithm is compared with five existing missing data detectors. These include four conventional algorithms and the Bornard Missing Data Detection algorithm. The four other algorithms are the SDIp algorithm [54, 61], the sROD detector [5, 104], the Morris algorithm [61, 73] and the Blotch MRF model [50] (see chapter 2, section 2.3.3).



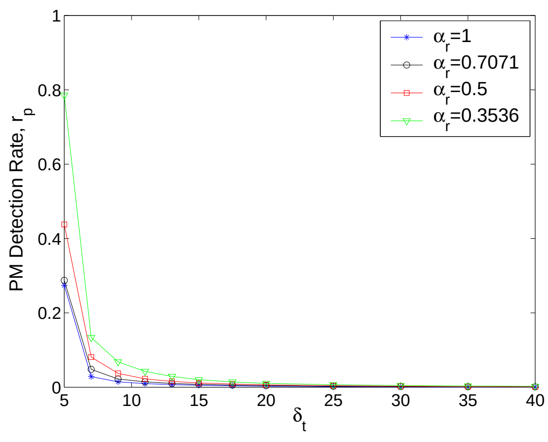
(a) ROC for the "dance" sequence



(b) PM rate curve for the "dance" sequence



(c) ROC for the "jacket" sequence

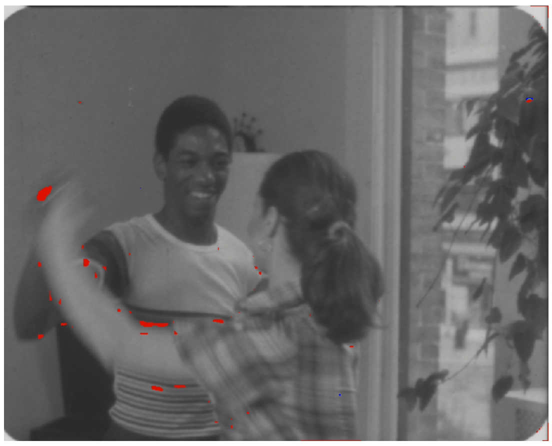


(d) PM rate curve for the "jacket" sequence

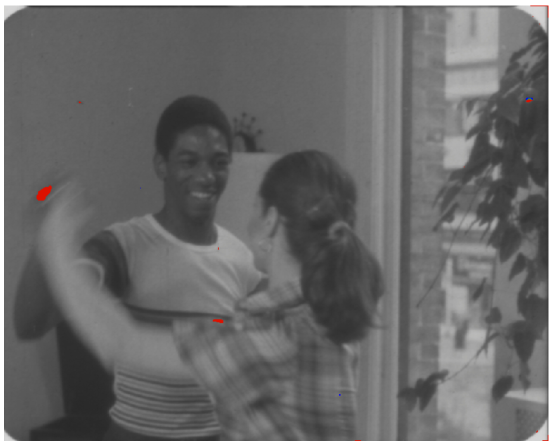
Figure 3.19: *This figure shows the quantitive evaluation for the four chosen values of the outer threshold ration* α_r.

Detector Configuration

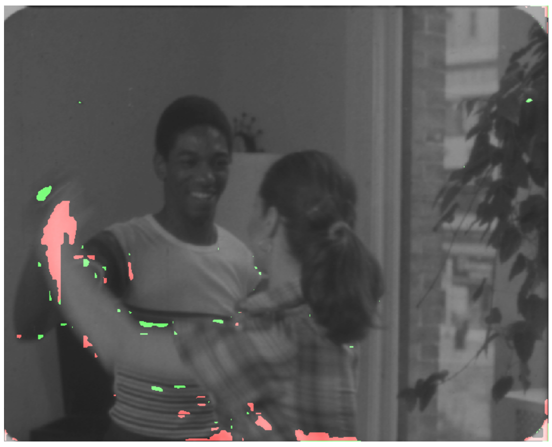
Both the SDIp and sROD algorithms have one associated parameter δ_t and are implemented directly from the description of the algorithms (see chapter 2, section 2.3). The four probabilistic algorithms are implemented with as similar parameter values as possible to ensure the fairness of the comparison. A multiresolution framework is employed for each algorithm, with a maximum of twenty iterations per level. An eight-pixel first order spatial neighbourhood is used for either the blotch field (blotch MRF algorithm), the temporal discontinuity field (TDFs) (morris/bornard algorithms) or the blotch/PM field (the proposed algorithm). The Bornard al-



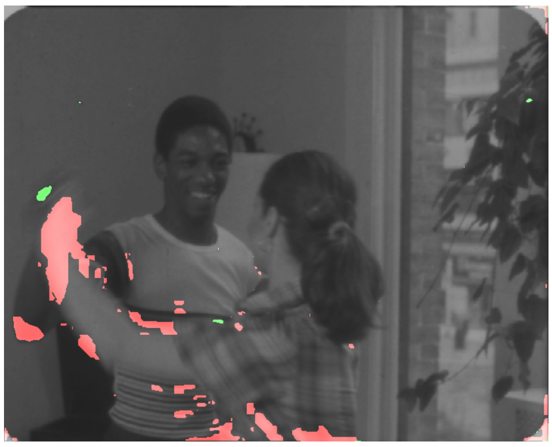
(a) Validation image for $\alpha_r = 1$ (red - false alarms, green - correct detections, blue - missed detections)



(b) Validation image for $\alpha_r = \frac{1}{2}$



(c) PM/blotch (red/green) detection for $\alpha_r = 1$

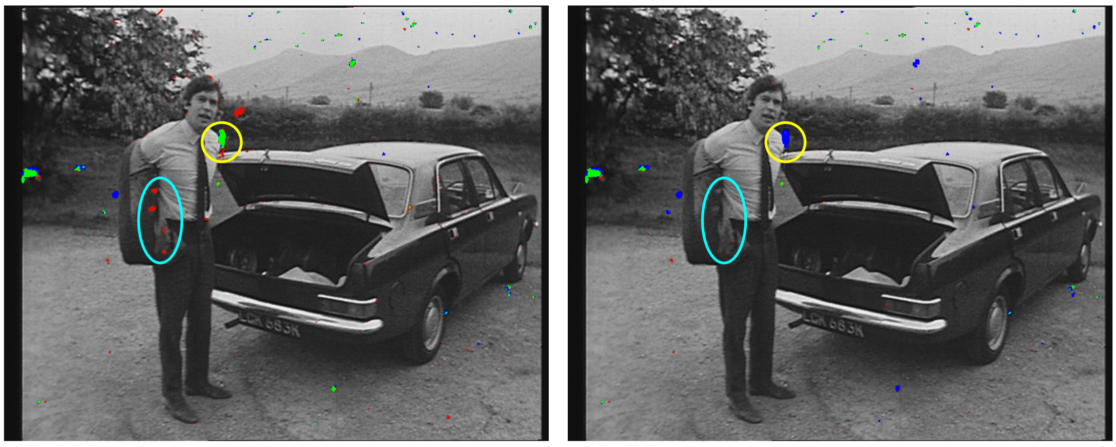


(d) PM/blotch detection for $\alpha_r = \frac{1}{2}$

Figure 3.20: *This figure visually demonstrates the effect of reducing the outer threshold ratio on a frame in the "dance" sequence.*

gorithm also employs a two-pixel temporal neighbourhood linking the TDFs. Like the proposed algorithm, in the Blotch MRF algorithm a smoothness prior is only enforced on the blotch field and not on the state field (*i.e.* $\Lambda_b = 1$, $\Lambda_s = 0$).

The value of the spatial smoothness weight in the Morris and Bornard algorithms is 1, and the temporal prior weight of the Bornard algorithm, β_3, is -0.1. As the value of the parameter has a negative value, the prior is effectively a smoothness prior, rather than a dissimilarity prior, which encourages temporal discontinuities to appear at the same spatial location in consecutive tdfs.



(a) Validation image for $\alpha_r = 1$ (red - false alarms, green - correct detections, blue - missed detections)

(b) Validation image for $\alpha_r = \frac{1}{2}$

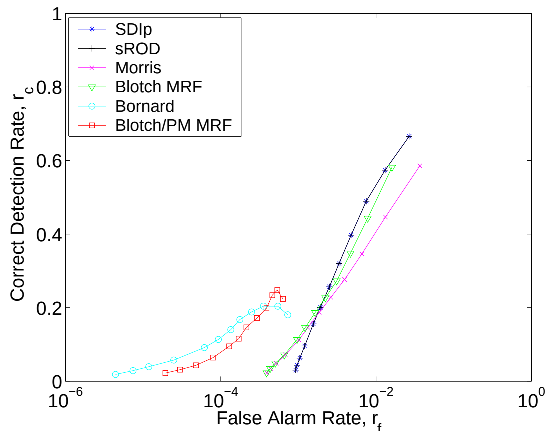
Figure 3.21: *This figure shows validation images of a frame from the "jacket" sequence for two outer threshold ratio values. The blue ellipse indicates where false alarms that are removed by using a lower ratio, while the yellow circle highlights a missed blotch detection at the lower ration value.*

This reflects the setup recommended by Bornard in his thesis ([8] Section 5.7.1.2), where the temporal prior weight is roughly one tenth the size of the spatial weight, with a negative value. The search radius in the interpretation stage is 20 pixels with a temporal discontinuity threshold of 1. Although less conservative than the recommended setup (a radius of 38 pixels), PM will be flagged if a temporal discontinuity exists in one of $2 \times 20^2\pi$ pixels which is approximately two and a half thousand pixels. For the proposed algorithm, the standard set of parameters described in Section 3.4.2 is applied.

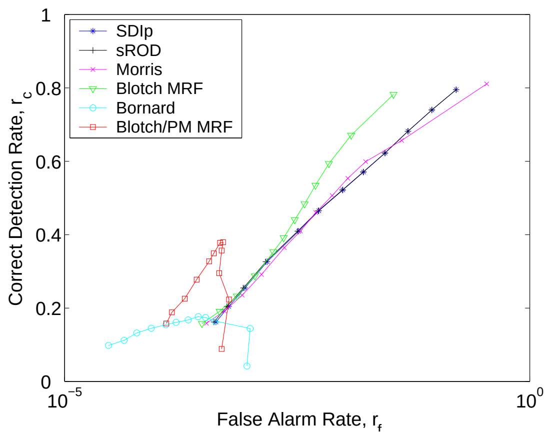
Comparison with the Conventional Detectors

ROC curves are plotted for each detector, with the same set of values of δ_t as before ($\delta_t = \{5, 7, 9, 11, 13, 15, 17.5, 20, 25, 30, 35, 40\}$). PM rate, r_p, plots are also generated for the proposed algorithm.

Experimental comparison of the missing data detectors shows that both the proposed algorithm and the Bornard algorithm achieve a significantly reduced false alarm rate when compared to the four standard missing data detectors, although the correct detection rates are also reduced. At higher values of the DFD threshold δ_t (*i.e.* before the max correct detection rate is reached), the false alarm rate of the proposed algorithm is 75% to 90% less for the "dance" sequence and 50% to 80% less for the "jacket" sequence. On the other hand, both the Bornard



(a) ROC for the "dance" sequence

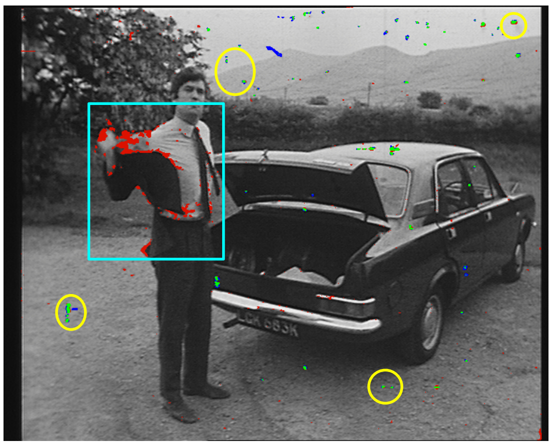


(b) ROC for the "jacket" sequence

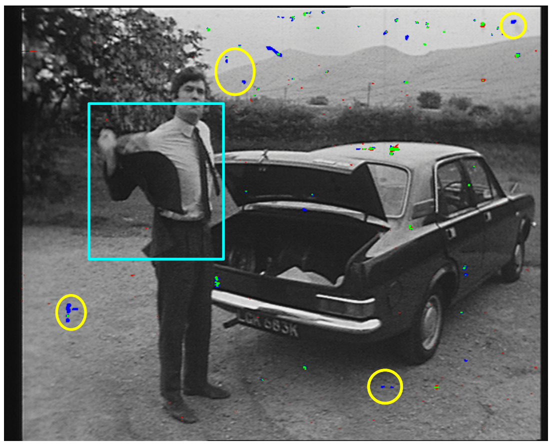
Figure 3.22: *This figure shows the ROC curves for the 6 detectors under test for each test sequence. The standard parameter set is used for the proposed algorithm. The set up of each detector is described in Section 3.4.2.*

and proposed algorithms have an upper limit on the rate of correct blotch detections. The average maximum correct detection rate of the proposed algorithm is two to three times less than the correct detection rates at the lowest threshold tested ($\delta_t = 5$), although the false alarm rate is 30 to 50 times lower.

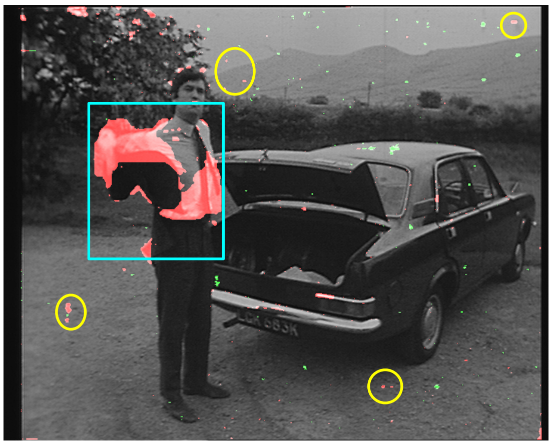
Figures 3.23 and 3.24 visually highlight the difference in performance between the proposed algorithm and the Blotch MRF algorithm, the standard blotch detector most closely related to the proposed algorithm. In the highlighted frames the proposed algorithm reduces the number of false alarms in the Pathological Motion regions (*i.e.* the jacket and the female dancer), preventing damage to image data (Fig. 3.25). The drop in blotch detections can mainly be attributed to two factors. Firstly blotches which are located near objects undergoing PM are likely to be missed (Fig. 3.26). This is an intended consequence of the algorithm design, which is encouraged to maintain image quality rather than remove the maximum number of blotches. The second major cause of missed detections occurs because of a high frequency of blotches over a number of frames. A fundamental assumption made by missing data detectors is that blotches are rarely located at the same spatial location in neighbouring frames. However, if there is a high frequency of blotches in a sequence, then the probability of blotches occurring at the same location in neighbouring frames increases. In such situations, a blotch is likely to be incorrectly classified as Pathological Motion (Fig. 3.27).



(a) Validation image for the Blotch MRF Detector (green - correct detections, red - false alarms, blue - missing data)

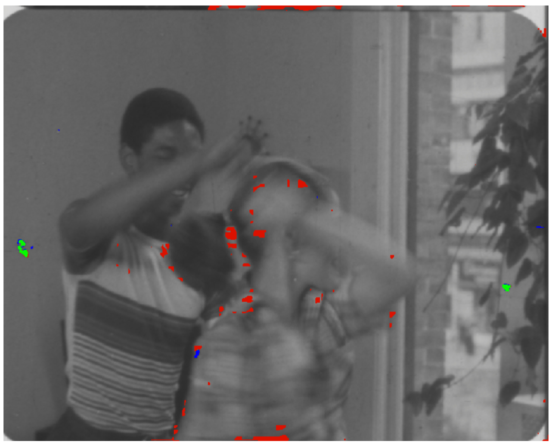


(b) Validation image for the proposed algorithm

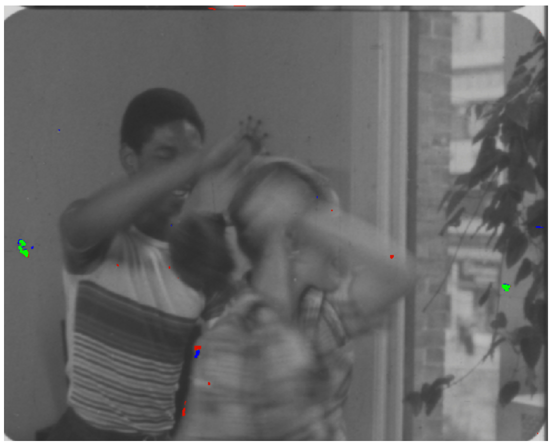


(c) PM/blotch detection for the proposed algorithm (red - pm, green - blotches)

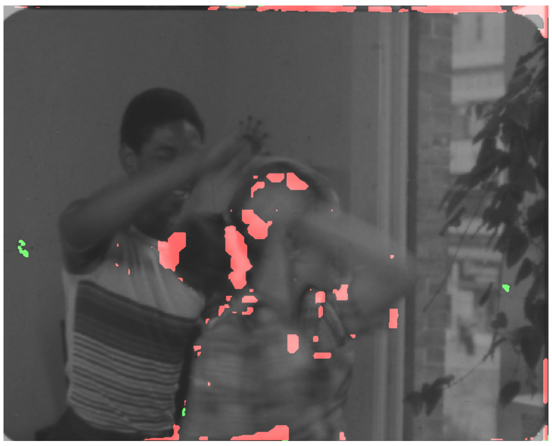
Figure 3.23: *This figure consists of validation images for the Blotch MRF algorithm and the proposed algorithm for a frame from the "dance" sequence. The value of the DFD threshold, δ_t, is 15. Also shown is the PM/blotch detection for the proposed algorithm. The blue square highlights the region of the image where the proposed algorithm eliminates false alarms caused by Pathological Motion. The yellow circles highlight missed blotch detections in the proposed algorithm, which have been wrongly detected as PM.*



(a) Validation image for the Blotch MRF Detector (green - correct detections, red - false alarms, blue - missing data)

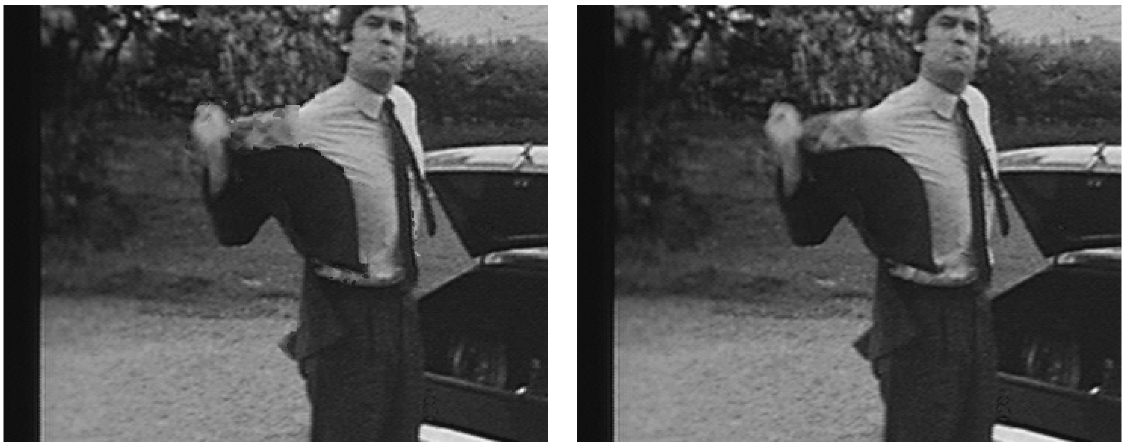


(b) Validation image for the proposed algorithm



(c) PM/blotch detection for the proposed algorithm (red - pm, green - blotches)

Figure 3.24: *This figure consists of validation images for the Blotch MRF algorithm and the proposed algorithm for a frame from the "jacket" sequence, as well as the corresponding PM/blotch detection for the proposed algorithm. The value of the DFD threshold, δ_t, is 15. Using the proposed algorithm reduces the number of false alarms in the region of the head of the female dancer.*



(a) Restored image using the Blotch MRF Detector

(b) Restored image using the proposed algorithm

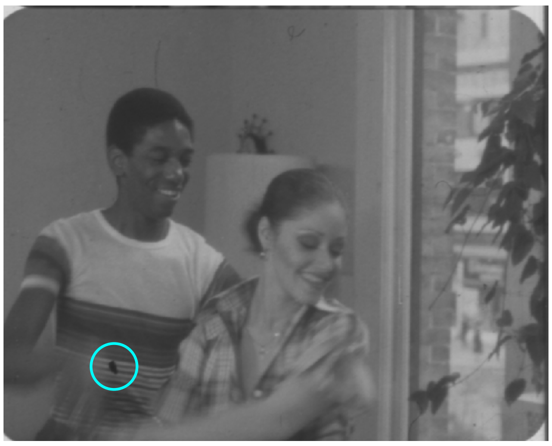
Figure 3.25: *This figure highlights a region of the frame shown in Fig. 3.23 which is restored using the Blotch MRF detector and the proposed algorithm. If the Blotch MRF detector is used to restore the frame significant damage occurs in the region of the jacket. This damage is prevented if the proposed algorithm is used.*

Comparison with the Bornard Algorithm

The missing data ROC curves in Fig. 3.22 also highlight the variable performance of the proposed algorithm when compared with the Bornard algorithm. The Bornard algorithm generally performs better on the "dance" sequence. At high thresholds the false alarm rate, r_f, of the Bornard algorithm is lower for similar correct detection rates, r_c. However, at lower threshold values the difference in the false alarm rates decreases and the proposed algorithm achieves a higher maximum correct detection rate. On the other hand, the proposed algorithm is much more effective on the "jacket" sequence and achieves a much higher blotch detection rate. The maximum correct detection rate is approximately 40%, roughly twice the maximum correct detection rate of the Bornard algorithm.

The poor correct detection rates of the Bornard algorithm on the "jacket" is clearly visible in the frame from the "jacket" sequence highlighted in Fig. 3.28. Many of the blotches in the frame are detected as PM by the Bornard algorithm, caused by the high frequency of visible blotches in the sequence. The proposed algorithm is less prone to missed detections in such circumstances.

It is reasonable to conclude that the Bornard detector is more conservative than the proposed algorithm. The interpretation stage of the Bornard is more conservative than the cumulative effect of the divergence likelihood, the PM bias and the outer threshold ratio. The interpretation


(a) A frame from the “dance” sequence

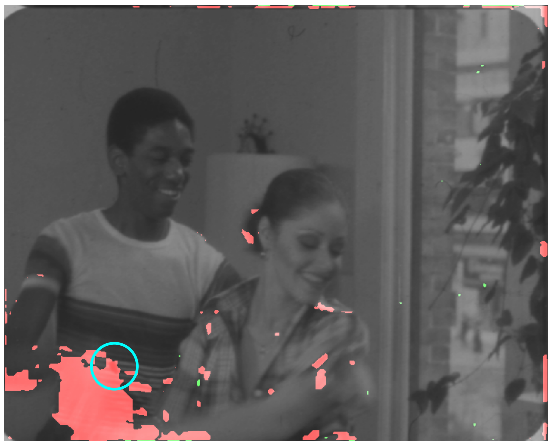

(b) A PM/blotch (red/green) detection for the frame ($\delta_t = 11$)

Figure 3.26: *This blue circle in the highlights a blotch located near a region of motion blur. Due to its location the blotch is detected as PM by the proposed algorithm.*

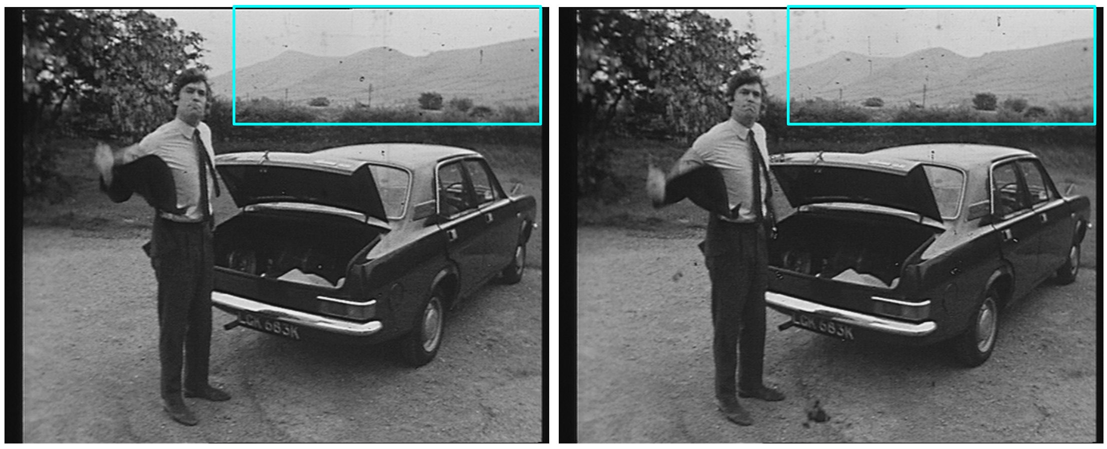
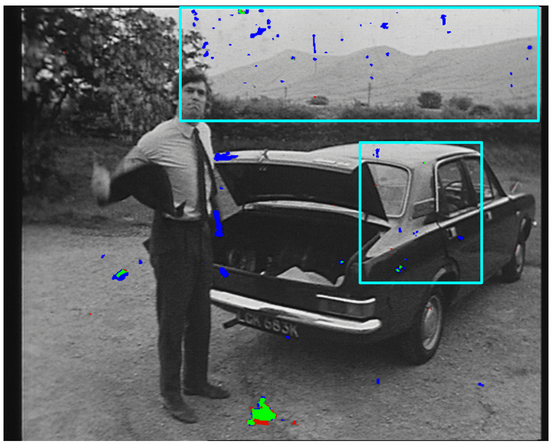
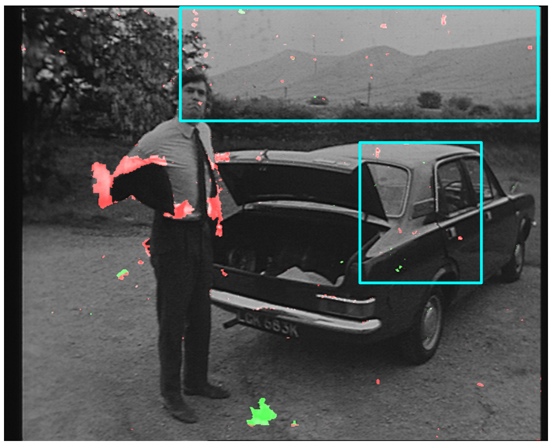


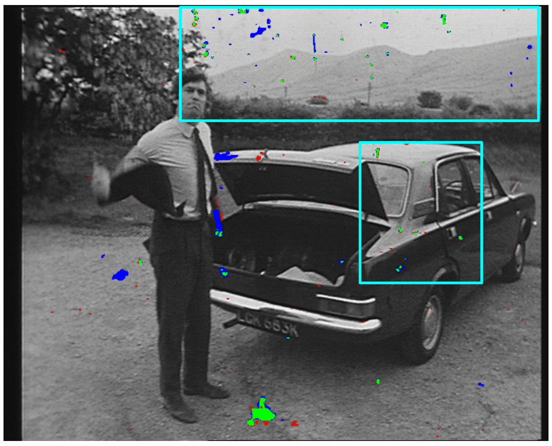
Figure 3.27: *This figure shows two consecutive frames from the “jacket” sequence. The blue rectangle highlights a region with a high frequency of blotches. The increased frequency of blotches increases the probability of blotches occurring at the same location, causing such blotches to be detected as PM.*



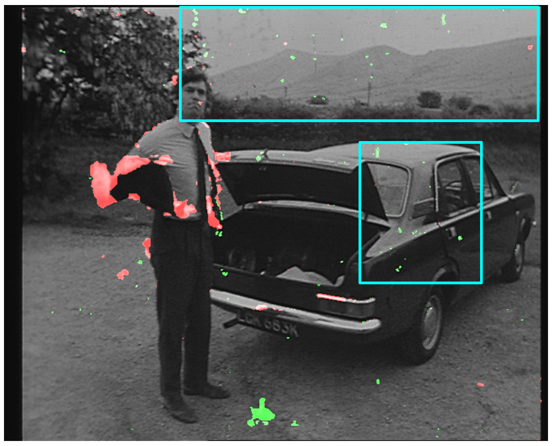
(a) Validation image for the Bornard detector (green - correct detections, red - false alarms, blue - missing data)



(b) PM/blotch detection for the Bornard detector (red - pm, green - blotches)



(c) Validation image for the proposed algorithm



(d) PM/blotch detection for the proposed algorithm

Figure 3.28: *This figures illustrates the superior blotch detection rate of the proposed algorithm. The highlighted regions of the frame indicate blotches missed by the Bornard algorithm (image (a)) that are detected by the proposed algorithm (c). The blotches missed by the Bornard algorithm are detected as PM (b). The value of δ_t used to generate the images in this figure is* 15.

stage acts as a "maximum caution" classifier, rejecting any candidate blotch site if a single temporal discontinuity exists within a radius (20 pixels) of the candidate in either of the outer temporal discontinuity fields. The ultra cautious nature of the interpretation allows the Bornard detector to achieve lower false alarm rates. However, it also results in more missed detections at lower threshold values or when there is a high frequency of blotches in a sequence.

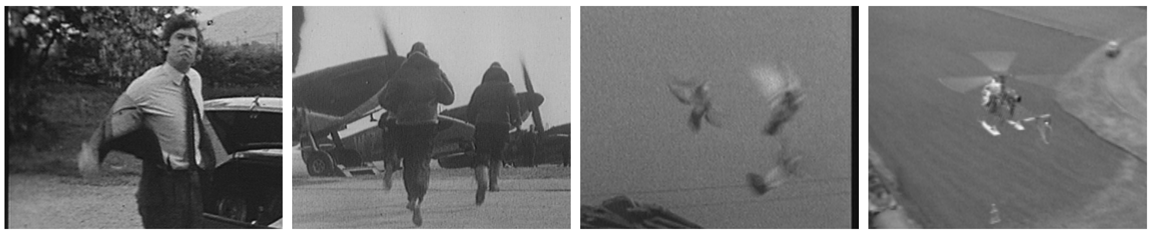
On the other hand, the proposed algorithm tries to make an informed decision and as such is less indiscriminate than the Bornard algorithm. For a site to be detected as PM rather than a blotch, the proposed algorithm requires either a temporal discontinuity to exist in at least one outer temporal discontinuity field at the same location or spatial propagation from neighbouring PM sites by means of the smoothness prior. The divergence likelihood, the PM bias and the outer threshold are all intended to make the proposed algorithm more likely to detect PM and have been shown, for the most part, to reduce the false alarm rate , r_f, without reducing the correct detection rate, r_c. The divergence likelihood increases the probability of sites with high motion field divergences being detected as PM, the PM bias encourages the spatial propagation of PM states and the outer threshold ratio increases the likelihood of temporal discontinuities being detected in the outer fields.

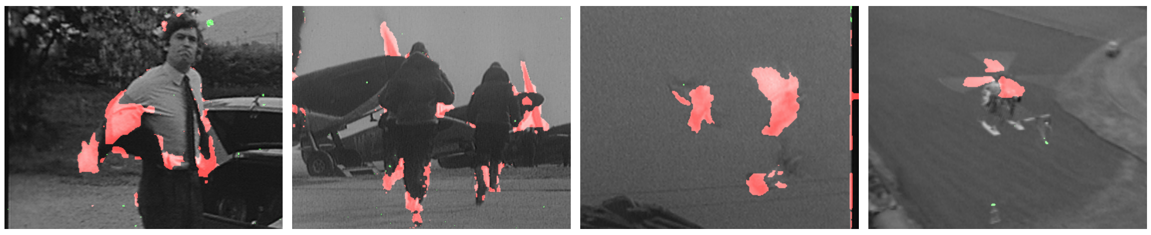
3.4.5 Blotch Restoration using the Proposed Algorithm

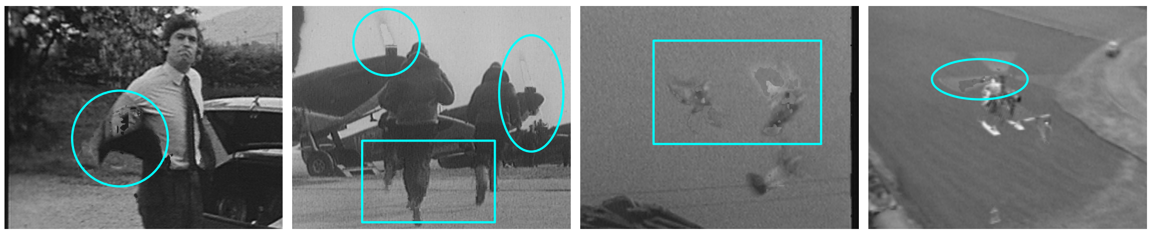
The critical design criteria of the proposed algorithm is that it eliminates image damage during missing data treatment. A primitive missing data treatment algorithm was implemented that takes a blotch mask from any detector and interpolates detected blotches with the mean motion compensated intensity of the previous and next frames.

Fig. 3.29 compares images restored using both the Blotch MRF algorithm and the proposed algorithm ($\delta_t = 15$). Frames from four sequences are shown. The first is the "jacket" sequence used in the ground truth experiments, in which the motion of the jacket is an example of self-occluding and non-rigid object motion. The second sequence shown is called the "vj" sequence and is an example of repetitive occlusions, as the aircraft propellors appear and disappear in alternative frames. The third sequence is the "birds" sequence. This sequence contains another example of repetitive occlusion (the bird's wings), however, in this case, the region of PM is moving across the frame. The final sequence (the "chopper" sequence) contains an example of motion blur. The rotation of the rotors is also an issue, as the motion estimator employs a purely translational motion model.

The four examples shown in Fig. 3.29 highlight the effectiveness of the proposed algorithm. Using the proposed algorithm prevents significant damage to each of the four images shown. However, the proposed algorithm is not always able to prevent all image damage, especially in cases where the region of PM is undergoing a translation (*i.e.* the motion of an object is the sum of PM and a non-pathological translation (see Fig. 3.30)). The assumption is made that motion compensation can be performed accurately, even in regions of PM. Consequently, the


(a) From left to right, frames from the "jacket", "vj", "birds" and "chopper" sequences.


(b) PM/blotch detections of the four frames for the proposed algorithm


(c) Frames restored with the Blotch MRF algorithm

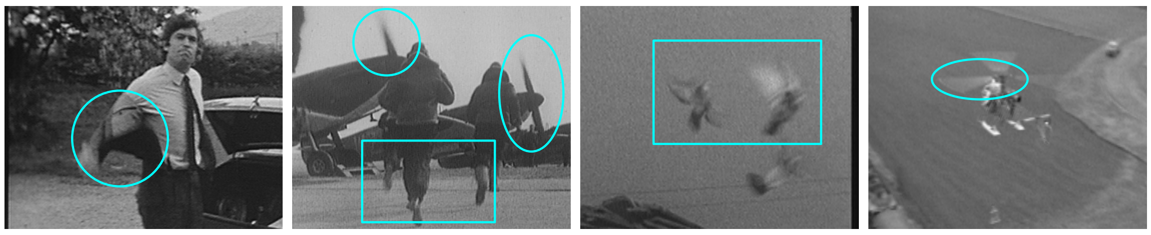
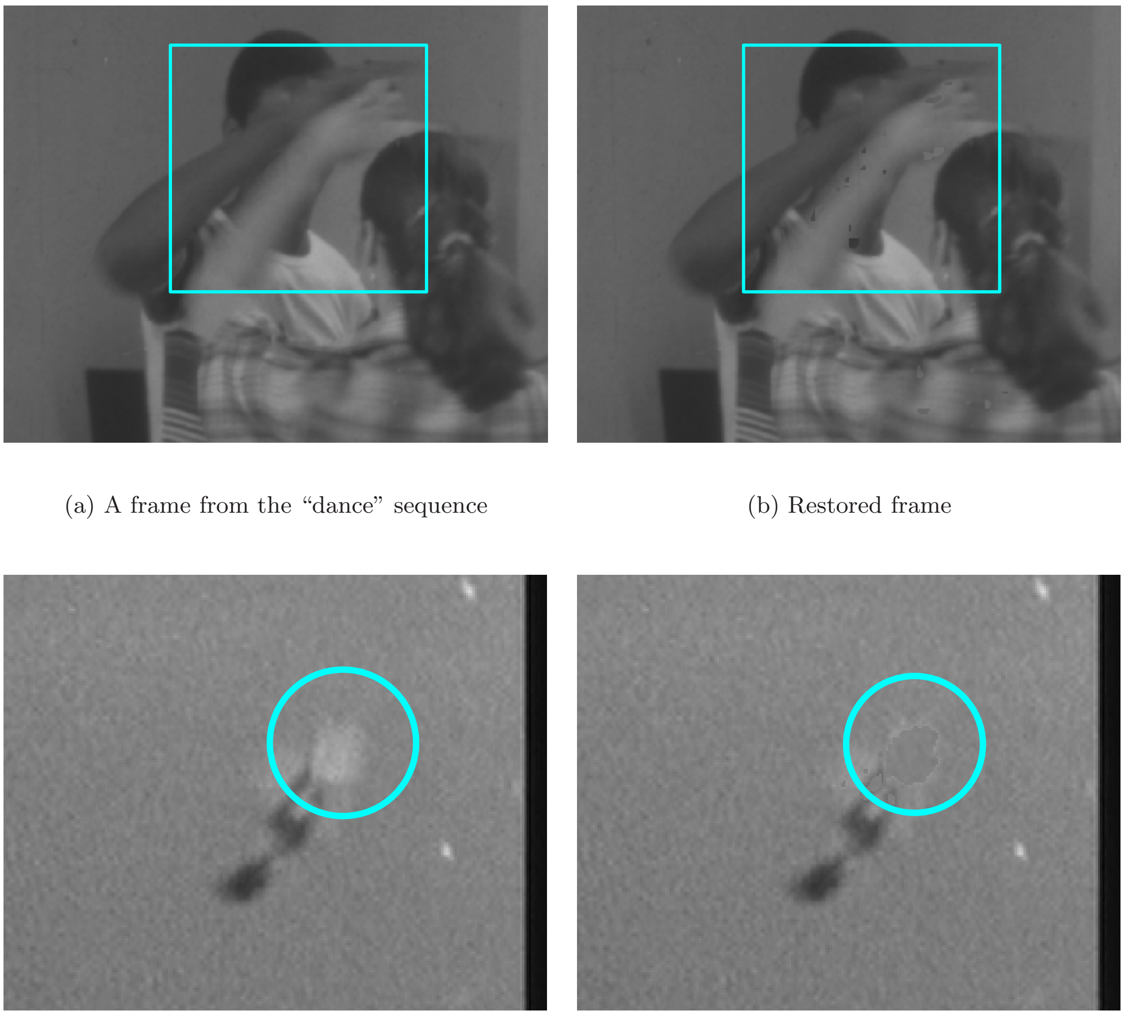

(d) Frames restored with the proposed algorithm

Figure 3.29: *This figure shows frames from four sequences restored using the the Blotch MRF and proposed algorithms as blotch detectors. The highlighted frames show image damage present if the Blotch MRF is used instead of the proposed algorithm.*



(a) A frame from the "dance" sequence

(b) Restored frame

(c) A frame from the "birds" sequence

(d) Restored frame

Figure 3.30: *This figure shows two examples where the proposed algorithm fails to prevent damage to image data (the highlighted regions). In each example, the object undergoing PM is also undergoing a translation (the motion of the arm in the "dance" sequence and the motion of the bird in the "birds" sequence).*

algorithm is prone to failure as it involves the motion compensation of four frames with respect to the central frame. However, it should also be noted that while some image damage may occur, any damage will be much less than the damage caused by performing missing data treatment without detection of PM.

3.4.6 Computational Complexity

Estimation of the prior energy and estimation of the MAP state at each pixel is the most computationally intensive aspect of the algorithm, amounting to approximately 90% of the execution time for a single frame. The upper bound for the complexity of the ICM optimisation is $\mathcal{O}(\#\text{ iterations } \times \#\text{ pixels } \times \#\text{ states } \times \#neighbours)$ where the number of states is 16 and the number of pixels in a given neighbourhood is eight. The actual order of complexity is slightly reduced as the smoothness prior is enforced on the label field (three states) rather than the full state field (16 states). The number of iterations necessary for convergence also depends on the number of states and the size of the neighbourhood, with a higher number of states and greater neighbourhood size requiring more iterations for convergence. From observation of the algorithm, the size of the δ_t threshold also impacts on the number iterations required. The average number of iterations performs decreases as δ_t increases. Computation time varies between 5 and 15 seconds per frame, depending on the number iterations performed.

3.5 Final Comments

In this chapter a robust Missing Data Detector designed to prevent false alarms caused by Pathological Motion was presented. The algorithm prevents false alarms by detecting regions of PM and excluding these regions from being detected as blotches. An integrated Bayesian framework is used to classify pixels as either PM or missing data based on the pattern of temporal discontinuities over a motion-compensated five-frame window. The algorithm also employs a divergence likelihood which biases the algorithm toward the detection of PM when the local motion fields of the window have high divergences.

The proposed algorithm was shown to reduce image damage when used as a detector in a Missing Data Treatment Framework (Section 3.4.5). A quantitative evaluation of the algorithm was presented by comparing it with four standard blotch detectors [50, 55, 61, 104] and the Bornard algorithm [8]. The mean correct detection and false alarm rates for missing data sequences of each detector were estimated on two ground truth sequences for which a ground truth exists. The comparison shows that the proposed algorithm gives a significantly reduced false alarm rate when compared with the standard detectors. Although a lower false alarm rate can be obtained with the Bornard algorithm at high thresholds, higher correct rates are possible with the proposed algorithm. In one of the test sequences the maximum correct detection rate is twice that of the Bornard algorithm.

The form of the divergence likelihood requires further consideration. Currently the divergence likelihood is evaluated from a divergence energy measure derived from the divergences of the motion fields over the five frame window. Use of the divergence information can give another independent measure of PM, improving the performance of the algorithm. Although the likelihood contains a model for the divergences of PM regions, it ignores the divergence of the motion field regions of missing data which can be large in the presence of large blotches. The quantitative evaluation of the algorithm also highlighted the variable influence on the two test sequences, indicating that the motion field divergence can not be used as a universal indicator of Pathological Motion. Furthermore, the influence of divergence may be affected by the choice of motion estimator - *e.g.* whether the motion field is defined on a block scale (like the motion estimator used in the experiments described in this chapter) or on a pixel scale.

The major area of further development in this area would be to integrate the algorithm into a Missing Data Treatment framework. The algorithm outlined in this chapter is presented as a stand-alone blotch detector which operates independently of a missing data interpolation stage. Previous work in Missing Data Treatment [53, 57, 60] has shown that integrating the detection, interpolation and motion estimation stages into a single bayesian framework allows for a more visually pleasing restoration. A new missing data treatment algorithm along the lines of the JONDI algorithm [57] using the proposed algorithm for the detection stage could result in an integrated algorithm resistant to Pathological Motion.

Part II

Global Motion Estimation

4 A Brief Review of the Globlal Motion Estimation State of the Art

In general, the observed motion between two images in a sequence is due to both object motion in the scene and global motion of the camera or entire scene. Local motion estimation typically refers to the process of estimating that portion of the motion in the sequece due to object motion. Global Motion Estimation is the process of estimating the component of motion due to camera or scene behaviour. In global motion estimation, a single set of motion parameters is used to describe the motion of the entire frame, rather than a set of parameters for each pixel or block of pixels. It has gained prominence recently in the video compression domain. Recent compression standards such as MPEG-4 [74] and H.264 [40] have used knowledge of the global motion parameters to improve the compression efficiency [32]. Global Motion Estimation is also used in restoration applications, most notably in image stabilisation [27, 69, 72] which aims to remove unintended high frequency camera motion (*i.e.* camera shake) from sequences.

While a simple two-parameter translation model is commonly sufficient for local motion estimation, in general more elaborate motion models are employed in global motion estimators, allowing them to represent the global motion more precisely. These models range from the simple two parameter translation model to the more complex affine models which are sufficient to model camera rotation and zoom, the eight parameter projective model which can detect perspective distortions and the twelve parameter quadratic model which allows for transformations with a parabolic curvature (See Table 4.1). The choice of model is effectively a compromise between precision and robustness. More complex models can give a more complete description of the

Motion Model	Number of Parameters	Definiton
translational	2	$(x, y) \rightarrow (x + m_0,\ y + m_1)$
zoom & translation	3	$(x, y) \rightarrow (m_2x + m_0,\ m_2y + m_1)$
zoom & rotation & translation	4	$(x, y) \rightarrow (m_2x - m_3y + m_0,\ m_3x + m_2y + m_1)$
affine	6	$(x, y) \rightarrow (m_2x + m_3y + m_0,\ m_4x + m_5y + m_1)$
projective	8	$(x, y) \rightarrow \left(\frac{m_2x+m_3y+m_0}{m_6x+m_7y+1}, \frac{m_4x+m_5y+m_1}{m_6x+m_7y+1}\right)$
quadratic/ parabolic	12	$(x, y) \rightarrow (m_2x + m_3y + m_6x^2 + m_7y^2 + m_8xy + m_0,$ $m_4x + m_5y + m_9x^2 + m_{10}y^2 + m_{11}xy + m_1)$

Table 4.1: *This table describes some common motion models used in global motion estimation algorithms. The right column describes the transformation of a pixel* $\mathbf{x} = (x, y)$ *by a parameter vector* $\mathbf{m} = \{m_0, ..., m_{k-1}\}$ *where* k *is the number of parameters in the model (i.e. it describes the transformation* $\mathbf{x} \rightarrow f(\mathbf{x}, \mathbf{m})$ *for each model).*

dominant motion. However, they are also less robust, requiring a greater amount of reliable data to generate reliable estimates. A typical compromise is the six parameter affine model which gives a sufficient approximation of the camera motion especially over small temporal intervals.

The most established method of estimating the dominant motion is to use the image sequence intensity function itself. The goal is to find the parameters that give the best match between two frames by minimizing the size of the global motion compensated difference over the full set of image pixels (*e.g.* minimum mean squared difference). An alternative strategy is to compute the global parameters from a local motion field by finding the parameters that most closely represents the dominant motion in the field. If access to the motion information of the sequence exists (*e.g.* for MPEG streams), algorithms using this strategy are typically less computationally intensive due to the lower resolution of most motion fields. However, the estimates are normally less precise, with the quality of the estimate being reliant on the quality of the algorithm used to calculate the local motion field.

Overcoming the influence of local motion is a major issue in the design of a global motion estimation algorithm. Foreground objects which are moving independently of the dominant motion can bias the estimation to the extant that the estimate is an average of the true global motion and the local motion. Consequently, foreground/background segmentation is often employed to reject objects undergoing local motion from the estimation process. Ill-conditioned image data is also a problem for global motion estimation. Although not as significant a factor as in local motion estimation where the support region is much smaller (*i.e.* a block of pixels rather than an entire image), the reliability of the estimate can be affected when large regions of an image are ill-conditioned, especially when more complex motion models are used.

This chapter presents a brief literature review of global motion estimation algorithms. It considers both image based and motion based algorithms. Each technique is assessed in terms of both the accuracy and robustness of the result and particular attention is paid to how each technique deals with local motion. Finally, the possibility for a new hybrid global motion estimation algorithm using both image based and motion field based techniques is explored.

4.1 Image Based Estimation

A similar image sequence model to that used in local motion estimation (See Section 2.1, Eq. 2.1) is also used for global motion estimation and is described by

$$I_n(\mathbf{x}) = I_{n-1}(f(\mathbf{x}, \mathbf{m})) + e(\mathbf{x}) \tag{4.1}$$

where I_n and I_{n-1} are the intensity functions of the n^{th} and $(n-1)^{th}$ frames and where $f(\mathbf{x}, \mathbf{m})$ describes the transformation function of each pixel, $\mathbf{x}$, between the two images given the global motion parameters $\mathbf{m}$. $e(\mathbf{x})$ is the assumed gaussian-distributed error in the model. The significant difference is that unlike local motion estimation the parameters $\mathbf{m}$ are fixed over the entire image. An appropriate method of finding the motion parameters is to chose the parameters, $\hat{\mathbf{m}}$, that minimises some enrergy function or the model error $e(\mathbf{x})$. Typically the mean square residual (model) error is used as follows

$$\begin{aligned}\hat{\mathbf{m}} &= \arg\min_{\mathbf{m}} \sum_{\mathbf{x}} (e(\mathbf{x}))^2 \\ &= \arg\min_{\mathbf{m}} \sum_{\mathbf{x}} (I_n(\mathbf{x}) - I_{n-1}(f(\mathbf{x}, \mathbf{m}))^2 \end{aligned} \tag{4.2}$$

One could use a direct approach to estimation by finding the parameters that minimise the motion compensated difference between the two reference frames. However, such an approach would be computationally intractable especially when more elaborate motion models are used due to the high number of unique parameter sets.

4.1.1 Parameter estimation using the Gauss-Newton Algorithm

A gradient based strategy can be employed to find the optimum set of parameters. A Taylor series expansion of the right hand side of equation 4.1 about an initial guess for the motion results in the approximation

$$I_n(\mathbf{x}) \approx I_{n-1}(f(\mathbf{x}, \mathbf{m}_0)) + \mathbf{u}^T \nabla_{\mathbf{m}} I_{n-1}(f(\mathbf{x}, \mathbf{m}_0)) \tag{4.3}$$

where the higher order terms of the expansion and the model error have been ignored and $\mathbf{u}$ is a parameter update vector the same length as $\mathbf{m}_0$ such that the new parameter estimate,$\mathbf{m}_1$, is then given by

$$\mathbf{m}_1 = \mathbf{u} + \mathbf{m}_0 \tag{4.4}$$

.

$\nabla_{\mathbf{m}}$ is a multi-dimensional gradient operator and for a k-parameter motion model is defined as

$$\nabla_{\mathbf{m}} = \frac{\partial}{\partial \mathbf{m}} = \left\{ \frac{\partial}{\partial m_0}, \frac{\partial}{\partial m_1}, \ldots\ldots\ldots\ldots, \frac{\partial}{\partial m_{k-1}} \right\} \tag{4.5}$$

with m_0 being the 1^{st} parameter of the model *etc.* . Equation 4.3 is an equation for a single pixel with k unknowns. Similar equations can be written for every pixel in the image resulting in an over-determined system of equations with a least squares solution of

$$\hat{\mathbf{u}} = \left(G^T G\right)^{-1} G^T \mathbf{z} \tag{4.6}$$

For convenience, the $\nabla_{\mathbf{m}} I_{n-1}(f(\mathbf{x}, \mathbf{m}_0))$ terms for each pixel have been collect into the matrix G (with dimensions of $\sum_{\mathbf{x}} 1 \times k$) where each row corresponds to the values $\nabla_{\mathbf{m}} I_{n-1}(f(\mathbf{x}, \mathbf{m}_0))$ for the corresponding pixels. Likewise the $I_n(\mathbf{x}) - I_{n-1}(f(\mathbf{x}, \mathbf{m}_0))$ are gathered into a vector $\mathbf{z}$.

This process can be iterated by expanding Eq. 4.1 about the new parameter estimate until the difference between successive thresholds is below a threshold. The chosen estimate $\hat{\mathbf{m}}$ is the value of the parameters at the iteration with the least mean square residual error.

The Levenberg-Marquardt algorithm [82] elaborates on the Gauss-Newton method by adding a damping term to Eq. 4.6 which becomes

$$\hat{\mathbf{u}} = \left(G^T G + \mu I\right)^{-1} G^T \mathbf{z} \tag{4.7}$$

where I is the $k \times k$ identity matrix. The non-negative damping term μ increases the stability of the solution when $G^T G$ is ill-conditioned. Equation 4.7 is similar to the wiener solution of the least squares problem. It handles the higher order terms of the Taylor series by considering them to contribute a gaussian white noise. The wiener solution is the parameter update vector $\hat{\mathbf{u}}$ that minimises the expect value of the squared error between $\hat{\mathbf{u}}$ the true update $\mathbf{u}$. A full derivation can be found in [4].

Robust Estimation of the Global Motion Parameters

The advantage of Gauss-Newton and Levenberg-Marquardt methods is that the global motion parameters can be estimated to an infinite precision. Furthermore, the method is compatible with all of the most common motion models. Global motion estimation algorithms have been implemented using the six parameter affine [59, 79, 94], eight parameter projective [32, 99] and 12-parameter parabolic [96] models.

The major weakness in the strategy is its failure to deal with local motion. Local motion is independent of the global motion and is thus considered an outlier. As a least squares approach is used significant local motion will bias the estimates of a global motion. Therefore, regions of local motion need to be excluded from the optimisation to achieve a robust estimate. To this end a weighting function $w(\mathbf{x})$ is defined. Regions that belong to the background and obey the camera motion are given high weights close to 1 and local motion regions are assigned low weights close to 0. The problem becomes one of minimising the mean square error of the weighted residual as follows

$$\begin{aligned}\hat{\mathbf{m}} &= \arg\min_{\mathbf{m}} \sum_{\mathbf{x}} w(\mathbf{x}) e(\mathbf{x})^2 \\ &= \arg\min_{\mathbf{m}} \sum_{\mathbf{x}} w(\mathbf{x})(I_n(\mathbf{x}) - I_{n-1}(f(\mathbf{x},\mathbf{m}))^2 \end{aligned} \tag{4.8}$$

The least squares estimate for the parameter update term $\hat{\mathbf{u}}$ then becomes

$$\hat{\mathbf{u}} = \left(G^T W G\right)^{-1} G^T W \mathbf{z} \tag{4.9}$$

where W is a diagonal matrix collecting the weights $w(\mathbf{x})$ over all pixel sites.

In effect, the problem now is to estimate the local/global segmentation of the frame defined by $w(\mathbf{x})$ and to perform the estimation of the parameters at pixels where $w(\mathbf{x})$ is close to 1. This problem is solved iteratively bu alternating between estimates of the segmentation and the motion parameters. At each step, the next estimate of the global motion is obtained using the most recent estimate of the segmentation and the segmentation is estimated given the new global motion estimate. This approach is known as Iteratively Re-weighted Least Squares (IRLS) [29, 43]. The next sections outline two approaches to computing the weights at each iteration.

M-Estimation

M-Estimation [43] presents a robust alternative to the least squares formulation by limiting the influence of outliers. In the standard least squares optimisation the contribution of each pixel to the overall energy function is a quadratic function of the residual value. Therefore, a single outlier can bias the overall energy as its energy will be greater than the energies of several inliers. In M-Estimation an energy function $\rho(.)$ is defined such that the optimisation problem becomes

$$\hat{\mathbf{m}} = \arg\min_{\mathbf{m}} \sum_{\mathbf{x}} \rho(e(\mathbf{x})). \tag{4.10}$$

The function ρ is known as the M-estimator which should limit the energy contributions of outliers. Several different M-estimators have been used in Global Motion Estimation including the Geman-McLure function used by Sawhney and Ayer [91] and Tukey's Biweight function used by Odobez and Bouthemy [79]. It can be shown the M-Estimation problem is equivalent to the

IRLS problem if the weighting function is defined by

$$w(\mathbf{x}) = \frac{\rho'(e(\mathbf{x}))}{2e(\mathbf{x})} \tag{4.11}$$

with $\rho'(.)$ being the derivative of the M-estimator, $\frac{d\rho}{de}$. Using either of the M-estimators described above results in a continuous weighting function with values close to 1 for low residual values, the value of the weight tending to 0 as the residual value increases. Hence, the influence of outliers is curtailed.

Truncated Quadratic

An alternate strategy is to define a binary weight function which gives 0 weight to residual values above a threshold [32, 59, 95] and is equivalent to a M-estimator defined by a truncated quadratic (See Fig. 4.1). The binary weighting function is given by

$$w(\mathbf{x}) = \begin{cases} 1 & e(\mathbf{x})^2 < e_t \\ 0 & e(\mathbf{x})^2 \geq e_t \end{cases} \tag{4.12}$$

where e_t is the threshold. In [95], Smolic and Ohm evaluate the threshold as being a multiple of the mean square residual (a factor of 1 is recommended). An alternative method of selecting the threshold is proposed by Dufaux and Konrad [32] to the select the residual value which would exclude the largest $T\%$ residual values.

Kokaram and Delacourt expand on the algorithm of Dufaux *et al.* by using the Levenberg-Marquardt extension of the IRLS framework. Consequently, equation 4.9 becomes

$$\hat{\mathbf{u}} \;=\; \left(G^T W G + \mu I\right)^{-1} G^T W \mathbf{z} \tag{4.13}$$

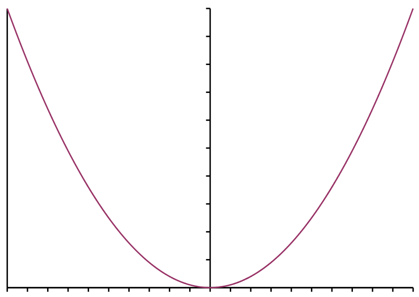
with the damping parameter μ in being adpatively set at

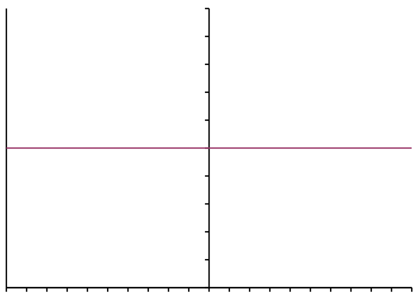
$$\mu = \|W\mathbf{z}\| \frac{\lambda_{\max}}{\lambda_{\min}} \tag{4.14}$$

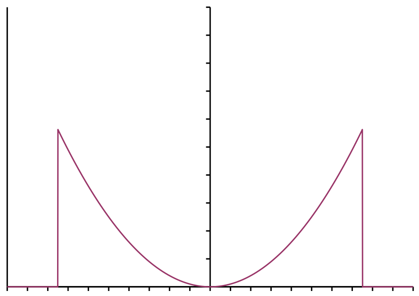
where $\lambda_{\max}$ and $\lambda_{\min}$ are the maximum and minimum eigenvalues of the matrix $G^T W G$. The ratio of maximum to minimum eigenvalue is a measure of the condition of the matrix $G^T W G$. Therefore, defining μ in this manner increases the damping when $G^T W G$ is ill-conditioned ensuring the stability of the system. A comparison of the different approaches to robust estimation is given in [28, 29].

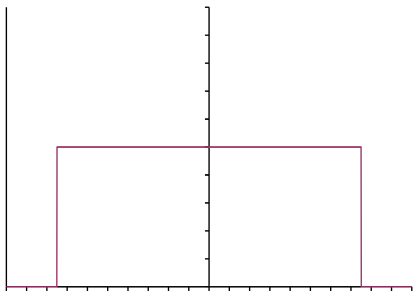
4.1.2 Phase Correlation

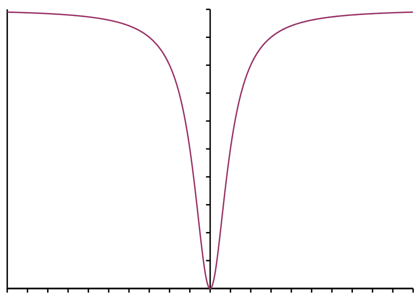
Phase Correlation [66] is another image based method which is more robust to local motion than the standard least squares approach. For the pure translation case (*i.e.* a 2 parameter


(a) Quadratic, M-Estimator


(b) Quadratic, Weighting Function


(c) Truncated Quadratic, M-Estimator


(d) Truncated Quadratic, Weighting Function


(e) Geman-McLure, M-Estimator

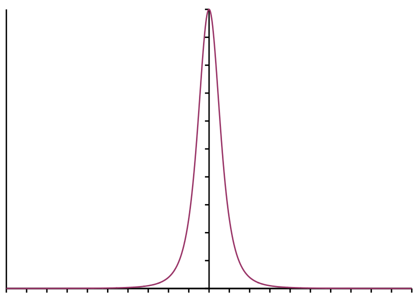

(f) Geman-McLure, Weighting Function

Figure 4.1: *Each row shows the 3 different M-Estimators ($\rho(e(\mathbf{x}))$ v $e(\mathbf{x})$) and their respective weighting functions ($w(\mathbf{x})$ v $e(\mathbf{x})$, $w(\mathbf{x}) = \frac{\rho'(e(\mathbf{x}))}{2e(\mathbf{x})}$). The requirement for the weighting function is that it gives high weights for low residual values and low weights for high residual values. The quadratic M-Estimator represents the case when a standard least squares regression (i.e. uniform weights) is performed, the truncated quadratic represents the binary weighting case and the Geman-McLure M-estimator is an example of an M-estimator that has a continuous weighting function.*

translation model), where $I_n(\mathbf{x}) = I_{n-1}(\mathbf{x} + \mathbf{d})$, the 2-D Fourier Transform of $I_n(\mathbf{x})$, $F_n(\mathbf{w})$ is given by

$$F_n(\mathbf{w}) = e^{j2\pi(\mathbf{d}.\mathbf{w})} F_{n-1}(\mathbf{w}) \tag{4.15}$$

The normalised cross-power spectrum of I_k and I_l is given by

$$\frac{F_n(\mathbf{w})F^*_{n-1}(\mathbf{w})}{|F_n(\mathbf{w})F^*_{n-1}(\mathbf{w})|} \;=\; e^{-j2\pi(\mathbf{d}.\mathbf{w})}, \tag{4.16}$$

the inverse Fourier transform of the which results in a delta function, $\delta(\mathbf{x}-\mathbf{d})$, where the impulse is located at the displacement d. For real images with local motion, a pure delta function is not obtained, but rather a surface which can be thought of as the pdf of the possible displacements between the images. Because the translation is typically estimated by choosing the location of the histogram mode, the algorithm is robust to local motion as long as the global motion is the dominant motion in the frame.

In [42] a technique for using phase correlation to estimate global motion using the four-parameter affine model given by

$$I_n(\mathbf{x}) = I_{n-1}\left(z_n \begin{bmatrix} \cos(\theta_n) & -\sin(\theta_n) \\ \sin(\theta_n) & \cos(\theta_n) \end{bmatrix} \mathbf{x} + \mathbf{d} \right) \tag{4.17}$$

is outlined where z_n is the zoom parameter and θ_n is the angle of rotation. Using the affine theorem for 2-D Fourier Transforms [13] and using log-polar coordinates, the magnitude spectrum of $I_n(\mathbf{x})$, $M_n(\mathbf{w})$, is given by

$$M_n(\log r, \theta) = M_{n-1}(\log r - \log z_n, \theta - \theta_n) \tag{4.18}$$

which is a shifted version of the magnitude spectrum of $I_{n-1}(\mathbf{x})$. Therefore, the zoom and rotation parameters can be estimated using standard phase correlation on the magnitude spectra and, after compensating for the zoom and rotation, the translation parameters are calculated as before.

The main advantage of phase correlation is its robustness to local motion and hence segmentation is not required. Furthermore, the process is not iterative which means that phase correlation is typically less computationally intensive than the gradient-based least squares techniques. However, estimating more than 4 parameters with phase correlation is substantially more complicated. This is in contrast with the gradient-based techniques which can be easily applied to more sophisticated motion models.

4.2 Motion Field Based Estimation

The common thread among many motion field based algorithms is that they recognise that the global motion parameters can be used to model a motion field over the entire image. Given a set of parameters defined by the vector $\mathbf{m}$, a motion vector field ($\mathbf{v}(\mathbf{x}, \mathbf{m})$) modelled on the parameters is described by

$$\mathbf{v}(\mathbf{x}, \mathbf{m}) = f(\mathbf{x}, \mathbf{m}) - \mathbf{x}. \tag{4.19}$$

The aim is to find the motion parameters that results in the modelled motion field that is closest to the obseved motion field $\mathbf{d}(\mathbf{x})$. This problem can be approached in a least squares fashion by trying to minimise the sum squared euclidean distance between the two motion fields as follows

$$\begin{aligned}\hat{\mathbf{m}} &= \arg\min_{\mathbf{m}} \sum_{\mathbf{x}} \|\mathbf{d}(\mathbf{x}) - \mathbf{v}(\mathbf{x}, \mathbf{m})\|^2 \\ &= \arg\min_{\mathbf{m}} \sum_{\mathbf{x}} \|(\mathbf{d}(\mathbf{x}) + \mathbf{x}) - f(\mathbf{x}, \mathbf{m})\|^2 \end{aligned} \tag{4.20}$$

where $\hat{\mathbf{m}}$ is the optimum set of parameters.

Taking the six parameter affine model described by

$$\begin{aligned} f(\mathbf{x}, \mathbf{m}) \;&=\; A\mathbf{x} \;+\; \mathbf{d}_g \\ A = \begin{bmatrix} m_2 & m_3 \\ m_4 & m_5 \end{bmatrix}&, \mathbf{d}_g = \begin{bmatrix} m_0 \\ m_1 \end{bmatrix} \end{aligned} \tag{4.21}$$

as an example (See also Table 4.1), the $f(\mathbf{x}, \mathbf{m})$ and $\mathbf{x} + \mathbf{d}(\mathbf{x})$ terms can be expressed in matrix form as

$$f(\mathbf{x}, \mathbf{m}) = B\mathbf{m}$$

$$\text{and } \mathbf{x} + \mathbf{d}(\mathbf{x}) = \mathbf{c} \tag{4.22}$$

$$\tag{4.23}$$

where

$$\begin{aligned} B &= \begin{bmatrix} 1 & 0 & x & y & 0 & 0 \\ 0 & 1 & 0 & 0 & x & y \end{bmatrix} \\ \mathbf{m} &= \begin{bmatrix} m_0 & m_1 & m_2 & m_3 & m_4 & m_5 \end{bmatrix}^T \\ \mathbf{c} &= \begin{bmatrix} x + d_x(x, y) \\ y + d_y(x, y) \end{bmatrix} \end{aligned}$$

and where x and y are the horizontal and vertical components of the pixel location vector $\mathbf{x}$ respectively (d_x and d_y represent the horizontal and vertical components of the observed motion field). By collecting the B and $\mathbf{c}$ terms for all the vectors defined in the motion field, an over-determined system of equations ($B\mathbf{m} = \mathbf{c}$) results which can be solved using a pseudo-inverse as follows

$$\hat{\mathbf{m}} = (B^T B)^{-1} B^T \mathbf{c} \tag{4.24}$$

Robust Estimation

Like the least squares techniques used in the image based methods, the above approach suffers from a lack of robustness to outliers in the observed motion fields. Once more an IRLS approach can be used to increase robustness to outliers. Inspired by the use of M-estimators in the image based techniques, Smolic *et al.* [94] proposed an M-estimator approach using a residual energy function defined by

$$e(\mathbf{x}) = |v_x(\mathbf{x}, \mathbf{m}) - d_x(\mathbf{x})| + |v_y(\mathbf{x}, \mathbf{m}) - d_y(\mathbf{x})|, \tag{4.25}$$

the sum of the absolute values of the horizontal and vertical differences between the modeled and observed motion fields.

Other approaches have adopted a binary weighting scheme. Tse and Baker [102] use an arbitrarily chosen threshold on the magnitude of the difference between $\mathbf{d}(\mathbf{x})$ and $\mathbf{v}(\mathbf{x}, \mathbf{m})$. An adaptive method for determining the threshold is proposed by McManus in [71]. Outliers are rejected based on the size ratio of $\|d(\mathbf{x}) - v(\mathbf{x}, \mathbf{m})\|$ to $d(\mathbf{x})$. If this ratio is greater than a threshold 0.5, the corresponding image locations are rejected as outliers. This ratio is recomputed if the proportion of outliers is less than 30%, ensuring that the proportion of outliers does not drop below this limit. Pilu [81] pre-filters the observed motion field in two stages to reject outliers. In the first stage, motion vectors in ill-conditioned regions are rejected by measuring the spatial gradient of the image intensities. This is followed by a spatial median-filtering of the observed motion field to reject isolated vectors.

4.2.1 Least Median of Squares Regression

Least Median of Squares (LMedS) Regression [90] can also be used to estimate the parameters. It has the advantage of being able to compute robust estimates for an outlier rate as high as 50%. In [68] Li *et al.* propose a motion field based algorithm. The optimum set of parameters is determined by minimising the median of the motion field difference as follows

$$\hat{\mathbf{m}} = \arg\min_{\mathbf{m}} \left(\operatorname*{median}_{\mathbf{x}}(\|\mathbf{d}(\mathbf{x}) - \mathbf{v}(\mathbf{x}, \mathbf{m})\|^2) \right). \tag{4.26}$$

In the LMedS framework a number of candidates for the motion parameters are generated. The Median of the square error is evaluated for each candidate and the candidate with the minimum value is chosen as the result. Each candidate is generated from the minimum amount of data required to produce a system of equations with a unique solution (*i.e.* $\mathbf{m} = B^{-1}\mathbf{c}$). Consequently, for a 2D motion field, the number of data samples required is half the number of parameters in the motion model (*e.g.* Three motion vectors are required for the six parameter affine model).

For a robust estimate to be obtained, the data set of at least one candidate must not contain any outliers. For a known outlier rate in the data of ϵ and a user-defined required probability of a robust estimate p, the minimum number of candidates, m, required for a robust estimate of the global motion is given by

$$m = \frac{\log(1-p)}{\log(1-(1-\epsilon)^{k/2})}. \tag{4.27}$$

where k is the number of parameters in the motion model [68]. So for a six parameter motion model and a required probability of robustness of 99%, 35 samples are required if the proportion of outliers in the data is expected to be 50%. A brief derivation of this equation is given in appendix C

4.2.2 Estimation using Histograms

The lower resolution of motion fields when compared with images makes the use of histograms for Global Motion Estimation computationally viable. Using histograms is a robust method of estimating the parameters as the modal parameter value is chosen as the estimate. However, computation of the histogram requires quantisation of the parameter space, resulting in a loss of precision.

Chiew *et al.* [14] describe a global motion estimation algorithm inspired by the Hough Transform [1] using the six parameter affine model. The algorithm involves the computation of a six dimensional histogram to describe the parameter pace. Each motion vector in the motion field polls for every set of global motion parameters that could model the vector. In the six parameter affine model the relationship between the observed motion vector $d(\mathbf{x})$ and the motion parameters is defined by

$$\mathbf{d}(\mathbf{x}) = f(\mathbf{x}, \mathbf{m}) - \mathbf{x} = (A - I)\mathbf{x} + \mathbf{d}_g \tag{4.28}$$

. For every possible set of affine parameters (A) the translation vector $\mathbf{d}_g$ is estimated and the corresponding histogram bin is incremented. The final parameter estimate corresponds to the histogram bin with the most votes.

Estimating the parameters in such a manner is a laborious process due to the high dimensionality of the histogram and the number of possible combinations of parameters for each motion vector[1]. The computational load can be reduced if separate histograms are used to estimate single parameters or pairs of parameters and if a neighbourhood of motion vectors votes for the one combination of motion parameters that can describe the motion parameters. Such an approach was adopted by Jozawa *et al.* in [47] which estimate a three parameter zoom and translation global motion model (See Table 4.1). Three histograms are used to estimate the parameters, with the modal parameter of each histogram being chosen as the estimate.

A similar approach is adopted by Coudray and Besserer to estimate global motion using the four [24] and six [26] parameter affine models. The translation parameters are estimated using a 2D histogram, with the remaining affine parameters are estimated using separate 1D histograms. The key observation of the algorithm is that the affine parameters are related to

[1]Assuming that n bins are used for each of the non-translation parameters there are n^{k-2} combinations for each vector.

the horizontal and vertical partial derivatives of the two components of the motion field. The parameters are estimated in two stages. First of all, the histograms on the affine parameters are accumulated by estimating the partial derivatives of the motion field. Once the affine parameters are known, the detected affine motion is removed from the motion field. Finally, the translation parameters are estimated by generating a 2D histogram of the compensated motion fields. In [26], an interesting application of the algorithm is introduced. A motion based segmentation is performed using watershed algorithm to isolate the peaks of the 2D translation histogram. Each peak corresponds to a different motion pattern in the image, one of which corresponds to the global motion. Consequently, this algorithm can be used to perform a foreground/background segmentation.

4.3 Discussion

In the ideal case, a Global Motion Estimation algorithm must be both robust and precise. It should be able to give accurate results even when significant local motion is present. Of all the methods outlined above the image gradient-based methods give the best precision, with the theoretical possibility of the parameters being estimated to an infinite precision. Both the phase correlation and the motion field based algorithms using histograms require quantisation of the parameter space limiting the precision of the final estimate. Furthermore, the motion field techniques have the added drawback of being dependent on the quality of the motion estimator used to compute the local motion field, adding a further source of imprecision to the estimate,

However, the gradient-based methods are not without their drawbacks. As has been outlined already, these methods are not robust to local motion and consequently, the IRLS framework was introduced to allow for local motion areas to be weighted out of the estimation process. Another limitation of the gradient based technique is that only small updates can be accurately computed at each iteration and, as a result, is not suitable for estimating large displacements. This limitation is a consequence of the truncated Taylor's Series expansion of $I_n(f(\mathbf{x}, \mathbf{m}))$, and hence the Taylor series only converges in a neighbourhood of the current guess of the motion $\mathbf{m}_0$. In the literature, this limitation has been mitigated by adopting a hierarchical approach [32, 59, 79, 91, 96, 99]. Similar to the multi-resolution approaches used in probabilistic frameworks (Section 3.3.4), a pyramid of images is created by successively low-pass filtering and down-sampling the target images. An initial estimate of the global motion at a given level of the pyramid is obtained by estimating the global motion at the next level up on the pyramid. Some algorithms have mitigated against this limitation further by adopting hybrid approaches. In these algorithms the hierarchical gradient based estimation is initialised using another technique, often using a less elaborate motion model than in the gradient based framework. These have included phase correlation [99], feature matching [96] and n-step search bock matching [17, 32].

The state of the art gradient based algorithms are reliant on accurate weighting to obtain robust estimates. However, when the residual based weighting methods outlined in Section 4.1.1

are used, a bad weighting can occur if the current estimate of the global motion is inaccurate. This in turn can lead to an erroneous motion estimate after the next estimate. Consequently, a vicious circle would result, whereby the estimated global motion would converge to a different value than the true global motion.

The fundamental problem is the suitability of using the residual values as a basis for foreground/background segmentation (Fig. 4.2). There are two ways in which failure can occur.

1. Because the segmentation is reliant on the estimation of motion compensated differences, inaccuracies in the motion compensation will result in high residual values in high spatial gradient regions (*i.e.* edges), even when these edges occur in the background. Consequently, these edges will be given low weights and will be excluded from the estimation. Background edges are important features in the estimation process and their exclusion could lead to ill-conditioning of the $G^T G$ matrix resulting in an unstable estimate.

2. Foreground objects with interiors of uniform intensities can be mistaken as background. The possibility exists that parts of the interior will be have low residual values and will be assigned high weights (*i.e.* for the background).

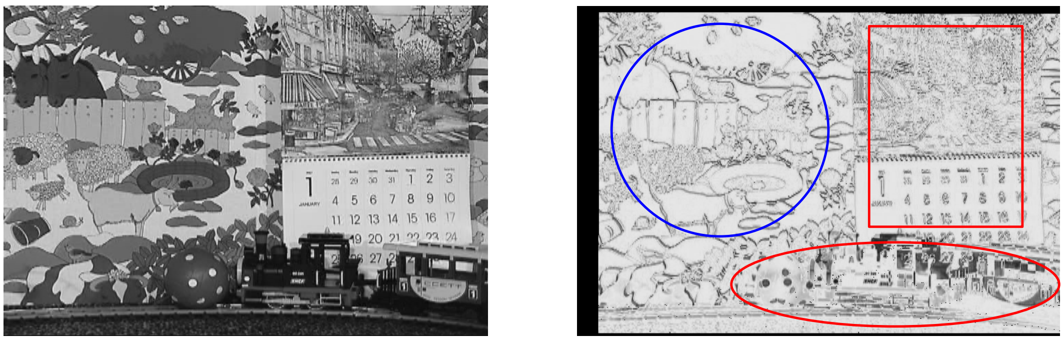
Noting this weakness in the residual based segmentation, it is useful to consider a new Global Motion Estimation algorithm based on the IRLS framework which does not employ a residual based segmentation scheme. There are two issues to be resolved, how to compute an initial estimate of global motion and how to perform the segmentation.

These issues are addressed by taking inspiration from the earlier work of Coudray and Besserer who propsed a robust method of estimating global motion from a local motion field [24]. This algorithm can be used to initialise the IRLS framework. Furthermore, the motion based segmentation algorithm of Coudray and Besserer [26] can be used as the basis for computing the weighting function. This new hybrid algorithm would be retain the precision of the gradient based methods and, through the use of the motion based segementation, would obtain the robustness of the motion based algorithm.

4.4 Final Comments

As part of this work, Global Motion Estimation was applied to two image processing applications, namely image sequence mosaicking and film tear correction.

Chapter 5 introduces the new hybrid algorithm for Global Motion Estimation (GME) for creating mosaics using motion information from an MPEG-2 stream. Mosaicking is the process of creating a single image consisting of a panaorama of an entire shot. It involves estimating the camera motion over the shot and the concatenation of the estimates over the shot to register each frame to a reference frame. As such, mosaicking is a suitable application to test the algorithm as any errors in the estimate will propagate through the mosaic.



(a) A frame from a sequence (b) Residual of the frame

Figure 4.2: *This figure highlights the limitations of using the residual value to perform foreground segmentation. In this sequence, the objects undergoing local motion are the calender (translating downwards), the train (translating left) and the ball (translation to the left and anti-clockwise rotation). The right image shows the residual computed with slightly inaccurate motion parameters. The bright regions indicate low residual values and high residual values are represented by dark regions. The blue circles a section of the background where high residual values can be seen at edges due to inaccurate motion compensation. These background edges are likely to be misclassified as foreground. The red circles highlight the regions of local motion in which low residual values exist and so would be mistaken as background.*

Chapter 6 outlines a framework to correct regional displacement introduced by the tearing of film material and is an example of global motion estimation applied to an image restoration application. In particular, the chapter introduces a novel algorithm to segment torn frames which draws on the motion based segmentation algorithm of Coudray and Besserer [26].

5 A Hybrid Algorithm for Robust and Precise Global Motion Estimation from MPEG Streams

MPEG standards exploit the temporal redundancy in image sequences in order to achieve a more efficient compression of the sequence. MPEG streams contain local motion fields which allow frames to be reconstructed from a reference frame and the motion information linking the two frames. Exploitation of the motion information in MPEG streams is desirable since the motion information is provided in the stream itself. However, MPEG motion vectors are generally unsuitable for image manipulation applications due to the poor quality of the motion fields (See Fig. 5.1).

In [24, 71] Global Motion Estimation (GME) algorithms are outlined which derive the estimates of the global motion parameters from vector fields obtained from an MPEG stream. Extracting the global motion parameters from MPEG streams is an attractive prospect because it allows for quick estimation of the parameters. In [24] GME is performed by using histograms of the local motion field between two frames to derive the global motion parameters, and hence it is robust to local motion. Furthermore, it allows for a quick and coarse motion-based background segmentation and allows for global motion estimates between consecutive I-frames, which are typically 12 to 15 frames apart. However, the precision of the estimate is dependent on the accuracy of the motion field. In other words, poor quality MPEG vectors will result in a poor parameter estimate. For clarity, it is referred to here as the Robust Accumulation for Global

This work has been published in part in [23].

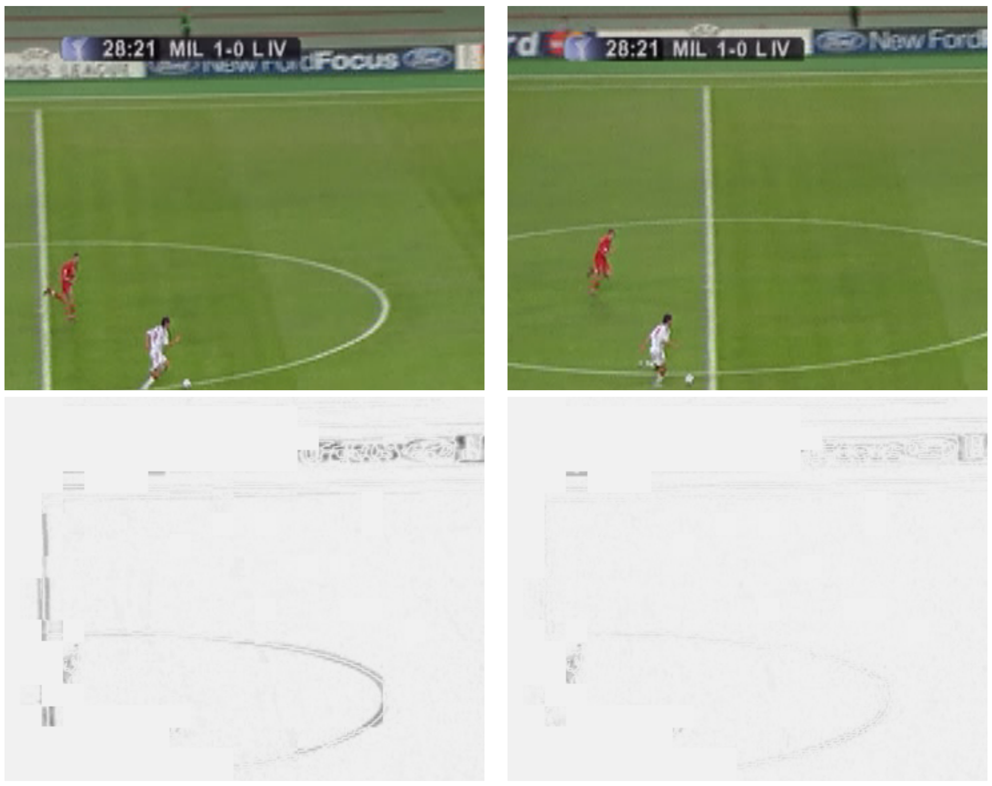


Figure 5.1: *This figure shows an example of the problems encountered when using MPEG-vectors for image manipulation. The bottom left image shows the motion compensation error (dark colours represent high errors) if MPEG motion vectors are used to compensate between the two images on the top row. The alignment of the pitch lines and the advertising hoardings is poor because of the poor quality of the MPEG vectors. The bottom right image shows the compensation error for the hybrid GME algorithm. In this image, the lines are much better aligned and hence the compensation error is lower.*

Motion-field mODelling (RAGMOD) algorithm.

In order to overcome these limitations, gradient-based techniques, [59,79], can be used to refine the estimate obtained from the RAGMOD algorithm. Gradient-based GME techniques can precisely estimate small global motions. However, gradient-based techniques are easily biased by local motion and so a background/foreground segmentation is needed to ensure robustness to local motion. The resulting hybrid algorithm aims to combine the robustness to local motion of RAGMOD with the precision of the gradient-based techniques.

A challenging application for the hybrid algorithm is mosaicking from MPEG-2 streams. Mosaicking is the formation of a panorama from a group of images or an image sequence. It is a well established application in Image Processing [77, 91, 99] and is also incorporated into MPEG-4 as the notion of sprites [106]. The major task in mosaicking is the estimation of the relative positions of each of the images in the sequence, which requires Global Motion Estimation

(GME). Typically, the position of each frame in the sequence is estimated relative to the first frame by combining the global motion estimates of all frames between the first and the current frames. This results in a single Global Motion Estimate which describes the transformation from the current to first frame of the sequence. Because matching is performed in this manner, an error in a global motion estimate between any two consecutive frames will affect the relative position of all subsequent images in the mosaic. Therefore, precise and robust Global Motion Estimation is essential for creating accurate mosaics.

This chapter outlines a hybrid GME algorithm which generates estimates from MPEG-2 streams. An initial guess for the global motion is obtained from the RAGMOD algorithm, which also produces a coarse background segmentation. The estimate and the background segmentation are then used to initialise a gradient-based GME framework producing the final result. Before the hybrid algorithm is outlined, a brief review of previous GME techniques is presented. The subsequent sections detail the hybrid algorithm, including a brief overview of the RAGMOD and gradient-based techniques. Finally, sections outlining the simulation results and conclusions are presented.

5.1 The Hybrid Algorithm

The Hybrid algorithm is a robust and precise Global Motion Estimation algorithm which exploits the motion information used in MPEG-2 streams. The algorithm proceeds in 3 stages.

Vector field aggregation The first stage of the algorithm involves preparing the MPEG motion information for GME (Appendix A). The current application for the GME algorithm is mosaicking and so it is not necessary to estimate motion between each frame. Since MPEG-2 streams are to be used and I-frames are the only frames which contain all spatial information, estimates of the global motion are obtained between consecutive I-frames. To achieve this goal, a motion vector field mapping consecutive I-frames is calculated by aggregating the motion information between two I-frames. With the standard MPEG profile, I-frames appear every 12 or 15 frames. Thus the mosaics are typically composed from images a half second apart in an image sequence. This also has the advantage of speeding up the mosaicking process as fewer estimates have to be carried out.

The vector field is then weighted by the texture information in each of the image macroblocks. Blocks with few texture features (*e.g.* a clear sky) are given a low weight, while blocks with significant texture features (*e.g.* edges) are given high weights. This ensures that vectors in flat textureless regions are weighted out of the estimation, making the GME more robust to the aperture effect. At the end of the first stage, a vector field mapping two consecutive I-frames has been obtained, where each vector is weighted according to the texture information in the corresponding image block.

Initial estimation using RAGMOD After the aggregation stage, the RAGMOD algorithm is used to estimate the global motion parameters from the aggregated vector field (Appendix B). The algorithm uses histograms of the motion field and so is robust to local motion. A four-parameter zoom and rotation affine model is employed, and the zoom and rotation parameters are estimated initially. Compensation of the motion field for the zoom and rotation is then performed, in theory leaving a purely translational motion field. A further histogram is then used to estimate the translation parameters from the compensated motion field. This histogram is also used as the basis for a motion-based segmentation, which is achieved by extracting the distinct peaks in the histogram. The translation histogram is typically multi-modal. Each peak corresponds to the different motion patterns in the motion field, one of which represents the background. By selecting this peak, the background is segmented. As the motion field is block-based, the segmentation map is coarse. However, it has the advantage of its accuracy being independent of the precision of the global motion estimate. At the end of the second stage, an initial estimate of the global motion parameters is obtained. Additionally, a coarse background segmentation map has been obtained from the motion based segmentation

Refinement using a gradient-based estimation A gradient-based framework using a six parameter affine model is used to refine the estimate obtained from the RAGMOD algorithm. Firstly, the four parameter result from the RAGMOD algorithm is converted to a six parameter estimate. The four parameters of the RAGMOD algorithm are the zoom and rotation parameters r and z and the translation parameters d_x and d_y. This corresponds to a six parameter motion model as follows

$$\begin{aligned} f(\mathbf{x},\mathbf{m}) &= A\mathbf{x} + \mathbf{d}_g \\ A &= \begin{bmatrix} a_1 & a_2 \\ a_3 & a_4 \end{bmatrix} = \begin{bmatrix} 1+z & -r \\ r & 1+z \end{bmatrix}, \quad \mathbf{d}_g = \begin{bmatrix} d_1 \\ d_2 \end{bmatrix} = \begin{bmatrix} d_x \\ d_y \end{bmatrix} \end{aligned} \tag{5.1}$$

The result is then used to initialise the gradient-based framework. In the hybrid algorithm, a standard IRLS framework based on the Gauss-Newton Algorithm (See Section 4.1.1 in the previous chapter) is employed at full image resolution which attempts to minimise the weighted sum squares residual according to

$$\hat{\mathbf{m}} = \arg\min_{\mathbf{m}} \sum_{\mathbf{x}} w(\mathbf{x})(I_n(\mathbf{x}) - I_{n-1}(f(\mathbf{x},\mathbf{m}))^2. \tag{5.2}$$

The novelty is in the method of rejection of local motion (*i.e.* of estimating $w(\mathbf{x})$). Instead of using the residual values (*i.e.* $I_n(\mathbf{x}) - I_{n-1}(f(\mathbf{x},\mathbf{m}))$, the motion based segmentation is used to reject local motion regions. The weighting function at each iteration, is given by the overlapping background of the current and motion compensated previous frames as computes by the motion based segmentation algorithm, the expression for which is

$$w(\mathbf{x}) = h_n(\mathbf{x}) \times h_{n-1}(f(\mathbf{x},\mathbf{m})) \tag{5.3}$$

where h_n and h_{n-1} are the binary background maps of frames n and $n-1$ estimated from the motion based segmentation. The motion based segmentation is not updated at each iteration. However, the weighting function is updated at each iteration according to Eq. 5.3 and the current value of the motion parameters.

5.2 Results

Finding an objective measure to assess the performance of a Global Motion Estimation algorithm is not a simple proposition. Ideally, the true global motion parameters for a sequence would be known, and the estimated parameters would be compared with this ground truth. Practically, for most image sequences, no such ground truth exists and an alternative measure of performance is needed. One practical measure of performance is to use the mean square value of the Displaced Frame Difference as an error measure of the estimation. The minimal value of the mean square DFD is said to represent the optimal global motion estimate. The main drawback with such a method is that local motion would bias the error measures as local motion causes high DFD values. A solution to this problem would be to consider the mean square value of a DFD weighted by a background segmentation. However, unless a ground truth segmentation exists, the fidelity of the error measure is affected be the choice of background segmentation used. In fact, it has already been mentioned in last chapter (Section 4.1.1, Eq. 4.8) that gradient-based techniques are optimised with respect to the local motion rejection strategy used and so using the same strategy for evaluation will bias the result in favour of the algorithm.

Another strategy for evaluation of performance is by visual inspection of the motion compensated images. Obviously, such a subjective measure is imprecise by its very nature. In this chapter, mosaicking is also used as a form of visual evaluation. As errors in GME propagate through the mosaic, a systematic error in the GME algorithm will be more easily detected in a mosaic than from visual inspection of a single estimation.

A number of experiments have been performed in order to evaluate the performance of the Hybrid GME algorithm. In order to have available a "ground truth" sequence, a sequence was shot of a scene in which markers were strategically placed to allow a set of parameters to be estimated by tracking the markers' positions in each frame. The hybrid algorithm is also compared to a gradient-based technique using M-estimation, the significant difference being the different segmentation strategies employed (*i.e.* picture-based v motion-based segmentation). The following sections detail the results over the sequences tested.

5.2.1 Ground Truth Estimation

The most obvious method to get ground truth data is to record all camera motion parameters while acquiring the video using a camera turret (and zoom controller) which would allow the camera to undergo a known predefined motion. This was attempted with a primative turret,

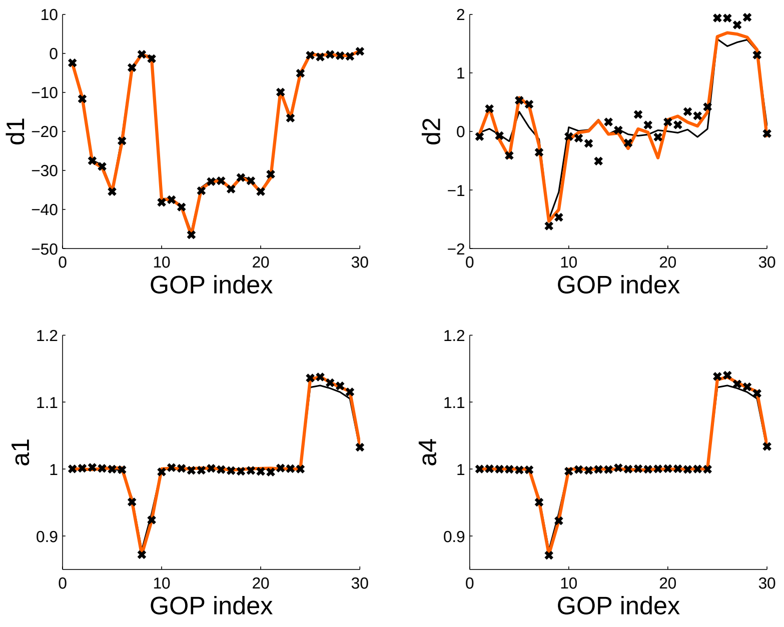


Figure 5.2: *Ground truth : parameters computed from marker tracking (x) constitute the reference estimation. RAGMOD estimation (black lines) is already close to the reference one, and hybrid estimation (orange lines) is similar. Improvements given by Hybrid estimation are clearly visible on scale parameters (a_1 and a_4).*

but the results were unreliable. Another option, although not strictly “ground truth”, is to add visual markers to the scene, which can be easily tracked and therefore deliver global motion parameters by different method to the hybrid algorithm. An attempt was made to capture a scene in which markers were positioned allowing them to be subsequently tracked (Fig. 5.5a). A road is part of the recorded sequence, so passing vehicles are a source of local motion and a possible disturber for the GME. During filming the camera was placed on a fixed tripod and moved by hand. Initially the camera moves to the left. This is followed by a zoom in and then another pan to the left. Finally the camera zooms out, revealing more of the scene.

In order to estimate the ground truth estimation, all visible markers are manually selected and tracked using the post-production compositing software *Combustion 4* [18]. A minimum of three markers are required in each frame along the sequence for the ground truth to be estimated using a six parameter affine model. The GME parameters are calculated from I-frame to I-frame by using a linear least squares fit ($f(\mathbf{x}, \mathbf{a}) = A\mathbf{x} + \mathbf{d}$) to the motion of the markers located in each frame.

To compare the results, parameters of the different estimations are plotted in the same reference chart (GOP (I-frame) index as x-axis, GME parameters (affine model) on the y-axis (Fig. 5.2). Although the complete affine model is used, rotation parameters a_2 and a_3 are close to zero for this sequence and hence are not plotted.

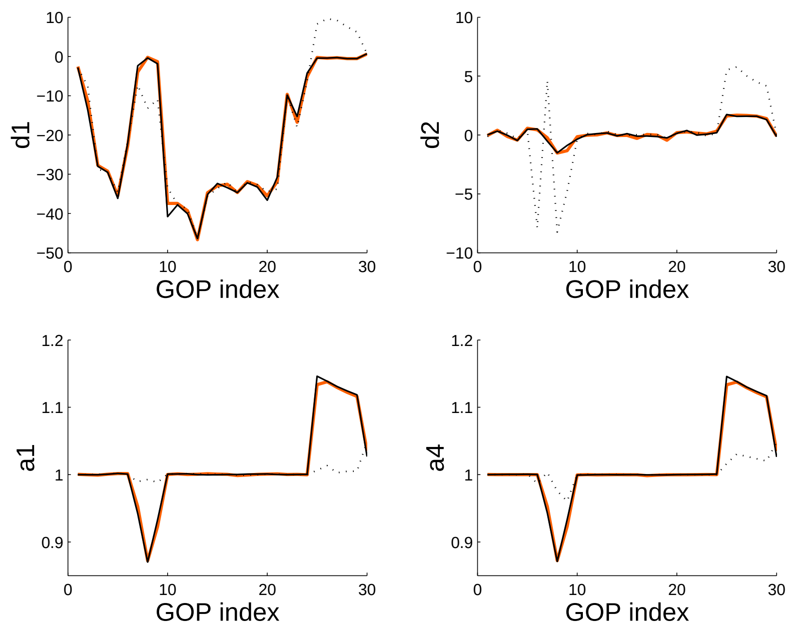


Figure 5.3: *Hybrid estimation (orange lines) is compared here to compared to Motion2D estimation. Black lines represent results when estimation is made frame by frame (parameters are combined along each GOP), and dotted lines when estimation is made directly between each I-Frames. Hybrid and frame-to-frame estimation give similar parameters, but direct estimation between I-Frames is impossible with Motion2D.*

The test sequence used for the "ground truth" data is encoded by the *Cinemacraft* [16] encoder (GOP=15, IBBPBBPBBPBBPBBI, VBR), and the hybrid algorithm is used to estimate the motion from the resulting MPEG sequence. These first plots show good GME with using the RAGMOD algorithm only, and an accuracy improvement after the second stage of the hybrid algorithm.

One advantage of making a initial estimation on MPEG data is to allow global motion estimation between distant frames in the time, while the precision (comparable to frame by frame estimation) is added by the second stage. To demonstrate this, the results of our method should be compared with estimations provided by a reference frame-by-frame estimation method. For this purpose the motion estimation tool Motion2D [78] based on work of the IRISA team [79] was chosen. Unlike the hybrid algorithm, Motion2D does not use any of the motion information from the MPEG stream and uses M-estimation for the local motion rejection in the gradient-based framework. Motion2D uses the fully decoded frames from the input MPEG stream, but the timestep for the results has be adapted for direct comparison with the hybrid method. Therefore, using Motion2D, two sets of estimation are computed. The first one is a frame by frame estimation, with the the estimated parameters being chained over the 15 frames in order to produce one estimation for a time span that matches a GOP. The second estimation is performed

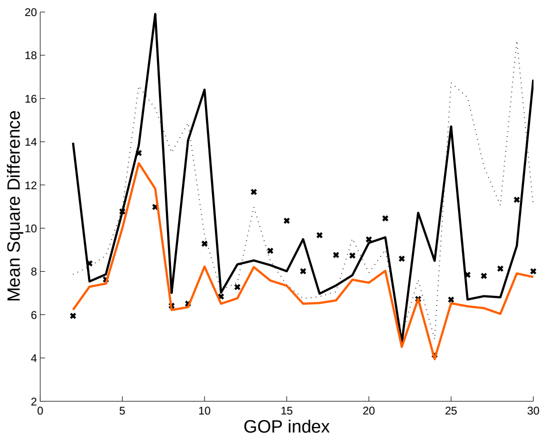


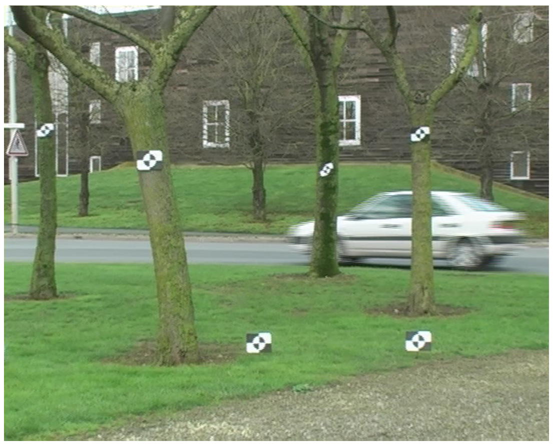
Figure 5.4: *Mean square difference between background pixels of current frame (given by motion based segmentation) and compensated previous frame. RAGMOD estimation (dot lines) is the worst (mean of 10.29) and hybrid estimation the best (7.31). Motion2D (black line) and estimation made from markers (crosses) are between (respectively 9.90 and 8.58).*

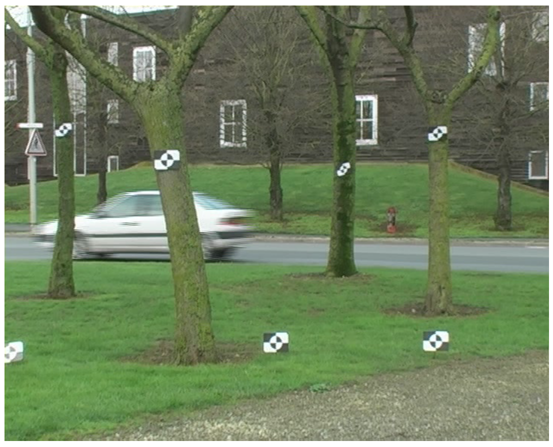
on a "1-out-of-15" basis (*i.e.* direct estimation over 15 frames), the chosen frames matching the I-frames of the MPEG sequence.

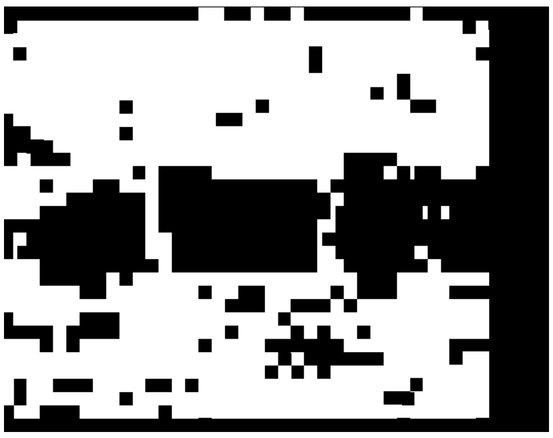
The estimated parameters are plotted on the same scales in Fig. 5.3. It can be clearly seen that an I-Frame to I-Frame (1-out-of-15) estimation with Motion2D leads to poor results as this time span is too large to allow direct estimation between I-frames. The results provided by the Hybrid method which calculates directly between I-frames gives results equivalent to a frame-to-frame estimation computed by Motion2D.

The mean square DFD between each frame, I_t, and the registered (motion compensated) previous frame, I_{t-1}, computed on background luminance pixels (Fig. 5.4) is used to compare the performance of the hybrid and Motion2D algorithms. As with the refinement stage, the common background obtained from the motion-based segmentation between the current and motion compensated previous frame is used to compute the mean square DFD. Although this would naturally bias the results in favour of the hybrid algorithm, this method was chosen because it is less likely to reject important background edges (See Fig. 5.5, Fig. 5.9). Despite little detected difference between the results, the hybrid estimation scheme gives a slightly better estimation. It was discovered that, besides error accumulation while chaining the frame-by-frame estimations, most differences between the parameter values are due to influence of local motion and the strategy used to reject the local motion.

Fig. 5.5 illustrates these differences. Although the Motion2D segmentation is more precise, many background edges (tree borders, marker borders, window borders) are rejected because of remaining estimation errors. Consequently, these edges are not considered in recursive GME, despite the fact that these edges are very important in motion estimation (aperture problem). In the hybrid method, these edges are labeled as background and so the estimation is made on more


(a) I-Frame n


(b) I-Frame $n+1$


(c) White pixels represent background pixels given by MPEG motion based segmentation for I-frame $n+1$. This result is used in the gradient based refinement.

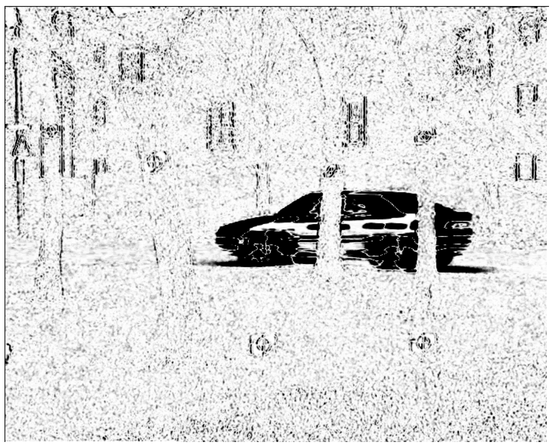

(d) Luminance represents the weight of each pixel in the Motion2D algorithm. (sample taken between two consecutive frames and not between I-Frames)

Figure 5.5: *MPEG segmentation is not very accurate (blockwise and noisy data) but some important background edges are rejected in Motion2D algorithm (markers,windows...).*

reliable data. Furthermore, imprecise estimates give high DFD values in the neighbourhood of these edges and hence rejecting these edges for evaluation purposes gives a false impression of the performance of the GME algorithm. Therefore, in order to evaluate the algorithms, is preferable to conserve the majority of these edges even if some background is ignored and so the motion-based segmentation is used.

In summary, it has been shown that the Hybrid GME algorithm gives an estimation with high accuracy, even in the presence of local motion. The initial estimation is made once per

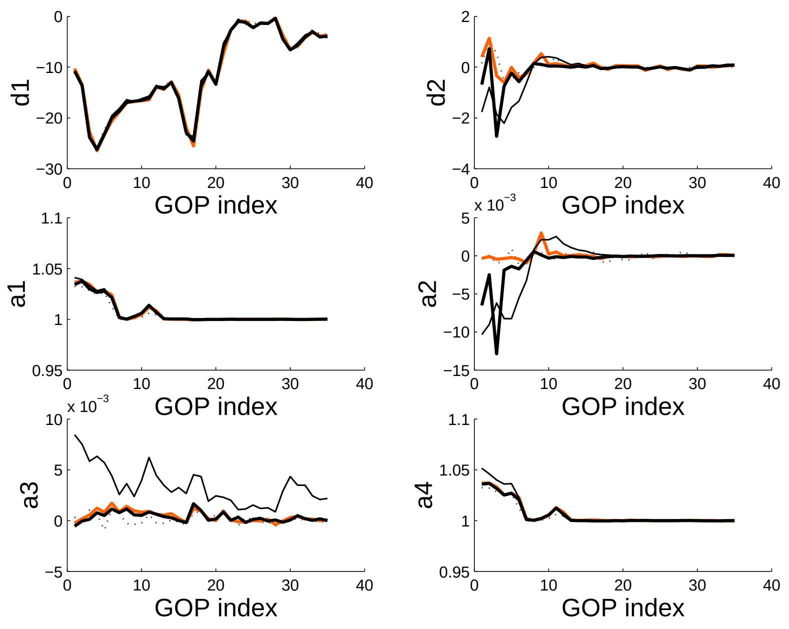


Figure 5.6: *This figures plots the motion parameters of "Mobile and calendar" sequence. The top row plots the translation parameters d_1 and d_2 while the middle and bottom rows show the affine parameters. Estimation made frame-to-frame by Motion2D (thin black lines) is biased by the calendar motion in the beginning of the sequence. The three other estimations (dot lines for RAGMOD only, orange lines for Hybrid and back lines for I-frame-to-I-frame Motion2D) give equivalent results except when calendar is prominent in the frame in which case Motion2D cannot distinguish between the background and calendar (See Fig. 5.9).*

GOP (depending of MPEG profile, every 12 or 15 frames) in real-time, and the second estimation stage provides an equivalent accuracy to a frame-by-frame robust estimator. The subsequent sections emphasise the advantages and also the limitations of the Hybrid algorithm, especially with regard to the motion-based segmentation.

5.2.2 Picture-based v Motion-based segmentation

For the ground truth sequence, the choice of segmentation method is not critical because there is enough texture present in the background to allow the Motion2D method to generate accurate results. However, this will not be the case for all sequences and an example of such a sequence will now be discussed.

"Mobile and Calendar"

To illustrate the hybrid method's resilience to bias caused by local motion, a sequence exhibiting widespread and slow local motion is used as a test sequence. This sequence is the well-known "Mobile and calendar" sequence, a commonly used reference in the research community in testing

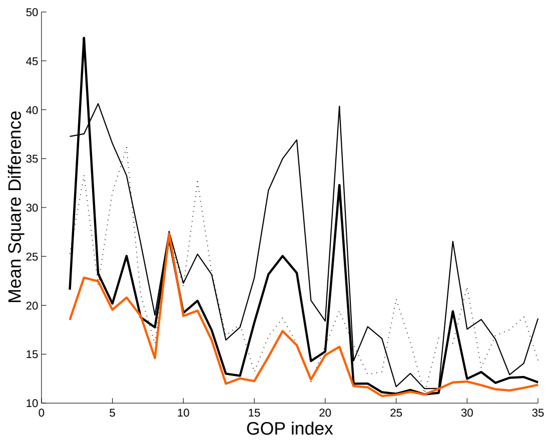
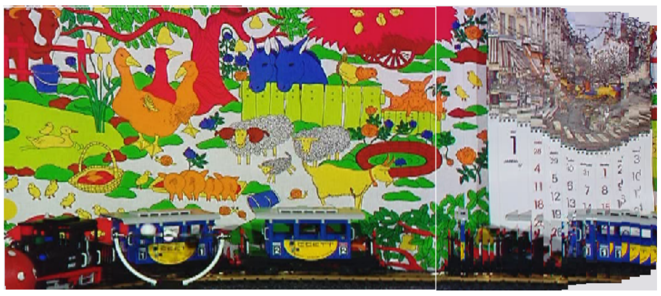


Figure 5.7: *Mean square DFD comparison on "Mobile and calendar" sequence. Frame-to-frame Motion2D estimation (thin black lines) gives the worst estimation because the calendar motion is not well rejected (mean of 23.23). Hybrid estimation (orange lines - 14.95) improves RAGMOD estimation (dot lines - 19.51), and I-frame-to-I-frame Motion2D estimation (black lines - 17.92) give intermediate quality.*

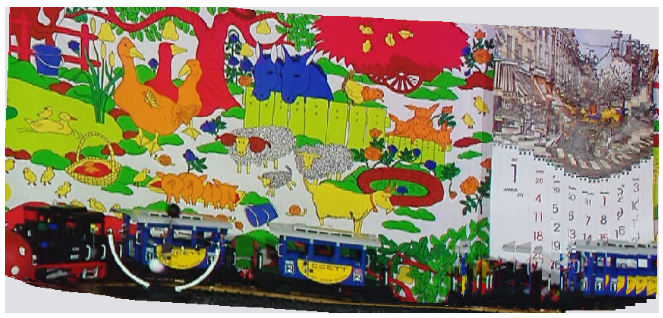
motion detection & segmentation algorithms. In this sequence the camera moves to the left, and slightly zooms-out. Additionally the scene contains many local motions (*e.g.* the calender is moving vertically and the train is moving to the left pushing the ball along the track). Since the camera motion is gentle, the Motion2D software is able to estimate global motion using a 1-out-of-12 scheme (estimation of the parameters directly from I-frame to I-frame). As with the previous test sequence, two estimation sets have been computed (frame-to-frame and I-frame-to-I-frame) with the Motion2D software. All obtained parameters have been plotted on the same scales in Fig. 5.6.

The Motion2D algorithm, or more generally all GME estimators using robust regression, could fail on this particular sequence. If global motion is estimated on a frame-by-frame basis, the calendar motion is not important enough to be "rejected". In fact, since the calendar shows a vertical motion (it is falling down) in the right area of the scene, a non-existent "global rotation" is estimated (See Fig. 5.8). This occurs because the motion of the calender is not different enough from the global motion. Therefore, the residual error in the Motion 2D segmentation is not large enough to allow the calender to be rejected and so the calender is mistakenly segmented as background (See Fig. 5.9).

However, when the estimation is performed over a complete GOP with Motion2D, the calender is correctly rejected and the estimation is more accurate because the global warping over the GOP is sufficiently large. Considering the hybrid approach, the motion-based segmentation is not as perturbed by the calendar motion. Therefore, the hybrid algorithm is more robust to local motion in this sequence. Once again, the hybrid approach is better at segmenting background edges, and so gives better error performance (Fig. 5.7).



(a) Mosaic using the Hybrid algorithm



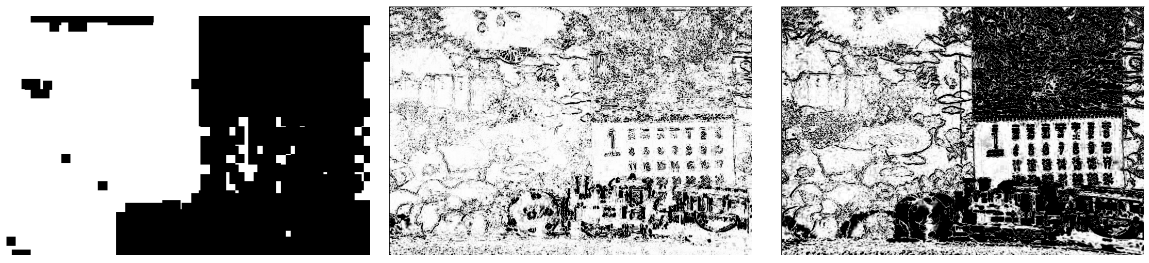
(b) Mosaic using Motion2D on a frame-by-frame basis

Figure 5.8: *This figure demonstrates the failure in the frame-by-frame Motion2D estimation. In this sequence no global rotation is present. However, when using Motion 2D, a false clockwise rotation, introduced by the calender motion, is detected and the mosaic appears to ramp upwards.*

The hybrid algorithm is able to estimate global motion between distant frames (not easy for more traditional methods) with more robust local motion rejection based on motion segmentation. In order to be less time consuming, the motion-based segmentation uses the motion vectors from an MPEG-2 stream. The next section illustrates some limitations of using the MPEG motion information for segmentation in sequences with more complex motion.

The "cathedral" sequence

A sequence exhibiting more complex motion is now discussed. In this "cathedral" sequence, the camera is focused on the top of the cathedral and then quickly tilts down with a rotation, until the queued people in front of the cathedral are revealed. As in the "ground truth" sequence, the warping is too large for estimating global motion directly between each I-frame (1-out-of-15) with Motion2D. Therefore a comparison is only made between the RAGMOD, Hybrid and Motion2D frame-to-frame methods.



(a) MPEG background selection

(b) Motion2D frame-to-frame background selection

(c) Motion2D "1-out-of-12" background selection

Figure 5.9: *White pixels represent pixels used in the compensation error minimization by Hybrid method(a) and by Motion2D frame-to-frame(b) and I-frame-to-I-frame(c). In the Motion2D frame-to-frame map(b)the motion of the calender is not different enough from the background and is incorrectly segmented as background. In (c) the motion of the calender is large enough and is largely correctly detected (apart from the bottom of the calender which has low texture). Again background edges are not taken into account with the Motion2D algorithm.*

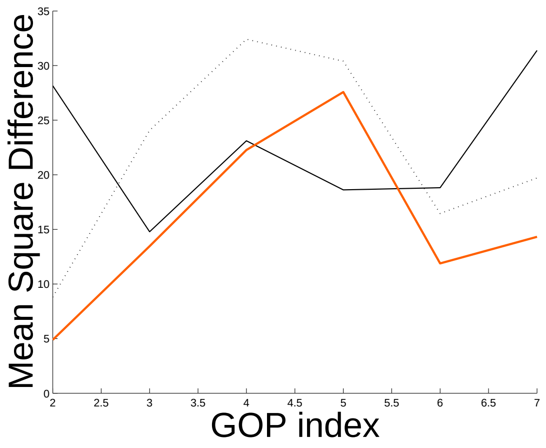
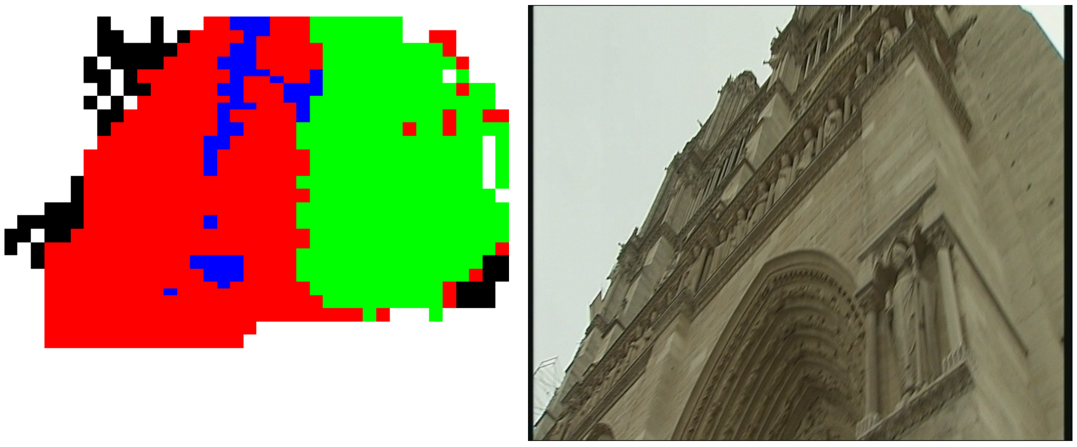


Figure 5.10: *Mean square difference comparison on the "Cathedral" sequence. Hybrid estimation (orange lines) gives better results than MPEG (dot lines) or Motion2D (black lines), except for GOP 5.*

Fig. 5.12 shows that the parameters estimated by the RAGMOD algorithm give significantly different values from those of Motion2D. Only after the refinement step in the Hybrid method do the results converge to the frame-by-frame method. Looking at the mean square DFDs for the sequence in Fig. 5.10, it can be seen that the Motion 2D algorithm gives a lower error than the hybrid estimation for the estimation between I-frames 4 and 5, despite the motion-based segmentation being used to weight the DFDs. This error is also apparent in the mosaic for this sequence (shown in Fig. 5.13 by the orange circles), where it can be seen that the cathedral



(a) MPEG motion based segmentation (b) 4th I-frame

Figure 5.11: *The motion-based background segmentation of I-frame 4. In some cases, the the quality of the motion-based segmentation is not sufficient. Here only one part of the cathedral is segmented as background (the red segment), so the second estimation will ignore a large part of the background and accuracy will be affected.*

edges are not aligned correctly over the boundary between the frames.

This weakness is introduced by the RAGMOD algorithm and the motion-based segmentation. In all previously described cases, the global motion was not so complex, and the motion-based segmentation was relatively good. For I-frame 4, the RAGMOD estimation is not sufficiently accurate, and camera rotation and zoom are not correctly compensated to correctly segment translations. As a consequence, only one part of the cathedral is considered for global motion estimation (Fig. 5.11) and the precision is not at its maximum. Clearly the 4-parameter affine model used in the RAGMOD algorithm is not suitable to describe the global motion in this sequence. The dramatic camera-motion, and the small distance from the camera to the building, result in a more complex global motion than a 4 parameter model can describe. Therefore, using a 6-parameter affine model or an 8-parameter perspective model in the RAGMOD technique might improve the reliability of the segmentation.

It has already been shown that the motion-based segmentation is efficient for selecting background area while preserving all of the interesting edges. However, if global motion is too complex, the quality of the segmentation will be affected. Despite the fact that in this sequence the hybrid algorithm is only using a section of the available background in the gradient-based framework, it can still obtain a reasonable estimate and minimise the error introduced by the motion-based segmentation.

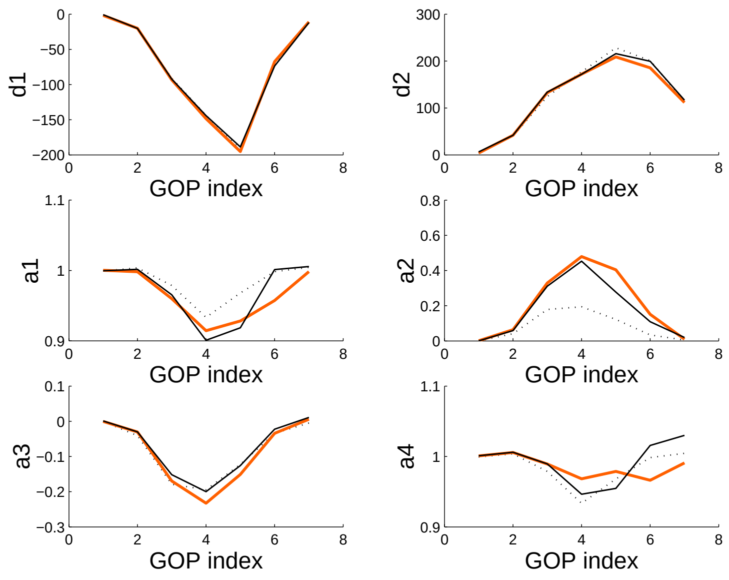


Figure 5.12: *Global motion estimation on "Cathedral" sequence. When affine motion is large, the RAGMOD parameters (dot lines) are not very accurate, but are well corrected by hybrid estimation (orange lines) which gives similar results to frame-to-frame Motion2D estimation (black lines).*

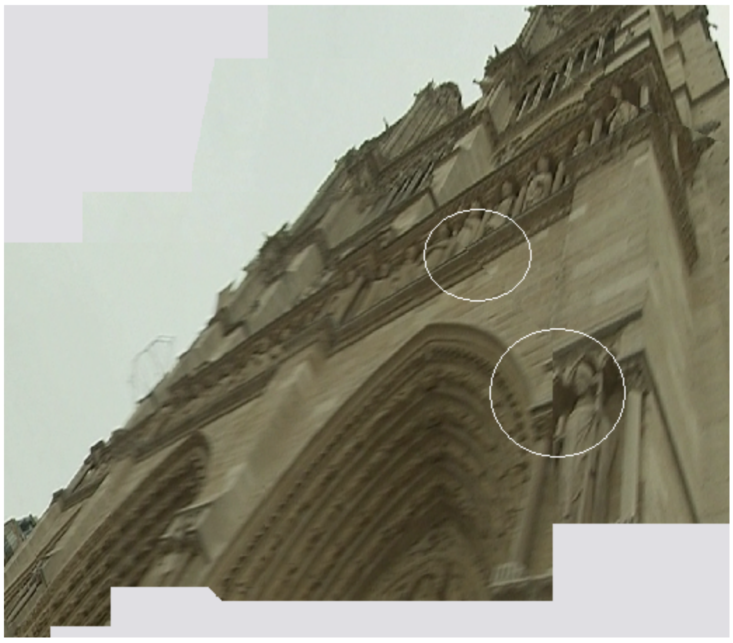
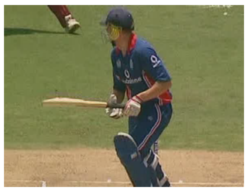
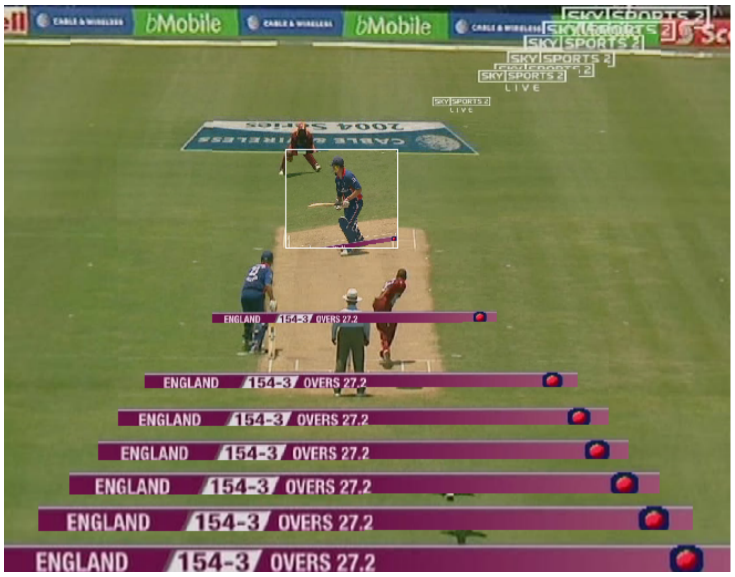


Figure 5.13: *A section of the "Cathedral" sequence mosaic created with Hybrid algorithm. The circles highlight the errors introduced into the mosaic by the poor estimates for GME given by the hybrid algorithm between I-frames 4 and 5. In the circles the edges jag as they are not correctly aligned over the boundary of the 2 frames.*

(a) First I-frame



(b) Final I-frame



(c) Mosaic

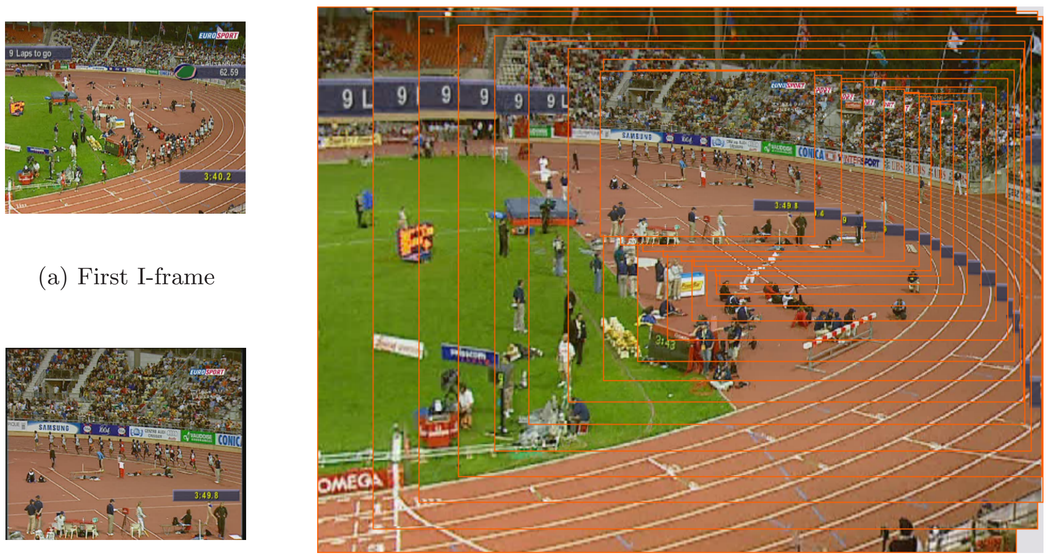
Figure 5.14: *Mosaic built on cricket sequence : In this sequence there is a strong zoom as the camera zooms-in to show a close-up of the batsman. The white box indicates the position of the last frame on the mosaic. In the last frame a false rotation is detected as the batsman is too prominent in the frame. See Fig. 5.19 for more details.*

5.2.3 Mosaicking of Sports Sequences

Mosaicking is a useful tool for summarisation of sports sequences. In many sports (*e.g.* football, cricket) the camera motion can be indicative of the the state of a game [59]. Typically, motion in sports sequences is faster than in the test sequences shown above and consists of combinations of panning and zoom. Figures 5.14 to 5.17 show examples of mosaics produced from footage of a number of sports events each with their own distinct camera motions.

The hybrid algorithm is very appropriate for this application, due to the fact that the broadcasted MPEG streams (DVB-S or DVB-T) hold motion compensation vectors of relatively high accuracy. Of course, this accuracy depends of the MPEG encoder. The broadcaster desires the best picture quality for a given limited bandwidth, and as a result high quality MPEG encoders are used. The use of such encoders results in higher quality motion fields which implies that lower error data rate is encoded. According to our experience, MPEG encoders used by major broadcasters deliver very good compensation information.

For all the mosaics shown in this section, the MPEG streams have been recorded from digital TV channels and are used as is. For the hybrid method, I-frames have been de-interlaced before being forwarded to the gradient-based refinement. As stated earlier, GME directly between I-frames is advantageous for moasaicking since not every frame needs to be included and it


(a) First I-frame

(b) Final I-frame

(c) Mosaic

Figure 5.15: *Athletics : In this sequence the camera zooms, pans and tilts as it tracks the athletes. For comprehension, a red frame frame is drawn around each compensated frame.*

reduces the computational load. However, in some circumstances it may be desirable to vary the length of the window over which GME is performed (*e.g.* if the motion is too fast/slow or for visualisation purposes) in order to optimise the visual quality of the mosaics. Furthermore, for the mosaics shown here, a primitive blending scheme is used, whereby every frame added to the mosaic overwrites the existing mosaic. Choosing a more suitable blending scheme would further improve the visual quality of the mosaics. The remainder of this section highlights various the various detected cases where the hybrid algorithm fails.

Lack of texture is a common occurrence in sequences acquired from some sports, like soccer, where the playing field has a largely homogeneous colour. In such scenes, often the only background features in a frame are the field markings (typically lines), which makes segmentation of these features crucial for robust GME. The motion-based segmentation has already been shown to be effective at correctly segmenting background edges. However, when there are not enough field markings present in the frame, the robustness of the hybrid algorithm will be reduced (See Figs. 5.17 and 5.18).

Another cause of failure in the hybrid algorithm manifests itself in the cricket mosaic (See Fig. 5.14). In this mosaic, a false global rotation is estimated in the final frame. Looking at Fig. 5.19 it can be seen that this error results from a failure in the motion-based segmentation. In the segmentation map shown, it can seen that the detected background largely consists of the batsman, an object undergoing a local motion. This results in the motion of the batsman being confused with the camera motion, causing the false rotation to be detected.

(a) First I-frame

(b) Final I-frame

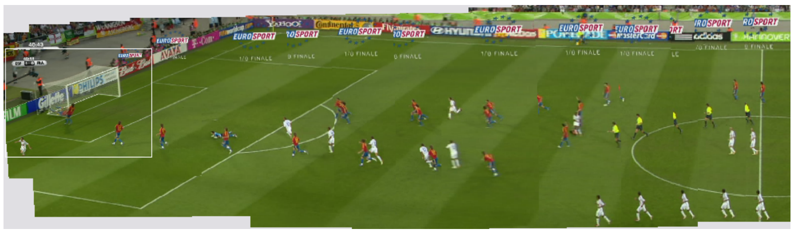

(c) Mosaic

Figure 5.16: *Mosaic built on a soccer sequence : The camera pans left and zooms in as it tracks the play. The white box indicates the position of the last frame in the mosaic.*

(a) First I-frame

(b) Final I-frame

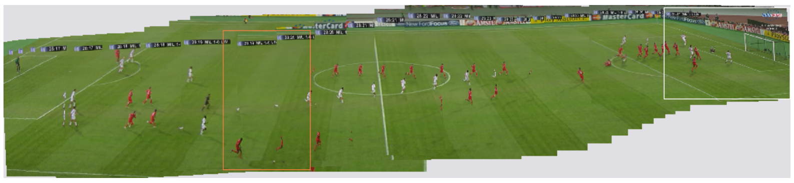
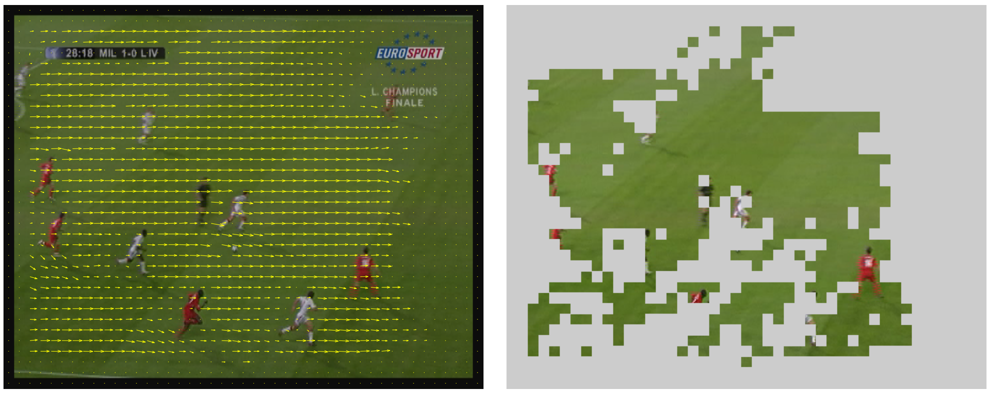

(c) Mosaic

Figure 5.17: *Mosaic built on a soccer sequence : Initially the camera pans left from the centre circle before panning right revealing the entire pitch. The white box shows the position of the last frame in the mosaic. The hybrid algorithm estimates the global motion accurately for most frames. However the orange box indicates an error in the mosaic caused by failure of the hybrid algorithm. This results in the remaining frames placed in the mosaic being tilted slightly upwards. See Fig. 5.18 for a more detailed description of the error.*

5.2.4 Computational Evaluation

The hybrid algorithm was implemented into the mosaicking interface developed by Renan Coudray. The four stages involved are the aggregation of the motion information over the GOP, the texture discrimination stage, the estimation of the motion parameters and motion based segmentation using the RAGMOD algorithm and refinement of the parameters using the modified IRLS framework. All test were run on a laptop with a 1.4 Ghz Pentium M processor and 1 GB of RAM. The first three stages take approximately 0.25 seconds per frame while the refinement stage takes about 0.065 seconds per iteration per frame when run on a sequence with a resolution of 720×576.

The computational complexity of the modified IRLS framework at each iteration consists mainly of performing motion compensation and of estimating the $G^T W G$ and $G^T W \mathbf{z}$ matrices. Motion compensation is performed using bilinear interpolation and must be preformed at each iteration on the both frame $n-1$ and also on the foreground/background segmentation map



(a) I-frame with superimposed motion vectors

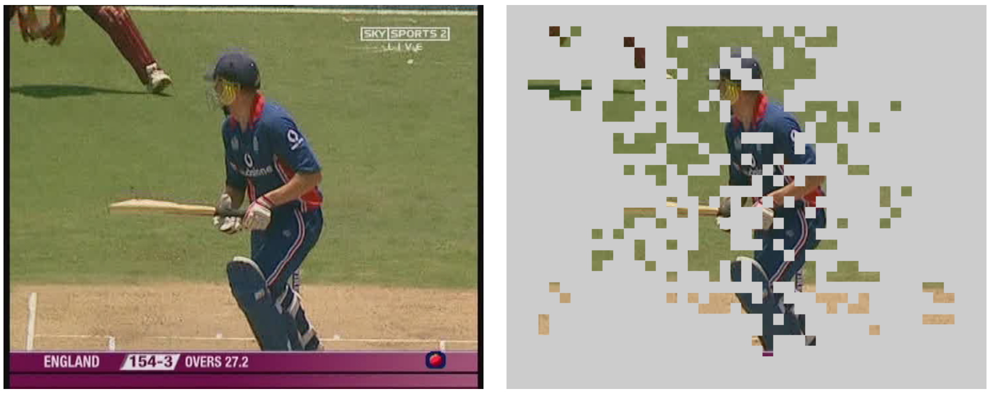
(b) Segmented background for I-frame

Figure 5.18: *This figure highlights an example when there is not enough background texture to allow for robust estimation of the parameters. In this sequence the background texture is largely determined by the field markings and the advertising hoardings. However, it can be seen from the segmented background that these features are not segmented, which results in an inaccurate global motion estimate for this frame. Additionally, it should be noted that there is a significant border at the margins of the frame which contains no motion information (shown as 0 vectors) and so are automatically excluded from the segmentation.*

---

of frame $n-1$. The Complexity is of $\mathcal{O}(\#$ number of pixels). Due to memory considerations, the G^TWG and $G^TW\mathbf{z}$ matrices (sizes of 6×6 and 6×1) are computed directly rather than by first generating the matrices G and $\mathbf{z}$ and performing the appropriate operations. As the foreground pixels are given zero weights, they can be ignored when estimating the matrices. Hence, the computational complexity of estimating the G^TWG and $G^TW\mathbf{z}$ matrices is $\mathcal{O}(36 \times \#$ number of background pixels) and $\mathcal{O}(6 \times \#$ number of background pixels) respectively.

The parameters chosen as the final estimate are the parameters at the iteration of the IRLS framework which minimises the mean squared error of the weighted residual rather than the value of the parameters at convergence. In the tests it was found that these points do not typically coincide and, as a result, choosing the number of iterations is a difficult decision. In the examples outlined in this chapter a hundred iterations were performed, although, in most frames, the optimum estimate was found in the first ten iterations. A reasonable compromise would be about 20 iterations of the IRLS framework resulting in an expected computation time of 1.3 seconds per frame for the modified IRLS framework and 1.55 seconds for the complete hybrid algorithm. As the mosaicking algorithm requires global motion estimation over a twelve frame interval, the required computation time is for a real time implementation time is 0.48 seconds per frame if the frame rate is assumed to be 25 frames per second. A real time C++



(a) I-frame (b) Segmented background for I-frame

Figure 5.19: *In this example the failure occurs because the motion of the batsman is confused as the camera motion. Thus the batsman is detected as the background rather than the true background. This results in a false global rotation being detected (See Fig. 5.14)*

implementation of the mosaicking software could be realised in the short term future.

5.3 Final Remarks

In this chapter, a hybrid approach to Global Motion Estimation from MPEG-2 streams has been presented. Firstly, an initial global motion estimate is obtained between I-frames from the motion information in the MPEG-2 stream using the RAGMOD algorithm and is refined using a gradient-based GME technique. The hybrid algorithm uses a motion-based segmentation to weight out local motion sites from the estimation, unlike previous gradient-based algorithms which rely on a residual-based weighting. It has been shown that although the motion-based segmentation is less precise than residual-based segmentations, it is better at correctly segmenting background edges which are important features for robust GME. It is this, along with the fact that the motion-based segmentation's robustness is independent of the accuracy of the current Global Motion Estimate, that makes the hybrid GME algorithm more robust than existing gradient-based approaches for the equivalent precision. Another advantageous feature of the hybrid algorithm is its ability to robustly and precisely estimate the camera motion directly over a larger temporal window (a typical window of 12 and 15 frames for the sequences shown here). Further work in this area would focus on making the hybrid algorithm more robust in cases where insufficient background texture is present and in cases where a foreground object provides the most prominent motion in an image.

6 Global Motion Estimation in Restoration: Automated Film Tear Restoration

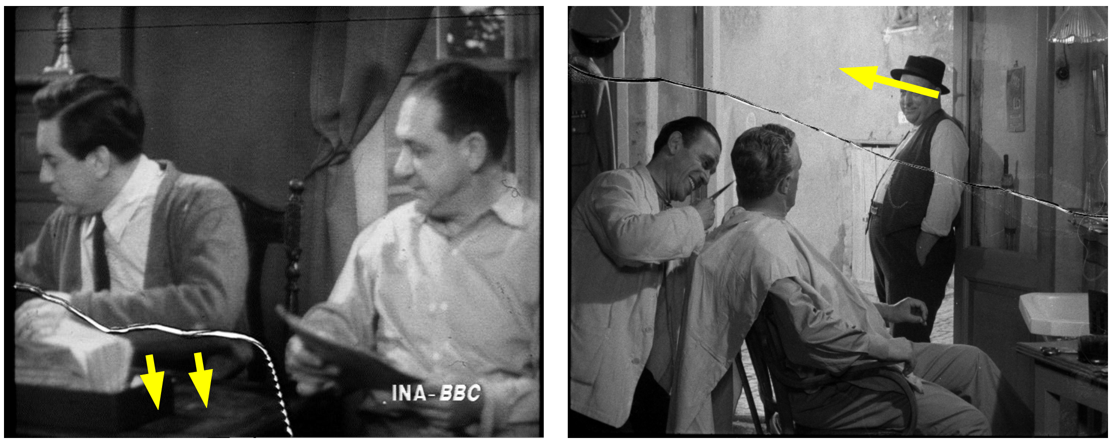
In the previous chapter, Global Motion Estimation is used in a mosaicking application. GME is also commonly used in restoration applications, most notably in stabilisation. In stabilisation, unintentional camera motion is eliminated by either using a low-pass filter on the global motion parameters of the sequence to smoothen the motion trajectory of the camera or by locking the camera to a fixed position [27]. This chapter is concerned with another restoration application, namely Film-Tear Restoration, which uses GME to digitally restore degraded frames.

Film tear is a common form of degradation in archived media. It is caused by the physical ripping of the film material, dividing the film in two. The two main problems resulting from film tear are

1. Relative displacements of the film sections either side of the tear (See Fig. 6.1). Looking at the local motion fields of each region (Fig. 6.2), it can clearly be seen that their distributions are significantly different. Also the edges crossing the tear boundary are no longer aligned (Fig. 6.1 (b)).

2. Destruction of Image Data along the tear boundary (Fig. 6.1). As the film has been torn, the film material along the boundary of the tear is deformed and so the image data in these areas is damaged. This damaged data can be considered as missing data.

This work has been published in part in [20].



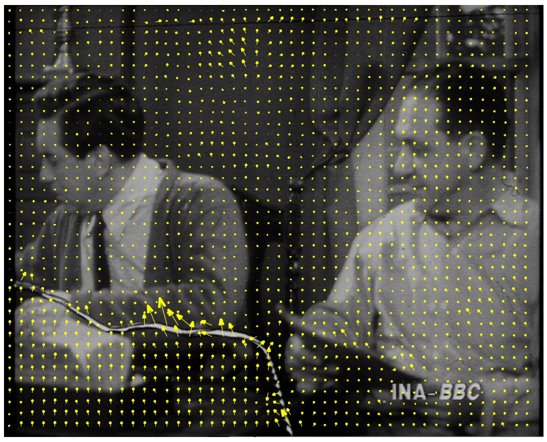
(a) Frame from "BBC" sequence (b) Frame from "scissors" sequence

Figure 6.1: *This figure shows an example of two real tears. The arrows indicate the relative displacement of the regions either side of the tear in each image. The image data along the edge of the tear has been destroyed.*

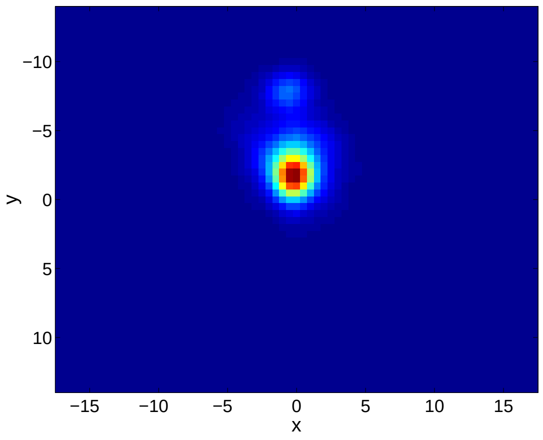
In order to restore a sequence degraded by tear, the relative displacement of the film must be corrected and the destroyed image data needs to be recovered. A general image-processing framework for restoring tear is given by the following stages:

1. Any torn frames which exist must be detected.
2. On torn frames the contour of the tear across the frame is delineated so that the frame can be divided into two regions.
3. The motion of each region is then estimated using a global motion estimator and the relative displacement between each region is corrected.
4. Finally a missing data treatment algorithm is applied to the torn frame to recover the damaged image data.

Until [21] no automated algorithm to digitally restore film tear had been proposed. Previously tear was restored by manually shifting the film sections into their correct position and using a transparent adhesive tape to fix the sections in place [88] (a process known as splicing) or by using image editing software to manually copy and paste one of the regions so that the edges align across the tear. Both these methods involve significant user-interaction and neither method can recover the damaged image data. [21] makes an attempt at producing an automated algorithm to restore film tear. It uses GME to estimate the dominant motion of the two regions separated by the tear boundary and compensates for tear by moving a region by the relative



(a) Torn frame from "BBC" sequence with superimposed motion vectors



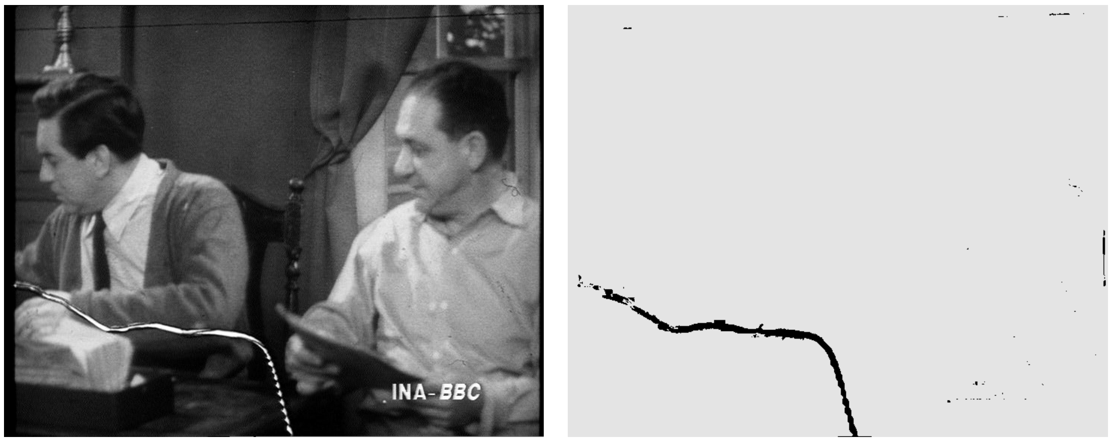
(b) Smoothened histogram of the motion field

Figure 6.2: *This figure demonstrates the different motion field distributions either side of the tear. This results in a histogram (right) which has 2 distinct peaks. A gradient-based motion estimator [56] is used to generate the motion field (shown left) which shows the relative position of each block in the previous frame.*

motion between the two regions. The translational motion of each region is estimated by choosing the modal local motion vector of each region. It then suggests that a missing data treatment algorithm be used to recover the damaged data once the displacement has been corrected.

However, no satisfactory method to divide the frame into two regions is proposed in that paper. It proposes a Bayesian framework to detect the tear boundary. Applying this algorithm to real torn sequences leaves gaps in the detected boundary (Fig. 6.3) and so is not sufficient to allow the frame to be divided into two regions. The contribution of this thesis is to introduce a new method of dividing torn frames which is fully automatic. The Graph-Cuts segmentation method proposed by Kolmogorov and Boykov in [12] is used to divide the frame in two. In [12] a field is divided into 2 segments given some initial hard-constraints and these constraints are derived from the local motion field of the torn frame. This technique has numerous advantages. As the problem is a 2-label problem, the graph-cuts technique will produce a global minimum energy solution [11]. Furthermore, the frame is divided directly, rather than detecting the tear boundary first as in [21].

The remaining sections of this chapter describes the novel frame division algorithm and outlines how the segmentations obtained from the algorithm can be used to restore torn frames . Subsequently, the results of experiments on both real and artificially torn sequences are presented. Section 6.6 describes how the tear division algorithm might be applied to dirty splice detection, before some final remarks are made.



(a) Torn frame (b) Binary result of tear delineation in [21]

Figure 6.3: *This figure shows the results of the tear delineation algorithm in [21] which detects the tear boundary (The black areas indicate the detected tear boundary). As there are gaps present in the detected tear boundary, the result is not sufficient to allow the frame to be divided.*

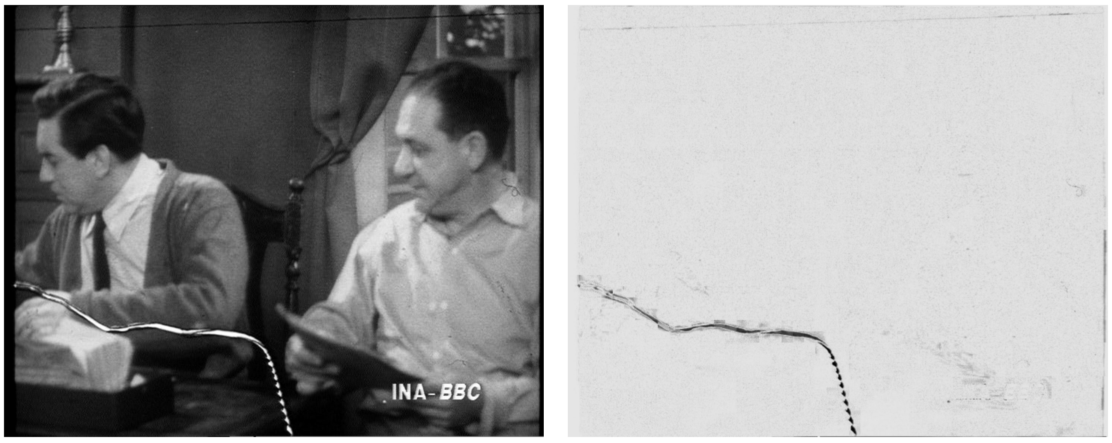
6.1 Tear Delineation

In order to compensate for the relative displacement between the 2 regions, an accurate segmentation of the frame is needed. In [21], it was proposed that the image be segmented by first detecting the tear boundary using a bayesian framework. The boundary is detected by looking for locations where the forward and backward Displaced Frame Differences (DFDs) are large. This comes from the observation that at the location of the tear boundary,there is a temporal impulse in the image intensity function similar to Missing Data or Pathological Motion.

This is demonstrated in Fig. 6.4 which shows the backward DFD for a torn frame. The large DFD values at the tear boundary are clearly visible. Intuitively the correct segmentation boundary can be considered to be the dark line in the DFD which follows the path of the tear. In other words, all pixels located on the same side of the ridge belong to the same region. Therefore, the DFD describes the boundary properties of the segmentation. Where the DFD is small, the local neighbourhood should all belong to the same region with the segmentation boundary coinciding with high DFD sites. The DFD is not used in this way in [21], which employs the DFD as the likelihood (in other words the region conditions) in a bayesian framework.

The use of the DFD as boundary information is the most novel feature of the segmentation algorithm outlined here. Typically, spatial gradient models are used as boundary information. However, while high gradient magnitudes are associated with the tear boundary, any other edge in the image will also have large gradients.

Sections 6.1.1 and 6.1.2 describe the background to some of the techniques used in the



(a) Torn frame (b) Backward DFD of Torn Frame

Figure 6.4: *The right image shows the absolute value of the DFD between the current (left image) and motion compensated previous frames. Dark intensities represent high DFD values.*

tear delineation algorithm. This is followed by a detailed description on how the delineation algorithm is implemented in practice.

6.1.1 Segmentation using the Graph-Cuts technique [11]

The Graph-Cuts technique is an optimisation strategy for minimising an energy function defined over an image. The main advantage of using Graph-Cuts technique over other optimisation strategies such as ICM [3] is that it finds the global minimum of the defined energy function for a two label problem.

The goal is to find the segmentation, l, that minimises some energy cost function given the initial conditions l_0. The problem here is to segment an image into two regions, therefore, the field l is a two-label field in which each pixel can have a value of 0 or 1. The value of l indicates the region the pixel belongs to. l_0 is a seeding function which indicates that certain pixels must belong to one region or another ($l_0 = 0$ or 1) while other pixels are classed as ambiguous ($l_0 \neq 0$ or 1). The cost functions typically used in Graph-Cuts are similar to those used in bayesian frameworks using MRFs (*e.g.* Chapter 2, Section 2.3.2). The cost function for the segmentation is described by

$$E(l) = \Lambda R(l) + B(l) \tag{6.1}$$

where

$$R(l) = \sum_{\mathbf{x}} R_{\mathbf{x}}(l(\mathbf{x})) \tag{6.2}$$

$$B(l) = \sum_{\mathbf{x}} \sum_{\mathbf{y} \in \mathcal{N}_s(\mathbf{x})} B_{\{\mathbf{x},\mathbf{y}\}} |l(\mathbf{x}) \neq l(\mathbf{y})| \tag{6.3}$$

In *Eq. 6.1*, $R(l)$ gives the likelihood (or regional) energy of a labelling of the image l and $B(l)$ gives the boundary cost of l, which describes the spatial smoothness energy of the segmentation. $R_{\mathbf{x}}(l(\mathbf{x}))$ is the region energy for each pixel and is the cost of assigning site $\mathbf{x}$ to a value of 0 or 1. The boundary term $B_{\{\mathbf{x},\mathbf{y}\}}$ is the cost of a discontinuity in the labelling between $\mathbf{x}$ and $\mathbf{y}$, where $\mathbf{y}$ is in the spatial neighbourhood of $\mathbf{x}$, $\mathcal{N}_s(\mathbf{x})$. The region energy bias Λ describes the weighting between the region and boundary terms.

In segmentation algorithms which use the graph cuts technique, the image is modelled as a Graph $\mathcal{G} = \langle \mathcal{V}, \mathcal{E} \rangle$ which is the set of nodes $\mathcal{V}$ and a set of edges $\mathcal{E}$. These graphs can either be directed or undirected graphs. Each pixel, $\mathbf{x}$, corresponds to a node which has an edge between itself and every pixel $\mathbf{y}$ in its spatial neighbourhood $\mathcal{N}_s(\mathbf{x})$ (called n-links). Every node also has two further links (called t-links) to a “source” node and a “sink” node which are associated with the two possible label values for each node (0 for the source and 1 for the sink). The edges in the graph are assigned a cost according to table 6.1. The cost of each n-link is related to the boundary energy between the two pixels connected to the n-link while the cost of the t-links is related to the energy of each pixel being assigned to either region.

Segmentation is performed by cutting a subset of the edges in the graph such that the graph is divided into two separate sub-graphs, one containing the source node and the other containing the sink. All pixels, $\mathbf{x}$, that belong to the the sub-graph containing the source are assigned $l(\mathbf{x}) = 0$. Likewise, pixels in the other sub-graph are assigned $l(\mathbf{x}) = 1$. The cost of the cut is given by the cost of all the n and t-links which are cut to divide the graph. [11] describes how the cut of minimum cost corresponds to the segmentation that minimises the cost function in Eq. 6.1. Thus the optimal segmentation for the image is given by the cut of minimum cost.

The tear segmentation model can be solved using this technique. In this application, the boundary energy (n-link) terms can be given by the size of the DFD. Since each region is undergoing a separate global motion, the region energy terms (t-links) of the graph can be described by the similarity of the motion at a site to the global motions of both regions.[1] The nearer the motion of a pixel is to the global motion of a region, the more likely the site is a member of that region. However one outstanding issue still remains, namely how to automatically produce the seed function l_0 to initialise the graph-cuts segmentation.

[1]Precise definitions of these energy terms will be given later in section 6.1.3.

edge	**cost**	**for**
n-link: $\{\mathbf{x}, \mathbf{y}\}$	$B_{\{\mathbf{x},\mathbf{y}\}}$	$\mathbf{y} \in \mathcal{N}_s(\mathbf{x})$
t-link to source	K	$l_0(\mathbf{x}) = 0$
	0	$l_0(\mathbf{x}) = 1$
	$\Lambda R_{\mathbf{x}}(0)$	$l_0(\mathbf{x}) \neq 0$ or 1
t-link to sink	K	$l_0(\mathbf{x}) = 1$
	0	$l_0(\mathbf{x}) = 0$
	$\Lambda R_{\mathbf{x}}(1)$	$l_0(\mathbf{x}) \neq 0$ or 1

Table 6.1: *The cost of each edge in the graph. K is some large number. Setting K large ensures that seeded sites cannot change label during segmentation.*

6.1.2 Initialising the Graph Cuts Segmentation

The segmentation is initialised by defining a seeding function in which pixels have one of 3 possible states. The pixels are marked as either belonging to one of the 2 required regions or as a pixel whose region is ambiguous. Only pixels which are certain to belong to a region should be labelled as such and it is imperative that no pixels are assigned to the wrong region since the labels of these pixels cannot be changed during the segmentation.

The seeding function in the tear delineation algorithm is derived from the vector field of the torn frame. Looking at Fig. 6.6(a), which shows the vector field superimposed on the torn frame, it can be seen that the motion either side of the tear is distributed differently. This difference is visible in the histogram of the motion field (Fig. 6.6(b)), in which two significant peaks are visible. The larger peak in the histogram corresponds to the motion on the top right side of the tear boundary, while the smaller peak corresponds to the motion in the bottom left corner of the image.

In appendix B, it was described how these histograms could be used to perform a coarse background segmentation using the watershed algorithm [26, 105]. Revisiting this idea, the watershed algorithm can also be used to segment the two main peaks in the motion field histogram (Fig. 6.6(c)). Like the motion-based background segmentation in [26], the segmentation can be generated by selecting which motion vectors contribute to the two segmented peaks. This allows a seeding function to be generated for the image, with blocks either contributing to one of the segmented peaks or to other less significant peaks (*i.e.* for ambiguous pixels) (Fig. 6.6(d)).

The image shown in (Fig. 6.6(d)) gives a coarse seeding function for the image. The segments shown in this map correspond roughly to the perception of what the regions should be. However there are a number of blocks which have been assigned to the wrong region. Furthermore, the true boundary (along the tear) between the two regions is coarsely and imprecisely defined and since graph-cuts will not change these labels, it is necessary to create a gap between the two regions so that the graph-cuts segmentation can define the boundary precisely. Therefore, an erosion operation is performed on the seeding function which removes the labels from blocks whose neighbours do not all have the same label. Fig. 6.6(e) shows the seeding function after the erosion operation applied. This map is then used to initialise the graph-cuts segmentation which generates the final segmentation shown in Fig. 6.6(f).

6.1.3 Algorithm Outline

Fig. 6.6(a) shows a flowchart of the tear delineation algorithm. Before the algorithm can proceed local motion estimation needs to be performed on the torn frame to generate the motion field which is to be included. The gradient-based motion estimator described in [4, 54] is used in the examples shown the chapter. This motion estimator employs a block size of 17×17 pixels with a 2 pixel overlap, resulting in a coarse representation of the motion information. Two motion fields are estimated per frame, one for motion in the current frame with respect to the previous frame and one for motion with respect to the next frame.

Motion Field Accumulation

The main consideration is to compute a histogram whose two highest local maxima correspond to the global translational motion of the two regions. The histogram is computed directly from the motion field supplied. The choice of the histogram bin width is important in computing the histogram. A relatively large bin width should be chosen to prevent multiple local maxima being associated with the motion of one region. A typical value used is 0.5 pel in the vertical and horizontal directions. After the histogram is computed, it filtered with a low-pass gaussian filter. The gaussian kernel has a size of 15×15 and a variance of 2.5.

Watershed Segmentation

The watershed algorithm is applied on the filtered histogram. As the standard implementation of the watershed algorithm [105] is to segment troughs rather than peaks, the histogram is first subtracted from its maximum before the watershed algorithm is applied. After the watershed algorithm is applied, the two most significant peaks are isolated. This can be decided by a number of criteria, including the size of the maxima (*i.e.* choosing the two largest peaks) or the number of vectors contributing to a segment (*i.e.* choosing the two peaks greatest number of vectors). In this implementation the size of the maxima is chosen as the criterion for selecting the segments. Once the two segments have been isolated, all vector blocks in the motion field are labelled as belonging to their respective segments or are marked as ambiguous if they belong to other peaks in the histogram.

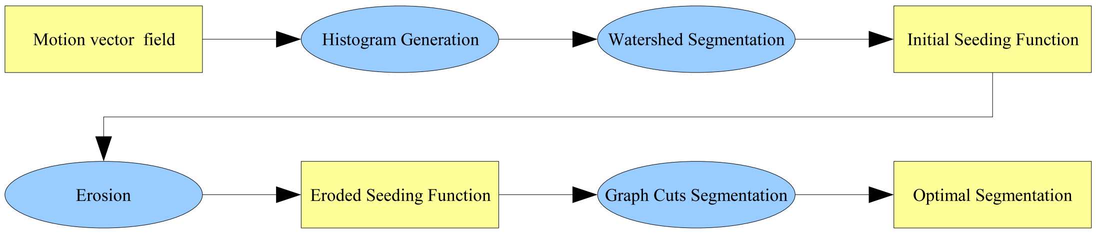
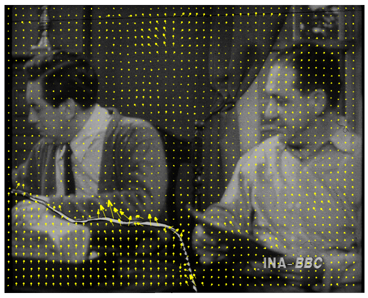
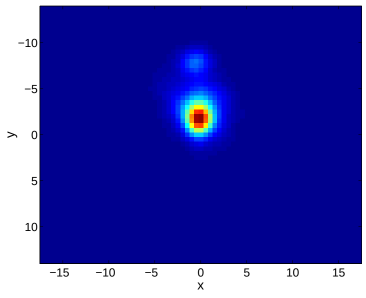


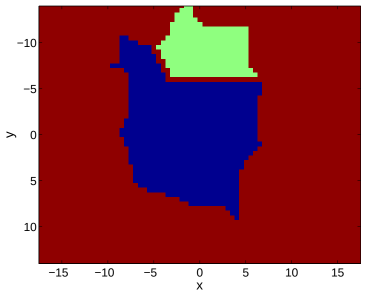
Figure 6.5: *The tear delineation algorithm*



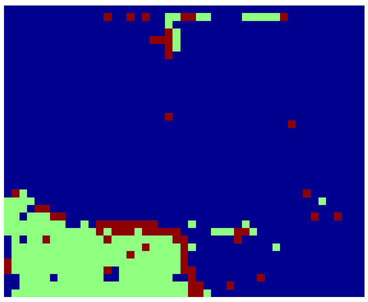
(a) Motion Field of Torn frame



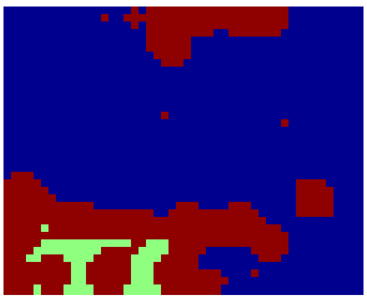
(b) 2D Histogram of the Motion Field



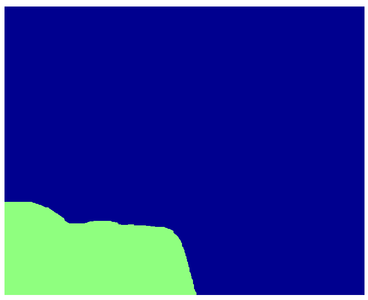
(c) First two histogram peaks segmented by watershed algorithm



(d) Segmented Image after Watershed Segmentation



(e) Segmented Image after Erosion



(f) Segmentation after Graph-Cuts

Figure 6.6: *This figure shows the results of each stage of the frame division algorithm for the torn frame shown in (b). In figures (d-g), the blue and green colours represent vectors (d), vector blocks (e-f) or pixels (g) belonging to the two desired regions, while the red colour represents unlabelled pixel sites.*

Erosion

The erosion operation is applied on the initial seeding function in two stages. The first erosion stage is mandatory as a band of unassigned blocks needs to be created at the tear boundary, so as to let the boundary be defined precisely. This band is created by marking blocks which have neighbour any blocks belonging to the other region as ambiguous. The second erosion stage is optional and involves changing the labels of blocks with neighbours of different labels (*i.e.* either blocks belonging to the other region or ambiguous blocks) to ambiguous. This stage can be iterated a number of times at the users discretion. If too few iterations are performed (or none at all), some blocks may be assigned to the wrong region, while choosing too many iterations would shorten the perimeter of the seeded regions to the extent where "shrinking" [10, 11, 64] occurs in the graph cuts segmentation. Typically one iteration of the second erosion stage is appropriate for motion fields generated from images of 720×576 pixels.

Graph-Cuts Segmentation

Before running the graph-cuts algorithm, the seeding function is upsampled to the full image resolution. Furthermore, an energy function of the form in *Eq. 6.1* is defined as follows:

- **The Region Term**, $R_{\mathbf{x}}(l(\mathbf{x}))$, for a site, $\mathbf{x}$, is associated with the distance between the actual motion at $\mathbf{x}$ and the global motion of each of the two regions. The global motion of each region is given by the modal vector of the portion motion-field histogram corresponding to the region. If the motion between the current frame n and a neighbouring frame k at $\mathbf{x}$ is given by $\mathbf{d}_{n,k}(\mathbf{x})$ then the region term is given by

$$\begin{aligned} R_{\mathbf{x}}\big(l(\mathbf{x}) = 0\big) &= ||\mathbf{d}_{n,k}(\mathbf{x}) - \mathbf{d}_g(l(\mathbf{x}) = 0)|| \\ R_{\mathbf{x}}\big(l(\mathbf{x}) = 1\big) &= ||\mathbf{d}_{n,k}(\mathbf{x}) - \mathbf{d}_g(l(\mathbf{x}) = 1)|| \end{aligned} \tag{6.4}$$

  where $\mathbf{d}_g(l(\mathbf{x}))$ is the global motion of a given region and $||.||$ is the euclidean distance metric. As $R_{\mathbf{x}}(l(\mathbf{x}))$ gives the cost of assigning $\mathbf{x}$ to label $l(\mathbf{x})$, the local motion of each region is constrained to be close to its global motion.

- **The Boundary term**, $B_{\{\mathbf{x},\mathbf{y}\}}$, is given by the size of the DFD. If the DFD is given by

$$\Delta_{n,k} = I_n(\mathbf{x}) - I_k(\mathbf{x} + \mathbf{v}_{d,k}(\mathbf{x}))$$

  where I_n and I_k are the intensity functions of the current and neighbouring frames, then the boundary term is as follows

$$B_{\{\mathbf{x},\mathbf{y}\}} = \max\{||\Delta_{n,k}(\mathbf{x})|\} - \frac{1}{2}\big(|\Delta_{n,k}(\mathbf{x})| + |\Delta_{n,k}(\mathbf{y})|\big). \tag{6.5}$$

  which is the difference between the maximum absolute value of the DFD in the image and the mean absolute DFD of the pixels $\mathbf{x}$ and $\mathbf{y}$. This term is designed to penalise discontinuities in the segmented image when the mean absolute DFD of the neighouring sites is small and encourages discontinuity when the mean is large.

The algorithm proceeds by setting up a graph with edge costs given in table 6.1. A 4 connected neighbourhood is used for the graph. The max-flow/min-cut algorithm presented in [12] by Boykov and Kolmogorov is used to find the segmentation. At the most basic level max-flow algorithms work by modeling the graph as a network flow, with the link energies corresponding to a flow capacity. The algorithm proceeds by finding a non-saturated path between the source and sink nodes on the graph and pushing a flow down the path until an edge on the path becomes saturated (*i.e.* The edge is cut). This process is repeated until no more paths can be found. The Boykov and Kolmogorov algorithm finds source/sink paths by building 2 search trees, with routes at the source and sink nodes, using a breadth first search. The theoretical upper bound on the computational complexity of the algorithm is $\mathcal{O}(mn^2|C|)$, where m is the number of edges on the graph, n is the number of nodes and $|C|$ is cost of the minimum cut. The complexity also depends on the length of the paths as it takes more time to find the saturated edge on a longer path, and on the number of paths found before the cut has been completed.

6.2 Displacement Correction [21]

The goal is to estimate and correct the relative displacement between the two segmented regions. The tear displacement model is defined by

$$\mathbf{d}_{n,k}(\mathbf{x}) = \mathbf{d}^l_{n,k}(\mathbf{x}) + \mathbf{d}^g_{n,k} + \mathbf{d}_t(l(\mathbf{x})) \tag{6.6}$$

where $\mathbf{d}_{n,k}(\mathbf{x})$ is the observed motion vector at $\mathbf{x}$, $\mathbf{d}^l_{n,k}(\mathbf{x})$ is the foreground motion at the current location, $\mathbf{d}^g_{n,k}$ is the true global motion of the frame[2] and $\mathbf{d}_t(l(\mathbf{x}))$ is the displacement[3] resulting from the tear. Therefore, the observed global motion is given by

$$\mathbf{d}_g(l(\mathbf{x})) = \mathbf{d}^g_{n,k} + \mathbf{d}_t(l(\mathbf{x})). \tag{6.7}$$

It follows that the relative tear displacement,

$$\begin{aligned}\mathbf{d}_{rel} &= \mathbf{d}_t(l(\mathbf{x}) = 1) - \mathbf{d}_t(l(\mathbf{x}) = 0) \\ &= \mathbf{d}_g(l(\mathbf{x}) = 1) - \mathbf{d}_g(l(\mathbf{x}) = 0). \end{aligned} \tag{6.8}$$

Hence, the relative tear displacement can be calculated by estimating the global translational motion of each region.

Histograms of the motion field for each region are used to calculate its global motion. The advantage[4] of using histograms is that such methods are more robust to local motion than IRLS methods. Furthermore, using the vector-field is more computationally efficient than using

[2]For simplicity it is assumed to be purely translational.

[3]Again assumed to be purely translational.

[4]See chapter 4 for a more detailed discussion on the merits of different motion estimators.

intensity based GME algorithms. However, there is a trade off in the precision of the estimate, which is constrained by the bin width of the histogram and the precision of the motion field.

The histograms for each region are computed from the motion field and the segmentation map given by the tear delineation algorithm. The precision of the estimate is more important here than in the tear delineation stage and so a smaller bin width is used. A typical size is 0.125 pel. In order to discriminate against vectors calculated from ill-suited image data, each vector is assigned a weight based on the texture information of the block. The weighting discriminates against blocks with flat or noisy textures and encourages the motion vector to be similar in phase to the spatial gradient to mitigate against the aperture effect. The expression for the weight at a pixel $\mathbf{x}$ belonging to a block $\mathcal{B}$ is given by

$$w(\mathbf{x}) = w_g(\mathbf{x}) \times \left|\cos\Big(\angle\big(\mathbf{d}_{n,k}(\mathbf{x})\big) - \angle\vec{\mu}_g(\mathbf{x})\big)\Big)\right| \tag{6.9}$$

where

$$\vec{\mu}_g(\mathbf{x}) = \frac{\sum_{\mathbf{y}\in\mathcal{B}} \nabla(\mathbf{y})}{\sum_{\mathbf{y}\in\mathcal{B}} 1} \tag{6.10}$$

is the mean gradient vector of the block $\mathcal{B}$ and where

$$w_g(\mathbf{x}) = \begin{cases} 1 & \|\vec{\mu}_g(\mathbf{x})\| > g_t \\ 0 & \text{otherwise} \end{cases}. \tag{6.11}$$

g_t is a threshold (a typical value of g_t is 2). $w(\mathbf{x})$ consists of two components. The first component $w_g(\mathbf{x})$ weights out blocks with flat or noisy textures since the magnitude of the mean gradient vector will be small. The second component assigns low weights to motion vectors which are approximately perpendicular to the mean gradient vector and high weights when the vectors are parallel.

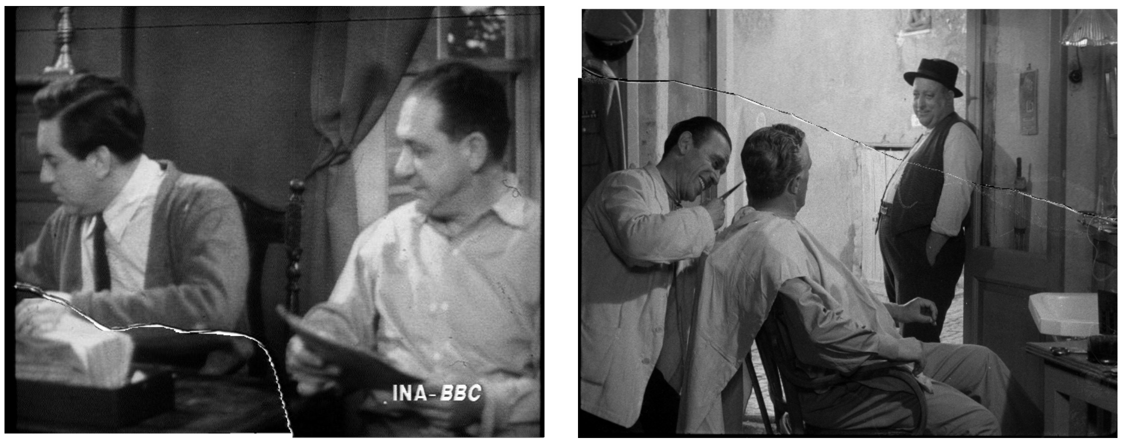
The global motion vector of the regions is given by the mode of their histograms. The tear displacement is corrected by moving one of the regions by the relative displacement $\mathbf{d}_{rel}$. Examples of frames with corrected displacements are shown in Fig. 6.7.

6.3 Frame Restoration

The final task in tear restoration process is to recover the image data damaged by the tearing of the film. This is a conventional missing data problem, as the true image data can be said to have been replaced by some random replacement noise. Therefore, any conventional missing data treatment algorithm can be used to recover the damaged data. The missing data treatment algorithm outlined in [60] is an example of such an algorithm which also acts as a noise reducer.

6.4 Results

The two fundamental components of the tear restoration process are the delineation and displacement estimation process. Accurate results for both processes are necessary for the good



(a) "BBC" Sequence (b) "Scissors" Sequence

Figure 6.7: *This figure shows two examples of torn frames compensated for tear displacement.*

performance of the displacement correction stage. A poor segmentation would cause parts of the frame to be wrongly compensated for (or not) and a poor displacement estimation would leave a visible displacement of a region after the displacement correction.

This section consists of two main parts. Firstly, the results of a ground truth are outlined which assess the performance of the algorithm. The second major part of the section tests the algorithm on real sequences damaged by tear.

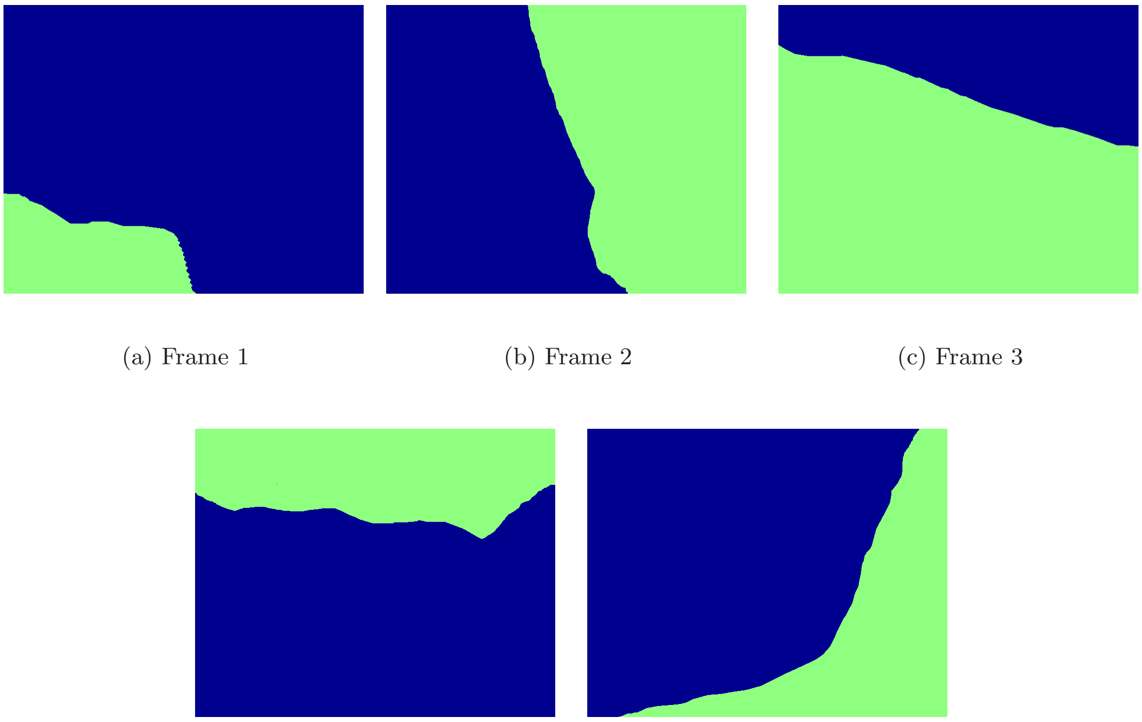
6.4.1 Ground Truth Experiments

In order to evaluate the performance of the process, a ground truth is needed. This is created by digitally "tearing" a sequence by displacing a region of a number of frames by a pre-defined displacement using a pre-defined segmentation mask. The results for two test sequences are presented in this section.

1. The first sequence tested is the "mobile and calendar" sequence previously discussed in Chapter 5. This sequence is a relatively clean sequence, without any missing data or significant noise, and has plenty of texture detail in the foreground and background. It also contains relatively large objects (the calendar, the train) undergoing local motion. Because of these properties, it is a relatively easy sequence for motion estimation.

2. The second sequence is the "jumper" sequence which is a more difficult sequence for motion estimation. The sequence is badly degraded, with high noise levels and a large amount of missing data. Furthermore, the motion of the jumper is pathological.

Five non-consecutive frames from each sequence are selected. Every frame is then split into two regions using pre-defined masks and a region is then displaced by the set displacement. A



(a) Frame 1 (b) Frame 2 (c) Frame 3

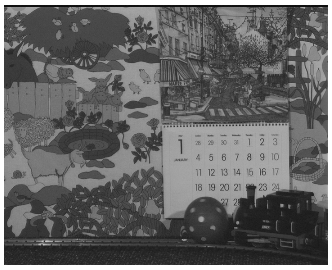
(d) Frame 4 (e) Frame 5

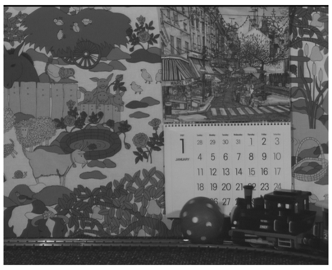
Figure 6.8: *This figure shows the pre-defined masks for the 5 selected frames in ground truth sequences. The blue and green colours represent the two regions, and the red colours represents pixel whose region is unknown.*

Frame	Horizontal Displacement	Vertical displacement
1	-2	7
2	5.439	-0.872
3	-1.345	6.254
4	0.1	-7.872
5	4.567	5.212

Table 6.2: *This table shows the displacements for each of the "torn" frames in the ground truth sequences. The region of the frame labelled green in the masks in Fig. 6.8 is the region that is displaced.*

table of each frame and their displacements is given in table 6.2 and the masks are shown in Fig. 6.8. For simplicity of implementation and comparison, the respective displacements and masks are the same for both ground truth sequences. The frames from the test sequences and their torn versions are shown in Figures 6.9 and 6.10.

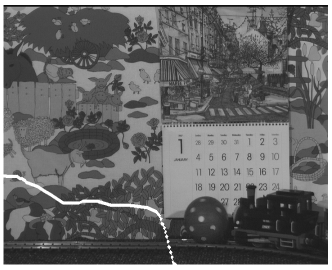

(a) Frame 1

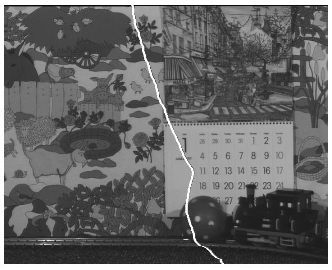

(b) Frame 2

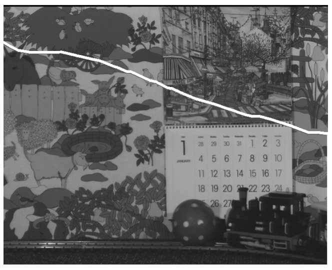
(c) Frame 3

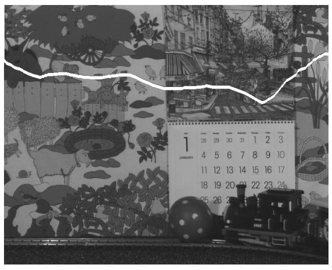
(d) Frame 4

(e) Frame 5


(f) Torn Frame 1


(g) Torn Frame 2


(h) Torn Frame 3


(i) Torn Frame 4

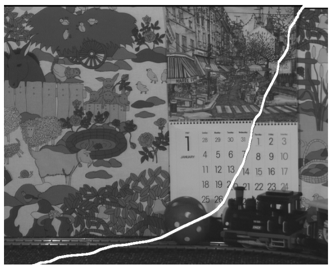
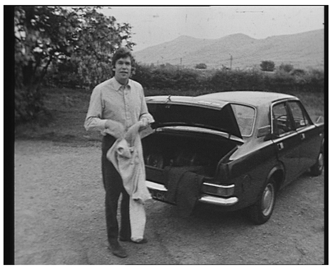

(j) Torn Frame 5

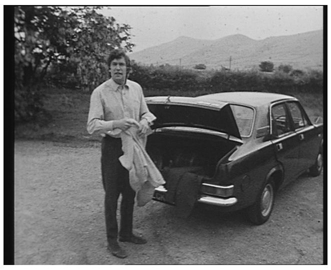
Figure 6.9: *The torn frames of the mobile and calendar sequence.*

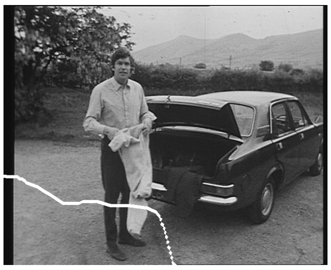
(a) Frame 1

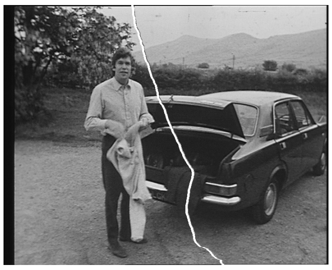

(b) Frame 2

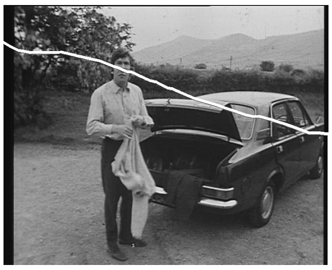
(c) Frame 3

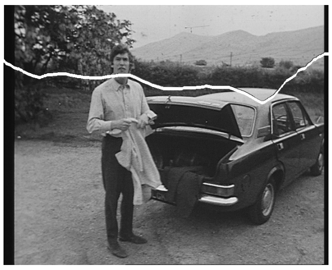
(d) Frame 4


(e) Frame 5


(f) Torn Frame 1


(g) Torn Frame 2


(h) Torn Frame 3


(i) Torn Frame 4

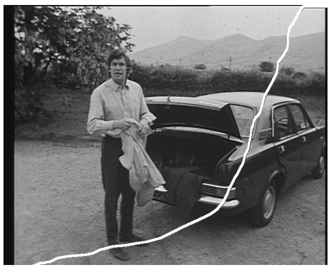
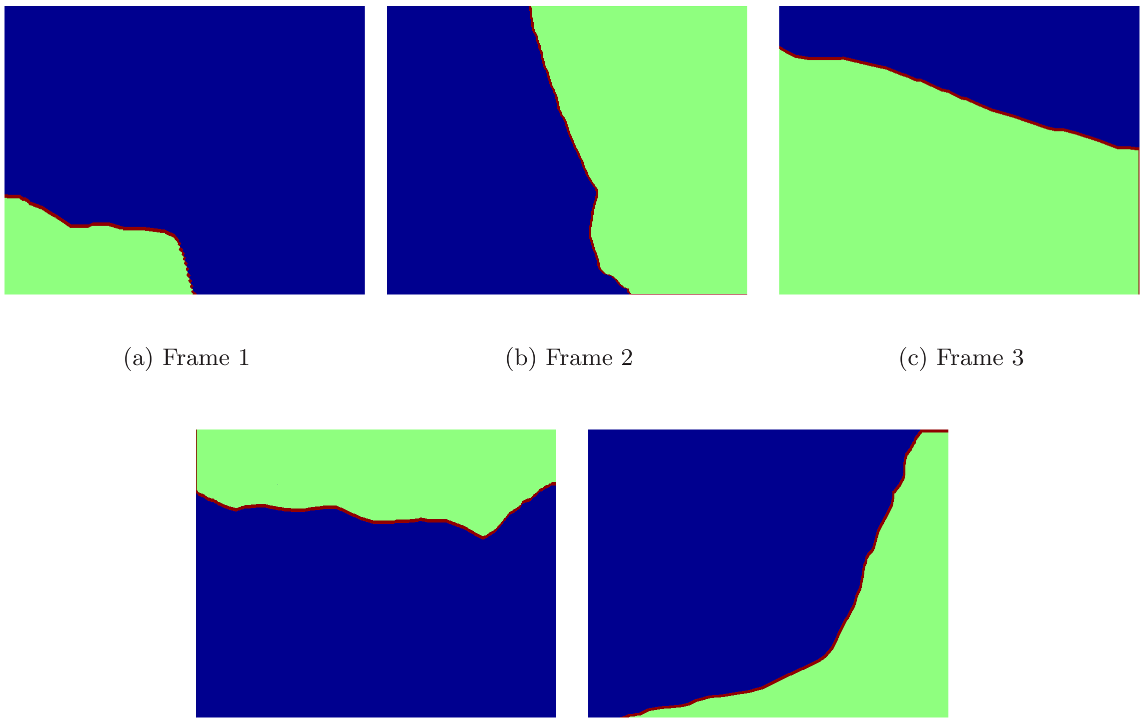

(j) Torn Frame 5

Figure 6.10: *The torn frames of the jumper sequence.*



(a) Frame 1 (b) Frame 2 (c) Frame 3

(d) Frame 4 (e) Frame 5

Figure 6.11: *This figure shows the tri-maps for the 5 selected frames in ground truth sequences. The red region represents the tear boundary where there is no preference as to which region the pixels belongs.*

It is easy to quantify the accuracy of the estimated displacement as it can be compared to the ground truth displacement. However, quantifying the performance of the delineation algorithm is not as straightforward. The main constraint on the segmentation is that all pixels on the same side of the tear boundary should belong to the same region. However, the tear boundary has a width and is not a sharp transition between the two regions. Therefore, any segmentation boundary which is located within the tear boundary will satisfy the constraint. As a result, there are three classes of pixel in a torn image. Two classes are for pixels which lie outside the tear boundary and belong to either region and a third which describes pixels in the tear boundary which could belong to both regions. For the purposes of this segmentation, the tear boundary is assigned to a third region, labeled in Fig. 6.11 by the red colour. Pixels with this label can belong to either region in the final segmentation.

6.4.2 Tear Delineation

There are two parameters in the tear delineation process which impact on the accuracy of the segmentation as well as the computational overhead of the algorithm. These parameters are,

1. The Region Energy Bias (Λ) - which is the weighting factor between region and boundary information in the graph-cuts framework (*Eq. 6.1*) and is a non-negative real number. A value of 0 excludes region information from the segmentation, while a large value biases the segmentation towards the region information (*i.e.* the likelihood).

2. The number of erosion iterations - the number of times the second erosion stage is iterated (See section 6.1.3) and is a non-negative integer. If the number is too small, then some pixels might be assigned to the wrong region. If the number is too large, then "shrinking" [10, 11, 64] might occur.

Notes on the Experimental Setup

- For this experiment, the tear restoration algorithm was implemented in Matlab. The graph cuts implementation used for delineation was adapted from the C++ implementation [9] of the max-flow/min-cut algorithm presented by Boykov and Kolmogorov in [12] and was incorporated into the matlab code. All other sections of the algorithm were implemented directly in Matlab. The algorithm was tested on a PC with a 1.4 GHz Pentium M processor with 1 GB of RAM.

- Both test sequences have a resolution of 720×576 and contain 8-bit grayscale images.

- The gradient-based motion estimator described in [4, 54] is used to generate the motion field for accumulation. The motion estimator produces one motion vector block of pixels with a block size of 17×17 pixels with a horizontal and vertical overlap of 2 pixels. As the resolution of the test sequences is 720×576 pixels, the resolution of the vector field is 47×38 blocks. The backward motion field (*i.e.* The vectors shows relative position of each block in the previous frame.) is used for the accumulation.

- The quality measurement used in the experiments is given by

$$\text{quality (\%)} = \left(\frac{\text{no. correct detections in region 1}}{\text{true size of region 1}} + \frac{\text{no. corr. detect. in reg. 2}}{\text{true size of reg. 2}} - 1 \right) * 100 \tag{6.12}$$

  This value ranges from 0 to 100. The quality is 0 if the labels in the segmentation are evenly distributed in the two ground truth regions and is 100 if the the labels agree exactly with the ground truth. The computation time is measured by using the "tic" and "toc" functions in Matlab to measure the time it takes to execute the algorithmic code.

edge	cost	for
n-link: $\{\mathbf{x}, \mathbf{y}\}$	$B_{\{\mathbf{x},\mathbf{y}\}}$	$\mathbf{y} \in \mathcal{N}_s(\mathbf{x})$
t-link to source	$\Lambda R_{\mathbf{x}}(0)$	$\forall \mathbf{x}$
t-link to sink	$\Lambda R_{\mathbf{x}}(1)$	$\forall \mathbf{x}$

Table 6.3: *The cost of each edge in the graph when segmenting without initialisation.*

- The values of the region energy bias , Λ, used in the experiment are

  $$\Lambda = \{0.001, 0.005, 0.01, 0.05, 0.1, 0.5, 1, 5, 10, 50, 100\} \tag{6.13}$$

  A significant feature of the implementation of the graph-cuts at [9] is that the energy terms are implemented as integers rather than floating point numbers. This requires that the energy terms are scaled to some maximum value. The value chosen for this is 255 which is the maximum value of the DFD in an 8-bit grayscale image and hence is the maximum value of the boundary energy term. As Λ represents the influence of the region energy term in the segmentation framework, any value of Λ below the critical value of $\frac{0.5}{255} \approx 0.002$ is equivalent to a value of 0. Hence, the region energy is 0 and the segmentation relies entirely on the initialisation and the boundary terms.

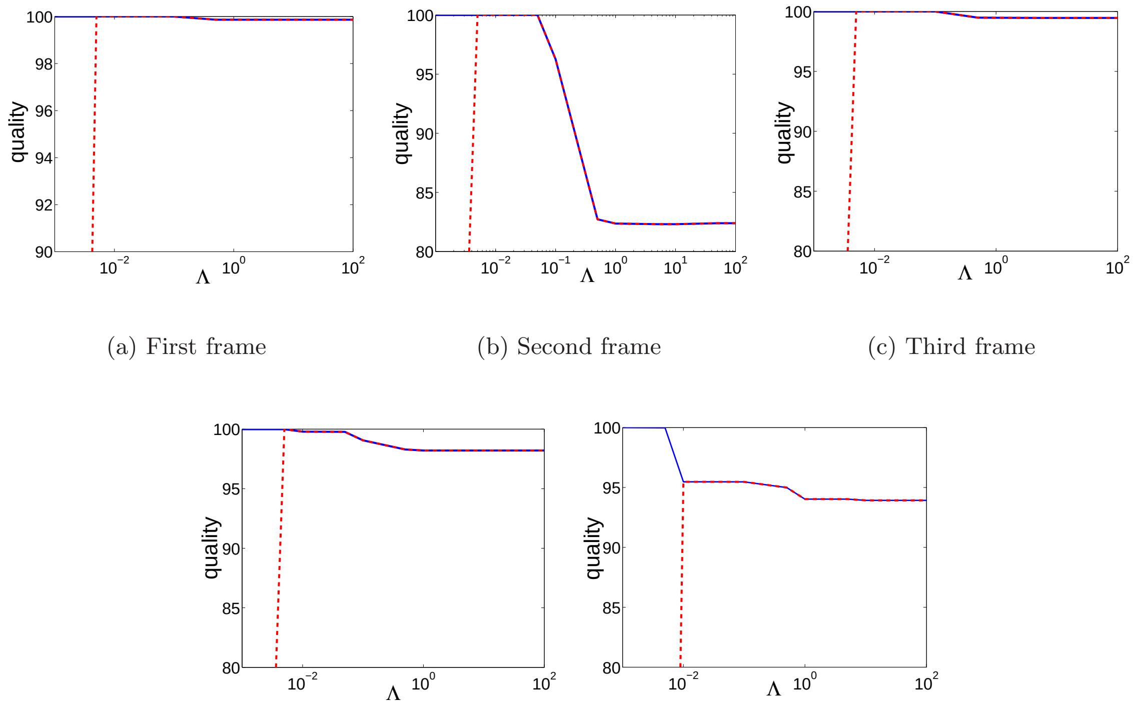
- When varying the region bias, quality and computation measurements are also taken for a segmentation which does not use the seeding function to initialise the graph-cut segmentation. In effect, all pixels are considered to belong to the ambiguous state. Table 6.3 shows the configuration of the graph used when not initialising the segmentation.

- The integer values between 0 and 20 are chosen for the erosion parameter and represents the number of iterations performed in the second erosion stage. The experiment is performed in the case where no region information is included (*i.e.* $\Lambda = 0$) and when region information is included.

The Region Energy Bias, Λ

Fig. 6.12 gives a typical example of how varying Λ affects the quality of the segmentation. Each graph consists of two plots, for segmentations generated with (blue -) and without (red - -) initialisation by the seeding function.

Looking at the quality measurements, it can be seen that when Λ is small the quality for the segmentation without seeding is 0 and this occurs because Λ is below the critical limit. For values of Λ greater than the critical value, the qualities of the segmentations are similar. For smaller values of Λ, in this case between 0.005 and 0.01, the segmentation is at its optimum (Fig. 6.13 (b)). However, as Λ increases and the region information has a greater weighting on the segmentation, there is a slight drop off in the quality of the segmentation. Since a block-based



(a) First frame (b) Second frame (c) Third frame

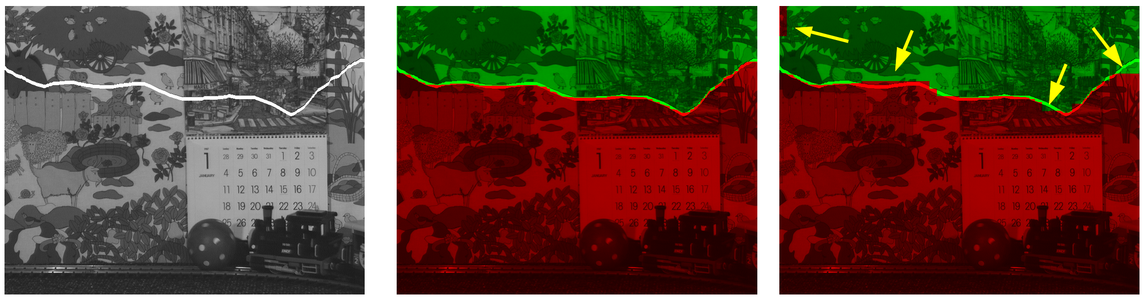
(d) Fourth frame (e) Fifth frame

Figure 6.12: *This figure shows plots of the segmentation quality against Λ for the five torn frames of the mobile and calendar sequence. The blue line marks the plot with initialisation of the graph cuts algorithm and the dashed red line represents the measurements of the segmentation without initialisation.*

motion field [54] is used, the region energy is also defined on a block scale and this causes the segmentation to obtain a "blocky" boundary (Fig. 6.13 (c)).

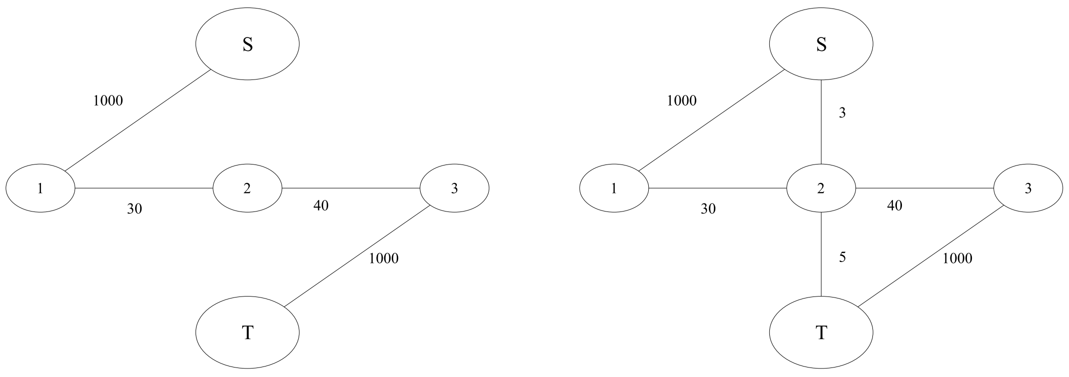
Fig. 6.15 shows the relationship of the computation time with respect to the region bias. On examination of the plot, it can be seen that increasing the bias above the critical value (*i.e.* using region information in addition to the boundary information), causes an increase in computation time for segmentation with initialisation. Effectively, more edges have been added to the graph as extra non-saturated t-links have been introduced (See Fig. 6.14). However, this computational penalty is mitigated as the bias increases.

It is also clear from the plot that, for small values of Λ (< 0.01), initialisation dramatically reduces the computation time. This difference does not exist for higher values of Λ.In fact, the computation time without initialisation is slightly less, which is due the computational cost of performing the initialisation.



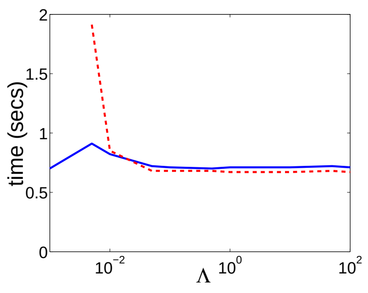
(a) Torn Frame (b) Good Segmentation with Small Λ (c) Blocky Segmentation with Large Λ

Figure 6.13: *This figures demonstrates the affect of the value of Λ on the visual quality of the segmentation. For smaller values of the region bias, the boundary of segmentation agrees with the path of the tear. The red and green colours represent the segmentation map of the image. However, for larger values, the segmentation boundary becomes blocky which is due to the influence of the motion field. The arrows point to locations where pixels have been mislabelled.*

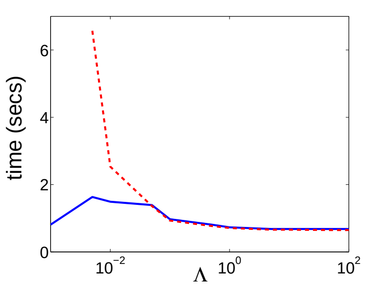


(a) Graph using boundary information only (b) Graph using both boundary and region information

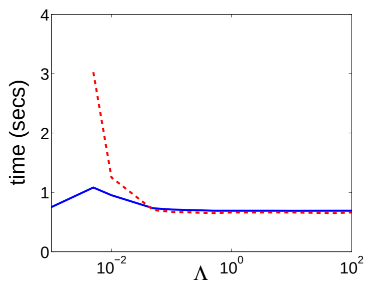
Figure 6.14: *This figure demonstrates how including region information adds to the computation time of the segmentation. There are five nodes in the graphs shown above: a source node a sink node and 3 numbered nodes that are to be assigned labels using the graph cuts algorithm. In both examples, node 1 is seeded as belonging to the source and node 3 is seeded as belonging to the sink. Node 2 remains unassigned. When boundary information is used exclusively, there are no non-saturated t-links connected to the source or sink (left). When region information is added to the framework (right), there are 2 non-saturated t-links at Node 2, the capacities of which describe the region information at Node 2. Effectively there are 2 extra edges on the graph and this increases the number of possible paths between the source and the sink. Hence, the computational complexity of the algorithm is increased.*



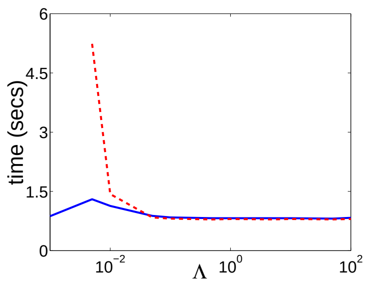
(a) First frame



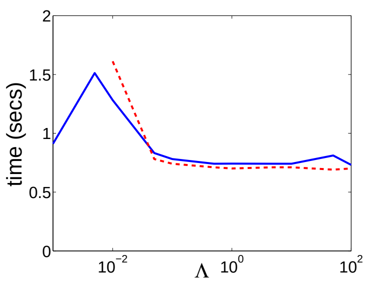
(b) Second frame



(c) Third frame



(d) Fourth frame



(e) Fifth frame

Figure 6.15: *This figure shows the computation time against Λ for the five torn frames of the mobile and calendar. Measurements are only given for values of Λ where the quality is non-zero (i.e. for the non-initialised measurements when the region energy bias is less than the critical value).*

The Erosion Parameter

The erosion parameter controls the number of extra erosion iterations performed on the seeding function (See Fig. 6.16) before it is used to initialise the graph-cuts segmentation. Fig. 6.17 shows how the quality of the segmentation is affected by the erosion parameter when region information is excluded from the segmentation (*i.e.* $\Lambda = 0$). Typically, the quality is unaffected by the number of iterations performed (although for some examples there is a slight quality penalty if no extra erosion is performed Fig. 6.17(b) and (e)) up to a breaking point after which the quality collapses. Examination of the initial seeding function and final segmentation map in this scenario shows that the boundary of the final segmentation "shrinks" to the boundary of one of the regions in the seeding function (See Fig. 6.19). This "shrinking" has been reffered to by Boykov and others in the past [10, 11, 64] and it occurs because the cost of cutting along the length of the seeding function boundary is lower than the tear boundary, due to its much shorter length.

From the graph in Fig. 6.18, the computation time is roughly proportional to the number of

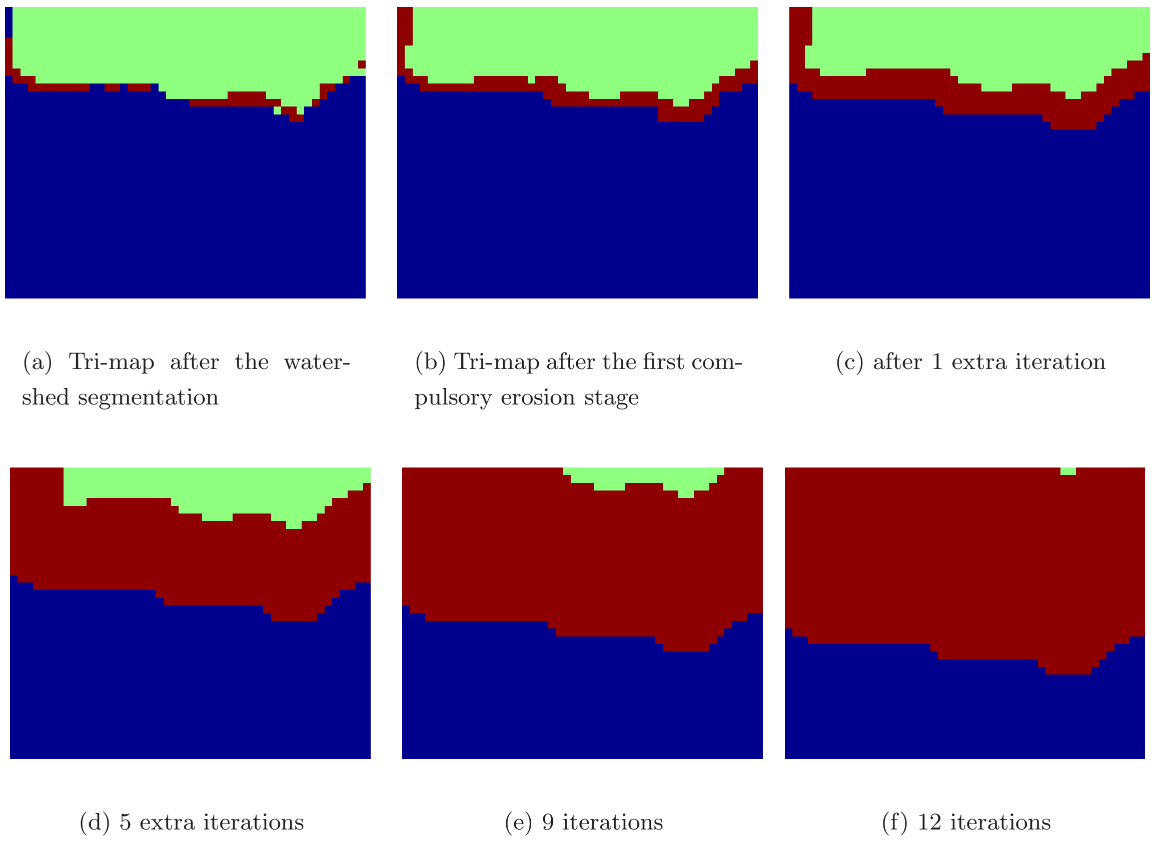

(a) Tri-map after the watershed segmentation
(b) Tri-map after the first compulsory erosion stage
(c) after 1 extra iteration
(d) 5 extra iterations
(e) 9 iterations
(f) 12 iterations

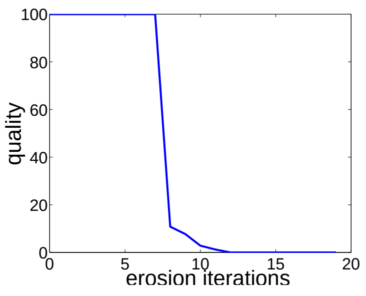
Figure 6.16: *This figure shows the seeding function after a number of different points of the erosion stage. The blue and green colours represent blocks assigned to the two regions and the red colour represents unassigned blocks.*

erosion iterations performed. There are two significant reasons for this relationship. Firstly, the overhead associated with the erosion operation itself increases with the number of iterations. More significant, however, is the increased computational overhead of the graph-cut segmentation. As the number of erosion iterations increases, the size of the seeded regions decreases and the number of unassigned pixels increases. Hence, the computational complexity of the algorithm increases. This relationship no longer applies once shrinking occurs.

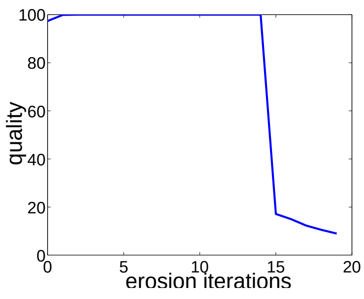
When region information is included in the segmentation framework (*i.e.* $\Lambda > 0$), shrinking no longer occurs and the quality remains constant (Fig. 6.20). Although the computation time initially increases with the number of iterations for small values of Λ (≈ 0.01), it tends to level off after a number of iterations (Fig. 6.21).

The Optimum Set of Parameters

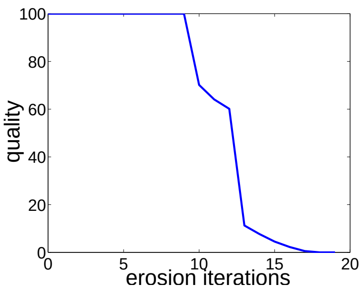
It is desirable at this stage to decide on a set of parameters that would allow the segmentation algorithm to achieve good performance over the widest variety of frames. The main choice is



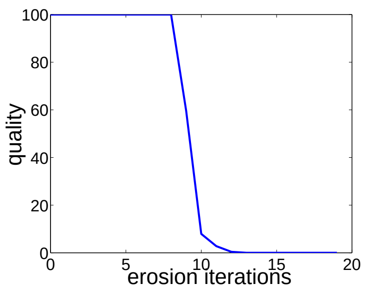
(a) First frame



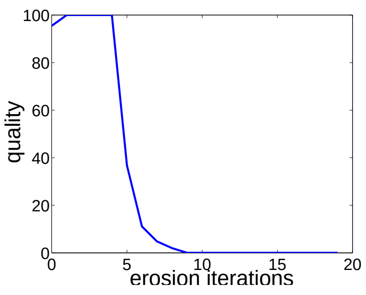
(b) Second frame



(c) Third frame



(d) Fourth frame

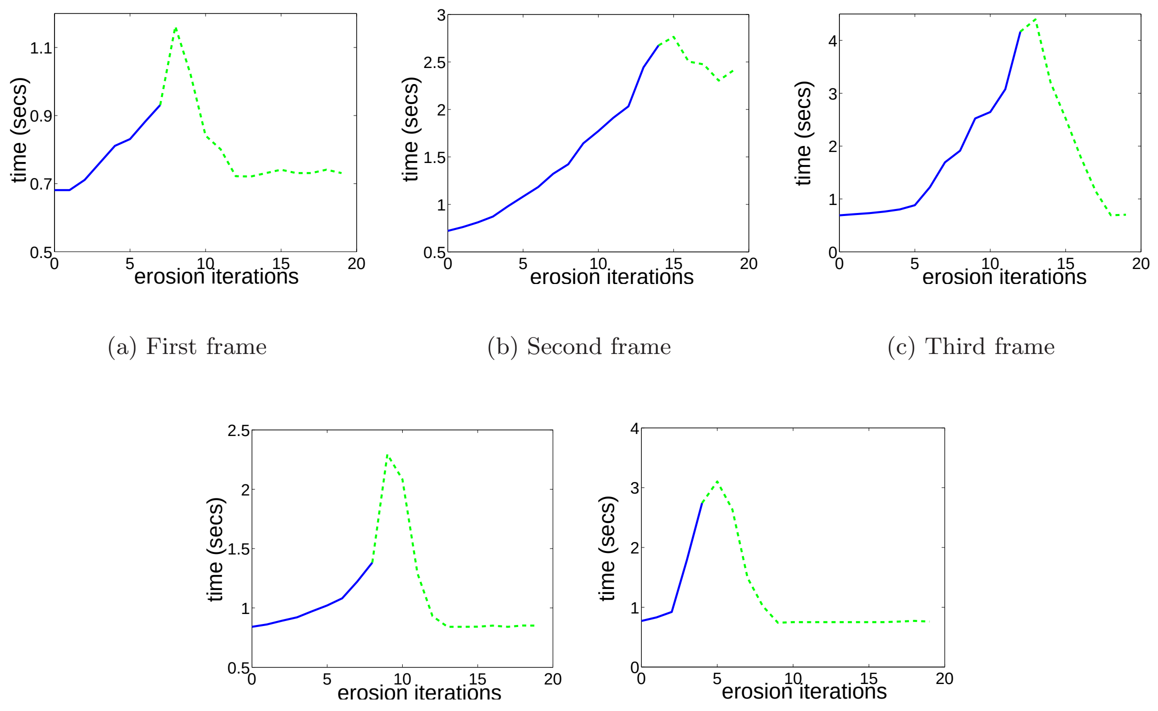


(e) Fifth frame

Figure 6.17: *Parts a to e give a plot of the segmentation quality against the number of erosion iterations for the five torn frames of the "mobile and calendar" sequence.*

to decide whether or not to initialise the graph-cuts segmentation using the seeding function. From Fig. 6.12 it can been seen that, for values of Λ greater than the critical value, initialisation does not greatly change the quality or the computational overhead of the result. Furthermore, shrinking is no longer a factor and, if initialisation is not performed, erosion becomes unnecessary. The main advantage of seeding the segmentation, however, is that it allows user interactivity to be introduced into the result. It allows the user to improve the quality of the result by manually adjusting the seeding function used for initialisation. This is important in restoration applications as the quality of the result is more important than automation of the process. If a seeding function is used, then the erosion operation becomes necessary and the number of iterations of the second erosion stage needs to be chosen.

The second choice to be made is what value of Λ is to be used. For values of Λ above the critical limit, it has been shown that increasing Λ reduces the computation time (Fig. 6.15). However, if Λ is too large, there is a drop off in quality. Furthermore, using a seeding function allows segmentation which excludes any region information for any pixels marked as ambiguous in the seeding function. This has been shown to be more computationally efficient than using small values of Λ.



(a) First frame (b) Second frame (c) Third frame

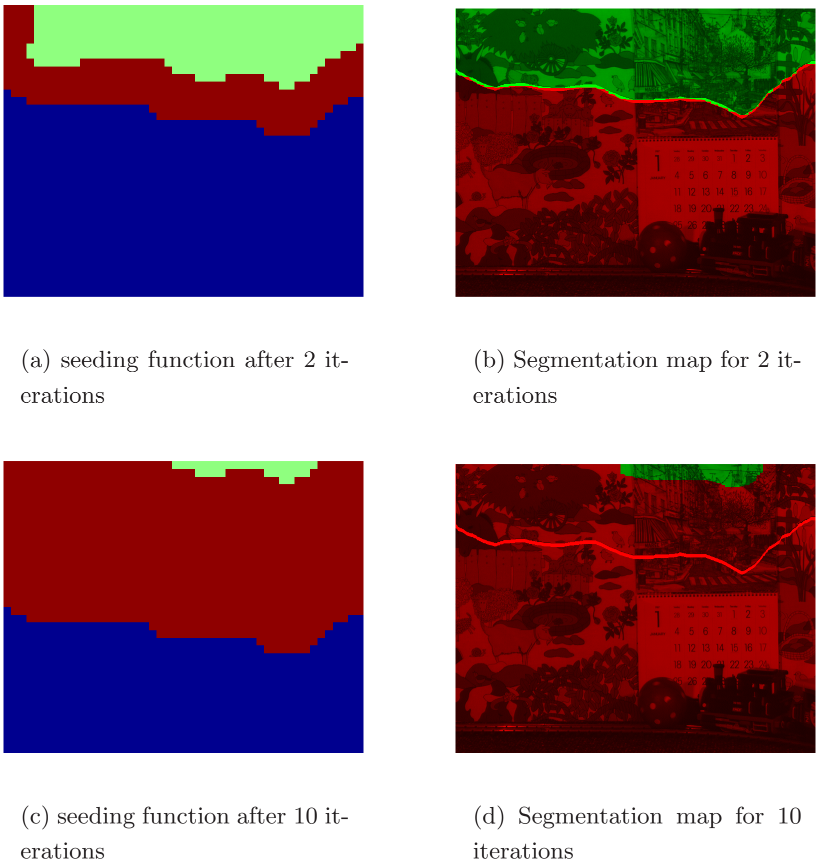
(d) Fourth frame (e) Fifth frame

Figure 6.18: *This figure shows the relationship between the number of erosion iterations and computation time of the algorithm for the second torn frame. The blue line indicates the relationship before shrinking and the dashed green line indicates the relationship after shrinking occurs.*

Taking these considerations into account, the most suitable setup for the algorithm is to use a seeding function to initialise the segmentation and to choose $\Lambda = 0$. Two iterations of the second erosion stage are chosen to make sure that all mislabelled blocks are removed from the seeding function before it is used to seed the graph cuts algorithm. Effectively, the segmentation relies entirely on the boundary conditions given the initialisation provided by the eroded seeding function. Table 6.4 shows the quality measurements for all the torn frames in both test sequences using this configuration.

For eight out of the ten frames, a high quality segmentation is obtained (See Fig. 6.22) including all of the frames from the mobile and calendar sequence. However, there are significant errors in two of the frames in the jumper sequence (Fig. 6.23). This sequence exhibits many forms of degradation, compared to the mobile and calendar sequence which is dirt free and largely noise free. Also regions of the background in the jumper have low texture content.

The frame with the worst quality is the second torn frame of the jumper sequence. Errors in the seeding function (Fig. 6.24) used to seed the segmentation are the cause for the failure



(a) seeding function after 2 iterations

(b) Segmentation map for 2 iterations

(c) seeding function after 10 iterations

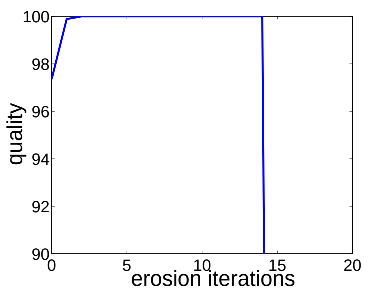
(d) Segmentation map for 10 iterations

Figure 6.19: *An example of shrinking. For 2 erosion iterations the segmentation gives a good result. However when 10 iterations are performed the segmentation boundary shrinks to the boundary of the green region in the seeding function.*

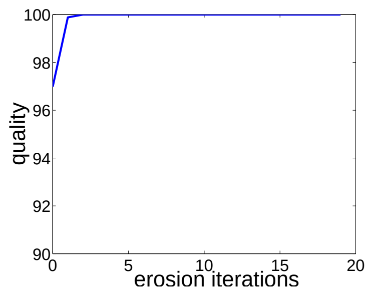
Torn Frame	**Quality** (%)	
	Mobile and Calendar Sequence	**Jumper Sequence**
1	99.998	99.990
2	100.000	69.954
3	99.998	99.949
4	99.989	99.845
5	99.996	93.764

Table 6.4: *The Segmentation quality values on all the test frames for the selected configuration.*

in this frame. These errors occur because of motion estimation failure in the frame due to lack of texture and the aperture effect. This causes vectors in the right region of the image to be similar in value to the global motion of the left region and consequently these vectors are

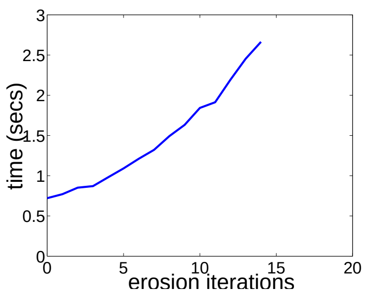


(a) $\Lambda = 0$

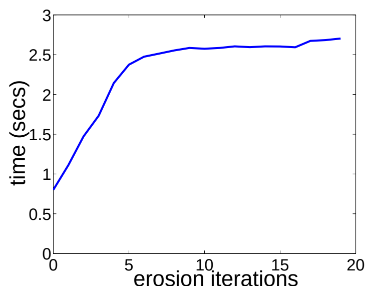


(b) $\Lambda = 0.01$

Figure 6.20: *This figure gives a plot of the segmentation quality against the number of erosion iterations for the second torn frame for two values of* Λ*. When region information is included (right), shrinking does not occur.*



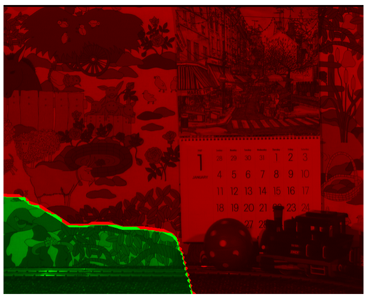
(a) $\Lambda = 0$

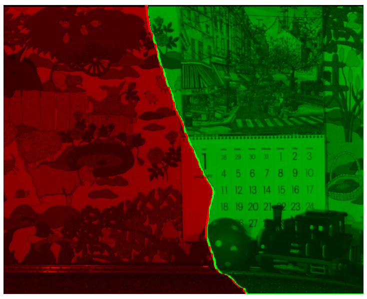


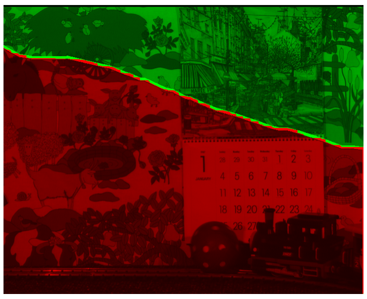
(b) $\Lambda = 0.01$

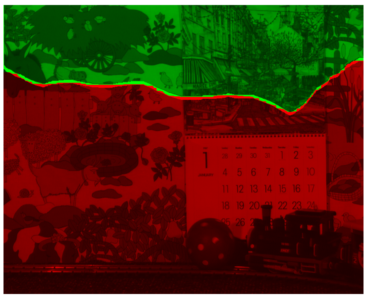
Figure 6.21: *This figure shows how including region information affects the relationship between the computation time and the number of erosion iterations. Initially, the time increases more rapidly as the number of iterations increases but levels off for larger numbers of iterations.*

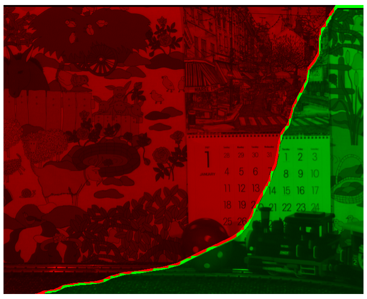
mislabelled in the seeding function. A higher value of the erosion parameter is needed to mark all mislabelled blocks as ambiguous. Another possible strategy to prevent this problem from happening would be to weight blocks on their texture content. However, this could significantly reduce the number of vectors used in the histogram and make shrinking more likely. The other frame which gives a bad result is the fifth frame of the jumper sequence. This failure is a result of shrinking. No blocks are mislabelled in the seeding function (Fig. 6.25). As the segmentation has been seeded, a good segmentation can be achieved for these frames by allowing the user to edit the seeding function (Fig. 6.26).

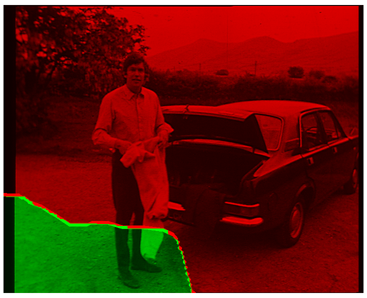

(a) Mobile and Calendar Sequence Frame 1

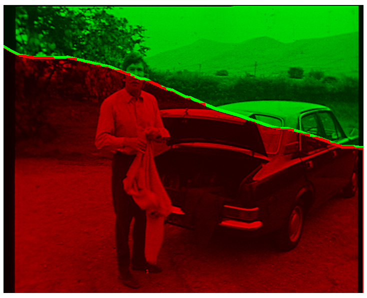

(b) M. and C. Sequence Frame 2


(c) Third frame


(d) Fourth frame


(e) Fifth frame


(f) Jumper frame 1


(g) Jumper frame 3

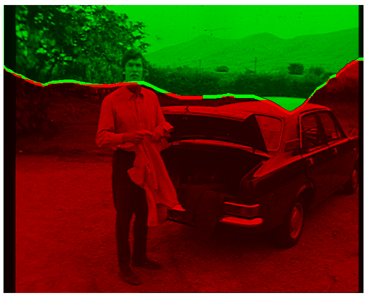

(h) Fourth frame

Figure 6.22: *Each image shows the frames which had a good quality segmentation with the chosen set of parameters. The segmentation quality measure for each image is above 99.5%.*

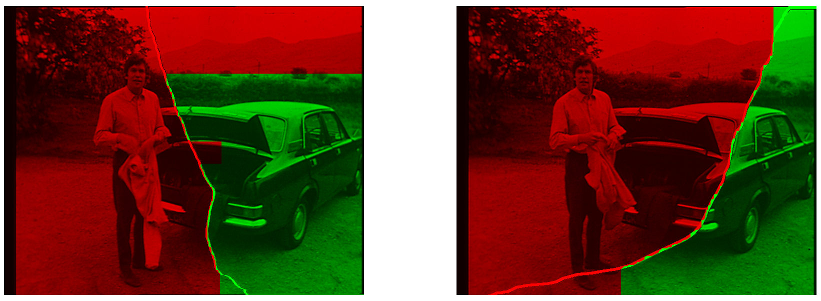

(a) Jumper Sequence Frame 2 (b) Jumper Sequence Frame 5

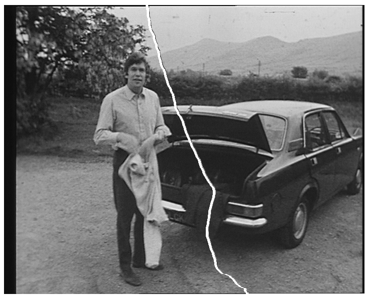
Figure 6.23: *This figures shows the 2 frames for which the segmentation fails with the current set of parameters.*

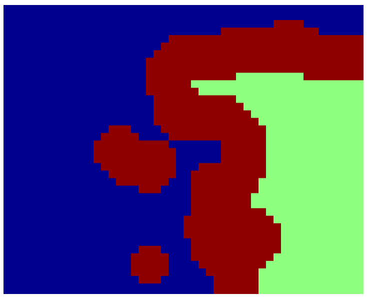
6.4.3 Displacement Estimation

Table 6.5 lists the estimated displacements of the ground truth sequences and compares them to the "mobile and calendar" and "jumper" sequences. For all of the frames in the ground truth, the estimated displacement is accurate to within a pixel of the true displacements. The estimates are more accurate for the mobile and calendar sequence which has a mean estimate distance of 0.23 pixels compared to 0.38 pixels for the jumper sequence. This is unsurprising given the relative high quality and the better texture content of the mobile and calendar which benefit the performance of the motion estimator. Since the displacement estimates are derived from the motion field directly, the accuracy of the estimate is dependent on the accuracy of the motion estimator.

In order to get a better idea as to the sufficiency of the estimates, it is necessary to correct the estimated tear displacement and to compare the result with the original image. As the sections of the tear boundary will still be present after the displacement is corrected, missing data treatment must also be applied. Fig. 6.27 shows a section of a frame of each ground-truth sequence at 4 different stages of the process. Comparing the original and restored images, any remaining tear displacement is hard to perceive.

Some artefacts have been introduced into final restored images(Fig. 6.29). These artefacts have been introduced, for the most part, by the Missing Data Treatment (MDT) algorithm [60]. Another noticeable feature of the restored images is that they are blurred compared to the originals. One cause of the blurring is the tear displacement compensation. It uses bilinear interpolation to compensate for non-integer displacements which acts as smoothing filter. Another cause of the blurring is the missing data treatment algorithm that is used, which also acts as a noise reducer in the image.


(a) Second Torn Frame


(b) Seeding Function used to Initialise Segmentation

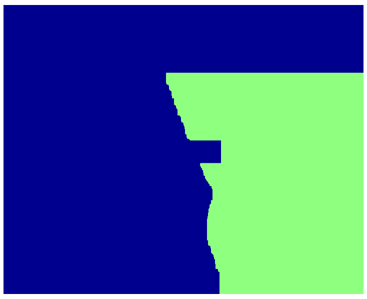
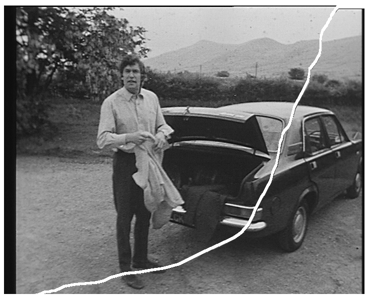
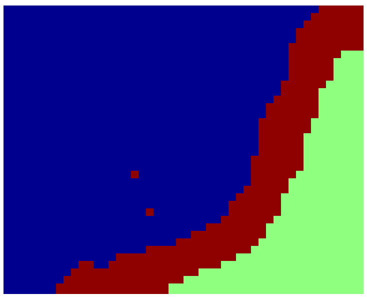

(c) Segmentation mask

Figure 6.24: *This figure shows the seeding function for the segmentation and the segmentation result for the second torn frame of the jumper sequence. The green and blue colours represents the two regions and the red region in the seeding function represents pixels whose region is ambiguous. Blocks in the top right region of the frame and in the dark region are mislabelled as belonging to the blue region even when they should be located in the green region.*


(a) Second Torn Frame


(b) Seeding function used to Initialise Segmentation

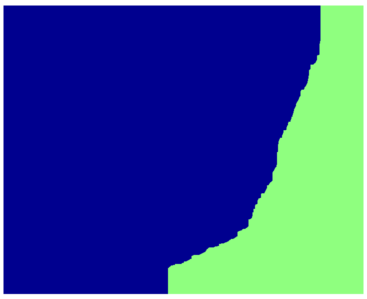
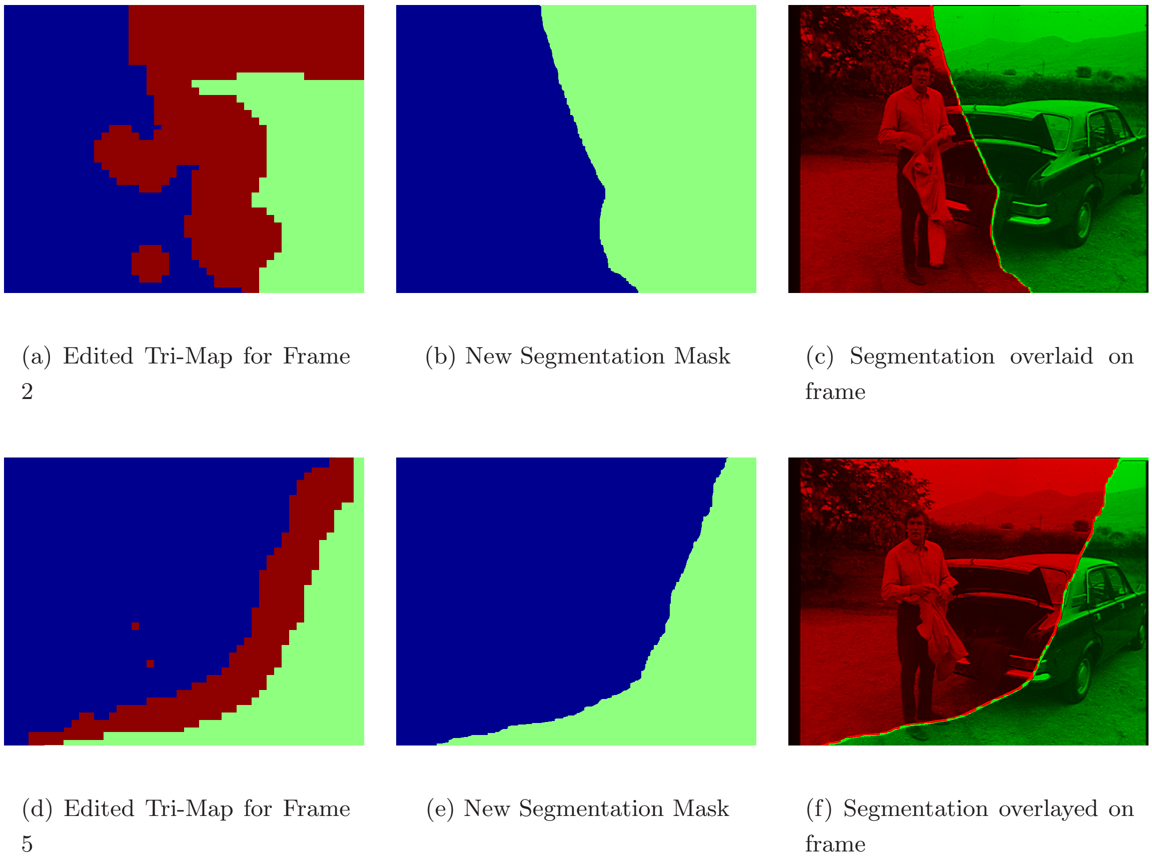

(c) Segmentation mask

Figure 6.25: *This figure shows the seeding function for the segmentation and the segmentation result for the fifth torn frame. In the seeding function (centre), no blocks are assigned to the wrong region. Shrinking occurs because the cost of the shorter path is lower even though the average energy of each cut edge is higher.*

6.4.4 Real Examples

The segmentation algorithm was tested on 4 torn frames in 3 sequences (Fig. 6.30). The first “BBC” sequence contains 2 torn frames on consecutive frames. In fact, the tear in both frames is caused by a single tear the boundary of which starts on the left of the first frame and passes through the bottom of the frame to the top of the second frame and then to the right of the image. This means that the relative tear displacement should be the same in both frames and



(a) Edited Tri-Map for Frame 2

(b) New Segmentation Mask

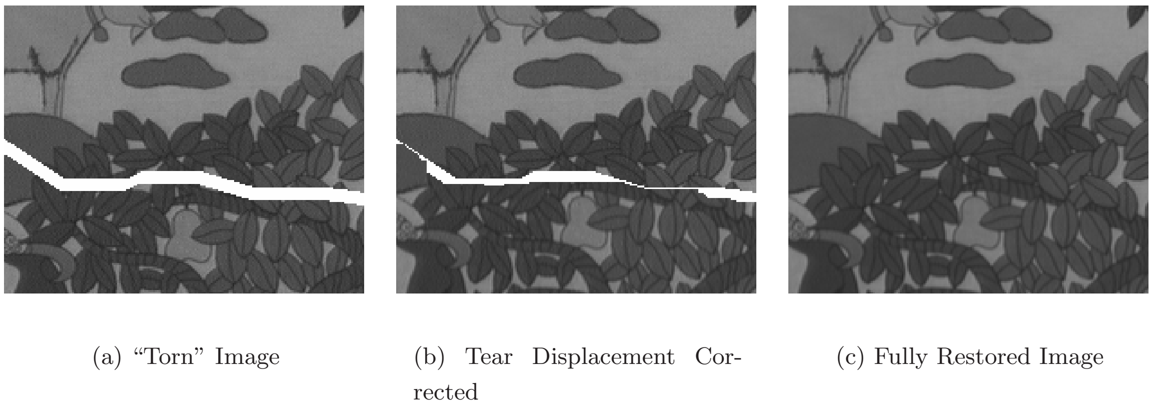
(c) Segmentation overlaid on frame

(d) Edited Tri-Map for Frame 5

(e) New Segmentation Mask

(f) Segmentation overlayed on frame

Figure 6.26: *This figure shows the segmentation masks (middle & right) obtained from the edited seeding functions (left).*



(a) "Torn" Image

(b) Tear Displacement Corrected

(c) Fully Restored Image

Figure 6.27: *This figure shows the close up of the fully restored first frame from the mobile and calendar sequence (c). Image (a) is from the artificially torn sequence and (b) is the image after the tear displacement has been removed.*

Torn Frame	**Ground Truth**		**Mobile and Calendar Sequence**			**Jumper Sequence**		
	$\mathbf{d}_x$	$\mathbf{d}_y$	$\mathbf{d}_x$	$\mathbf{d}_y$	distance	$\mathbf{d}_x$	$\mathbf{d}_y$	distance
1	-2	7	-2	7.5	0.50	-1.875	6.375	0.64
2	5.439	-0.872	5.375	-1.25	0.38	5.375	-1.125	0.28*
3	-1.345	6.254	-1.5	6.25	0.16	-1.625	6.25	0.28
4	0.1	-7.872	0.125	-7.875	0.03	0.375	-8.25	0.47
5	4.567	5.212	4.5	5.125	0.11	4.375	5.375	0.25*

Table 6.5: *A table of the estimated tear displacements for the ground truth sequences. The distance measurements is the euclidean distance between the ground truth and the estimated displacements. Displacements are measured to an accuracy of* 0.125 *pel.*
**The measurement for the second frame in the jumper sequence is obtained using the segmentation from the edited seeding function. The measurement for the fifth frame of the sequence is the same whether the bad segmentation or the edited segmentation is used.*

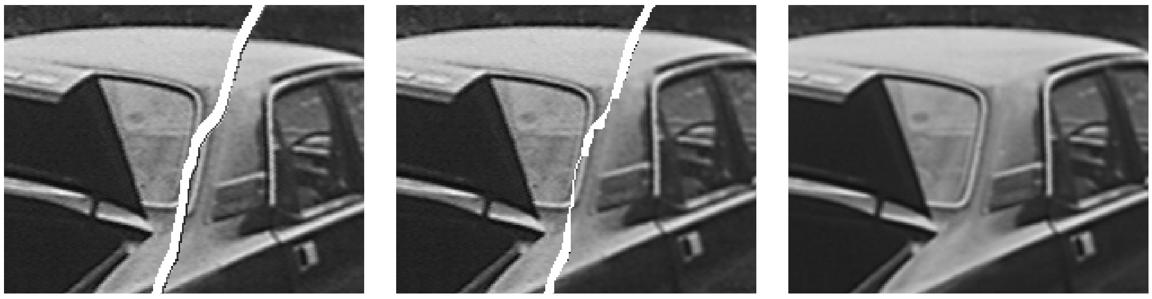
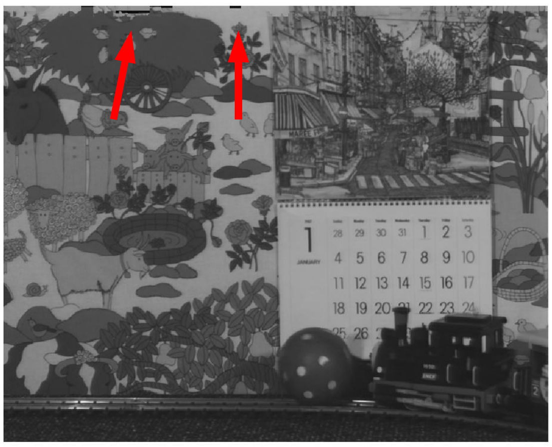

(a) "Torn" Image (b) Tear Displacement Corrected (c) Fully Restored Image

Figure 6.28: *This figure shows the close up of the fully restored fifth frame from the jumper sequence (c). Image (a) is from the artificially torn sequence and (b) is the image after the tear displacement has been removed. The segmentation obtained from the edited seeding function is used to restore this frame.*

appears to be roughly vertical. The second "scissors" sequence consists of a single torn frame. In this sequence, the relative displacement is approximately horizontal as can be seen from the misalignment of the vertical edges across the tear boundary. The final sequence is referred to here as the "singer" sequence in which one frame is torn.

The segmentation algorithm was applied to these frames using the parameters discussed in Section 6.4.2 (*i.e.* $\Lambda = 0$ and a value of 2 for the erosion parameter) and the results are shown in Fig. 6.31. For the "BBC" and "scissor" sequences, the segmentation agrees well with the


(a) Mobile & Calendar, 4th Frame

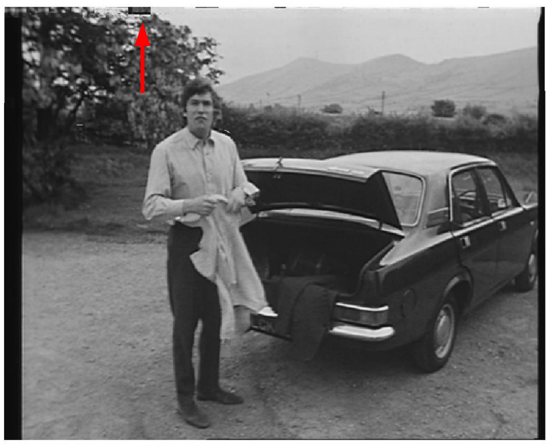
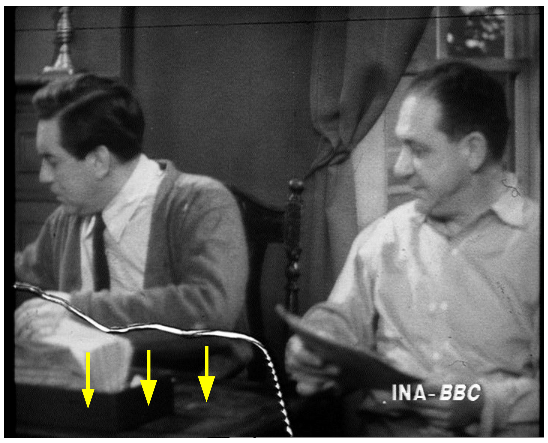

(b) Jumper, 4th Frame

Figure 6.29: *This figure shows 2 fully restored images which contain some artefacts introduced by the tear restoration process. These artefacts are highlighted by the arrows.*

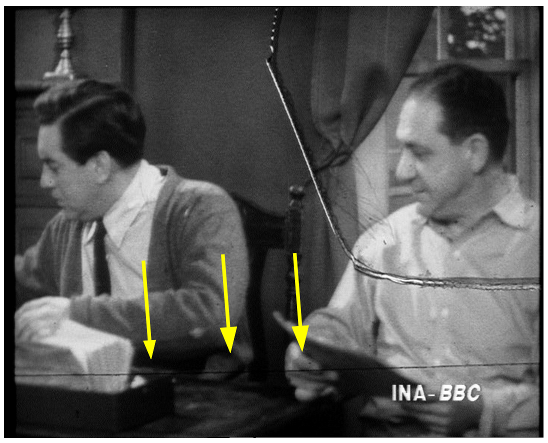
perceived correct segmentation (*i.e.* The segmentation boundary follows the path of the tear). However, some shrinking occurs near the edge of the first "BBC" frame and the torn frame from the "scissor" sequence (Fig. 6.32). Instead of following the tear boundary, the segmentation boundary follows the shortest path to perimeter of the frame.

As has been stated previously, the motion estimator used in the segmentation algorithm [4,54] produces 2 motion fields per frame for motion with respect to the previous or next frame in the sequence. Normally, for isolated torn frames, either motion field can be used (*e.g.* all the torn frames in the ground truth sequences). However, this choice is restricted when two or more consecutive frames are torn, as is the case in the "BBC" sequence. A problem occurs because the displacement introduced by the tear in the first frame biases the backward motion field of the second torn frame (*i.e.* motion with respect to the first frame). This results in failure of the motion-based segmentation which seeds the graph-cuts segmentation (See Fig. 6.33). This problem can be overcome in a number of ways. One method is the use the backward motion field for the first torn frame and forward motion field for the second torn frame as in Fig. 6.31. Another possible method is to first segment and correct the displacement in the first torn using the backward motion field and to recompute the motion before restoring the next torn frame.

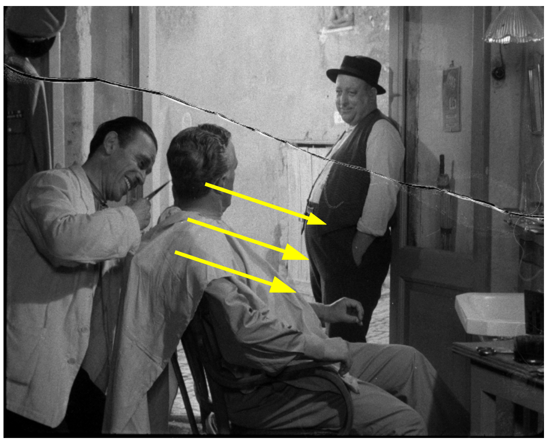
The segmentation map in Fig. 6.34 for the "singer" sequence is not a visually agreeable segmentation of the frame. The segmentation fails because the two largest peaks in the 2-D motion histogram do not correspond to the regions divided by the tear which in turn causes the graph-cuts segmentation to fail. Neither the forward nor the backward motion fields for this frame, shown in Fig. 6.35, are well behaved. There are large areas of the frame undergoing local motion and the motion in the background is also erratic which makes it difficult to find



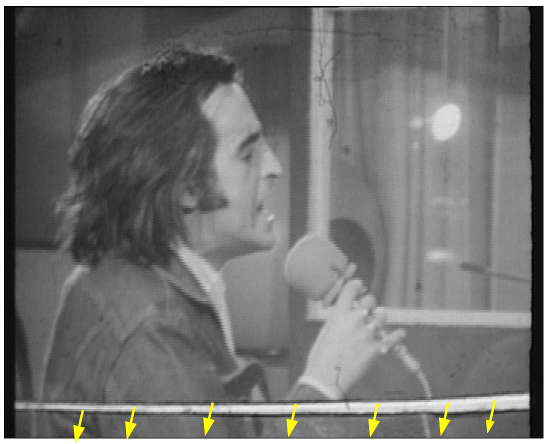
(a) “BBC” sequence, 1st frame



(b) “BBC” sequence, 2nd frame



(c) “scissors” sequence



(d) “singer” sequence

Figure 6.30: *This figure shows the 4 torn frames. The arrows indicate the displacement introduced by the tear.*

the translational global motion from the motion field. A better segmentation can be obtained with a user defined seeding function (Fig. 6.36).

Frame Restoration

Having estimated the segmentation map for each frame, the tear displacements for the frames are estimated and then corrected. The estimated displacements are shown in table 6.6 and the corresponding compensated images are shown in Fig. 6.37. As expected, the displacements for the two frames in the “BBC” sequence are roughly similar, although there is a slight difference. The torn frame from “scissors” sequence (Fig. 6.37(c)) gives a good visual indication of the

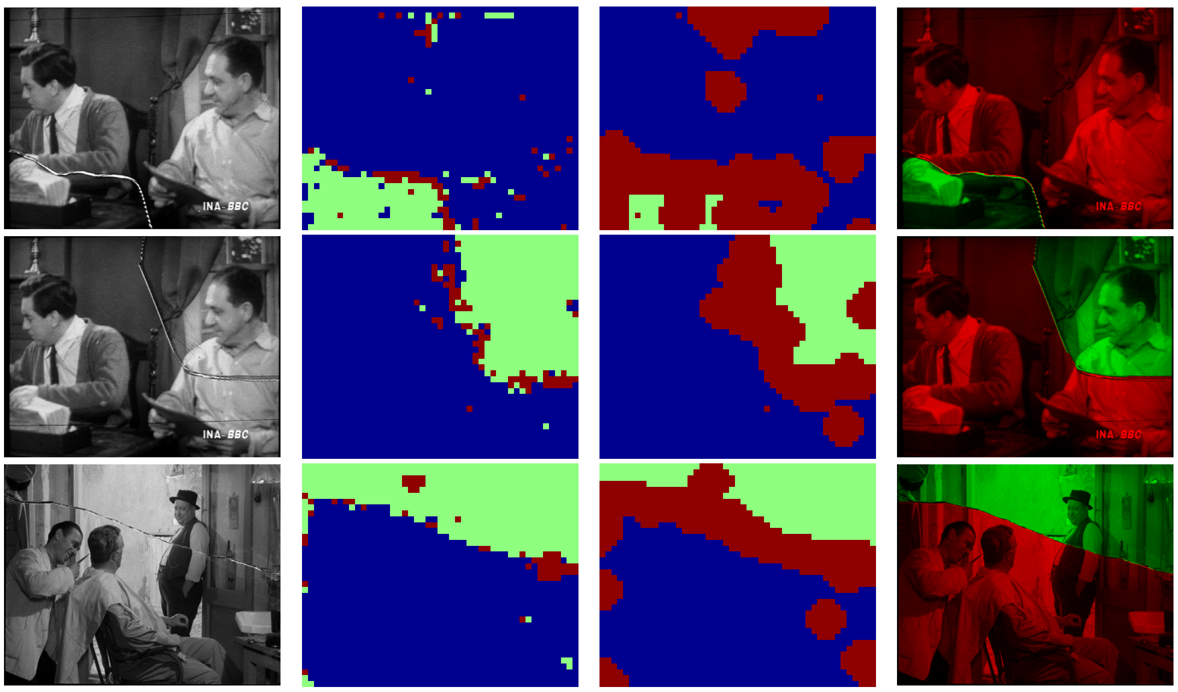
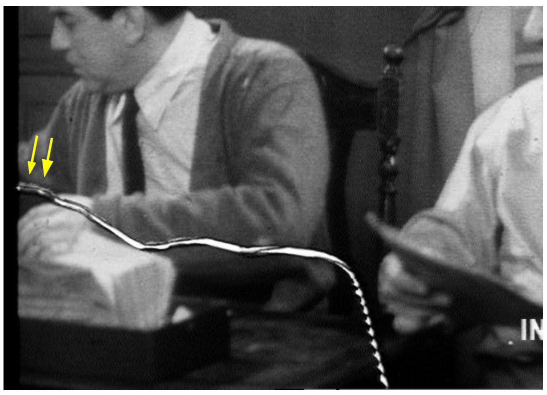


Figure 6.31: *First Column (from left): The torn frames from the "BBC" and "scissors" sequences; Second Column: The tri-maps after the watershed segmentation; Third: The tri-maps after the erosion stage; Fourth: The torn frames with super-imposed segmentation maps.*

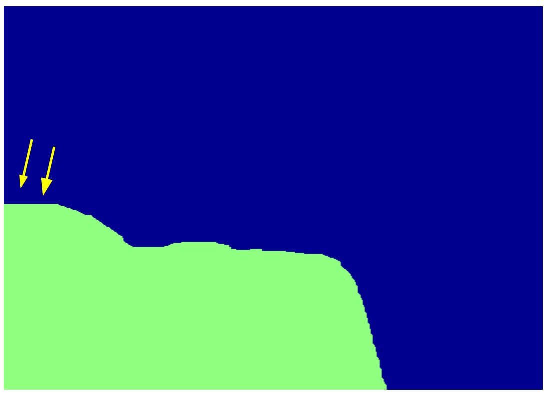
Torn Frame	Horizontal Displacement	Vertical Displacement
"BBC" Sequence, frame 1	0	6
"BBC" Sequence, frame 2	-0.05	7.25
"Scissors" Sequence	-4.5	4
"Singer" Sequence, backward motion field	-0.25	8
"Singer" Sequence, forward motion field	0	0

Table 6.6: *This table gives the estimated relative tear displacements for the listed frames. The global motion for each region is estimated from the backward motion field for the first "BBC" frame and the frame from the "scissors" sequence. The forward motion field is used for the* 2^{nd} *"BBC" frame.*

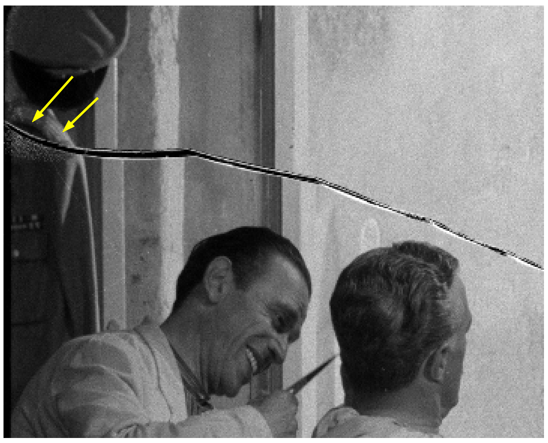
correctness of the displacement estimate for the frame. The edges which cross the tear are now visibly aligned. The estimated displacements for the "singer" sequence are not as accurate. A zero relative displacement is estimated using the forward motion field. Even the non-zero displacement, estimated from the backward motion field, does not give a good visible result (Fig. 6.37(d)) as the edges are not aligned correctly across the tear. This failure of the displacement estimation is a direct consequence of the poor quality of the motion field for this frame (Fig.



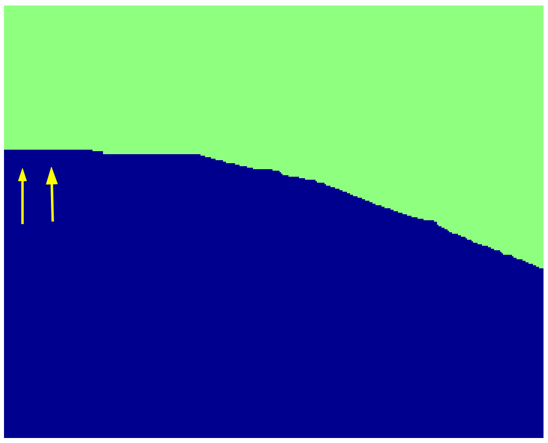
(a) Torn Frame from "BBC" sequence



(b) Segmentation Map



(c) Torn Frame from "scissors" sequence



(d) Segmentation map

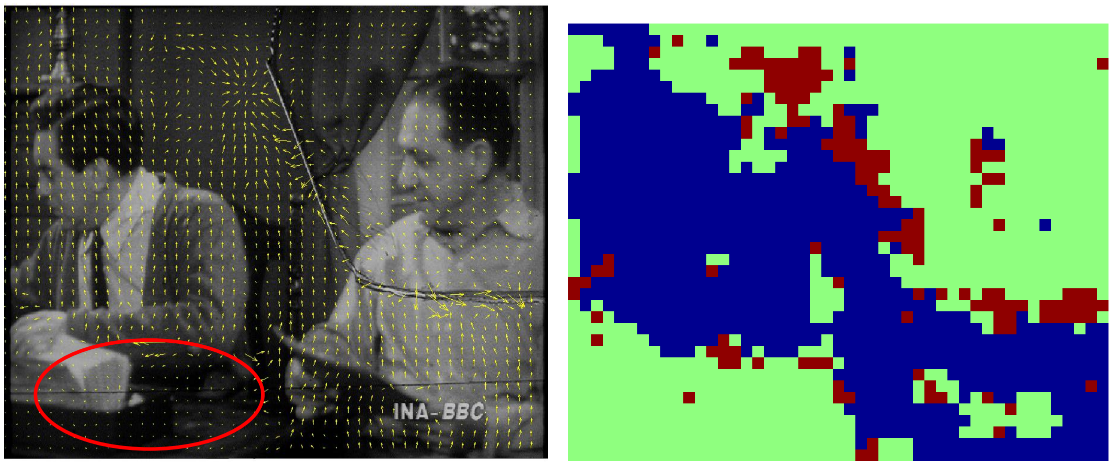
Figure 6.32: *This figure demonstrates shrinking in two of the torn frames. The arrows indicate where the segmentation boundary follows the shortest path to edge of the frame rather than following the tear boundary.*

6.35).

The frames from the "BBC" and "scissors" sequences are fully restored by recovering the lost image data along the tear using the missing data treatment outlined in [60]. Summaries of the restoration of these frames are given in Fig. 6.38

6.5 Computation Complexity of the Segmentation Algorithm

In order to determine the computational complexity of the algorithm, the complexity of each stage of the algorithm must be estimated. The following list is a breakdown for the complexity



(a) 2^{nd} Torn Frame from "BBC" sequence with superimposed backward motion field

(b) Tri-map after watershed segmentation

Figure 6.33: *The circle on the left indicates a section of the image whose motion is different from the the motion in that region of the image. It corresponds to a region in the previous torn frame. If the motion-based segmentation was applied to this motion field, then the circled section is assigned to the wrong region (right).*

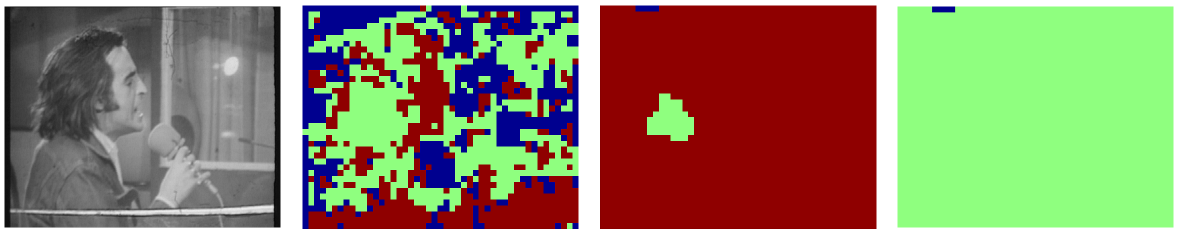
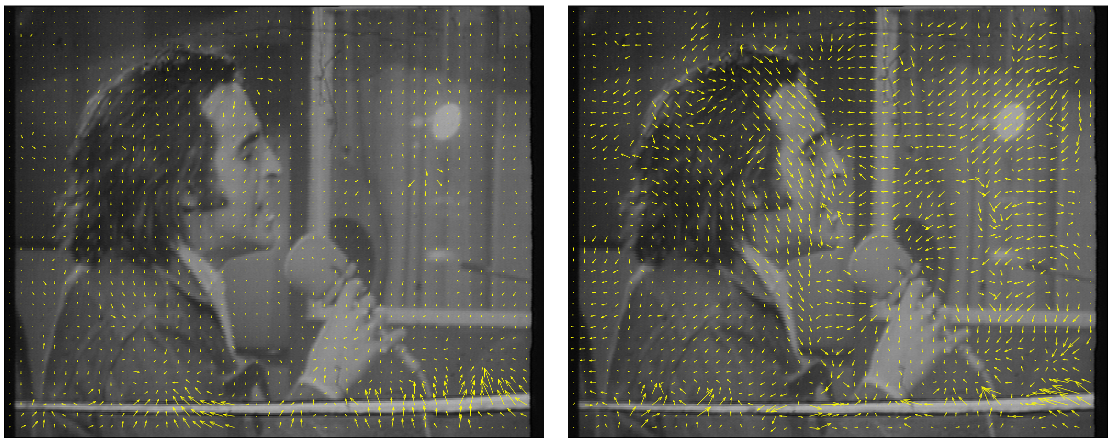


Figure 6.34: *First Column (from left): The torn frames from the "singer" sequence; Second Column: The seeding function after the watershed segmentation; Third: The seeding function after the erosion stage; Fourth: The resulting segmentation map.*

of each stage.

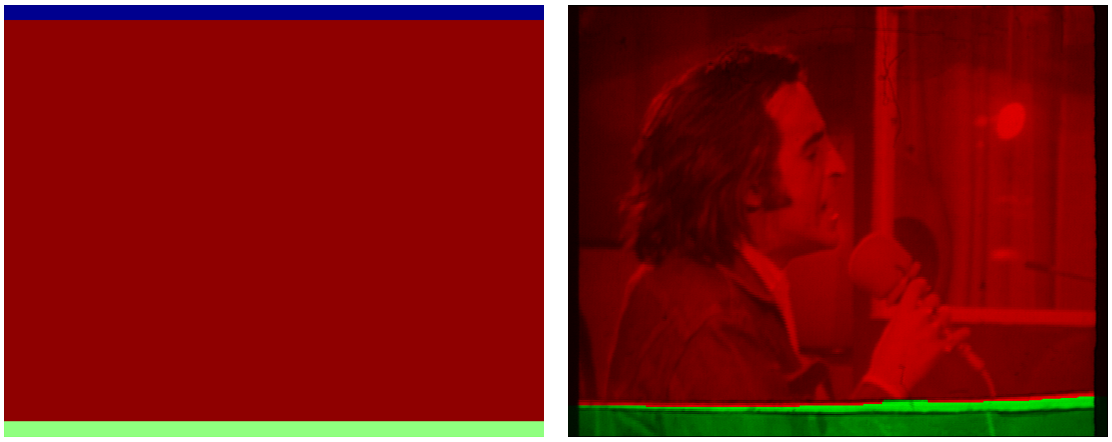
1. 2D Histogram computation - The histogram is populated by incrementing the appropriate bin in the histogram for every motion vector. The complexity is linear with respect to the number of vectors which is roughly proportional to the number of pixels in the image. After the histogram is computed it is filtered with a low pass filter using a 2-D convolution. The complexity of this operation is linear with respect to the histogram size and also the filter size which is fixed. Therefore, the overall complexity for the first stage is $\mathcal{O}(n) + \mathcal{O}(\#\text{ histogram bins})$, where n is the number of pixels in the image

2. Watershed Segmentation - According to [105] the computational complexity of the water-



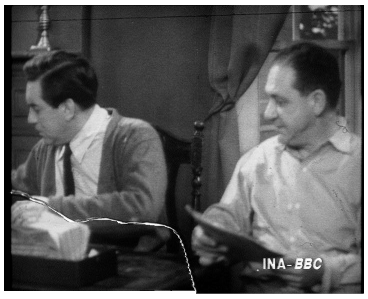
(a) Backward Motion Field (b) Forward Motion Field

Figure 6.35: *The two images show the 2 motion fields for the torn frame of the "singer" sequence superimposed on the frame.*

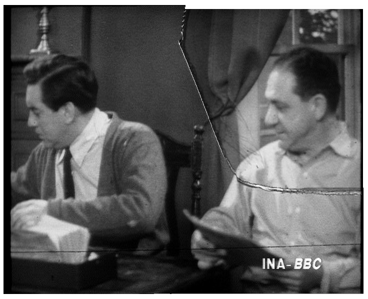


(a) Manually edited Tri-Map (b) Segmented Image

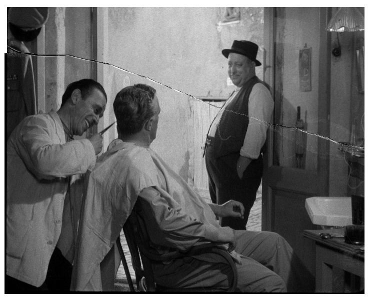
Figure 6.36: *This figure presents the segmentation of the torn frame in the "singer" sequence obtained from the edited seeding function. A band of pixels at the top and bottom of the image are marked as belonging to the two regions.*



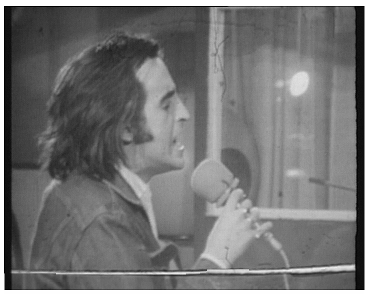
(a) "BBC" frame 1



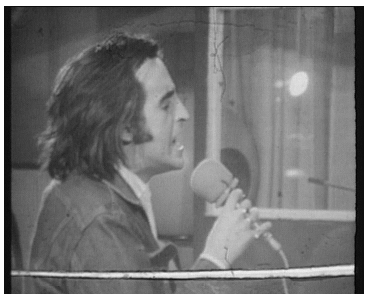
(b) "BBC" frame 2



(c) "Scissors" Sequence



(d) "Singer Sequence" using backward motion field



(e) "Singer Sequence" using forward motion field

Figure 6.37: *This figure shows each torn frame compensated for tear displacement. The segmentation masks computed from the estimated tri-maps are used for the "BBC" and "scissors" sequence and the segmentation mask estimated from the edit seeding function (Fig. 6.36) is used for the frame in the "singer" sequence.*

shed segmentation is linear with respect to the size of the input image, which in this case is the 2D histogram. The initial seeding function must subsequently be constructed from the segmented histogram. This operation is linear with respect to the size of the seeding function(*i.e.* the number of vectors in the motion field). Therefore, the complexity of this stage is $\mathcal{O}(n) + \mathcal{O}(\#\text{ histogram bins})$.

3. Erosion - The complexity of each erosion iteration is linear with respect to the size of the seeding function. Therefore, the complexity of the entire erosion operation is given by $\mathcal{O}(n \times \#\text{ erosion iterations})$.

4. Graph Cuts Segmentation - The theoretical upper limit on the computational complexity is given by $\mathcal{O}(mn^2|C|)$ [12], where m is the number of edges in the graph and $|C|$ is the cost of the minimum cut.

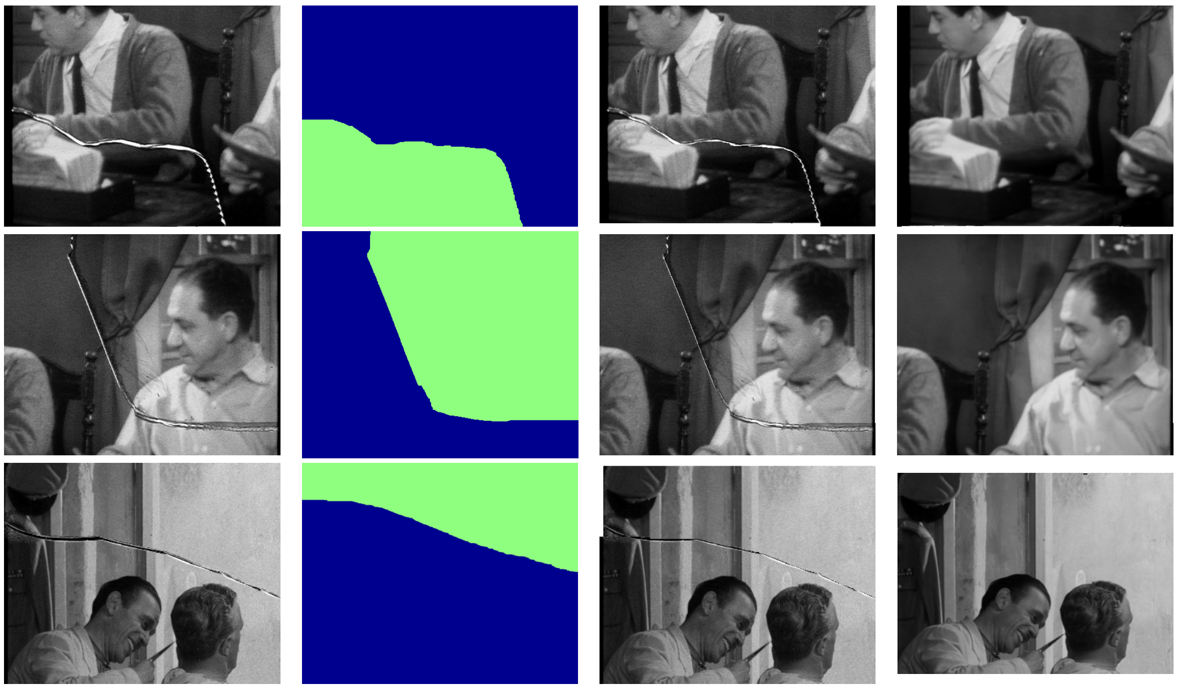


Figure 6.38: *First Column (from left): Close-up of torn frames; Second Column: Close-up of the segmentation; Third: The torn frame with compensated displacement. The edges should now correctly align across the tear boundary; Fourth: The fully restored frame after missing data treatment. The missing data treatment algorithm used [60] also acts as a noise reducer.*

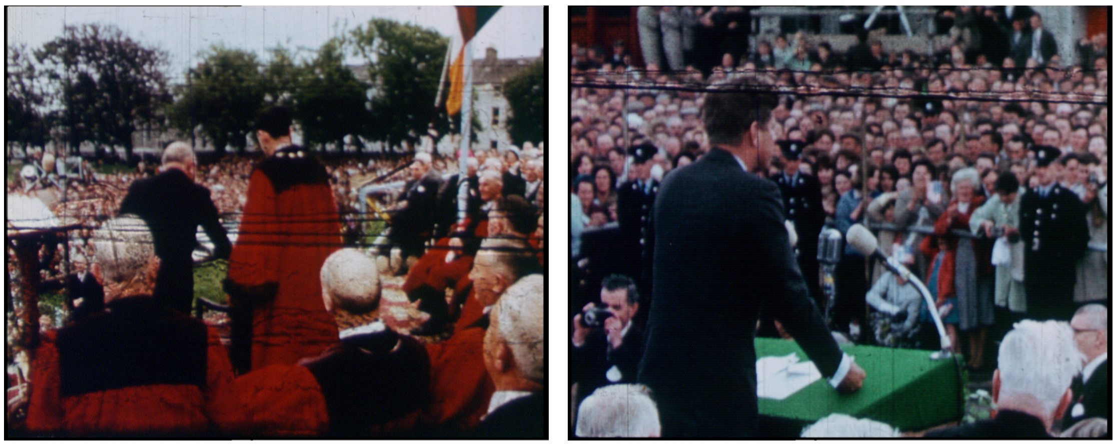
The overall computational complexity of the segmentation algorithm is therefore $\mathcal{O}(mn^2|C|)+\mathcal{O}(n)+\mathcal{O}(\#\text{ histogram bins})+\mathcal{O}(n\times\#\text{ erosion iterations})$.

6.6 Other Applications of the Segmentation Algorithm

An interesting feature of the segmentation algorithm used for tear delineation is that the boundary conditions (or the spatial smoothness conditions) are derived from the temporal DFD rather than the spatial gradient, which is often used in segmentation algorithms. This allows temporally impulsive edge features (such as tear) to be distinguished from actual image edges. Furthermore, the segmentation framework outlined in this chapter allows for segmentation without region information if the seeding function used to initialise the segmentation is user-defined.

Dirty splices are another form of degradation sometimes present in archived media (Fig. 6.39). In the past, footage was often edited by splicing together film strips from each different shot with scotch tape. Over time, dirt tends to accumulate on the film at both edges of the splice. This results in a horizontal band of dirt, visible on the two frames either side of the shot cut.

Like tear, a dirty splice is a temporally impulsive defect which forms a band across a frame.

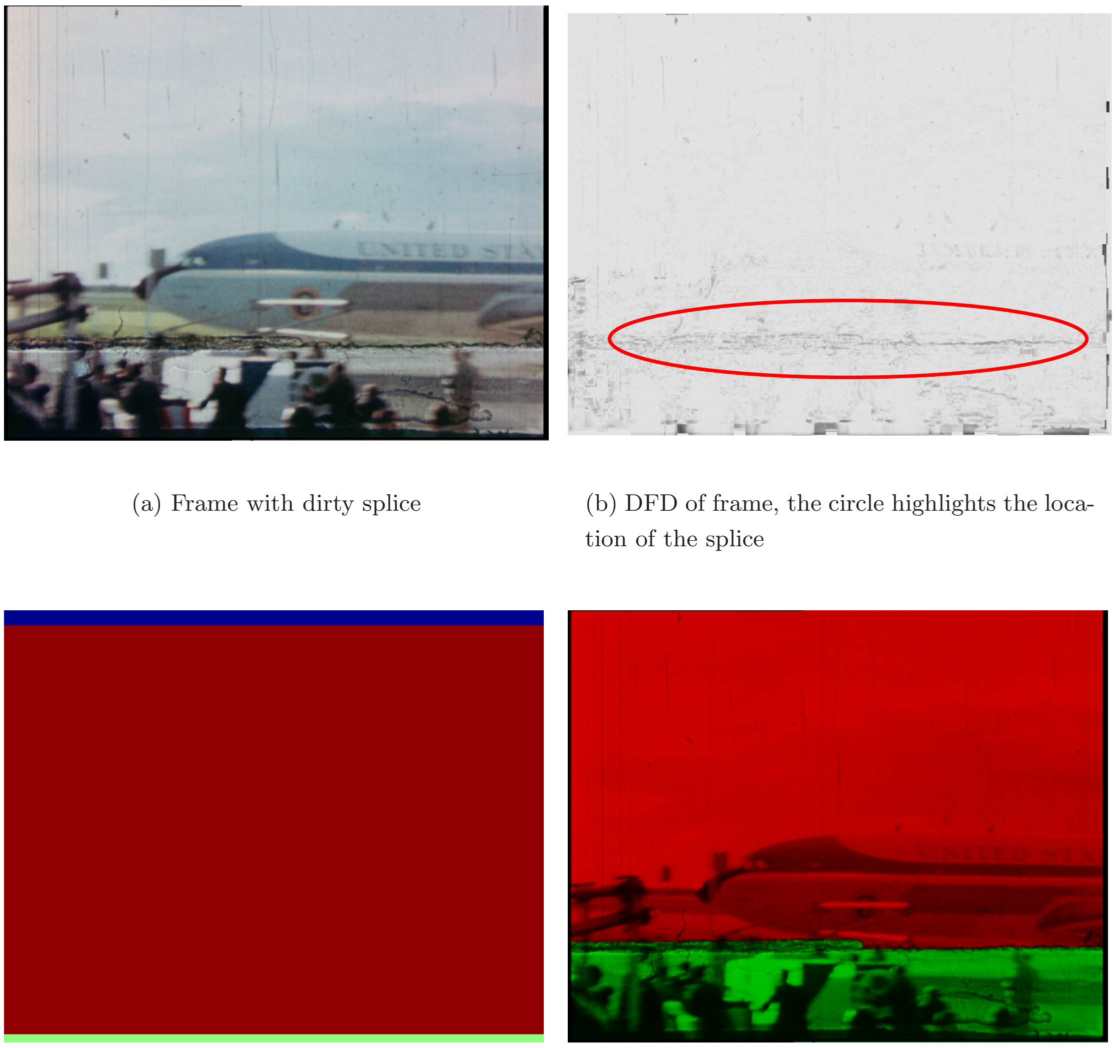


(a) Before Shot Cut (b) After Shot Cut

Figure 6.39: *This figure shows two frames with dirty splices either side of a shot cut. The horizontal lines of dirt in the images are the consequence of the splice.*

Therefore, the tear delineation algorithm could also be used to divide a frame along the path of a splice (Fig. 6.40). The DFD again describes the boundary properties of the segmentation. However, unlike tear, splices do not induce a relative displacement either side of the tear. This means that the segmentation cannot be initialised using a motion-based segmentation. The segmentation can still be performed automatically by exploiting the observation that dirty splices forms a horizontal band spanning the image. As the path of the splice is horizontal, the top and bottom rows of the image must be located on opposite sides of the splice. Therefore, the segmentation could be initialised automatically by seeding the segmentation frame work in that manner.

This raises the possibility of using the segmentation algorithm to detect dirty splices in film sequences. In effect, the goal is to detect temporally impulsive defects which form horizontal lines in an image. As splices are associated with shot cuts, only frames close to shot cuts need to be tested. If the segmentation algorithm is applied to a suspect frame and the segmentation boundary follows a horizontal path, then that indicates the presence of a dirty splice. However, if the segmentation algorithm is initialised in the manner described above, the boundary will always follow a horizontal path on any frame. The outstanding issue is to find a method of initialising the segmentation framework that would allow frames with and without dirty splices to be distinguished (Fig. 6.41).



(a) Frame with dirty splice

(b) DFD of frame, the circle highlights the location of the splice

(c) Seeding Tri-Map

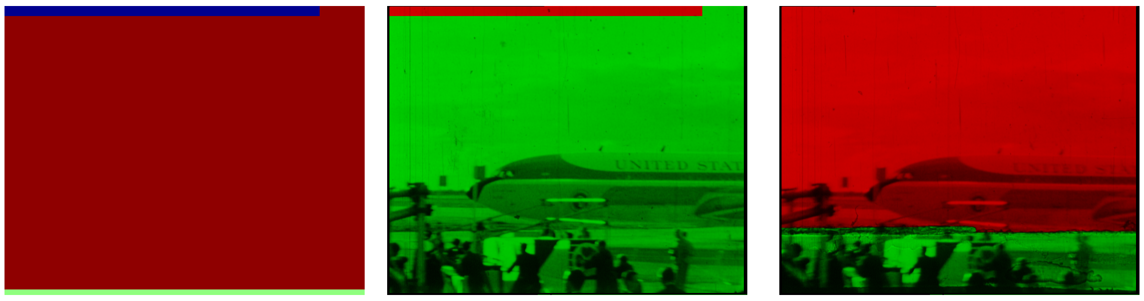
(d) Segmented Image

Figure 6.40: *This figure presents an example of the segmentation algorithm applied to a frame with a dirty splice. In the segmented image, the segmentation boundary follows the the path of the splice across the image.*

6.7 Final Remarks

This chapter has described a new automated algorithm for segmenting torn frames using the graph cuts segmentation framework outlined in [11]. The segmentation algorithm forms part of a restoration framework for torn frames, which was first outlined in [21]. The main advantage of the Graph Cuts optimisation technique over other techniques such as ICM [3] is that it estimates the globally optimum segmentation for the two label segmentation algorithm.

The algorithm was tested on two ground truth sequences in order to quantify the accuracy



(a) Seeding Tri-Map (b) Segmentation for frame with no splice (c) Segmentation for frame with splice

Figure 6.41: *A possible method to initialise the segmentation is to only label sections of the top and bottom row (a). If a dirty splice is not present in the frame, then the segmentation boundary may shrink to the boundary of the labelled section. In this case, the segmentation boundary would not completely span the image (b). If a splice is present, then the boundary will still span the image (c).*

and computational efficiency of the algorithm. Various configurations of the algorithm were tested, including applying the algorithm without any initialisation. However, it has been shown that initialising the graph-cuts segmentation was preferable as it allows for user-intervention to improve the segmentation in cases where the automated segmentation fails.

There are number of outstanding issues which need to be addressed with regard to both the segmentation algorithm and the tear restoration framework. Failures in the segmentation framework tend to occur in frames due to the nature of the frame's motion field. These failures typically are associated with local motion or motion estimation failure due to either pathological motion or ill-conditioned image data. Further development of the algorithm is necessary to make it more robust to these types of failure. Also, further investigation into the choice of energy function for the boundary conditions of the segmentation (given by equation 6.5) would be desirable. A better defined function might reduce the frequency of shrinking in the segmentation.

Currently, the tear displacement estimation algorithm in the restoration framework uses a translational model for global motion. Consequently, the tear displacement algorithm is not suitable for torn frames where rotation of a region occurs or where the motion of the camera is non-affine. The performance of the displacement correction process could be improved by considering an affine global motion model. Furthermore, additional testing of the torn frame detection algorithm proposed in [21] is needed.

7 Conclusions

This thesis was focused on improving the robustness of image processing applications to motion estimation failure. It was divided into two parts, with the first being concerned with applications involving local motion estimation and the second with applications involving global motion estimation. The following sections give a brief review of the work presented in this thesis and propose some ideas for future work in these areas.

7.1 Pathological Motion Detection

Chapter 2 introduced and described the phenomenon of Pathological Motion as well as presenting a review of the prominent missing data detection algorithms. It introduced a classification of the different forms of PM in terms of their cause and manifestation in sequences. The missing data detectors were classified in terms of the how they deal with long term Pathological Motion. These included the standard detectors that made no direct effort to model PM, those that make missing data detection more conservative in suspected PM regions and those that take a preventative approach of preventing the detection of blotches in detected PM regions. The deterministic and probabilistic methods were also discussed. Finally, the scope for a new PM-robust blotch detection algorithm using a fully probabilistic framework was highlighted.

The new PM-robust blotch detector was then introduced in chapter 3. The ground truth comparison with the standard detectors showed that the effect of the new algorithm was to significantly reduce the rate of false detections of blotches. The reduction in false alarms had the positive impact of reducing the damage to images caused by the false detection of blotches in PM regions. However, the new algorithm also had the effect of capping the correct detection rate of blotches.

The algorithm was also compared with the algorithm of Bornard, another blotch detector which allows for PM. The comparison showed that the new algorithm had a higher false alarm rate but also had a higher correct detection rate. The new algorithm performed significantly better on the dirtier of the two test sequences and achieved a much higher correct detection rate than the Bornard algorithm. This difference in performance occurs because the new algorithm uses a more discriminate method of classifying between blotches PM. Instead of employing maximum caution approach of the Bornard algorithm, the new algorithms tries to make informed guesses of the true classification of each candidate.

7.2 Global Motion Estimation

A review of the global motion estimation state of the art was presented in chapter 4. It introduced the IRLS framework for estimating the motion parameters from either the image intensities or local motion fields. The IRLS method attempts to exclude the influence of local motion by weighting out the local motion regions based on the size of the residual. The limitations of the IRLS framework were discussed and in particular the suitability of using the residual as the means of rejecting local motion. This led to the conclusion that the robustness of the IRLS framework could be improved by using a motion based segmentation to reject local motion, resulting in a hybrid approach to local motion.

The hybrid algorithm was then introduced in the following chapter. It was applied to the generation of mosaics from MPEG-2 sequences. The most significant novelty in the algorithm is the use of the motion based segmentation to reject local motion. This ensures that the robustness to local motion of the histogram techniques is combined with the precision of the least squares techniques. Mosaicking was a good test application of the hybrid technique as errors in the motion parameters will propagate through the mosaic. For this reason, mosaicking demands both robustness and precision of the motion parameters. It was also demonstrated how the algorithm can be used to estimate global motion over long inter-frame differences by aggregating the motion information over the temporal window.

7.2.1 Film Tear Restoration

A new segmentation algorithm for torn frames was introduced in chapter 6. The algorithm allows torn frames to be divided into two regions so that the relative displacement between the two regions can be computed. An interactive segmentation framework is adopted using the Graphs Cut technique. The algorithm uses the DFD to define the boundary conditions of the segmentation, with the frame being cut along the line of the ridge in the DFD surface. A note-worthy strength of the algorithm is its flexibility. It allows for an automated segmentation using a motion based initialisation and, when failure occurs in the initialisation, the algorithm allows for a user defined initialisation. Another feature of the algorithm is that it can proceed solely with the given initialisation and knowledge of the boundary conditions. Thus, the algorithm can be used in many situations where knowledge of the region properties does not readily exist. The example given in this chapter was the detection of dirty splices in archived media.

7.3 Future Work

There is no end in sight in the demand for image processing applications in the digital age. With improvements in technology the demand for applications is only likely to increase, with motion estimation remaining an integral process in many of these applications. Consequently, robustness to motion estimation failure will remain an issue. This section outlines the outstanding issues and possible extensions that can be made to the algorithms described in this thesis and outlines potential applications of these techniques.

Pathological Motion Detection

The test results highlighted that the divergence likelihood had a varied impact on the test sequences chosen. In the tests carried out in the thesis it was shown that the inclusion of the motion field smoothness information in the framework helps to reduce the false alarm rate on one of test sequences while not having any affect on the second test sequence. Further testing to obtain a proper evaluation of the likelihood is required. This would require testing on more sequences exhibiting a wide range of PM patterns and testing using a number of different motion estimation algorithms. However, the main limitation of the algorithm is that it assumes that motion compensation can always be performed in a reliable manner, even in the presence of PM. Obviously, this cannot be the case and hence such methods can never detect all of the harmful Pathological Motion. In that sense the problem is a “chicken and egg” type problem. The goal is to detect PM but all PM cannot be detected due to the presence of PM. However, using the approach outlined here it has been shown that the damage to image data caused by PM can be dramatically reduced.

The logical application for the new blotch detector would be to integrate it into an integrated missing data treatment framework. The integrated frameworks recognise that by treating the blotch detection and blotch interpolation and motion estimation process in an integrated manner rather than as stand-alone stages improvement is possible in the overall performance of the overall process. A suitable example of an integrated technique is the JONDI algorithm [57].

The Hybrid Global Motion Estimation Algorithm

The biggest outstanding issue to be addressed with the hybrid algorithm how to deal with cases when the entire background is ill-conditioned. This problem can be seen to affect the mosaics of the football sequences shown in chapter 5, which contain backgrounds with very little texture. Another mode of failure occurs when foreground objects become more prominent in a frame than the background, a problem that affects the mosaic of the cricket sequence. The obvious solution would be to interpolate the motion parameters temporally from frames where the motion parameters can be estimated reliably. Such an approach was considered by Kelly in his PhD thesis [49] who used a particle filter to predict the motion parameters of each frame.

Mosaicking of sports sequences is an interesting application of the hybrid algorithm. It has been shown in this work it is possible to generate mosaics from only a fraction of the frames in a sequence if knowledge of the motion information over the sequence exists. This raises the prospect of quick estimation of good quality mosaics. These mosaics can be used as a means of summarising the action of an entire shot in a single image, possibly in the form of a motion history image [7,30,63].

Film Tear Restoration

At the present stage of development of the tear restoration framework, reliable algorithms exist to fully restore torn frames given a frame that is known to be torn. These stages include the segmentation algorithm outlined in this thesis, a relative tear displacement correction stage which is followed by an image stabilisation stage and a missing data treatment stage. The only outstanding stage in the framework that needs to be addressed is the torn frame detection stage. An algorithm to achieve this was mooted in [21]. It proposed that histograms of spatial gradient be used to diagnose torn frames as torn frames are high gradient features in an image. However, further testing of this algorithm is required to verify its correctness.

There is also further room for improvement in displacement correction algorithm. Currently the translational motion of each segmented region is estimated from a histogram of the local motion field. Obviously the quality of the estimated displacement is reliant on the quality of the motion field and a more elaborate motion model should be used, the ideal compromise being the six parameter affine model. In global estimation algorithms these limitations are straightforward to overcome. However, there is an additional drawback in the tear displacement algorithm in that the region size is smaller than the entire frame. Consequently, it is more difficult to get reliable estimates of the regional displacement with elaborate motion models.

7.4 Final Thoughts

The problems affecting both local and global motion estimation have much in common. Pathological Motion occurs when local motion estimators cannot model the motion at a given pixel or block. This can occur due to natural phenomena such as occlusion or if the resolution of the motion field is too coarse to fully represent the motion of the object (*i.e.* in regions of motion discontinuity). The presence of local motion is the equivalent problem in global motion estimation. The difference being that the block is now the entire image which contains within it, with one motion pattern corresponding to the global motion of the image and other motion patterns corresponding to the various local motions. In global motion estimation local motion is rejected by the selection of a suitable M-estimator. A similar approach has been used in local motion estimation algorithm of Black and Anandan [6] to model spatial discontinuities in the motion field.

Both processes also suffer from problems of ill-conditioning (*e.g.* the aperture effect in local motion estimation, textureless regions in global motion estimation). As global motion estimation uses more image data to estimate the motion parameters (*i.e.* an entire image rather than a small block), it is generally more robust to ill-conditioning. In both global and local motion estimators, stability to ill-conditioning can be improved by increasing the damping coefficient in the Levenberg-Marquardt framework (*e.g.* [31] in local motion estimation and [59] in global motion estimation). Beyond this, a guess for the motion of ill-conditioned blocks by interpolation from well-conditioned neighbours. In local motion estimation the motion information can be interpolated spatially from neighbouring blocks [54]. As spatial interpolation of motion information is not possible in global motion estimation, the motion parameters must be interpolated temporally [49].

In closing, it appears that motion estimation development should be at the stage where the effort is concentrated on real phenomena that have always concentration estimators. The notion of handling motion blur [80] and transparent motion [93] holds much material for future work.

A
Vector Field Aggregation

In the hybrid algorithm, Global Motion is estimated between consecutive I-frames. In order to achieve this, a local motion field between the two frames must be constructed from the motion information encoded in the MPEG stream [25]. The most important feature of MPEG streams is that there are three picture types:

- I-frame which has no motion information.
- P-frame which has a motion field for prediction (forward motion compensation)
- B-frame which has two motion fields for prediction (backward and forward motion compensation)

Fig. A.1 represents a Group Of Pictures (GOP) profile and indicates how motion information in the MPEG stream can be used to construct one motion vector field between two I-frames. The motion field is constructed in two phases. Firstly, each P-frame motion field (blue line) is accumulated into a single field. Since the motion fields are block based and the vectors are given in sub-block precision, the predicted location of the corresponding macroblock can overlap with up to four macroblocks in the previous frame. Therefore the vector at these positions are calculated using an average of the overlapping blocks weighted by the respective overlapping area of each block. In order to construct the motion field from the last P-frame to the end of the GOP, the bidirectional information (green line) of the last B-frames is added together (with a negative sign for backward prediction data) and is aggregated with the motion field for the rest of the GOP, resulting in a motion field describing the combined motion over the GOP.

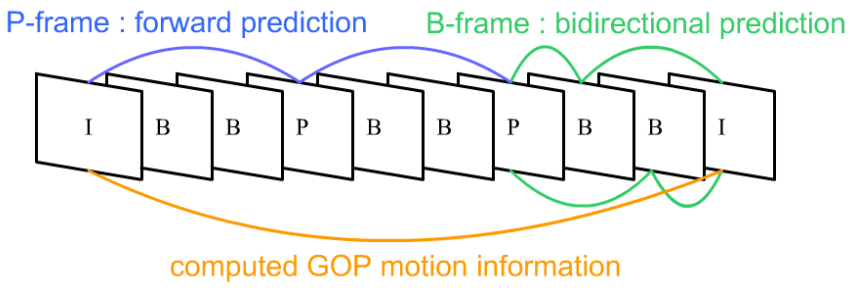

P-frame : forward prediction
B-frame : bidirectional prediction
I
B
B
P
B
B
P
B
B
I
computed GOP motion information

Figure A.1: *Motion prediction scheme for a GOP. The blue lines represent the motion information for the P-frames, a motion prediction field from the previous P(or I)-frame to the current P-frame. The green lines outline the prediction scheme for the B-frames. Each B-frame is associated with a prediction field from the previous and next P-frame or I-frame. Aggregation of these fields is needed to compute the motion field between two consecutive I-frames (yellow line).*

A.1 Texture Discrimination

Blocks which contain no texture information are unreliable for motion estimation. It is possible to use the DCT coefficients of an 8×8 image block to generate indicators of horizontal and vertical texture, as presented in [35]. This yields an efficiently generated weighting directly from information present in the MPEG-2 stream. *Eq. A.1* gives the horizontal (g_x) and vertical (g_y) gradient indicators of a DCT block B.

$$g_x(B) = \sum_{i=1}^{N} B(0,i)^2 \quad g_y(B) = \sum_{i=1}^{N} B(i,0)^2 \tag{A.1}$$

In fact, there is one motion vector (16×16 pel block) (or two if odd and even lines are separately compensated) for four DCT blocks (8×8) (which can contain separated odd or even lines). By combining all DCT blocks (B_1, B_2, B_3, B_4) into a single block (B) [33], an equation to compute gradient indicators,G_x and G_y, related to the four DCT blocks can be given by

$$\begin{aligned} G_x(B) = &\frac{1}{2}\left(\frac{g_x(B_1+B_3)}{4} + \frac{g_x(B_2+B_4)}{4}\right) \\ &+ \frac{1}{4}\left(\frac{B_1(0,0)+B_3(0,0)}{2} - \frac{B_2(0,0)+B_4(0,0)}{2}\right)^2 \end{aligned} \tag{A.2}$$

$$\begin{aligned} G_y(B) = &\frac{1}{2}\left(\frac{g_y(B_1+B_2)}{4} + \frac{g_y(B_3+B_4)}{4}\right) \\ &+ \frac{1}{4}\left(\frac{B_1(0,0)+B_2(0,0)}{2} - \frac{B_3(0,0)+B_4(0,0)}{2}\right)^2 \end{aligned} \tag{A.3}$$

The G_x and G_y indicators are then used to weight the aggregated vector field in the RAG-MOD algorithm. Precise details of how the weighting is applied is given in the following chapter.

B

Initialisation of the Hybrid Global Motion Estimator using the RAGMOD algorithm

This algorithm was first presented in [24] and estimates a four-parameter affine global motion model from a local motion field. Although in this work motion fields from MPEG-2 streams are used, the algorithm can be applied to any type of motion field. The algorithm uses histograms to estimate zoom, rotation and translation parameters and is robust to local motion. It is also much less computationally intensive than the Hough transform methods because instead of estimating every possible set of motion parameters from a single motion vector, a neighbourhood of vectors is used to estimate the only possible set of global motion parameters which describes the vectors. Additionally, the parameters are estimated independently (a histogram each for zoom, rotation and translation) and so the dimensionality of the histogram space is reduced. The algorithm was extended to a six-parameter affine model in [26]. Although this approach gives more precise results than the four parameter case, it is not as robust. Due to this robustness issue, the four parameter implementation of the this approach is used in the hybrid algorithm.

[26] also introduces the motion based segmentation algorithm which is used here to perform background/foreground segmentation. The segmentation exploits the translation histogram used in the GME algorithm to generate a coarse segmentation. Although the segmentation is coarse, its accuracy is not reliant on the accuracy of the global motion estimate. This is because, unlike the IRLS approaches, the segmentation is not based on a residual error measure calculated from the image intensity data. The following sections briefly describe both the global estimation algorithm and the motion based segmentation algorithm.

B.1 Global Motion Estimation

The global motion is estimated, using a 4-parameter model which includes parameters for zoom, rotation and translation, from a motion vector field $V(x, y)$ where (x, y) are the spatial co-ordinates of the motion vector. It is achieved by modeling motion fields as a linear combination of pure zoom, rotation and translation motion fields as follows

$$V(x,y) = \begin{bmatrix} d_x \\ d_y \end{bmatrix} + z. \begin{bmatrix} x \\ y \end{bmatrix} + r. \begin{bmatrix} -y \\ x \end{bmatrix} \tag{B.1}$$

where d_x and d_y are the horizontal and vertical components of the translation and where z and r are the zoom and rotation parameters respectively. Rearranging *Eq. B.1* it can be seen that

$$\begin{aligned} z &= \frac{\partial V_x}{\partial x} = \frac{\partial V_y}{\partial y},\ r = -\frac{\partial V_x}{\partial y} = \frac{\partial V_y}{\partial x}, \\ d_x &= V_x - z.x + r.y, \\ \text{and } d_y &= V_y - z.y - r.x\ . \end{aligned} \tag{B.2}$$

where V_x and V_y are the horizontal and vertical components of the vector field. The algorithm proceeds by using *Eq. B.2* to generate estimates of the parameters from the vector field obtained from the MPEG stream. From the expressions of d_x and d_y, it can be seen that zoom and rotation must be compensated for before the translation parameters can be calculated. z and r are calculated by generating histograms of the appropriate partial derivatives of the vector field. The contribution of each block to the histogram is given by a gaussian distribution weighted according its confidence value. Here V_x and V_y are weighted according to the horizontal (G_x) and vertical (G_y) gradient indicators respectively. In order to overcome the problem of quantisation [1], a second order regression centred about the mode of the each histogram is preformed and the estimated values of z and r are given by the maxima of these regressions

The translation parameters are then estimated by first compensating $V(x, y)$ for zoom and rotation and then by computing a 2D histogram of the compensated field. Once more, contributions from each block are given by a (2-D) gaussian distribution and are weighted according to the texture information of the block. The weights are given by the lesser gradient indicator (G_x and G_y) of each block. A 2-D second order regression is carried out centred on the mode of the histogram and the values of d_x and d_y are given by the coordinates of the maxima. A full flow chart of the algorithm is given in Fig. B.1.

This result is equivalent to the six parameter affine motion model where the affine matrix A and translation $\mathbf{d}$ are given by $A = [1 + z\ \ -r;\ r\ 1 + z]$ (matlab notation) and $\mathbf{d} = [d_x;\ d_y]$.

[1] MPEG vectors are typically given in half-pixel precision

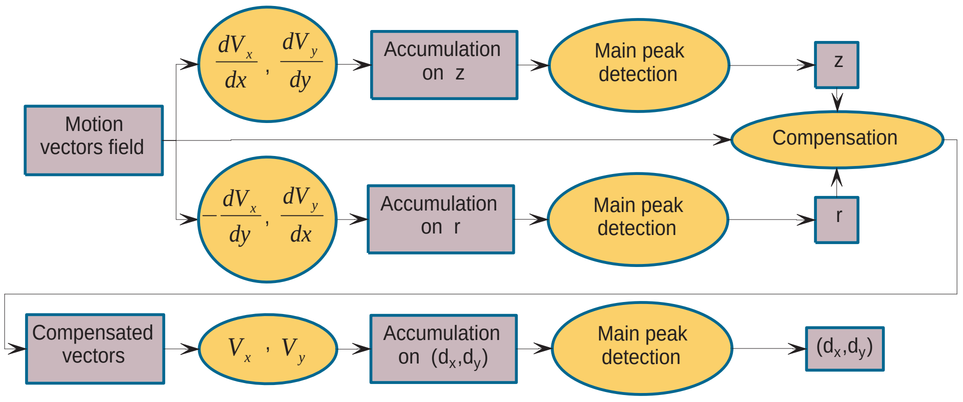


Figure B.1: *The non-parametric Global Motion Estimation Algorithm.*

B.2 Motion-based segmentation

The motion segmentation algorithm segments the vectors in $V(x, y)$ into those contributing to each motion pattern in the motion field. This is performed by using a watershed segmentation [105] on the 2D translation histogram to extract the major peaks from the histogram. Each peak in the histogram describes a different motion pattern which includes both the background motion and local motions in the field. An example of the peak extraction is shown in Fig. B.2. The motion field is then segmented according to the peaks to which each vector contributes. The background is segmented by selecting the segment whose vectors contribute to the modal peak of the translation histogram, resulting in a binary field, $b(\mathbf{x})$, where a value of 1 indicates that site $\mathbf{x}$ is in the background. Fig. B.3 shows an example of the segmentation algorithm applied to an image. In addition [26] proposes an alternative peak extraction technique to the standard watershed method. This method takes into account the sparsity of the histogram space in order to optimise the extraction process.

In contrast to many other state of the art algorithms, the motion-based segmentation has the advantage of being non-recursive. In other words, a global motion estimate is not needed to estimate the segmentation. In recursive segmentation algorithms such as IRLS [32, 79], the first global motion estimation can easily be biased by local motion if the local motion is significant in the image. Local motion rejection based on this first estimation would then be erroneous. To achieve a good segmentation with the motion based segmentation, the background of the image must only be associated with the most prevalent motion in the image (*i.e.* represented by the mode in translation histogram) and not necessarily associated with the size of the error between each motion vector and the estimated global motion.

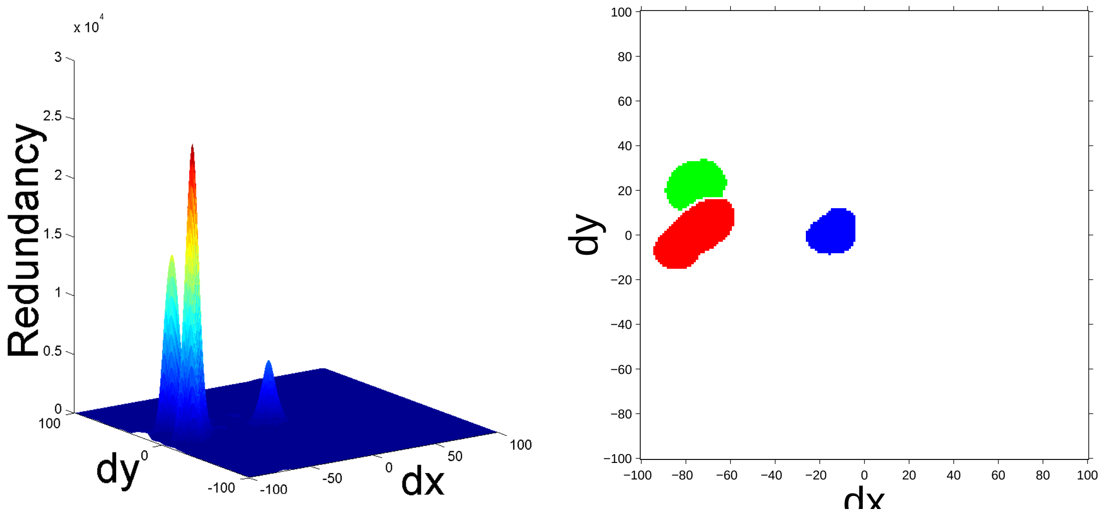


Figure B.2: *This figures shows an example of the histogram peak extraction. The graph on the left shows a typical 2-D translation histogram generated by the GME algorithm. The graph on the right shows the results of the peak extraction. The colours correspond to each extracted peak.*

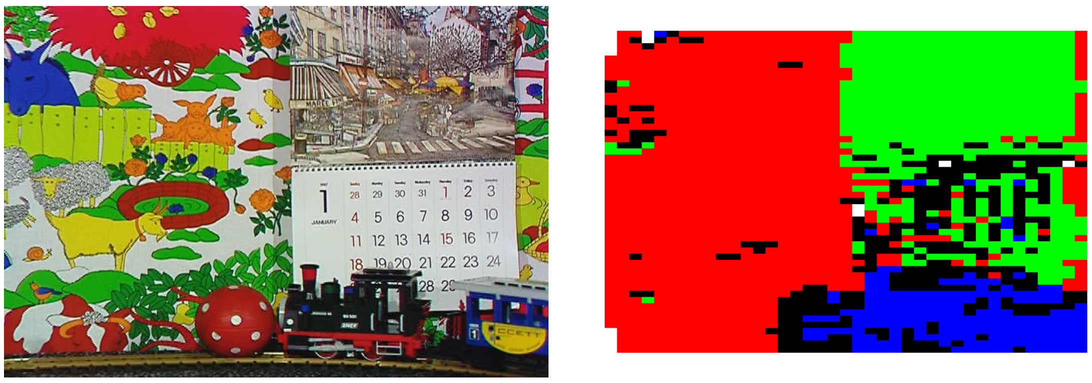


Figure B.3: *This figures shows an example of the segmentation algorithm applied to a frame (shown on the left) from the "mobile and calender" sequence. In this sequence the global motion comes from a camera pan to left. The calender in this sequence is undergoing local motion (moving downwards) and the train and ball is also undergoing local motion. The image on the right shows the results of the motion based segmentation. The red segment corresponds to the motion of the background and the other colours correspond to the various local motions.*

C
The Number of Samples required for Robust Least Median of Squares Regression

For a robust estimate to be obtained, the least Median of Squares estimate must correspond to a candidate estimate calculated from data containing no outliers. It is assumed that the median square error of a candidate containing an outlier is always greater than a candidate uncontaminated by outliers. Thus for a robust estimate to exist, it is necessary for at least one candidate to correspond from a sample consisting of inliers only.

The outlier probability of a data element is known and is given by ϵ say. Hence the probability of the data point being an inlier is $1 - \epsilon$. When estimating a k-parameter global motion model from a 2D vector field, exactly half the number of data elements (*i.e.* vectors) are required. There the probability of a data sample being outlier free is

$$P(\text{sample has no outliers}) = (1-\epsilon)^{k/2} \tag{C.1}$$

and hence

$$P(\text{sample contains outliers}) = 1 - (1-\epsilon)^{k/2}. \tag{C.2}$$

If the number of candidates to be estimated is m, then the probability of one of the candidates being outlier free can be estimated by considering a binomial distribution aas follows

$$\begin{aligned} p = P(\text{at least one sample outlier free}) &= 1 - P(\text{no sample outlier free}) \\ &= 1 - \frac{m!}{0!(m-0)!}\big((1-\epsilon)^{k/2})\big)^0\big(1-(1-\epsilon)^{k/2})\big)^{m-0} \\ &= 1 - \big(1-(1-\epsilon)^{k/2}\big)^m. \end{aligned} \tag{C.3}$$

Rearranging the above equation, the minimum number of samples required for a required probability of robustness to outliers of p is given by

$$m = \frac{\log(1-p)}{\log(1-(1-\epsilon)^{k/2})}. \tag{C.4}$$

Bibliography

[1] D. H. Ballard. Generalized hough transform to detect arbitrary patterns. *IEEE Transactions on pattern Analysis and Machine Intelligence (PAMI)*, 13(2), 1981.

[2] J. E. Besag. Spatial interaction and the statistical analysis of lattice systems. *Journal of the Royal Statistical Society B*, 36:192–236, 1974.

[3] J. E. Besag. On the statistical analysis of dirty pictures. *Journal of the Royal Statistical Society B*, 48:259–302, June 1986.

[4] J. Biemnond, L. Looijenga, D. E. Boekee, and R. H. J. M. Plompen. A pel-recursive wiener based displacement estimation algorithm. *Signal Processing*, 13:399–412, 1987.

[5] J. Biemond, P. M. B. van Roosmalen, and R. L. Lagendijk. Improved blotch detection by postprocessing. In *IEEE International Conference on Acoustics and Signal Processing*, pages 3101–3104, March 1999.

[6] M. J. Black and P. Anandan. A framework for the robust estimation of optical flow. In *IEEE International Conference on Computer Vision*, pages 231–236, 1993.

[7] A. F. Bobick and J. W. Davis. The recognition of human movement using temporal templates. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23:257–267, 2001.

[8] R. Bornard. *Probabilistic Approaches for the Digital Restoration of Television Archives.* PhD thesis, Ecole Centrale Paris, 2002.

[9] Implementation of max-flow/min-cut algorithm. http://www.csd.uwo.ca/~yuri/Implementations/maxflow-v3.0.src.tar.gz.

[10] Y. Boykov and G. Funka-Lea. Graph cuts and efficient n-d image segmentation. *International Journal of Computer Vision (IJCV)*, 70:109–131, 2006.

[11] Y. Boykov and M.-P. Jolly. Interactive graph cuts for optimal boundary & region segmentation in n-d images. In *IEEE Internation Conference on Computer Vision*, 2001.

[12] Y. Boykov and V. Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE Transactions on pattern Analysis and Machine Intelligence (PAMI)*, 26, 2004.

[13] R. N. Bracewell, K.-Y. Chang, A. K. Jha, and Y.-H. Wang. Affine theorem for two-dimensional fourier transform. *Electronics Letters*, 29, 1993.

[14] T. K. Chiew, P. Hill, D. R. Bull, and C. N. Canagarajah. Robust global motion estimation using the hough transform for real-time video coding. In *Proc. of Picture Coding Symposium*, 2004.

[15] M. N. Chong, P. Liu, W. B. Goh, and D. Krishnan. A new spatio-temporal mrf model for the detection of missing data in image sequences. In *IEEE International Conference on Acoustics and Signal Processing*, pages 2977–2980, April 1997.

[16] Cinema craft mpeg-2 encoder. www.cinemacraft.com/eng/sp.html.

[17] M. Z. Coban and R. M. Mersereau. Fast rate-constrained n-step search algorithm for motion estimation. In *Proc. of IEEE International Conference on Accoustics, Speech and Signal Processing*, volume 5, pages 2613–2616, 1998.

[18] Autodesk combustion 4. http://www.autodesk.com/combustion.

[19] D. Corrigan, N. Harte, and A. Kokaram. Pathological motion detection for robust missing data treatment in degraded archived media. In *IEEE ICIP*, pages 621–624, Atlanta, USA, 2006.

[20] D. Corrigan, N. Harte, and A. Kokaram. Automated segmentation of torn frames using the graph cuts technique. In *IEEE International Conference on Image Processing (ICIP)*, San Antonio, Texas, USA, September 2007.

[21] D. Corrigan and A. Kokaram. Automated tear treatment in degraded archived media. In *IEEE International Conference on Image Processing*, volume 3, pages 1823 – 1826, 2004.

[22] D. Corrigan and A. Kokaram. Detection and treatment of film tear in degraded archived media. In *IEEE International Conference on Pattern Recognition*, volume 4, pages 779 – 782, 2004.

[23] D. Corrigan, A. Kokaram, R. Coudray, and B. Besserer. Robust global motion estimation from mpeg streams with a gradient based refinement. In *IEEE ICASSP*, volume 2, pages 285–288, Toulouse, France, 2006.

[24] R. Coudray and B. Besserer. Global motion estimation for MPEG-encoded streams. In *Proc. of IEEE Int. Conf. on Image Processing*, Singapore, 2004.

[25] R. Coudray and B. Besserer. Agregation, selection et utilisation de l'information de mouvement issue d'un flux MPEG. In *GRETSI*, 2005.

[26] R. Coudray and B. Besserer. Motion based segmentation using mpeg streams and watershed method. In *Int. Symposium on Visual Computing - Lecture Notes in Computer Sciences*, 2005.

[27] A. Crawford, H. Denman, F. Kelly, F. Pitie, and A. Kokaram. Gradient based dominant motion estimation with integral projections for real time video stabilisation. In *IEEE International Conference on Image Processing*, Singapore, 2004.

[28] R. Dahyot and A. Kokaram. Comparison of two algorithms for robust m-estimation of global motion parameters. In *Irish Machince Vision and Image Processing Conference (IMVIP)*, Dublin, Ireland, 2004.

[29] P. Delacourt, A. Kokaram, and R. Dahyot. Comparison of global motion estimators. In *Irish Signals and Systems Conference*, Cork, Ireland, 2002.

[30] H. Denman, N. Rea, and A. Kokaram. Content based analysis for video from snooker broadcasts. *Journal of Computer Vision and Image Understanding, Special Issue on Video Retrieval and Summarization*, 92:176–195, 2003.

[31] J. Driessen, L. Boroczky, and J.Biemond. Pel-recursive motion field estimation from image sequences. *Visual Communication and Image Representation*, 2:259–280, 1991.

[32] F. Dufaux and J. Konrad. Efficient, robust and fast global motion estimation for video coding. *IEEE Transactions on Image Processing*, 9, 2000.

[33] R. Dugad and N. Ahuja. A fast scheme for image size change in the compressed domain. *IEEE Trans. on Circuits and Systems for Video Technology*, 11:461–474, April 2001.

[34] A. A. Efros and T. K. Leung. Texture synthesis by non-paramteric sampling. In *IEEE International Conference on Comupter Vission (ICCV)*, pages 1033–1038, September 1999.

[35] R. E. Frye and R. S. Ledley. Texture discrimination using discrete cosine transformation shift-insensitive (DCTSIS) descriptors. *Pattern Recognition*, 33(10):1585–1598, 2000.

[36] C. Gallagher and A. Kokaram. Non-parametric wavelet-based texture synthesis. In *IEEE International Conference on Image Processing (ICIP)*, volume 2, pages 462–465, September 2005.

[37] S. Geman and D. Geman. Stochastic, relaxation, gibbs distributions, and the bayesian restoration of images. *IEEE Transactions on Pattern Analysis and machine Intelligence*, 6, November 1984.

[38] M. Ghanbari. The cross-search algorithm for motion estimation. *IEEE Transactions on Communications*, 38:950–953, March 1990.

[39] D. Greig, B. Porteous, and A. Seheult. Exact maximum a posteriori estimation for binary images. *Journal of the Royal Statistical Society B*, 52:271–279, 1989.

[40] H.264, advanced video coding for generic audiovisual services. ITU-T Recommendation H.264, 2003.

[41] F. Heitz, P. Perez, and P. Bouthemy. Multiscale minimization of global energy functions in some visual recovery problems. *CVGIP: Image Understanding*, 59:125–134, January 1994.

[42] L. Hill and T. Vlachos. On the estimation of global motion using phase correlation for broadcast applications. In *Proc. of IEE International Conference on Image Processing and its Applications*, volume 2, pages 721–725, 1999.

[43] P. J. Huber. *Robust Statistics.* John Wiley and Sons, 1981.

[44] Institut national de l'audiovisuel. http://www.ina.fr/index.en.html.

[45] Intel performance primitive (ipp) libraries. http://www.intel.com/cd/software/products/asmo-na/eng/302910.htm.

[46] A. K. Jain and J. R. Jain. Radar image modelling and processing for real-time rf simulation. Technical report, Department of Electronic Engineering, State University of New York Buffalo, 1978.

[47] H. Jozawa, K. Kamikura, A. Sagata, H. Kotera, and H. Watanabe. Two-stage motion compensation using adaptive global mc and local affine mc. *IEEE Transactions on Circuits and Systems for Video Technology*, 7:75–85, Februaury 1997.

[48] J. K. Kearney, W. B. Thompson, and D. L. Boley. Optic flow estimation: An error analysis of gradient-based methods with local optimization. *IEEE transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 9:229–243, March 1987.

[49] F. Kelly. *Fast Probabilistic Inference and GPU Video Processing.* PhD thesis, Trinity College Dublin, 2006.

[50] B. Kent, A. Kokaram, B. Collis, and S. Robinson. Two layer segmentation for handling pathological motion in degraded post production media. In *IEEE Int. Con. on Image Processing (ICIP)*, Singapore, 2004.

[51] S. Kirkpatrick, C. D. Gellat, and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220:671–680, 1983.

[52] A. Kokaram. *Motion Picture Restoration.* PhD thesis, Cambridge University, England, 1993.

[53] A. Kokaram. *Motion Picture Restoration: Digital Algorithms for Artefact Suppression in Degraded Motion Picture Film and Video*, chapter 7. Springer Verlag, 1998.

[54] A. Kokaram. *Motion Picture Restoration: Digital Algorithms for Artefact Suppression in Degraded Motion Picture Film and Video*, chapter 2. Springer Verlag, 1998.

[55] A. Kokaram. *Motion Picture Restoration: Digital Algorithms for Artefact Suppression in Degraded Motion Picture Film and Video*, chapter 6. Springer Verlag, 1998.

[56] A. Kokaram. *Motion Picture Restoration: Digital Algorithms for Artefact Suppression in Degraded Motion Picture Film and Video.* Springer Verlag, 1998.

[57] A. Kokaram. On missing data treatment for degraded video and frame archives: A survey and a new bayesian approach. *IEEE trans. on Image Processing*, 13:397–414, March 2004.

[58] A. Kokaram, B. Collis, and S. Robinson. Practical motion based video matting. In *proceedings of the IEE European Conference on Visual Media Production (CVMP'05)*, pages 130–136, London, UK, November 2005.

[59] A. Kokaram and P. Delacourt. A new global motion estimation algorithm and its application to retrieval in sports events. In *IEEE International Workshop on Multimedia Signal Processing*, Cannes, France, 2001.

[60] A. Kokaram and S. Godsill. Mcmc for joint noise reduction and missing data treatment in degraded video. *IEEE trans. on Signal Processing*, 50:189–205, February 2002.

[61] A. Kokaram, R. Morris, W. J. Fitzgerald, and P. Rayner. Detection of missing data in image sequences. *IEEE Transactions on Image Processing*, 4:1496–1508, November 1995.

[62] A. Kokaram, R. Morris, W. J. Fitzgerald, and P. Rayner. Interpolation of missing data in image sequences. *IEEE Transactions on Image Processing*, 4:1509–1519, November 1995.

[63] A. Kokaram, F. Pitie, R. Dahyot, N. Rea, and S. Yeterien. Content controlled image representation for sports streaming. In *Fourth International Workshop on Content-Based Multimedia Indexing*, 2005.

[64] V. Kolmogorov and Y. Boykov. What metrics can be approximated by geo-cuts or global optimisation of length/area and flux. In *IEEE International Conference on Computer Vision*, Beijing, China, 2005.

[65] J. Konrad and E. Dubois. Bayesian estimation of motion vector fields. *IEEE trans.on Pattern Analysis and Machine Intelligence*, 14, September 1992.

[66] C. D. Kuglin and D. C. Hines. The phase correlation image alignment method. In *IEEE International Conference on Cybernetics and society*, pages 163–165, 1975.

[67] W. Li and E. Salari. Successive elimination algorithm for motion estimation. *IEEE Transactions on Image Processing*, 4:105–107, January 1995.

[68] Y. Li, L. Q. Xu, G. Morrison, C. Nightingale, and J. Morphett. Robust panorama from mpeg video. In *Proc. of IEEE International Conference on Multimedia and Expo*, volume 1, pages I–81–4, 2003.

[69] L. Marcenaro, G. Vernazza, and C. Regazzoni. Image stabilization algorithms for video-surveillance applications. In *IEEE International Conference on Image Processing*, volume 1, pages 349–352, Thessaloniki, Greece, 2001.

[70] D. M. Martinez. *Model-Based Motion Estimation and its Application to Restoration and Interpolation of Motion Pictures*. PhD thesis, Massachusetts Institute of Technology, 1986.

[71] L. McManus. Algorithms for robust digital video transmission using MPEG-4. Master's thesis, University of Dublin, Trinity College, 2002.

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

[73] R. D. Morris. *Image Sequence Distribution using Gibbs Distributions*. PhD thesis, Cambridge University, 1995.

[74] Mpeg-4, coding of moving pictures and audio. ISO/IEC JTC1/SC29/WG11 14496-2:Visual, 1998.

[75] M. J. Nadenau and S. K. Mitra. Blotch and sratch detection in image sequences based on rank order differences. In *5th International Workshop on Time-Varying Image Processing and Moving Object Recognition*, September 1996.

[76] A. N. Netravali and J. D. Robbins. Motion-compensated television coding: Part i. *The Bell System Technical Journal*, 58:631–670, March 1979.

[77] H. Nicholas. New methods for dynamic mosaicing. *IEEE Transactions on Image Processing*, 10:1239–1251, August 2001.

[78] Motion 2d. http://www.irisa.fr/vista/Motion2D/.

[79] J. M. Odobez and P. Bouthemy. Robust multiresolution estimation of parametric motion models. *Journal of Visual Communication and Image Representation*, 6, 1995.

[80] O. J. Omer, S. Kumar, R. Bajpai, K. S. Venkatesh, and S. Gupta. Motion estimation for motion smear - a system identification approach. In *IEEE International Conference on Image Processing (ICIP)*, volume 3, pages 1855–1858, 2004.

[81] M. Pilu. On using raw mpeg motion vectors to determine global camera motion. In *SPIE Visual Communications and Image Processing*, volume 3309, pages 448–459, 1998.

[82] W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling. *Numerical Recipies in C: The Art of Scientific Computing.* Cambridge University Press, 1992.

[83] Prestospace: An integrated solution for audio-visual preservation and access. http://www.prestospace.org/.

[84] A. Rares. *Archived Film Analysis and Restoration.* PhD thesis, Delft University of Technology, 2004.

[85] A. Rares, M. J. T. Reinders, and J. Biemond. Complex event classification in degraded image sequences. In *IEEE Int. Conf. on Image Processing*, Thessaloniki, Greece, 2001.

[86] A. Rares, M. J. T. Reinders, and J. Biemond. Statistical analysis of pathological motion areas. In *IEE Seminar on Digital Restoration of Film and Video Archives*, London, UK, 2001.

[87] A. Rares, M. J. T. Reinders, and J. Biemond. Edge-based image restoration. *IEEE Transactions on Image Processing*, 14:1454–1468, October 2005.

[88] P. Read and M.-P. Meyer, editors. *Restoration of motion picture film*, pages 91–94. Butterworth-Heinemann series in conservation and museology. Butterworth-Heinemann, 2000.

[89] J. Ren and T. Vlachos. Dirt detection for archive film restoration using an adaptive spatio-temporal approach. In *IEEE International Conference on Acoustics and Signal Processing*, pages 3101–3104, March 1999.

[90] P. J. Rousseeuw. Least median of squares regression. *Journal of the American Statistical Association*, 79:871–880, 1984.

[91] H. Sawhney and S. Ayer. Compact representation of video through dominant and multiple motion estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18, August 1996.

[92] P. Schallauer, A. Pinz, and W. Haas. Automated restoration algorithms for 35mm film. *Journal of Computer Vision Research*, 1(3):59–85, 1999.

[93] M. Shizawa and K. Mase. Simultaneous multiple optical flow estimation. In *IEEE International Conference on Pattern Recognition (ICPR)*, volume 1, pages 274–278, 1990.

[94] A. Smolic, M. Hoeynck, and J. R. Ohm. Low-complexity global motion estimation from p-frame motion vectors for mpeg-7 applications. In *Proc. of IEEE International Conference on Image Proceesing*, volume 2, pages 271–274, 2000.

[95] A. Smolic and J. R. Ohm. Robust global motion estimation using a simplified m-estimator approach. In *Proc. of IEEE International Conference on Image Proceesing*, 2000.

[96] A. Smolic, T. Sikora, and J. R. Ohm. Long-term global motion estimation and its application for sprite coding, content description and segmentation. *IEEE Transactions on Circuits and Systems for Video Technology*, 9:1227–1242, December 1999.

[97] C. Stiller and J. Konrad. Estimating motion in image sequences. *IEEE Signal Processing Magazine*, 16:70–91, July 1999.

[98] R. Storey. Electronic detection and concealment of film dirt. *SMPTE Journal*, pages 642–647, June 1985.

[99] R. Szeliski. Image mosaicing for tele-reality. Technical Report 94/2, Digital Equipment Corporation, Cambridge Research Lab, Cambridge MA, May 1994.

[100] A. M. Tekalp. *Digital Video Processing*. Signal Processing Series. Prentice-Hall, 1995.

[101] G. A. Thomas. Television motion measurement for datv and other applications. Technical Report BBC-RD-1987-11, BBC Research Department, 1987.

[102] Y. T. Tse and R. L. Baker. Global zoom/pan estimation and compensation for video compression. In *Proc. of IEEE International Conference on Accoustics, Speech and Signal Processing*, pages 2725–2728, 1991.

[103] P. M. B. van Roosmalen. High-level analysis of image sequences. Technical report, INA - Paris, 1999.

[104] P. M. B. van Roosmalen. *Restoration of Archived Film and Video*. PhD thesis, Delft University of Technology, 1999.

[105] L. Vincent and P. Soille. Watersheds in digital spaces: An efficient algorithm based on immersion simulations. *IEEE transactions on Pattern Analysis and Machine Intelligence*, 13:583–598, 1991.

[106] J. Watkinson. *The MPEG handbook : MPEG-1, MPEG-2, MPEG-4*. Focal Press, 2001.