Discontinuity Analysis for Video Processing

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

Hugh Denman
Trinity College Dublin, July 2007

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

To my family.

Abstract

This thesis is concerned with analysis of digital video through discontinuity detection. A discontinuity here is a sudden change that corresponds to some meaningful event in the video content.

The paradigm of discontinuity-based analysis is applied within three distinct contexts. Firstly, the problem of shot-change detection is explored. This amounts to the detection of signal-level discontinuities. An extensive review and bibliography of existing methods proposed for shot change detection are presented.

The next area of activity focuses on discontinuities as events, specifically in snooker video. It is described how many of the significant events within televised snooker play can be detected by targeted discontinuity detection, exploiting the characteristics of sports footage. Techniques for detecting ball pots and near misses, for summarising the episodes of play, and for tracking individual balls are described.

Finally, the detection of discontinuities in local motion is examined. A number of methods for the description of local motion in video are evaluated. The detection of extreme values in these descriptions is established as a powerful means for edit point detection in dance video. As part of this work, a Bayesian video segmentation algorithm is designed which incorporates new refinements for maintaining temporal coherence in the segmentation over motion discontinuities.

Successful edit point detection in dance video is shown to enable a number of novel applications, including the resynchronisation of the dance video to a new music signal. A DVD demonstrating some results of this technique is included.

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,

Hugh Denman

July 2, 2007.

Acknowledgments

After a long period of immersive study, I think it fair to claim that I am at present the leading expert on the enduringly collegiate atmosphere of the Sigmedia group; my considered pronouncement is that you couldn't hope to work with a better bunch of people. I particularly want to thank Dr François Pitié, Dr Francis Kelly, and Andrew Crawford for the pleasure of working with them; Dr Claire Catherine Gallagher and Dr Naomi Harte for their invaluable help in preserving my sanity in the final days; and Deirde O'Regan, Dan Ring, and Daire Lennon for the enlivened atmosphere that attends them everywhere.

I have been generously funded in my research career by the Enterprise Ireland projects MUSE-DTV (Machine Understanding of Sports Events for Digital Television) and DysVideo, and the European Commission Framework projects BRAVA (BRoadcast Archives through Video Analysis) and PrestoSpace (PREServation TechnOlogy: Standardised Practices for Audiovisual Contents in Europe). The recent funding offered to the group by Adobe Systems Incorporated has provided for equipment that proved instrumental to the completion of the work described here. My thanks to all of these bodies.

I am profoundly grateful to my supervisor, Dr Anil Kokaram, for his unflagging support and encouragement over more years than he expected. And to my parents and my brother Feargus, for everything.

Contents

List of Acronyms

DCT Discrete Cosine Transform

DFD Displaced Frame Difference

EM Expectation Maximisation

EMD Earth Mover's Distance

GME Global Motion Estimation

HMM Hidden Markov Model

ICM Iterated Conditional Modes

IRLS Iteratively Reweighted Least Squares

LDA Linear Discriminant Analysis

LMM Local Motion Map

LMMSE Linear Minimum Mean Least Square Error

MAD Mean Absolute Difference

MAP Maximum *a posteriori*

MHI Motion History Image

MSE Mean Squared Error

ROC Receiver Operating Characteristic

SVD Singular Value Decomposition

1 Introduction

The digital video revolution began in earnest with the introduction of the DVD format in the mid-1990s. Since that time, vast quantities of digital video have been produced, through the increasing use of digital cinema technology in feature film production, the pervasive adoption of the DVD format for consumer video cameras, and the digitisation of film material for re-release in DVD and HD-DVD / Blu-Ray form.

Tools for manipulating digital video are thus in continuously increasing demand. A number of these software tools for video manipulation are available, and the capacity of these tools for digital video editing is increasing rapidly in line with the ever-expanding power of modern computer systems. However, the *nature* of the operations which are facilitated by these systems has remained essentially static, being limited to operations at the frame level. Manipulation using higher level concepts, such as shots, scene settings, and story arcs, is not available because the content of the video is *opaque* within these systems. The development of *content-aware* systems to address this *semantic gap* is therefore an area of considerable research activity.

This thesis approaches content analysis through *discontinuities*. A discontinuity here is a sudden change that corresponds to some meaningful event in the video content. This paradigm of discontinuity-based analysis is applied within three distinct contexts. Firstly, the problem of shot-change detection is explored. This amounts to the detection of signal-level discontinuities. The next area of activity focuses on discontinuities as events, specifically in snooker video. Here it is described how many of the significant events within televised snooker play can be detected by targeted discontinuity detection, exploiting the characteristics of sports footage. Thirdly, the detection of discontinuities in local motion is examined. It is shown that convincing analyses of

dance video can be created by exploiting these methods.

In each application area, the approach adopted is the extraction of a targeted description of the material, followed by the detection of discontinuities in this description. In shot change detection, the video is described in terms of frame-to-frame similarity; in snooker, the description targets the appearance of the table pocket areas and the ball tracks; and in dance video, the motion characteristics of the foreground region are extracted. In each case, discontinuities in the description of the media are found to correspond to time-points of interest in the video. This approach is applied successfully to feature extraction across a range of semantic levels: low-level in the case of shot changes, an intermediate level in the case of dance footage, and at a high semantic level in the case of snooker video.

1.1 Thesis outline

The remainder of this thesis is organised as follows.

Chapter 2: Image Sequence Modelling

This chapter describes the key ideas underpinning image sequence modelling, outlining the main approaches to global motion estimation, local motion estimation, and video segmentation. A new technique for detecting and recovering from global motion inaccuracies in 'simple' scenes is proposed. 'Simple' here refers to scenes with a well-contained foreground and a near-homogenous background; counter-intuitively, global motion estimation is particularly difficult in such video.

Chapter 3: Video Shot Change Detection: A Review

Here the problem of shot change detection in video is described and analysed in some depth. An extensive review and bibliography of the methods proposed for shot change detection are presented.

Chapter 4: New Approaches to Shot Change Detection

In this chapter, two new contributions to the problem of shot change detection are presented. The first is an approach to cut detection that performs well even in very difficult material. The second is a new dissolve detection scheme based on explicit modelling of the dissolve process. Both issues are considered as Bayesian classification problems.

Chapter 5: Detecting Snooker Events via Discontinuities

The work discussed here constitutes a number of tools facilitating the analysis of broadcast snooker video. It is shown how analysis can be directed towards the most salient areas in the video (the pockets and the balls), and the detection of events in these regions is constructed

in terms of discontinuity detection. This chapter also introduces the use of Motion History Image (MHI) for summarisation of sports events, here snooker shots.

Chapter 6: Low-level Edit Point Detection

This chapter introduces the problem of edit point detection in dance video. An extensive analysis of various low-level approaches to edit point detection is presented, targeting several different descriptors of the motion content of the video. The chapter concludes with a probabilistic framework for fusion of these different descriptors for edit point detection.

Chapter 7: Segmentation For Edit Point Detection

In this chapter, a Bayesian video segmentation system is developed, designed for application to edit point detection in dance video. This application requires strong temporal coherence in the segmentation, particularly at motion discontinuities, and the novel contributions of this work are directed at these aspects.

Chapter 8: Applications of Edit Point Detection

Here three applications of edit point detection in dance video are described: interactive edit point browsing, dance phrase summarisation; and resynchronisation of dance video to new music.

Chapter 9: Conclusions

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

1.2 Contributions of this thesis

The new work described in this thesis can be summarised by the following list:

- A means for detecting and recovering from inaccuracies in global motion estimation in simple scenes
- Refinements to video cut detection particularly appropriate for very difficult sequences
- A new technique for model-based dissolve transition detection
- An algorithm for detecting full-table views in snooker footage
- A perspective-invariant approach to recovering in-game geometry from snooker and tennis video

- An algorithm for detecting and masking out the player, where the player occludes the table, in snooker
- The use of the MHI for summarisation of sports events
- A method for the detection of 'ball pot' and 'near miss' events in snooker based on monitoring of the pockets
- A colour based particle filter approach to ball tracking, with applications to event detection
- The use of motion detection, foreground bounding box analysis, vector field analysis, motion blur analysis, and sound track analysis for the detection of edit points in dance video
- Refinements to video segmentation incorporating the use of the Earth Mover's Distance (EMD) for colour modelling, polar co-ordinates for vector modelling, and motion histories for temporal coherence
- The use of edit points in dance video for interactive keyframe selection and dance phrase summarisation
- The exploitation of edit points in dance video and music beat detection for real-time resynchronisation of dance with arbitrary music

1.3 Publications

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

- "Content Based Analysis for Video from Snooker Broadcasts" by Hugh Denman, Niall Rea, and Anil Kokaram, in *Proceedings of the International Conference on Image and Video Retrieval 2002 (CIVR '02)*, Lecture Notes in Computer Science, vol. 2383, pages 186–193, London, July 2002.
- "Content-based analysis for video from snooker broadcasts" by Hugh Denman, Niall Rea, and Anil Kokaram, in *Journal of Computer Vision and Image Understanding - Special Issue on Video Retrieval and Summarization*, volume 92, issues 2-3 (November - December 2003), pages 141-306.
- "A Multiscale Approach to Shot Change Detection" by Hugh Denman and Anil Kokaram, in *Proceedings of the Irish Machine Vision and Image Processing Conference (IMVIP '04)*, pages 19–25, Dublin, September 2004.
- "Gradient Based Dominant Motion Estimation with Integral Projections for Real Time Video Stabilisation" by Andrew Crawford, Hugh Denman, Francis Kelly, Francois Pitié and

Anil Kokaram, in *Proceedings of the IEEE International Conference on Image Processing (ICIP'04)*, volume V, pages 3371-3374, Singapore, October 2004.

- "Exploiting temporal discontinuities for event detection and manipulation in video streams" by Hugh Denman, Erika Doyle, Anil Kokaram, Daire Lennon, Rozenn Dahyot and Ray Fuller, in *MIR '05: Proceedings of the 7th ACM SIGMM International Workshop on Multimedia Information Retrieval*, pages 183–192, New York, October 2005.

- "Dancing to a Different Tune" by Hugh Denman and Anil Kokaram, in *Proceedings of the IEE European Conference on Visual Media Production (CVMP'05)*, pages 147–153, London, November 2005.

2

Image Sequence Modelling

In this chapter the imaging process is introduced and an overview of image sequence modelling techniques is presented. A new technique for refining global motion estimation in sequences with low background detail is described. For additional material on the essential principles of digital image processing, the reader is referred to the excellent book by Tekalp [298].

2.1 Image Sequence Modelling Overview

Figure 2.1 illustrates the three essential stages in the video process. A camera or other imaging device is directed towards a three-dimensional scene in the real world, and captures a two-dimensional projection of this scene. These images are recorded in succession to some storage

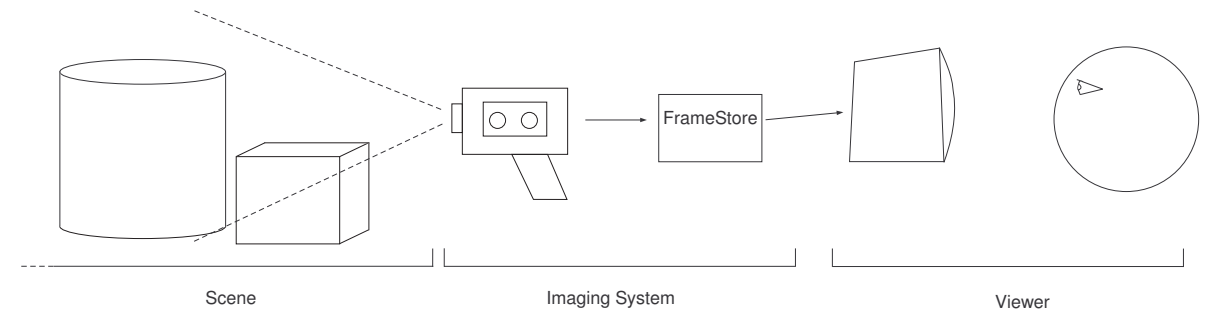


Figure 2.1: Stages in an imaging system

medium. Finally, the images are presented to a viewer through some display technology.

To apply digital signal processing techniques to film, it is necessary that the film be available in digital format. Modern video and film cameras record directly to digital, typically through the use of a charge-coupled device, or CCD, as the light sensor. Early video cameras record by analogue means, such as onto magnetic tape or optical film. This film can subsequently be digitised in a separate stage. In either case, the result is a *digital image sequence*.

A digital image sequence is an ordered set of N rectangular arrays $\{I_n(\mathbf{x}) : 0 \leq n < N\}$. Here n is the temporal index of the image in a sequence, and $\mathbf{x} = (r, c)^T$ denotes the pixel site at row r and column c of the image. For intensity images, $I_n(\mathbf{x})$ is scalar valued; for colour images, $\mathbf{I}_n(\mathbf{x})$ is a vector indicating intensity and colour—for example, a three-tuple representing red, green, and blue light intensities at the site.

For an image sequence arising from the continuous operation of one camera, changes in pixel intensity are caused by four distinct processes. First, changes in the orientation and position of the camera change which portion of the scene is depicted; this is the camera motion process. To an observer, the scene background appears to move relative to the image space. As this perceived motion is coherent across the entire image plane, it is termed *global motion*.

Secondly, individual objects in the field of view of the camera may themselves move; this is designated object motion, or local motion. This motion may be translational or rotational, and can include changes in size and self-occlusion.

Thirdly, changes in the illumination incident on the depicted scene will affect the readout values of the image acquisition device.

Lastly, flaws and aberrations in the acquisition device, the digitisation process, or arising during storage of the sequence, introduce noise to the image sequence.

An image sequence model is a set of hypotheses concerning the nature of the material depicted. From these hypotheses, constraints can be derived. These constraints facilitate analysis of the variations at pixel sites over time in terms of the four processes identified above.

2.1.1 Image-centric sequence modelling

The essential hypothesis underlying image-centric sequence modelling is that the images depict objects that are much larger than the pixel size. Thus adjacent pixels with similar intensity values are likely to depict the same object.

Many image processing applications rely on constraints introduced by this hypothesis. Spatial noise reduction, for example, exploits the constraint that object surfaces should be smooth. Motion estimation generally relies on the constraint that adjacent pixels move identically, as described below.

Motion estimation in image-centric sequence modelling was for many years centred on a model in which motion in the sequence is modelled as deformations of a rubber sheet covering the frame area. While this model has proven effective for simple sequences, it fails to account

for the effects of occlusion and revealing. The layer model proposed by Wang and Adelson [321] supercedes this rubber sheet model in these respects.

The directly perceived motion in a sequence is due to the displacement of spatial gradients. The motion of the gradients is termed optic flow. Motion that does not result in the displacement of gradients cannot readily be extracted from the sequence. For example, if a camera is directed at a rotating smooth sphere under constant illumination, no optic flow results [149]. Optic flow estimation is then the analysis of the motion through gradients.

2.1.2 Scene-centric sequence modelling

The hypotheses underlying scene-centric sequence modelling are in fact a description of the imaging process itself. In other words, scene-centric sequence modelling treats video explicitly as the perspective projection of three-dimensional objects onto the imaging plane. The goal is to recover as much information as possible about the configuration of the objects being projected, in terms of their relative position and size, the lighting conditions obtaining, and their three-dimensional motion. Optic flow estimation is an essential precursor to scene-centric modelling.

The difficulty with this approach is that deriving constraints on the image sequence is very difficult with a scene-centric model. For example, Nagel in [231] illustrates the ambiguity between changes in object depth (motion perpendicular to the image plane) and changes in object size. Further ambiguities arise from illumination effects: changes in the light source and shadows can result in intensity changes that are difficult to distinguish from motion. Even assuming that object size and illumination are constant, recovering the three-dimensional scene from the optic flow involves inherent ambiguities [3, 295, 338].

The scene-centric approach has been more common in the computer vision community than in signal processing, and has resulted in a very considerable body of research [110, 138].

2.1.3 Affective sequence modelling

Inasmuch as scene centric modelling considers real world objects rather than patches of pixels, it is a higher level approach than image-centric modelling. Essentially, scene-centric modelling encompasses both the scene and imaging stages shown in figure 2.1, while image-centric modelling is only concerned with the imaging stage. The next level of video modelling, then, encompasses all three stages, including the human observer.

This affective media processing [136, 331] tries to infer the emotional content of a signal and its likely effect on a human observer, and is still very much in its infancy. This aspect of information processing in general was identified by Warren Weaver as early as 1949 [272]. He suggested that all information processing operates at a technical, semantic, or influential level. In the case of video processing, as in media processing in general, present-day research is principally at some intermediate stage between the technical and semantic levels. As more sophisticated models are introduced to account for more of a signal's meaning in relation to a

human observer, the scope and relevance of computer-assisted media processing will increase to encompass these affective aspects.

2.2 Global Motion

A marked distinction is drawn in the literature on motion estimation between *global motion estimation* and *local motion estimation* [298]. In practical terms, this distinction is well founded. However, from a theoretical standpoint, one is always fundamentally concerned with an image equation of the form

$$I_n(\mathbf{x}) = I_{n-1}(\mathbf{F}(\mathbf{x}, \theta)) + \epsilon(\mathbf{x}) \tag{2.1}$$

The vector function $\mathbf{F}(\mathbf{x})$ represents co-ordinate transformation according to the motion model, parameterized by θ. The term $\epsilon(\mathbf{x})$ represents errors arising from noise in the image sequence, or due to model inadequacy. In this section, motion models $\mathbf{F}(\mathbf{x}, \mathbf{\Theta})$ suitable for global motion are described, along with the means of estimating, or fitting, these models to the data. Section 2.4 section describes local motion models and estimation techniques.

2.2.1 Global motion models

Global motion estimation methods attempt to fit a single, low-order motion model to the image sequence contents as a whole. An accessible discussion of global motion models and their representational capacities has been presented by Mann and Picard [213]. The simplest choice for the form of $\mathbf{F}$ is the translational model, $\mathbf{\Theta} = (dx, dy)$, $\mathbf{F}(\mathbf{x}, \theta) = \mathbf{x} + [dx, dy]^T$. The commonly used affine model is given by $\mathbf{\Theta} = (\mathbf{A}, \mathbf{d})$, $\mathbf{F}(\mathbf{x}, \theta) = \mathbf{A}\mathbf{x} + \mathbf{d}$. Here, $\mathbf{A}$ is a 2×2 transformation matrix accounting for scaling, rotation, and shear; $\mathbf{d} = [dx, dy]^T$ as before. This is the model used throughout this thesis for global motion estimation. Higher-order models, including bilinear and biquadratic models, can also be applied. These models are linear in that they can be written in the form

$$\mathbf{F}(\mathbf{x}, \mathbf{\Theta}) = B(\mathbf{x})\mathbf{\Theta} \tag{2.2}$$

where $B(\mathbf{x})$ is a matrix-valued function of $\mathbf{x}$. For example, in the case of the affine model,

$$\begin{aligned} B(\mathbf{x}) &= \begin{bmatrix} x_1 & x_2 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x_1 & x_2 & 1 \end{bmatrix} \\ \mathbf{\Theta} &= [a_1, a_2, d_1, a_3, a_4, d_2]^T \end{aligned} \tag{2.3}$$

where a_i, d_i, and x_i are the components of $\mathbf{A}$, $\mathbf{d}$, and $\mathbf{x}$.

The eight-parameter projective transform takes

$$\mathbf{F}(\mathbf{x}, \mathbf{\Theta}) = \frac{\mathbf{A}\mathbf{x} + \mathbf{d}}{\mathbf{c}^T\mathbf{x} + \mathbf{1}} \tag{2.4}$$

This model can exactly account for all changes due to camera motion, assuming that the scene contents are effectively planar (the zero-parallax assumption). However, this is a non-linear transform[1], and as such more difficult to solve.

2.2.2 Global motion estimation

Once a model for the global motion has been chosen, it is necessary to estimate the parameters that best account for the observed image data. Global motion estimation is normally carried out between a pair of frames I_n, I_{n-1}. The error associated with some parameter values $\mathbf{\Theta}'$ is assessed via the Displaced Frame Difference (DFD)

$$\mathrm{DFD}_{\mathbf{\Theta}}(\mathbf{x}) = I_n(\mathbf{x}) - I_{n-1}(\mathbf{F}(\mathbf{x}, \mathbf{\Theta})) \tag{2.5}$$

In general parameter estimation terminology, the DFD measures the residuals for parameters $\mathbf{\Theta}$. The aim then is to find the optimal parameters $\hat{\mathbf{\Theta}}$ minimising some function of the DFD. A least squares solution, for example, entails solving

$$\hat{\mathbf{\Theta}} = \arg\min_{\mathbf{\Theta}} \sum_{\mathbf{x}} \mathrm{DFD}^2_{\mathbf{\Theta}}(\mathbf{x}) \tag{2.6}$$

$\mathrm{DFD}(\mathbf{x})$ will have a high value if the parameters $\hat{\mathbf{\Theta}}$ do not describe the motion at site $\mathbf{x}$. For inaccurate parameters, this will be over much of the image. However, at sites $\mathbf{x}$ containing local motion, $\mathrm{DFD}(\mathbf{x})$ will take a high value for all parameter values. Thus the minimisation technique adopted should be robust against outliers.

2.2.3 Robust estimation approaches

Odobez and Bouthemy [244] applied an M-estimation approach to robust estimation for global motion parameters. In M-estimation, the aim is to solve

$$\hat{\mathbf{\Theta}} = \arg\min_{\mathbf{\Theta}} \sum_{\mathbf{x}} \rho\left(\mathrm{DFD}_{\mathbf{\Theta}}(\mathbf{x})\right) \tag{2.7}$$

where rather than squaring the DFD, a function $\rho(.)$ is used to limit the influence of large residuals on the optimal parameters. This formulation is equivalent to solving

$$\hat{\mathbf{\Theta}} = \arg\min_{\mathbf{\Theta}} \sum_{\mathbf{x}} w(\mathbf{x})\, \mathrm{DFD}^2_{\mathbf{\Theta}}(\mathbf{x}) \tag{2.8}$$

for suitable weights at each site [351].

[1] In the sense that $\mathbf{x}' = \mathbf{F}(\mathbf{x}, \mathbf{\Theta})$ is related to $\mathbf{x}$ by

$$\begin{bmatrix} x_1' \\ x_2' \end{bmatrix} = \begin{bmatrix} x_1 & x_2 & 1 & 0 & 0 & 0 & -x_1x_1' & -x_2x_1' \\ 0 & 0 & 0 & x_1 & x_2 & 1 & -x_1x_2' & -x_2x_2' \end{bmatrix} \cdot [a_1, a_2, d_1, a_3, a_4, d_2, c_1, c2]^T$$

with $\mathbf{x}'$ appearing on both sides of the equation.

The optimal parameters $\hat{\mathbf{\Theta}}$ are determined using an iterative scheme. Two factors motivate the iterative approach. Firstly, a Taylor series expansion must be used to linearise $\mathrm{DFD}(\mathbf{x})$ with respect to $\mathbf{\Theta}$ around some initial estimate $\mathbf{\Theta}^0$. Thus, for some update $\mathbf{u}$, the weighted DFD at $\mathbf{x}$, $\mathrm{WDFD}_{\mathbf{\Theta}^0+\mathbf{u}}(\mathbf{x})$, is found by

$$\mathrm{WDFD}_{\mathbf{\Theta}^0+\mathbf{u}}(\mathbf{x}) = w(\mathbf{x})\left[I_n(\mathbf{x}) - I_{n-1}(\mathbf{B}(\mathbf{x})\mathbf{\Theta}^0)\right] - \nabla I_{n-1}(\mathbf{B}(\mathbf{x})\mathbf{\Theta}^0)\mathbf{B}(\mathbf{x})\mathbf{u} + \hat{\epsilon}(\mathbf{x}) \tag{2.9}$$

where $\hat{\epsilon}(\mathbf{x})$ represents both the higher-order terms of the series and the error term. The Taylor series expansion is only valid for small updates $\mathbf{u}$, and as such is applied iteratively until some convergence criterion is met (typically a threshold on the magnitude of $\mathbf{u}$). This also suggests that a multi-resolution scheme be employed, as described in section 2.6.

The second reason that iterative estimation is used pertains to the choice of weights. The weight at a site $\mathbf{x}$ should be high if $\mathbf{x}$ is subject to global motion, and low otherwise. Thus, the ideal weights can only be determined where the global motion is already known. The weights and parameters are therefore estimated iteratively, resulting in an Iteratively Reweighted Least Squares (IRLS) estimation scheme.

At each iteration i, the value of $\mathbf{u}$ for which

$$\frac{\partial \mathrm{WDFD}(\mathbf{x})}{\partial \mathbf{u}} = 0 \tag{2.10}$$

is found. This value is given by

$$\hat{\mathbf{u}} = [\mathbf{G}^T\mathbf{W}^{i-1}\mathbf{G}]^{-1}\mathbf{G}^T\mathbf{W}^{i-1}\mathbf{z} \tag{2.11}$$

Here $\mathbf{G}$ is the matrix of gradient values $\{\nabla I_{n-1}(B(\mathbf{x})\mathbf{\Theta}^{i-1})B(\mathbf{x})\}_{\mathbf{x}}$, W^i is the vector containing the weights over the entire image, and $\mathbf{z}$ is the vector of residuals $\{I_n\mathbf{x} - I_{n-1}(B(\mathbf{x})\mathbf{\Theta}^{i-1}\}_{\mathbf{x}}$.

Given the parameters $\mathbf{\Theta}^i$ at iteration i, the weights W^i can then be found by

$$w(\mathbf{x}) = \frac{\rho(\mathrm{DFD}_{\mathbf{\Theta}^i}(\mathbf{x}))}{\mathrm{DFD}_{\mathbf{\Theta}^i}(\mathbf{x})} \tag{2.12}$$

where various choices for ρ can be used [351]. In [244], Tukey's biweight function [156] is used for ρ.

Dufaux and Konrad [93] also use M-estimation, with the function ρ taken as

$$\rho(e) = \begin{cases} e^2 & \text{if } |e| < t \\ 0 & \text{otherwise} \end{cases} \tag{2.13}$$

The threshold t is chosen at each iteration so as to exclude the top 10% of absolute DFD values. This is equivalent to using binary valued weights to exclude the influence of the sites having the largest residuals. Their work targets the eight-parameter projective transform, and therefore uses a gradient descent method to find the minimum [258].

2.2.4 Wiener solution approach

Kokaram and Delacourt presented a Wiener-based alternative for finding the parameter update [182]. In this approach,

$$\hat{\mathbf{u}} = [\mathbf{G}^T\mathbf{W}\mathbf{G} + \mu_w I]^{-1}\mathbf{G}^T\mathbf{W}\mathbf{z} \tag{2.14}$$

This introduces the regularising term $\mu_w = \left(\sigma_{ee}^2/\sigma_{uu}^2\right)$, where σ_{ee}^2 is the variance of the residuals and σ_{uu}^2 is the variance of the estimate for $\hat{\mathbf{u}}$. In practice, $\mu_w = |W\mathbf{z}|\frac{\lambda_\wedge}{\lambda_\vee}$, where $\frac{\lambda_\wedge}{\lambda_\vee}$ is the condition number of $[\mathbf{G}^T\mathbf{W}\mathbf{G} + \mu_w I]$. This regularisation limits the update where the matrix is ill-conditioned or where the DFD takes large values. Binary weights are used in [182], although other weighting functions could be used.

2.2.5 Fourier domain approaches

The normalised cross-correlation of two images can be efficiently computed in the spatial frequency domain using the Fast Fourier Transform (FFT). This suggests an efficient approach for computing translational displacement between two images. Let I_n and I_{n-1} be two frames that differ only by a displacement $\mathbf{d} = [d_x, d_y]$ such that

$$I_n(x, y) = I_{n-1}(x + d_x, y + d_y) \tag{2.15}$$

The corresponding Fourier transforms, F_n and F_{n-1}, are related by

$$F_n(u, v) = e^{j2\pi(ud_x+vd_y)}F_{n-1}(u, v) \tag{2.16}$$

The cross correlation function between the two frames is defined as

$$c_{n,n-1}(x, y) = I_n(x, y) \circledast I_{n-1}(x, y) \tag{2.17}$$

where $\circledast$ denotes 2-D convolution. Moving into the Fourier domain results in the complex-valued cross-power spectrum expression

$$C_{n,n-1}(u, v) = F_n(u, v)F_{n-1}^*(u, v) \tag{2.18}$$

where F^* is the complex conjugate of F. Normalising $C_{n,n-1}(u, v)$ by its magnitude gives the phase of the cross-power spectrum:

$$\begin{aligned}\tilde{C}_{n,n-1}(u, v) &= \frac{F_n(u, v)F_{n-1}^*(u, v)}{|F_n(u, v)F_{n-1}^*(u, v)|} \\ &= e^{-j2\pi(ud_x+vd_y)}\end{aligned} \tag{2.19}$$

Taking the inverse Fourier transform of $\tilde{C}_{n,n-1}(u, v)$ yields the phase-correlation function

$$\tilde{c}_{n,n-1}(x, y) = \delta(x - d_x, y - d_y) \tag{2.20}$$

which consists of an impulse centered on the location $[d_x, d_y]$, the required displacement.

The phase correlation method for translation motion, then, involves evaluating (2.19) and taking the inverse Fourier transform. This yields a phase correlation surface; the location of the maximum in this surface corresponds to the translational motion. Sub-pixel accuracy can be obtained with suitable interpolation.

Phase correlation for translational motion was presented by Kuglin and Hines in 1975 [186], for registration of astronomical images. In 1993, Bracewell published the affine theorem for the two-dimensional Fourier transform [44]; a number of researchers then developed Fourier-domain systems for affine image registration [209, 261] and global motion estimation [147, 188]. An accessible presentation of how Fourier domain methods can be used for affine global motion estimation appears in [167, appendix A].

2.2.6 Integral projections

The vertical and horizontal *integral projections* I_n^v, I_n^h of an image I_n are the vectors formed by summing along the rows (respectively columns) of the image:

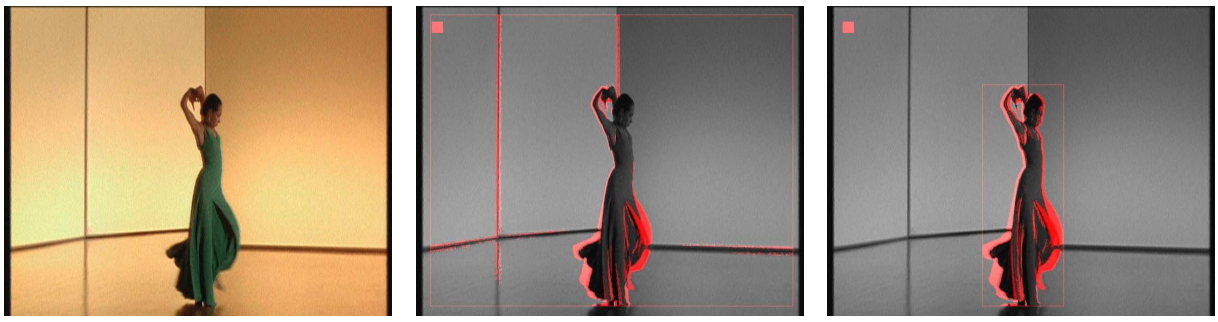
$$I_n^v = \sum_x I_n(x, y) \tag{2.21}$$

$$I_n^h = \sum_y I_n(x, y) \tag{2.22}$$

The use of integral projections was originally proposed by Lee and Park in 1987 for block matching in local motion estimation [191], described further below. In 1999, Milanfar showed that motion estimation applied to these two one dimensional vectors reveals the global motion of the image itself [222]. This work considered integral projections as a form of Radon transform, and established the theoretical soundness of the approach on this basis; the affine motion model is used. A differently motivated analysis appears in [71], targeting translation motion. The appeal of this approach lies in its simplicity of implementation and suitability for real-time applications.

2.3 Refined Global Motion Estimation for Simple Scenes

As described above, motion in an image sequence can only be directly estimated in picture areas containing edge information. This can lead to difficulties in global motion estimation for sequences with little detail in the background. This situation is common in videos depicting theatrical dance performance, for example, where a dance is performed in front of a flat backdrop. These can be considered 'simple' scenes, in the sense that there is a well-defined foreground actor and a simple, flat background. In this section, some examples of this problem are illustrated, and a new approach for improving Global Motion Estimation (GME) performance is described. The global motion estimation algorithm described by Odobez and Bouthemy in [244] is used. The



(a) Video frame (b) Result of inaccurate GME (c) Result of accurate GME

Figure 2.2: Improvement in global motion estimation wrought by excluding the foreground region. The DFD after global motion compensation is superimposed in red in (b) and (c), along with its bounding box.

global motion model here is affine, with multiplicative parameter $\mathbf{A}$ and translation parameter $\mathbf{d}$.

2.3.1 Diagnosing GME Failure

As described above, the DFD at frame n after global motion estimation is the difference between compensated frame $n-1$ and frame n:

$$\mathrm{DFD}(\mathbf{x})_n = I_n(\mathbf{x}) - I_{n-1}(\mathbf{A}\mathbf{x} + \mathbf{d}) \tag{2.23}$$

Where GME has been successful, all the energy in the DFD will correspond to foreground regions of the image sequence. Figure 2.2 shows DFDs for successful and unsuccessful motion estimation. Figure 2.2 (a) shows a frame from the *greenDancer* sequence. There is no camera motion at this frame of the sequence. Image (b) shows the DFD energy and bounding box in red, after inaccurate global motion estimation. The DFD bounding box is the smallest rectangle containing all non-zero elements of the DFD image. Image (c) shows the DFD energy and bounding box after successful global motion estimation, using the technique described here. Where GME is accurate, the DFD bounding box corresponds to the foreground region of the frame.

Although the region of the frame corresponding to the foreground cannot be known in advance, it can be assumed that GME is accurate more often than not, and so the DFD bounding box usually contains the foreground motion. This assumption is generally justified, particularly at the start of a sequence as the subject begins to move. Any sudden change in the DFD bounding box can then be interpreted as an indication of GME failure.

The bounding box at frame n is defined by four components, for its top, bottom, left, and right locations: $BB_n = [BBt_n, BBb_n, BBl_n, BBr_n]$. Given the bounding boxes of the previous k frames, a Gaussian distribution can be proposed to describe the expected location of the

bounding box in the current frame. Hence $BBt_n \sim \mathcal{N}(\overline{BBt}, \sigma_t^2)$ and similarly for the other bounding box components. These components can be estimated using the k measurements from the previous frames by

$$\overline{BBt} = \frac{1}{k} \sum_{j=n-k}^{n-1} BBt_j \tag{2.24}$$

$$\sigma_t^2 = \frac{1}{k} \sum_{j=n-k}^{n-1} (BBt_j - \overline{BBt})^2 \tag{2.25}$$

GME failure is then diagnosed if any component k lies outside the 99% confidence interval for the associated Gaussian, i.e. where

$$\left| BB_k - \overline{BB_k} \right| > 2.576\sigma_k \tag{2.26}$$

2.3.2 Recovering from GME failure

Once GME failure has been diagnosed at a particular frame, various methods to improve the estimate can be employed. The simplest approach is to exploit the fact that camera motion is temporally smooth in most cases, by proposing the GME parameters from the previous frame as candidate parameters for the frame causing difficulty. These are designated θ^2. A second candidate set of parameters can be generated by performing GME a second time, but excluding the region inside the DFD bounding box of the previous frame from the estimation. This prevents the foreground motion from affecting the global motion estimate. The resulting parameters are designated θ^3. A third, related candidate is generated by finding the median of the last $N = 25$ DFD bounding boxes, designated the median foreground rectangle. GME is then performed with this region excluded from estimation, and the parameters found are designated θ^4.

In cases where GME failure has been diagnosed, there are thus four candidate sets of parameters–the original parameters and the three alternatives described above. These are designated θ^i, with $i \in \{1 \dots 4\}$. Each of these parameter sets is used to generate a DFD, and the bounding box of each DFD image is found. The set of GME parameters resulting in the DFD bounding box having the closest match to the historic distribution is selected as the best estimate for the current frame. Specifically, the selected parameters for frame n, $\theta_n = \{\mathbf{A}, \mathbf{d}\}$, are found by

$$\theta_n = \theta^{\hat{\imath}}, \text{ where } \hat{\imath} = \underset{i \in \{1 \dots 4\}}{\arg\max} \sum_{k \in \{t,b,l,r\}} \exp\left(\frac{(BBk_n^i - \overline{BBk})^2}{\sigma_k} \right) \tag{2.27}$$

Here BBk_n^i is bounding box component k resulting from candidate parameters θ^i.

The procedure is illustrated in flowchart form in figure 2.3.

2.3.3 Results and Assessment

Ground-truth global motion data is not available for the sequences analysed here, and therefore a complete quantitative assessment of the improvements introduced by this technique is not

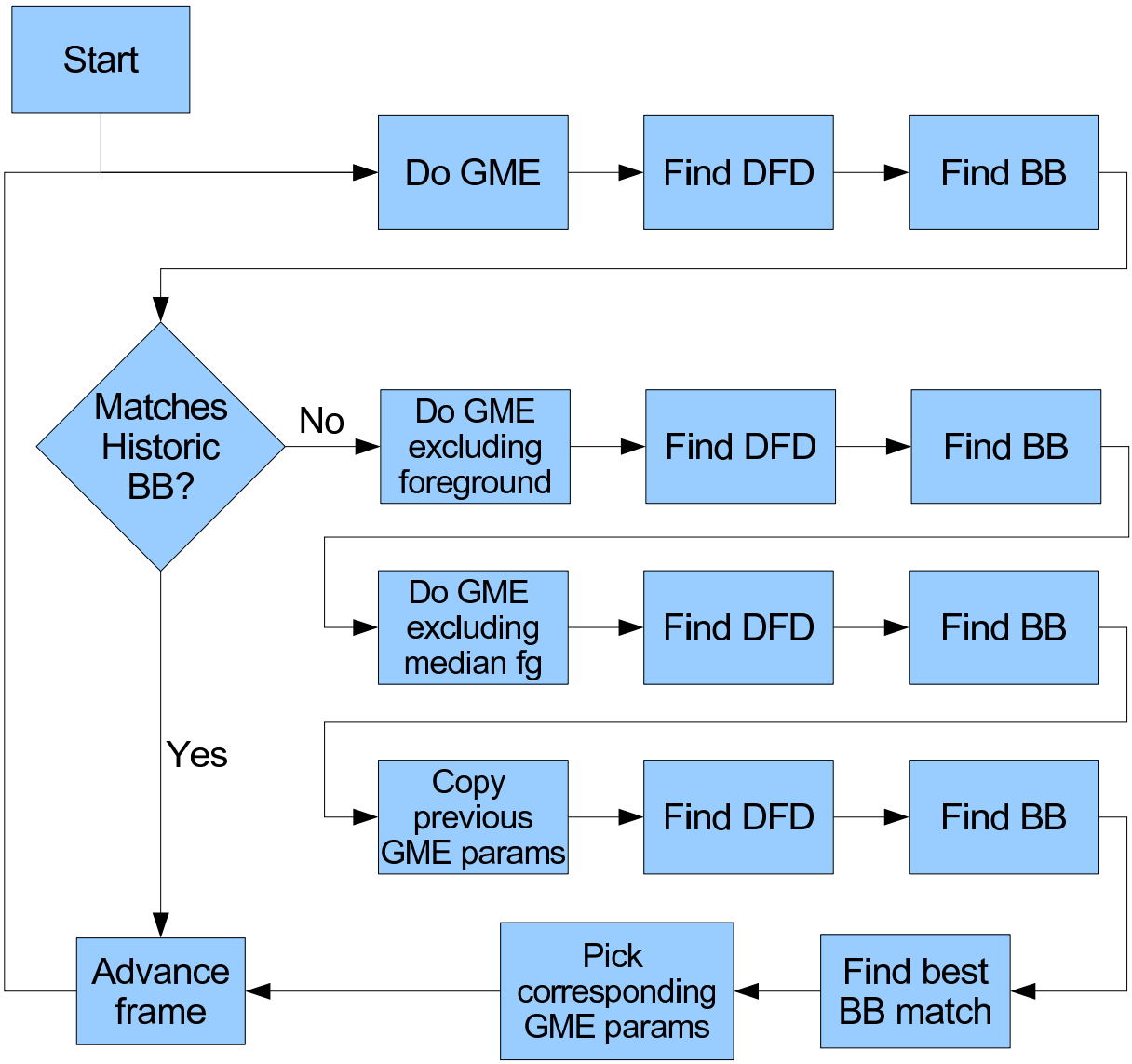
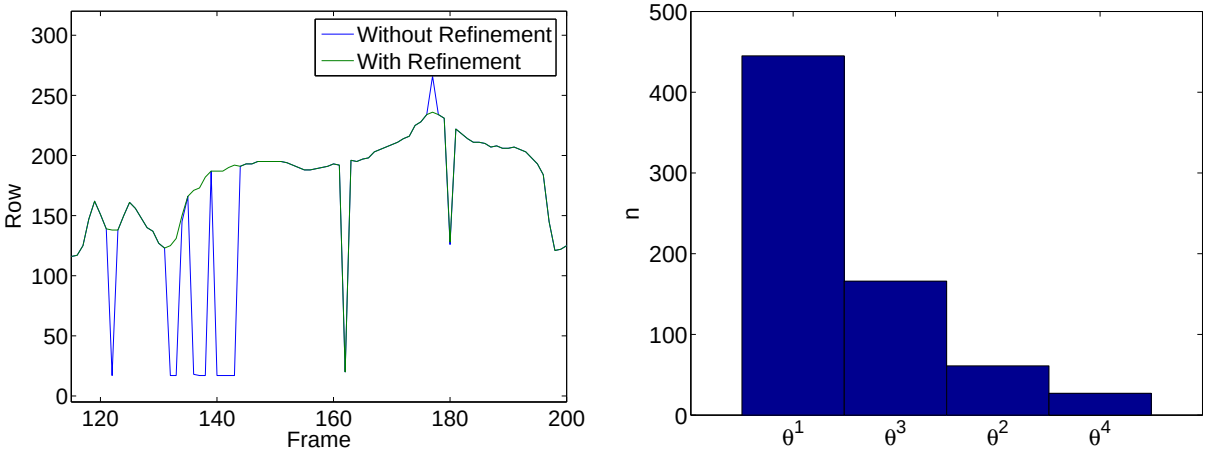


Figure 2.3: The GME refinement algorithm. 'BB' stands for the DFD bounding box. The 'BB match' refers to the similarity between the current DFD bounding box and those found in previous frames.

possible. Figure 2.4 (a) shows the location of the bottom edge of the DFD bounding box over 80 frames of the *greenDancer* sequence, identified by unmodified GME and also using the technique described here. The refinement technique results in a considerably smoother signal, indicating that GME performance has been improved.

Figure 2.4 (b) shows how often each set of global motion parameters was selected when the refinement technique was applied to the 1900 frame *greenDancer* sequence. The initial parameters, θ^1, were found to be a poor match to the historic distribution in 699 cases. In 445 cases, the θ^1 parameters resulted in the best fit of all candidate parameters. This suggests



(a) Location of DFD bounding box bottom edge using unrefined GME and the technique described here. The plot covers 80 frames from the *greenDancer* sequence.

(b) Selection counts of different candidate parameters. N=699.

Figure 2.4

that the refinement is being invoked somewhat more than necessary, and that a looser fit to the historic distribution should be considered a match. Of the three alternative candidate parameters, θ^3 most often results in the closest-matching bounding box, but each alternative is selected for some frames.

Video material illustrating the application of this approach to a number of sequences is provided in the accompanying DVD. These videos demonstrate that the technique results in a considerable improvement in GME accuracy for numerous frames. It is noted that this technique should not be incorporated into GME systems by default, but rather selected by an operator for application to suitable sequences.

2.4 Local Motion Estimation

The problem of local motion estimation in image sequences has been the target of considerable research effort over a number of decades. A brief overview of the principles and principal techniques is presented here. Comprehensive review material is available in a number of publications [94, 214, 230, 288, 298, 314].

2.4.1 Problem statement

In local motion estimation, the aim is to describe the motion of each region of the image individually. The equation for the image motion remains $I_n(\mathbf{x}) = I_{n-1}(\mathbf{F}(\mathbf{x}, \mathbf{\Theta}))$, but the motion

model $\mathbf{F}(\mathbf{x}, \mathbf{\Theta})$ varies with the site $\mathbf{x}$. The most commonly used formulation is

$$\mathbf{F}(\mathbf{x}, \mathbf{\Theta}) = \mathbf{x} + \mathbf{d}_{n,n-1}(\mathbf{x}) \tag{2.28}$$

Here $\mathbf{d}_{n,n-1}(\mathbf{x})$ is a vector field describing the translational motion at each pixel. The effects of zooming and rotation can be approximated as translational motion provided the motion between frames is small.

Numerous approaches to fitting a model of the form (2.28) to an image sequence have been developed, and will be outlined here. It is first noted that (2.28) fails to account for the effects of occlusion and revealing in the image sequence. These effects are found in all sequences containing motion: as any object moves, it will occlude some areas of the previous frame and reveal others. Modelling these effects requires extensions to the motion model that are not described here; approaches addressing this issue have been presented in [31, 160, 179, 184, 286].

Consider motion estimation in video having dimensions $M \times N$ pixels. There are then MN equations of the form

$$I_n(\mathbf{x}) = I_{n-1}(\mathbf{x} + \mathbf{d}_{n,n-1}(\mathbf{x})) \tag{2.29}$$

and each equation is to be solved for the two components of $\mathbf{d}(\mathbf{x})$. There are thus twice as many unknowns as there are equations, and so the problem is underconstrained. In fact, at each site $\mathbf{x}$, only the motion prependicular to the gradient at $\mathbf{x}$ can be estimated, known as the 'normal flow' [298, pp. 81-92]. This difficulty is known as the aperture effect, because in effect the aperture, or window, used for motion estimation is too small.

The most common way of dealing with this difficulty is to assume that the motion is constant over some window larger than one pixel, typically a block B pixels on a side. Motion estimation within the block is then described by B^2 equations with two unknowns in total, and the problem is no longer underconstrained. Motion estimation over blocks of pixels also improves resilience to noise in the image sequence. The aperture effect can still arise where the block size is small relative to object size. Using a larger block size reduces the effect of this problem, but for large block sizes the assumption that the motion within the block is constant is less likely to hold.

2.4.2 Correspondence matching

Block matching is a robust, readily implemented technique for motion estimation. For each $B \times B$ block b in frame n, the most similar $B \times B$ block in frame $n-1$ is found. The motion vector assigned to the block in frame n is then the displacement to the matched block.

Block similarity is measured by examination of the DFD over the block. The most commonly used measures are the Mean Squared Error (MSE) and Mean Absolute Difference (MAD):

$$\mathrm{MSE}(b, \mathbf{d}) = \frac{1}{B^2} \sum_{\mathbf{x} \in \mathcal{B}(b)} (\mathrm{DFD}(\mathbf{x}, \mathbf{d}))^2 \tag{2.30}$$

$$\mathrm{MAD}(b, \mathbf{d}) = \frac{1}{B^2} \sum_{\mathbf{x} \in \mathcal{B}(b)} |\mathrm{DFD}(\mathbf{x}, \mathbf{d})| \tag{2.31}$$

where $\mathcal{B}$ is the set of sites $\mathbf{x}$ in block b, and $\text{DFD}(\mathbf{x}, \mathbf{d}) = I_n(\mathbf{x}) - I_{n-1}(\mathbf{x} + \mathbf{d})$. The vector $\mathbf{d}$ resulting in the lowest value of the error measure is assigned to block b. For translational motion under constant illumination in a sequence uncorrupted by noise, the correct vector will result in a zero block error.

A block matching scheme requires that two parameters be specified: the search width, and the search resolution. The search width limits the maximum displacement magnitude that can be estimated, while the search resolution limits the maximum obtainable accuracy. Increasing the search width or search resolution results in a considerable increase in computational cost. For a search width $\pm w$ pixels in both the horizontal and vertical directions, with a resolution of $\frac{1}{r}$ pixels, block matching requires $B^2(2rw + 1)^2$ operations per block. Fractional accuracy imposes an additional cost for interpolation of the target frame (frame $n - 1$).

Evaluating every candidate vector in the search area is known as Full Motion Search. A number of alternative schemes aiming to reduce the amount of computation required for a given search width and resolution have been presented. These include Three Step Search [177], Cross Search [119], and the Successive Elimination Algorithm [199], amongst others [46, 346]. Although these approaches do offer considerable computational savings, they are not as reliable as full motion search block matching.

Integral projections were described for global motion estimation above, but were originally proposed for local motion estimation, by Lee and Park [191]. This approach reduces the computational cost of block matching by making comparing blocks cheaper. Rather than compare all B^2 pixels across two blocks, the integral projections of the block are compared. Matching block integral projections is shown to provide similar results to full block matching, at a cost of only $2B$ operations per block comparison. This approach has been further developed by various researchers [172, 192].

2.4.3 Gradient-based methods

Block matching is effectively an exhaustive search approach to find the vector $\mathbf{d}$ minimising the DFD at $\mathbf{x}$. An alternative approach is to linearise the image model about $\mathbf{d}$ using a Taylor series expansion, such that

$$I_n(\mathbf{x}) = I_{n-1}(\mathbf{x}) + \mathbf{d}^T_{n,n-1}(\mathbf{x})\nabla I_{n-1}(\mathbf{x}) + e_n(\mathbf{x}) \tag{2.32}$$

where ∇ is the two-dimensional gradient operator and $e_n(\mathbf{x})$ accounts for both the higher order terms in the Taylor series and any model error. Rearranging this equation yields

$$z(\mathbf{x}) = \mathbf{d}^T_{n,n-1}(\mathbf{x})\nabla I_{n-1}(\mathbf{x}) + e_n(\mathbf{x}) \tag{2.33}$$

where $z(\mathbf{x}) = I_n(\mathbf{x}) - I_{n-1}(\mathbf{x})$. This approach was described in 1976 by Caffario and Rocca [50].

The Taylor series approximation for linearisation about $\mathbf{d}$ is only valid over small displacements. To allow for this limitation, estimation of $\mathbf{d}$ is performed iteratively, with an update $\mathbf{u}$

applied at each step. This scheme can be applied to single pixels, where the inital estimate for the displacement at each site is taken from the previous pixel in raster-scan order. Hence these schemes are described as *pel-recursive* approaches. When estimation is performed at pixel level, the aperture effect means that the update at each iteration will be always be perpendicular to the image gradient. The use of the pel-recursive scheme, however, means that the scheme can converge to the two-dimensional motion after a number of pixels have been processed.

The earliest of these pel-recursive schemes was presented by Netravali and Robbins [236]. Here the update for the displacement at $\mathbf{x}$ is found using a steepest-descent approach, yielding

$$\mathbf{d}^{i+1}(\mathbf{x}) = \mathbf{d}^{i}(\mathbf{x}) - \epsilon \,\mathrm{DFD}(\mathbf{x}, \mathbf{d}^{i}) \nabla I_{n-1}(\mathbf{x} - \mathbf{d}^{i}) \tag{2.34}$$

where the step size ϵ must be chosen. Walker and Rao then proposed an adaptive step size given by

$$\epsilon = \frac{1}{2||\nabla I_{n-1}(\mathbf{x} - \mathbf{d}^{i})||^2} \tag{2.35}$$

This effectively reduces the step size in the neighborhood of large gradients, improving accuracy, and increases the step size where the gradient is small, improving the speed of convergence.

Gradient-based motion estimation can also be applied to blocks of pixels, reducing the effect of the aperture problem and imposing a stronger regularity constraint. The approach described by Biemond *et al.*using Wiener estimation is the first example of this approach [27], being essentially an extension of the Netravali-Robbins method applied to block motion in which the error term $e_n(\mathbf{x})$ is not discarded [298]. Instead, the error is assumed to be effectively Gaussian. This facilitates Linear Minimum Mean Least Square Error (LMMSE) estimation of the update, such that

$$\mathbf{d}^{i+1}(\mathbf{x}) = \mathbf{d}^{i}(\mathbf{x}) + \left[\mathbf{G}^T\mathbf{G} + \mu\mathbf{I}\right]^{-1}\mathbf{G}^T\mathbf{z} \tag{2.36}$$

where $\mathbf{G}$ and $\mathbf{z}$ collect the gradient and image difference values for each pixel inside the block. μ is a damping parameter given by $\frac{\sigma_{ee}^2}{\sigma_{uu}^2}$; σ_{ee}^2 is the variance of the error values and σ_{uu}^2 is the variance of the estimate for the update $\mathbf{u} = \mathbf{d}^{i+1}(\mathbf{x}) - \mathbf{d}^{i}(\mathbf{x})$.

The damping parameter μ is designed to improve stability in the computation of $\left[\mathbf{G}^T\mathbf{G}\right]^{-1}$. Errors in the motion estimation can nevertheless result when this matrix is very ill-conditioned. This ill-conditioning can be measured by examining the ratio of the eigenvalues of the matrix, and tailoring the update accordingly. This refinement has been explored by various researchers [40, 95]. Kokaram in [178] described a system in which

$$\begin{aligned}
\mathbf{u}^i &= \begin{cases} \bar{\alpha}\mathbf{e}_x & \text{if } \frac{\lambda_x}{\lambda_n} > \tau_\alpha \\ \left[\mathbf{G}^T\mathbf{G} + \mu\mathbf{I}\right]^{-1}\mathbf{G}^T\mathbf{z} & \text{otherwise} \end{cases} \\
\mu &= |\mathbf{z}|\frac{\lambda_x}{\lambda_y} \\
\bar{\alpha} &= \frac{\mathbf{e}_x^T}{\lambda_x}\mathbf{G}^T\mathbf{z}
\end{aligned} \tag{2.37}$$

Here λ_x and λ_y are the largest and smallest eigenvalues of $\mathbf{G}^T\mathbf{G}$, and $\mathbf{e}_x$ is the eigenvector of $\mathbf{G}^T\mathbf{G}$ corresponding to λ_x. α is a threshold on the conditioning of $\mathbf{G}^T\mathbf{G}$. The idea here is that where $\mathbf{G}^T\mathbf{G}$ is very ill-conditioned, the Martinez [214] solution is used; otherwise, the damping parameter μ is found as described by Driessen *et al.* [90].

2.4.4 Transform domain methods

The use of the Fourier transform for global motion estimation was described above. This approach can also be applied to individual blocks in an image for local motion estimation. Jain and Jain used this method in 1978 for analysis of images from radar [161]; it has also been used in frame rate conversion for television [301] and in motion estimation for restoration [276]. The complex wavelet transform (CWT) has also been used for motion estimation [212].

2.4.5 Bayesian methods

The approaches described above can be considered maximum likelihood methods, in that they are entirely data driven. As such, they do not explicitly take into account prior intuitions concerning the nature of motion fields. In particular, motion fields are generally piecewise smooth, and this smoothness can be incorporated as a constraint to condition the estimate. This incorporation of prior knowledge, combined with data-driven evaluation of the likelihood, results in a Maximum *a posteriori* (MAP) method. This is distinct from the implicit smoothness introduced by pel-recursive schemes or motion estimation over blocks.

The *a posteriori* probability of a motion field $\mathbf{D}$ is given by Bayes' theorem

$$p(\mathbf{D}|I_{n-1}, I_n) = \frac{p(I_n|I_{n-1}, \mathbf{D})p(\mathbf{D})}{p(I_n|I_{n-1})} \tag{2.38}$$

Here $p(I_n|I_{n-1}, \mathbf{D})$ is the likelihood of the vector field $\mathbf{D}$, and $p(\mathbf{D})$ is the prior probability of $\mathbf{D}$ (typically some measure of smoothness). The denominator of this expression describes the marginal probability of I_n, and does not vary with $\mathbf{D}$. Thus the MAP estimate of $\mathbf{D}$ is given by

$$\hat{\mathbf{D}} = \arg\max_{\mathbf{D}} p(I_n|I_{n-1}, \mathbf{D})p(\mathbf{D}) \tag{2.39}$$

$\mathbf{D}$ here represents the entire flow field, and it is not generally feasible to maximise 2.39 over a space of such large dimensionality. Therefore Bayesian schemes generally address flow field smoothness at each site individually.

Maximising the *a posteriori* probability is equivalent to minimising the negative logarithm of the probability, a quantity described as the energy of the flow field. The energies due to the likelihood and prior may considered separately:

$$\hat{\mathbf{D}} = \arg\min_{\mathbf{D}} L(\mathbf{D}) + V(\mathbf{D}) \tag{2.40}$$

The likelihood energy $L(\mathbf{D})$ is typically some DFD related measure such as the MSE. Thus incorporating prior information is equivalent to adding some smoothness term to the error

associated with a given vector. Conversely, any scheme in which a smoothness term is added to the vector error can be described within the Bayesian framework. Kelly has described how a number of estimation schemes, including the optical flow method of Horn and Schunck [148], and the 3D recursive search block matching algorithm of de Haan *et al.* [76–78], can be unified in the Bayesian framework in this manner [167].

More explicitly Bayesian methods incorporate iterative smoothing into the estimation of the flow field. Konrad and Dubois employ stochastic relaxation techniques such as the Metropolis algorithm and the Gibbs sampler [184]. Kelly has described motion smoothing using deterministic methods such as Iterated Conditional Modes (ICM) and Belief Propagation [167]. Numerous other variations on MAP estimation have been presented [2, 350].

The presence of moving objects in a scene introduces discontinuities in the optic flow field. If the smoothness constraint is applied globally, then, the motion vectors at object boundaries become blurred. Numerous researchers have investigated approaches to this problem—encouraging *piecewise* smoothness as opposed to global smoothness [31,140,160]. Using the *oriented smoothness constraint*, introduced by Nagel [232, 233], the optic flow field is smoothed perpendicular to the image brightness gradient. Konrad and Dubois [184] and Heitz and Bouthemy [141] introduced *line fields* for stochastic motion smoothing. A line field is a lattice in between pixel sites controlling how much smoothness should be applied across adjacent pixels. These stochastic techniques have a high computational cost. A number of researchers have recognised that dealing with these motion discontinuities effectively requires some level of video segmentation—which is itself often predicated on motion estimation. Thus motion estimation and video segmentation are complementary problems [57, 58, 140, 287].

2.5 Video Segmentation

Video segmentation is the problem of discovering a set of labels corresponding to each object in an video, and assigning one of these labels to each pixel in each frame describing which object that pixel depicts. A brief outline of some of the main methods is presented here; the review article by Zhang and Lu [348] is recommended for further information.

2.5.1 Motion segmentation

Early approaches to the problem of video segmentation relied exclusively on motion features, and as such were described as motion segmentation. The use of motion for segmentation is justified by various findings in psychology. For example, the Gestalt psychologists realised that common motion, or 'common fate', is one of the strongest cues for grouping in visual perception [323], and more recently cognitive psychologists have shown that infants treat any surface that is cohesive, bounded, and moves as a unit as a single object—ignoring shape, color, and texture [16,281,282].

2.5.1.1 Dominant motion methods

A number of early motion segmentation techniques adopted a *dominant motion* approach [19, 157]. This is a recursive scheme relying on the assumption that where multiple motions are present, one is the 'dominant' motion, in the sense that global estimation of motion over the entire frame will result in a good estimate of this dominant motion. Those regions in the image that obey this dominant motion can then be excluded from consideration, and estimation applied to the remaining areas to find the next dominant motion. The procedure is recursively applied until all of the frame is accounted for. These schemes are unreliable in processing video containing a number of comparably dominant motions.

2.5.1.2 Expectation-Maximization methods

Dominant motion analysis is a global, or top-down, approach to motion segmentation, and has been largely superseded by a local, bottom-up approach in which small areas undergoing similar motion are merged. This approach is exemplified by the seminal work by Wang and Adelson [320, 321], which also introduced the *layer model* underpinning much of the work in motion segmentation. Here, rather than considering a single, piecewise smooth flow field, the video sequence is considered as made up of occluding layers. Discontinuities in the motion field arise as the result of occlusion between layers. Thus explicit piecewise smoothness need not be imposed through regularisation [118,255] or robust estimation [30,73,82]; rather globally smooth motion is estimated for each layer separately.

In [320], the motion of each layer is described using an affine model. In outline, the scheme is as follows: First, optic flow estimation is carried out. The first frame is divided into blocks to provide an initial set of regions. The frame is then segmented using a number of iterations of the following steps:

1. An affine model is fitted to the motion vectors within each region, using linear least squares.
2. Adaptive k-means clustering is applied to merge similar motion models.
3. The regions are updated by assigning each pixel to the motion model best describing the motion vector at that pixel.
4. Regions containing disjoint areas are split such that all regions are contiguous.
5. Small regions are discarded.

This is essentially an Expectation Maximisation (EM) scheme altering between estimation of model parameters and maximisation of the likelihood given these models. Typically fewer than 20 iterations are required to segment the first frame. The segmentation of subsequent frames is initialised using the final result of the previous frame, which encourages temporal coherence. Once the layer assignments have been made for a number of frames in the sequence, resynthesis

of frames in the sequence from the layer models can be used to determine the depth relationship of the layers.

A number of variants on this EM approach have been presented, including methods combining segmentation with motion estimation [14, 57, 162]. Elias and Kingsbury in [97] presented some refinements exploiting forward and backward optic flow for more accurate modelling of uncovering and occlusion. Weiss presented a variant in which layer motion is modelled as a smooth dense flow field rather than using the affine model [333].

In [41], Borshukov *et al.*point out that the adaptive k-means clustering used by Wang and Adelson in [320] results in layers being assigned a mean affine motion model, averaged over regions in the layer. They suggest replacing this clustering step with a merge, such that the model chosen for each layer is the candidate model having the best fit. The implementation of this approach is in a sense a combination of the dominant motion approach described by Bergen [19] and the Wang-Adelson algorithm.

2.5.1.3 Bayesian approaches

A number of Bayesian approaches to motion segmentation have been described, from the early work of Murray and Buxton in 1987 [226] to the more recent papers by Torr *et al.* [304], Vasconcelos and Lippman [243], and Kumar *et al.* [187]. An accessible, representative example of these approaches is the 1997 work by Chang *et al.*describing a Bayesian framework for simultaneous motion estimation segmentation and segmentation [58]. Their treatment is exemplary and comprehensive, and is outlined here.

The aim is to find optimal flow field $\hat{\mathbf{D}}$ and segmentation map $\hat{\mathbf{M}}$ given frames I_n and I_{n-1}:

$$\begin{aligned}\left(\hat{\mathbf{D}}, \hat{\mathbf{M}}\right) &= \arg\max_{\mathbf{D},\mathbf{M}} \left[p(\mathbf{D}, \mathbf{M}|I_n, I_{n-1})\right] \\ &= \arg\max_{\mathbf{D},\mathbf{M}} \left[\frac{p(I_n|\mathbf{D}, \mathbf{M}, I_{n-1})p(\mathbf{D}|\mathbf{M}, I_{n-1})p(\mathbf{M}|I_{n-1})}{p(I_n|I_{n-1})}\right]\end{aligned} \tag{2.41}$$

The denominator here is constant with respect to the unknowns and need not be considered.

The first term in the numerator, $p(I_n|\mathbf{D}, \mathbf{M}, I_{n-1})$, describes how well $\mathbf{D}$ and $\mathbf{M}$ fit the given frames. This is modelled by a Gibbs distribution with potential function $U_1(I_n|\mathbf{D}, \mathbf{M}, I_{n-1})$ based on the DFD:

$$U_1(I_n|\mathbf{D}, \mathbf{M}, I_{n-1}) = \sum_{\mathbf{x}} |\,\mathrm{DFD}(\mathbf{x}, \mathbf{D}(\mathbf{x}))|^2 \tag{2.42}$$

$p(\mathbf{D}|\mathbf{M}, I_{n-1})$ describes how well the label assignments $\mathbf{M}$ accord with the motion field $\mathbf{D}$; the dependence on I_{n-1} is ignored. The motion of the object with label m is modelled as a parametric mapping $\mathbf{\Theta}_m$ (a six parameter affine model is used in [58]). The flow field is modelled by

$$\mathbf{D}(\mathbf{x}) = \mathbf{\Theta}_{\mathbf{M}(\mathbf{x})}(\mathbf{x}) + \mathbf{D}_r(\mathbf{x}) \tag{2.43}$$

where $\mathbf{\Theta}_{\mathbf{M}(\mathbf{x})}(\mathbf{x})$ is the vector assigned to site $\mathbf{x}$ by the model for label $\mathbf{M}(\mathbf{x})$ and $\mathbf{D}_r(\mathbf{x})$ is the residual vector at $\mathbf{x}$. A least-squares estimate for the motion models $\mathbf{\Theta}_m$ can be found given

estimates for $\mathbf{D}$ and $\mathbf{M}$. The conditional pdf on the flow field is then modelled by a Gibbs distribution with potential function $U_2(\mathbf{D}|\mathbf{M})$, where

$$U_2(\mathbf{D}|\mathbf{M}) = \alpha \sum_{\mathbf{x}} ||\mathbf{D}(\mathbf{x}) - \mathbf{\Theta}_{\mathbf{M}(\mathbf{x})}(\mathbf{x})||^2 + \beta \sum_{\mathbf{x}} \sum_{\mathbf{x}' \in \mathcal{N}(\mathbf{x})} ||\mathbf{D}(\mathbf{x}) - \mathbf{D}(\mathbf{x}')||^2 \delta(\mathbf{M}(\mathbf{x}) - \mathbf{M}(\mathbf{x}')) \tag{2.44}$$

The first term here encourages a minimum norm estimate of the residual vectors $\mathbf{D}_r$, and the second is a piecewise smoothness constraint activated for adjacent sites sharing the same label. $\mathcal{N}(\mathbf{x})$ is the neighborhood of $\mathbf{x}$, typically taken to be represent four- or eight-way connectivity. α and β are scalars governing the relative influence of these terms.

The third term of the numerator in (2.42) represents the *a priori* probability of the label field $\mathbf{M}$, and is designed to encourage smoothness—again, the dependence on I_{n-1} is ignored. A Gibbs distribution is again used, with potential function $U_3(\mathbf{M})$ given by

$$U_3(\mathbf{M}) = \gamma \sum_{\mathbf{x}} \sum_{\mathbf{x}' \in \mathcal{N}(\mathbf{x})} V_2(\mathbf{M}(\mathbf{x}), \mathbf{M}(\mathbf{x}')) \tag{2.45}$$

where γ governs the influence of this term and

$$V_2(\mathbf{x}, \mathbf{x}') = \begin{cases} -1 & \text{if } \mathbf{x} = \mathbf{x}' \\ +1 & \text{otherwise} \end{cases} \tag{2.46}$$

represents the potentials for two-pixel cliques.

Overall, then, the problem is to find $\mathbf{D}, \mathbf{M}$ minimising the energy $U_1(\cdot) + U_2(\cdot) + U_3(\cdot)$, i.e.

$$\left(\hat{\mathbf{D}}, \hat{\mathbf{M}}\right) = \arg \min_{\mathbf{D},\mathbf{M}} \sum_{\mathbf{x}} \begin{bmatrix} |\,\mathrm{DFD}(\mathbf{x}, \mathbf{D}(\mathbf{x}))|^2 \\ +\alpha \sum_{\mathbf{x}} ||\mathbf{D}(\mathbf{x}) - \mathbf{\Theta}_{\mathbf{M}(\mathbf{x})}(\mathbf{x})||^2 \\ +\beta \sum_{\mathbf{x}} \sum_{\mathbf{x}' \in \mathcal{N}(\mathbf{x})} ||\mathbf{D}(\mathbf{x}) - \mathbf{D}(\mathbf{x}')||^2 \delta(\mathbf{M}(\mathbf{x}) - \mathbf{M}(\mathbf{x}')) \\ +\gamma \sum_{\mathbf{x}} \sum_{\mathbf{x}' \in \mathcal{N}(\mathbf{x})} V_2(\mathbf{M}(\mathbf{x}), \mathbf{M}(\mathbf{x}')) \end{bmatrix} \tag{2.47}$$

Solving this is achieved by iteration over estimation of the motion field, estimation of the segmentation map, and calculation of the model parameters. Initialisation is provided using optic-flow based motion estimation [298] and an initial segmentation based on the method described by Wang and Adelson [320]. This also determines the number of models used. Vasconcelos and Lippman [243, 311] describe an 'empirical Bayesian' approach in which the weights assigned to the smoothness priors (α, β, γ) are determined from the data.

This scheme illustrates the essentials of Bayesian motion estimation and segmentation, and the authors describe how (2.47) is a general framework within which many other motion estimation and motion segmentation schemes can be described. For example, with $\alpha = \gamma = 0$ and disregarding the segmentation map (i.e. setting $\mathbf{M}(\mathbf{x})$ to 0 at all sites $\mathbf{x}$) describes Bayesian motion estimation with global smoothness [298]. The Bayesian scene segmentation approach of

Murray and Stiller [226] is equivalent to setting $\beta = 0$ and disregarding the DFD term. The Wang-Adelson approach relies on the α term for motion segmentation, with a threshold on a DFD term (using the motion model $\mathbf{\Theta}_m$ rather than the optic flow vector [321, equation 9]). The methods of Stiller [287] and Iu [160] are also shown to fit into this framework.

2.5.2 Segmentation using motion and colour

The schemes described above approach motion segmentation only. Colour information can also be used, and is readily incorporated in the Bayesian framework described above [187, 304]. Some researchers have used colour segmentation as a preprocessing stage which then informs subsequent motion segmentation [20, 299, 334], but it is more common to combine motion and colour features. Some of the earliest work in video segmentation, presented in 1980, takes this approach [302]. Black [29] describes a Bayesian approach with priors on intensity, edge, and motion smoothness. Khan and Shah [168] use colour and motion features, and also employ spatial location as a feature (modelled non-parametrically) to encourage smoothness. In this work, the weight affecting the influence of the motion feature relative to the colour feature varies adaptively with motion estimation confidence.

2.5.3 Video volume analysis

The approaches described above are on-line, or frame-recursive, in that each frame is segmented in sequence. A second class of segmentation algorithms are off-line and process the video as a volume, or three-dimensional stack of images. The work of Shi and Malik in 1998 [273] falls into this category. Here the video segmentation is considered as a graph partitioning problem, where each pixel is a node in the graph and the edges of the graph connect pixels in a spatiotemporal neighbourhood. The weight applied to each edge corresponds to the similarity of feature vectors in the connected pixels. In this work only the motion at each pixel is used as a feature. The normalized cut algorithm is applied recursively to find the most salient partitions of the video volume. This paper is a part of a body of work exploring graph cuts for image and video segmentation. The algorithms concerned have been known for some decades [309, 347], but the approach is attracting increasing attention in recent years as it becomes computationally feasible to to apply these methods to image [163, 327] and video [49] data.

Mean shift analysis is another example of a clustering algorithm of long standing [114] recently applied to image and video segmentation. The technique is based on a local, kernel-based estimate of the density gradient around each sample. For each sample a track can be found from the sample to a local density maximum in the feature space. All the samples drawn to the same local maximum are classified as belonging to the same cluster. The essential parameter of the method is the bandwidth of the kernel used for density estimation.

The work of Comaniciu *et al.*brought this technique to prominence [67, 87], demonstrating applications to tracking [86, 88] and image segmentation [84, 85]. A mean-shift method for video

volume segmentation was described by DeMenthon [79] in which multiple passes using increasing kernel bandwidths are used; this improves computational efficiency, and also has sympathies with the object-grouping behaviour of human vision [197]. Wang *et al.*describe video segmentation using mean shift with an anisotropic kernel [318]. The kernel shape adapts to the covariance matrix of samples in each neighbourhood. This approach has generated convincing results in the generation of cartoon-like images from video [319].

2.5.4 Other approaches

The methods described above encompass the main techniques in use for two-dimensional video segmentation. Some authors have proposed three-dimensional segmentation in which the video is decomposed into layers and the depth of each pixel in each layer is also estimated; the paper by Steinbach *et al.* [285] is representative. It is also noted that the advent of the MPEG-4 video coding standard [225], with its emphasis on video object planes (VOPs, effectively video layers in the Wang-Adelson sense), led to considerable research into video segmentation and video sprite extraction [61, 218, 297]. Most of this work can be described in terms of the Bayesian framework outlined above.

2.6 Multiresolution Schemes

It is briefly noted here that most image processing tasks can be applied using a multi-resolution approach, in which the image is decomposed into a pyramid using successive low-pass filtering and subsampling. Analysis of the smallest image in the pyramid (at the lowest resolution) is computationally cheap and can be used to provide a good initial estimate for the image at the next resolution level.

The first such representation was the Laplacian image pyramid described by Burt and Adelson [47], used for image coding. Image pyramid schemes are used in most global and local motion estimation estimation algorithms, with early use described in block matching [28], phase correlation [99], and pel-recursive [98] techniques. A number of multi-resolution image segmentation approaches have also been described [251, 322].

3

Video Shot Change Detection: A Review

One of the fundamental types of discontinuity in video is the shot change, connecting footage from two different recordings. Detection of these shot change events is a key first step in any video retrieval or analysis system of significant scale, and has been an active area of work for over fifteen years. In this chapter, an problem overview and review of the literature is presented. Because of the vast quantity of work done in this area, a fully comprehensive review is outside the scope of this work. For further background, the reader is directed to the review papers by Costaces [69], Koprinska and Carrato [185], Lienhart [202], Boreczky [37, 38], and Ahanger and Little [5]. The review paper by Lefevre *et al.* [195] focuses on real-time techniques in the uncompressed domain.

3.1 Transitions in Video

The earliest films were made using a camera fixed in orientation and position, and consisted of very short depictions of particular scenes. The technology of cinema developed rapidly, and the narrative power of cinema was greatly increased by the advent of increased film stock lengths, film editing techniques, and mobile cameras. At this stage, a film could be constructed by concatenating individual *shots*. The shot is the fundamental film component, consisting of a single contiguously recorded video sequence [75]. Within a shot, scene transitions such as a movement from indoors to outdoors can be introduced by changes in camera position. Thus film and video footage can contain both inter-shot transitions (or simply shot transitions) and intra-shot transitions. Figure 3.1 shows a selection of shots from the 1999 film 'La Fille sur Le

Pont' [190]. The shots are connected by cut transitions. Each row shows the first, middle, and last frame of the shot.

Shot change detection, also called shot boundary detection and temporal video segmentation, has been an active area of video processing research since at least as early as 1990 [228], and continues to attract attention in the present day [69]. It is a key prerequisite for automated and semi-automated video processing systems of any scale. In many video indexing and content-based retrieval systems, the shot is the retrieval unit. Automatic video summarization is generally shot-oriented, where representative frames from each shot are chosen to represent the entire video. Spatiotemporal video processing algorithms, such as motion estimation, rely on assumptions of temporal smoothness in video; shot boundary detection is required here because shot boundaries violate these assumptions. In computer assisted video processing, having the source material automatically decomposed into shots facilitates more rapid setting of parameters than having the operator select ranges of frames.

As well as enabling the processing of individual shots as units, shot transitions in themselves have considerable significance in film semantics. Rules informing the selection and juxtaposition of shots constitute a significant portion of *film grammar*. Experiments by Kuleshov in 1918 showed how editing could dramatically affect an audience's interpretation of film footage. The *montage theory* developed by Sergei Eisenstein in the 1920s built on this phenomenon to describe how artful editing techniques constitute in themselves an important part of the narrative and affective content of a film [68]. Vasconcelos and Lippman [313] have shown that shot length is correlated with genre.

The large majority of shot change detection algorithms target inter-shot transitions. The detection of other, intra-shot transitions is considered a false alarm in this context. On the face of it, this approach seems appropriate—however, there are two pertinent concerns. Firstly, shot change detection in this strict sense is not always a well-posed problem, for reasons discussed below. Secondly, in most applications of shot change detection, including those mentioned above, significant intra-shot transitions are as important as inter-shot transitions [135].

Shot transitions are introduced at the editing stage, and fall into two principal types. *Cuts* are transitions where two successive frames are from different shots. The abrupt nature of these transitions makes them relatively easy to detect in most cases. Gradual transitions are slower, involving a temporal region of overlap between two sources. There are various types, including *fade in* and *fade out* transitions, where the shot is faded in from, or out to, a black background; *dissolves*, where the overlapping regions of two shots are linearly blended; and *wipes* and *page turns*, in which one shot appears move out of the frame to reveal or be replaced by the succeeding shots, and *morphs*. These transition types have been categorised by Hampapur [133] into *spatial effects*, including wipes and page turns, *chromatic effects*, being mostly fades and dissolves, and *spatio-chromatic effects*, such as morphing. Of these gradual transition types, most work has been directed towards the detection of fades and dissolves, with some researchers also targeting wipe transitions.

Figure 3.1: The first, middle, and last frames of six shots from the film 'La Fille Sur Le Pont' [190].

3.2 Shot Change Detection Systems

Shot change detection is predicated on the assumption that there will be a change in the image content across the shot boundary. Therefore, all shot change detection systems rely on the extraction of some features from the video stream to characterise the individual frames. Comparing the features of two frames then provides a measure of how similar they are. Adjacent frames that are highly dissimilar are likely to belong to different shots. This process is illustrated in overview in figure 3.2.

A wide variety of features for characterising frame content have been proposed in the shot boundary detection literature, and for many kinds of features numerous similarity metrics have been proposed. In figure 3.2, intensity histograms and points with high local gradient are presented as illustrative features. Whatever the features and feature comparison method selected, they are used to generate *dissimilarity traces*, describing frame-to-frame similarity over the temporal extent of the film. High dissimilarity values generally indicate that some transition has occurred between two frames. The final stage of the process is then to classify this transition as an inter-shot / editing transition, such as a cut or a fade, or an intra-shot transition resulting from a change in camera position. Subsequent sections of this chapter present a comprehensive review of the features and feature comparison methods used in shot change detection systems, and of the techniques used to detect and classify boundaries in the dissimilarity traces. These follow the section below, describing why automatic shot change detection is a challenging undertaking.

3.3 Factors Complicating Shot Change Detection

Since the earliest shot change detection research, most papers in the area have reported very good detection performance, particularly for cut detection. However, it must be borne in mind that in a practical sense, the shot change detection task is getting more difficult as the limitations on digital video storage capacity expand. For example, the cut detection method proposed by Arman in 1993 [12] was tested on about 6 minutes of broadcast news footage, whereas in the current TRECVID shot boundary detection task, 7.5 hours of video material is provided [275]. This vast increase in the size of the typical digital video corpus results in a much wider range of video characteristics. The shortcomings of early, simpler shot change detection systems are revealed in evaluation against these larger corpora, and continuing research in shot change detection is driven by the challenge of maintaining high performance over an ever-widening range of digital video material.

3.3.1 Similarity characteristics

Of the various characteristics of video that make shot change detection difficult, high intra-shot activity is the most commonly mentioned. Fast camera or object motion, and rapid changes

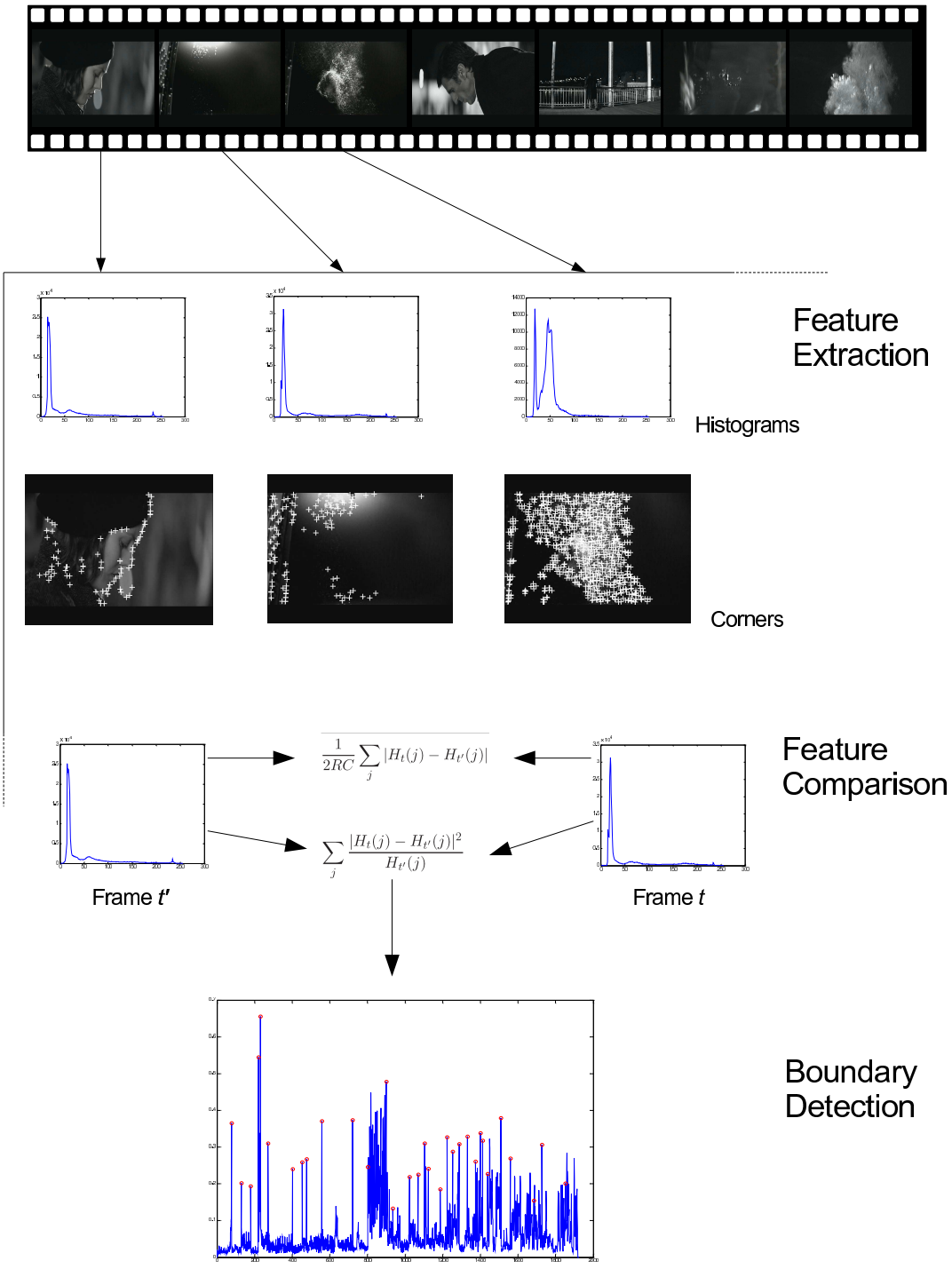


Figure 3.2: Shot Change Detection System Overview

of illumination (as with camera flashes), can introduce high frame-to-frame dissimilarity, and thereby a high false alarm rate.

The complementary issue is that of sequences with low inter-shot variation, for example where successive shots have similar intensity distributions [210]. In sports footage, successive shots can have very similar chrominance characteristics [1,56], for example where they show the playing field from only slightly different angles.

3.3.2 Film degradation

It is generally more difficult to detect shot transitions in video or film footage exhibiting defects or degradations, and various authors have presented shot change detection algorithms designed with this consideration in mind. Flicker is a rapid temporal variation in the mean brightness of a sequence [254]; several systems for shot change detection in the presence of flicker have been presented [10,131]. Systems designed to be robust in the face of camera flashlight activity [144] may also be expected to cope with flicker. Impulsive defects, such as dropout and dirt, can trigger false cut detections, especially where adaptive thresholds are in use. A histogram-based system using sub-frame blocks designed to be resilient to large blotch defects is described in [180]. Machi and Tripiciano suggest a semi-automatic shot cut detection system suitable for use with heavily degraded footage [211].

3.3.3 Shot dynamics

The dynamics of shot changes in a particular sequence can introduce difficulties. The simplest example of this is where shot changes occur in rapid succession, possibly violating *a priori* assumptions regarding shot length. This causes particular difficulty when the shots in question display high levels of motion activity—which is the kind of sequence in which rapid shot changes is stylisticaly appropriate. The shot in the second row from the bottom in figure 3.1, for example, is only 11 frames long, and the cut connects two shots with high activity.

3.3.4 Editing style

The great majority of mainstream and Hollywood movies are edited in a particular style known as *continuity editing*. The aim of this style is to make the work of the film editor as unobtrusive as possible, so that the audience's attention is focused on the narrative. One of the conventions of continuity editing is the 30° rule [139], which states that a shot change should involve a shift in camera angle of at least 30°. Any smaller change of angle between shots (potentially including 0°, where the camera is stopped and started with no change in orientation) tends to be perceived as a glitch in the film, rather than a shot change.

The 30° rule facilitates shot change detection in that the new camera angle will generally introduce a substantial change in the video background and composition. However, it is a convention commonly flouted in less mainstream material, where *jump cuts* [139] are often used.

The technique was introduced in the 1960 film 'À Bout de Souffle' [120] and has been a hallmark of *nouvelle vague* cinema since. Jump cuts are also commonly used in music videos. Figure 3.3 shows some examples of jump cuts from 'À Bout de Souffle', and figure 3.4 shows some from a 2004 music video [198]. Shot change detection in these cases is substantially more difficult than in more conventionally edited sequences, because of the high similarity of frames on either side of the shot change boundary. Furthermore, it is not always clear whether these small jump cuts should be considered shot changes at all.

Some researchers have found that an observer-generated ground truth for cut transitions exhibits variation between observers and inconsistency within repeated evaluations by a single observer [116,117,166]. In other words, cut detection is in some degree a subjective measurement.

3.3.5 Non-Sequential Shot Structure

The assumption is generally made that video and film material consists of a sequence of shots, each from a single camera, possibly having small areas of overlap at gradual transitions. Modern video compositing techniques violate that assumption. For example, a broadcaster's logo, as commonly displayed in an upper corner of the screen, will persist unaffected over cuts and dissolves, which can have implications for model-based approaches. In news footage, it is common for a subtitle panel to be superimposed over the start of a segment, introducing the presenter and location of the shot. This panel may persist over an early shot change in the segment, and will usually abruptly disappear or fade out after a few seconds later. In some highly stylised music video footage, shots are effectively not sequential at all, but rather overlapping and composited by various techniques over the entire sequence. In all these cases, what is at work is the superimposition of multiple sources, each of which can contain shot transitions or other discontinuities. Two examples of this sort of effect are shown in the top two cuts of figure 3.4. In both examples, footage of a face is composited over a background source, and a jump cut is introduced in the face source while the background source remains continuous. These considerations are not addressed by current work in shot change detection.

3.4 Features for Shot Change Detection

The first stage in shot transition detection is the choice of features to represent an image, along with a measure to describe frame dissimilarity based on these features. As outlined above, the design criteria for these features is that as far as possible, they should be insensitive to camera motion, object motion, and changes in illumination. This section describes the commonly employed features for shot change detection. For further review material specifically addressing features for frame similarity, see the discussions presented by Otsuji *et al.* [246], Ford *et al.* [112] and Bescos *et al.* [22].

Figure 3.3: Two jump cuts from the film 'À Bout de Souffle' [120]. Each row of images shows four consecutive frames. The cut occurs after two frames in each case.

Figure 3.4: Three jump cuts from the music video 'Mutescreamer' [198]. Each row of images shows four consecutive frames. The cut occurs after two frames in each case.

3.4.1 Direct Image Comparison

The simplest approach to measuring frame similarity is simply to compare pixel values at corresponding sites in a pair of frames. In its most basic form, this approach suggests the mean absolute difference measure:

$$D_{MAD}(n, n') = \frac{\sum_{r=1}^{R} \sum_{c=1}^{C} |I_n(r, c) - I_{n'}(r, c)|}{RC} \tag{3.1}$$

where n and n' are the frames being compared, and $I_n(r, c)$ is the intensity value at site (r, c) of frame n in a video with dimensions $R \times C$. Use of D_{MAD} extends back at least to 1992 [170, 229, 349].

The D_{MAD} measure is sensitive to all intra-shot activity, including local and camera motion, noise, and illumination changes. Local motion, in particular, can result in a small fraction of pixels having a very high absolute intensity difference. Using D_{MAD}, this is indistinguishable from a shot change resulting in a greater number of pixels having a moderate intensity difference. A slightly more complex distance measure disambiguates these situations. Instead of taking the mean absolute difference, the fraction of pixel sites that change value by more than some threshold T is used. The resulting measure is

$$D_{CF} = \frac{|\{(r, c) : |I_n(r, c) - I_m(r, c)| > T\}|}{RC} \tag{3.2}$$

where the subscript CF stands for 'change fraction'. Use of D_{CF} actually precedes use of D_{MAD}, having been first presented for cut detection in 1991 by Otsuji *et al.* [174].

Most shot changes can still be perceived in video after considerable downsampling—and downsampling, or smoothing, reduces the effects of image noise and motion. The more the image is downsampled, the greater the attenuation of these effects. Thus downsampling the image sequence prior to applying shot change detection can be helpful, particularly when using direct image comparison measures.

Among the researchers to recognise this, Zhang *et al.*are among the earliest. They employed a 3×3 averaging filter prior to pixel differencing in their 1993 paper [349]. Ardizzone *et al.*used downsampling to an image size of 16×16 before applying pixel differencing [11]. This extensive data reduction was motivated by the desire to facilitate real-time processing of the difference image by a neural network, but the authors recognise that resilience to motion is introduced by the downsampling. Good cut detection performance is reported, indicating that even 16×16 is not too small for cut detection.

Any direct image comparison metric will be sensitive to camera motion, as this can introduce changes in intensity values at a large fraction of pixel sites over the entire extent of the image. Global motion compensation can be used to reduce this effect. One of the more sophisticated implementations of this refinement was presented by Bouthemy in 1996 [42, 43]. Here affine camera motion parameters are estimated using an iterative robust estimation technique (outlined in chapter 2); the number of pixels in the *support region* (i.e. those pixels obeying the affine

model) is then used as the similarity measure. As the support region is updated according to the pixelwise error after each iteration, this scheme is essentially an application of D_{CF} with global motion compensation.

The measures D_{MAD} and D_{CF} described above consider the pixel differences in aggregate, i.e. summed or counted over the entire frame. A more sophisticated approach to direct image comparison is to use the *distribution* of pixel value changes from one frame to the next. The work by Aigrain and Joly [6] is an early example in this category. The distribution of absolute intensity differences, pixel-to-pixel, is analysed with reference to models describing the expected difference values for noise, motion (both local and global), and shot transitions. The number of pixels in the difference histogram falling inside ranges determined by the models is counted, and this count determines whether a shot transition is declared or not. Up to 100% performance on cut detection is reported, and 80% for gradual transition detection.

The two-dimensional histogram $H_{n,n'}(i, i')$, describing the number of pixels sites which change from value i to value i' between frames n and n', can also be used for shot change detection. For similar images, most of the mass of the histogram will lie along the main diagonal of H, as most pixel sites will not change in value.

One of the earliest applications of this two-dimensional histogram approach, introduced by Li and Wei in 2000 [200], exploited this fact directly. Here the histogram is called the *Joint Probability Image* (JPI). A scene change is declared if the fraction of histogram mass more than five pixels away from the diagonal exceeds a fixed threshold. This measure is in fact equivalent to thresholding D_{CF}, and so this paper does not offer an advance in transition detection. However, the paper does describe how the JPI can be used for transition classification. This contribution is summarised in the section on transition classification below.

Applying information theoretic measures to the change histogram does improve on the subtraction based methods, however. One such measure is the mutual information between frames, defined as

$$D_{MI}(t, t') = -\sum_{x,y} H_{t,t'}(x, y) \log \left(\frac{H_{t,t'}(x, y)}{H_t(x) H_{t'}(y)} \right) \tag{3.3}$$

Here $H_t(x)$ is the fraction of pixel sites having the value x in frame t.

The first use of D_{MI} for shot change detection was by Butz and Thiran in 2001 [48]. The mutual information is found using frame intensity values, and camera motion is compensated for using an affine model.

D_{MI} has also been deployed by Cernekova *et al.*in 2002 [55, 56]. Here the similarity metric is the sum of the mutual information over the R, G, and B image channels. An additional information-theoretic measure, the joint entropy, is also employed. For a single channel, the joint entropy is defined by

$$D_{JE}(t, t') = -\sum_{x,y} p_{t,t'}(x, y) \log p_{t,t'}(x, y) \tag{3.4}$$

D_{JE} is summed over the R, G, and B channels to obtain an additional dissimilarity measure.

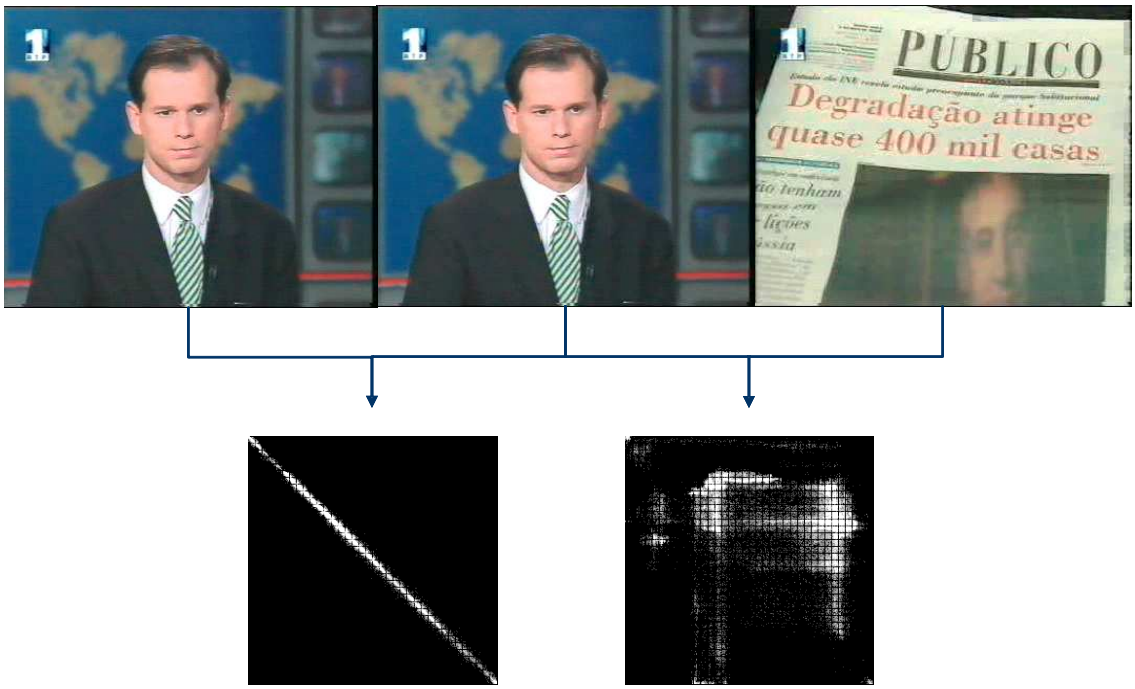


Figure 3.5: Joint probability images [200] within a shot and across a shot boundary

D_{MI} suffices to detect cut transitions, while D_{JE} is used to detect fades. Good resilience against the effects of camera flashes is demonstrated. However, this method is only applicable to fade in and fade out transitions, and not to dissolves between shots. Global motion compensation is not employed.

3.4.2 Statistical Image Comparison

The measures considered above are all based on quantifying the change in image value at corresponding locations. Motion in the video sequence will inevitably be registered by these measures. For the purposes of shot change detection, a difference metric insensitive to the effects of motion is preferable.

In most cases, motion in an image sequence results in a displacement or re-arrangement of the pixel values. The distribution of pixel values over the entire frame is largely unaffected by motion. This consideration motivates *statistical image comparison*, using on features that encapsulate the distribution of pixel values in a location-invariant way.

The fundamental feature for statistical image comparison is the pixel value histogram, $H_t(j)$, which describes the number of pixels having value j at time t. This is the earliest and most

widely-employed feature for shot change detection, first exploited in 1990 [228].

A great variety of histogram based frame comparison systems have been proposed; the variations include the resolution (bin width) of the histogram, the colour space used, and the method used to calculate the difference between two histograms.

Varying the bin width of the histogram is generally motivated by computational concerns and to avoid the 'curse of dimensionality', especially in three-dimensional colour histogramming. Using eight bits of resolution for each of three channels, for example, results in a 16 million bin histogram. Much coarser colour histograms have been shown to have good performance for shot change detection. For example, Zhang *et al.*use a histogram over R, G, B values where each colour level is quantised to two bits, resulting in a 64-bin histogram [349]. Outside of computational concerns, the question of whether there is an optimal bin width for shot change detection has not been considered.

Many shot change detection algorithms operate using intensity information alone. Shot transitions are readily perceptible to human observers in monochrome video footage, so there is sufficient information in the intensity channel for detection. Furthermore, algorithms using only intensity information are equally applicable to colour and monochrome material. However there are advantages to incorporating colour information in a shot change detection algorithm. For example, camera flashes introduce a change in the intensity histograms of video frames, but will not affect the chrominance information. This suggests the use of a colourspace in which luminance and chrominance are independent, such as YUV, HSV, or XYZ.

One difficulty in incorporating colour information in shot change detection is that chrominance information is indeterminate where luminance levels are low. Lee and Ip presented a system in 1994 operating in the HSV colourspace addressing this difficulty. Here each pixel value is recorded in either the hue histogram or the intensity histogram, depending on whether the saturation and brightness values both exceed fixed thresholds, or not. Frame comparison is based on the difference of both the intensity and hue histograms.

Numerous measures for histogram comparison have been proposed for shot change detection, and much of the work in histogram-based shot change detection focuses on selection and evaluation of this measure. The simplest histogram distance is the bin-to-bin difference, defined by

$$D_{B2B}(t,t') = \frac{1}{2RC}\sum_{j}|H_t(j) - H_{t'}(j)| \tag{3.5}$$

where j is an intensity level, typically varying from 0-255, and $H_t(j)$ is the number of pixels in frame t taking the value j. This measure is analogous to the D_{MAD} measure for direct image comparison. The system presented by Tonomura in 1991 [303] used D_{B2B}.

A second commonly employed histogram comparison is the χ^2 test

$$D_{\chi^2}(t,t') = \frac{1}{(RC)^2}\sum_{j}\frac{|H_t(j) - H_{t'}(j)|^2}{H_{t'}(j)} \tag{3.6}$$

also called the colour correlation. The χ^2 test applied to image intensity histograms is the earliest employed frame similarity measure for shot detection, used by Nagasaka in 1990 [228] and numerous systems since then [155, 229, 308]. Squaring the difference value essentially introduces a higher weight on large difference values, and so is supposed to enhance the dissimilarity across a shot boundary. However, as histogram differences due to motion are also squared, performance is not necessarily improved [349].

The histogram intersection [292] has also been proposed. In its basic form, it is defined by

$$D_{HI} = 1 - \sum_j \frac{\min(H_t(j), H_{t'}(j))}{RC} \tag{3.7}$$

A modified version was evaluated for shot change detection by Kasturi *et al.* [166], defined by

$$D_{HI'} = 1 - \sum_j \frac{\min(H_t(j), H_{t'}(j))}{\max(H_t(j), H_{t'}(j))} \tag{3.8}$$

A comprehensive review and evaluation of histogram-based cut detection was presented by Kasturi *et al.*in 1996 [166]. This work encompassed eight colour spaces (RGB; HSV; YIQ; XYZ; L*a*b*; L*u*v*; Munsell; opponent colour axes) and four histogram comparison methods (bin-to-bin; histogram intersection; average colour difference, bin differencing with a neighbourhood, χ^2). No single histogram distance emerges as the clear winner in this evaluation.

The Kolmogorov-Smirnov measure, D_{KS}, is the maximum bin difference between cumulative histograms:

$$D_{KS}(t, t') = \max_j (|CH_t(j) - CH_{t'}(j)| \tag{3.9}$$

This measure was first proposed for shot detection by Patel and Sethi [248]. It was shown to outperform D_{B2B} and D_{χ^2}; the improved performance is attributed to the smoothing inherent in the use of cumulative histograms.

In 2000, Ford *et al.*published a comprehensive evaluation of the histogram difference measures proposed up to then for shot change detection [112], including those described above and some others. This paper establishes that the D_{KS} measure has the best performance, in terms of the ROC area, for global intensity histograms—though D_{χ^2} is better for colour histograms. Tan *et al.*in 2003 [296] proposed an improved Kolmogorov-Smirnov measure, $D_{KS'}$. Their suggestion is that the global smoothing of the cumulative histogram be combined with the sensitivity of bin-to-bin comparison. The new measure is denoted $D_{KS'}$, where

$$D_{KS'}(t, t') = \sum_k \frac{D_{KS}(t, t'; k)}{N_S} \tag{3.10}$$

This is the sum of the Kolmogorov-Smirnov distances over sections k of the intensity range. The distance over each section is found using

$$D_{KS}(t, t'; k) = \max_j (|CH_t(j; k) - CH_{t'}(j; k)|) \tag{3.11}$$

and the cumulative histogram value for intensity level j in section k of the range is given by

$$CH_t(j;k) = \sum_{m=kM}^{kM+j-1} H_t(m) \tag{3.12}$$

where M is the section size.

More recently proposed histogram distance measures include the Kullback-Leibler histogram distance

$$D_{KL}(p,q) = \int p(x) \log \frac{p(x)}{q(x)} dx = \int (1-p(x)) \log \frac{1-p(x)}{1-q(x)} dx \tag{3.13}$$

and the cross-entropy measure

$$D_{CE}(p,q) = A \int q(x) \log \frac{q(x)}{p(x)} dx + B \int p(x) \log \frac{p(x)}{q(x)} dx \tag{3.14}$$

Both of these measures have been described and evaluated by Kim and Park [173]. Both measures are shown to outperform the histogram difference and histogram intersection measures, in particular on material with luminance changes.

Shih *et al.* [52] model the change in frame histograms as a flow and deploy the Reynolds Transport Theorem [257] to detect significant changes in this flow, and thereby detect shot changes. This approach is appealingly well-founded, and suited to detection of all kinds of shot transitions. However, the authors do not show results on videos containing significant motion, and suggest that the method would be less effective on such sequences.

The Earth Mover's Distance (EMD) [264, 265] is a histogram difference metric well-suited to image histograms and widely used in image retrieval. It appears not to be extensively used in shot change detection [262]. A description of this distance appears in chapter 7 of this thesis.

The great majority of of histogram-based shot change detection methods use pixel *values*. In other words, frames are characterised by the distribution of intensity or colour levels they contain. An interesting variation on the histogram approach developed by Sze *et al.* [294] attempts to calculate the distribution over *structural elements*. Each frame is divided into small blocks, and for each block the *coloured pattern appearance model* (CPAM) is found. The CPAM is a representation of the structure and colour in a block, informed by the perceptual qualities of the human visual system. Vector quantisation is used to reduce each block to one of 256 codewords, and frame similarity is computed based on histograms of the codewords in each frame.

3.4.3 Block Based Similarity

A shot change generally introduces change across the entire spatial extent of the frame. This suggests that global features will be sufficient for shot change detection. However, strong localised changes due to rapid motion, pathological activity such as explosions, or large frame defects, can introduce large changes in global features and result in false alarms. Furthermore, calculation and comparison of features over the entire frame extent can be relatively expensive

in computation effort. For both of these reasons, block-based systems have been introduced, in which the frame extent is divided into blocks, or windows, which may be overlapping or non-overlapping. Comparison between multiple block pairs is then used as the frame dissimilarity measure. Comparing the entire image block-by-block improves resilience to localised effects. Alternatively, comparing only some subset of blocks introduces a computational saving.

Any block based scheme will be based on a difference measure between image regions. All the direct and statistical image comparison methods described above can be applied to sub-blocks for this purpose. Pixel differencing over individual blocks has been proposed by Zhang [349] and Yusoff [342]. Systems using block-based histogram comparison have been developed by Nagasaka and Tanaka [228, 229], Ueda *et al.* [308], Swanberg [293], and Kokaram *et al.* [180].

A very simple approach to block comparison is to use block intensity statistics [132, 165]. Here, the block comparison measure is defined by

$$D_{BS}^{k} = \frac{\left[\frac{(\sigma_t^k)^2+(\sigma_{t'}^k)^2}{2} + \left(\frac{\mu_t^k-\mu_{t'}^k}{2}\right)^2\right]^2}{(\sigma_t^k)^2(\sigma_{t'}^k)^2} \tag{3.15}$$

where μ_t^k and $(\sigma_t^k)^2$ are the mean and variance of the gray values within block k of frame t. This can be considered as a slightly more sophisticated approach to extreme downsampling, in that the inclusion of the block variance gives a crude measure of how representative the block mean is of the actual block content. The individual block differences can be summed to give a frame difference, or the number of block differences exceeding a threshold can be used.

Normalised correlation in the frequency domain can also be used for block similarity [256, 315]. This provides some resilience to changes of illumination, and it is also noted that high performance Fourier transform implementations are available for most platforms.

Various refinements to block-based shot change detection systems can be considered, irrespective of the block difference measure used. For example, in early schemes blocks are compared in spatially corresponding pairs. The effects of motion can be mitigated by comparing each block in frame n with the most similar block in some neighborhood in frame $n+1$. This was first proposed by Shahraray [271], and has been used in numerous systems since then [135, 256, 335, 342]. The result is a discontinuity measure essentially the same as the displaced frame difference (DFD)

$$D_{DFD}(t,t',k) = \min_{-M\le u,v\le M} \frac{1}{M^2} \sum_{x=x_k}^{x_k+M-1} \sum_{y=y_k}^{y_k+M-1} (|I_t(x+u,y+v) - I_{t'}(x+u,y+v)|) \tag{3.16}$$

where block number k has its top left corner at image coordinates x_i, y_i.

Establishing these block correspondences in the best-match sense is closely related to optic flow estimation, which has itself been used for shot change detection. Within a shot, adjacent blocks are likely to move together, while across a shot boundary the blocks will be matched somewhat randomly and the coherence of the flow field will be lost. Thus the flow field smoothness

can be used as a frame similarity measure. Systems exploiting this include the work of Akutsui in 1992 [7] using block matching, and the more recent approach of Cheong and Huo [59, 60], exploiting the optic flow constraint.

Park *et al.* [247] have pointed out that the the blur introduced by fast motion can introduce a high blockwise difference within a correctly matched pair of blocks. They propose inversely weighting D_{DFD} by the Euclidean distance between matched blocks to account for this.

As a scene change will affect the entire spatial extent of an image, it may be sufficient to compare a portion of the blocks to determine whether a scene change has occurred. The work of Xiong *et al.* [329, 330] uses comparison of block mean value in less than half of the block windows to detect transitions; this approach is called *net comparison*. Adjeroh *et al.* [4] present a formalised approach for determining the appropriate block size, given a tolerable probability ρ of a false match between frames. The calculations are an extension of work on probabilistic histogram matching [193]. The minimum proportion of the image to match to attain the desired probability of error, f_{min}, is given by:

$$f_{min} \geq \frac{\log(\rho)}{\sqrt{N}\log\left(\frac{1}{1+\frac{4}{\epsilon\sqrt{N}}}\right)} \tag{3.17}$$

where ϵ is a small constant (e.g. 0.001) relating to the amount of noise in the image sequence. The optimal window size, W_o, which minimizes the number of comparisons necessary, is found by

$$W_o = \left[N\left(1-\exp\left(\frac{1}{N}\ln\left(\frac{1}{2(N+1)}\right)\right)\right)\right]^2 \tag{3.18}$$

and the dimensions of this optimal window size, $R_o \times C_o$, for an image with dimensions $R \times C$, are then

$$R_o = \left(\frac{W_o R}{C}\right)^{\frac{1}{2}}; C_o = \left(\frac{W_o C}{R}\right)^{\frac{1}{2}} \tag{3.19}$$

The block grid will contain $N_o = N/W_o$ windows. Of these, the number that should be compared to attain a probability of false match, ρ, is the smallest integer N_{min} satisfying.

$$N_{min} \geq N_o f_{min} \tag{3.20}$$

For example, for an image sequence with dimensions 720×576, the optimal window size according to 3.18 is about 16×12. With the ρ and ϵ parameters set to 0.0001 and 0.0001 respectively, the f_{min} parameter is found to be 0.155, and so 335 of the 2160 blocks should be compared to assure a match. The authors note that for some combinations of N, ρ, and ϵ, the formulae result in $f_{min} > 1$. This suggests that the specified probability of a false match cannot be attained given the noise characteristics of the image sequence. In the absence of further elaboration of the quantitative relationship of the ϵ parameter to the noise characteristics, it is difficult to make a definitive interpretation.

Two aspects of these calculations are not addressed in [4]. The authors do not investigate how the chosen fraction of windows should be distributed about the block grid, although they

do identify this as a topic for investigation. Of greater concern is the fact that the derivations of the formulae do not seem to have been published.

Whether all blocks are being compared, or only a subset, it is likely to be the case that the largest block differences between two frames in the same shot will be due to localised effects such as motion. Thus discarding the largest block differences reduces the effects of large localised motion and image defects [68, 180, 256]. The earliest shot detection system recognised this fact, in which colour histograms are compared for 16 blocks spanning the frame, and the sum of the 8 smallest block difference measures used as the frame similarity measure [228]. In 1995, Shahraray [271] introduced a non-linear order statistics filter to combine block match values. Here the weight assigned to a block match value depends on its rank in an ordered list of match values, such that poorer matches are assigned lower weights.

3.4.4 Structural feature similarity

The similarity measures described above are essentially 'blind', or content-unaware, in that no analysis of the video content is made to inform how frames can be compared most effectively for shot change detection. The approaches considered here are feature-based, in that the most informative or important regions of the video frames are detected, and shot change detection is based on analysis of these regions in particular. Because successive frames within a shot depict very similar scenes, the location of most of these local features should not change significantly between two successive frames in the same shot. The similarity measure is thus based on the extent to which features can be tracked from one frame to the next.

One of the earliest works in this vein is the edge-based system published by Zabih *et al.* [344, 345]. They define the *edge change fraction* as their similarity measure, based on thresholding the distance from an edge pixel in one image to the closest edge pixel in an adjacent image. A translational global motion compensation step is included to account for camera movement—interestingly, the authors reject the use of MPEG compression vectors for camera motion extraction. Their algorithm generates smoother dissimilarity traces than both intensity histogram difference and the chromatic scaling method of Hampapur [133]. However Lienhart states that overall, histogram techniques can be just as effective as this edge tracking method [202].

Finer-grained feature tracking has also been proposed for shot change detection. Whitehead *et al.* [324] have presented a system using corner tracking in the intensity channel, with a translational motion gradient-based feature tracking algorithm. The scheme is applied using a multi-resolution image pyramid, for robustness to large motion. Because of the lack of spatial context implied by the small scale of the features used, features may occasionally be erroneously tracked across a shot cut boundary. However, because features are generally tracked in clusters corresponding to objects in the scene, the erroneous matches can be removed by analysis of the minimum spanning trees of the tracked and untracked features. The algorithm is shown to outperform histogram based and pixel-differencing approaches on a variety of genres with

challenging cut dynamics.

An interesting variation on feature tracking for shot change detection has recently been presented by Boccignone *et al.* [34, 35]. This method is grounded in properties of attention in the human visual system. In particular, the *fovea*, the high-acuity area of the retina, is so small that only a small proportion of a scene can be taken in in a glance. To compensate for this, the eyes *saccade* (move rapidly) over a scene, taking in localised windows at each movement. This gives the illusion of perception of the entire scene, but in fact changes to visual detail across scene cuts, and continuity errors, are rarely noticed by an observer due to this property of foveated attention. An idealised model of a human observer is used to detect features and rank them in order of saliency. This results in a spatio-temporal visuomotor trace, essentially an idealisation of the foveation points of an idealised observer when presented with the scene. The colour, shape, and texture characteristics of each foveation point are extracted. The frame similarity measure is then based on comparison of the visuomotor traces for each frame, using the characteristics of corresponding foveation points.

Some researchers perform object segmentation of frames in the video, and compare the results of this segmentation between frames as their similarity measure. The work of Hsu [150, 151] presented in 1994 uses implicit segmentation to extract video elements in the 3-D video volume. Motion discontinuities can be detected by consideration of the sign of the Gaussian and mean curvature of these volumes, or *spatiotemporal surfaces*. Heng and Ngan use edge tracing and tracking over multiple frames to delineate object boundaries [142, 143], incorporating colour information from each side of the edge as an additional feature. The approach is shown to have smoother behaviour than the edge change factor presented by Zabih [344] over gradual transitions. More recently, Vadivel [310] uses segmentation of the I-frames of MPEG-compressed video in the HSV colour space, and compares the locations, sizes, and colours of the resulting objects to detect a transition.

3.4.5 Shot Modelling and Feature Clustering

The systems described above are all based on comparison between two frames. No wider context is incorporated in direct or statistical image comparison. A different approach to shot change detection is to perform feature extraction for the entire sequence first, and then to consider the features for all frames simultaneously for shot change detection.

These approaches can be thought of as constructing a spatio-temporal image, or sequence matrix, summarising the video sequence. Shot transitions are then supposed to be readily identifiable within this image. An early example of this approach is due to Han, presented in 1999 [134]. Here each frame is represented by its horizontal and vertical projections. Figure 3.6 (a) shows a spatiotemporal image made up of the horizontal projections of some news footage. In [134], PCA is applied to these matrices and the largest principal component is disregarded to reduce the effect of luminance changes.

(a) Horizontal projection image

(b) Visual rhythm image

Figure 3.6: Two varieties of spatio-temporal image for 640 frames of news footage; the columns of the spatio-temporal images correspond to the frames of the video. The shot transitions in the video are clearly visible in these images. It also clear in these images how compositing effects can break the strict sequential shot model. The rectangular feature in the lower middle of both images and the unbroken white line near the top of (b) correspond to composited image elements that persist over shot transitions in the underlying video.

The *visual rhythm*, presented by Chung in 2000 [66], constructs a similar spatio-temporal image made up of pixels sampled uniformly along the main diagonal of each frame. An example of a visual rhythm image is shown in figure 3.6 (b). Transitions such as cuts, wipes, and dissolves can then be identified by analysis of the image. The visual rhythm is also employed by Guimaraes [127–129].

The approaches above use spatial domain features to construct the spatio-temporal image. The sequence matrix can also be constructed using frame histograms. Guimaraes in 2002 presented an approach where the sequence matrix is described as the 'visual rhythm by histogram' [126]. Here fades are detected by direct analysis of this image.

Recent work by Lu and Tan [207,208] uses image histograms in HSV space computed over the entire sequence. Their system is presented for the refinement of an existing temporal segmentation, but could equally be used for *ab initio* shot change detection with an inital assumption that the entire sequence is a single shot. The colour histogram for each shot is simply the bin-wise median of the histograms of the frames within the shot. The model for each shot is the shot histogram along with a truncated exponential describing the distribution of histogram differences within the shot. The likelihood of a frame histogram X_t being generated by shot k is

$$p_k(X_t) = \begin{cases} \beta_k \exp(-\alpha_k |X_t - S_k|), & \text{if } 0 \leq |X_t - S_k| \leq 1 \\ 0, & \text{otherwise} \end{cases} \tag{3.21}$$

The parameter α_k relates to the visual activity in the shot and can be determined from the data. β_k is a normalising constant which can be determined analytically.

Given an M-frame sequence divided into two shots S_1 and S_2, the log-likelihood ratio for

shot membership of a frame at t with histogram X_t is

$$\begin{aligned} J(t) &= \sum_{n=t}^{M} \frac{p_2(X_n)}{p_1(X_n)} = \sum_{n=t}^{M} \frac{\beta_2 \exp(-\alpha_2 |X_t - S_2|)}{\beta_1 \exp(-\alpha_1 |X_t - S_1|)} \\ &= \sum_{n=t}^{M} (-\alpha_2 |X_n - S_2| + \alpha 1 |X_n - S_1| + (M - t + 1) \log(\frac{\beta_2}{\beta_1}) \end{aligned} \tag{3.22}$$

New shot detections are detected by splitting each initial shot S into two proposed shots of equal length S_1 and S_2, and evaluating the log-likelihood ratio of the shot membership hypotheses for each frame in S. This partitioning introduces discontinuities in the slope of the log-likelihood ratio at shot boundaries. The procedure is recursively applied until no further shot boundaries are found. The log-likelihood test can also be used to detect false shot boundaries in the initial temporal segmentation.

Sequence matrices made up of image histograms, especially colour histograms, can be very large, and suggest dimensionality reduction for shot change detection. For example, in the work by Gong and Liu [121], each column of the sequence matrix is the 1125-dimensional vector comprising the 125-bin 3D RGB histogram of each of nine blocks covering the spatial extent of the frame. The singular value decomposition (SVD) of the matrix is found, and the 150 largest singular values retained. The frame difference measure is then the weighted Euclidean distance between their feature vectors in this reduced-dimensionality space. Cernekova *et al.*have presented a similar approach, also employing SVD on the sequence matrix and using a cosine similarity measure [53, 54].

The approaches described above are off-line, in that shot changes are detected using data from the entire sequence, after an initial feature extraction pass. On-line sequence modelling can also be used. Liu and Chen [204] use PCA to analyse the temporal statistics of frame features. The model is recursively updated with temporal decay. The similarity measure is then the reconstruction error after projection of the feature vector into the eigenspace found by the model.

3.4.6 Frame Similarity in the Compressed Domain

A vast quantity of digital video material is stored and transmitted in compressed form. Most of this material is compressed using MPEG compression; DVD video and digital television use MPEG-2, while MPEG-4 is very popular for internet video. There has been considerable research into video processing in the compressed domain, i.e. into systems that can process stored digital video without fully decompressing it. The chief advantage of this is that the computational overhead of fully decompressing the video can be avoided. Furthermore, in some cases the information in the compressed data can be exploited, taking advantage of the computational effort already expended in compressing the material. For example, video compression schemes generally incorporate motion estimation, and for some purposes the motion vectors in the compressed data can be used instead of decompressing the material and applying optic flow analysis.

Numerous systems for shot change detection in the compressed domain have been presented [21, 25, 51, 107, 108, 145]. Wang *et al.*have recently presented a clear review of features in MPEG-compressed data for indexing and analysis [316]. Systems for shot change detection in the compressed domain can be categorised in two classes. *Image-based* approaches use methods similar to those described above for uncompressed data, adapted and applied to compressed video. Generally, the advantage of these schemes is the computational saving of avoiding full decompression. Non-image-based approaches exploit the *compression characteristics* of the compressed data, such that the compression of the video is used as a useful preprocessing step to furnish additional features for shot change detection.

Most video compression schemes use a blockwise Discrete Cosine Transform (DCT) transform for spatial compression. This results in two sets of features that can be used for shot change detection: the DC coefficients, related to the average luminance / chrominance in the block, and the AC coefficients, which account for variations within the block. The DC coefficients together make up the DC sequence, which is essentially a heavily downsampled copy of the original sequence. The downsampling factor is the same as the block size of the compression scheme, typically 8×8 or 16×16. As described above, downsampling does not introduce difficulties in shot change detection, and can be advantageous. Therefore, most shot change detection techniques developed for uncompressed data can be applied directly to the DC sequence.

Examples of schemes targeting the DC sequence include Yeo and Liu [336], who use pixel differencing, and Patel and Sethi [248], who use intensity histogram differencing. Hanjalic describes a block-matching approach designed to be particularly suited to a DC sequence [135]. Increasingly, researchers present results obtained on both uncompressed and compressed video [104, 105, 173].

Several researchers have recognised that edge features can be extracted or approximated directly from the AC coefficients, resulting in compressed domain analogues of edge matching techniques. including Lee *et al.* [194] and Song and Ra [278]. Song and Ra [277] have also developed a system for prediction of AC coefficients using only the DC coefficients, which has a lower computational cost than decoding and processing the AC coefficients in the video stream.

One of the earliest schemes targeting compression characteristics, due to Arman *et al.*, used an innner product of the DCT coefficients in JPEG-encoded video to quantify frame similarity [12, 13]. Bao and Guan [17] use histogram differencing on the DCT coefficients of each frame.

A significant source of temporal compression in MPEG video is the *bi-directional* encoding of some frames. Each frame is made up of macroblocks 16 pixels on a side, and in a bidirectionally predicted frame (B-frame) each macroblock can be intra-coded (I-block), prediction encoded (P-block), or bi-directionally encoded (B-block). I-blocks use spatial compression only. P-blocks are encoded with reference to a preceeding frame, and B-blocks with reference to a succeeding frame, or both the succeeding and preceding frames. P- and B- blocks, collectively known as inter-blocks, exploit the temporal redundancy typical in video material. The encoder chooses

block types within B-frames to maximise compression while preserving fidelity.

Because frames on either side of a shot transition are dissimilar, there is little temporal redundancy between them. This affects the block types chosen by the encoder for B-frames. Specifically, a B-frame that is the first frame of a shot will have few or no P-blocks, as the preceding frames are not useful for compression here. Similarly, where a B-frame is the last frame of a shot, very few B-blocks will be used. The ratio of B-frames to other frame types is an encoding parameter, and is generally high enough to facilitate shot change detection by analysis of B-frames.

Several cut detection methods exploit this property [51,106,176,219,249,250,349]. A method tuned to video encoded using H.264/AVC compression [205] uses a histogram of the intra-block prediction modes within a GOP to locate GOPs potentially containing a shot boundary, and uses the number of macroblocks with different inter-prediction modes within the GOP to localise the transition.

Each inter-coded block have an associated translational displacement, or motion vector, for motion-compensated compression. These motion vectors are also useful for transition detection and classification [109]. Haoran *et al.*have described a method for extracting both shot transitions and subshot transistions such as camera pan, tilt, and zoom, using the macroblock types to detect cuts and gradual transistions, and an analysis of the motion vectors to recover camera motion with a six-parameter affine model [137]. Camera motion from vectors and cut detection from macroblock types and vectors are also employed by Hesseler [146].

3.4.7 Genre Specific Approaches

Various shot change detection schemes targeting particular genres, particularly sports footage, have been presented. Rea in [260] characterises the geometry of a scene using the second order moment of the Hough transform; this approach works well for snooker and tennis video. Tekalp [1] has developed an approach suited to soccer footage, which is robust to changes of dominant colour.

3.5 Transition Detection

The feature extraction and comparison stage results in a dissimilarity measure between two frames, denoted $\delta(t, t')$. For similarity comparisons of adjacent frames, the dissimilarity trace $\delta(t)$ can be considered, where $\delta(t) = \delta(t-1, t)$ and $\delta(0) = 0$. The problem of transition detection then becomes that of detecting those values of $\delta(t, t')$ which correspond to a shot transition.

Figure 3.7 shows a difference trace created by direct pixel differencing for a sequence of almost 2000 frames from the film 'La Fille Sur Le Pont'. The locations of scene cuts are indicated by red circles. It is evident that in some regions of the difference trace, the scene cuts are marked by isolated spikes in the difference trace. However in more challenging regions of the sequence,

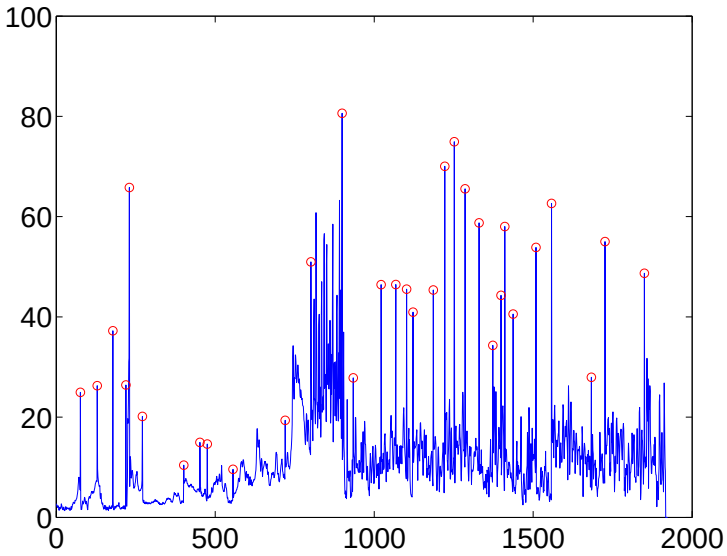


Figure 3.7: Difference trace over 1924 frames from 'La Fille Sur Le Pont', using frame-to-frame mean absolute difference of intensity values. Cuts are indicated by red circles.

---

the intra-shot differences are comparable in magnitude to the inter-shot differences. A more sophisticated frame difference measure would produce a cleaner difference trace, in which the scene transitions are better separated from intra-shot frame differences. Various degrees of sophistication can also be applied to the problem of identifying those difference values which correspond to a transition; it is this aspect of the transition detection problem that is now considered.

3.5.1 Thresholding

3.5.1.1 Fixed thresholds

The simplest method for transition detection is to detect locations where $\delta(t)$ exceeds a fixed threshold. The set of cut locations C is simply

$$C = \{t : \delta(t) > \tau\} \tag{3.23}$$

where τ is some empirically determined threshold. This approach is used in numerous systems [13, 174, 200, 228]. Zabih *et al.* [344] require that a discontinuity value exceed a threshold, and also that it be the largest value in some window, before the location is declared a scene break.

While cuts introduce a single spike in $\delta(t)$, over a fade or dissolve transition a lower plateau may be expected. The *twin-comparison* method due to Zhang *et al.* [349] relies on this behaviour using two thresholds: a high threshold T_h, and a low threshold T_l. Wherever the dissimilarity measure exceeds T_h, a cut transition is declared. Potential gradual transitions are supposed

to start where $\delta(t)$ transitions from below T_l to above T_l. If t_s is the start of the potential transition, the end point, t_e, is the largest t such that

$$(t > t_s) \wedge \big(\forall t' \in [t_s \dots t].\delta(t') > T_l\big) \tag{3.24}$$

If the sum of $\delta(t)$ over $[t_s \dots t_e]$ exceeds T_h, a gradual transition is declared.

Bouthemy *et al.*employ the cumulative sum Hinkley's test [18] to detect jumps in the frame dissimilarity measure in their shot change detection systems [42,43]. They advocate the test on the strength of its simplicity and low computational load, the fact that the entire signal history is taken into account, and the suitability of the test for detecting both gradual and abrupt transitions without modification of the parameters. Two thresholds are required in this scheme; it is reported in [43] that the same values were found effective over a range of footage types.

Various authors have proposed different schemes for smoothing the difference trace before applying a fixed threshold for cut detection. For example, Otsuji and Tonomura presented a projection detecting filter in 1993 [245,246]. This is essentially the application of one-dimensional dilation and erosion to better isolate peaks due to cuts. It is described as especially effective at dealing with the effects of film-to-video conversion, slow motion, and animation. Han and Tewfik describe a progressive nonlinear filtering step to isolate impulsive changes and smooth gradual transitions [134]. This approach is based on successive smoothing by averaging, but using the unsmoothed difference value where $\delta(t)$ has a high rate of change. They show that this smoothing is more effective than Gaussian, Wiener, and wavelet-based smoothing. Ewerth and Freisleben [100, 101] have a refinement for filtering difference traces arising from MPEG data.

In the work by Bescos *et al.* [23], an empirically determined fixed threshold is employed, along with other tests on the local characteristics of the difference trace including the slope, the average value, and others. Although described as clustering, the process as described is more a hierarchical decision process.

3.5.1.2 Adaptive thresholds

For the frame difference measures currently proposed, there will be no difference fixed threshold separating transitions from inter-shot differences in all footage. It is more prudent to use adaptive thresholding such that the data can itself be used to determine whether a value is significantly large or not. One of the first systems to use adaptive thresholds was described by Yeo and Liu in 1995 [336]. Here a shot cut is declared at time t if

$$\big(\delta(t) = \max(\{\delta(t') : t - m \leq t' \leq t + m\})\big) \wedge (\delta(t) > n\delta_{sm}) \tag{3.25}$$

i.e. where the $\delta(t)$ is the maximum value in a *sliding window* of size $2m$, and $\delta(t)$ is n times bigger than the second biggest value in the window, denoted δ_{sm}. The second criterion here is useful in distinguishing camera flashes from shot transitions; although both introduce high

difference values, a camera will result in two spikes in rapid succession, whereas a shot cut introduces a single, isolated spike.

Numerous other researchers have adopted similar approaches [36, 247, 263, 343]. Truong *et al.*have proposed using a slightly modified ratio test [307]:

$$c + \delta(t) \geq n \left(c + \sum_{t' \in \{t-m \ldots t+m\} t} \delta(t') \right) \tag{3.26}$$

Rather than using a sliding window, the threshold can be determined from consideration of difference values over the entire sequence. For example, Günsel *et al.* [130] presented a system in 1998 using K-means clustering to divide values from a single similarity measure into two classes, a transition class and a no-transition class. The entropic thresholding described by Sze *et al.* [294] chooses the threshold value so as to maximise the combined entropy of the transition values and the no-transition values.

3.5.2 Multiresolution Transition Detection

While cut transitions can in general be detected by comparing successive frames, gradual transitions require a wider window of comparison. Here comparisons are made between frames separated by a *skip parameter*, typically 15 or 20 frames. The intention is that the interval should span the transition, or enough of it, such that the frames being compared show a significant difference. Differencing frames with a high skip parameter increases the effects of camera and local motion, and so the resulting difference traces tend to be noisier than those calculated with successive frames.

As well as facilitating detection of gradual transitions, examining the video at a coarse temporal resolution is quicker than comparing all pairs of frames. A fine-grained frame comparison can then be invoked to isolate a transition identified within the skip interval. The system described by Drew *et al.*in 1999 [89] takes this approach, where a skip of 32 frames is used to localise transitions and binary search used within the interval for precise detection.

Bescos *et al.*calculate differences at every multiple of 5 frames from 5 up to 40, and detect shot transitions at each scale [24]. Intra-shot transitions due to fast camera motion are considered desirable along with inter-shot transitions. A transition introducing a peak in the difference trace at one scale will manifest as a plateau at all coarser scales, while a visual discontinuity, such as a camera flash or large local motion, will result in two peaks at all scales. This property is exploited to distinguish these two event types.

Chua *et al.* [65] apply Canny wavelet analysis to colour histogram difference values to detect both sudden and gradual transitions. The locations of all maxima in the difference trace at the finest resolution are considered potential transition points. Of these, locations that are not also maxima at coarser resolutions are rejected as due to noise.

3.6 Feature fusion

Many features have been described for measuring frame similarity, and it is unlikely that any single such feature will be appropriate for all footage. Various schemes for combining multiple features have been proposed to improve the performance of shot detection systems.

One straightforward consideration is that often the simplest and computationally cheapest features have poor performance, while more sophisticated measures have considerably higher computational cost. In this situation, the simple method can be applied first, and if it exceeds a 'certainty threshold' τ_c, a cut is declared. If the difference value is less than τ_c, but greater than a 'possible-cut threshold' τ_p, the more expensive similarity measure is invoked to make the decision. This is the approach adopted by Arman in 1993 [12] for compressed data, where the inner product of the DCT coefficients is the simple measure, and colour histogram comparison (which requires full decompression of the frames in question) is used as the expensive measure. A similar approach has been described by Huang [155].

Various authors have suggested the use of multidimensional clustering to detect video transitions. Here the values of each of the difference measures are combined to form a feature vector in a multidimensional space. Ferman and Tekalp use K-means in two dimensions, where the similarity measures used are a histogram difference and pixel differencing [103]; they report an improvement over the performance attainable with either measure alone. A similar approach has been proposed by Gao and Tang [115], using fuzzy c-means in two dimensions; the similarity measures here are again a histogram difference and pixel differencing.

Ren and Singh in their 2004 paper [262] evaluate both k-nearest neighbour and neural network classification for detection of cut, fade, dissolve, tilt, and pan transitions. In this work 18 features are combined, selected from a total of 139 similarity measures using sequential forward feature selection. It is found that neural network classification slightly outperforms the k-nn technique, with a 99.7% recognition rate and a 0.3% false alarm rate. The combined-feature classification approach is shown to outperform histogram comparison and motion comparison approaches.

Cernekova *et al.*in 2005 applied clustering to 11-dimensional feature vectors consisting of 10 features from Singular Value Decomposition (SVD) and 1 feature from a mutual information metric [53]. This approach is referred to as *feature level fusion*.

Instead of using clustering for classification of difference values, the *temporal evolution* of the features can be modelled. The most popular framework applied to this is the Hidden Markov Model (HMM) approach. One of the earliest such approaches was presented in 1998 by Boreczky *et al.* [39]. The dissimilarity measures here are luminance difference, motion vectors, and audio cepstral vector difference. In this work, each camera transition type is a state in the model. This limits the capacity of the model to capture within-state temporal behaviour of the feature vectors. In the more recent work by Bae *et al.* [15], three HMMs are used: one for each of the non-transition, gradual transition, and abrupt transition events. The feature vector in this case

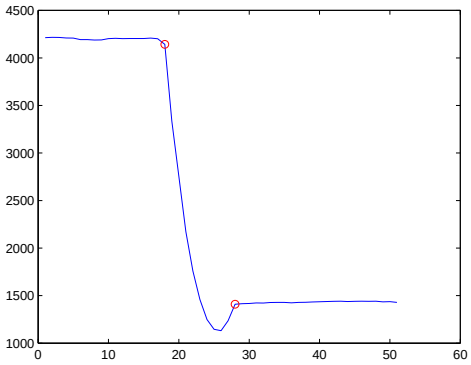


Figure 3.8: A plot showing the variance of intensity values across a dissolve transition. The start and end points of the transition are marked with red circles.

comprises the frame-to-frame colour histogram difference, the change in low-frequency audio energy, and differences in colour histogram, texture, and edge distribution evaluated over a 10-frame interval.

3.7 Transition Classification

As described above, transitions in video can be abrupt (cuts), gradual (fades and wipes), or intra-shot (flash photography, camera motion).The modelling and clustering approaches described above detect each transition type separately, and further classification is not necessary. However threshold-based systems do require further analysis to determine what kind of transition has occurred. Considerable research effort has been expended on examining what features in the local region of the difference trace, or in the frames in the region of the suspected transition, can be used for transition classification. Some of these features are examined in this section.

3.7.1 Fade and Dissolve Transitions

The characteristics of dissolve transitions are most easily considered with reference to the dissolve generation process. In a dissolve, two shots are overlapped and mixed with a dissolve parameter α which increases monotonically from 0 to 1 over the duration of the dissolve. Each frame within the dissolve transitions is then generated by

$$f_t(\mathbf{x}) = (1-\alpha)S_t^1(\mathbf{x}) + \alpha S_t^2(\mathbf{x}) \tag{3.27}$$

where S_t^1 and S_t^2 are frames from the overlapping shots at time t. In a fade in or fade out, S^1 or S^2 will be 0 everywhere. Consideration of 3.27 informs most dissolve detection approaches.

One of the earliest recognised characteristics of dissolves arising from this model is that under most circumstances, the variance of intensity values for frames in the dissolve region will follow

a downward parabolic curve. To see why, let σ_1 and σ_2 be the variances of intensity within the connected shots (assuming here that these variances do not vary with time). The variance of intensity values within a frame in the dissolve region is then

$$\sigma_t = (1 - \alpha_t)^2 \sigma_1 + \alpha_t^2 \sigma_2 \tag{3.28}$$

and given that $0 \leq \alpha_t \leq 1$, σ_t will always be less then $\max(\sigma_1, \sigma_2)$, and in dissolves generally less than both σ_1 and σ_2. In the case of a fade in or a fade out, σ_1 or σ_2 will be 0, and so the intensity variance will follow a descending parabola that does not rise from the minimum (0). This characteristic was first remarked by Alattar in 1993 [8] and has been used in numerous systems since then [135,219]. Systems detecting frames with zero intensity variance as a starting point for fade detection have also been proposed [201,306]. Figure 3.8 shows an example of this behaviour.

Over the duration of a fade in or fade out, assuming that the effects of motion are negligible, the relative frame difference

$$\frac{f_{t+1}(\mathbf{x}) - f_t(\mathbf{x})}{f_{t+1}(\mathbf{x})} \tag{3.29}$$

should be a constant. This has been exploited by Corridoni and del Bimbo [68] and Hampapur *et al.* [133]. In both papers, it is suggested that a dissolve be modeled as a fade-out followed by a fade-in. This approach is only effective where the frame-to-frame change is dominated by the dissolve effect, rather than by any motion in the sequence, and as such it has not been widely employed since 1995.

The dissolve generation equation can be used as a direct model for the video in the dissolve transition. In this approach, the frames on either side of the suspected transition are used to predict intermediate frames, assuming a dissolve is taking place. Comparing the predicted frame against the observed frame gives an indication of whether a dissolve is actually occurring.

The earliest model-based system is the *double chromatic difference* (DCD) method, presented in 1997 by Yu *et al.* [340]. Here a single frame is synthesised from the frames at either end of the transition region and compared to all the intervening frames. If the difference values follow a bowl-shaped curve, the transition is accepted as a dissolve.

The disadvantages of this system are that only one synthesised frame is used for comparison, and that it is very sensitive to the initial localisation of potential transition locations. These shortcomings are recognised and addressed in the *analysis by synthesis* method, presented by Covell and Ahmed in 2002 [70]. In this approach, an entire dissolve is synthesised between the endpoints of the suspected transition, and compared frame-by-frame to the observed video. An advantage of this method is that the region being analysed need not correspond exactly to the range of the dissolve; partial dissolves can be recognised and merged. In consequence, this method is much less sensitive to the initial transition detection approach, and in fact almost the entire video is analysed for dissolve characteristics. Good recall and precision are reported, and the method is computationally efficient. However, dissolves containing strong camera motion

cause difficulties, and the authors do not discuss how local motion in the dissolve region affects performance.

An interesting variant on the use of synthesis for dissolve analysis was presented by Lienhart in 2001 [203]. Here it is recognised that one drawback to applying machine learning techniques to dissolve detection is the tedium involved in labelling examples for training. This is overcome here by deploying a dissolve synthesiser, which can generate an effectively infinite number of dissolves from a video database. These generated dissolves are used to train a neural-network classifier, where the features are the contrast strength of the frames and a 24-bin YUV histogram. A post-processing step applies global motion estimation to filter out transitions due to camera pans and tilts. The reported performance is a detection rate of 75% and 16% false alarm rate. The effects of local motion in the dissolve region are not discussed.

A recent model-based approach is described by Denman and Kokaram in [80], in which a maximum-likelihood method is used to estimate the values of the mix parameter α in a transition region. If the estimated values follow the pattern expected for a linear mix between two shots, the transition is declared a dissolve. This approach is one of very few taking explicit account of local motion in the dissolve region. A full description of the method appears in the next chapter.

Huang and Liao have shown that a difference trace made up of dissimilarity values between a fixed frame and each following frame has different characteristics depending on whether the fixed frame is at the start of a dissolve or not [155]. Let $\delta'(t)$ be the 'rooted' difference trace, given by

$$\delta'(t) = \delta(T, T+t) \tag{3.30}$$

where T is the index of the frame under consideration. If f_T is the first frame of a dissolve, $\delta'(t)$ will increase linearly over the duration of the dissolve, and will subsequently be approximately constant. If f_T is not at the start of a dissolve, but rather in the middle of a contiguous shot, $\delta'(t)$ will increase exponentially up to a limit value. This characteristic is found in dissolves from a wide range of sources, and the reported performance on gradual transition is 93% recall, 78% precision.

3.7.2 Wipe Transitions

A wipe transition is a *spatial* mix between two sources, as opposed to the chromatic mix of a fade or dissolve transition. In other words, in any frame in the transition region, some fraction p_1 of pixels will take their values from the first shot, and the remaining portion, $1 - p_1$, from the second. p_1 declines from 1 to 0 over the duration of the transition. The spatial organisation of the pixel sites varies with the style of wipe transition. For example, the second shot may be 'revealed' by a sliding boundary; it may slide in, appearing to displace the first shot; or pixel sites may be switched from the first shot to the second according to a random schedule. Wipes very rarely occur in feature film material, but are quite frequently introduced in broadcast sports footage and home video.

As the wipe progresses, the image statistics will change smoothly from the values for the first shot to those for the second. For example, in video where a wipe transition connects two shots over frames $K \ldots L$, the mean intensity value at frame t will be governed by

$$m(t) = \begin{cases} m_1 & \text{if } t < K \\ \frac{1}{N}\left(p_1(n)Nm_1 + (1 - p_1(n))Nm_2\right) & \text{if } K \leq t \leq L \\ m_2 & \text{if } t > K \end{cases} \tag{3.31}$$

where m_1 and m_2 are the the means of the two shots (assumed constant), and $p_1(n)$ is the fraction of sites taking their value from the first shot at frame n. A similar equation holds for the variance of intensity values $\sigma(t)$ across the wipe transition.

The transition values will follow the same curve shape as $p_1(t)$. For example, in a random wipe the number of pixels from each shot varies linearly, whereas for an iris wipe (in which the region of the second shot is an expanding rectangle centered in the frame) $p_1(t)$ follows a quadratic curve. In either case, the transition can be detected by consideration of the derivatives of $m(t)$: the transition will be delimited by spikes in the second derivative, and the first derivative will have a non-zero average value over the transition region. This is the basis of one of the earliest algorithms specifically targeting wipes, presented in 1998 by Alattar [9]. Note that equation 3.31 assumes that the statistics of the pixels from each shot in the wipe will be the same as the statistics of the entire frames from each shot. This is more likely to hold true for wipes where the pixel sites are switched to the second shot in a spatially uniform schedule, such as a random wipe. For other kinds of wipe, the pixel sites from the second shot will all be from some small sub-area in the frame, and so their statistics are less likely to be the same as those of the entire frame in that shot.

A second system for wipe detection based on the center of mass of a frame difference image was proposed by Yu and Wolf in 1998 [341]. This approach is less versatile than that of Alattar, in that only 'sliding' wipes, where a moving boundary reveals the second shot, can be detected. The paper is very short on implementation details.

3.7.3 Transition classification from spatio-temporal images

Each of the various transition types introduces a discontinuity on some scale in the spatio-temporal images described above. Cuts result in a hard vertical edge, dissolves in a soft vertical edge, and some kinds of wipe result in a diagonal edge. This property has been exploited for transition detection and classification by a number of researchers. Kim *et al.*use detection of lines and curves to locate wipes [171]. Ngo *et al.*have published numerous systems targeting spatio-temporal images. One such system applies directed derivative-of-Gaussian and Gabor filters to characterise the local coherency of the spatio-temporal image; low local coherency indicates a discontinuity in the image and hence a transition [237, 238]. An adaptive threshold on local coherency is used to identify cuts; for wipes the Hough transform is applied. They have also explored the use of tensor histograms [239, 240]

The more recent approach by Bezerra [26] exploits the *longest common subsequence* of successive video slices to detect shot transitions.

3.8 Statistical Hypothesis Testing

The various aspects of transition detection and classification described above include methods for detecting outliers in the difference trace, incorporating intuitions regarding shot length, and algorithms for examing video data in the region of a suspected transition to determine whether a transition is present and what class of transition it is. It is desirable that a shot change detection algorithm incorporating these aspects should avoid ad-hoc decision processes and arbitrarily chosen threshold parameters. A Bayesian approach, also called statistical hypothesis testing, enables combination of these features in a rigorous way and also removes the need for threshold selection.

For any two frames t and t', two hypotheses are under consideration:

- $\mathcal{H}_0$: both frames belong to the same shot (event $\bar{B}$)
- $\mathcal{H}_1$: each frame belongs to a different shot (event B)

The optimal decision rule is then that a transition is declared if

$$\left.\frac{p(\delta|B)}{p(\delta|\bar{B})}\right|_{\delta=\delta(t,t')} > \frac{1-P_{t'}(B)}{P_{t'}(B)} \tag{3.32}$$

$P_{t'}(B)$ can be decomposed into an *a priori* probability $P^{\alpha}_{t'}(B)$ depending only on the number of frames since the last declared shot boundary, and a conditional probability $P_{t'}(B|\phi(t'))$ taking into account certain features extracted from the video in the region of frame t' [135].

In these systems, the likelihoods $p(\delta|B)$ and $p(\delta|\bar{B})$ are generally determined via fitting parametric distributions to empirically obtained data.

One of the first presentations of this Bayesian approach is due to Vasconcelos and Lippman [312, 313]. Here the *a priori* information incorporated is a distribution over shot lengths. The Poisson, Erlang, and Weibull distributions are considered, where the parameters for each distribution are obtained via a maximum-likelihood fit to empirically obtained shot-length data. The Weibull assumption is shown to yield the best performance, resulting in a 20% reduction in the error rate versus the optimal fixed threshold. The approach is conceptualised as an adaptive threshold method in which the dissimilarity threshold is very high immediately after a transition, and decays with distance from the last shot boundary. This behaviour is illustrated in figure 3.9. The bin-to-bin colour histogram distance is used as the dissimilarity metric.

In 2002, Hanjalic developed this approach further [135]. A more sophisticated similarity measure based on block matching is used here, for improved resilience to the effects of motion. Frame similarity is computed for adjacent frames and also over a 22-frame interval, and the intra-shot distribution $p(\delta|\bar{B})$ is shown to be essentially the same at both scales.

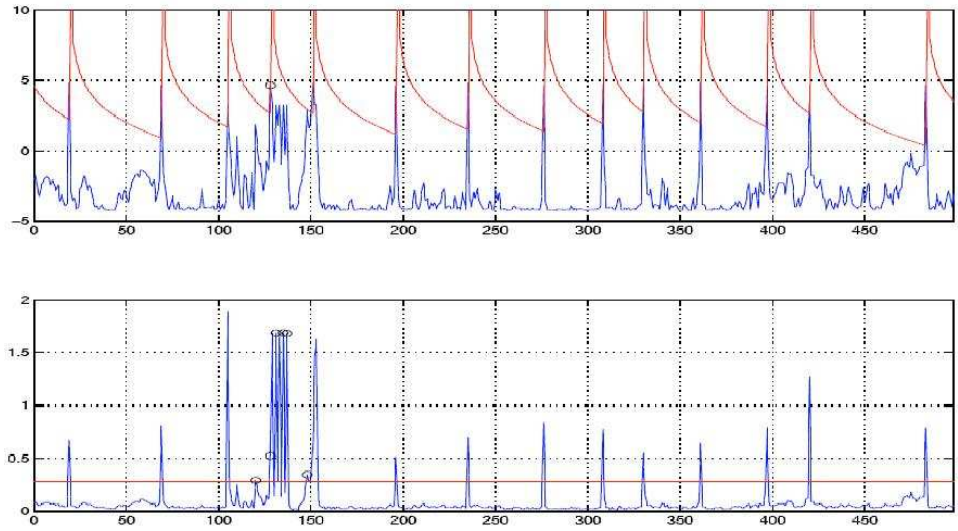


Figure 3.9: The influence of the Weibull prior on shot duration proposed by Vasconcelos. The blue line shows the difference values, and the red line shows the threshold value. In the lower graph, the optimal fixed threshold is shown; the upper graph shows the influence of the prior. Cut transitions are marked with circles. Reproduced from [313].

The principal contribution of [135] is the introduction of the data-conditional probability $P_{t'}(B|\phi(t'))$. This function incorporates examination of data in the region of a suspected transition, seeking the characteristics corresponding to various transition types. For example, for cuts, the conditional probability is

$$\phi_c(k) = \begin{cases} 100 \cdot \frac{\delta(t-1,t)-z_{sm}}{\delta(t-1,t)}\%, \\ \quad \text{if } \delta(t-1,t) = \max\left(\{\delta(t-1,t) : t \in \left(f = \frac{N-1}{2} \ldots t + \frac{N-1}{2} - 1\right)\}\right) \\ 0 \qquad\qquad\qquad\qquad \text{otherwise} \end{cases} \tag{3.33}$$

where δ_{sm} is the second largest value δ in the neighbourhood of t, and N is the size of this neighbourhood. This encodes the intuition that the dissimilarity across a cut should be much larger than nearby dissimilarity values. For dissolve transitions, two data-dependent functions are applied. Firstly, over a dissolve transition, a triangular peak is expected in the dissimilarity trace at the coarse scale $\delta(t-22,t)$. A function $\phi_{d1}(t)$ is proposed to detect such peaks, taking the value 1 if characteristics indicative of a triangular shape centered on k are found, and 0 otherwise. Secondly, as described above, the variance of intensity values for frames across a dissolve ideally follows a downward-parabolic pattern [8, 219]. A function $\phi_{d2}(t)$ is introduced quantifying the relative change in intensity variance between the middle of the window and the edges to detect this behaviour.

Boccignone *et al.* [35] use a similar analysis of dissimilarity traces to inform probabilistic detection of cuts and fades.

3.9 Directions in Shot Change Detection

At this stage, shot change detection is a very mature area of research and impressive results have been reported on a wide variety of footage. The broad features of a cut detection system are unlikely to evolve significantly in future research. Frame-to-frame similarity can be measured quite reliably, and incorporated into the Bayesian decision framework to allow for prior intuitions. It is likely that future cut detection work will be increasingly application-specific, and as such undertaken by industrial researchers. On the other hand, considerable improvement is desirable in detection of other transitions, such as dissolves and wipes.

The chief difficulty in comparing shot change detection systems is the lack of a common corpus of video material to form a basis for assessment. The activities of the TRECVID conference [275] are addressing this concern. Each system submitted to this conference is evaluated against a large supplied test set of commercial video for which the ground truth data is available. This makes comparison of different systems considerably easier. At present, the conference corpus is only available to participants, and participation is limited to some eighty institutions per year. However, as the conference grows it is expected to become the principal focus of research in shot change detection.

4

New Approaches to Shot Change Detection[1]

In the previous chapter, a comprehensive review of shot change detection techniques was presented. In this chapter, two new contributions to the shot change detection problem are presented. The first is a video analysis technique for detecting dissolve transitions based on direct modelling of the dissolve process. The second is a refinement to the transition detection technique described by Hanjalic [135], which improves performance on challenging video sequences.

4.1 Cut Detection

The Bayesian approaches to shot change detection developed by Vasconcelos [313] and Hanjalic [135] are close to the state of the art. The Bayesian formalism enables combining sophisticated frame similarity measures with *a-priori* intuitions concerning shot length and feature analysis from frames in a suspected transition region. The approaches described by these authors can still fail for very difficult sequences. In this section, a very difficult video is analysed and a number of refinements for cut detection are proposed on the basis of this analysis.

4.1.1 Frame Similarity Measure

As in the previous chapter, the similarity measure between frames t and t' is denoted $\delta(t, t')$. For a fixed interval n, $\delta(t, t-n)$ is denoted $\delta_n(t)$. The block matching frame similarity measure

[1] An early version of the work described here was published as "A Multiscale Approach to Shot Change Detection" by Hugh Denman and Anil Kokaram, in *Proceedings of the Irish Machine Vision and Image Processing Conference (IMVIP '04)*, pages 19–25, Dublin, September 2004

used in [135] is also used in this work. Frame I_t is divided into B non-overlapping blocks $b_t(i)$. For each block, the most similar block in frame t' is found. The block similarity measure is the absolute difference in the average Y, U, and V values of each block:

$$D(b_i(t), b_j(t')) = |\bar{Y}_{t,i} - \bar{Y}_{t',j}| + |\bar{U}_{t,i} - \bar{U}_{t',j}| + |\bar{V}_{t,i} - \bar{V}_{t',j}| \tag{4.1}$$

where $\bar{Y}_{t,i}$ is the average value of Y values in block i of frame t. The index of the best match, $\hat{b}_{t,t'}(i)$, is given by

$$\hat{b}_{t,t'}(i) = \arg \min_j \big(D(b_i(t), b_j(t'))\big) \tag{4.2}$$

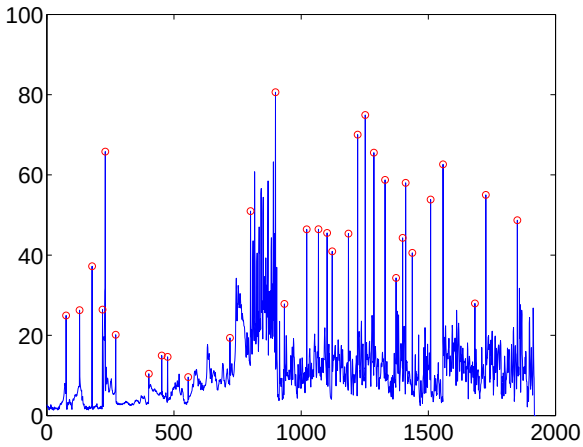
The frame difference measure is then the sum of most-similar-block differences over the entire frame:

$$\delta(t, t') = \frac{1}{B} \sum_{i=1}^{B} D(b_t(i), \hat{b}_{t,t'}(i)) \tag{4.3}$$

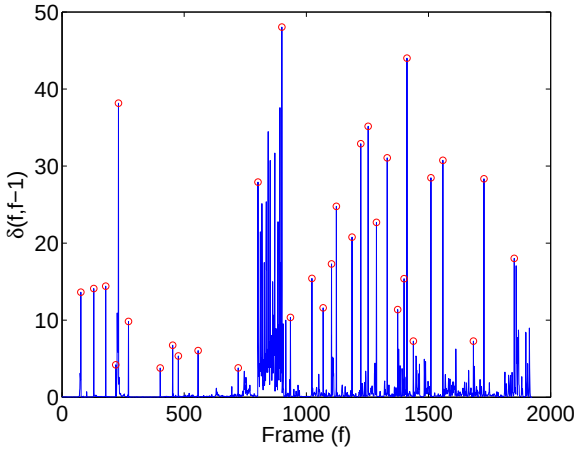
This measure is closely related to the Displaced Frame Difference (DFD) as formulated for local motion estimation.

A number of parameters need to be specified to implement this similarity measure, specifically the block size, the search range, and the search resolution. Appropriate values for these parameters depend on the resolution of the video and the interval between frames being computed, and also on the desired compromise between computational cost versus motion resilience. Small block sizes, wide search ranges, and fine search resolution improve the motion resilience but increase the computational cost, and can also reduce the value of $\delta(t, t')$ across a scene change, when blocks are matched across the transition. In this work, video is processed at a resolution of 180×144, using square blocks 4 pels on a side, with 1-pixel search resolution. As in [135], a search range of 1 block (4 pels) is used for computing $\delta_1(t)$.

As a case study of cut detection in a particularly challenging sequence, figure 4.1 shows values for the δ_1 measure calculated over 2000 frames from the film 'La Fille Sur Le Pont'. Two factors make detection of the shot transitions difficult in these signals. Firstly, some shots in this sequence are very short; for example, those occurring near frames 200 and 1400. These result in nearby peaks that should both be detected. Secondly, the sequence contains significant motion and lighting changes, in footage depicting splashing and swimming, and ambulance lights. These factors result in the high dissimilarity levels between frames 800 and 900, and towards the end of the sequence. Figure 4.2 shows how a low δ_1 value can be found across a cut transition. Figure 4.3 shows an example of footage where δ_1 values between some frames within a shot are as high as the values across a cut transition.



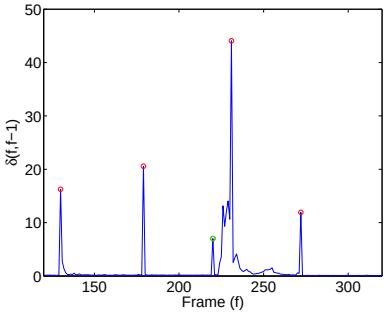
(a) Using frame-to-frame mean absolute difference of intensity values.



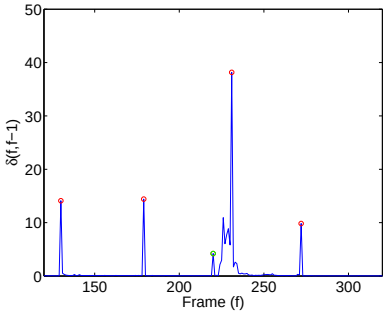
(b) Using the block-matching measure in (4.3).

Figure 4.1: Frame similarity traces for a challenging sequence from 'La Fille Sur Le Pont'. Cut transitions are shown with red circles.

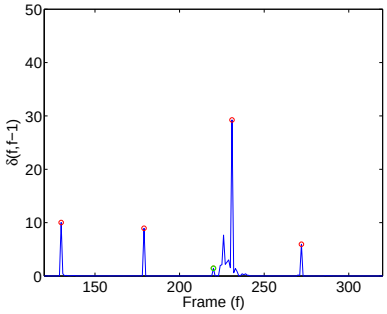
(a) Four consecutive frames spanning a cut in 'La Fille Sur Le Pont'



(b) $\delta_1(t)$ calculated with a blocksize of 4 pels.



(c) $\delta_1(t)$ calculated with a blocksize of 8 pels.

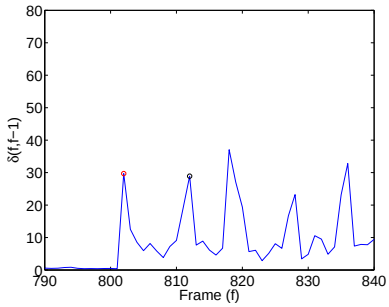


(d) $\delta_1(t)$ calculated with a blocksize of 16 pels.

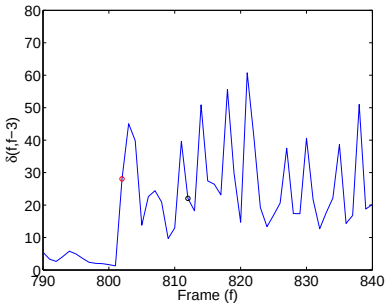
Figure 4.2: A cut for which the block-matching dissimilarity measure $\delta_1(t)$ gives a low value. In the dissimilarity signals, the cut illustrated is marked with a green circle, and other cuts in the sequence are marked with red circles. This is particularly pronounced in the signal produced using a block size of 16 pels.

(a) Two consecutive frames across a cut transition from 'La Fille Sur Le Pont'

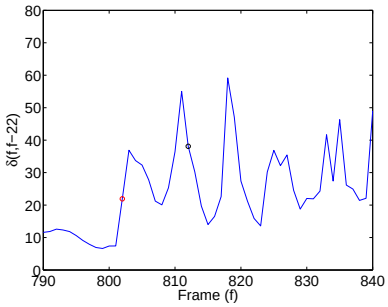
(b) Two consecutive frames within a shot from 'La Fille Sur Le Pont'



(c) $\delta_1(t)$ calculated with a blocksize of 4 pels.



(d) $\delta_1(t)$ calculated with a blocksize of 8 pels.



(e) $\delta_1(t)$ calculated with a blocksize of 16 pels.

Figure 4.3: This figure illustrates how difference values within a shot can be as large as those occurring over a cut. The cut shown in (a) is indicated in the difference traces by a red circle. The location of the pair of frames shown in (b) is indicated by a black circle; these frames are taken from a shot containing a close-up of flashing ambulance lights with considerable global motion. A number of the difference values within the shot are comparable in magnitude to that corresponding to the cut.

4.1.2 Dissimilarity Likelihood Distributions

The frame difference signal $\delta_1(t)$ is the starting point for detection of cuts in a video sequence. It is expected that in the main, peaks in this signal should correspond to cuts in the video. As described in the previous chapter, in the contemporary Bayesian approach, the optimal decision rule is to declare that a shot boundary exists between frames t and t' if

$$\left.\frac{P(\delta|B)}{P(\delta|\bar{B})}\right|_{\delta=\delta(t,t')} > \frac{1-P_t(B)}{P_t(B)} \tag{4.4}$$

Here B and $\bar{B}$ are the events 'cut has occurred' and 'cut has not occurred', and $P(\delta|B)$ and $P(\delta|\bar{B})$ are found using likelihood distributions determined with reference to ground-truth-marked video data. Figure 4.4 shows these distributions determined using 170742 frames of video. The blue curves show parametric probability distribution functions fit to this data; a γ distribution is used for the within-shot values, and a Gaussian distribution used for the inter-shot values. The method of moments is used to form an initial estimate of the parameters, followed by a number of iterations of a maximum-likelihood refinement procedure. Further details of this approach are given in chapter 6.

If these distribution functions are used directly, then for very low difference values δ, including $\delta = 0$, $P(\delta|B)$ will be greater than $P(\delta|\bar{B})$—in other words, low δ values will have a higher 'cut' likelihood than 'no cut' likelihood. There are two reasons for this. Firstly, the γ distribution for $p(\delta|\bar{B})$ will always assign a value of 0 to the likelihood of a δ value of 0, where the Gaussian $p(\delta|B)$ will be nonzero. Secondly, very low intra-shot δ values may not be encountered in great numbers in the training set, resulting in the estimated γ distribution taking a low value for these values—possibly lower than those given by the Gaussian $p(\delta|B)$. It is clear that very low δ values are more likely to be found within a shot than across a shot boundary, and so some manipulation of the distributions to avoid these effects is justified.

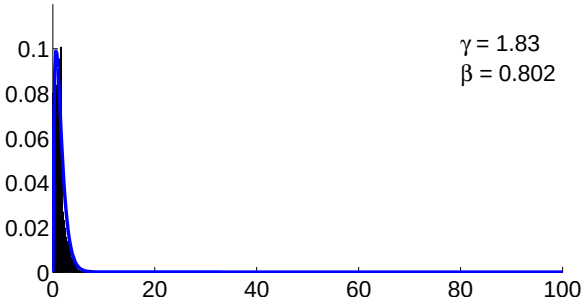
Firstly, before the parameters of the γ distribution for $p(\delta|\bar{B})$ are found, the histogram of intra-shot δ values is manipulated such that all counts to the left of the mode value, $\hat{\delta}_{\bar{B}}$, are set to be equal to the histogram value at the mode. Secondly, the mathematical forms of the distribution functions are modified, such that

$$p(\delta|\bar{B}) = \begin{cases} 1 & \text{if } \delta = 0 \\ \left(\frac{\delta}{\beta}\right)^{\gamma-1} \exp\left(-\frac{\frac{\delta}{\beta}}{\beta\Gamma(\gamma)}\right) & \text{otherwise} \end{cases} \tag{4.5}$$

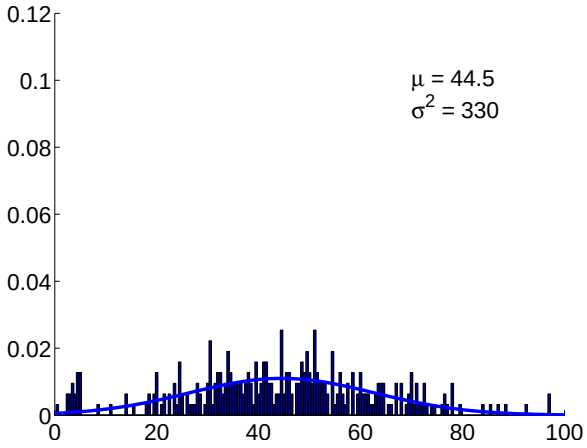
$$p(\delta|B) = \begin{cases} 10^-7 & \text{if } \delta =< \hat{\delta}_B \\ \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(\frac{-(x-\mu)^2}{2\sigma^2}\right) & \text{otherwise} \end{cases} \tag{4.6}$$

Here $\Gamma(x)$ is the Gamma function

$$\Gamma(x) = \int_0^\infty t^{(x-1)} e^{-t} dt \tag{4.7}$$



(a) Distribution of within-shot values of $\delta_1(t)$. The estimated γ distribution is shown. This is an approximation of $p(\delta|\bar{B})$. N=170742.



(b) Distribution of values of $\delta_1(t)$ across shot boundaries (cuts). The estimated normal distribution is shown. This is an approximation of $p(\delta|B)$. N = 315.

Figure 4.4

The parameters of the Gaussian distribution can be estimated directly from the data: μ and and σ^2 are the sample mean and variance for values across a cut transition. The parameters of the Gamma distribution, γ and β, are estimated using the method of moments [72]:

$$\gamma = \left(\frac{\mu'}{\sigma'}\right)^2, \ \beta = \frac{\sigma'^2}{\mu'} \tag{4.8}$$

where μ' and σ' are the sample mean and standard deviation for values within a shot.

These modifications ensure that $p(\delta|\bar{B}) < p(\delta|B)$ for δ values less than $\hat{\delta}_{\bar{B}}$.

4.1.3 Peak Analysis in the δ Signal

$P_t(B)$ is composed of two elements. The *a priori* distribution $P_t^{\alpha}(B)$ encodes prior knowledge concerning typical shot lengths, and depends only on the number of frames since the last declared shot boundary. The data-conditional probability $P_t(B|\psi(t))$ allows additional analysis, supplementary to the value of $\delta_1(t)$, to be incorporated via an auxiliary function $\psi(t)$. This can include analysis of the shape of the δ signal in the region of t', and / or information obtained by further examination of the frames in this region.

Peak Shape Function: The peak shape auxiliary function $\psi(t)$ is intended to detect isolated peaks within some temporal window. A cut should not be declared at a point that is not a local maximum in $\delta_1(t)$, so $\psi(t)$ is zero in this case. If $\delta_1(t)$ is at a maximum within the local window, $\psi(t)$ is some measure of the magnitude of the peak relative to the values in the window. In [135],

these considerations lead to the function

$$\psi_{HC}(t) = \begin{cases} \frac{\delta_1(t) - z_{sm}}{\delta_1(t)} & \text{if } \delta_1(t) = \max_{t' \in (t-10, t+10)} \delta_1(t') \\ 0 & \text{otherwise} \end{cases} \tag{4.9}$$

where z_{sm} is the second largest value in the the local window.

In this work, a similar $\psi(t)$ function is used to quantify the peakiness of $\delta_1(t)$ at a given point, but a slightly different approach is adopted.

Firstly, two asymmetric neighbourhood regions are used, rather than a symmetric window centered on the point in question. This is motivated by consideration of material with very high inter-shot dissimilarity, as shown earlier, where the value of $\delta_1(t)$ at a shot cut is of comparable magnitude to some inter-shot values in an adjacent shot. Provided the δ_1 value at the cut is larger than the values typical of the other shot, use of an asymmetric window is helpful in detecting cuts in such instances.

Secondly, the $\delta_1(t)$ is evaluated with reference to the distribution of local values, rather than using the second largest value in a window alone. Using the local distributions in this way means that rather than modelling difference values entirely in terms of within shot / between shot, to some extent the difference value process for each shot is modelled. If the $\delta_1(t)$ value is atypical of either of the local distributions, a shot change is declared.

The two considerations described above lead to an auxiliary function for cuts of the form

$$\psi_1(t) = \begin{cases} \max\left(D_L(t), D_R(t)\right) & \text{if } \delta_1(t) = \max_{t' \in (t-3, t+3)} \delta_1(t') \\ 0 & \text{otherwise} \end{cases} \tag{4.10}$$

where D_L is a measure of the magnitude of $\delta_1(t)$ compared to the N preceding values

$$D_L(t) = \min(100, \tfrac{\delta_1(t) - \mu_L}{\sigma_L}) \tag{4.11}$$

$$\mu_L = \tfrac{1}{N} \textstyle\sum_{t'=t-N}^{t-1} \delta(t', t'-1) \tag{4.12}$$

$$\sigma_L = \tfrac{1}{N-1} \textstyle\sum_{t'=t-N}^{t-1} \left(\delta(t', t'-1) - \mu_L\right)^2 \tag{4.13}$$

and D_R is a similar measure based on the succeeding values. The window size N used here is 22 frames. Because the use of this window implicitly incorporates an idea of minimum shot length, a separate shot-duration prior is not incorporated in this approach.

To compare this asymmetric peak shape function to that proposed by Hanjalic, each measure was applied to the difference values shown in figure 4.1. The performance of each was then characterised by their Receiver Operating Characteristic (ROC) curves. For the measure proposed by Hanjalic, the area under the ROC was 0.912, and optimal performance, in the sense of the maximum mean of precision and recall, was 75% recall and 100% precision, obtained with a threshold of 42. The $\psi_1(t)$ measure described above had an area under the ROC of 0.987, with 100% recall and and 91.4% precision obtained using a threshold value of 7.6. The new $\psi_1(t)$, then, offers greater separation of peaks in $\delta_1(t)$ due to cuts from other values.

Multiresolution Peak Detection: A second auxiliary function for cuts is introduced using a multi-resolution approach. Here information from the $\delta_1(t)$ signal is combined with frame comparison at a greater interval, i.e. the $\delta(t, t-n)$ signal with $n > 1$. This facilitates distinguishing peaks in $\delta_1(t)$ due to a cuts from those due to some sudden extreme effect such a flashing light. Where a cut has occurred at some time T, all frames after T are dissimilar to all frames before T, and so there will be a plateau in $\delta(t, t-n)$ from T to $T+n$. In the case of a peak due to a sudden event at time T, however, frame $T+1$ should be similar to frame $T+1-n$—provided the event has a duration less than n. Thus if the peak at T is due to a sudden event, the values of $\delta(t, t-n)$ should decline after time T. These considerations lead to a second auxiliary function for cut detection

$$\begin{aligned} \psi_2(t) = &\quad \min\left(1, 1.2 - \tfrac{\delta_n(t)-\delta_n(t+1)}{\delta_n(t)}\right) \\ \times &\quad \min\left(1, 1.2 - \tfrac{\delta_n(t-1)-\delta_n(t)}{\delta_n(t-1)}\right) \end{aligned} \tag{4.14}$$

The first term in the product penalises regions where the dissimilarity at time t declines by more than 20% at time $t+1$. The second term penalises locations where the dissimilarity value has dropped by over 20% of the previous value. This term is introduced as the δ_n signal should be rising, rather than falling, at a cut transition. This additional auxiliary function is combined with ψ_t to give the final auxiliary function for cuts $\psi_c(t) = \psi_1(t)\psi_2(t)$. Although introducing the ψ_2 has no effect on detection performance using the data in figure 4.1, it can be seen to attenuate those peaks introduced by extreme factors. Figure 4.5 shows this effect for a subset of the frames from this sequence.

4.1.4 Mapping The Auxiliary Functions to a Probability

The auxiliary function for cuts must be mapped to a conditional probability $P(B|\psi_c(t))$. As in [135], a soft mapping based on the erf function is employed for this purpose.

$$P(B|\psi_c(t)) = \frac{1}{2}\left(1 + \mathrm{erf}\left(\frac{\psi_c(t) - \mu_e}{\sigma_e}\right)\right) \tag{4.15}$$

where

$$\mathrm{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2} dt \tag{4.16}$$

The values of $\psi_c(t)$ were found for 170742 frames of video, for which the ground truth data was previously prepared. Estimates of $P(B|\psi_c(t))$ for various values of $\psi_c(t)$ were determined from this data, shown by the blue stems in figure 4.6 (a). Inspection of these values suggested $\mu_e = 0.5$ and $\sigma_e = 0.001$ as appropriate initial parameters. A Nelder-Mead minimisation strategy [235] was then used to refine the parameters, based on a weighted square error function. It was found that estimation of the optimal parameters was unstable, due to the very uneven distribution of the weights. To allow for this, the minimisation procedure was performed 100 times, using randomly perturbed initial parameters, and the best fit of all iterations accepted. The best fit curve had parameters $\mu_e = 0.283$, $\sigma_e = 0.118$; this curve is shown in green in figure 4.6 (a).

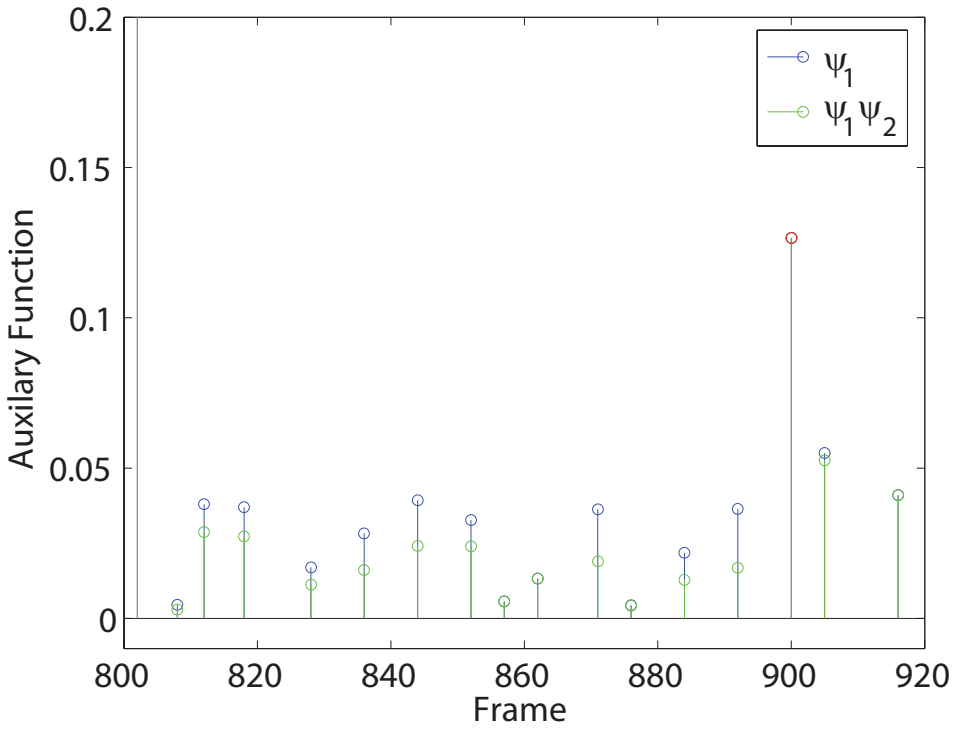
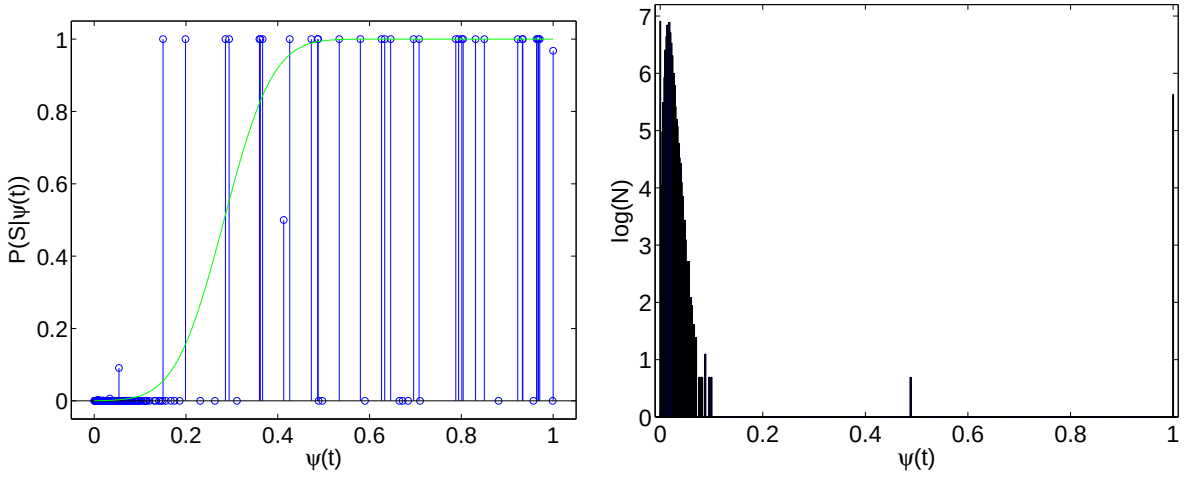


Figure 4.5: Blue stems show the $\psi_1(t)$ function for cut detection; green stems show $\psi_1(t)\psi_2(t)$, where the multi-resolution peak shape function has been introduced. The $\psi_2(t)$ function attenuates the peak shape function at extreme events, but the values at cuts—indicated by red circles—are not affected.



(a) The blue stems show individual values of $P(S|\psi(t)$. The green line shows the fitted curve.

(b) The weights for each of the values of $P(S|\psi(t)$ shown in blue at left.

Figure 4.6: Determining $P(S|\psi(t)$ based on 169,000 training frames.

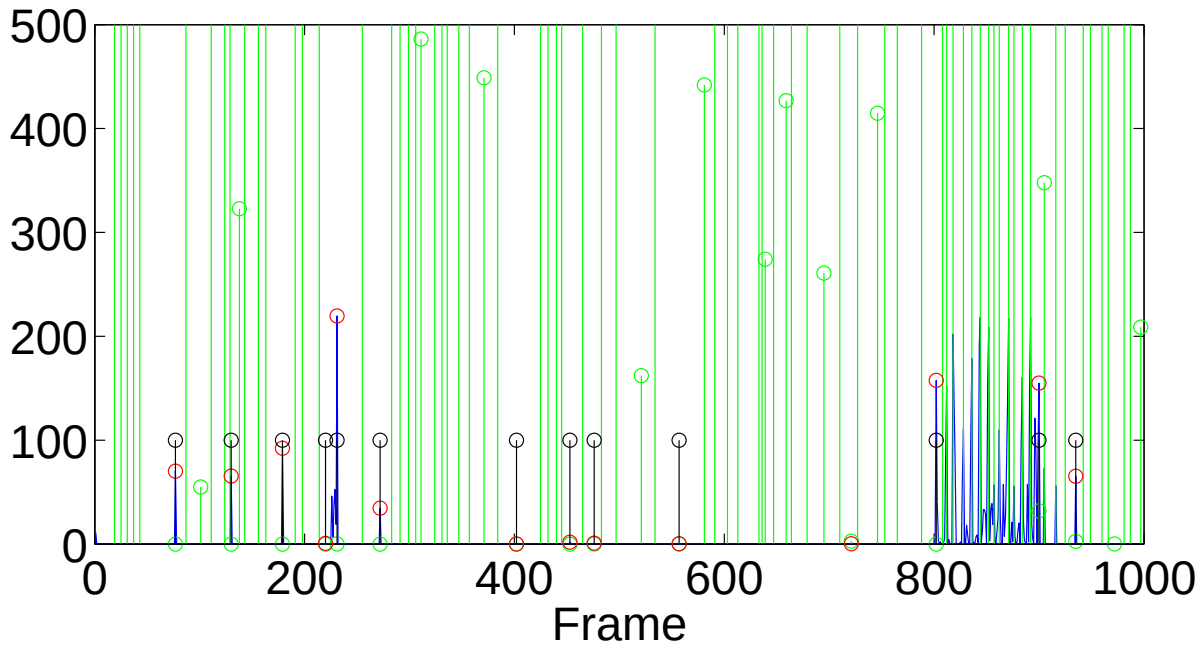


Figure 4.7: Shot change detection in a challenging sequence: 1000 frames from 'La Fille Sur Le Pont'. The blue trace shows $\frac{p(\delta_1(t)|B)}{p(\delta_1(t)|\bar{B})}$; red circles on the blue trace indicate the locations of cuts in the sequence. The green stems show $\frac{1-\psi_c(t)}{\psi_c(t)}$. Black stems have been placed where a cut transition is declared. In this section, one cut is missed, at frame 720.

4.1.5 Performance Evaluation

At this stage, all the elements required to employ the decision rule for cuts

$$\left.\frac{p(\delta|B)}{p(\delta|\bar{B})}\right|_{\delta=\delta_1(t)} > \frac{1-P(B|\psi_c(t))}{P(B|\psi_c(t))} \tag{4.17}$$

have been found. Performance on the difficult sequence shown in 4.1 is very good, with 93.75% recall and 100% precision. An example of the output is shown in 4.7. The measure was also tested on 12 videos from the TrecVid02 collection, totaling 288702 frames. Here 93.7% recall was achieved, with 84% precision. The precision here is reduced, however, by the presence of numerous fast dissolve transitions in the corpus, which are detected as cuts. As these are instances of shot transitions, they can be considered acceptable detections. When this provision is made, precision rises to 90%.

4.2 Frame Similarity for Dissolve Detection

Dissolve transition detection is based on three components: the value of a dissimilarity signal, the shape of the peak in the dissimilarity signal, and examination of the frames in the region

of the peak. These are combined using the same Bayesian decision framework applied to cut detection above. The decision rule is

$$\left.\frac{p(\delta|B)}{p(\delta|\bar{B})}\right|_{\delta=\delta_{22}(t)} > \frac{1-P(B|\psi_d(t))}{P(B|\psi_d(t))} \tag{4.18}$$

where δ_{22} is the dissimilarity signal, described below, and $\psi_d(t)$ is a data-conditional function combining consideration of the peak shape with analysis of the frames in the peak region.

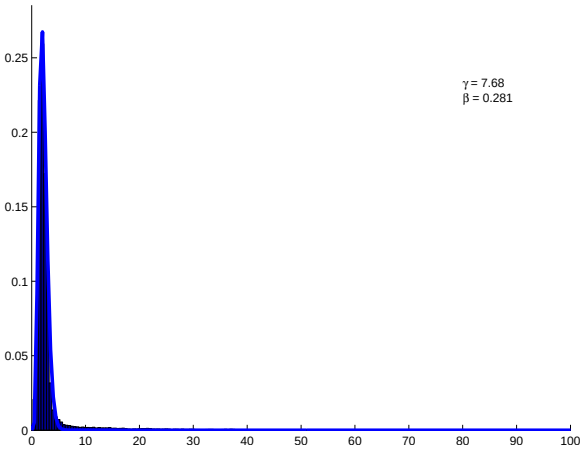
In this section, the δ_{22} signal is described, and an method for peak detection in this signal is introduced. It is shown that examination of peak shape alone is not sufficient to distinguish dissolves from other transitions or the effects of motion. Thus examination of the video frames in peak regions is necessary. In the preceding chapter, two previously proposed methods for detecting dissolves by examination of frame data were described. Firstly, the variance of intensity values in each frame is expected to follow a downward parabola over the dissolve region. Secondly, the median pixel-to-pixel intensity difference is expected to be a non-zero constant over a dissolve region. In the section following this one, a new model-based approach to detecting whether a sequence of frames contains a dissolve is introduced and compared to these previous methods. The final section of this chapter illustrates how the model-based approach is incorporated into a dissolve detection system.

In this section and those following, the characteristics and parameters of various components of a dissolve-detection system are presented. These parameters are computed with reference to a training set of 288 dissolve transitions (including fades). These transitions were taken from a training corpus of 170742 frames, consisting of a subset of the TrecVid02 [275] test data.

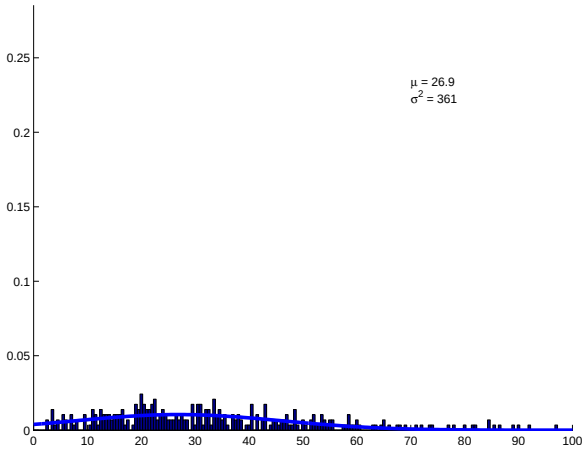
4.2.1 Similarity Measure and Likelihood Distributions

The block-matching approach to frame similarity is used for dissolve detection as well as cut detection here. Because the difference between successive frames over a dissolve is generally much lower than that typical of an abrupt cut transition, the frame dissimilarity must be computed over some interval of several frames. Comparing frames over a larger interval increases the effects of pans, zooms, and motion in the video footage, and so the search area for block matching must be increased. However, this larger search window can reduce the dissimilarity values found across shot transitions, as the chance of randomly finding a good match for a block across a shot transition are increased. In this work, as in that described in [135], an interval of 22 frames is used, applied to videos of dimension 160×120 pixels. The block size is 4 pixels on a side, with a search region width of 40 pixels. The resulting signal is denoted $\delta_{22}(n) = \delta(n, n-22)$, which is the starting point for dissolve detection.

Figure 4.8 (a) shows the distribution of δ_{22} values within a shot, and the γ distribution used to approximate $p(\delta_{22}|\bar{B})$. The γ parameter for this distribution is found to be 7.68, considerably higher than the value 1.83 found for $p(\delta_1|\bar{B})$. This is due to the amplification of the effects of motion by the 22-frame interval. Figure 4.8 (b) shows the distribution of values of δ_{22}



(a) Distribution of within-shot values of $\delta(t, t-22)$. The estimated γ distribution is shown. This is an approximation of $p(\delta|\bar{B})$. N=170742.



(b) Distribution of values of $\delta(t, t-22)$ within dissolves. The estimated normal distribution is shown. This is an approximation of $p(\delta|B)$. N = 288.

Figure 4.8

corresponding to dissolve transitions; the largest value of δ_{22} within each transition region was taken to build this distribution.

As was done in the case of $p(\delta_1|\bar{B})$ and $p(\delta_1|B)$, the histogram data and distributions are modified to ensure that for δ values less than the mode value, $\hat{\delta}_B$, $p(\delta|\bar{B}) < p(\delta|B)$. Specifically, the histogram of intra-shot δ_{22} values is manipulated such that all counts to the left of $\hat{\delta}_B$ are set to be equal to the histogram value at the mode, and the mathematical forms of the distribution functions are modified such that

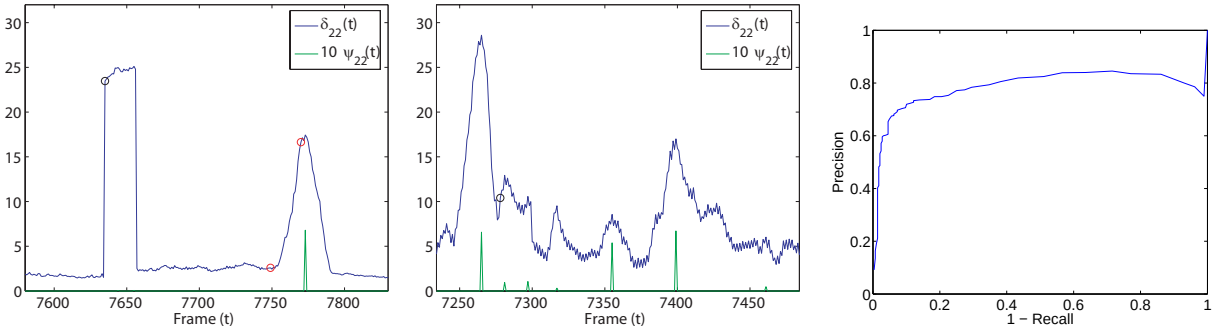
$$p(\delta_{22}|\bar{B}) = \begin{cases} 1 & \text{if } \delta_{22} = 0 \\ \left(\frac{\delta_{22}}{\beta}\right)^{\gamma-1} \exp\left(-\frac{\frac{\delta_{22}}{\beta}}{\beta\Gamma(\gamma)}\right) & \text{otherwise} \end{cases} \tag{4.19}$$

$$p(\delta_{22}|B) = \begin{cases} 10^-7 & \text{if } \delta_{22} =< \hat{\delta}_B \\ \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(\frac{-(x-\mu)^2}{2\sigma^2}\right) & \text{otherwise} \end{cases} \tag{4.20}$$

where γ, β, μ, and σ^2 are distribution parameters estimated from the data, and $\Gamma(x)$ is the Gamma function as before.

4.2.2 Peak Analysis in the δ_{22} Signal

Figure 4.9 (a) shows the two kinds of peak that are introduced in δ_{22} by cuts and dissolves. Cuts are visible as a sharply delineated plateau in the signal, while fades result in a smooth triangular peak. The specific shape of the peak—the width, height, and the slope of the sides—will be different for each transition, depending on the dissolve speed and the similarity of the connected



(a) The blue trace shows the δ_{22} signal; the peak detection function $\psi_{22}(t)$ is shown in green. The footage used contains a cut, marked with a black circle, and a dissolve transition, shown by the red circles.

(b) The δ_{22} signal and the peak detection function $\psi_{22}(t)$ over very visually active footage of violent waves at sea. A cut occurs in this sequence, marked with a black circle.

(c) ROC of peak detection using thresholds on $\psi_{22}(t)$. The area under the ROC is 75%.

Figure 4.9: Peak detection in δ_{22}.

shots. Similar smooth peaks may also be introduced in the signal by other kinds of gradual transition, such as camera pans, or in very active footage. Figure 4.9 (b) shows an example of peaks in δ_{22} that do not correspond to gradual transitions in the video. This confirms that analysis of the similarity signal is not sufficient in itself for dissolve detection; examination of the video frames in the regions corresponding to these peaks is also necessary.

4.2.2.1 Peak Detection:

Several criteria are combined to detect triangular peaks in the δ_{22} signal. These are expressed through the function $\psi_{22}(t)$

$$\psi_{22}(t) = \begin{cases} M_L(t)M_R(t)D_L(r)D_R(t)K_L(t)K_R(t)\max(P_L(t), P_R(t)) \\ \qquad \text{if } \delta_1(t) = \max_{t' \in (t-4, t+4)} \delta(t', t'-1) \\ 0 \qquad \text{otherwise} \end{cases} \tag{4.21}$$

taking values in the range $0 - 1$. The intention is that $\psi_{22}(t)$ should take a high value when there is a peak at t that matches the triangular shape expected at a dissolve. Each component of the function accounts for one of the peak shape criteria being applied. Each criterion is applied twice, to the neighbourhood on the left (indicated by the subscript L), and on the right (subscripted R), of the point under consideration. The neighbourhood size is fixed at $N = 22$ frames.

If there is a triangular peak at t in δ_{22}, the largest value in the neighbourhood to either side should be similar in value to the maximum at t. This is not the case for isolated spikes in $\delta_{22}(t)$, which can be introduced by extreme events such as a camera flash going off. $M_L(t)$ is a function

that ensures that $\psi_{22}(t)$ takes a low value at such isolated spikes. The peak at $\delta_{22}(t)$ is penalised if the maximum in the neighbourhood to the left, M_L, is less than $0.9\delta_{22}(t)$, according to

$$M_L(t) = \min\left(1, 1.1 - \frac{(\delta_{22}(t) - M_L)}{\delta_{22}(t)}\right) \tag{4.22}$$

For a triangular peak at t, the location of the largest value in each neighbourhood should be close to t. $D_L(t)$ penalises peaks in $\delta_{22}(t)$ where this is not the case, based on how far the maximum to the left is from t relative to the window size:

$$D_L(t) = \left(1 - \frac{(t - I_L)}{N}\right) \tag{4.23}$$

Here I_L is the index of the maximum value in the neighbourhood to the left. This is an additional penalty on isolated spikes in $\delta_{22}(t)$, especially where two such spikes occur in rapid succession.

$K_L(t)$ is designed to eliminate the plateaus that are introduced to δ_{22} due to cuts. This function penalises the peak at t if there are large discontinuities in the value of $\delta_{22}(t)$ in the neighbourhood to the left. Let M'_L be the largest absolute difference between successive values of δ_{22} in the neighbourhood to the left. Then

$$K_L(t) = \max\left(0, 1 - \frac{M'_L}{\delta_{22}(t)}\right) \tag{4.24}$$

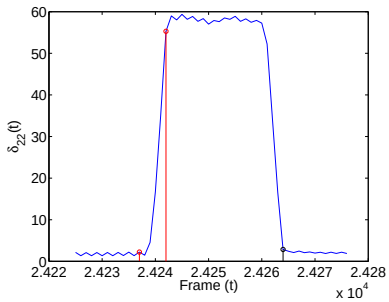
The last component, the function $P_L(t)$, gives an indication of the relative magnitude of the peak at $\delta_{22}(t)$, based on the difference between $\delta_{22}(t)$ and the minimum value in the neighborhood to the left, m_L, relative to $\delta_{22}(t)$. If this is less than 0.3, the maximum is considered 'not significant', and $K_L(t)$ is set to 0. If the relative difference is greater than 0.7, no penalty is applied to the peak at t.

$$P_L(t) = \min\left(1, \frac{\max\left(0, \frac{\delta_{22}(t) - m_L}{\delta_{22}(t)} - 0.3\right)}{0.4}\right) \tag{4.25}$$

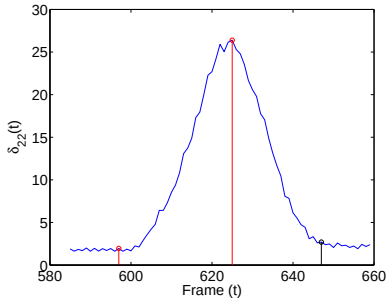
Performance Assessment: Figures 4.9 (a) and (b) show that the $\psi_{22}(t)$ function is effective in detecting rounded peaks while rejecting the plateaus introduced by cuts. The ROC curve for peak detection, using increasing thresholds on $\psi_{22}(t)$, is shown in figure 4.9 (c); the area under the ROC curve is 78%.

4.2.2.2 Peak Width Estimation

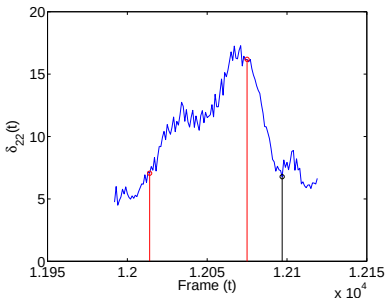
Most peaks due to dissolves in the δ_{22} signal have clearly visible start and end points. Where a peak does correspond to a dissolve, the dissolve transition starts at the start of the peak and ends 22 frames before the end of the peak. Estimating the start and end points of a peak is then useful in two ways. It gives an indication of which frames should be examined for dissolve characteristics, as described in the next section, and also means that where a dissolve is detected, an estimate can be given of its start and end frames.



(a) A 9-frame dissolve. Because the dissolve has less than 22 frames, frames from the shots on either side of the dissolve are compared. This results in a high value within the plateau.



(b) A 32-frame dissolve. Such medium-length dissolves produce rounded peaks rather than plateaus.



(c) A 65-frame dissolve. Motion effects can introduce noisy ascent regions in long dissolves.

Figure 4.10: Peaks corresponding to dissolves in $\delta_{22}(t)$. The red stems show the start and end frames of the dissolves. The black stem marks the frame 22 frames after the end of the dissolve.

Figure 4.10 illustrates some of the peak shapes typical of dissolves in δ_{22}. The correspondence between the peak extent and the frames in the dissolve is clear. Three methods for estimating the peak extent have been investigated, and these are now described. Once a peak has been identified, the values of d_{22} on either side are examined to find the end of the peak on that side. Each side of the maximum is treated identically; the problem is essentially one of changepoint detection in these windows. The window is designated $W(t)$.

Line fitting: The first approach adopted relies on fitting a line to the first N values in the window, and detecting where the fit between the line and the window values starts to exceed some threshold. Here N is set to 20, and an iteratively reweighted least squares scheme is used to estimate the line parameters, o, λ. The threshold E_1 is the maximum squared error over the first N frames of the window:

$$E_1 = \max\{(W(t) - (o + \lambda \cdot t))^2\} \tag{4.26}$$

A 'large error' signal, $L(t)$, is then found, according to

$$L(t) = \begin{cases} 1 & \text{if } (W(t) - (o + \lambda \cdot t))^2 > E_1 \\ 0 & \text{otherwise} \end{cases} \tag{4.27}$$

This signal is then filtered with a 5-tap median filter to produce $\hat{L}(t)$. The changepoint is then defined as

$$c_1 = \min\{t : \forall t' > t . \hat{L}(t) == 1\} \tag{4.28}$$

On-line Slope Change Detection: In this approach, the changepoint is declared at the first

Method	$\bar{E}_s$	$\sigma^2_{E_s}$	$\bar{E}_e$	$\sigma^2_{E_e}$
Line fit	0.48	60.385	0.13	29.5
Slope Change	1.05	61.61	1.02	55.94
Gaussian Separation	0.88	55.255	-0.36	23.1

Table 4.1: Peak start and end detection performance. $\bar{E}_s$ and $\sigma^2_{E_s}$ are the mean and variance of the error in estimating the peak start. $\bar{E}_e$ and $\sigma^2_{E_e}$ are the mean and variance of the error in estimating the peak end.

index where the window value stops descending.

$$c_2 = \min\{c : \text{mean}_{c' \in \{c, c+4\}}(W(c')) \leq \text{mean}_{c' \in \{c+5, c+9\}}(W(c'))\} \tag{4.29}$$

This is an on-line method, and as such is insensitive to the window size used.

Separation of Gaussians: Here it is assumed that the derivative of the window can be modelled as two sequential Gaussian distributions, one for the peak descent and one for the post-descent region. The changepoint is declared at c_3, where

$$c_3 = \underset{c \in \{12 \ldots |W|-8\}}{\arg\max} \frac{m_2(c) - m_1(c)}{\sigma_2(c)\sigma_1(c)} \tag{4.30}$$

where $m_1(c)$ and $\sigma_1(c)$ are the mean and variance of $W(t) - W(t-1)$ over $1 \ldots c$, and $m_2(c)$ and $\sigma_2(c)$ are the mean and variance of $W(t) - W(t-1)$ from $c+1$ to the end of the window. A separation point where the slope increases is desired, rather than one where the slope decreases. Therefore the absolute value of the distance is not taken. This is a global method, and is observed to give very good results when the window stretches over two distinct regions, the side of the peak and a flat or ascending following region. The method is prone to failure when values from an adjacent peak are included in the window.

Performance Assessment: The d_{22} signal for the videos in the training set was found, and the peak detection procedure $\psi_{22}(t)$ applied. Those peaks for which $\psi_{22}(t) > 0.1$, which also corresponded to a dissolve, were found. The start and end points of these peaks were estimated, and compared to the start and end points of the dissolve as indicated by the ground truth. Table 4.1 shows the results of this assessment. Detection of peak end-points is more reliable than detection of peak start points for all methods. This can be ascribed to the fact that for most peaks, the descent is smoother than the ascent. The Gaussian separation method is adopted for finding the peak start- and end-points.

4.3 Model-Based Dissolve Estimation

As described in the previous chapter, gradual transitions in video may be inter-shot transitions introduced by editing, such as dissolves or wipes, or they may be intra-shot transitions due to

camera motion or lighting changes. Such gradual transitions are typically detected by frame differencing at a coarse temporal scale, i.e. with an interval of 10 or 20 frame times between the frames being compared. Once a coarse-scale transition has been identified, the question of how to identify the kind of gradual transition arises. In this section a dissolve modeling approach is introduced. Assuming that the transition region does contain a dissolve, a dissolve parameter α is estimated for each frame. The values for each frame form an α-curve, and examination of this curve then informs classification of the video region as being a dissolve, or otherwise.

4.3.1 Dissolve model

A dissolve is made by mixing two shots chromatically, according to a mix parameter α. The values in each channel of the transition are generated by

$$I_t(\mathbf{x}) = (\alpha)I_t^1(\mathbf{x}) + (1-\alpha)I_t^2(\mathbf{x}) \tag{4.31}$$

where $\mathbf{x}$ is a pixel site, I_t^1 and I_t^2 are images from the two overlapping shots at time t, and I_t is the final composited image.

Given the images I_t^1, I_t^2, and I_t at a particular time t, the likelihood of a dissolve parameter value α is governed by

$$p(\alpha|I_t^1, I_t^2, I_t) \propto \exp(-\sum_{\mathbf{x}} \left[I_t(\mathbf{x}) - (\alpha I_t^1(\mathbf{x}) + (1-\alpha)I_t^2(\mathbf{x}))\right]^2) \tag{4.32}$$

However, in general the unmixed frames I_t^1, I_t^2 are not available. To facilitate analysis, the contents of the two shots over the dissolve region are approximated using two *template frames*, designated I_{T_0} and I_{T_1}. These frames are chosen from positions preceding and succeeding the transition region. The image predicted by this model for a given crossdissolve strength α is designated $I_{M(\alpha)}$, where

$$I_{M(\alpha)}(\mathbf{x}) = \alpha I_{T_0}(\mathbf{x}) + (1-\alpha)I_{T_1}(\mathbf{x}) \tag{4.33}$$

The likelihood of a given value of α can now be approximated using the agreement between the image predicted by the model and the observed image at time t:

$$p(\alpha|I_t) \propto \exp(-\sum_{\mathbf{x}} \left[I_t(\mathbf{x}) - I_{M(\alpha)}(\mathbf{x})\right]^2) \tag{4.34}$$

The maximum-likelihood value of alpha given I_{T_0}, I_{T_1}, I_t can be estimated by solving

$$\frac{d}{d\alpha} p(\alpha|I_t) = 0 \tag{4.35}$$

and it transpires that the optimal value is given by

$$\alpha_{opt} = \frac{\sum I_t \nabla_{T_0,T_1} - \sum I_{T_1} \nabla_{T_0,T_1}}{\sum \nabla_{T_0,T_1}^2} \tag{4.36}$$

where ∇_{T0,T_1} is the difference image $I_{T_0} - I_{T_1}$. The derivation is presented in appendix A.

The assumption underpinning this model is that the template frames I_{T_0} and I_{T_1} closely approximate the actual frames mixed in the dissolve transition. For ideal dissolves, in which two static shots are mixed, this will be be the case. However, this will not hold where the shots contain camera or object motion in the dissolve region. These motion effects are now considered and addressed.

4.3.2 Global motion

Changes in camera orientation and zoom introduce a difference between successive frames that reduces the extent to which a template image I_{T_0} is representative of the frames in a shot. To counteract this effect, global motion compensation is applied. Any global motion estimation technique can be employed; in this work a robust estimator presented by Odobez and Bouthemy is used [244]. A six-parameter affine motion model is used; the motion parameters between frames I_t, $I_{t'}$ are a 2×2 matrix $\mathbf{A}$ and a displacement vector $\mathbf{d}$. For image content subject to global motion, the two frames will be related by

$$I_{t'}(\mathbf{x}) = I_t(\mathbf{A}\mathbf{x} + \mathbf{d}) \tag{4.37}$$

For convenience of computation, the affine parameters are combined into a matrix $\mathbf{M}$

$$\mathbf{M}_{(t,t')} = \begin{pmatrix} A_{00} & A_{01} & 0 \\ A_{10} & A_{11} & 0 \\ d_0 & d_1 & 1 \end{pmatrix} \tag{4.38}$$

which can be applied to homogeneous coordinates.

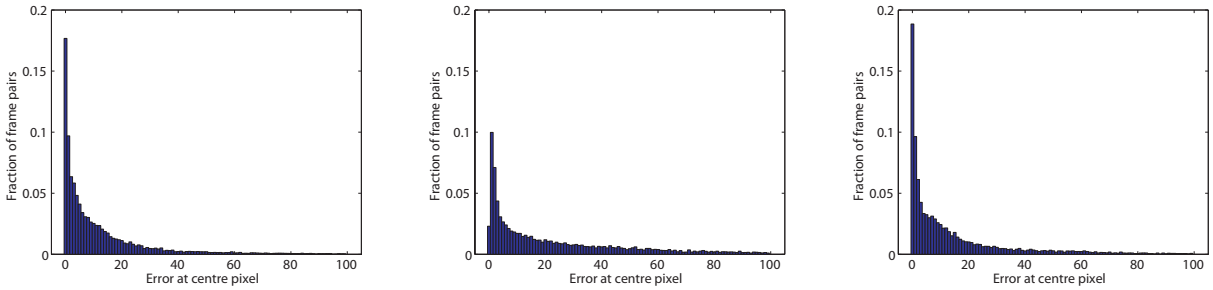
For the purposes of α-curve estimation, the global motion is estimated between every pair of frames I_t, I_{t+1}. This is the *forward* global motion. The *backward* global motion is the parameters relating frame I_{t+1} to I_t, and is assumed here to be the inverse of the forward parameters, i.e. $\mathbf{M}_{(t+1,t)} \approx (\mathbf{M}_{(t,t+1)})^{-1}$.

These parameters are then used to compensate both template images to register them with image I_t before the estimation of α is carried out. Rather than estimate $\mathbf{M}_{(T_0,t)}$ directly, the intermediate global motion estimates are combined according to

$$\mathbf{M}_{(T_0,t)} \approx \prod_{t'=T_0}^{t-1} \mathbf{M}_{(t',t'+1)} \tag{4.39}$$

and the cumulative backward motion is found similarly.

After these compensations, only a partial region of each template frame will contain valid data. Estimation of α is performed over the area of intersection of the two valid regions. One potential difficulty here is that where the global motion is very fast, the valid regions of the temporal frames will have no overlap. For example, the motion $\mathbf{M}_{(T_0,t)}$, with $t = T_0 + 10$, may be so large that I_{T_0} and I_t have no content in common. Where this happens, there is no basis



(a) 0-diff method (b) 1-diff method (c) 3-diff method

Figure 4.11: Global motion prediction errors for three different prediction orders. The data set was 14399 frames of broadcast cricket footage, containing a wide variety of camera motion types.

for estimating α using the template frames. Such fast global motion is very rarely encountered in standard video, and still less often within a dissolve region. If no common region is available, then, the transition is classified as 'not a dissolve'.

Estimation of the global motion $\mathbf{M}_{(T_0,t)}$ may be particularly difficult if there is a partial dissolve between frames I_{T_0} and I_t, as the global motion estimation cannot take the effects of the dissolve into account. One approach to this problem is to rely on extrapolating global motion parameters from the frames outside the suspected transition into the transition range, rather than using motion estimation within this region. Three simple prediction methods are assessed here, generated by

$$\mathbf{M}^0_{(T_0,t)} = \left(\mathbf{M}_{(T_0,T_0+1)}\right)^{(t-T_0)} \quad \text{0-diff method} \tag{4.40}$$

$$\mathbf{M}^1_{(T_0,t)} = \prod_{i=0}^{(t-T_0)-1} \mathbf{M}_{(T_0,T_0+1)} + i\left(\mathbf{M}_{(T_0,T_0+1)} - \mathbf{M}_{(T_0-1,T_0)}\right) \quad \text{1-diff method} \tag{4.41}$$

$$\mathbf{M}^3_{(T_0,t)} = \prod_{i=0}^{(t-T_0)-1} \mathbf{M}_{(T_0,T_0+1)} + i\left(\nabla\mathbf{M}\right) \quad \text{3-diff method} \tag{4.42}$$

$$\text{where } \nabla\mathbf{M} = \frac{1}{3}\sum_{i=0}^{2} \mathbf{M}_{(T_0-i,T_0-i+1)} - \mathbf{M}_{(T_0-i-1,T_0-i)} \tag{4.43}$$

The *0-diff* method assumes that the global motion is essentially constant over short intervals. In the the *1-diff* method, each parameter is modeled linearly; in other words, here $\frac{d\mathbf{M}}{dt}$ is assumed constant. The *3-diff* method relies on the same assumption, but incorporates a smoothing of the slope by taking the average of three preceding values of $\frac{d\mathbf{M}}{dt}$. The performance of these prediction methods over an interval of 10 frames over within-shot footage (i.e. footage not containing cuts or partial dissolves) is shown in figure 4.11. Equation 4.39 is used to generate the correct $\mathbf{M}_{(T_0,t)}$. The error measure employed is the displacement of the centre pixel, for example

$$e^0 = \left|\begin{pmatrix}0 & 0 & 1\end{pmatrix}\left(\mathbf{M}_{(T_0,t)} - \mathbf{M}^0_{(T_0,t)}\right)\right| \tag{4.44}$$

Figure 4.11 shows that none of these measures is consistently reliable for prediction over a 10-frame interval. Furthermore, it is found that simply using global motion estimation directly across the fade region results in acceptable α-curves over a wide variety of dissolve transitions. For these reasons, the use of motion extrapolation has not been pursued.

(a) I_{T_0} (frame 487) (b) I_t (frame 507) (c) I_{T_1} (frame 527) (d) Residual image, I_e

Figure 4.12: Residual image after one iteration of α estimation for a cut. It can be seen that the highest residual values correspond to local motion in this example. These locations are assigned a low weight in subsequent iterations. In this example, the inital value for α is 0.944, and after 10 iterations it has risen to 0.985, closer to the ideal value of 1.0.

4.3.3 Local motion

Local motion in the dissolve region will introduce localised discrepancies between the template image and the current frame. These discrepancies violate the fade model assumptions, and so confound the α estimation process. This local motion effect can be greatly reduced using an iteratively reweighted least-squares (IRLS) method. The idea is to assign a low weight to pixel sites where the model error is very high, to reduce the influence of these sites on the α estimation.

An estimate for α is obtained by maximizing the weighted likelihood

$$p(\alpha|I_t) \propto \exp(-\sum_{\mathbf{x}} w(\mathbf{x}) \left[I_t(\mathbf{x}) - I_{M(\alpha)}(\mathbf{x})\right]^2) \tag{4.45}$$

at each iteration, using

$$\alpha_{opt} = \frac{\sum w_t^2 I_t \nabla_{T_0,T_1} - \sum I_{T_1} w_t^2 \nabla_{T_0,T_1}}{\sum w_t^2 \nabla_{T_0,T_1}^2} \tag{4.46}$$

For the first iterations, the weights are initialised to 1 everywhere. The weights are updated according to the residual image, $I_e = I_{M(\alpha)} - I_t$, according to

$$w(\mathbf{x}) = \frac{1}{(1+r^2(\mathbf{x}))} \tag{4.47}$$

$$\text{where } r(\mathbf{x}) = \frac{I_e(\mathbf{x})}{\tau s \sqrt{(1-h(\mathbf{x}))}} \tag{4.48}$$

$$s = \frac{\text{median}(|I_e - \text{median}(I_e)|)}{0.6745} \tag{4.49}$$

$$h(\mathbf{x}) = \frac{I_e^2(\mathbf{x})}{\sum_{\mathbf{x}} I_e^2(\mathbf{x})} \tag{4.50}$$

This is a Cauchy weighting function where the residuals $I_e(\mathbf{x})$ are being scaled to take into account the leverage h. s is a robust estimator for the standard deviation of the residuals. τ is a tuning constant; the value 2.385 is standard for Cauchy weighting. Further details on this approach are available in any work on robust statistics, such as [156].

The estimation is terminated when either of two conditions is met:

- The number of residuals exceeding a certain threshold is small.
- The number of iterations exceeds a fixed limit.

The appropriate thresholds must be chosen; currently α estimation is discontinued if less than 2% of the image has an error of more than 20 graylevels, and the number of iterations is limited to 10. These limits have been found to generate good results for a variety of sequences. In general termination will only be triggered by the first condition in sequences with negligible local motion; for many sequences, the procedure runs to the limit of iterations. It is observed that the value of α found has almost always stabilised at this stage.

Figure 4.12 shows the residuals found for α estimation across a cut transition.

4.3.4 Examples of α-curves

Figure 4.13 shows α-curves calculated for a variety of sections of footage. The characteristic shape of the α-curve for a dissolve transition is shown in figure 4.13 (c). This is made up of three regions: where α is close to 1, as frames are from the first shot; a smoothly descending region within which α changes from 1 to 0 (usually linearly); and a final flat region where the frames are from the second shot, where α is close to 0.

The figure shows α-curves for some other transition types as well. The curves for the cut and wipe transitions are somewhat similar to that of the dissolve transition, differing in the steepness of the transition region. Cuts result in a large drop in the α value across the cut boundary, while wipes introduce a descent region steeper than that of the typical dissolve. However, dissolves of only one frame's duration have been observed. These cannot be distinguished from wipes on the basis of the α-curve alone.

The effectiveness of the motion compensating techniques described above is illustrated in figure 4.14. Some frames from a particularly difficult dissolve is shown in figure 4.14 (a). Here there is a fast pan in both shots over the dissolve region, and significant local motion is also present. Figures 4.14 (b)-(d) show the α-curves found by estimation over these frames with each combination of global and local motion compensation. Only when both global motion compensation and local motion reweighting are applied is the α-curve shape characteristic of a dissolve observed.

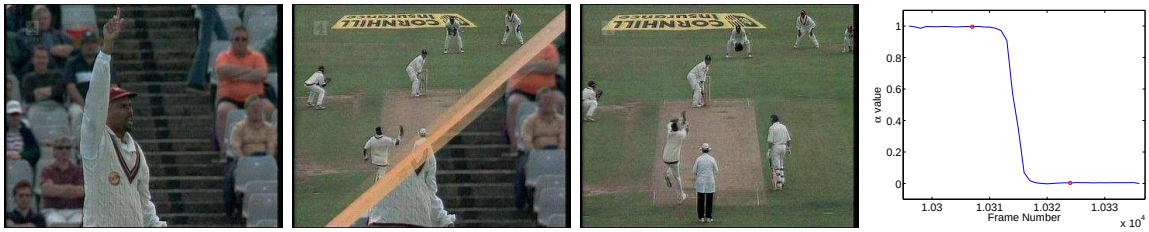
4.3.5 Longer Dissolves

The examples shown thus far have shown dissolves of between ten and twenty frames' duration. Some dissolves can be considerably longer, which has implications for their characterisation by this method. In particular, the template images may be less representative of the shot content within the transition. In archival footage, the increased transition length means that there is a greater probability of some defect, such as flicker or large blotches, occurring in the footage and violating the dissolve model. These considerations are offset to some degree by the fact

(a) α-curve over a static shot

(b) α-curve across a cut

(c) α-curve over a dissolve transition

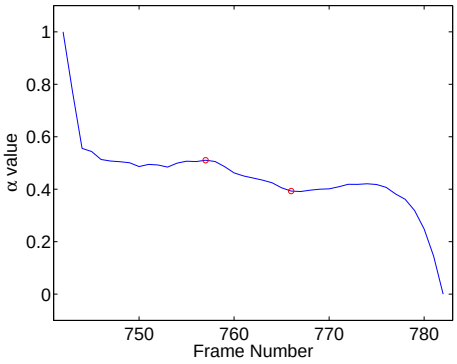


(d) α-curve across a wipe transition

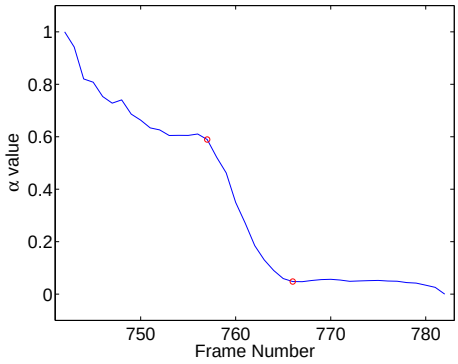
(e) α-curve over a camera zoom transition

Figure 4.13: α-curves for a variety of shot content types.

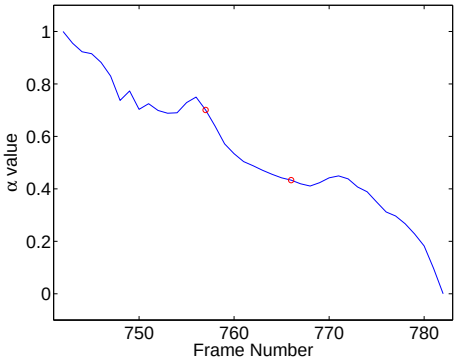
(a) A dissolve between two shots with significant global and local motion.



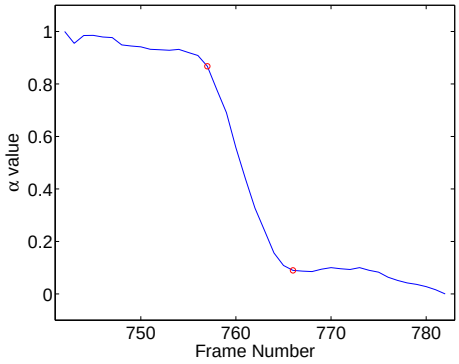
(b) Direct estimation



(c) Local motion reweighting only

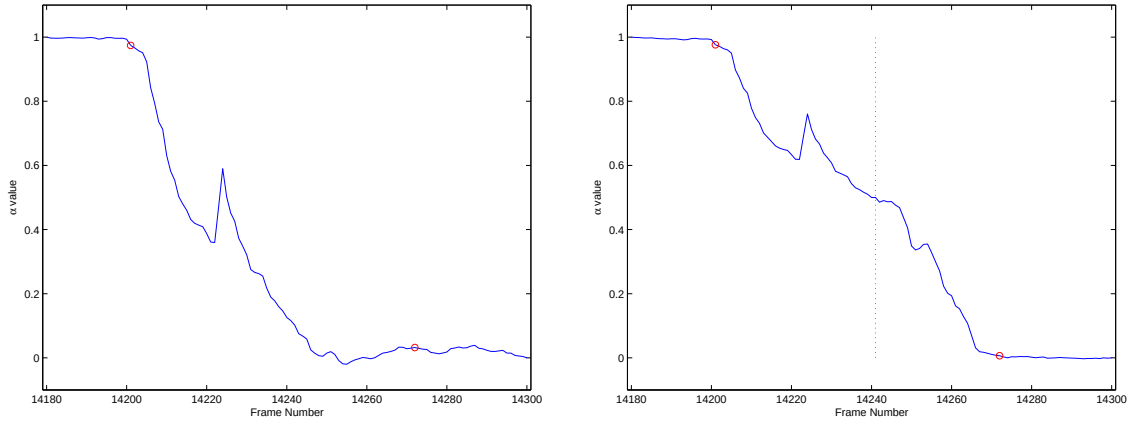


(d) Global motion compensation only



(e) Global motion compensation and local motion reweighting

Figure 4.14: Alpha estimation across a fast dissolve with significant global and local motion. The dissolve starts at frame 19 and ends at frame 27.



(a) Analysis over 80 frames.

(b) Analysis over adjacent windows of 40 frames each. The dashed vertical line indicates where the curves have been joined.

Figure 4.15: Alpha traces computed over a 70 frame dissolve.

that stylistically, longer transitions are generally considered appropriate for footage with little motion. In less active footage, the template frames will be representative of the shot content over a wider temporal range.

The dissolve model equation is also valid over a partial dissolve region. This can be exploited to improve performance in analysis of a long transition region. The area of the video containing the suspected transition can be split into two non-overlapping windows, and the α-curve for each computed separately. Let the transition region under investigation consist of N frames. This range is split at frames $S = N/2$, and the α-curves for each region, denoted α_a and α_b, are found. The overall curve $\alpha(t)$ is found by

$$\alpha(t) = \begin{cases} 0.5 + 0.5\alpha_1(t), & \text{if } 0 \leq t < S \\ 0.5\alpha_2(t - S), & \text{if } S \leq t \leq N \end{cases} \tag{4.51}$$

Figure 4.15 shows an example of α values computed over a long (70 frame) dissolve. In figure 4.15 (a), the values are computed over the entire transition duration in one pass, whereas in figure 4.15 (b) the values are found over two adjacent windows. The second trace improves on the first in a number of respects. The downward sloping region corresponds more closely to the actual duration of the dissolve; the flat regions corresponding to the regions outside the dissolve transition are smoother; and the spike in the curve at frame 14225, which is due to momentary strong flicker in the video, is considerably attenuated in the second trace. This factors make analysis of the dissolve curve easier, as described below.

4.3.6 Changepoints in the α-curve

The α values for each frame in the transition region form the α-curve. The transition is to be classified as a dissolve or not a dissolve on the basis of this curve. It has been observed that the α-curve over a dissolve transition consists of two flat regions connected by a smoothly descending transition region corresponding to the duration of the dissolve. Classification of α-curves into those corresponding to dissolves versus other transitions is based on the assumption that each of the three segments can be modelled as approximately linear. The method is to fit three lines to consecutive segments of the curve. If these fits have a low error, and the slopes of the lines correspond with those expected for a dissolve, the curve is classified as such.

A general-purpose model estimation technique, such as Expectation Maximization (EM), could be used to find an optimal three-line fit to the curve. However, these techniques are not generally designed to exploit *a priori* information regarding the ordering of the data. In this case, it is known that the points assigned to each line will be adjacent, and so the problem reduces to finding the optimal changepoints along the curve. While numerous sophisticated changepoint detection have been presented, a simple exhaustive search approach is adopted here. Two changepoints need to be found, so an exhaustive search is an order N^2 computation, where N is the length of the curve. The curve only contains 40 to 80 values, so the computation is feasible.

For an α-curve ranging over frames $T_1 \ldots T_2$, the optimal changepoints $C = \{\hat{c}_1, \hat{c}_2\}$ are found by solving

$$C = \arg\min_{c_1,c_2} \left(\omega E(T_1, c_1) + E(c_1, c_2) + \omega E(c_2, T_2)\right) \tag{4.52}$$

using exhaustive search, with the constraints $c_1 < T_2 - 5$ and $c_2 > c_1 + 2$. $E(t_1, t_2)$ is a sum squared error function describing how well the points $t_1 \ldots t_2$ along the α-curve can be modelled using a line:

$$E(t_1, t_2) = \sum_{t=t_1}^{t_2} \left(O(t_1, t_2) + (t - t_1)\lambda(t_1, t_2) - \alpha(t)\right)^2 \tag{4.53}$$

$O(t_1, t_2)$ and $\lambda(t_1, t_2)$ are the least-squares estimate of the parameters of the line along the α-curve over $t_1 \ldots t_2$. ω is a weight intended to bias the procedure such that a good fit in the first and third segments is more important than a good fit over the second (dissolve) section. This is introduced because the value of α may not change linearly over the dissolve; for example, a manually controlled dissolve will have a more rounded α-curve.

To evaluate the accuracy of this changepoint detection method, the α-curves and changepoints of the dissolve transitions in the training data were found. The changepoints of each α-curve were found, and compared to the start and end of the dissolve transitions as indicated by the ground truth data. Let E_1 be the error in estimating the dissolve start from the α-curve (in frames), and E_2 be the error in estimating the dissolve end. The mean and variance of these two error measures over the 288 dissolves in the training set were found as shown in table 4.2.

ω	$\bar{E}_1$	$\sigma^2_{E_1}$	$\bar{E}_2$	$\sigma^2_{E_2}$
2^0	0.788194	5.46369	-1.40278	6.97309
2^1	0.621528	5.05835	-1.0625	4.64416
2^2	0.517361	4.81503	-0.961806	4.62223
2^3	0.368056	4.51215	-0.857639	4.24099
2^4	0.211806	4.23721	-0.670139	4.57722
2^5	0.0173611	4.35858	-0.4375	4.58841
2^6	-0.305556	4.82617	-0.138889	5.14441
2^7	-0.628472	5.89285	0.1875	5.3027
2^8	-0.986111	7.28204	0.545139	6.21398
2^9	-1.40972	8.44478	0.9375	6.92988
2^{10}	-2.01736	8.81851	1.55903	7.84319

Table 4.2: Performance of dissolve endpoint detection based on α-curve analysis, for different values of ω.

It is evident that ω values greater than unity improve performance up to about 2^5; this is the value is adopted for use in the procedure.

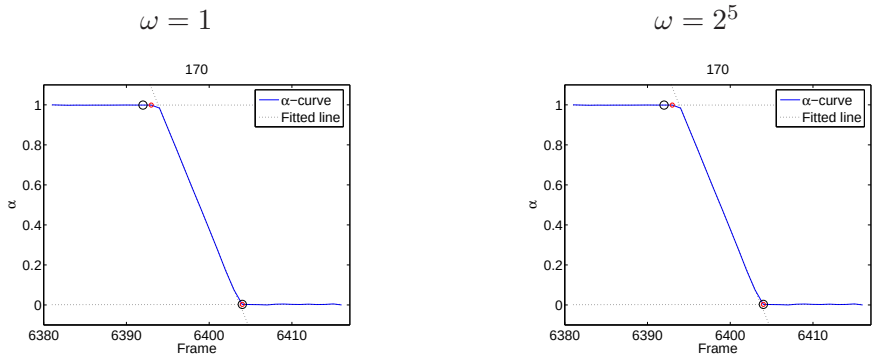
Figure 4.16 shows some examples of α-curves partitioned using the method described above. In the figures in the left column, ω is set to 1, i.e. no weighting. In the figures in the right column, ω is set to $2^5 = 32$, as described in the previous paragraph. The value of ω has no effect on the segmentation in example (a), as this α-curve is almost perfectly linear in each of the three segments.

In example (b), the non-linearity of the α-curve in the dissolve section has resulted in a mis-segmentation for $\omega = 1$. With $\omega = 32$, errors in the dissolve section are preferred to errors in the first and third sections, allowing for the non-linearity and resulting in an accurate segmentation.

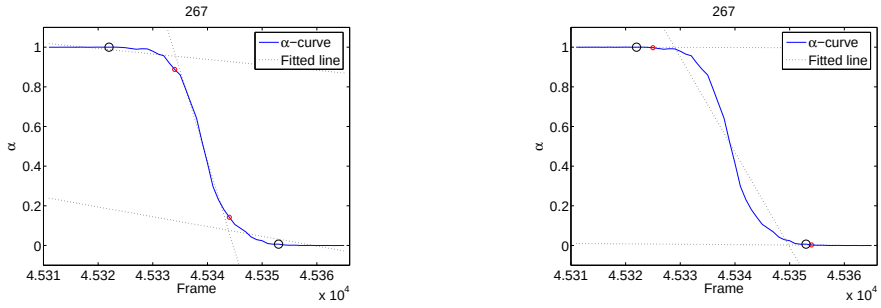
The line-fitting procedure is such that points in the pre- and post-dissolve regions of the α-curve can be assigned to the middle segment even when they have a lower error when assigned to an adjacent segment. This occurs where the α-curve is noisy in the dissolve region. Example (c) illustrates a case where this has occurred.

The use of $\omega > 1$ can cause problems where the pre- and post-dissolve regions are noisy, as shown in example (d). Here the start of the dissolve is missed by 6 frames where $\omega = 32$, as opposed to 2 frames when $\omega = 1$. On the other hand, detection of the dissolve end is more accurate with $\omega = 32$. The results in table 4.2 suggest that non-linear dissolves are more common than non-linear pre- and post-dissolve regions, as overall detection performance is best with $\omega = 32$.

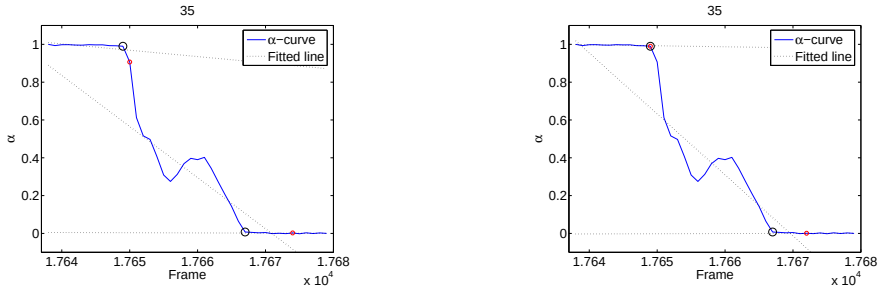
Various adjustments to the line-fitting procedure were investigated to address the issues raised by examples (c) and (d) above. In example (c), noise in the dissolve region of the α-curve



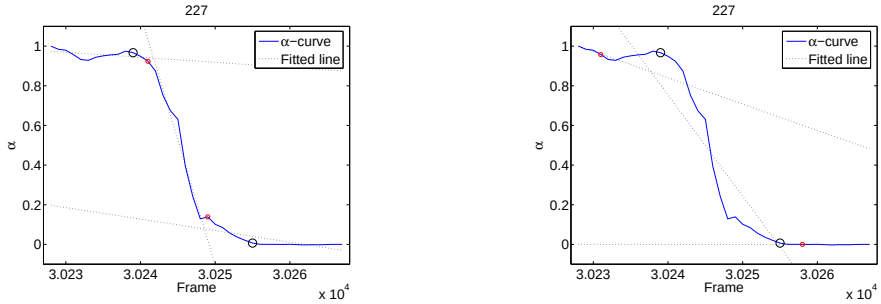
(a) Linear dissolve.



(b) Non-linear dissolve.



(c) Noisy, non-linear dissolve.



(d) Non-linear dissolve with non-linear pre-dissolve region.

Figure 4.16: Examples of α-curves and the lines fitted to them. The ground truth data for the dissolve region is indicated by black circles. The detected changepoints are indicated by red circles.

resulted in the some points of the flat, post-dissolve region being assigned to the dissolve region. Two refinements to address this were evaluated. The first was an alternative fitness function, designed to encourage a segmentation with as short a dissolve region as possible while ignoring the goodness-of-fit of the line over the dissolve region. Here the changepoint estimation equation was modified to

$$C = \arg\min_{c_1,c_2} \left(\omega'(c_2 - c_1) + E(T_1, c_1) + E(c_2, T_2)\right) \tag{4.54}$$

Performance of this measure was evaluated for values of ω' ranging from 2^{-15} to 2^{-25} (i.e. of comparable magnitude to the mean squared error of the lines fit in the pre- and post-dissolve regions). However, no value of ω' resulted in detection performance better than that obtained with $\omega = 32$.

The second modification investigated was to refine the changepoints after the best-fit lines had been found. Let O_i and λ_i be the parameters of the line i. The pre- and post- dissolve regions α-curve are then extended to the last point that is closer to the associated line than either of the other two, as follows:

$$\begin{aligned} \hat{c}_1 &= \sup\{c_1 : E(O_1, \lambda_1, c_1) < \min\left(E(O_2, \lambda_2, c_1), E(O_3, \lambda_3, c_1)\right)\} \\ \hat{c}_2 &= \inf\{c_2 : E(O_3, \lambda_3, c_2) < \min\left(E(O_1, \lambda_1, c_2), E(O_2, \lambda_2, c_2)\right)\} \end{aligned} \tag{4.55}$$

Here $E(O_1, \lambda_1, c_1)$ is the squared distance from $\alpha(c_1)$ to the line defined by (O_1, λ_1). While this did improve performance in some cases (including that illustrated in example (c) above), segmentation performance overall was not greatly affected.

Example (d) illustrates the problems that arise with non-linear pre- and post-dissolve regions. An Iteratively Reweighted Least Squares (IRLS) technique for estimation of the line parameters (O, λ) within the exhaustive search procedure was implemented to address these issues. This was not found to improve performance, and increased processing time considerably.

4.3.7 Dissolve Detection using the α-curve

The technique discussed above is a means to identify the dissolve start and end points, given an α-curve generated from a sequence in which a dissolve does occur. Deciding *whether* a sequence contains a dissolve, based on analysis of the α-curve, is a separate issue.

Eight parameters of the segmented α-curve are exploited for this purpose. These are the slopes of the three line segments, denoted λ_{1-3}; the error in estimation of the dissolve start and end points relative to other indicators, denoted $\hat{c}_{1,2}$; and the error in the line fit over each segment, E_{1-3}.

These heuristics are used to generate an Bayesian α-curve classifier, using likelihood distributions for each parameter. For example, for the slope of the first linear segment λ_1, the distributions $p(\lambda_1|D)$ and $p(\lambda_1|\bar{D})$ are found, where D is the event 'dissolve'. An α-curve can be classified as corresponding to a dissolve (event D) or a non-dissolve (event $\bar{D}$) based on comparison of these likelihood distributions.

The $\psi_{22}(t)$ function described in the previous section is used to identify candidate regions in the training set; α-curve estimation is undertaken everywhere $\psi_{22}(t) > 0$. These frames correspond to peaks in d_{22}. The width of the peak is estimated as in the previous section. Denote the estimated peak start and endpoints by P_s and P_e. As outlined above, the end of the dissolve is expected to be at $P_e - 22$, as frames are compared over a 22 frame interval. α-curve estimation is then carried out over the frames $T_1 = P_s - 10 \ldots T_2 = P_e - 22 + 10$, i.e. ten frames are added on either side of the expected dissolve region. The changepoints in the α-curve should then be found ten frames from either end. The $\hat{c}_{1,2}$ metrics used for classification of the α-curves are then defined by

$$\hat{c}_1 = \frac{|c_1 - 10|}{T_2 - T_1} \tag{4.56}$$

$$\hat{c}_2 = \frac{|10 - (T_2 - c_1)|}{T_2 - T_1} \tag{4.57}$$

Figures 4.17 - 4.19 show the distributions of these metrics for dissolve and non-dissolve α-curves.The distributions of the slope metrics λ_1 and λ_3, as well as the line fit errors, are modelled with exponential distributions. The distributions of slope over R_2 are modelled as Gaussians, and the changepoint error metrics $c_{1,2}$ are modelled using a gamma function. Some metrics are multiplied by a constant to increase their range of values; these constants are indicated in the figures. Some of the distributions for the non-dissolve metrics have spikes at the upper end of the range of values that are not explicitly modelled by the approximating function. This is not of concern, however, as in all cases the approximating function of the non-dissolve regions is several orders of magnitude larger than that of the dissolve regions for these values.

Classifying an α-curve as either a dissolve transition or not can then be achieved by evaluation of

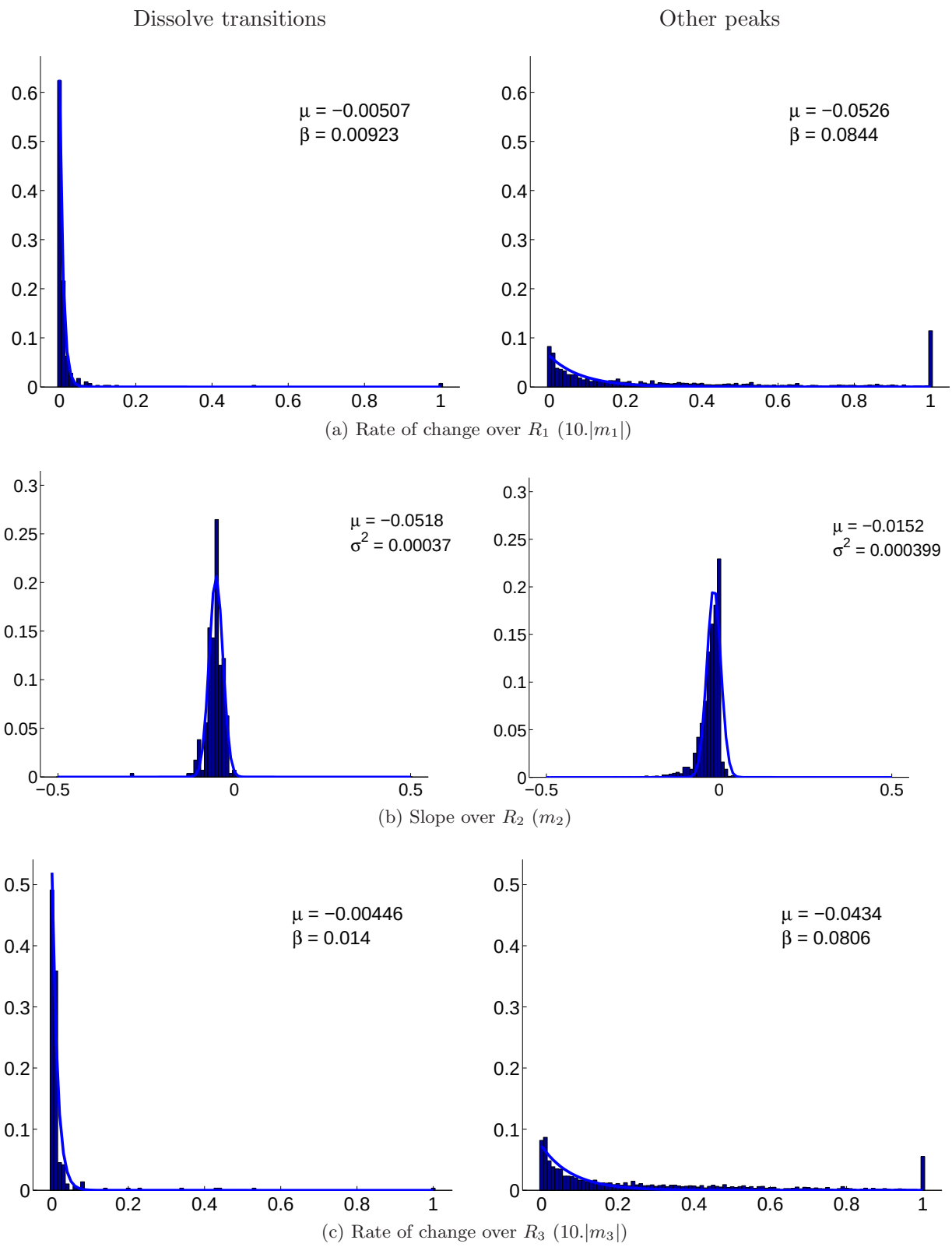
$$\begin{aligned} \psi'_\alpha &= \frac{P(\lambda_1, \lambda_2, \lambda_3, \hat{c}_1, \hat{c}_2, E_1, E_2, E_2|D)P(D)}{P(\lambda_1, \lambda_2, \lambda_3, \hat{c}_1, \hat{c}_2, E_1, E_2, E_2|\bar{D})P(\bar{D})} \\ &\simeq \frac{P(\lambda_1|D)P(\lambda_2|D)P(\lambda_3|D)P(\hat{c}_1|D)P(\hat{c}_2|D)P(E_1|D)P(E_2|D)P(E_2|D)P(D)}{P(\lambda_1|\bar{D})P(\lambda_2|\bar{D})P(\lambda_3|\bar{D})P(\hat{c}_1|\bar{D})P(\hat{c}_2|\bar{D})P(E_1|\bar{D})P(E_2|\bar{D})P(E_2|\bar{D})P(\bar{D})} \end{aligned} \tag{4.58}$$

If $\psi'_\alpha > 1$, the likelihood distributions suggest that the α-curve corresponds to a transition. The prior probability for the presence of a dissolve, $P(D)$, was found to be 0.0932, with $P(\bar{D}) = 1 - P(D)$.

4.3.8 Comparison to other dissolve-detection metrics

As described in the previous chapter, other measures have been proposed for dissolve analysis, intended to be used in a similar fashion to the α-curve. The *median relative frame difference* [68, 133] is defined as

$$M(t) = \underset{\mathbf{x}}{\text{median}} \left(\frac{I_t(\mathbf{x}) - I_{t-1}(\mathbf{x})}{I_t(\mathbf{x})} \right) \tag{4.59}$$

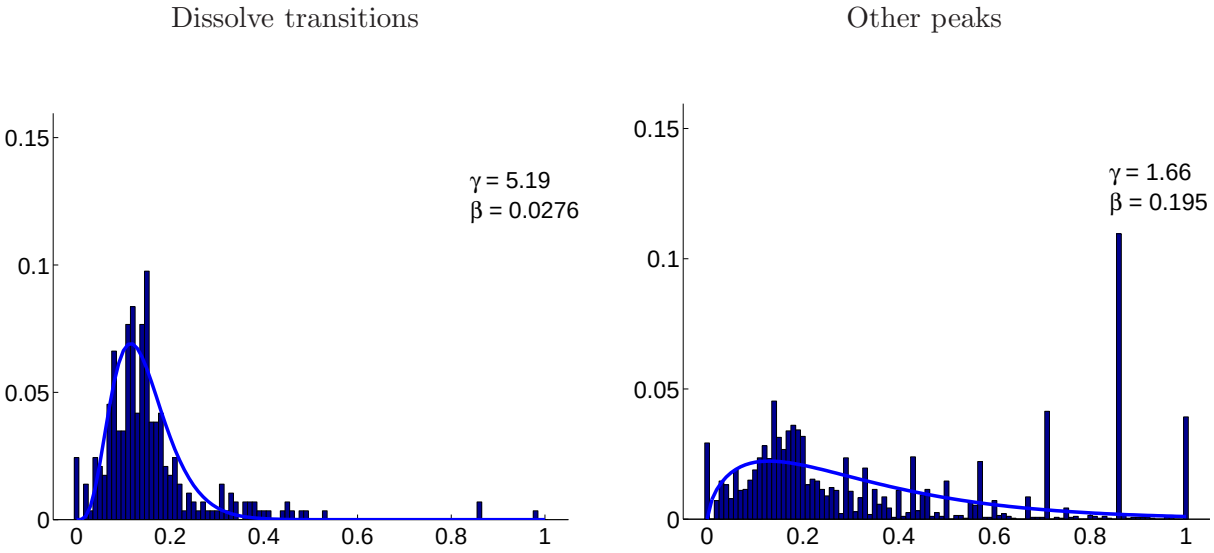


(a) Rate of change over R_1 $(10.|m_1|)$

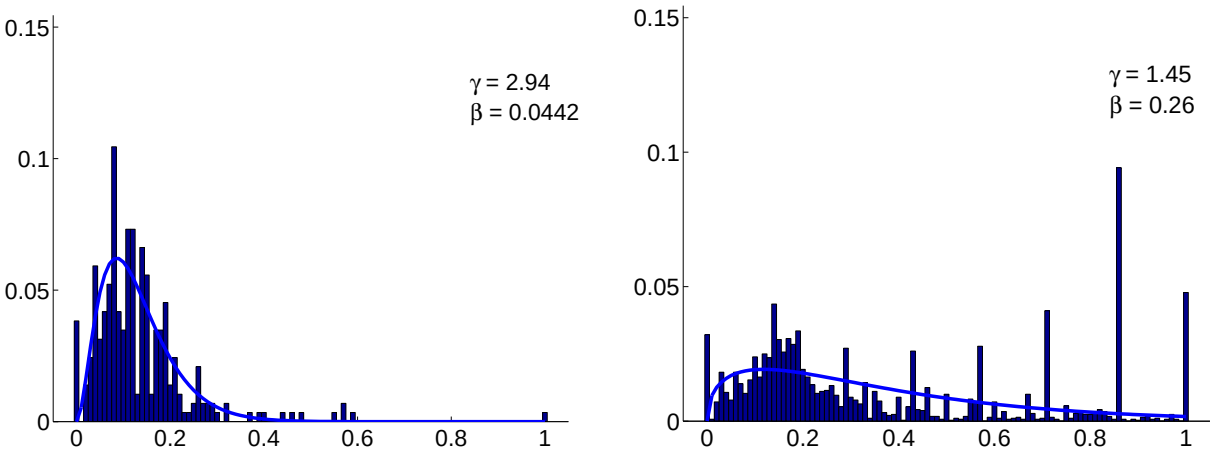
(b) Slope over R_2 (m_2)

(c) Rate of change over R_3 $(10.|m_3|)$

Figure 4.17: Distributions of α-curve slope metrics



(a) Relative error α-curve changepoint c_1 versus δ_{22} peak start



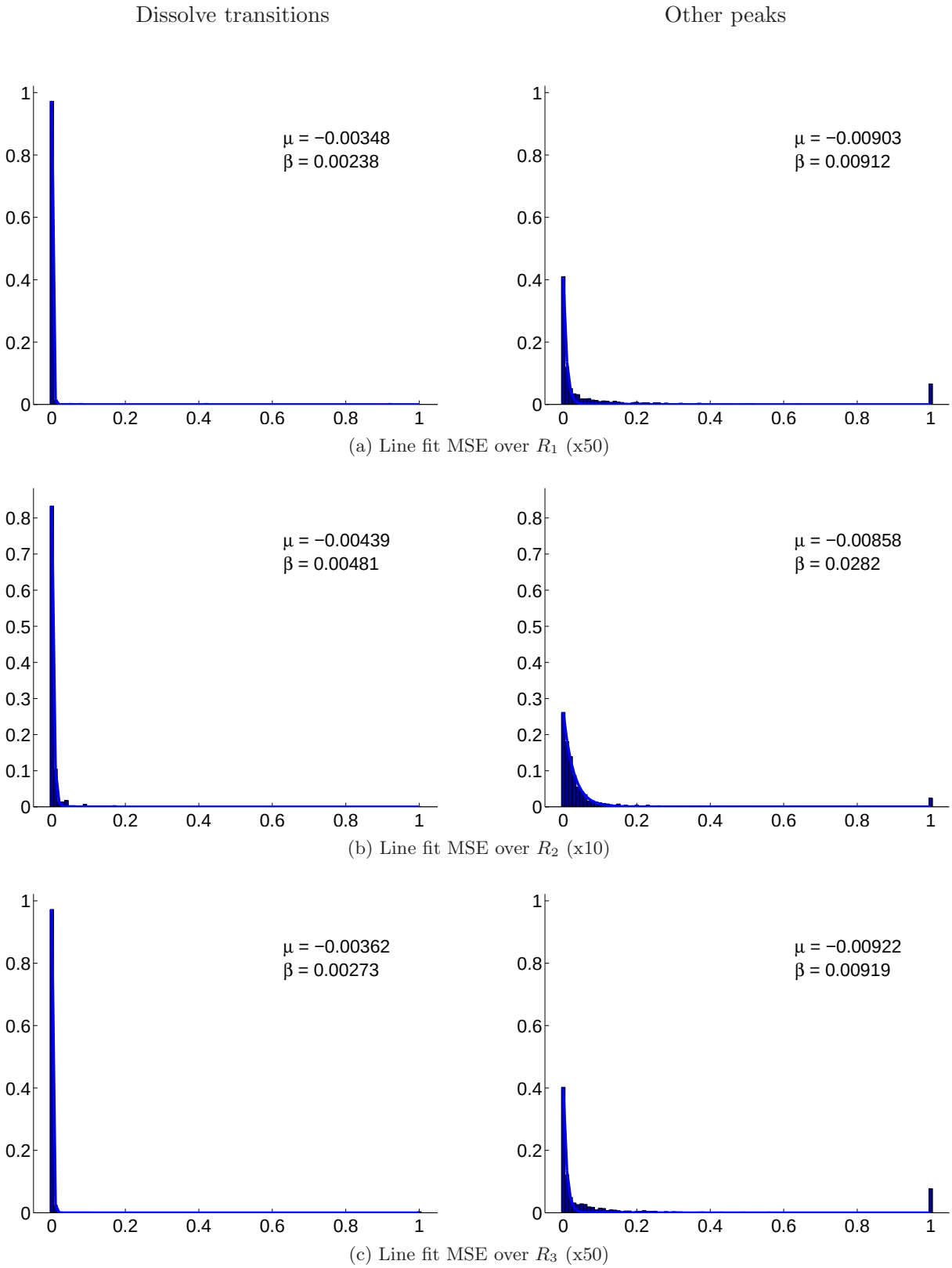
(b) Relative error α-curve changepoint c_2 versus δ_{22} peak end

Figure 4.18: Distributions of α-curve changepoint metrics

which should be zero within a shot, and some non-zero constant over a dissolve. The *variance of intenstity values* [8] is simply

$$\sigma(t) = \mathrm{var}_{\mathbf{x}}\left(I_t(\mathbf{x})\right) \tag{4.60}$$

which should follow a downward parabola over the dissolve region. Two aspects of these signals are under discussion here. Firstly, the overall quality of each of the signals can be considered, i.e. how well they conform to their expected shape over a dissolve region. Figure 4.20 shows the curves generated by each of these measures over a dissolve from a cricket sequence. For this dissolve, the α-curve generates a smoother sequence of values than either of the other two measures. A separate question is how to quantify how close the curve is to the shape expected for a dissolve. This is achieved via some shape analysis function $\psi(t)$, similar to the peak shape



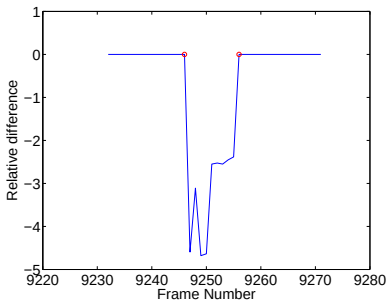
(a) Line fit MSE over R_1 (x50)

(b) Line fit MSE over R_2 (x10)

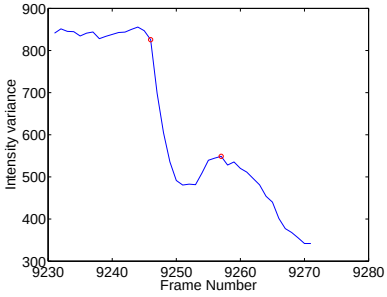
(c) Line fit MSE over R_3 (x50)

Figure 4.19: Distributions of α-curve line-fit error metrics

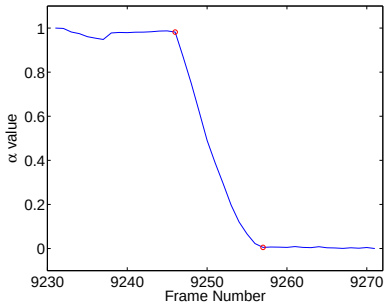
(a) Frames from a dissolve transition in cricket footage



(b) Median of relative intensity difference [68, 133]



(c) Variance of intensity values [8]



(d) α-curve

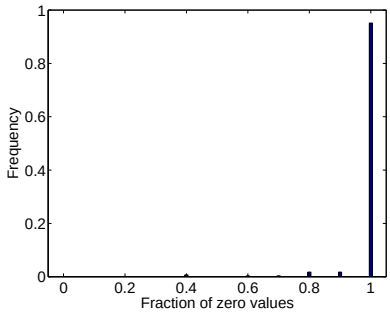
Figure 4.20: Three different means of characterising frames for dissolve detection. The α-curve is the 'best-behaved' measure.

analysis functions applied to the $\delta(t)$ signals above.

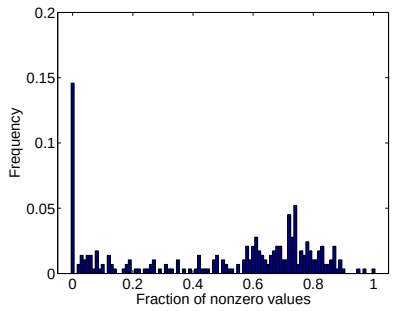
To further examine the relative reliability of the three dissolve characteristics above, the median difference, intensity variance, and α-curves were found for the dissolves in the training set. For each dissolve in the training corpus, three regions were defined: the pre-dissolve region, r_1, consisting of 10 frames before the start of the dissolve; the dissolve region, r_2, consisting of the dissolve frames; and the post-dissolve region, r_3, consisting of 10 frames after the end of the dissolve. The properties of $M(t)$, $\sigma(t)$, and $\alpha(t)$ in each of these regions, for each dissolve, are used to characterise their reliability.

Median frame-to-frame relative intensity difference: $M(t)$, is expected to be zero within a shot, and non-zero within a dissolve transition. Figure 4.21 shows how well the dissolves in the training corpus conform to these criteria. Within the pre- and post-dissolve regions, the fraction of frames having a zero median intensity difference is close to 1 for most of the dissolves in the training set. However, the fraction of frames for which $M(t)$ is nonzero within r_2 is much more variable. For almost 15% of the 288 training dissolves, $M(t) = 0$ for all frames in r_2; these are found to be gradual dissolves between similar scenes. On this evidence, the median relative intensity difference is not a very reliable tool for dissolve detection.

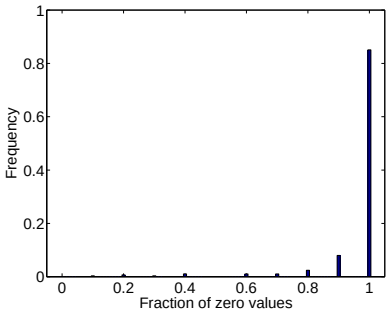
Variance of frame intensity values: It was shown in the previous chapter that the variance of



(a) Fraction of frames where the median intensity difference is zero in the ten frames preceding a dissolve.



(b) Fraction of frames where the median intensity difference is nonzero over a dissolve.



(c) Fraction of frames where the median intensity difference is zero in the ten frames after a dissolve.

Figure 4.21: The reliability of the median intensity difference as a dissolve detection metric.

intensity values of frames within a dissolve should be less than that of the frames in the adjoining shots. Figure 4.22 (a) illustrates an example of a well-formed $\sigma(t)$ signal over a dissolve. The curve is flat over r_1 and r_3, and dips in a downward-parabolic shape over r_2.

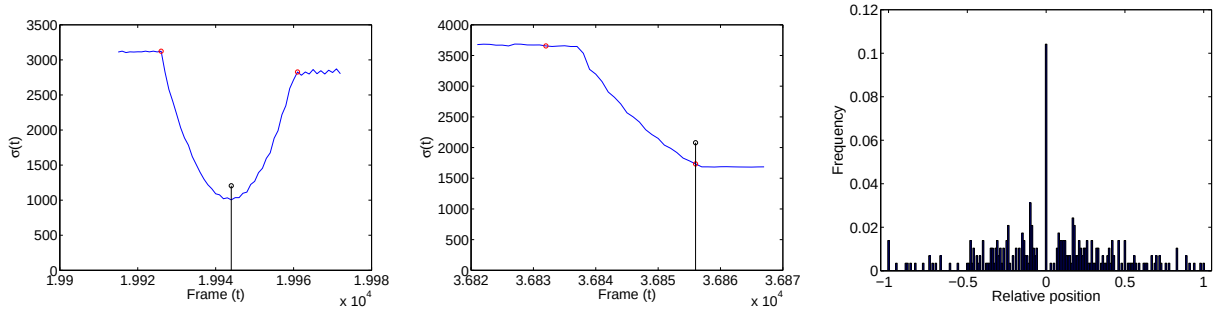
It is notable that the frame having minimum variance within the dissolve, designated $\hat{t}$, is not always in the middle of the dissolve. Where one of the connected shots has a much lower characteristic intensity variance than the other, the minimum value is at the start or end of the dissolve. The curve in this case is smoothly descending or ascending, as shown in the example in 4.22 (b). Fade transitions are a particular instance of this, as one of the connected shots is black and therefore has a variance of zero. Of the 288 dissolves in the training set, 18, or 6.25%, exhibit a descending or ascending $\sigma(t)$ curve.

The *relative position* $D(r_2)$ is a measure of how far the $\hat{t}$ is from the midpoint of the dissolve, $\bar{r}_2$:

$$D(r_2) = 2\frac{(\hat{t}-\bar{r}_2)}{|r_2|}, \text{ where } \bar{r}_2 = \frac{\max(r_2)+\min(r_2)}{2} \tag{4.61}$$

In a dissolve for which $D(r_2) = 0$, $\hat{t}$ is exactly in the middle of r_2. Where $D(r_2) = \pm 1$, the minimum is at one end. Figure 4.22 (c) shows the distribution of $D_R(2)$ for the dissolves in the training set. While 0 is the mode value, most of the dissolves exhibit some measure of assymetry.

A shape function, $\psi(t)$, is required to quantify how well a given σ-curve corresponds to one of the shapes characteristic of a dissolve. Hanjalic uses the function shown in equation 4.62 [135]. This function is based on the assumption that the variance of a frame in the middle of a dissolve should be less than that of the frames at either side of the transition. N here is the size of the analysis window, set to 21 frames. The function is applied at a point $t' = t - 11$, where t is the location of a maximum in δ_{22}, and 11 corresponds to half the interval used for frame differencing. Figure 4.23 shows the values taken by $\psi_H(t)$ over the dissolves in the training set.



(a) Parabolic dip in intensity variance over a dissolve.

(b) Smoothly descending variance over a dissolve.

(c) Position of minimum variance frame relative to dissolve midpoint.

Figure 4.22: Variance of intensity values in frames over a dissolve transition.

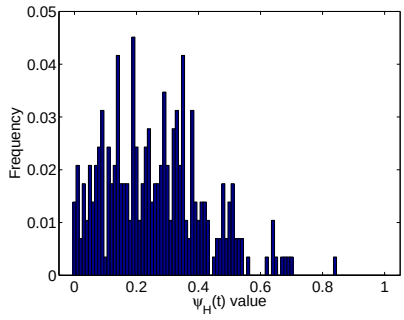
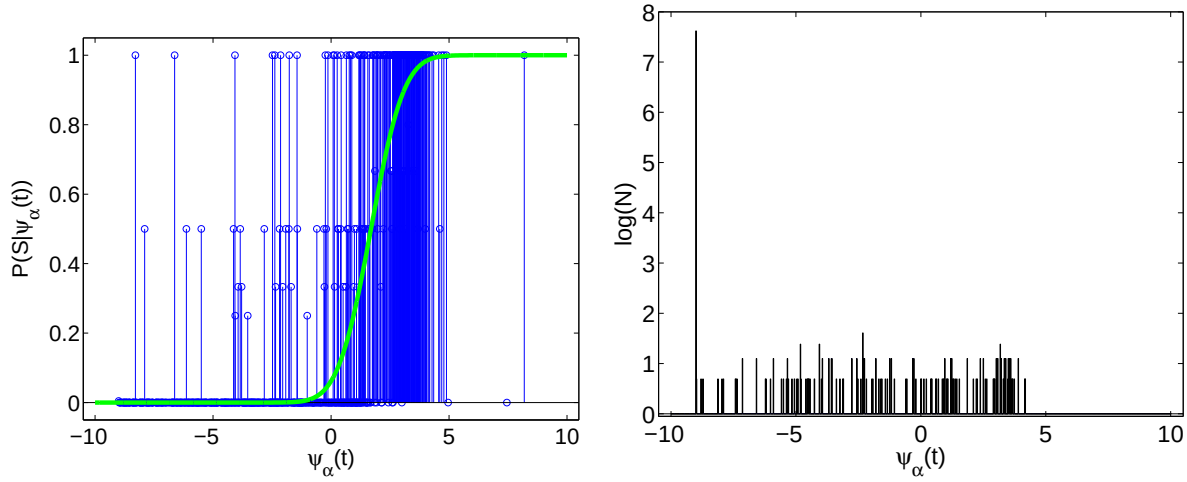


Figure 4.23: Values of $\psi_H(t)$ for the dissolves in the training set.

$$\psi_H(t) = \begin{cases} \left(1 - \frac{\min\left(\sigma(t'), \sigma\left(t' + \frac{N}{2}\right)\right)}{\max\left(\sigma(t'), \sigma\left(t' + \frac{N}{2}\right)\right)}\right) & \text{if } \left|\left(\sigma(t') - \sigma\left(t' + \frac{N}{2}\right)\right)\right| \leq \left|\left(\sigma(t') - \sigma\left(t' - \frac{N}{2}\right)\right)\right| \\ \left(1 - \frac{\min\left(\sigma(t'), \sigma\left(t' - \frac{N}{2}\right)\right)}{\max\left(\sigma(t'), \sigma\left(t' - \frac{N}{2}\right)\right)}\right) & \text{if } \left|\left(\sigma(t') - \sigma\left(t' + \frac{N}{2}\right)\right)\right| > \left|\left(\sigma(t') - \sigma\left(t' - \frac{N}{2}\right)\right)\right| \\ 0 & \text{if } \sigma(t') > \sigma\left(t' - \frac{N}{2}\right) \wedge \sigma(t) > \sigma\left(t' + \frac{N}{2}\right) \end{cases} \quad (4.62)$$

where $t' = t - 11$

The ψ_H function was applied to detection of dissolves in the training set for performance assessment. The area under the ROC curve was found to be 66.4% percent. This suggests that while analysis of the σ-curve can be informative in dissolve detection, it is not in itself sufficient. The chief difficulty is that ψ_H essentially measures the relative change in intensity variance, and the variance of frame intensity values can change when a dissolve has not occurred. This is particularly common in sports footage, where a camera pan ranges over an empty section of the playing field, or sky (as in a tracking shot in golf). To characterise a σ-curve as a dissolve or not, it would be preferable to exploit the quadratic characteristics predicted by the theory. Detecting



(a) The blue stems show individual values of $P(D|\psi_\alpha(t)$. The green line shows the fitted curve.

(b) The weights for each of the values of $P(D|\psi_\alpha(t)$ shown in blue at left.

Figure 4.24: Determining $P(D|\psi_\alpha(t)$ based on 169,000 training frames.

where a σ-curve exhibits *quadratic* change over some region could be expected to yield more reliable dissolve detection. However, characterising this quadratic behaviour is challenging, as the σ-curve can be affected by noise and, in the case of fast dissolves, is too short to be analysed reliably.

4.4 Integrated Dissolve Detection

The auxiliary function for cuts must be mapped to a conditional probability $P(D|\psi_\alpha(t))$. The erf function is used for this mapping, as in section 4.1.4:

$$P(D|\psi_\alpha(t)) = \frac{1}{2}\left(1 + \text{erf}\left(\frac{\psi_\alpha(t) - \mu_e}{\sigma_e}\right)\right) \tag{4.63}$$

Again, the parameters for the erf function, μ_e and σ_e, are determined by examination of training data. The values of $\psi(t)$ were found for over 170742 frames of video, for which the ground truth data was previously prepared. Estimates of $P(D|\psi_\alpha(t))$ for various values of $\psi_\alpha(t)$ were determined from this data, shown by the blue stems in figure 4.24 (a). These values suggested $\mu_e = 0$ and $\sigma_e = 1$ as appropriate initial parameters. As in the case of cuts, a Nelder-Mead minimisation strategy with a weighted square error function was then used to refine the parameters. Also as in the case of cut detection, the very uneven distribution of the weights made estimation of the optimal parameters unstable. The minimisation procedure was performed 100 times, using randomly perturbed initial parameters, and the best fit of all iterations accepted. The best fit curve had parameters $\mu_e = 1.694$, $\sigma_e = 1.554$; this curve is shown in green in figure 4.24 (a).

The decision rule can then be applied, such that a dissolve is declared at locations t where

$$\left.\frac{p(\delta|B)}{p(\delta|\bar{B})}\right|_{\delta=\delta_{22}(t)} > \omega\frac{1-P(D|\psi_\alpha(t))}{P(D|\psi_\alpha(t))} \tag{4.64}$$

A weight ω has been introduced to allow a weighting factor to be applied to the ψ_α function. Table 4.3 shows the performance obtained in dissolve detection for different values of ω.

Performance was evaluated on the full corpus (including training material), and also on the training material only. In both cases, the best performance (in the sense of highest mean value of precision and recall) was obtained with $\omega = 2^4 = 16$. The system obtains 85% precision and 59% recall on the full corpus, for a mean performance of 72%. On the training corpus, performance is considerably better; 89% precision and 81% recall are obtained, for a mean performance of 85%. This suggests that the distributions of dissolve characteristics found in the training data are not representative of those over the corpus as a whole. Examination of some of the dissolves included tends to support this conclusion. For example, in the training set, a number of the dissolves included are transitions where programme credits are faded in over a static background, a kind of transition much less common in the non-training data.

The system published by Hanjalic in [135] achieved 80% in both recall and precision for dissolve detection. It is noted that the test corpus used in that work contains only 23 dissolve transitions, and the video corpus used is not available for comparative evaluation. As mentioned previously, the video data and ground truth used here are the data set from the TrecVid 2002 shot boundary detection task. The performance of the participants in this project is available [275], and most systems evaluated obtain performance comparable to that reported here. The best of these systems reported mean performance values around 75%.

4.5 Conclusions and Future Work

The ideas presented in this chapter offer refinements to cut detection, and a new approach to dissolve transition analysis. For cuts, the use of an asymmetric window and a multiscale peak detection have been proposed. These are shown to be beneficial to cut detection in very difficult sequences.

The α-curve is an entirely novel approach to dissolve analysis. It is theoretically well-founded and the results described here indicate that it has considerable use as a tool for dissolve detection. The computational cost of this measure is considerably higher than that of other metrics, such as the variance of intensity values. This aspect may be ameliorated by careful selection of the areas to compute α-curves over or computing α-curves over every second frame. As computing performance continues to increase, the computational cost will become less important. It must also be noted that dissolve detection via the α-curve relies on the reliable performance of many components, including the frame similarity measure, peak finding procedure, peak width estimation, and global motion estimation. The true potential of the α-curve approach can only

	Full corpus			Training corpus		
ω	Precision	Recall	Mean(P,R)	Precision	Recall	Mean(P,R)
2^{-10}	0.638498	0.683417	0.660957	0.756598	0.895833	0.826216
2^{-9}	0.64988	0.680905	0.665392	0.762611	0.892361	0.827486
2^{-8}	0.664198	0.675879	0.670038	0.772727	0.885417	0.829072
2^{-7}	0.670854	0.670854	0.670854	0.776074	0.878472	0.827273
2^{-6}	0.685567	0.668342	0.676954	0.7875	0.875	0.83125
2^{-5}	0.698413	0.663317	0.680865	0.791139	0.868056	0.829597
2^{-4}	0.713115	0.655779	0.684447	0.796774	0.857639	0.827207
2^{-3}	0.733711	0.650754	0.692232	0.80456	0.857639	0.8311
2^{-2}	0.745665	0.648241	0.696953	0.811881	0.854167	0.833024
2^{-1}	0.764179	0.643216	0.703698	0.822148	0.850694	0.836421
2^{0}	0.780186	0.633166	0.706676	0.84083	0.84375	0.84229
2^{1}	0.786834	0.630653	0.708744	0.84669	0.84375	0.84522
2^{2}	0.794872	0.623116	0.708994	0.854093	0.833333	0.843713
2^{3}	0.814815	0.60804	0.711428	0.867159	0.815972	0.841565
2^{4}	**0.835664**	**0.600503**	**0.718083**	**0.889734**	**0.8125**	**0.851117**
2^{5}	0.857143	0.572864	0.715004	0.910931	0.78125	0.846091
2^{6}	0.865079	0.547739	0.706409	0.915612	0.753472	0.834542
2^{7}	0.870293	0.522613	0.696453	0.915929	0.71875	0.81734
2^{8}	0.876712	0.482412	0.679562	0.914286	0.666667	0.790476
2^{9}	0.897959	0.442211	0.670085	0.921466	0.611111	0.766289
2^{10}	0.914634	0.376884	0.645759	0.925926	0.520833	0.72338

Table 4.3: Dissolve detection performance over the test and training video corpus for different values of ω. The bold line indicates the value of ω achieving highest mean performance.

be assessed when all the requisite components are optimally implemented. This would require a very considerable quantity of training material, and is outside the scope of this work. The approach adopted by Lienhart [203], in which an arbitrary quantity of training data is generated by synthesising dissolves from a library of footage, would be necessary for a complete investigation.

An aspect of α-curve analysis not investigated here is how the spatial evolution of the local motion map can inform transition detection. For example, in a wipe or pan transition, the error map will be spatially coherent, while in a dissolve it would be expected to be more randomly distributed. This is left for future work.

5

Snooker Events via Discontinuities[1]

The preceding chapters have described approaches to shot change detection. This is spatially coarse discontinuity detection for video signals in general, and as such is *domain agnostic*. Such domain-agnostic systems are limited to event detection at a semantically low level.

Higher level events are those that convey information about the meaning of the video. However, it is presently not feasible to design domain-agnostic systems for high-level event detection in general. Consideration is restricted therefore restricted to some specific genre of video, such that *a priori* domain specific knowledge can be exploited. This enables *content-aware* video processing applications, in which video can be manipulated using high-level concepts such individual characters, scene settings, or narrative events.

In this chapter, a number of techniques for high level event detection and summarisation of broadcast snooker footage are discussed. These techniques form the essential components of a domain-specific system, enabling analysis of snooker video *as snooker video*. The key to this high level event detection is selection of the most salient regions of each frame—specifically, the snooker table and pockets. Feature extraction is then applied to these regions. These features are chosen so that discontinuities in the feature values correspond to events in the game.

Three new tools for analysing and summarising snooker footage are presented here, based

[1]The material in this chapter has been published as "Content-based analysis for video from snooker broadcasts" by Hugh Denman, Niall Rea, and Anil Kokaram, in *Journal of Computer Vision and Image Understanding - Special Issue on Video Retrieval and Summarization*, volume 92, issues 2-3 (November - December 2003), pages 141–306. The work was sponsored by Enterprise Ireland Project MUSE-DTV (Machine Understanding of Sports Events for Digital Television)

on identifying meaningful table views and ball movement events. The tools are a technique for effectively summarising snooker shots in the game; a means for detection of 'ball pot' events; and an approach to ball tracking suitable for event detection. Unlike previous work in sports retrieval [289], the content analysis engine *does not need to extract the 3D scene geometry* to parse the video effectively.

This chapter proceeds with a brief review of related work in semantic-level event detection for sports footage. In the following section, some prerequisite techniques for snooker video processing are presented, including localisation of the snooker table and the table geometry, and locating and removing the player. The new tools are then described.

The scope of this work is limited to the development and initial assessment of the tools described. Other researchers in this laboratory subsequently refined and integrated the techniques into a complete system, showing how they generalise to other sports with strong geometry such as tennis [241, 259, 260].

5.1 Related Work

The domain of sports video analysis has attracted considerable attention for a number of reasons. Sports footage lends itself to machine analysis in ways that film and video in general (e.g. feature films) do not. The action takes place in some well-delineated space with clearly marked features; these features can be used to situate analysis in the domain of play. Within a given sport, the variety of possible episode types is not large, and there are rigid constraints on how events can follow each other in sequence. Ultimately, because sports are governed by rules, sports footage is rule-bound to a far greater extent than other video material. These rules reduce the space of admissible interpretations greatly, facilitating machine analysis.

A second factor is the significance of sports culturally and commercially. For major sports, broadcast audiences are potentially global and are a major revenue source to broadcasters. From this perspective, technologies that can enhance or streamline sports broadcasting are highly desirable.

One of the first stages in sports video retrieval is distinguishing sports video from other kinds of video; an example of this is the work by Kobla *et al.*which detects slow motion replays (which are most commonly used in sports footage) in the compressed domain [175]. The next stage is identification of which sport is present. Killeretal *et al.* [63] have described a method using texture and colour-pair features to identify athletics, boxing, swimming, and tennis footage.

A number of systems targeting analysis of specific sports have been presented. Commonly addressed sports include basketball [267, 332], tennis [169, 252, 289, 317], soccer [96, 122, 305, 328, 339], and baseball [64, 266, 274]. All of these systems exploit the constrained structure of broadcast sports footage for parsing, event detection and summary generation. Snooker and pool have not been addressed by other researchers.

Recently a number of researchers have become interested in exploiting the features common

Figure 5.1: A typical full-table snooker shot

to all sports analysis, and have begun discussing sports analysis frameworks suitable for specialisation to a variety of related sports. Examples include the work of Zhong and Chang [352], Duan *et al.* [91], and Kokaram *et al.* [183].

The potential of sports video analysis for analysing and improving athlete performance has been recognised [196]. Many well-known sports coaches offer consulting services in which a team sends in video of training sessions and the coach offers recommendations based on the video; a number of commercially available systems aiming to offer this functionality through computer assisted parsing and assessment of sports footage have become available in recent years [74, 206, 217, 283, 284]. These are intended for use at the training ground as part of the training regimen.

5.2 Preliminaries

In this section, some preprocessing techniques necessary to summary generation and event detection are described. Firstly, the snooker footage must be parsed into individual camera shots[2] Secondly, those shots containing a view of the entire snooker table, as shown in figure 5.1, are detected. These shots are the most informative, and they are the focus of the techniques described here. A novel means for extracting the table geometry, without the use of three-dimensional perspective analysis, is then described.

Once the table is located, analysis is directed to the table surface; it is the motion of the balls in this area that constitutes the snooker game. However, as the players walk around the table, they will introduce uninformative motion in the table region. To suppress this, a means for 'player-masking' is described: replacing the player regions that occlude the table with a

[2]Snooker video contains two kinds of 'shot' entity: camera shots and snooker shots.

table-like texture, inhibiting detection of the motion of the player. Lastly, a heuristic method for identifying the initial locations of the balls in the table is presented.

5.2.1 Shot-Change Detection

Shot cut detection is accomplished using the normalised bin-to-bin histogram difference method, in the YUV colourspace. The normalised histogram of the Y component of frame F, H'_{Y_F}, is computed according to

$$H_{Y_F}(i) = \sum_{Y_F=i} 1,\ i \in \{0...255\} \tag{5.1}$$

$$H'_{Y_F}(i) = \frac{H_{Y_F}(i)}{\sum_i H_{Y_F}(i)} \tag{5.2}$$

The U and V histograms are computed in an analogous fashion. The bin-to-bin histogram difference between successive frames F, $F+1$ is

$$\sum_i \left| H'_{Y_F}(i) - H'_{Y_{F+1}}(i) \right| \tag{5.3}$$

for the Y component, and similarly found for the U and V components. If any two bin-to-bin differences exceed 0.5, a shot change is deemed to have occurred.

Compared to the shot change systems described in the previous two chapters, this is a relatively straightforward approach. It is found to work effectively in detection of the simple transitions typical of broadcast snooker footage.

5.2.2 Snooker Table Detection

Once the shot boundaries have been found, the first frame of each shot is analysed to determine whether it contains full-table footage. The frame is converted from the YUV colourspace into RGB, made up of red (R), green (G), and blue (B) components, and a mask corresponding to the green area of the frame is generated. The mask is a binary image M generated by the conjunction of the two binary images

$$M(\mathbf{x}) = \begin{cases} 1 & \text{if } (G(\mathbf{x}) - R(\mathbf{x})) > \tau \wedge (G(\mathbf{x}) - B(\mathbf{x})) > \tau \\ 0 & \text{otherwise} \end{cases} \tag{5.4}$$

For component values in the range [0..255], a threshold of $\tau = 25$ is used. This threshold was found empirically and has proved suitable for footage captured from a variety of sources. The table mask is then convolved with a Gaussian blurring kernel, to remove interlacing and noise artifacts. An example of a mask generated by this method is shown in figure 5.2 (a).

The edges in the table mask are found using Sobel edge detection, as shown in figure 5.2 (b). It is then necessary to determine the dominant straight lines in this edge map, to find the table boundaries. These lines are detected using the linear Hough (Radon) transform. Briefly, the Hough transform maps each point in the edge map to a set of accumulators corresponding

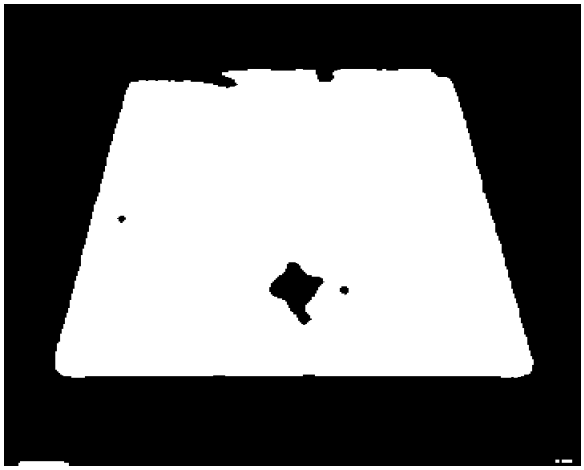
to all the lines in the parameter space which pass through that point. For each nonzero point in the edge map, all the corresponding accumulators are incremented. Peaks in the parameter space then correspond to lines in the image. The parameter space ranges over ρ, the distance of the line from the origin, and θ, the angle the line makes with the x-axis; the equation of a line in this formulation is $y\sin(\theta) + x\cos(\theta) = \rho$.

For the purpose of table detection, θ is restricted to the the regions of parameter space where the table edges are likely to be found. These are angles in the ranges [3..25], [89..90], and [155..177]. Restricting θ in this way increases computation speed considerably. Lines in the configuration corresponding to a snooker table create a characteristic pattern in the transform space: one peak in the range $\theta \in [3..25]$, one in the range $\theta \in [155..177]$, and two in the range $\theta \in [89..90]$. If peaks in this configuration are found, the shot is selected as a full-table footage shot. The Hough transform of a table edge map is shown in figure 5.2 (c). The four peaks in the transform space are clearly visible. The four lines corresponding to these peaks are shown superimposed on the table in figure 5.2 (d).

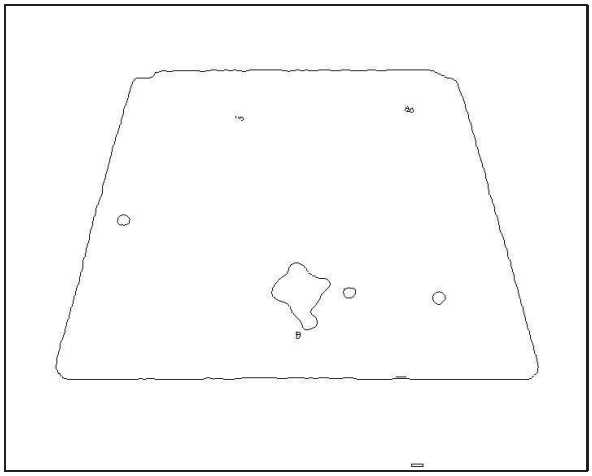
Figure 5.3 shows how this method deals with a variety of snooker video frames. Figures 5.3 (a)-(d) show the frames being processed; figures 5.3 (e)-(h) show the green edges of each frame, and figures 5.3 (i)-(l) show the Hough transforms of the green edges of each frame. The characteristic pattern corresponding to a snooker table is detected in figures 5.3 (i) and (l), and not in figures 5.3 (j) or (k).

The implementation described here uses a threshold on the Hough transform image in the regions expected to contain peaks to determine if the pattern characteristic of a snooker table is present. This is found to be effective against a number of footage clips.

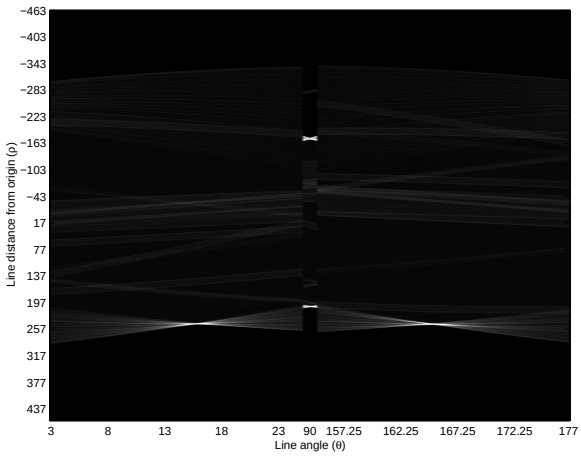
Many sports such as tennis, snooker, badminton, and cricket, occur within predefined playing limits. Most of the video footage from these events contains well delineated field lines in those views which contain the most information about the play - for example, the court lines in tennis, and the edge of the table in snooker. Because of this, the Hough transform is a powerful tool for characterising shots of interest in sports footage, as it encapsulates the geometry of the camera view. In fact, Hough transform features can be used to characterise the camera view of every frame in sports footage, and thus applied to shot change detection as well as view detection. This immediately exploits the context of such sports video and is a more powerful approach than the generic use of histogram based shot cut detection. The importance of each shot for summary purposes can be assessed based on the geometry of the view. For instance, in both tennis and snooker, shots of the crowd and of the players can be considered less important than shots containing game events, and summarised simplistically, or discarded entirely. These ideas are developed in a system described in [259], building on the work described here. Here Hidden Markov Models are used to analyse the evolution of the second order moment of the linear Hough transform of each frame. It is shown that unified shot change detection and view identification can be effectively achieved by these means.



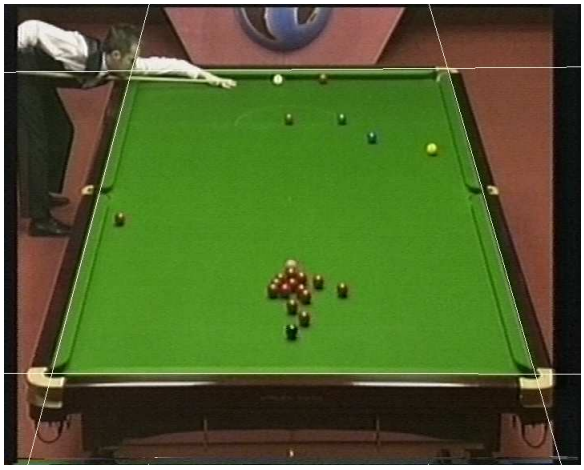
(a) $M(\mathbf{x})$, the green region of the frame.



(b) Sobel edge detection.



(c) The restricted-domain linear Hough transform.



(d) Lines found in (c) overlaid on the original image.

Figure 5.2: Steps in the table finding algorithm. The disposition of the four peaks in (c) is characteristic of lines corresponding to a snooker table.

5.2.3 Table Geometry

Having determined that a given shot contains full-table footage, the next stage is to find the frame-relative table geometry—i.e. the locations in the video frame corresponding to the ball spots and the pockets. This relies on knowledge of the real-world dimensions of a championship snooker table, as shown in figure 5.4. Various approaches to this problem are possible. As the table is distorted by a perspective projection resulting from the camera angle in full-table footage, attempting the inverse perspective transform is an obvious candidate. However, this is not practical: the inverse perspective transform is an ill-posed computation, especially when only one view is available.

Here a new method is proposed relying only on detection of the table. First, the corner

(a) (b) (c) (d)

(e) (f) (g) (h)

(i) (j) (k) (l)

Figure 5.3: Examples of table finding. The top row shows original frames from the snooker footage. The middle row presents the edges found in the green mask. In the bottom row, the Hough transforms of these edge maps are shown. The characteristic four-peak pattern is present only in those frames containing the full-table view.

pockets are located; these are at the points of intersection of the edges of the table. The use of the Hough transform space to find the edges gives direct access to the parameters ρ and θ for each edge. The point of intersection (x, y) for any two lines (θ_1, ρ_1), (θ_2, ρ_2) can be found using

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} \cos(\theta_1) & \sin(\theta_1) \\ \cos(\theta_2) & \sin(\theta_2) \end{bmatrix}^{-1} \begin{bmatrix} \rho_1 \\ \rho_2 \end{bmatrix} \tag{5.5}$$

It remains to find the other points of interest on the table. A key point here is that the perspective distortion typical to full-table snooker shots consists of foreshortening along the vertical axis. Relative distances along the horizontal are not affected. The perspective distortion is dealt with in this work by exploiting a perspective-invariant geometric property. The invariant used is that the diagonals of a trapezoid intersect at the midpoint of the trapezoid, irrespective of perspective transformation. This is a novel approach to geometry localisation.

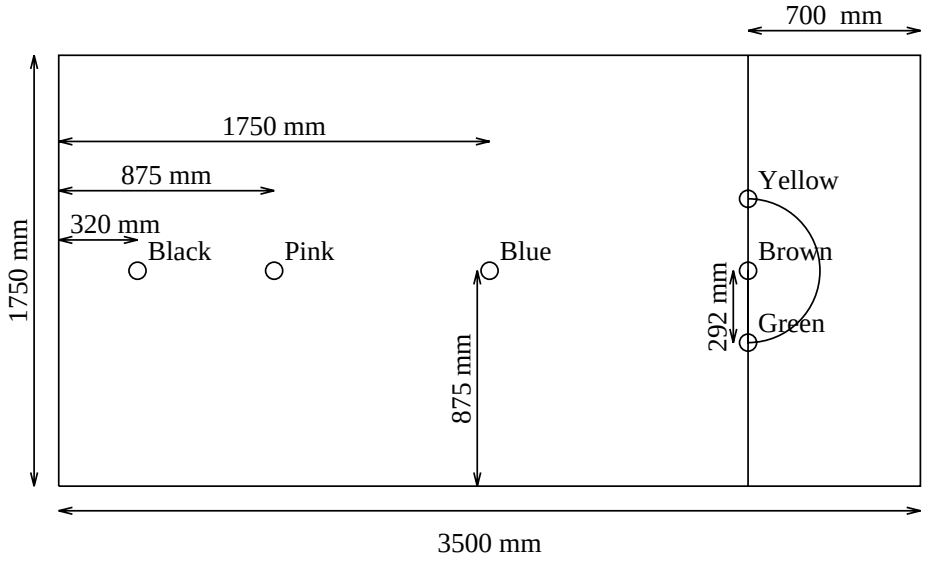


Figure 5.4: Championship snooker table dimensions

The invariant can be used to find the real-world midpoint of any trapezoid in frame-space—such as the table. Furthermore, the table can be repeatedly subdivided vertically to find any vertical position. For example, the black spot is 320 mm up from the bottom edge of the table, whose total length is 3500mm. 320/3500 expressed in binary is approximately 0.0001011101, so $320 = 3500 * (2^{-4} + 2^{-6} + 2^{-7} + 2^{-8} + 2^{-10})$. Each of these negative powers of two can be found by division along the vertical dimension of the table. In practice, the position of the black ball is adequately approximated with 0.00011, or 3/32 along the vertical. Repeated subdivision along the vertical yields the positions 1/8 and 1/16 of the way up; subdivision between these two points yields the black spot approximation, 3/32 of the way up.

As perspective projection does not affect horizontal proportion, the table dimensions can be used directly to find horizontal positions. For example, the black, pink, blue, and green balls all lie exactly half-way between the vertical table edges, despite being at varying depths.

An example of a successfully extracted table geometry is shown in figure 5.5. While the spot positions are not exact, they are quite close enough for the purpose.

5.2.4 Player Masking

Event detection in snooker footage is essentially concerned with the movements of the balls. In general, motion of the balls can be detected simply by performing motion detection on the table region of the video. However, where the player is obscuring some portion of the table, it is difficult to distinguish ball motion from player motion. For this reason, a technique is developed here whereby the region of the table obscured by the player can be identified and filled-in with a synthesised table-like texture. This simplifies the initial detection of the balls, and suppresses

Figure 5.5: Table geometery extraction.

any motion due to the player.

In snooker, the players' clothing typically uses colours darker or brighter than the table, such as black and white. Similarly, the snooker cue is composed exclusively of dark and bright regions. The table is found to have a moderately high brightness. Thus extreme intensity values are more likely to correspond to the player than the table. Regions with extreme intensity values that are connected to the edge of the table are made part of the player mask:

$$P(\mathbf{x}) = \begin{cases} 1 & \text{if } (M(\mathbf{x}) = 1) \wedge (I(\mathbf{x}) < \tau_1 \vee I(\mathbf{x}) > \tau_2) \\ 0 & \text{otherwise} \end{cases} \tag{5.6}$$

where $I(\mathbf{x})$ is the intensity component of pixel $\mathbf{x}$. τ_1 is set to 100 and τ_2 set to 180; these values have been found effective for footage captured from a a variety of sources. Any region in this map touching the edge of the table is considered to be the player, and filled with a table-like texture. Some examples of successful player masking are shown in figures 5.6.

Some areas of the table have very low intensity, including the pockets and the shadow under the lip of the topmost table edge. This shadow can cause problems when the player is leaning over it and there are balls near the top of the table. In this circumstance, the balls will be part of a region have extreme intensity, connected to the edge of the table by the shadow and the player, and thus will be add to the mask. Such a situation is shown in figure 5.6 (c); both the white ball and the targeted red would be lost if the mask were made on the basis of extreme intensity values alone.

To address this, balls near the edge under the upper lip are detected and removed from the

player mask. First, the shadow at the top of the table is found; all dark regions close to the top of the table are considered part of this shadow. Secondly, this shadow region is segmented into table and non-table regions, on the basis of hue values. The most commonly occurring hue values in the frame will be those of the table. To find the range of hue values that correspond to the table, $H_1 \ldots H_2$, the 255-bin normalised histogram of hue in the frame is computed. The largest peak in this histogram will correspond to the table; starting at this peak, the range of hue values is expanded until 75% of the pixels is accounted for. This expansion is done according to a greedy algorithm: the range is expanded, either to the left or the right, so as to incorporate the larger of the two adjacent bins. The resulting peak bounds are H_1 and H_2, the upper and lower limits of the table hue range. Regions in the shadow mask having a hue outside this range are considered balls, and removed from the player mask.

The area corresponding to the player mask is filled with a green texture similar to the table. The fill-in texture is generated by finding a region of the table with no high frequency components. To find this region, the table region is filtered using a differential filter $[-1, 0, 1]^T.[-1, 0, 1]$, and a square region 20 pixels on a side is found where the difference values are all less than 15. It is assumed that this region contains only empty table and is free of balls or holes. Each point in the player mask is filled in by randomly sampling the pixel values in this region.

5.2.5 Initial localisation

Once the table and player regions have been identified, the next step is the identification and localisation of the individual balls. This is achieved using analysis of the first frame of each shot. The first step is a first-approximation segmentation using the watershed algorithm. An example of a such a segmentation is shown in figure 5.7 (a).

Next, objects that are too small (containing fewer than twenty pixels) or too large (containing more than 100 pixels) are discarded. Objects that are long and thin are also discarded, i.e. where the ratio of the long side of the object bounding box to the short side is greater than three. The frame after these tests is shown in figure 5.7 (b). The binary objects found are then classified based on their relative colour properties.

First, the three brightest balls are found. These are the pink, yellow, and white balls. Of these balls, the yellow ball has the highest saturation. Generally, the pink ball will have a higher hue than the white, so these two can be distinguished on that basis.

The black ball is the ball with the lowest median brightness. The green and the blue balls will have the highest hue values, with the blue having the higher hue value in most cases. The remaining balls are the brown and the red balls. While the brown ball has a slightly lower hue value, it is difficult to reliably distinguish these on the basis of their colour properties.

Leaving the brown / red distinction aside, this simple initialisation scheme is found to work well in a variety of footage; two examples of successful ball finding are shown in figures 5.7 (c) and (d). Not considered here are the further cues for object localisation which would become

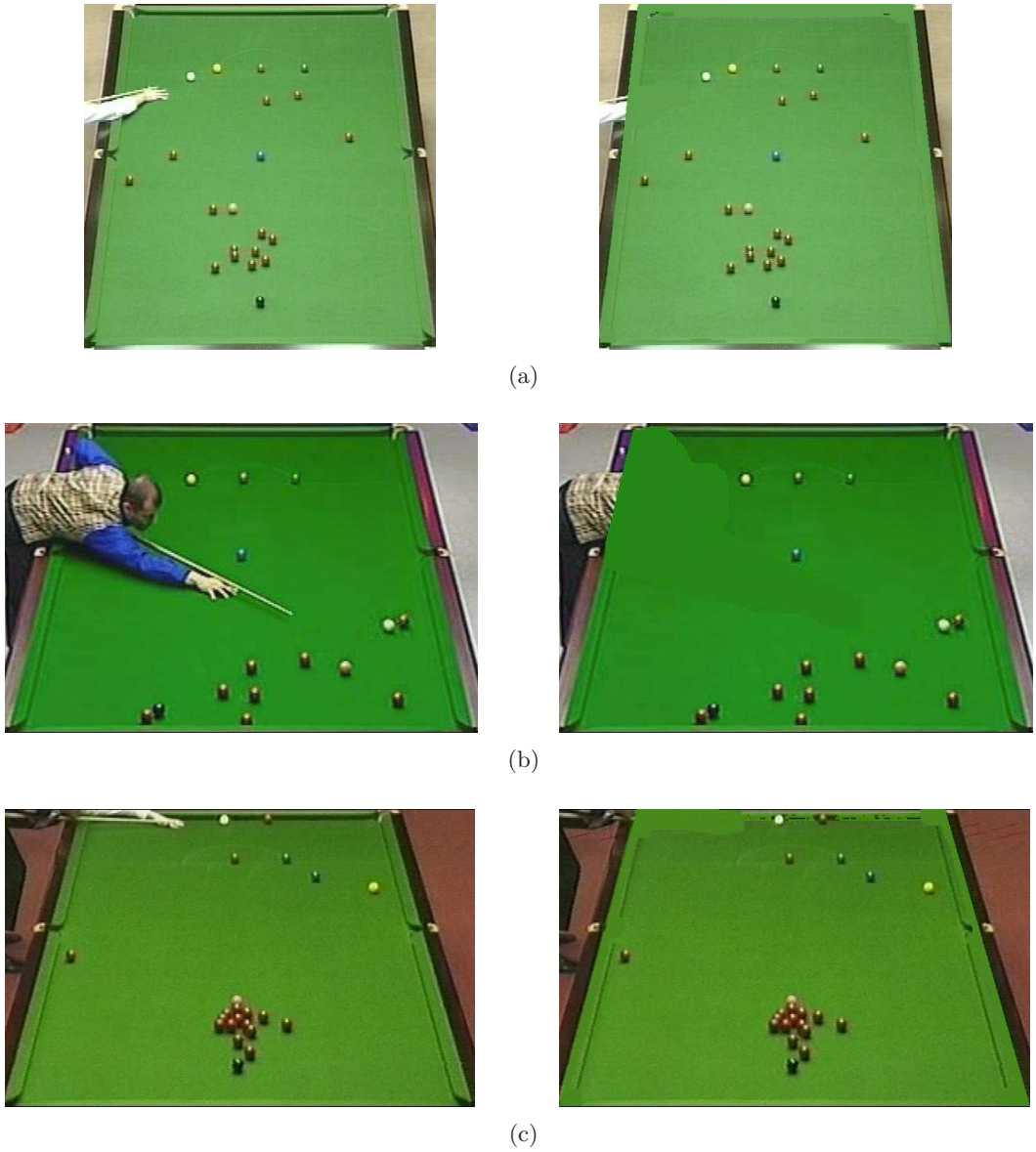
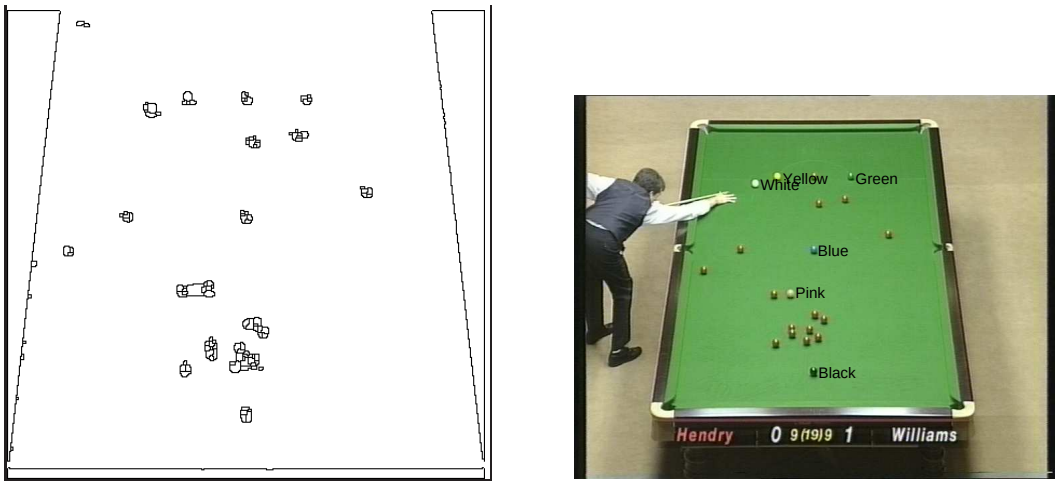
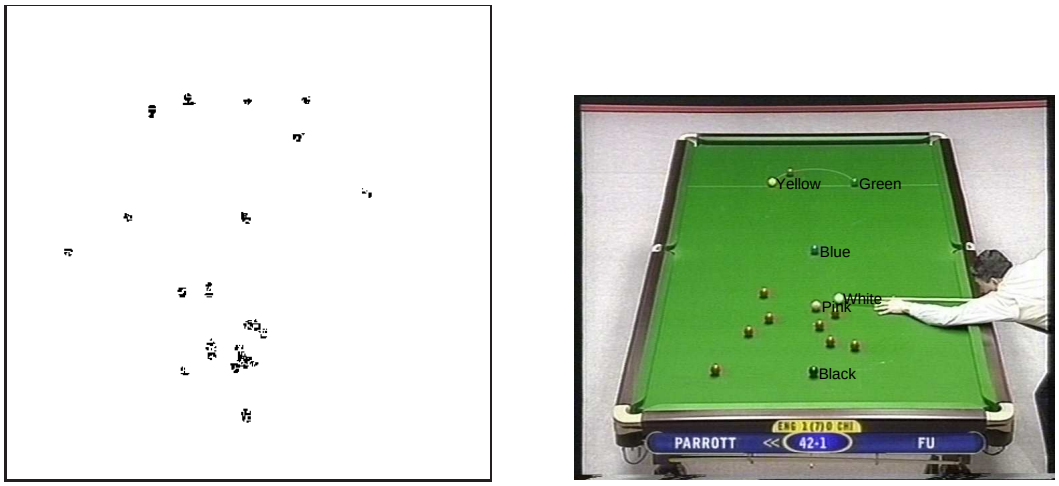


Figure 5.6: Three examples of player masking.

available if the analysis techniques described here were integrated into a single system. For example, at the start of play the balls are located at their 'spot' positions. Where player masking is successful, the white ball will usually be a bright object close to the player mask; this could be used to assist in pink / white disambiguation. Ball tracking, described below, can be used to update an estimate of ball position from one shot to the next. Finally, the game state can also provide cues. For example, if the player should be shooting at a red, then the ball the white first collides with is probably not the brown. Some of these higher level aspects have been developed in [259].



(a)



(b)

Figure 5.7: Two successful examples of ball finding. The watershed segmentation is shown on the left, and the ball labels are shown on the right. Note that the red and brown balls are not distinguished.

5.3 Semantic Applications

5.3.1 Clip Summaries

A clip of snooker footage generally shows a single snooker shot: an event starting with the contact of the cue with the white ball, and lasting until no balls are moving. In this section, a means for summarising clips showing a snooker shot in a single frame is presented. This is a novel application of the the Motion History Image (MHI) [32,33], originally proposed by Bobick

and Davis for gesture recognition. The MHI of frame t is defined by

$$\mathrm{MHI}(\mathbf{x},t) = \begin{cases} M & \text{if } |I(\mathbf{x},t) - I(\mathbf{x},t-1)| > \tau_m \\ \max(0, \mathrm{MHI}(\mathbf{x},t-1) - 1) & \text{otherwise} \end{cases} \tag{5.7}$$

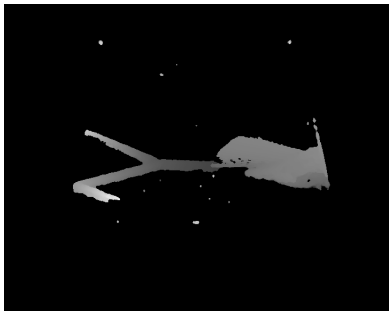
where $I(\mathbf{x},t)$ is the intensity value of pixel site $\mathbf{x}$ at time t, τ_m is a motion detection threshold, and M is a large constant. High values of the MHI correspond to recent motion. The value M is chosen to determine how long the effects of motion persist. A sample MHI is shown in figure 5.8 (a). Shot summaries can then be generated by averaging the first and last frames of the shot, as shown in figure 5.8 (b) and overlaying the MHI on this composite. In order that the the summary trace should not obscure the positions of the balls in the ghosted composite, regions of the MHI corresponding to very early or very recent motion are suppressed. This ensures that the original and final positions of the balls are visible in the final summary image. An example result is shown in figure 5.8 (c). The value of M is chosen to be equal to the length of the clip being summarised.

The motion of the player can introduce significant 'motion clutter' in the MHI–i.e. undesired motion regions which obscure the motion of the balls. An example of this is shown in figure 5.8 (d). Applying the player masking algorithm described above reduces this motion clutter considerably, as shown in figure 5.8 (e).

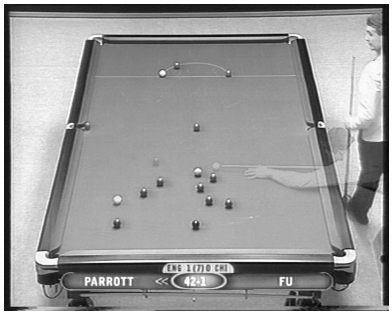
An entire video shot, as defined by the result of shot change detection, often includes footage before the snooker shot begins. This footage typically consists of the player walking around the table, considering angles and ball positions, and it is not desired to include this footage in the clip summary. Using player masking enables the isolation of the footage containing the actual snooker shot. Summary generation is not initiated until motion is detected in the table region—which, when player masking is in use, will be due to the commencement of ball motion. When detecting the start of ball motion, motion regions containing fewer than 5 pixels are assumed to be due to noise. Summary generation begins only when a motion region in the table area contains more than five pixels. Similarly, summary generation is terminated at the end of the shot, or when no significant motion is detected in the table region.

The techniques described above for refinement of the summary image can only be applied where the footage contains the full table view considered for table detection and player masking. However, the MHI can still be used to generate effective summaries of clips containing other camera views in snooker footage. Figure 5.9 shows an MHI-based summary of 2421 frames of a snooker game. In the bottom-left image, camera motion (here zoom) has introduced motion clutter. This problem could be alleviated by using global motion estimation and compensation.

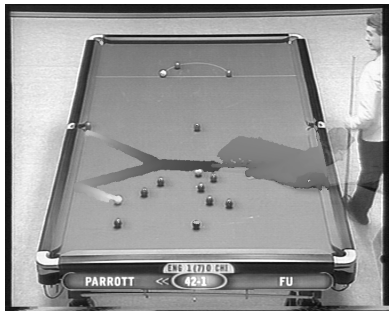
Summary images 4 and 9 illustrate the result of event detection; ball pot events have been automatically highlighted with a red circle is overlaid on the target hole. In image 4, the motion track of the white ball is overlaid on the MHI summary image. These two aspects of the summary generation system are described next.



(a) MHI image



(b) Composite of first and last frames



(c) Summary image: MHI overlaid on composite

(d) Summary image generated without player masking

(e) Summary image generated with player masking

Figure 5.8: Use of the MHI and player masking to generate summary images for snooker shots.

5.3.2 Pot Detection

The most important events in snooker are ball-pot events, and in this section a method to detect these events is described. The method exploits the extracted in-frame geometry to define regions of interest around each of the pockets of the snooker table. These regions can be monitored to detect *object disappearances*. When the player masking described earlier is in use, the only moving objects on the table are the snooker balls. When an object disappears in the vicinity of a pocket, it can safely be inferred that a 'ball-pot' event has occurred.

Two regions around each pocket are monitored, for a total of twelve regions in total. At each pocket, a small, inner region and a larger region encompassing the smaller are defined. The regions must be large enough that fast moving balls cannot traverse them in less than 1 frame time while in motion. The sizes used are 1/15 of the table width on a side for the small regions, and 1/8 of the table width on a side for the large regions. Figure 5.10 shows the locations and sizes of these regions on a snooker table.

Having defined the regions, the next stage is to detect when objects enter and leave these

Figure 5.9: MHI summary of 2421 frames of broadcast snooker footage.

regions. This is achieved using detection of non-table pixels within in region. Let T_1 be the frame time at the start of the first shot containing a full-table view. Then $I(\mathbf{x}, T_1)$ is the intensity image at the start of the shot. It is assumed that no balls are in any of the regions at this time. The detection of non-table pixels within each region at frame t, $M_R(t)$, is defined by

$$M_R(t) = |\{(\mathbf{x} \in R) \wedge (|I(\mathbf{x}, t) - I(\mathbf{x}, T_1)| > \tau_m)\}| \tag{5.8}$$

where $R \in (1 \ldots 12)$ is the region under consideration. The difference threshold τ_m is set to 30 intensity levels. As pixels corresponding to the player are replaced with table-like values via player masking, changes in $M_R(t)$ are assumed to be generated by ball motion. Peaks are generated in the traces by balls entering and leaving the regions.

The start and end points of peaks in the $M_R(t)$ correspond to balls entering and leaving the region. Comparing the ball entry and exit times between the large and small regions for each hole can be used to detect pots and near misses. As a ball approaches a hole, it will enter the

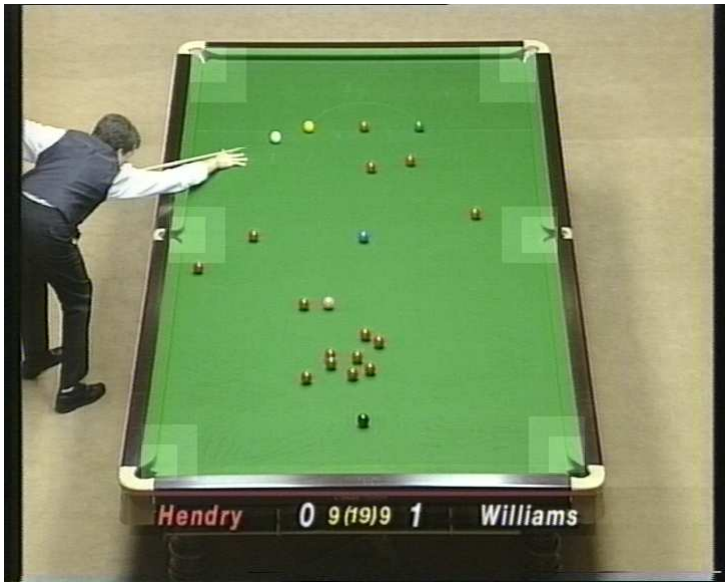


Figure 5.10: Regions monitored for ball potting detection.

large region first, and then subsequently enter the small region. If the ball is potted, it will be detected leaving both regions simultaneously. If it leaves the small region first, and then the large, it has bounced back from the cushion and has not been potted. If it enters the small area but does not leave, it has stopped moving in close proximity to the hole. These latter two scenarios are 'near miss' events, of interest in themselves in summary generation.

To detect the event scenarios described above, thresholding is applied to the traces for each region. At least 20 pixels within the region must be 'non-table' for the motion to be considered significant. Values greater than 80 in the trace indicate that nearly all the pixels are non-table, suggesting that the object in the region is too large to be a ball. Such large values are usually caused by a failure in the player masking algorithm, and so these values are excluded. The thresholded signal is then

$$\hat{M}'_R(t) = \begin{cases} 1 & \text{if } 20 < M_R(t) < 80 \\ 0 & \text{otherwise} \end{cases} \tag{5.9}$$

This signal is then filtered with a three-tap median filter, to remove the effects of noise.

$$\hat{M}_R(t) = \text{median}(\{M'_R(t-1), M'_R(t), M_R(t+1)\}) \tag{5.10}$$

Figure 5.11 shows how the traces for each region evolve for a pot and a near miss. In the images on the left, the blue ball is potted into the left center pocket. The purple traces in the graphs on the left show the non-table pixel count for the large and small regions around this pocket. The ball enters the large region at frame 20, and the small region at frame 25. It leaves both regions simultaneously, at frame 33. These events are reflected in the start and end points of the purple peaks, and thus the pot is detected.

The images on the right of figure 5.11 illustrate detection of a near miss, in which a red is almost potted to the bottom left pocket. The ball enters the large area in frame 17 and the small area in frame 18. It leaves the small area in frame 25, and then the large area in frame 27. A near miss is flagged. Both traces return to zero briefly at frame 22. This is due to a failure in the player masking algorithm, and is dealt with by median filtering.

The event detection algorithm was applied to a 16-minute game of snooker, consisting of 24250 frames and including 14 pots. All pots were selected correctly, along with 5 false alarms, for 100% recall and 74% precision. The false alarms were caused by failures in the player masking algorithm. The same footage contained four near misses, of which three were found, with no false alarms: thus 75% recall and 100% precision for near miss detection. The missed 'near miss' event was again due to a malfunction of the player masking algorithm.

5.3.3 Explicit motion extraction

MHI summary images convey motion information implicitly, in a form suitable for a viewer. This representation is not suited to machine analysis of ball motion, as it does not explicitly describe information regarding how many balls are moving; which balls are moving; whether balls have collided; and the position of the balls at each moment in time. Information at this level can only be extracted using explicit tracking to establish the trajectories of balls in motion.

Object tracking is achieved here using a colour-based particle filter. It is noted that a similar tracking system has been presented by Nummiaro *et al.* [242]; the system described here was developed independently [81]. The underlying algorithm is called *condensation* [159] (for **con**ditional **dens**ity propag**ation**), and can be thought of as a non-parametric Kalman filter. In outline, the algorithm operates as follows. Let $p(\mathbf{x}_t)$ be the time-varying distribution of interest (here the position of the ball at time t), and $\mathbf{z}_t$ be observations at time t (here colour values at particular pixels). By Bayes' law

$$p(\mathbf{x}_{t+1}|\mathbf{z}_{t+1}) \propto p(\mathbf{z}_{t+1}|\mathbf{x}_{t+1})p(\mathbf{x}_{t+1}|\mathbf{z}_t) \tag{5.11}$$

Here $p(\mathbf{z}_{t+1}|\mathbf{x}_{t+1})$ is the likelihood of observing $\mathbf{z}_{t+1}$ at time t given that the ball is at position $\mathbf{x}_{t+1}$. The prior distribution, $p(\mathbf{x}_{t+1}|\mathbf{z}_t)$, is generated according to

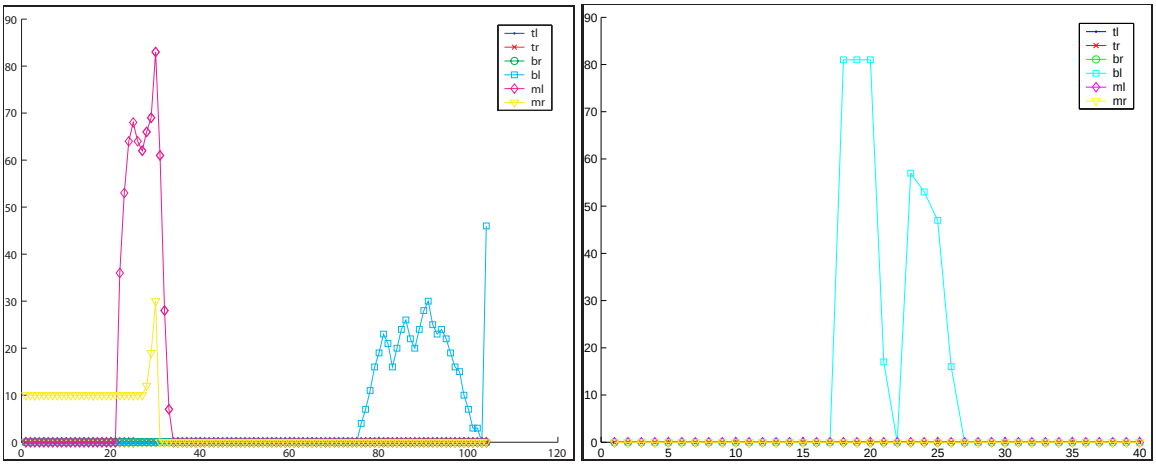
$$p(\mathbf{x}_{t+1}|\mathbf{z}_t) = \int p(\mathbf{x}_{t+1}|\mathbf{x}_t)p(\mathbf{x}_t|\mathbf{z}_t) \tag{5.12}$$

The distribution $p(\mathbf{x_t}|\mathbf{z_t})$ is represented by N samples $\mathbf{s}_t^i$, $i \in 1 \ldots N$. Each sample has an associated weight π^i, corresponding to the likelihood at that location $P(\mathbf{z}|\mathbf{x}_t) = \mathbf{s}_t^i)$. At each iteration, the samples are updated according to a linear model

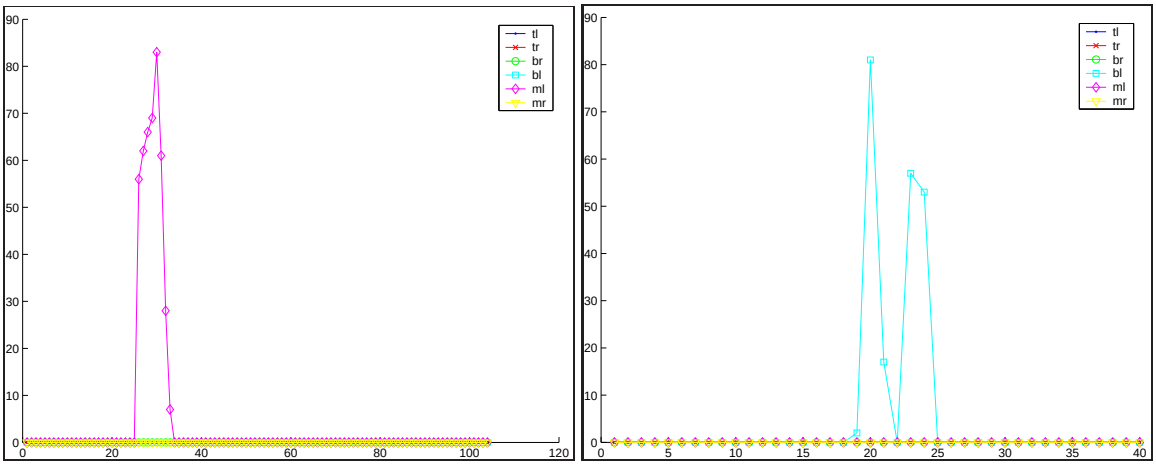
$$\mathbf{s}_{t+1}^i = \mathbf{s}_t^i + (\bar{\mathbf{x}}_t - \bar{\mathbf{x}}_{t-1}) + \epsilon, \text{ where } \epsilon \sim \mathcal{N}(0, \sigma) \tag{5.13}$$

where $\bar{\mathbf{x}}_t$ is the weighted mean of the samples at time t, and σ introduces a random scattering of the particles.

(a) MHI summary image of shot



(b) $M_R(t)$ for large regions



(c) $M_R(t)$ for small regions

Figure 5.11: Event detection via region monitoring. The images on the left show a pot. Those on the right show a near miss.

At each stage the distribution of colour values corresponding to the ball is represented by a Gaussian distribution for each of the H, S, and V channels. Where $\bar{H}_t$, $\sigma^2_{H_t}$ are the mean and variance of the distribution of hue values of the ball, and h, s, v are the values observed in $\mathbf{z}$, the likelihood is

$$P(\mathbf{z}_t|\mathbf{x}_t) = \frac{1}{3}\left(\frac{\bar{H}_t - h}{\sigma^2_{H_t}} + \frac{\bar{S}_t - s}{\sigma^2_{S_t}} + \frac{\bar{V}_t - v}{\sigma^2_{V_t}}\right) \tag{5.14}$$

These distributions are updated at each iteration based on the sample set:

$$\bar{H}_{t+1} = \frac{1}{\sum_i \pi^2}\sum_i H(\mathbf{s_t^i})\pi^i \tag{5.15}$$

$$\sigma_{H_{t+1}} = \frac{1}{(N-1)\frac{1}{N}\sum_i \pi^2}\sum_i \pi^i(\bar{H}_{t+1} - H(\mathbf{s_t^i}))^2 \tag{5.16}$$

$$\tag{5.17}$$

The tracking algorithm will sometimes fail to locate the ball in a particular frame. This is the problem of 'loss of lock'. Failure to locate the ball can be diagnosed by examining the sum of the particle likelihoods; if the sum is less than a threshold, the new position of the ball has not been located. In this case, the scatter parameter σ can be increased according to some schedule, and the particles redistributed in a widening pattern until the ball is found. If the ball is not found after several iterations of this widening procedure, it is assumed not to be on the table, and thus potted. This assumption is reinforced if the last known location of the ball is near a pocket.

Equation 5.13 represents a semi-deterministic update of the ball position, assuming that the ball motion is essentially constant. Incorporating this update greatly speeds acquisition of the ball from frame to frame. A more complicated motion model that detects when the ball is about to hit the table edge could be used, to automatically incorporate bounces as a reflection in the horizontal or vertical velocity. The linear model described was found to deal with bounces in all the test footage, so the more sophisticated model was not deployed.

Two examples of successful ball tracking using this method are shown in figure 5.12. Video examples of this tracking algorithm are presented in the accompanying DVD.

Figure 5.13 illustrates a preliminary approach to extracting semantic information from the results of object tracking. The bounces of the white ball can clearly be seen in the position and speed graphs, and the position of the ball at the time of each bounce is known. All three bounces are at a table edge, so it is likely that in this track, the white ball has not hit any other ball. The speed of the white ball decreases to zero in the clip, so it can be assumed that this video clip represents the complete snooker shot. Thus, there is a higher than normal probability that this shot is a foul (as it is a foul in snooker not to strike a coloured ball with each shot).

(a) White ball track superimposed on snooker frame. (b) Yellow ball track superimposed on snooker frame.

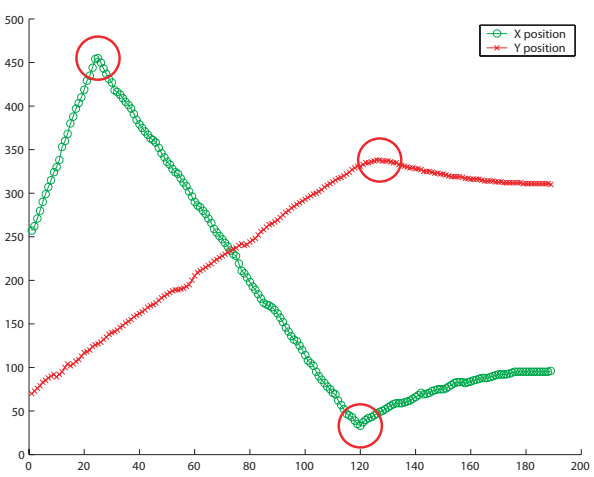
Figure 5.12: Ball tracking via condensation

5.4 Conclusion

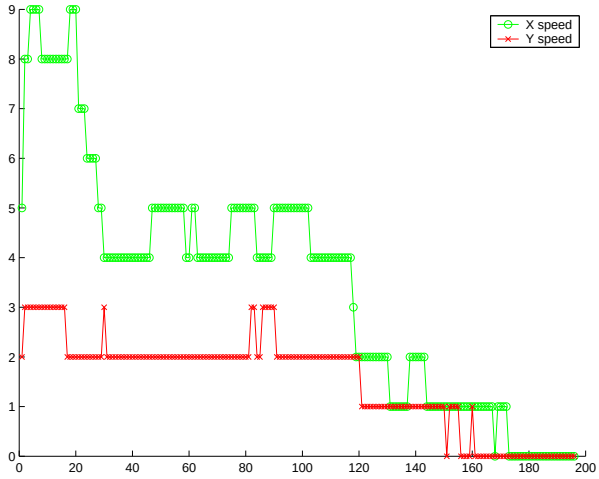
The work described in this chapter offers powerful tools for summarising sports footage, and a basis for machine understanding of snooker video. It is useful to note that manually generated summaries of sports in general simply do not exist in this form. Sports summaries are typically shortened video footage, and textual annotation can take 10 hours for 1 hour of summary generation [63]. The computational load of the tools presented here is not high. Furthermore, as the techniques are designed for batch use, running entirely unattended, performance optimised implementation of these tools has not been necessary; many of the tools are implemented in Matlab. Aside from player masking, the tools, run in cascade on a single 1 GHZ PIII, take about 3 hours to summarise 16 minutes of footage. Player masking introduces significant computational load; this can take up to 7 seconds per frame, principally because of its use of morphological operations such as dilation.

Three new ideas for sports summarisation have been presented here. The most important of these is that 3D information is not necessary to summarise footage according to geometry. Using in-image analysis, Hough transform analysis, and a-priori knowledge about diagonal intersections allows a great deal of content-rich information to be extracted from the footage. The results presented show successful detection of table view shots and ball-potting events, and tracking of ball trajectories. The use of the MHI is a novel concept in the generation of content rich summary key frames. Further research undertaken subsequent to that described here has shown that the use of the Hough transform to characterise frame geometry, and the use of the MHI to generate summary images, are particularly applicable to a wide range of court sport events.

(a) Snooker frame with white ball track superimposed.



(b) X and Y components of white ball position. Bounce locations are indicated by red circles.



(c) X and Y components of white ball velocity.

Figure 5.13: An example of a foul, in which the white ball does not strike a colour. The two graphs are of white ball position and speed. Observe that bounces are clearly visible in the graphs.

6

Low-level Edit Point Identification [1]

In this chapter ways to exploit low level video characteristics for the detection of edit points are considered—in particular, edit points in dance video suitable for synchronisation with a beat event in a stream of music. The term 'low-level' is employed here as a characteristic of algorithms that operate with no concept of an object. In the methods presented here, no attempt is made to explicitly segment the video stream into its constituent objects. Video object segmentation is considered in the next chapter. In this chapter, implicit characteristation of the motion content of the scene is sought, in the interest of robustness and more general applicability. Several of the techniques presented rely only on motion *detection*, which is a particularly low level of image sequence analysis, but has the advantage that it can be robustly applied in many more situations than more high level processes.

Edit points in a video stream arise predominantly as a result of the motion content of the video. Therefore the techniques described in this chapter rely on low-level characterisation of the motion in each frame of the video under consideration. However, meaningful edit points are always generated by the motion of a specific object. The approaches described below will thus be most successful in footage where the motion content over the entire spatial extent is representative of the motion of an object—for example, when only one object is depicted in the video. For example, consider pieces of footage which portray a dancer performing in front of a relatively homogeneous background; motion detection applied to such streams can reveal points

[1] An early version of this work was described in "Dancing to a Different Tune" by Hugh Denman and Anil Kokaram, in *Proceedings of the IEE European Conference on Visual Media Production (CVMP'05)*, pages 147–153, London, November 2005

where the dancer stops or starts suddenly. These edit points are instances of *percussive motion*, a term introduced here to describe movements in dance that are analogous to beats in music.

The first section of this chapter presents some examples of such target edit points, along with a short discussion outlining the difficulty of automatically assessing edit point detection. This is followed by a description of a method for locating edit points based on motion detection. Within this section, techniques for parsing a signal, or *trace*, into peaks, and classifying these peaks as desired (corresponding to an edit point) or undesired, are developed. There then follow several sections describing other low-level signals for edit point detection. These analysis of the DFD bounding box (introduced in chapter 2), local motion vector analysis, motion blur detection, and audio energy analysis. The same approach for parsing a signal into peaks and then classifying the peaks is applied to each of the resulting low-level traces. A framework for probabilistic fusion of these signals is then described. The chapter concludes with an assessment of the techniques presented.

6.1 Instances of Percussive Motion

In order to consider resynchronising footage of a dance performance to a new music track, it is first necessary to describe what aspects of a dance video give rise to the percept of synchronisation in the original footage. For many conventional forms of dance, the movement of the dancer can be viewed as consisting of a succession of phrases. These phrases are presented in succession, in time to the accompanying music; in particular, dance phrases generate a compelling aesthetic effect when they start and end simultaneously with beats in the music.

The key enabling assumption for this work is that there are some intrinsic characteristics particular to the movement between phrases. The term *percussive movement* is introduced as characterising this kind of motion. This term is chosen to reflect the supposition that the function of these particular motions is to imply or reinforce the perception of rhythm, similar to the role played by percussion instruments. Typical examples include a sudden change of direction of a limb, or the stamping of a foot. Among the most salient percussive motion events are those in which the motion of the dancer suddenly stops or starts. The focus of this chapter is therefore on the identification of these local motion discontinuities.

6.1.1 Difficulties in Edit Point Localisation

The chief difficulty in assessing tools for edit point detection lies in the lack of a well-specified ground-truth for a given sequence. A number of factors making establishing a ground truth a difficult problem are now described. Each factor is illustrated by video material on the accompanying DVD.

Firstly, edit point selection is essentially a subjective judgment. While the larger gestures of a dance can be clearly demarcated, many more subtle movements could be considered either as

dance phrases in their own right, or as constituent parts of a larger gesture.

Secondly, few dancers have sufficiently precise motor control for a stop or phrase transition to be localised to one frame of video (typically a forty millisecond interval). For example, the dancer's foot may land three or four frames before her hand comes to rest. In such cases, there is some ambiguity as to which frame represents the end of the movement; while the movement is clearly at an end when both limbs have stopped, the perception of the stop is more likely to be associated with the foot motion. This problem is exacerbated when several dancers are dancing to the same music, as the lack of frame-level precision is more marked with multiple dancers. Figure 6.1 illustrates an example of this phenomenon.

A related issue is that a dancer may decelerate the motion of a limb rapidly to coincide with a musical beat, but allow the limb to continue moving at a lower rate until it comes to rest a short time later. While the strong deceleration is the movement correlated with beat, it may not be the most appropriate choice of edit point, as it is not the end of the dance phrase.

A technical difficulty with edit point identification is that while the sudden stopping of a limb might be clearly visible to a human observer, the 'low level stop' visible in the sequence in terms of scene motion content may not occur until several frames later due to the motion of the dancer's clothing or hair. The distinction between a dancer's limb and her clothing is an example of semantic content in a sequence and, as previously discussed, this is a level of signal interpretation not readily available to signal processing techniques.

Finally, not all dance movements are demarcated by motion stops. A commonly occurring example is that of a dancer executing a series of spins. Here, each spin can be considered an independent phrase, but the choice of where one spin starts and another ends is somewhat arbitrary, and need not correspond to signal-level 'stops' at all.

6.1.2 Assessment of Edit Point Detection

To facilitate assessment of the techniques presented here, ground truth data was prepared for several sequences. The set of ground truth frames is designated $\mathcal{G}$. In the presentation of detection performance below, it should be borne in mind that different assessors would have marked the sequences differently—in particular, some frames detected by these techniques are defensible as edit points despite their not appearing in the prepared ground truth.

The performance of edit point identification algorithms is assessed with reference to a ground truth prepared by hand for several sequences. It has been outlined above why establishing a precise ground truth for a sequence is a difficult undertaking. A further difficulty in assessing edit point detection performance is that the 'cost' of a missed, spurious, or inaccurately localised edit point will vary with the application for which the edit points are destined. In the main application under consideration in this thesis, dance footage is to be resynchronised such that edit points are aligned with a new beat sequence (described in chapter 8. Here spurious edit points are often less offensive than missed edit points, and the impact of inaccuracies in edit

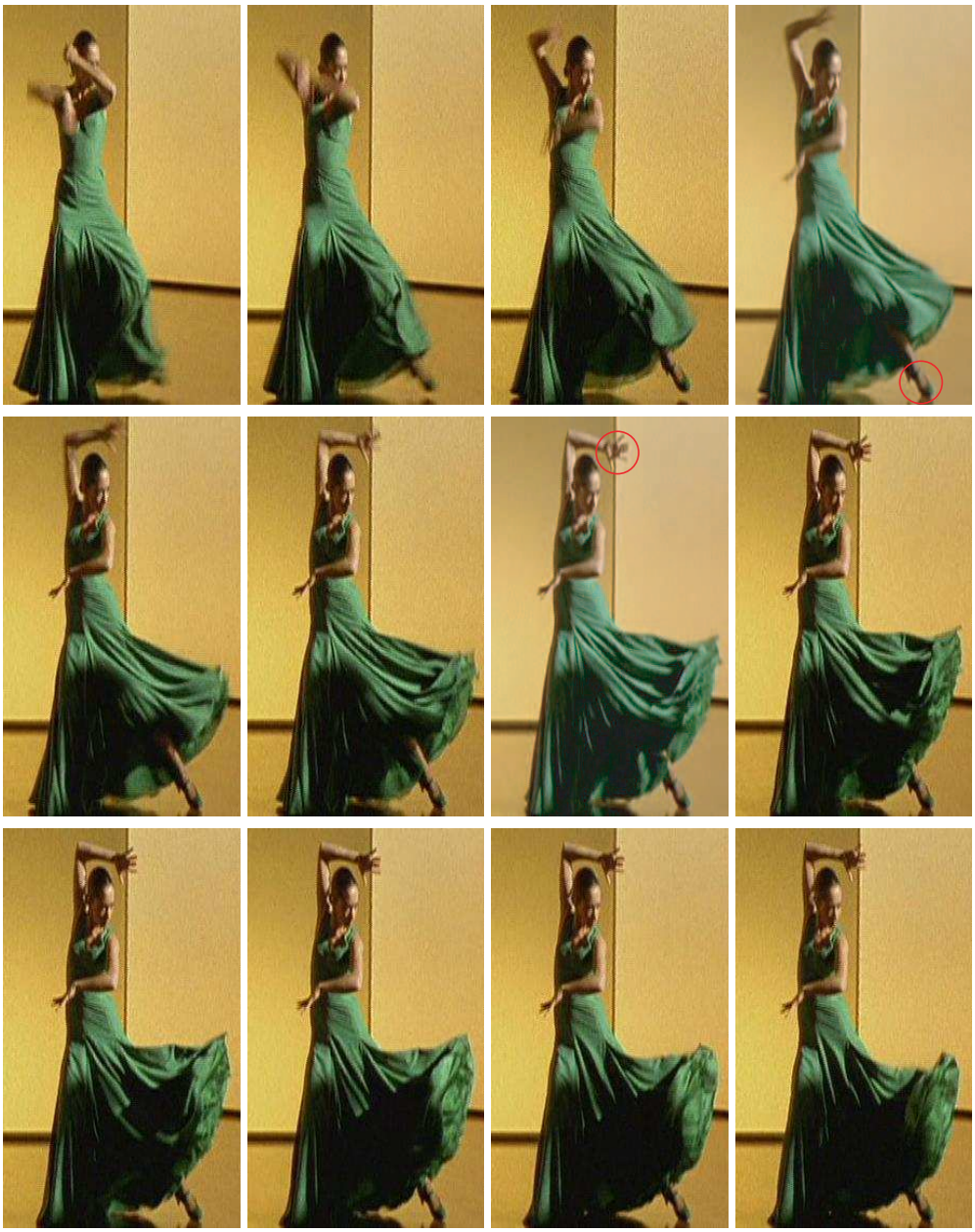


Figure 6.1: Sudden stops are not always localised to within one frame; here, the dancer's foot stops first, followed by her hand (indicated by red circles); her dress continues to move for some frames after.

point localisation depends greatly on the edit point in question. The numerical results presented in this chapter should therefore be considered as roughly indicative of the relative performance of different approaches, rather than as an objective measure of the absolute quality of the result. At the end of the chapter, and in the accompanying videos, detected edit points are visually presented, and this gives a more accurate indication of detection performance.

An edit point detection procedure results in a set ep of estimated edit points, which is compared to some ground truth for the sequence, designated $\mathcal{G}$. An estimated edit point is considered a match if it lies within 5 frames of an edit point in the ground truth data. For edit point detection schemes described in this chapter, performance in detection is described in terms of the number of hits, number of false alarms, and number of missed edit points. These three measures are denoted by #H, #FA, and #M, respectively. The recall and precision, denoted by R and P, are also presented.

The set of matched edit points, $\hat{ep}$, can be assessed in its own right by the distribution of distances between matched pairs $\hat{ep}(g) - \mathcal{G}(g')$, where $\mathcal{G}(g')$ is the ground truth edit point closest to $\hat{ep}(g)$. The distribution is assessed by

$$MD_{median} = \underset{g}{\text{median}} \left\{ \hat{ep}(g) = \mathcal{G}(g')) \right\} \tag{6.1}$$

$$MD_{mean} = \text{mean}_g \left\{ \hat{ep}(g) = \mathcal{G}(g') \right\} \tag{6.2}$$

$$MD_{var} = \text{var}_g \left\{ \hat{ep}(g) = \mathcal{G}(g') \right\} \tag{6.3}$$

For all these measures, lower magnitude represents better performance, in the sense of a closer correspondence between estimated edit points and the corresponding ground truth edit points.

In the first sections of this chapter, the results described are for the three training sequences described in appendix B. This numerical assessment of edit point detection is used primarily to determine the optimal parameters for the different techniques. The final section of the chapter, and the accompanying DVD, present results on other sequences to illustrate how the techniques described here generalise to other material.

6.2 Edit Point Identification using Amount of Local Motion

One of the simplest ways to characterise the motion content of a video sequence is to consider only the amount of motion in each frame. In this section, a method for edit point identification is presented which relies on analysis of the *motion trace*. This is a signal representing the amount of local motion at each frame of a sequence. Sudden decreases in the motion trace are found where the amount of local motion in the sequence decreases suddenly; these locations often correspond to a particular kind of edit point termed a sudden stop. In outline, then, the approach proceeds by computing the motion trace of a sequence, dividing it into a series of peaks, and considering the minima between peaks as candidate edit points. Not all edit points in a sequence will correspond to minima in the motion trace, nor will all minima correspond to edit points—but the approach generates aesthetically pleasing results nevertheless.

6.2.1 Computing the Motion Trace

The first stage in computing the motion trace is to quantify the amount of local motion between a pair of frames. This is approximated by the Displaced Frame Difference (DFD) after global motion compensation, using the refinement method described in chapter 2. This DFD is then thresholded with a a value of 15, generating the Local Motion Map (LMM): a map such that $LMM(\mathbf{x}) = 1$ wherever local motion has been detected. Small blotches in the LMM arising due to noise or defects in the video are suppressed by discarding any connected regions in the LMM containing fewer than 20 pixels. The motion trace is then defined by

$$motion(n) = \sum_{\mathbf{x}} (LMM_n(\mathbf{x})) \tag{6.4}$$

Using the DFD values themselves to quantify the amount of local motion was found to be less effective than the binary-valued LMM. This is because the DFD values are sensitive to variations in the contrast between background and foreground regions.

6.2.2 Minima in the Motion Trace

As described above, it is expected that edit points in the sequence should correspond to local minima in the *motion* trace. Figure 6.2 shows this effect for an number of frames from the *greenDancer* sequence. In this section, the approach to minima detection used is described. The *motion* trace will be referred to as $M(n)$ for clarity in this section and the next.

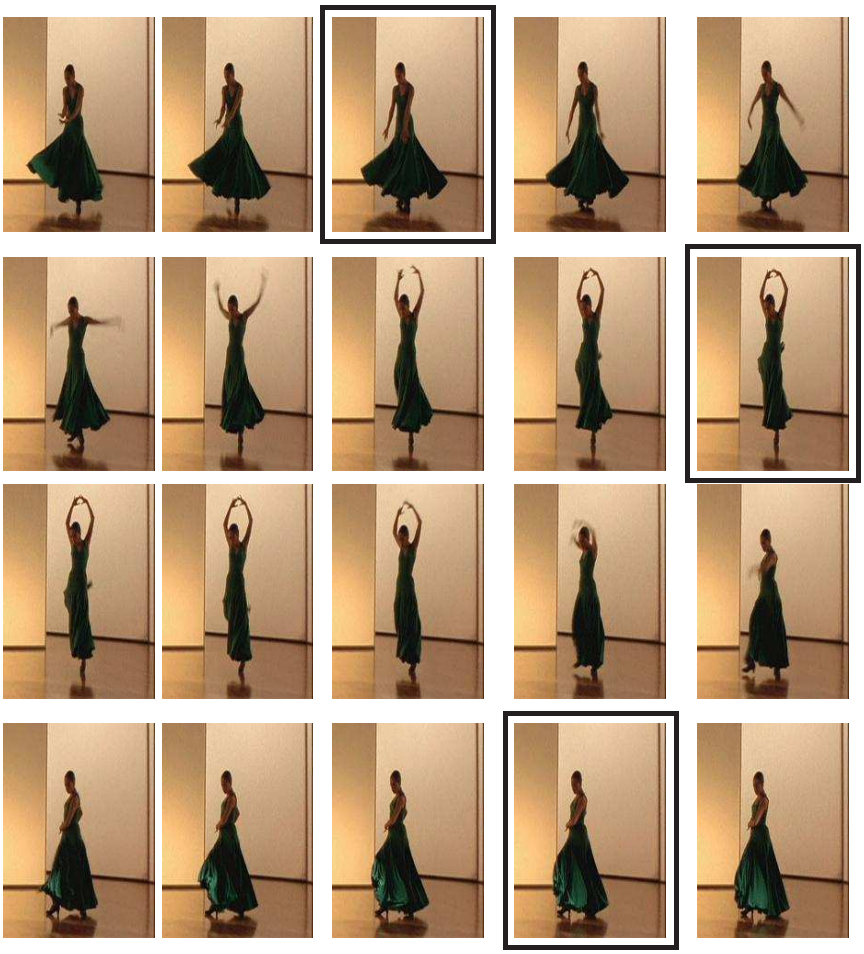
Minima are detected using the sign of the first derivative of the *motion* trace. The first derivative is designated $d(n) = M(n) - M(n-1)$, and the sign of $d(n)$ is denoted by $s(n) = \text{sgn}(d(n))$. Minima in the motion trace are identified wherever the $s(n)$ changes from -1 to 0 or 1. The set of minima resulting is denoted $\mathcal{M}$.

Only significant minima in the *motion* trace will correspond to edit points; minima arising due to noise or small changes in the signal value will not. Three filtering components for suppressing detection of such spurious minima are now described.

6.2.2.1 Savitzky-Golay filtering

The motion trace is typically a very noisy signal, and spurious local minima arising due to noise are common. Dealing with this noise is the first consideration. The first stage of processing is therefore the application of Savitzky-Golay filtering [258, 268].

In this smoothing method, a smoothed value, $\hat{M}(n)$, is obtained by fitting a polynomial to a window of values about $M(n)$. For this work, a polynomial order of 3 and a window size of 7 points are used. Consider that fitting a polynomial of order 3 to $\{(m, M(m)) : |m - n| \leq 3|\}$ results in coefficients α, β, γ, δ. The smoothed value is then given by $\hat{M}(n) = \alpha n^3 + \beta n^2 + \gamma n + \delta$. This smoothing method is chosen as peaks and troughs in the smoothed signal are comparatively well preserved, whereas these features can be attenuated if simpler low-pass filtering is employed.



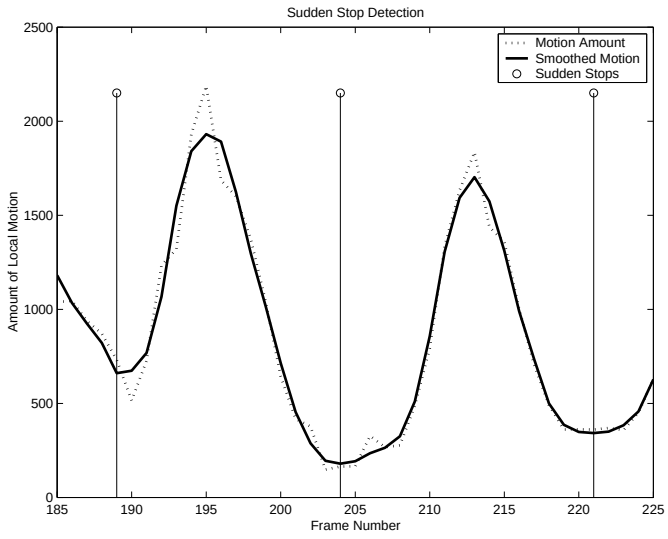


Figure 6.2: A motion trace and the corresponding frames from a sequence depicting a dancer. Sudden stop locations are identified by vertical lines traversing the motion trace, and by a black border surrounding the corresponding video frames. It can be seen that minima in the motion trace correspond to locations which constitute a parsing of the phrases in the dance.

One drawback of polynomial filtering methods such as the Savitzky-Golay technique is that values outside of the range of the original signal can be introduced. In particular, where the motion trace rises suddenly from zero, the smoothed trace will contain values below zero. These values are replaced with zero.

Figure 6.3 (a) shows how this effects the motion trace. The two lower traces in the figure show how the $s(n)$ is also smoothed by this filtering, resulting in fewer detected minima.

6.2.2.2 Relative difference gating

Using the sign change of $d(n)$ to detect minima results in undesired detection in portions of the signal where the actual changes in signal level are relatively small. Signal differences are gated using a locally defined threshold to prevent detection of these minima. The threshold $\hat{d}(n)$ is found by

$$\hat{d}(n) = \underset{(n-8)\leq n' \leq (n+8)}{\text{median}} (|d(n')|) \tag{6.5}$$

$d(n)$ is then gated according to

$$d(n) = \begin{cases} d(n), & d(n) \geq 0.4\hat{d}(n) \\ 0, & d(n) < 0.4\hat{d}(n) \end{cases} \tag{6.6}$$

Figure 6.3 (b) shows how small changes in the signal are flattened out by this method, thereby suppressing spurious minima.

6.2.2.3 Median filtering the sign signal

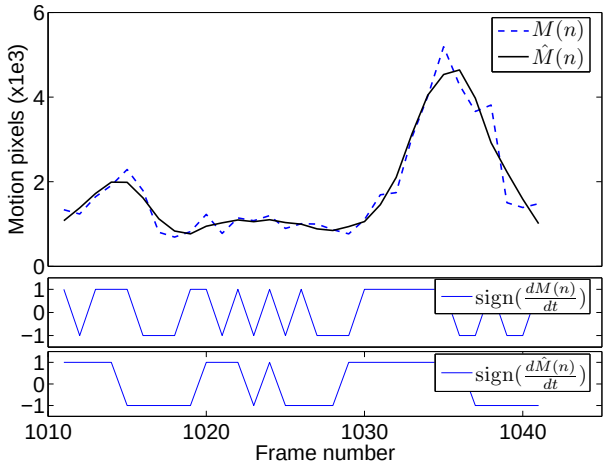
Brief changes in the sign of $d(n)$ are unlikely to correspond to significant minima in the motion trace. Figure 6.3 (c) shows a segment of the motion trace in which two spurious minima are detected due to noise in the signal. These spurious minima are visible as spikes in the middle plot. Median filtering is the preferred method for dealing with such impulsive noise; here, a three-tap median filter is applied. The median filtered sign signal is designated $\hat{s}(n)$. The bottom trace in figure 6.3 (c) shows $\hat{s}(n)$, where the spurious minima have been removed.

6.2.2.4 Trace minima as edit points

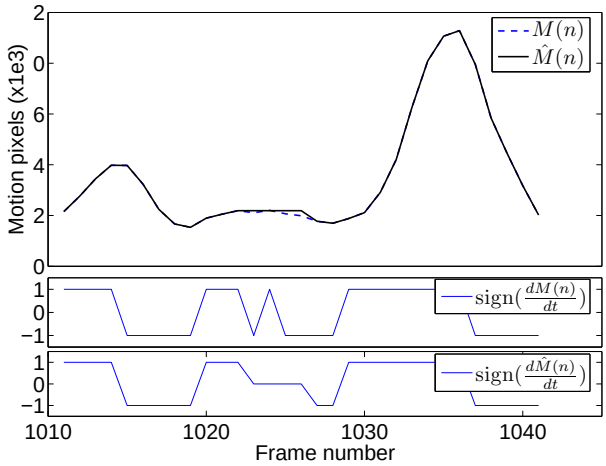
To investigate the effectiveness of the filtering methods described above, edit point detection is evaluated using all the located minima as candidate edit points, for each possible filtering scheme. Table 6.1 illustrates the performance of each filtering approach for three sequences.

In the table, M is the unfiltered motion trace and $\hat{M}$ is the trace with Savitzky-Golay filtering applied. The subscript s indicates that median filtering of $s(n)$ has been applied, and the subscript r shows that relative difference gating has been applied. A total of eight possible smoothing schemes are possible by combining these steps.

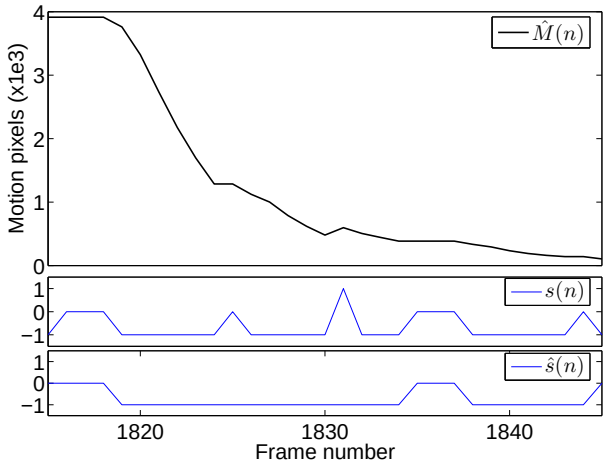
No single scheme attains the best performance (in the M_{PR}—mean of precision and recall—sense) in all three sequences. However, in each case the best performing scheme incorporates



(a) Savitzky-Golay filtering



(b) Relative magnitude of differential



(c) Median filtering $s(n)$

Figure 6.3: Minima detection in the Motion Trace.

Savitzky-Golay smoothing. On the basis of the table, the $\hat{M}_r$ signal is chosen as the best compromise over all three sequences. In the remainder of this work, the *motion* trace filtered with Savitzky-Golay filtering and relative difference gating, but median filtering is not applied to $s(n)$.

The low precision scores evident in the table suggest that not all spurious minima are due to noise. There are motion characteristics that introduce minima in the *motion* trace which do not correspond to edit points. For example, in sequences with a non-homogeneous background, the movement of a dancer's limb over background areas of differing colour can result in different levels of motion detection, despite the actual motion remaining constant. A related problem is that of self-occlusion, where a dancer's limbs cross one in front of the other. The limbs are generally the same colour, so as they cross over the amount of motion detected drops suddenly. Such cases will introduce false alarms even when the smoothing and minima detection schemes applied are optimal.

6.2.3 Peaks in the Motion Trace

As a higher level of motion trace analysis, the signal can be considered as a series of peaks. Each peak is expected to correspond to a movement in the video, and it is at the end of each peak that the percussive motion is found. The set of minima found in the *motion* trace as described above are used to parse the trace into peaks: each peak is the region between two successive minima. The goal is to classify peaks in the *motion* trace to inform whether they correspond to movement in the dance or not.

The pth peak is specified by its start and end locations and the location of the maximum value within the peak:

$$\zeta_{leftmin}(p) = \mathcal{M}(p) \tag{6.7}$$

$$\zeta_{rightmin}(p) = \mathcal{M}(p+1) \tag{6.8}$$

$$\zeta_{max}(p) = \underset{p' \in (\mathcal{M}(p), \mathcal{M}(p+1))}{\arg\max} M(p') \tag{6.9}$$

The widths of the ascent and descent regions of the peak are denoted by $\zeta_{ascentwidth}(p)$ and $\zeta_{descentwidth}(p)$, given by

$$\zeta_{ascentwidth}(p) = \zeta_{max}(p) - \zeta_{leftmin}(p) \tag{6.10}$$

$$\zeta_{descentwidth}(p) = \zeta_{rightmin}(p) - \zeta_{max}(p) \tag{6.11}$$

These characteristics are illustrated in figure 6.4.

The remainder of this section is concerned with classifying peaks into desired and undesired for the purpose of identifying sudden stop edit points.

	#H	#FA	#M	P	R	M_{PR}	MD_{median}	MD_{mean}	MD_{var}
Sequence *greenDancer*									
No Savitzky-Golay filtering									
M	77	297	2	0.21	0.97	0.59	1	0.91	3.35
M_s	65	130	14	0.33	0.82	0.58	0	0.35	6.08
M_r	79	275	0	0.22	1.00	0.61	0	0.47	3.15
M_{rs}	66	112	13	0.37	0.84	0.60	0	0.50	6.99
With Savitzky-Golay filtering									
$\hat{M}$	63	116	16	0.35	0.80	0.57	1	0.81	6.90
$\hat{M}_s$	59	82	20	0.42	0.75	0.58	1	0.75	7.12
$\hat{M}_r$	66	108	13	0.38	0.84	0.61	1	0.59	7.51
$\hat{M}_{rs}$	63	70	16	0.47	0.80	0.64	1	0.71	7.85
Sequence *maleDancer*									
No Savitzky-Golay filtering									
M	100	328	3	0.23	0.97	0.60	1	0.55	3.38
M_s	87	124	16	0.41	0.84	0.63	1	0.80	6.58
M_r	101	315	2	0.24	0.98	0.61	1	0.63	3.01
M_{rs}	86	107	17	0.45	0.83	0.64	1	0.80	5.38
With Savitzky-Golay filtering									
$\hat{M}$	91	122	12	0.43	0.88	0.66	2	1.21	5.68
$\hat{M}_s$	73	72	30	0.50	0.71	0.61	2	1.11	6.35
$\hat{M}_r$	89	114	14	0.44	0.86	0.65	1	0.87	5.32
$\hat{M}_{rs}$	75	69	28	0.52	0.73	0.62	1	1.08	5.89
Sequence *ballet2*									
No Savitzky-Golay filtering									
M	115	130	4	0.47	0.97	0.72	1	0.57	2.98
M_s	105	47	14	0.69	0.88	0.79	0	0.32	3.95
M_r	115	134	4	0.46	0.97	0.71	0	0.05	2.40
M_{rs}	108	46	11	0.70	0.91	0.80	0	-0.09	3.62
With Savitzky-Golay filtering									
$\hat{M}$	110	50	9	0.69	0.92	0.81	0	0.29	3.55
$\hat{M}_s$	101	34	18	0.75	0.85	0.80	0	0.30	3.51
$\hat{M}_r$	111	54	8	0.67	0.93	0.80	0	0.04	3.49
$\hat{M}_{rs}$	101	39	18	0.72	0.85	0.79	0	0.07	3.17

Table 6.1: Motion trace minima as edit points: performance in three sequences

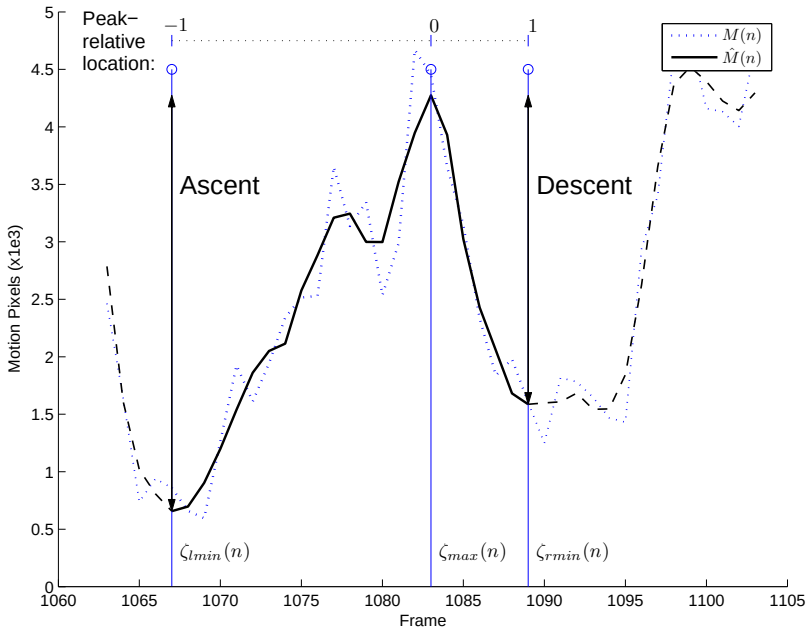


Figure 6.4: Anatomy of a peak.

6.2.4 Peak-relative location of edit points

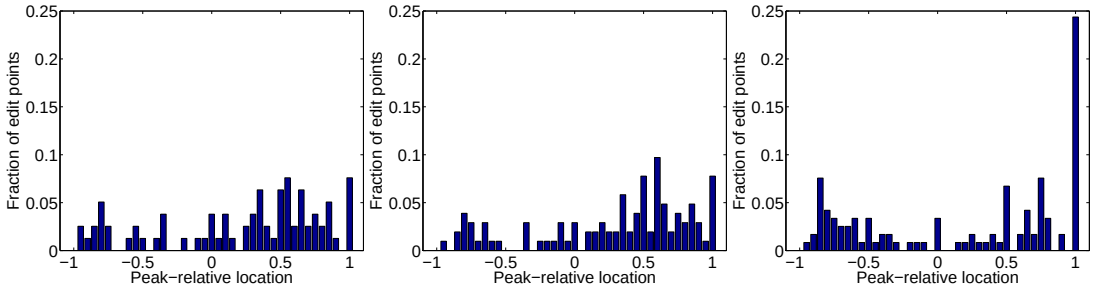
In order to describe the location of a frame relative to peak features, the peak-relative location is introduced, defined by

$$\zeta_{prel}(n,p) = \begin{cases} 1 - \frac{n-\zeta_{leftmin}(p)}{\zeta_{ascentwidth}(p)}, & \text{if } n \leq \zeta_{max}(p) \\ \frac{n-\zeta_{max}(p)}{\zeta_{descentwidth}(p)}, & \text{if } n > \zeta_{max}(p) \end{cases} \tag{6.12}$$

such that values from -1 to 0 cover the ascent region of the peak, and 0 to 1 lie in the descent region. The peak-relative location is also illustrated in figure 6.4.

It has been outlined above that due to noise, differences in background contrast, and self-occlusion, not all minima in the *motion* trace correspond to edit points. Analysis of the peak-relative location of stops in the ground truth data shows that nor are all edit points located at minima in the *motion* trace. The distributions of peak-relative location for the ground-truth edit points in three sequences are shown in figure 6.5.

In general, this phenomenon is due to the fact that a 'stop' or phrase transition in a dance is rarely located within a single frame. As described above, a dancer coming to rest rarely exhibits a single stop frame due to a lack of synchronisation between limbs, residual motion in the dancer's clothing, or simply the difficulty of decelerating instantaneously to a full stop. This is true in particular of sequences where the dancer is wearing flowing clothing, where the



(a) *greenDancer* sequence (b) *maleDancer* sequence (c) *ballet2* sequence

Figure 6.5: Edit point location within peaks.

dancer is shot relatively close up (this amplifies the residual motion after a stop), and where the dance is characterised more by definite stops, as in flamenco dance, than in flowing dance such as ballet.

These factors account for the differences in the three distributions shown in figure 6.5. The *greenDancer* and *maleDancer* sequences depict flamenco dance. This staccato dance style features numerous very definite stops, which means that the peaks in the motion trace have long descents. Therefore the location of the edit point within the peak descent need not correspond to the peak minimum. In the *greenDancer* sequence in particular, the motion due to the dress often results in a long, shallow peak descent after the dancer herself has stopped moving.

The *ballet2* sequence depicts a ballet dancer dancing a solo ballet routine. As this dance is much more flowing, transitions between dance phrases are more likely to consist of deceleration followed by acceleration than to stops where the *motion* trace declines to zero. For this sequence, then, peaks in the motion trace correspond more exactly to individual dance movements. Furthermore, the dancer's clothing in this sequence is relatively inert, and does not exhibit great amounts of secondary motion. For these reasons, many more edit points are located exactly at the peak minimum.

6.2.5 Peak Characteristics

Various characteristics of the peak can be employed to determine which peaks are most likely to correspond to a distinct motion phrase in a video. These heuristics are combined to rank the peaks according to the likelihood of their corresponding to a stop. In this section, four peak characteristics are described, and the distributions found for each characteristic in desired and undesired peaks are presented. Classification can then proceed based on parametric or non-parametric representation of these distributions.

Desired peaks are those which correspond to motion phrases, and so their minima correspond to edit points. In this analysis, a peak is taken to correspond to an edit point if there is a ground-

truth edit point in the peak's descent region, i.e. if

$$\exists(i).\mathcal{G}(i) \in [\zeta_{max}, \zeta_{rightmin} + 3] \tag{6.13}$$

where a tolerance of three frames after the peak's right minimum is allowed.

For each characteristic described, a histogram of the distribution of the characteristic values in desired and undesired peaks is found. These distributions must then be represented by some means appropriate for classification. Two approaches to this issue, parametric and nonparametric distribution modelling, are now described. A description of the four peak characteristics used follows.

6.2.5.1 Parametric Peak Distributions

To model an empirical distribution, i.e. a histogram of values, as a parametric distribution, the appropriate form of distribution must first be chosen. This is achieved by comparing the histogram to a 'gallery' of distribution functions, such as that which appears in [72]. The parameters for the analytic distribution are then found by obtaining an initial estimate via the method of moments, and then refining this estimate by some iterative procedure. In this work, the parameters are refined using a simplex search [235, 258] to minimize the sum of the squared bin-to-bin differences between the histogram and the estimated distribution; the parameters found by this criterion are then the least-squares estimate.

6.2.5.2 Nonparametric Peak Distributions

Instead of choosing a parametric probability distribution function by inspection of the histogram, the histogram can itself be used as the distribution after suitable smoothing. This is approach is referred to as nonparametric density estimation, though in most cases some parameter must in fact be chosen to determine the amount of smoothing applied. Here, a Parzen density estimation approach is adopted. The probability of a value z, according to an existing set of samples $\mathcal{X}$, is given by

$$P_{\text{Parzen}}(z) = \frac{1}{|\mathcal{X}|} \sum_{x \in \mathcal{X}} K(z, x) \tag{6.14}$$

where K is some suitable kernel. In this work, a Gaussian kernel function is used:

$$K(z, x) = \frac{1}{\sqrt{(2\pi)\sigma^2}} \exp\left(-\frac{1}{2\sigma}(z - x)^2\right) \tag{6.15}$$

The smoothing parameter is the variance of the Gaussian, σ^2, and is set to 0.5 times the histogram bin width.

6.2.5.3 Peak Descent

The first characteristic considered is the peak descent. The descent of peak p, designated $\zeta_{descent}(p)$, is the fall in the value of the motion trace between the peak maximum and the

peak's end.

$$\zeta_{descent} = \hat{M}(\zeta_{max}(p)) - \hat{M}(\zeta_{rightmin}(p)) \tag{6.16}$$

It is expected that smaller decreases in the amount of motion in the sequence are less likely to be significant. Peaks are classified on $\log(\zeta_{descent}(p))$, as the range in descent values is very high. Figure 6.6 illustrates the distributions of log peak descents for desired and undesired peaks in several sequences. It can be seen from the figure that in each sequence, the distribution for undesired peaks is different to that for desired peaks, having a greater propensity to lower descent values. To capture the asymmetry of the distribution, it is modelled as a Gumbel minimum distribution:

$$f(x) = \frac{1}{\beta} e^{\frac{-(x-\mu)}{\beta}} e^{(-e^{\frac{(x-\mu)}{\beta}})} \tag{6.17}$$

6.2.5.4 Peak Symmetry

A second consideration is that a peak that descends only a small fraction of its ascent is unlikely to correspond to a desired stop. Define the peak ascent to be the rise in the value of the motion trace between the peak start and the maximum point,

$$\zeta_{ascent} = \hat{M}(\zeta_{max}(p)) - \hat{M}(\zeta_{leftmin}(p)) \tag{6.18}$$

Then the logarithm of the ratio of the peak descent to the peak ascent is a measure of the peak symmetry:

$$\zeta_{symmetry} = \log\left(\frac{\zeta_{descent}}{\zeta_{ascent}}\right) \tag{6.19}$$

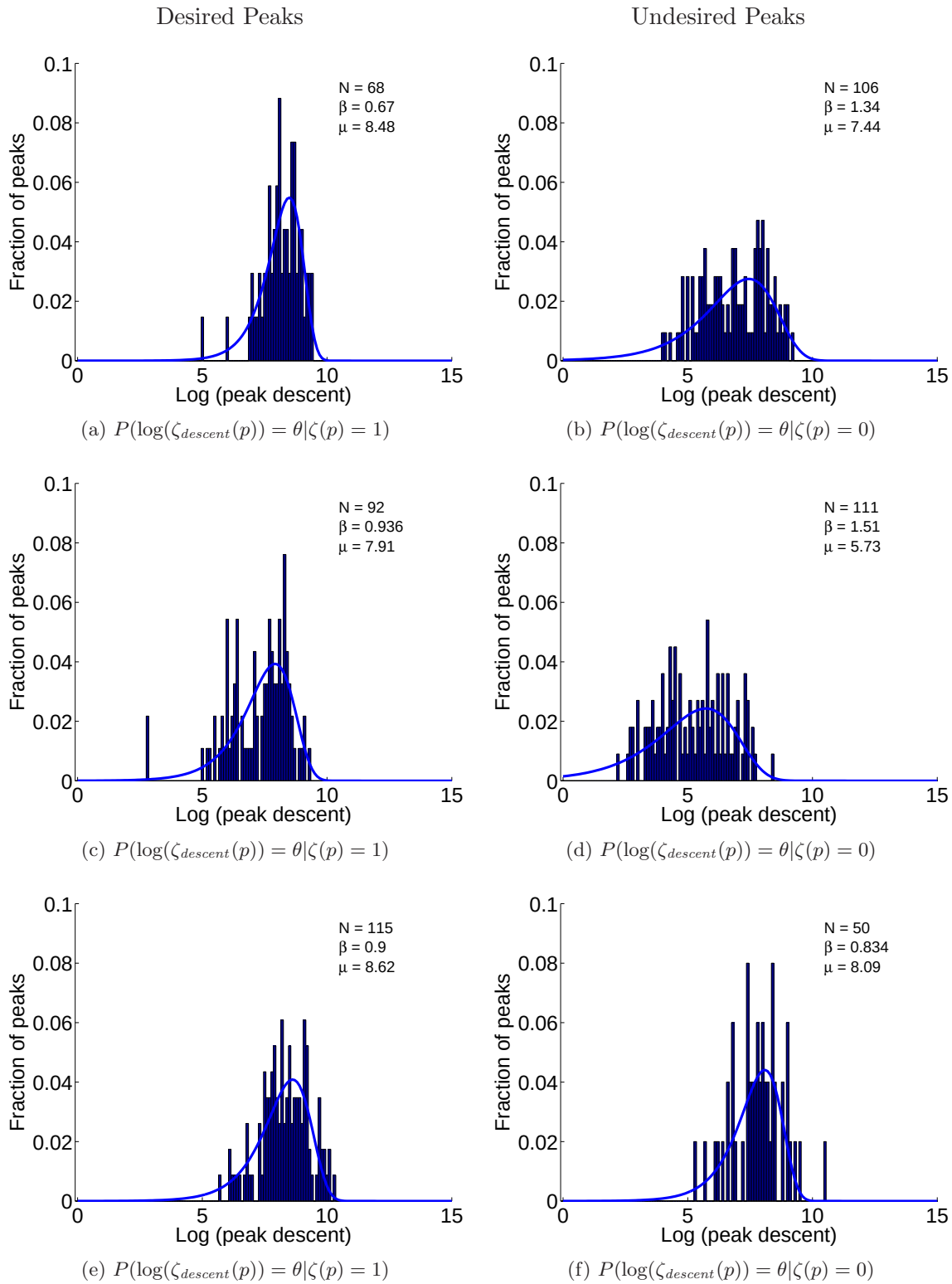
Figure 6.7 illustrates peak symmetries for desired and undesired peaks in several sequences. An asymmetric distribution is chosen for the symmetry values, as a peak whose descent is greater than its ascent may well correspond to a desired stop. Therefore, this characteristic is modelled using the Gumbel maximum distribution.

$$f(x) = \frac{1}{\beta} e^{\frac{(-x-\mu)}{\beta}} e^{(-e^{\frac{-(x-\mu)}{\beta}})} \tag{6.20}$$

However, it is evident from the empirical distributions that several peaks have very skewed asymmetries, saturating the histogram range. To accommodate this feature of the histogram, it is modelled as a mixture distribution, where the Gumbel maximum distribution parameters are found over the histogram excluding the final bin, and a third parameter is added, p_1, corresponding to the fraction of peaks having symmetry scores greater than or equal to 5.

6.2.5.5 Peak Relative Descent

A further characteristic is the magnitude of the descent of a peak, and hence the corresponding stop, relative to the amount of motion at that point in the sequence. This is designated the



(a) $P(\log(\zeta_{descent}(p)) = \theta|\zeta(p) = 1)$

(b) $P(\log(\zeta_{descent}(p)) = \theta|\zeta(p) = 0)$

(c) $P(\log(\zeta_{descent}(p)) = \theta|\zeta(p) = 1)$

(d) $P(\log(\zeta_{descent}(p)) = \theta|\zeta(p) = 0)$

(e) $P(\log(\zeta_{descent}(p)) = \theta|\zeta(p) = 1)$

(f) $P(\log(\zeta_{descent}(p)) = \theta|\zeta(p) = 0)$

Figure 6.6: Peak descent histograms: from top to bottom, the *greenDancer*, *maleDancer*, and *ballet2* sequences. The estimated Gumbel minimum distribution is shown.

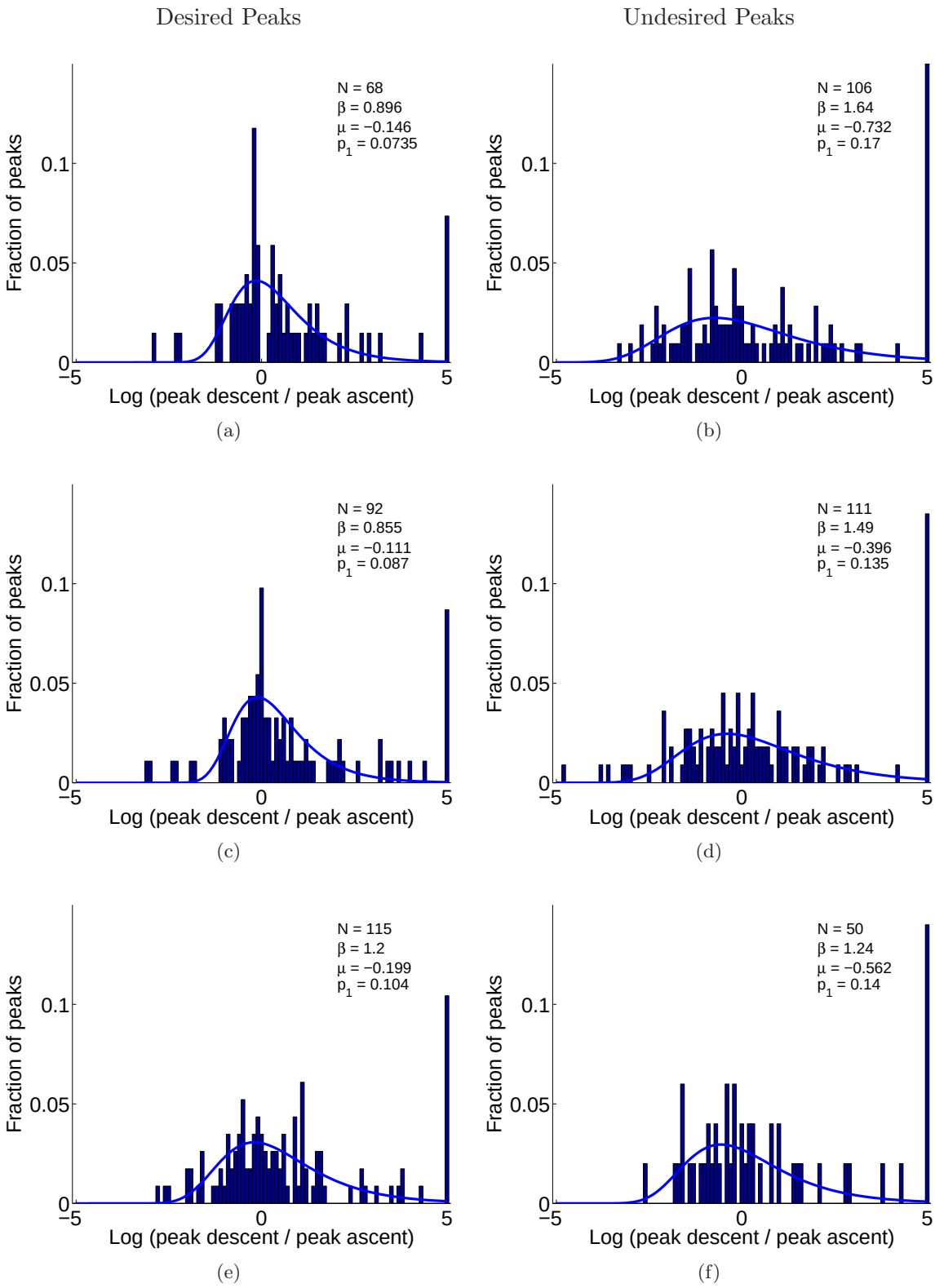


Figure 6.7: Peak symmetry histograms: from top to bottom, the *greenDancer*, *maleDancer*, and *ballet2* sequences. The estimated Gumbel maximum distribution is shown.

peak relative descent:

$$\zeta_{reldesc} = \frac{\zeta_{descent}}{\hat{M}(\zeta_{max})} \tag{6.21}$$

Figure 6.8 illustrates the distribution of this value for several sequences. As this characteristic is limited to the range $0 \ldots 1$, the distribution of values is modelled as a Beta distribution.

$$f(x) = \frac{x^{(p-1)}(1-x)^{(q-1)}}{B(p,q)} \quad \text{for } 0 \leq x \leq 1,\ p,q > 0 \tag{6.22}$$

Where the motion trace ascends suddenly from a low magnitude, Savitzky-Golay filtering can introduce values below zero. This results in an artificially high relative descent score for the preceding peak, particularly in the *maleDancer* and *ballet2* sequences. To allow for this, the Beta distribution is fit to the histogram excluding the final bin, and the fraction of peaks having relative descent 1 is considered a third parameter of the distribution, designated p_1. This third parameter has little discriminatory power, and the chief benefit of the approach is more accurate characterisation of the remainder of the histogram by the Beta distribution.

6.2.5.6 Peak Noise

Many spurious peaks are the result of excessive levels of noise in the motion trace. Therefore, more confidence is associated with peaks in regions where the motion trace is relatively smooth. To quantify this, the absolute difference between the smoothed motion trace $\hat{M}(n)$ and the unfiltered $M(n)$ is found for each point in the peak descent, and the median of these values found.

$$\zeta_{noise}(p) = \underset{\zeta_{max} \leq n \leq \zeta_{descent}}{\text{median}} \left\{ |M(n) - \hat{(}M)(n)| \right\} \tag{6.23}$$

Higher median values correspond to noisier sections of the motion trace, and thus less reliable peaks. Figure 6.9 illustrates that peaks with very high noise scores are more likely to be false alarms. The distribution of this characteristic is modelled as a Gamma distribution.

$$f(x) = \frac{x^{(\gamma-1)}e^{(-x)}}{\Gamma(\gamma)} \quad \text{for } x \geq 0 \tag{6.24}$$

6.2.5.7 Summary

Four peak characteristics have been described, and their distributions for each of three sequences presented. The peak shown in figure 6.4 illustrates these characteristics: this peak has an ascent of 3617 motion pixels and a descent of 2687; the symmetry score is 0.74, and the relative descent 0.85. The median relative noise in the ascent region is 0.12, while in the descent region it is 0.04.

Figure 6.10 show the distributions obtained by combining the peaks for each of the three sequences, and the least-squares fit for the parametric distributions shown. Figure 6.11 shows the Parzen window fit to the same distributions. These combined distributions are then used

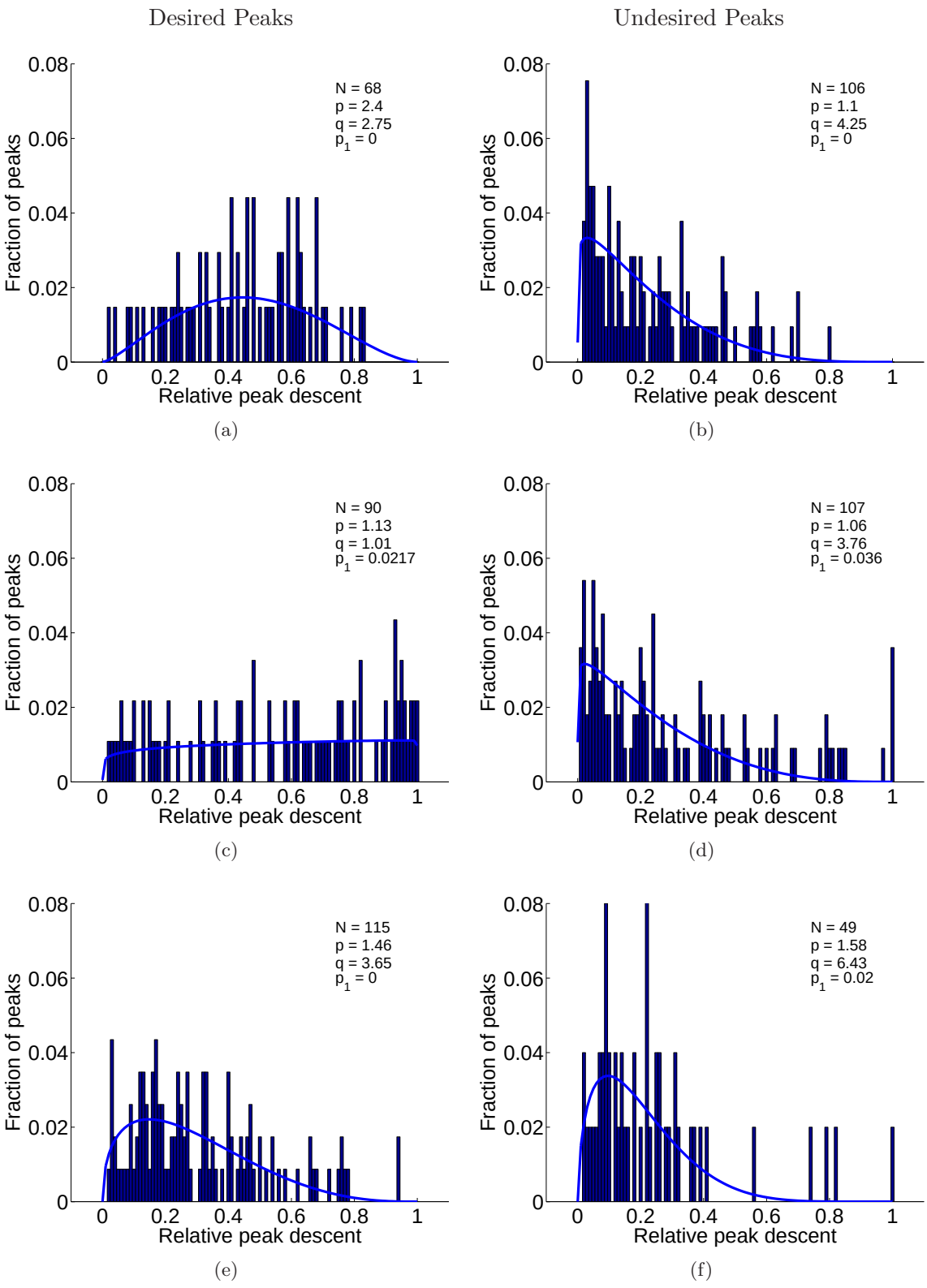


Figure 6.8: Peak relative descent histograms: from top to bottom, the *greenDancer*, *maleDancer*, and *ballet2* sequences. The estimated Beta distribution is shown.

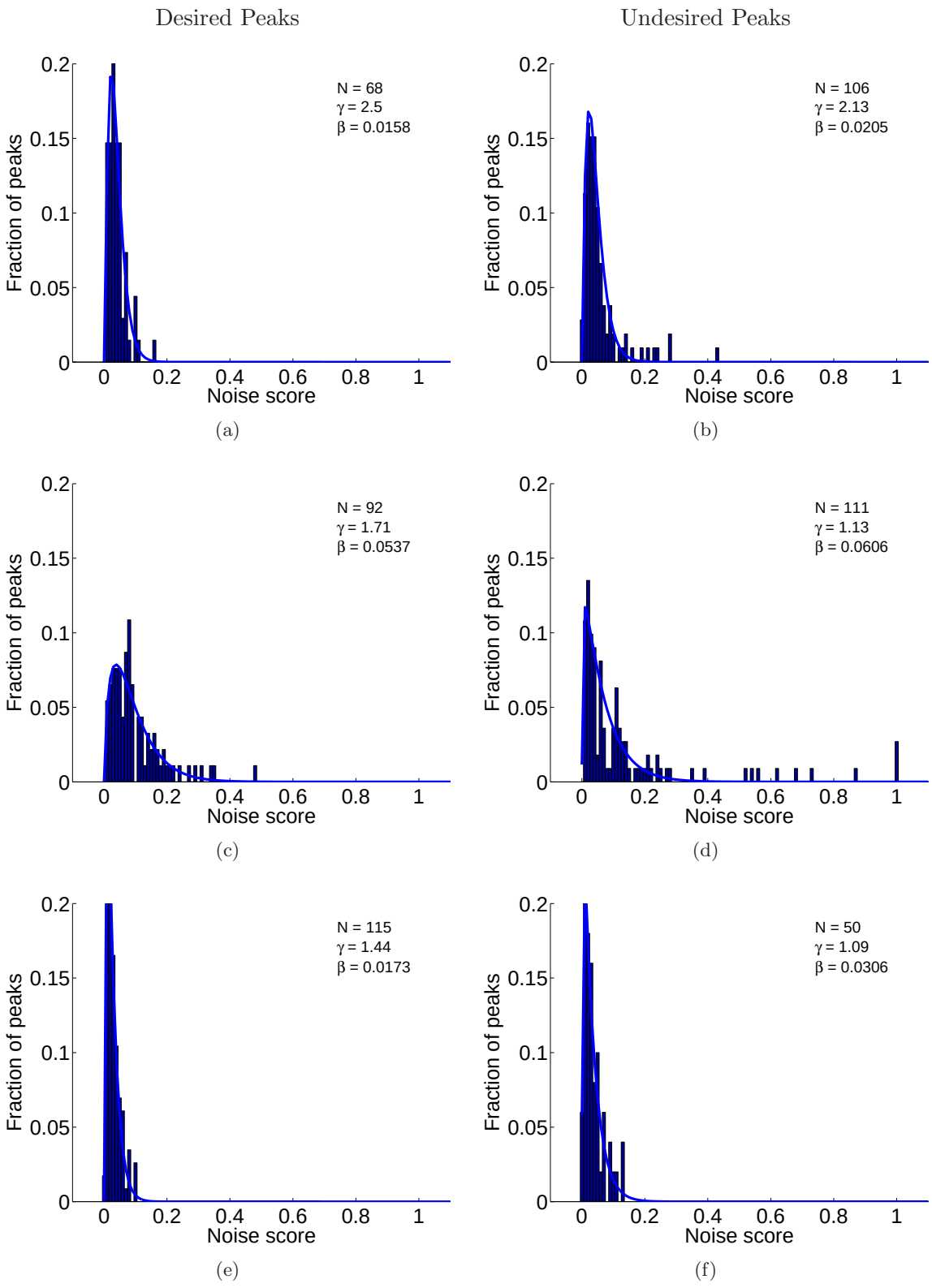


Figure 6.9: Peak noise histograms: from top to bottom, the *greenDancer*, *maleDancer*, and *ballet2* sequences. The estimated Gamma distribution is shown.

for peak classification. The prior probability that a peak in the *motion* trace should correspond to an edit point was found to be 0.507.

6.2.6 Peak Classification

The distributions derived above for peak characteristics can be used to build a classification system distinguishing undesired peaks from desired peaks. Consider peak classification based on only one of the characteristic distributions developed above, the peak descent. Let $\zeta(p)$ be a random variable taking the value 1 when peak p is desired, and 0 otherwise, and $\zeta_{descent}(p)$ is the descent of the pth peak. Applying Bayes' law gives

$$P(\zeta(p) = 1 | \log(\zeta_{descent}(p)) = \theta) \propto P(\log(\zeta_{descent}(p)) = \theta | \zeta_{descent}(p) = 1) P(\zeta(p) = 1) \quad (6.25)$$

$$P(\zeta(p) = 0 | \log(\zeta_{descent}(p)) = \theta) \propto P(\log(\zeta_{descent}(p)) = \theta | \zeta(p) = 0) P(\zeta(p) = 0) \quad (6.26)$$

where the normalising factor $P(\log(\zeta_{descent}(p)) = \theta)$ has been disregarded, as it is common to both expressions. The distributions $P(\log(\zeta_{descent}(p)) = \theta | \zeta(p) = 1)$ and $P(\log(\zeta_{descent}(p)) = \theta | \zeta(p) = 0)$ are determined empirically, as described above; these are the *likelihood* distributions. The *prior* distribution over $P(\zeta(p))$ can be determined from ground truth data. The peak is then classified according to the class having higher posterior probability.

Class assignments are computed using the negative logarithm of the posterior, a quantity referred to here as the energy associated with a particular classification. Peaks are assigned to the class having lower energy, which is equivalent to the class having the higher posterior probability. This avoids issues of numerical stability.

The usual approach to classification when multiple features are being combined is to assume that the features are independent. The posterior probability for features F_1, F_2, F_3 can then be factorised:

$$P(\zeta(p)|F_1, F_2, F_3) \propto \; P(F_1|\zeta(p))\, P(F_2|\zeta(p))\, P(F_3|\zeta(p))\, P(\zeta(p)) \quad (6.27)$$

This is equivalent to summing the energies for each feature. However, the four peak characteristics described above are not necessarily independent, so this approach may be erroneous. To investigate this, the classification system is also evaluated where each class energy is taken as the minimum energy of the combined features, which corresponds to

$$P(\zeta(p)|F_1, F_2, F_3) \propto \max\{P(F_1|\zeta(p)),\; P(F_2|\zeta(p)),\; P(F_3|\zeta(p))\}\, P(\zeta(p)) \quad (6.28)$$

More sophisticated techniques, such as the use of Linear Discriminant Analysis (LDA) [92] to form a weighted combination of each of the likelihood values, could also be applied.

6.2.6.1 Classification Performance

The main lobes of the distributions for each peak characteristic are generally overlapping, and so classification of peaks is a difficult problem. A number of variations are possible within the

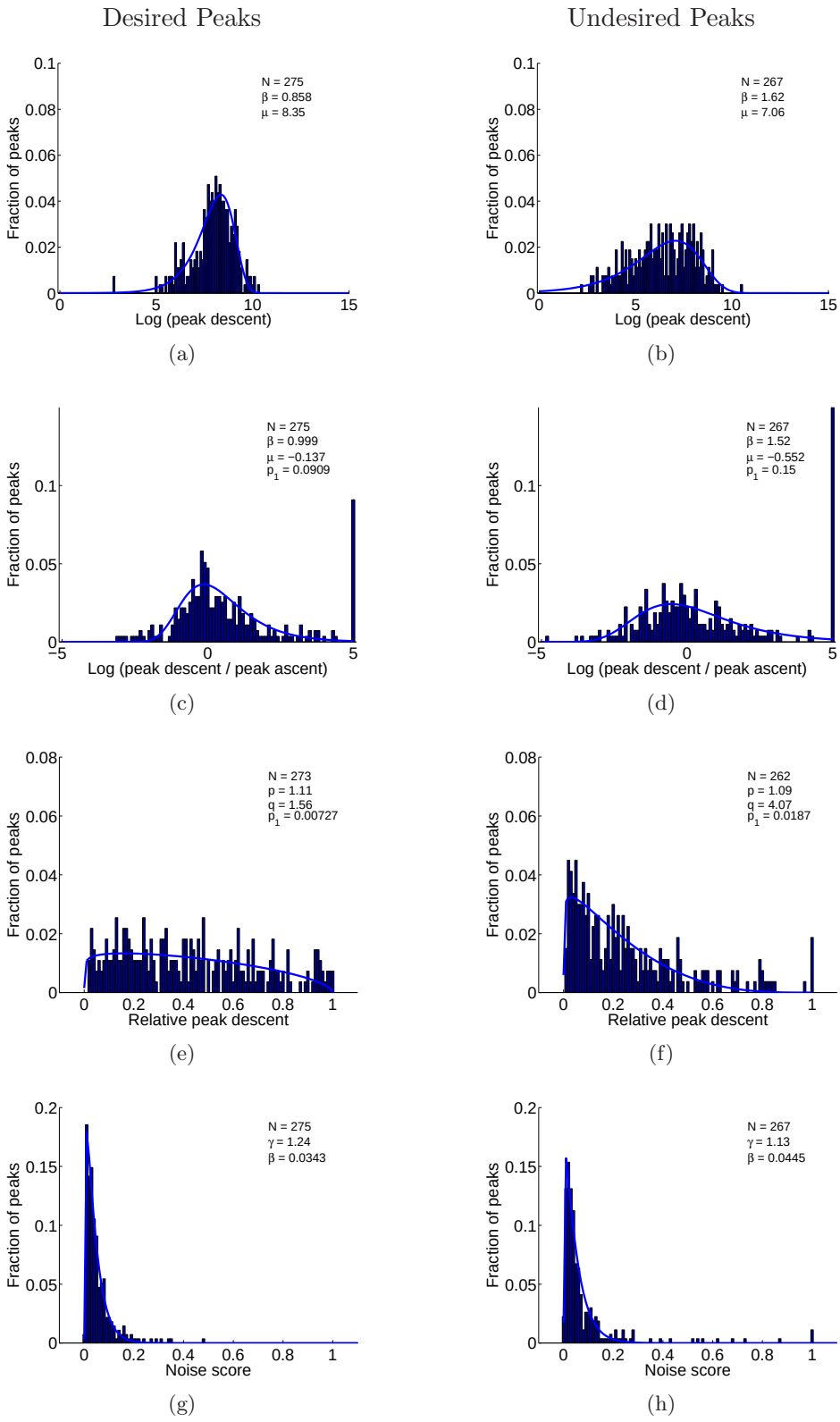


Figure 6.10: Maximum Likelihood estimation of densities for peak statistics of the *greenDancer*, *maleDancer*, and *ballet2* sequences combined.

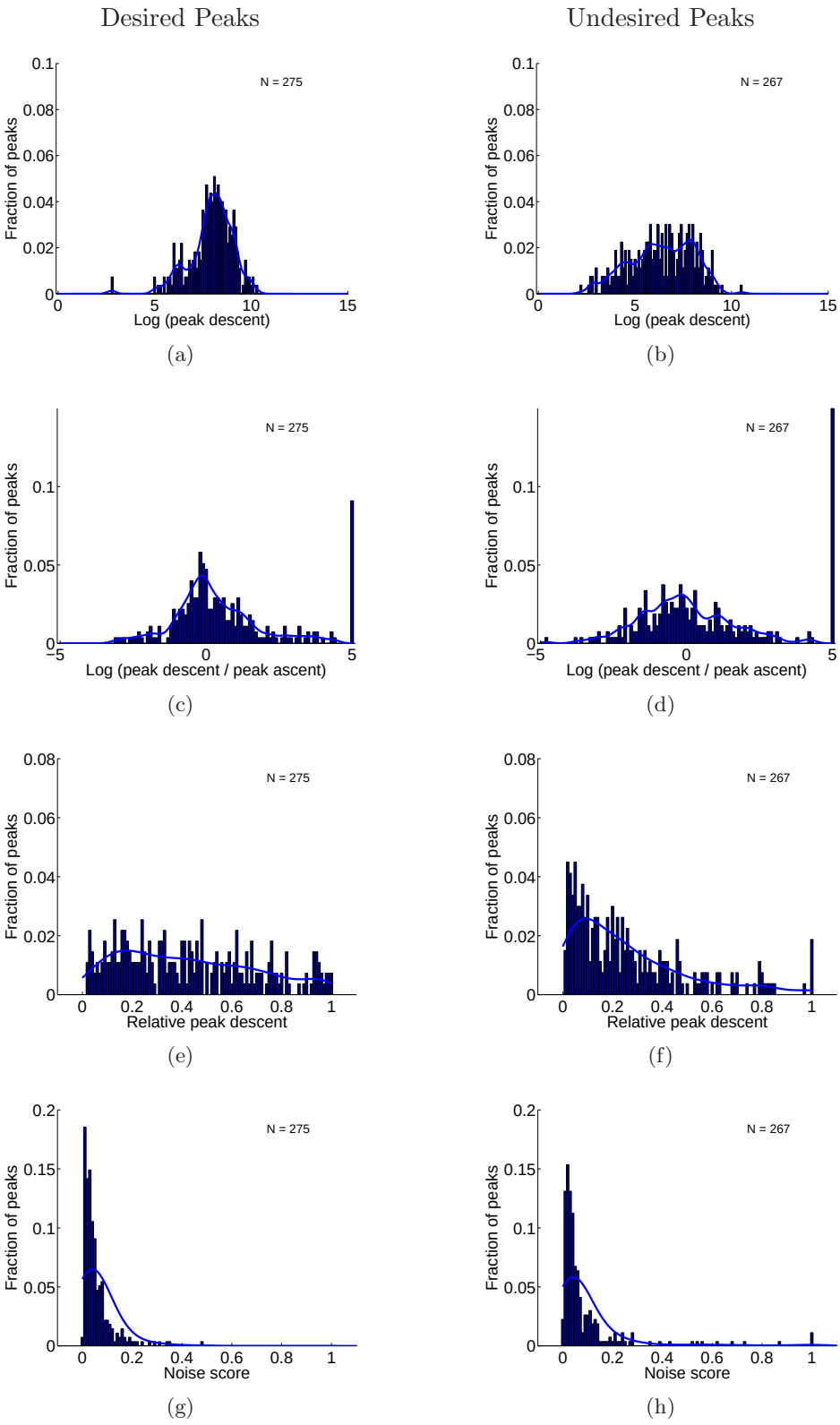


Figure 6.11: Parzen estimations of the densities for peak statistics for the *greenDancer*, *maleDancer*, and *ballet2* sequences combined.

Characteristics	ROC Area
0001	0.78558
0010	0.567169
0100	0.689396
1000	0.531386
0011	0.786952
0001	0.78558

(a) Parametric, Method of Moments fit

Characteristics	ROC Area
0001	0.776828
0010	0.60421
0100	0.689271
1000	0.545382
1101	0.781299
0001	0.776828

(b) Parametric, Least Squares fit

Characteristics	ROC Area
0001	0.791739
0010	0.636308
0100	0.698463
1000	0.550925
1111	0.806034
0001	0.791739

(c) Parzen fit

Table 6.2: Peak classification performance as area under the ROC curve. In this table, combinations of peak characteristics are represented as a four-digit binary number whose digits, from left to right, correspond to the ζ_{noise}, $\zeta_{reldesc}$, $\zeta_{symmetry}$, and $\log(\zeta_{descent})$ characteristics. For example, the 1101 classifier uses all characteristics except symmetry. The top four rows present the result for each of the four characteristics in isolation. The next row shows the best feature combination with factorised likelihood, as in equation 6.27, and the last shows the best combination using 6.28.

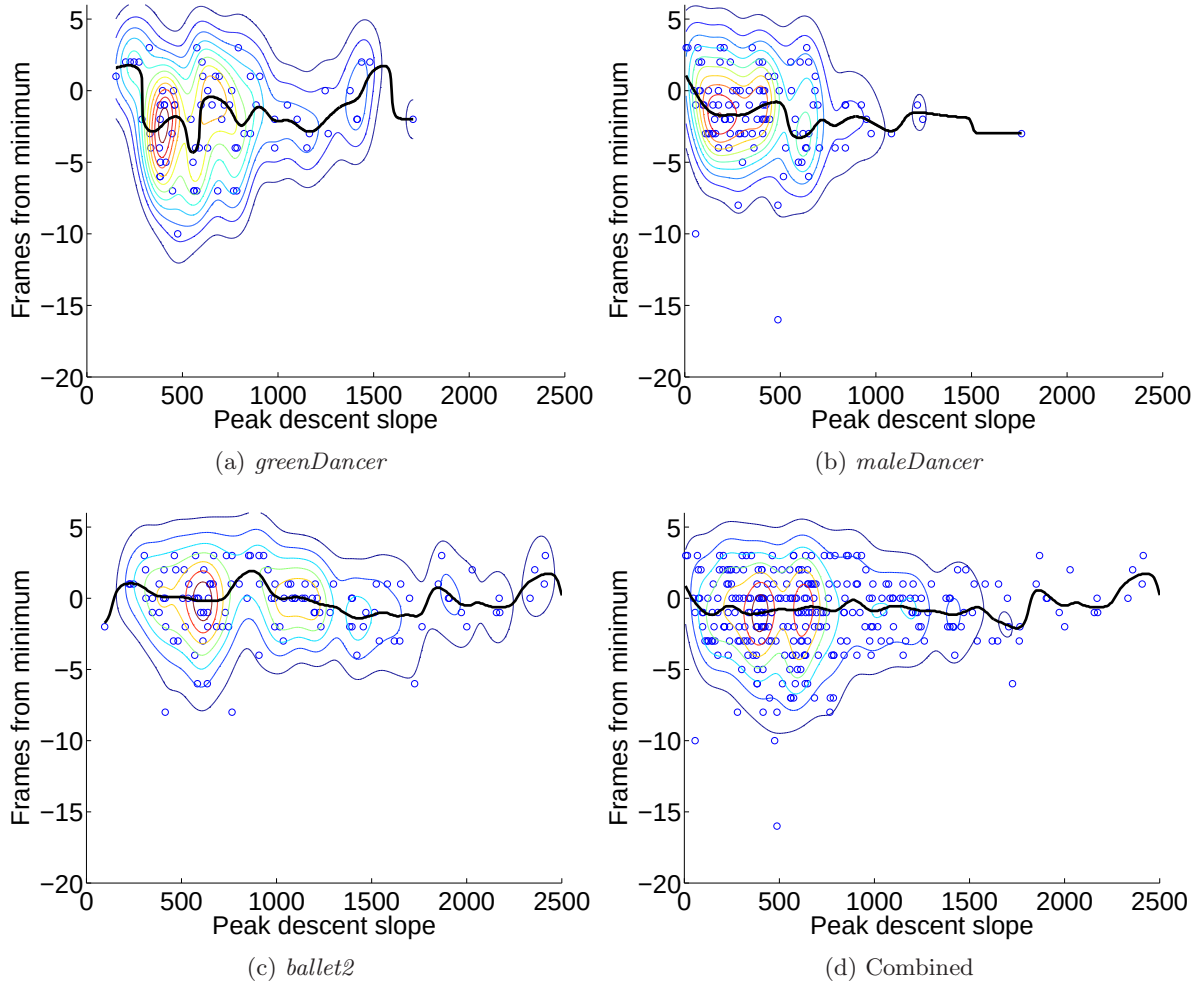
classification system: the combination of peak characteristics to employ; whether to represent the distributions parametrically or using Parzen density estimation; and whether to assume the features are independent, as in equation 6.27, or to classify on the basis of the minimum feature energy as in equation 6.28.

The standard measure of performance for classification systems is the area under the Receiver Operating Characteristic (ROC) curve [92]. Table 6.2 shows the value of this metric for each approach to classification. The best classifier uses Parzen density estimation and all four peak characteristics, achieving an area under the ROC of 0.80. It is clear that most of the classification power is generated by the $\zeta_{descent}$ characteristic, which used alone results in an area under the ROC curve of 0.79.

6.2.7 Edit Point Location and Peak Descent Slope

Where peaks in the motion trace correspond to movements constituting a sudden stop in the dance, it is not always the case that the respective edit point is exactly at a minimum in the motion trace. In some sequences it is found that the edit point is more likely to in fact lie at some intermediate point along the descent. For example, as described above, in the *greenDancer* sequence the continued motion of the dancer's clothing after the dancer's stop results in edit points being more likely to occur at some point between the peak maximum and its minimum point.

Observation of edit point locations in the motion trace suggests that edit points are more



(a) *greenDancer* (b) *maleDancer*

(c) *ballet2* (d) Combined

Figure 6.12: Location of edit points against the descent slope of the associated peak. Negative values indicated that the edit point is before the peak minimum. The black line indicates the most likely location for a given slope.

likely to be close to the minimum point of a motion peaks with steeper descents. Figure 6.12 shows the distribution of edit point location relative to the peak minimum, versus peak descent slope. The black line traversing the figures shows the offset from the peak minimum most likely to correspond to the edit point for each slope value. As anticipated, the variance of the distance from the edit points to the peak minimum is greatest for shallow peak descents. This is particularly true of the *greenDancer* and *maleDancer* sequences. Figure 6.12 (d) shows this distribution found over the three training sequences. Here it is clear that overall, the most likely location for an edit point is about one frame before the peak minimum, for all peak descent slopes.

6.2.8 Edit point detection using peak classification

Recall that the classification procedure operates by assigning to each peak an energy, and assigning each peak to the class having the least energy. The difference between the two energies is effectively a confidence measure pertaining to the classification. Thresholds on this confidence measure can be used to bias the classification procedure towards higher precision, or higher recall; different values of the threshold correspond to different points on the ROC curve. Table 6.3 illustrates the application of this threshold.

Each line of results for each sequence is the outcome of edit point detection after peak classification using the 1111 classifier (i.e. all four peak characteristics), with Parzen density estimation and product-likelihood. Where the bias is log(0.5), a peak is only classified as desired if $(P(\zeta(p)) = 1) \geq 2(P(\zeta(p) = 0))$. The second line corresponds to a bias of 0, and hence a peak is classified as desired if $(P(\zeta(p)) = 1) \geq (P(\zeta(p) = 0))$. In the third line, peaks are classified as undesired only where $(P(\zeta(p)) = 0) \geq 2(P(\zeta(p) = 1))$, and the last line has 'infinite bias'—in other words, all peaks are accepted. Increasing the bias value improves the recall at the expense of precision.

In these analyses, the edit point associated with each peak is placed at the peak minimum. While figures 6.5 and 6.12 illustrate that this is not always accurate, it remains about the best compromise for edit point location, as evidenced by the low scores in table 6.3 in the MD_{median} and MD_{mean} measures.

For the *maleDancer* sequence, using peak classification introduces a considerable improvement in performance, in the M_{PR} sense. The best result achieved with classification is 0.75, compared to 0.66 using the minima only (table 6.1). No improvement is achieved for the other two sequences tested. However, using peak classification does allow each detected edit point to be measured for confidence. To some extent, this confidence accords with the perceptual strength of the edit point, as illustrated on the accompanying DVD. For these reasons, the peak classification approach is key to edit point detection using a combination of traces, described at the end of this chapter.

6.3 Edit Point Identification using the Foreground Bounding Box

The bounding box of the local motion map, equivalent to the DFD bounding box introduced in chapter 2, can be useful in the identification of some edit points. This is because often a dance phrase culminates in a limb outstretched to its maximal extent. Furthermore, points in the sequence where a dancer's limbs are at maximal extent can be satisfying edit points even where these points do not demarcate dance phrase transitions.

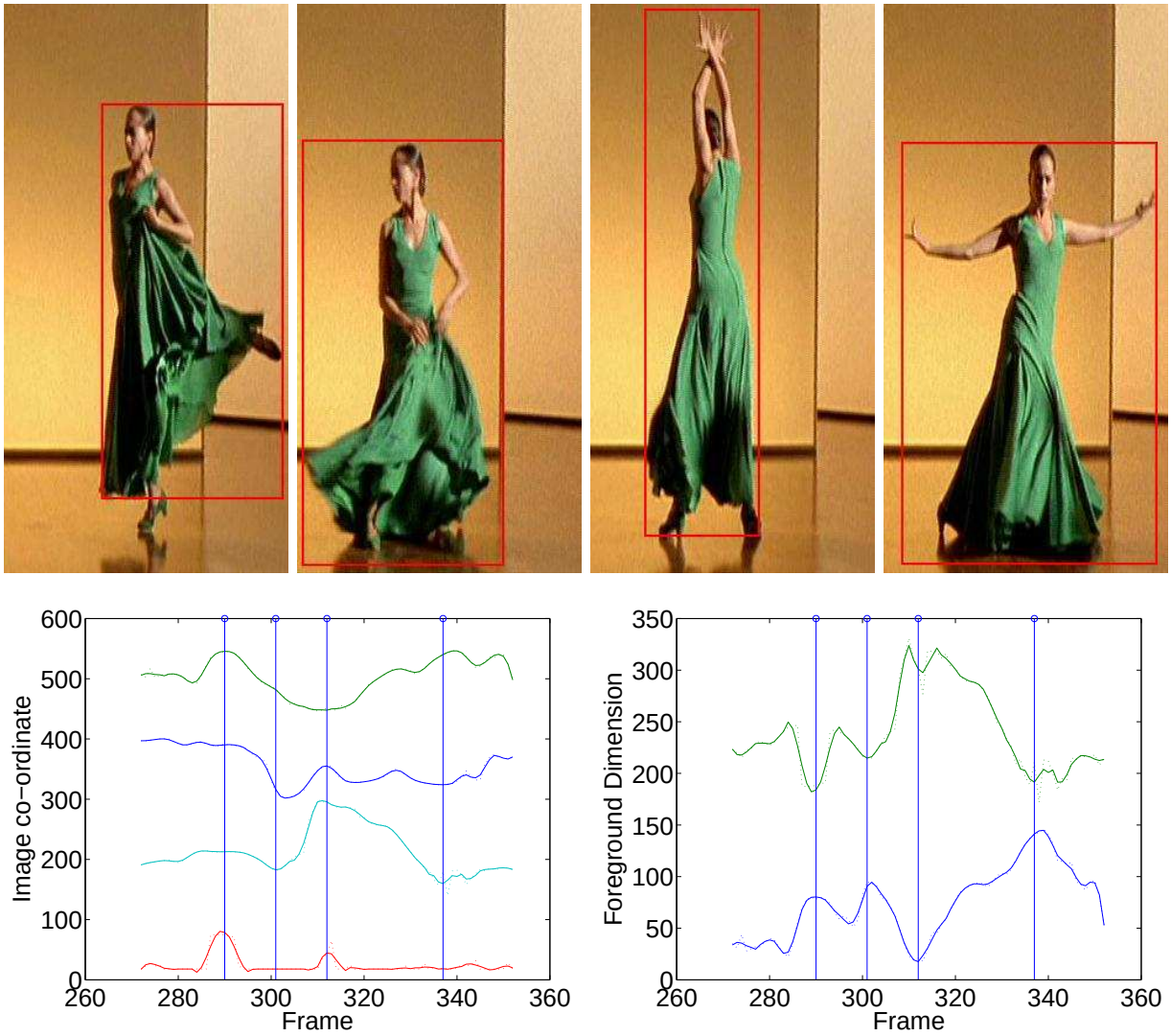
The foreground bounding box can be characterised by the position of each edge, and also its width and height. Figure 6.13 shows each of these measures over a range of 81 frames in the

bias	#H	#FA	#M	P	R	M_{PR}	MD_{median}	MD_{mean}	MD_{var}
Sequence *greenDancer*									
log(0.5)	38	21	41	0.64	0.48	0.56	2	1.32	8.60
log(1)	52	39	27	0.57	0.66	0.61	1	0.88	8.30
log(2)	58	47	21	0.55	0.73	0.64	1	0.93	7.54
∞	66	108	13	0.38	0.84	0.61	1	0.59	7.51
Sequence *maleDancer*									
log(0.5)	48	6	55	0.89	0.47	0.68	1	1.08	5.48
log(1)	66	10	37	0.87	0.64	0.75	1	1.24	5.02
log(2)	73	31	30	0.70	0.71	0.71	1	0.99	5.74
∞	89	114	14	0.44	0.86	0.65	1	0.87	5.32
Sequence *ballet2*									
log(0.5)	88	38	31	0.70	0.74	0.72	0	0.17	3.18
log(1)	97	42	22	0.70	0.82	0.76	0	0.07	3.30
log(2)	103	50	16	0.67	0.87	0.77	0	0.15	3.11
∞	111	54	8	0.67	0.93	0.80	0	0.04	3.49

Table 6.3: Motion trace minima as edit points: biased peak classification

greenDancer sequence. There are four edit points in the ground truth for this sequence; their locations are shown by the vertical lines in the bounding box trace, and the edit point frames themselves are shown above the plots. Each edit point is located at or near a local extremum in the various bounding box traces.

The question of whether to use the location of the bounding box edges, as shown in figure 6.13 (a), or to use the derived measures of the bounding box width and height, shown in figure 6.13 (b), has been given some consideration. Clearly, the mutual information between these two approaches is very high. The more intuitively appealing measure is the bounding box dimensions, and the correspondence between edit points and trace extrema is more apparent here. However, estimates of the foreground region bounding box are highly susceptible to even slight global motion failure, and this can result in very high levels of noise in the bounding box trace. Where this happens, it may be that only one side of the bounding box is affected by global motion estimation failure. The opposite side, unaffected by this failure, may still be useful for edit point identification. If the derived measures of bounding box dimension are used, on the other hand, the 'clean' information in the uncorrupted side is lost. Therefore, all six traces are assessed for their relevance to edit point detection. These are designated the bb_{min_x}, bb_{max_x}, bb_{min_y}, bb_{max_y}, $bbwidth$, and bb_{height} traces.



(a) Foreground bounding box edge positions. From top to bottom: right; left; top; bottom.

(b) Foreground bounding box dimensions. From top to bottom: width; height.

Figure 6.13: Edit points and the foreground bounding box. The vertical blue lines correspond to the four edit points shown. Some bounding box traces have been vertically shifted to improve separation.

6.3.1 Filtering the bounding box traces

Detecting the foreground bounding box reliably depends on a static background, a localised foreground region, and highly accurate global motion estimation. It is found that this last factor in particular introduces a high level of impulsive noise into the bounding box traces. While the refinement to global motion estimation presented in chapter 2 improves the quality of the bounding box greatly, failures can still occur. These failures are often associated with particularly difficult passages in the sequence, and so multiple temporally adjacent failures are

common. Median filtering is an appropriate technique for mitigating the effect of Global Motion Estimation (GME) failure on the traces.

The effect of the resulting noise can be seen in the top trace in figure 6.14. It can be seen that in several regions, multiple adjacent values are lost. Dealing with these large areas of loss requires a very large median filter. However, median filtering the whole signal with a large filter would result in considerable loss of resolution. Therefore, an iterative scheme is employed where the signal is repeatedly smoothed with progressively larger median filters.

Denote the trace before smoothing by t_0. The smoothing scheme is

$$t_i^m = \text{medianFilter}(t_i, 2i+3) \tag{6.29}$$

$$t_{i+1}(n) = \begin{cases} t_i^m(n), & |t_i(n) - t_i(n-1)| < 100 \\ t_i(n), & \text{otherwise} \end{cases} \tag{6.30}$$

Here $\text{medianFilter}(S, n)$ denotes the result of median filtering a signal S with a median filter of width n. Jumps in the trace value of more than 100 pixels are taken to be due to noise. These noisy values are then replaced with values from the signal after median filtering, where the median filter width is increased by two at each iteration. The algorithm is terminated when there are no jumps greater than 100 in the trace, or when the median filter width is greater than 50.

The lower trace in figure 6.14 illustrates the results of the process, applied to the bb_{min_x} trace of the *maleDancer* sequence. This process is applied to all six bounding box traces. It is noted here that noise detection based on comparison of the filtered signal with the original, i.e. using

$$t_{i+1}(n) = \begin{cases} t_i^m(n), & |t_i(n) - t_i^m(n)| < 100 \\ t_i(n), & \text{otherwise} \end{cases} \tag{6.31}$$

might be considered a more intuitive scheme for noise reduction with median filtering. This scheme was evaluated, and found to yield worse results for trace smoothing than 6.30.

6.3.2 Trace Extrema as Edit Points

To first establish whether bounding box trace extrema can be of any use in dance phrase detection, the correspondence between all trace extrema and ground truth edit points is assessed. Here every local minimum in the trace is taken as a candidate edit point. Minima detection in each trace was assessed using each of the eight smoothing schemes described in section 6.2.2.4. The results are presented in appendix C.1.1.

For these traces, the best smoothing scheme overall includes Savitzky-Golay filtering and relative difference gating, but not median filtering of the sign of the first derivative. Performance is lower than in the *motion* trace, with precision values between 0.45 and 0.55 and recall typically in the range 0.7—0.8.

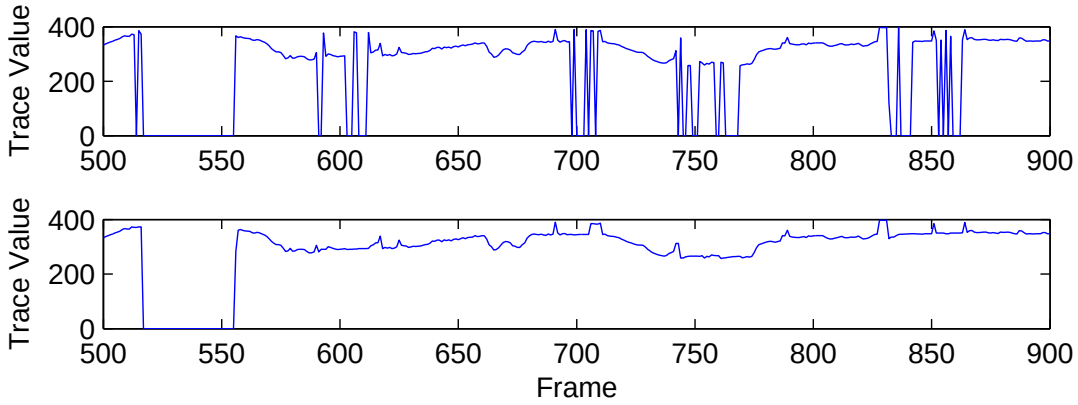


Figure 6.14: Impulsive noise reduction in the bb_{min_x} trace of the *maleDancer* sequence. The upper figure shows the unfiltered values, while the lower plot shows the values after filtering.

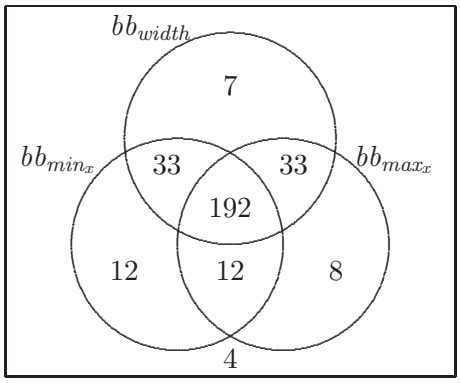
6.3.3 Mutual redundancy between bounding box traces

The extent to which these traces are mutually redundant can be investigated by examining how many ground truth edit point detections are unique to each one. This information is shown in figure 6.15. The minima detected by each trace were detected, and the overlap between the edit points thus detected was found. For example, figure 6.15 (a) shows that the bb_{width} trace discovered 7 edit points that were not also discovered by either the bb_{min_x} or bb_{max_x} traces. 192 edit points were found common to all three traces, and 4 edit points were missed by all three. A similar analysis has been applied to the false alarms detected by the bounding box traces.

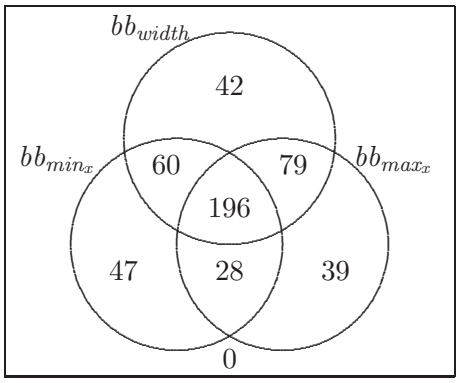
It is apparent that each trace discovers some unique edit points. Furthermore, the false alarms detected by each trace are moderately disjoint. It is therefore to be expected that some combination of the traces should improve edit point detection performance. Edit point detection through combined trace analysis is investigated in the final section of this chapter.

6.3.4 Peaks in the bounding box traces

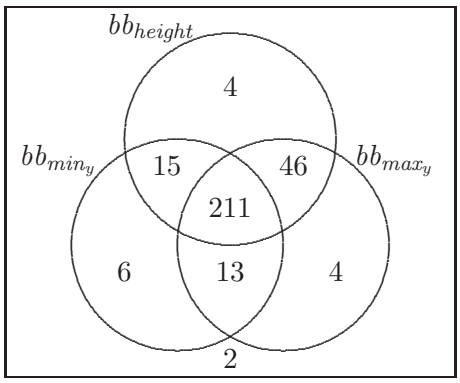
The six bounding box measures are parsed into a succession of peaks in the same way as the motion trace, described above. Figure 6.16 shows the distribution of peak-relative locations for ground truth editpoints, aggregated over three sequences. The figure illustrates that in most cases, the distribution is quite close to uniform—in other words, many edit points are not located near extrema in the bounding box traces. There are moderate peaks at 0 and 1 in most of the distributions, suggesting that between 10 and 20 percent of edit points are located at local extrema. It may be speculated that these should be amongst the more visually arresting edit points.



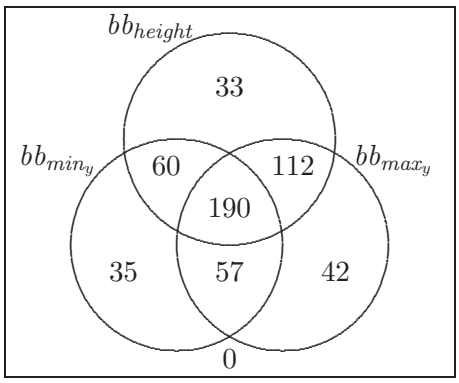
(a) Bounding box trace matches



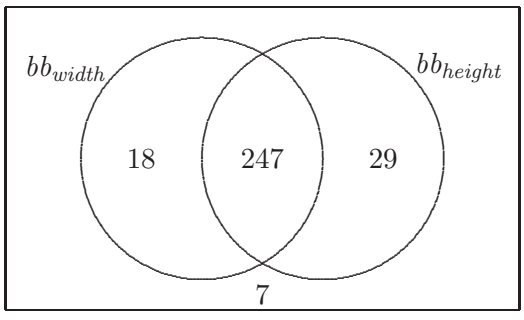
(b) Bounding box trace misses



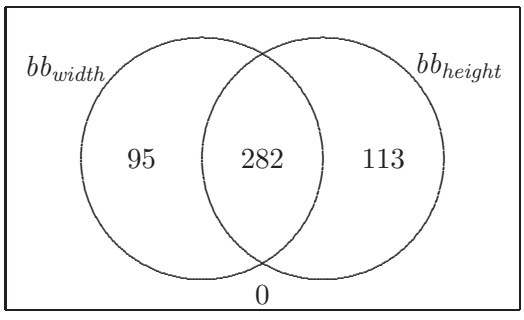
(c) Bounding box trace matches



(d) Bounding box trace misses

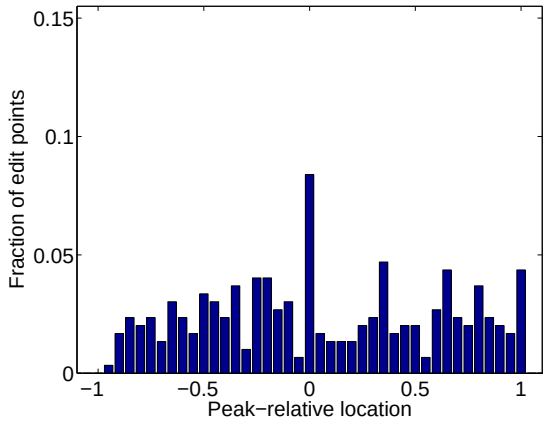


(e) Bounding box trace matches

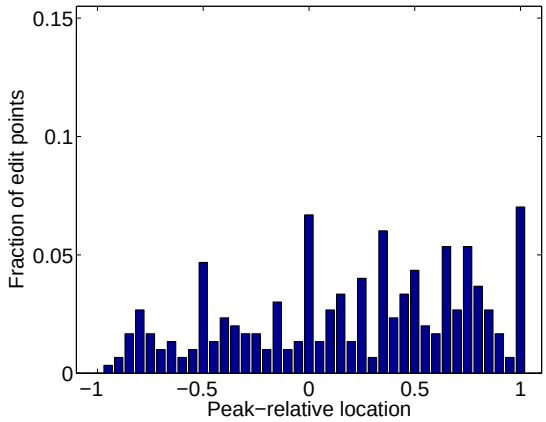


(f) Bounding box trace matches

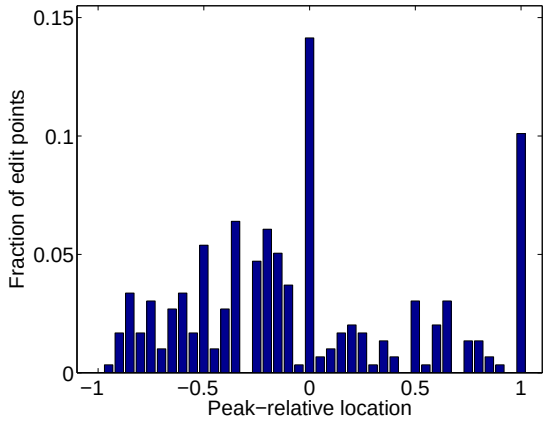
Figure 6.15: Overlap in edit point detection and false alarms for the bounding box traces, over the *greenDancer*, *maleDancer*, and *ballet2* sequences.



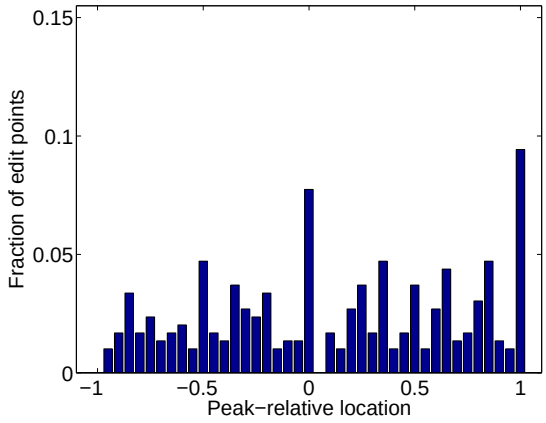
(a) Left edge trace



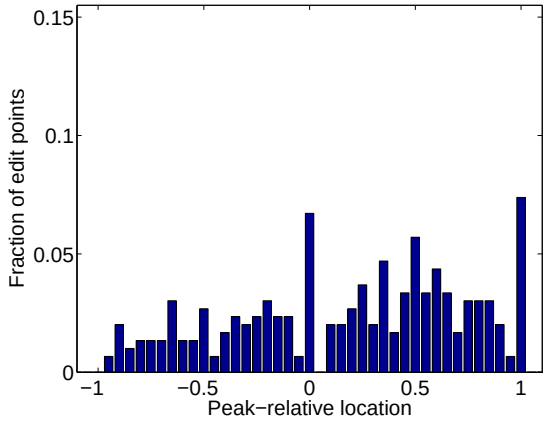
(b) Right edge trace



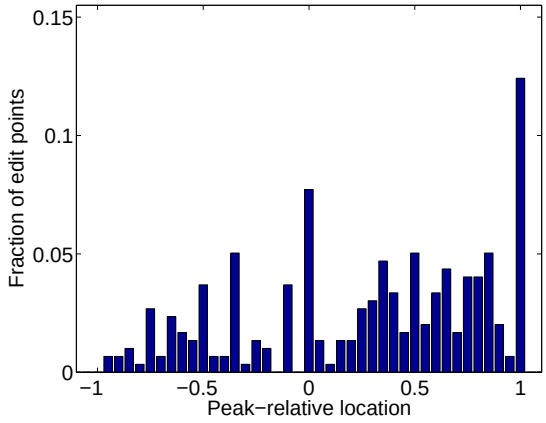
(c) Top edge trace



(d) Bottom edge trace



(e) Width trace



(f) Height

Figure 6.16: Peak-relative location distributions for edit points relative to peaks in the foreground region bounding box traces, in the *greenDancer*, *maleDancer*, and *ballet2* sequences combined.

bb_{min_x}	0.51895	bb_{max_x}	0.42776	bb_{min_y}	0.51843
bb_{max_y}	0.49422	bb_{width}	0.43799	bb_{height}	0.41972

Table 6.4: Prior probability of a peak's corresponding to an edit point for the bounding box traces.

6.3.5 Classifying bounding box trace peaks

As in the case of the motion trace, classifying the peaks in each trace as either desired or undesired should improve edit point detection. To this end, the distributions of the four peak characteristics introduced previously have been examined for desired and undesired peaks in each of the six traces. Here, a desired peak is one which contains a ground truth edit point near its maximum point or either minimum, i.e. those peaks for which

$$\begin{aligned} &\exists(i).\mathcal{G}(i) \in [\zeta_{max} - 0.1\zeta_{ascentwidth}, \zeta_{max} + 0.1\zeta_{descentwidth}] \\ \vee&\exists(i).\mathcal{G}(i) \in [\zeta_{leftmin}, \zeta_{leftmin} + 0.1\zeta_{ascentwidth}] \\ \vee&\exists(i).\mathcal{G}(i) \in [\zeta_{rightmin} - 0.1\zeta_{descentwidth}, \zeta_{rightmin}] \end{aligned} \tag{6.32}$$

The same four peak characteristics are used as for the *motion* trace, with the variation that here the peak noise characteristic is assessed over the entire peak range rather than over the descent only. The distributions found are presented in appendix C.1.2. The prior probabilities for each trace are shown in table 6.4.

The histograms show that the peak features have very similar distributions in both the desired and undesired classes in all six bounding box traces. The similarity of the classes makes distinguishing between them a challenging undertaking. Classification performance is presented in table 6.5. As in table 6.2, the performance of classifiers using each characteristic singly is presented, followed by the best combination of classifiers with product-likelihood, and the best combination with best-likelihood. In all cases, Parzen density estimation is used to represent the peak characteristic distributions.

In the case of most traces, the best scoring single characteristic is the peak symmetry, while the noise and relative descent characteristics offer comparatively poor performance. The best performance overall is achieved in the bb_{min_x} trace, with an area under the ROC curve of 0.77.

Performance of edit point detection using the bounding box traces after peak classification is not assessed here. However, the traces and classification scores will be employed in a method combining all traces for edit point detection, described in section 6.7.

Characteristics	ROC Area
0001	0.656869
0010	0.670318
0100	0.581573
1000	0.492788
1111	0.777469
0010	0.670318

(a) bb_{min_x}

Characteristics	ROC Area
0001	0.657609
0010	0.619706
0100	0.611278
1000	0.524403
0011	0.694991
0001	0.657609

(b) bb_{max_x}

Characteristics	ROC Area
0001	0.633026
0010	0.670392
0100	0.643503
1000	0.526547
0011	0.734637
0010	0.670392

(c) bb_{min_y}

Characteristics	ROC Area
0001	0.582663
0010	0.660786
0100	0.547858
1000	0.607486
1011	0.689373
0010	0.660786

(d) bb_{max_y}

Characteristics	ROC Area
0001	0.609192
0010	0.649232
0100	0.583862
1000	0.639986
1011	0.706823
0010	0.649232

(e) bb_{width}

Characteristics	ROC Area
0001	0.606779
0010	0.623221
0100	0.573015
1000	0.524777
1111	0.684614
0010	0.623221

(f) bb_{height}

Table 6.5: Peak classification performance for foreground bounding box traces: area under ROC curve. Again, combinations of peak characteristics are represented as a four-digit binary number whose digits, from left to right, correspond to ζ_{noise}, $\zeta_{reldesc}$, $\zeta_{symmetry}$, and $\log(\zeta_{descent})$.

Figure 6.17: An edit point and the local motion vectors pertaining to it. The change of direction in the dancer's foot is clearly reflected in the vector field in that region. The vectors shown are backward motion, at a blocksize of 17x17.

6.4 Motion Estimation Based Edit Point Identification

Local motion estimation, introduced in chapter 2, assigns a motion vector to each block in a video frame, intended to indicate the position of the corresponding block in an adjacent frame. This analysis yields considerable information regarding the motion content of the sequence, which could be expected to be of use in identifying edit points. Figure 6.17 shows how a change in direction of a dancer's limb is reflected in the associated vector field.

Seven measures are extracted from the vector fields for each sequence. The first is the mean vector magnitude, with gives an alternative measure of the amount of motion in each frame (alternative to the *motion* trace, which uses motion detection). Where this measure descends to a low value, a drop in the amount of motion in the sequence is expected—perhaps indicating a sudden stop edit point in the sequence. This trace is designated vf_M.

The relationship between the mean vector magnitude in a frame and the amount of motion is clear. However, in some cases an individual component of the vector can be more informative than the overall vector magnitude. For example, a dancer's 'bounce' may involve a transition from moderate-magnitude positive y vector components, to moderate-magnitude negative y components. This change in direction need not be indicated in the vf_M trace, and so the mean x and y components of the vector field are also considered, in traces designated vf_x and vf_y. These changes of direction will appear as zero crossings in the traces. Points where either

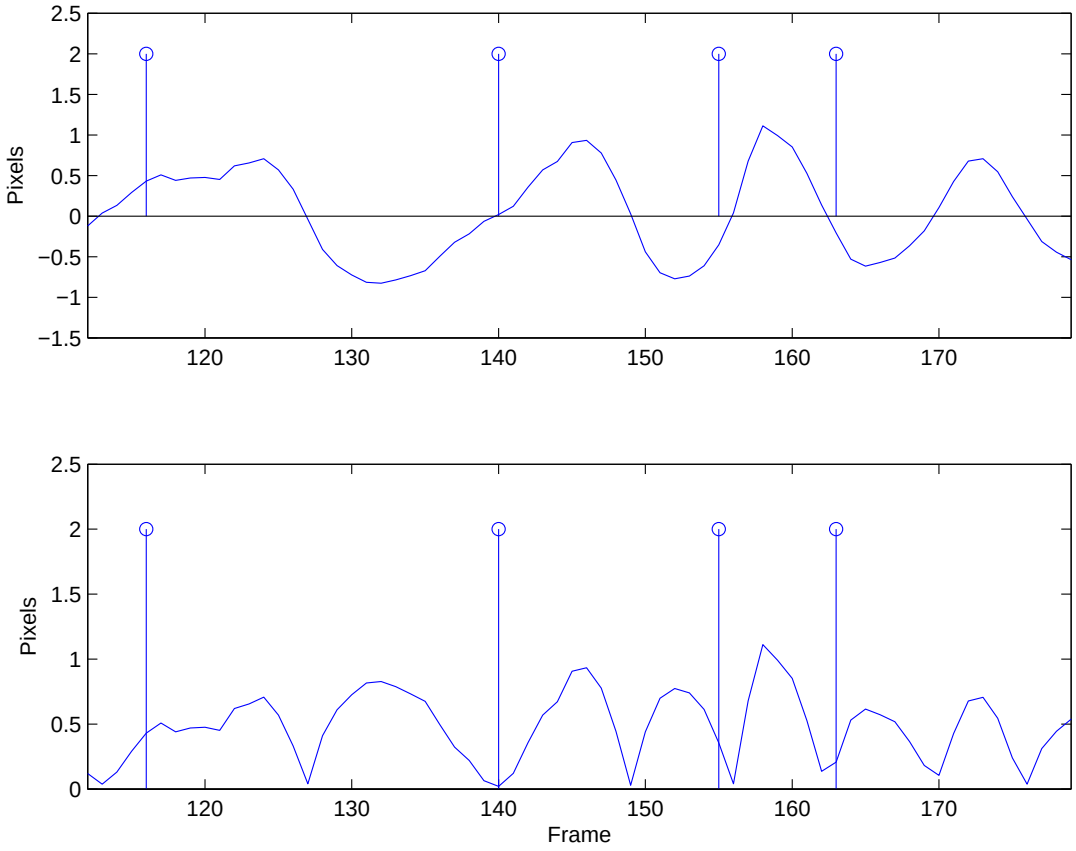


Figure 6.18: Taking the absolute value of vector component values converts zero crossings to peak minima. The trace shown is the vf_y trace over some 70 frames of the *ballet2* sequence; the upper trace is the original signal, and the lower shows the full-wave rectified version. The vertical stems mark ground-truth edit points.

component declines to zero are also of interest, so the traces are calculated using the absolute mean magnitude. Both such points of interest then correspond to peak endpoints, as in other traces. Figure 6.18 illustrates this correspondence.

A second way to incorporate direction changes in the vector field is to employ the mean vector angle; this trace is designated vf_θ. In the case of this trace, changes in the angle are of interest, and so both maxima and minima can be considered candidate edit points.

Finally, because most of the vectors in a vector field for a sequence depicting a single dancer will be background vectors, with a magnitude of zero, the variance of the vector field magnitude as a whole is at a minimum when the local motion vectors are also close to zero. The three magnitude traces, then, are supplemented by the variances of the same measures. The resulting three traces are designated $vf_{\sigma^2_M}$, $vf_{\sigma^2_x}$, and $vf_{\sigma^2_y}$, and correspond to the variance of the vector magnitudes, vector x components, and vector y components, over the entire vector field. These may be expected to offer more robust detection of sudden stops than the magnitude measures.

6.4.1 Trace extrema as edit points

As a first measure of the usefulness of the vector field traces in edit point identification, the performance of vector field trace extrema as edit point detectors is assessed. For the vf_θ trace, both trace minima and maxima are considered; in all other cases, trace minima only are selected. Each of the eight smoothing methods was evaluated, and the results are presented in appendix C.2.1. The best smoothing scheme overall uses Savitzky-Golay smoothing, relative difference gating, and median filtering of the sign of the first derivative. All the vector field traces achieve mean precision / recall performance in the 0.60 to 0.70 range.

6.4.2 Mutual redundancy between vector field traces

A second initial assessment is to determine the overlap amongst the seven traces derived from the motion vector field. This is assessed in a similar manner to the bounding box traces. The results are shown in figure 6.19. It can be seen that each trace discovers some unique edit points, and that the false alarm sites are moderately disjoint.

6.4.3 Peaks in the vector field traces

The traces are smoothed and parsed into peaks in the same way as the motion trace, described above. To confirm the intuitions presented above, the location of ground truth edit points relative to peaks in the traces was investigated. The results are presented in figure 6.20. The top row shows that edit points are indeed associated with minima in the vf_M, vf_x, and vf_y traces. Figure 6.20 (d) shows that for the vf_θ trace, edit points are most likely to be found at changepoints in the signal—i.e. peak maxima and minima. Lastly, in the bottom three plots, it is observed that in the $vf_{\sigma_M^2}$, $vf_{\sigma_x^2}$, and $vf_{\sigma_y^2}$ traces, edit points are more likely to occur further away from peak maxima.

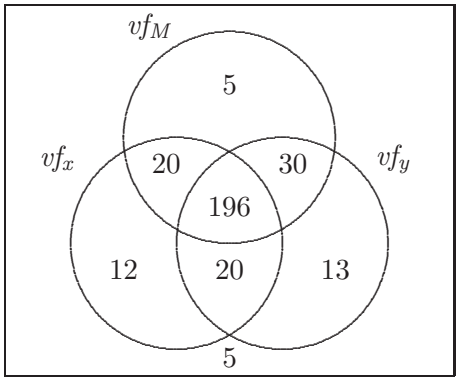
6.4.4 Classifying vector field trace peaks

Peak classification is also attempted for each of the seven traces. The distributions found for the peak characteristics are shown in appendix C.2.2. For the vf_M, vf_x, and vf_y traces, a peak is classified as desired if there is an edit point near the right minimum, i.e. if

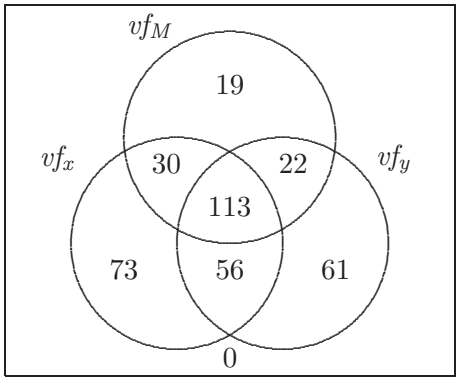
$$\exists(i).\mathcal{G}(i) \in [\zeta_{rightmin} - 0.1\zeta_{descentwidth}, \zeta_{rightmin} + 3] \tag{6.33}$$

In the case of the vf_θ trace, the criterion is

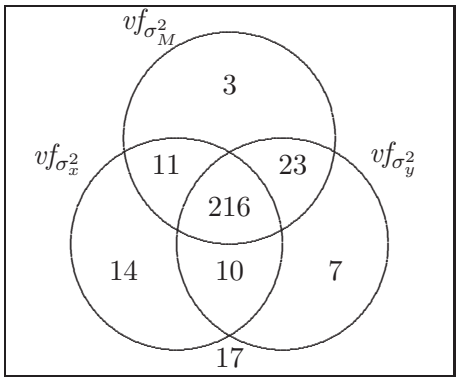
$$\begin{aligned} &\exists(i).\mathcal{G}(i) \in [\zeta_{max} - 0.1\zeta_{ascentwidth}, \zeta_{max} + 0.1\zeta_{descentwidth}] \\ \wedge&\exists(i).\mathcal{G}(i) \in [\zeta_{leftmin}, \zeta_{leftmin} + 0.1\zeta_{ascentwidth}] \\ \wedge&\exists(i).\mathcal{G}(i) \in [\zeta_{rightmin} - 0.1\zeta_{descentwidth}, \zeta_{rightmin}] \end{aligned} \tag{6.34}$$



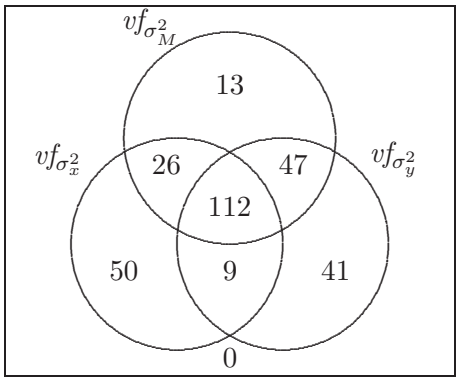
(a) Edit point matches



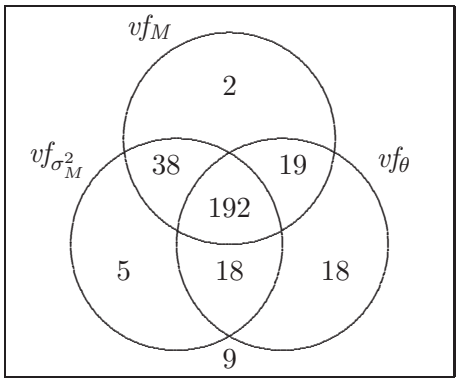
(b) False alarms



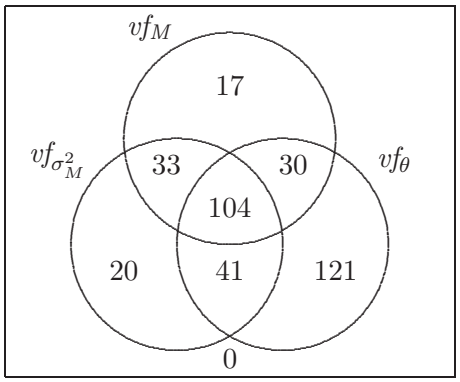
(c) Edit point matches



(d) False alarms

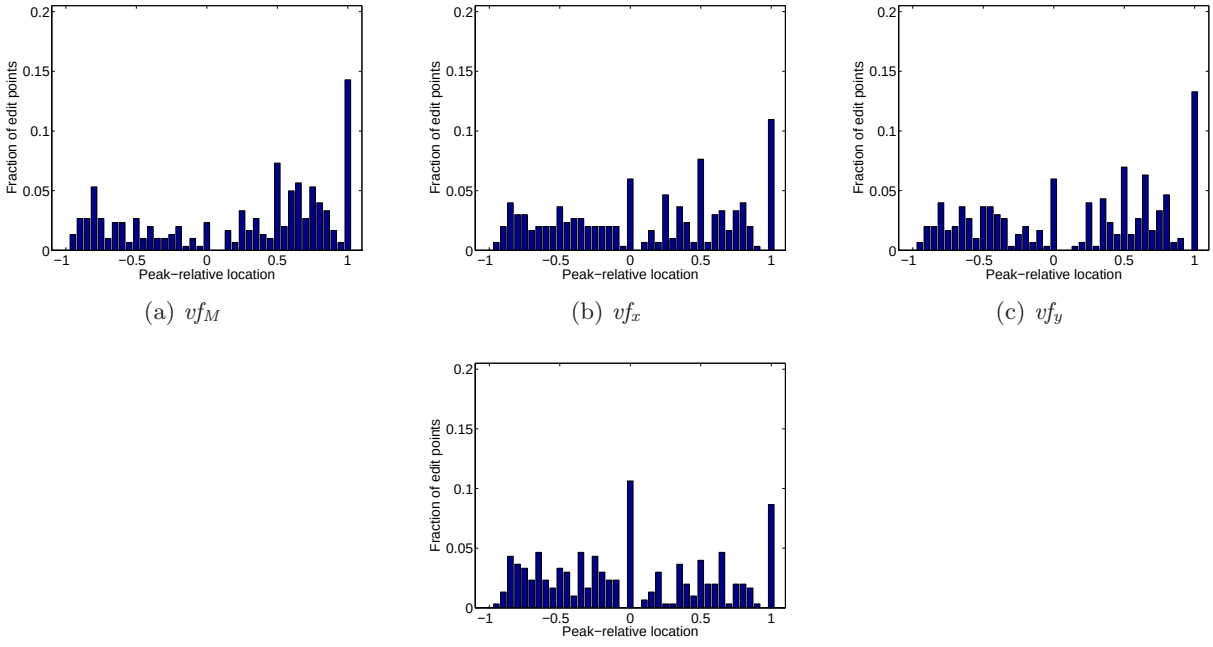


(e) Edit point matches



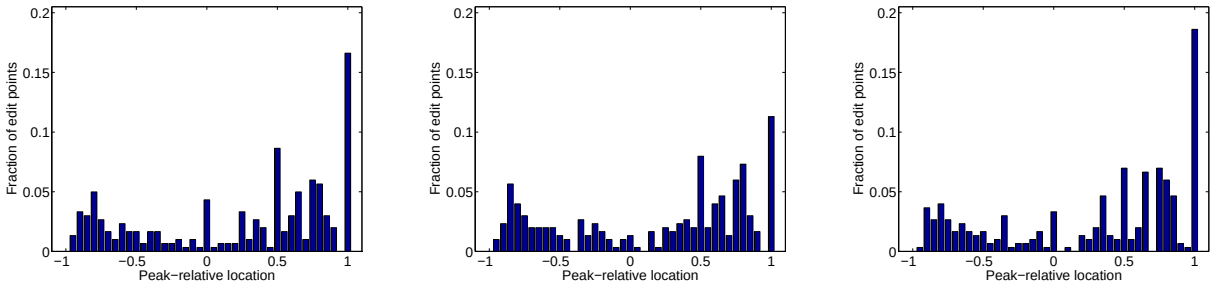
(f) False alarms

Figure 6.19: Overlap in edit point detection and false alarms for vector field traces, over the *greenDancer*, *maleDancer*, and *ballet2* sequences.



(a) vf_M (b) vf_x (c) vf_y

(d) vf_θ



(e) $vf_{\sigma^2_M}$ (f) $vf_{\sigma^2_x}$ (g) $vf_{\sigma^2_y}$

Figure 6.20: Peak-relative locations of edit points in the vector field traces, for the *greenDancer*, *maleDancer*, and *ballet2* sequences combined.

Figures 6.20 (e)-(g) show that in the case of the $vf_{\sigma^2_M}$, $vf_{\sigma^2_x}$, and $vf_{\sigma^2_y}$ traces, the probability of the peak-relative location of an edit point is quite well correlated with increasing distance from the peak maximum. For these traces, then, a peak that contains an edit point anywhere within its support is considered desired, and edit-point detection biased away from the maxima of these peaks in the section on Combined Edit Point Detection, below. Classification performance is shown in table 6.7. The highest performance is achieved by the vector field variance traces.

vf_M	0.28539	vf_x	0.22364	vf_y	0.25411
$vf_{\sigma^2_M}$	0.58606	$vf_{\sigma^2_x}$	0.55983	$vf_{\sigma^2_y}$	0.57971
vf_θ		0.3557			

Table 6.6: Prior probabilities of a peak's corresponding to an edit point for the vector field traces.

Characteristics	ROC Area
0001	0.681342
0010	0.586037
0100	0.581115
1000	0.66962
1011	0.771903
0001	0.681342

(a) vf_M

Characteristics	ROC Area
0001	0.675285
0010	0.58881
0100	0.606326
1000	0.592294
0111	0.705253
0001	0.675285

(b) vf_x

Characteristics	ROC Area
0001	0.67234
0010	0.598405
0100	0.603507
1000	0.584796
1111	0.698717
0001	0.67234

(c) vf_y

Characteristics	ROC Area
0001	0.635128
0010	0.64456
0100	0.590275
1000	0.544593
1011	0.721124
0010	0.64456

(d) vf_θ

Characteristics	ROC Area
0001	0.793779
0010	0.651292
0100	0.745439
1000	0.613479
1111	0.853028
0001	0.793779

(e) $vf_{\sigma^2_M}$

Characteristics	ROC Area
0001	0.765647
0010	0.641531
0100	0.712049
1000	0.60677
1111	0.823331
0001	0.765647

(f) $vf_{\sigma^2_x}$

Characteristics	ROC Area
0001	0.765816
0010	0.651764
0100	0.683894
1000	0.613533
1111	0.807915
0001	0.765816

(g) $vf_{\sigma^2_y}$

Table 6.7: Peak Classification Performance: Area Under ROC Curve

6.5 Motion Blur and Edit Point Identification

A prominent feature of many images corresponding to edit points is that they exhibit much less motion blur than images depicting dance phrases in execution. Assessing the level of motion blur in each frame, then, could be a powerful aid to the detection of edit points. The frames in figure 6.22 illustrate this phenomenon; the dancer's foot is most sharply resolved in the middle image, which is suitable as an edit point separating the leg's ascent from its descent.

Assessing images for blur has been an active area research in digital image processing for some decades. Most work focuses on blur due to defects in the imaging system, such as when the objective lens is out of focus [326], or on blur arising due to camera motion [62, 234]. Fewer systems targeting motion blur have been described.

For this work, relatively simplistic approaches involving the AC energy of the Fourier spectrum have been employed. The specific approach has to be tailored to the sequence in question, depending on the process chain involved in its preparation.

6.5.1 Motion blur in interlaced footage

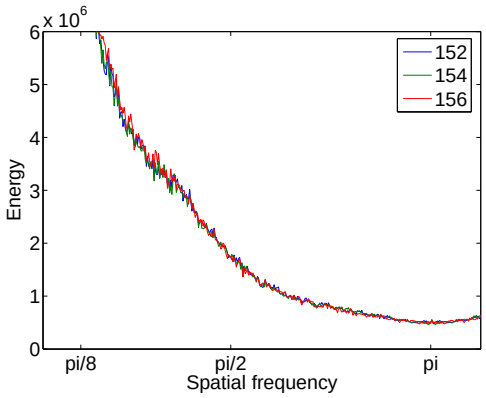
Video cameras typically produce *interlaced* footage. Each frame of the video is in fact two separate half-height fields, interlaced so that one field supplies the even lines, and the other the odd lines. Exposure times for the individual fields are typically about 1/250 to 1/1000 of a second, and so motion blur within each field is rare. However, the fields are sampled at twice the frame rate, so the fields in one frame are separated by 1/50 or 1/60 of a second. Fast motion is then clearly visible in the disparity between fields, introducing interlacing artifacts. When the video is played back, these interlacing artifacts are perceived as a kind of motion blur.

Detecting stop frames in interlaced footage, then, can be approached by seeking frames in which the effects of interlacing are locally least visible. As the fields are interleaved line-by-line, the disparities due to interlacing are perceived as lines in the image at very high vertical frequencies. Figure 6.21 shows how the Fourier energy is distributed over horizontal (figure 6.21 (a)) and vertical (figure 6.21 (b)) frequencies, for three frames. For all three frames, high energy in vertical frequencies near π radians / sample is apparent, corresponding to interlacing artifacts. Furthermore, this energy is least for the desired middle frame, corresponding to the green trace.

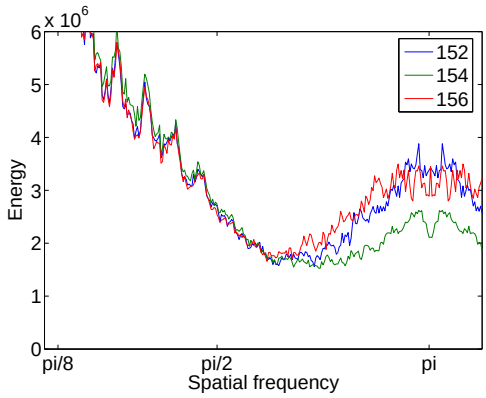
To quantify this effect, the relative energy at high vertical frequencies is defined. This is given by

$$freq_{interlace}(n) = \frac{\sum_{x=0\ldots\frac{\pi}{2};y=\frac{\pi}{2}\ldots\pi} FE_n(x,y)}{\sum_{x=0\ldots\frac{\pi}{;}y=0\ldots\pi} FE_n(x,y)} \tag{6.35}$$

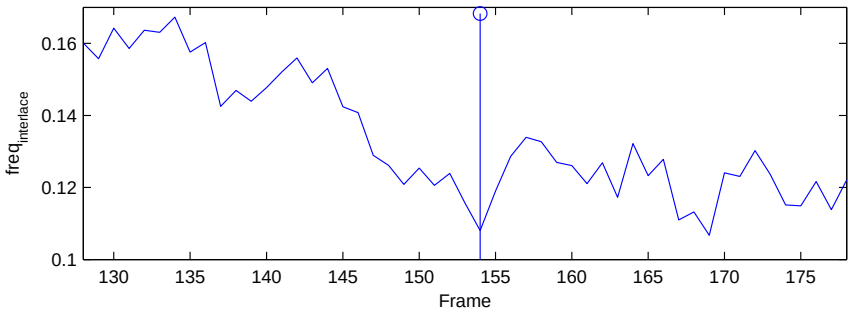
where $FE_n(x,y)$ is the Fourier spectrum magnitude at spatial frequency (x,y) for frame n.



(a) Spectrum distribution over horizontal frequencies



(b) Spectrum distribution over vertical frequencies



(c) Amount of interlacing per frame.

Figure 6.21: Sharp frame detection for interlaced footage. Frames 152, 154, and 156 from the *ballet2* sequence are shown at top; frame 154 is the desired sharpest image.

6.5.2 Motion blur in non-interlaced footage

Film cameras operate at 24 frames per second, and typically use exposure times of 1/50 to 1/200. This can be sufficiently slow for motion blur to be visible in the resulting frames. However, when material shot on film is to converted to NTSC video, temporal resampling from 24 fps to 60 fields per second is necessary. The resampling scheme used is called 3:2 pulldown [164], and it results in video in which interlacing artifacts are introduced to certain frames. In straight 3:2 pulldown conversion, two out of every five frames will be interlaced. However, variations in the transfer and encoding scheme can mean that this pattern is not always observed. The *greenDancer* and *maleDancer* sequences, for example, show evidence of having been converted from film, where slight interlacing artifacts and higher noise values are apparent in every third frame.

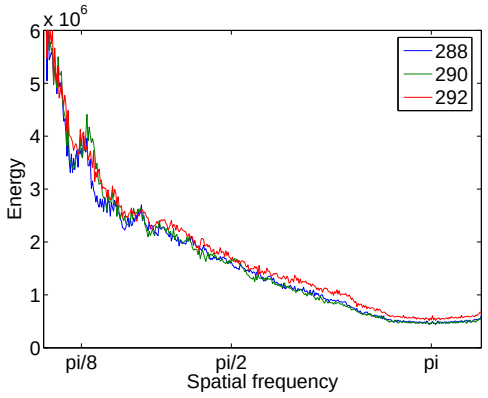
The approach adopted is to assess the image sharpness by examining the energy at non-zero spatial frequencies in the Fourier spectrum of each frame. This results in the $freq_{sharp}$ trace, found by

$$freq_{sharp}(n) = \sum_{x \neq 0 \vee y \neq 0} FE_n(x, y) \tag{6.36}$$

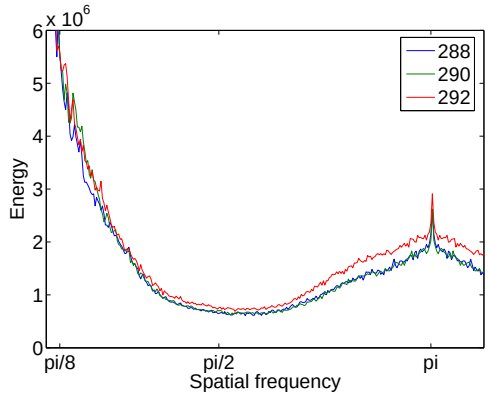
However, some smoothing of this trace may be necessary depending on the production chain of the video in question. An approach appropriate to the *greenDancer* and *maleDancer* sequences is illustrated here. Figure 6.22 (a) shows the energy of the Fourier spectrum of three frames, summed along columns. This shows how the energy is distributed among horizontal frequencies; the green trace corresponds to the middle frame (containing the desired edit point), and the peak at $\pi/8$ can be assumed to correspond to the increased resolution due to the stop in the region of the shoe. Note that the red trace is slightly higher than the other two at all frequencies. Figure 6.22 (b) shows how Fourier energy is distributed over vertical frequencies. The peaks at π in the vertical energy spectrum common to all three traces suggest that some interlacing artifacts are still present. Again, the red trace has more energy at all frequencies, but particularly at the Nyquist rate, characteristic of interlacing artifacts. In this sequence, every third frame has markedly higher high frequency energy, resulting in the noise pattern shown in figure 6.22 (c). Every third value is a spike, including frame 292. To discount the effects of this phenomenon, the affected values are replaced with the average value of their two neighbours. The underlying trend in the AC energy of the video frames is then visible. The edit point shown is at frame 290, and this is the highest value in the smoothed trace—in other words, this frame is detected as a notably sharp image.

6.5.3 Extrema detection in the motion blur traces

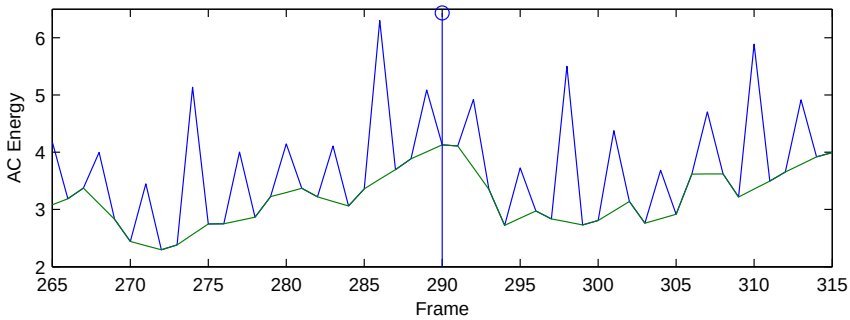
To assess the correspondence of edit points to extrema in the motion blur traces, edit point detection was assessed for each trace. The maxima of the $freq_{sharp}$ trace are used as candidate edit points, while for the $freq_{interlace}$ trace the minima are used. Minima detection was assessed



(a) Spectrum distribution over horizontal frequencies

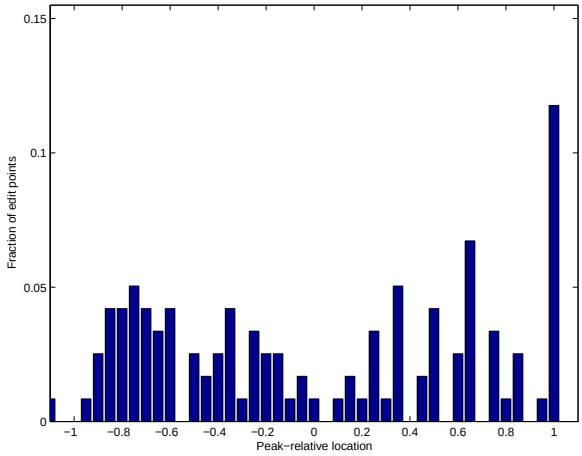


(b) Spectrum distribution over vertical frequencies

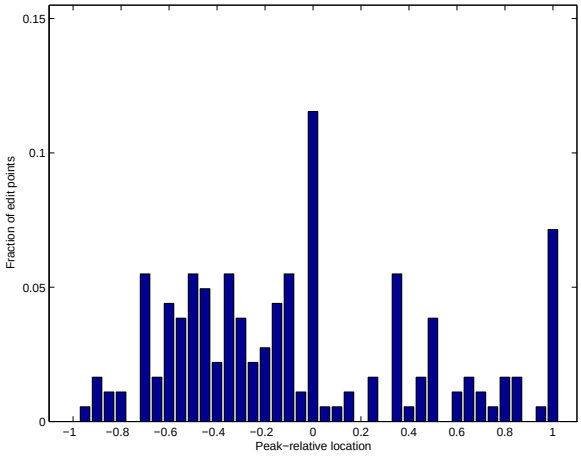


(c) Total AC energy per frame.

Figure 6.22: Sharp frame detection for 3:2 pulldown footage. Frames 288, 290, and 292 from the *greenDancer* sequence are shown at top; frame 290 is the desired sharpest image.



(a) Edit point location relative to peaks in the $freq_{interlace}$ trace, for the *ballet2* sequence



(b) Edit point location relative to peaks in the $freq_{sharp}$ trace, for the *greenDancer* and *maleDancer* sequences combined.

Figure 6.23: Edit point location relative to peaks in the motion blur traces.

for each of the eight smoothing schemes described; the results for edit point detection using these minima are presented in appendix C.3.1. For the $freq_{sharp}$ trace, no smoothing gives the best performance. Savitzky-Golay filtering is applied to the $freq_{interlace}$ trace. In both cases, recall is good but precision is poor, at 0.24 and 0.19 for the two traces.

6.5.4 Peaks in the motion blur traces

As in the case of the other traces, the $freq_{interlace}$ and $freq_{sharp}$ traces can be parsed into peaks and these peaks used to isolate trace features proposed for edit points. Edit points are expected to occur at minima in the case of the $freq_{interlace}$ trace and maxima in the case of the $freq_{sharp}$ trace. Figure 6.23 shows that these are the most probable peak-relative locations for edit points in these traces.

To establish the ground truth for purposes of peak classification, a peak $\zeta(p)$ in the $freq_{interlace}$ trace is designated as desired if

$$\exists(i).\mathcal{G}(i) \in [\zeta_{max} + 0.9\zeta_{descentwidth}, \zeta_{rightmin} + 3] \tag{6.37}$$

For the $freq_{sharp}$ trace, a peak is desired if

$$\exists(i).\mathcal{G}(i) \in [\zeta_{max} - 0.1\zeta_{ascentwidth}, \zeta_{max} + 0.1\zeta_{descentwidth}] \tag{6.38}$$

6.5.5 Classifying motion blur peaks

Classification of the peaks in the motion blur traces is undertaken using the same four peak characteristics described previously. The distributions found are shown in C.3.2. The prior

Characteristics	ROC Area
00 0001	0.609678
0010	0.520234
0100	0.615032
1000	0.367559
1111	0.669115
0101	0.615032

(a) $freq_{interlace}$

Characteristics	ROC Area
0001	0.576721
0010	0.532287
0100	0.545496
1000	0.598621
1000	0.598621
1000	0.598621

(b) $freq_{sharp}$

Table 6.8: Peak classification performance for the motion blur traces.

probability that a peak in the $freq_{sharp}$ trace corresponds to an edit point was found to be 0.076. For the $freq_{interlace}$ trace, the prior probability was 0.3319.

Table 6.8 presents the results of peak classification performance for the $freq_{interlace}$ and $freq_{sharp}$ traces. Again, it is apparent that classifying peaks in these traces is a challenging undertaking. The motion blur traces are incorporated in the combined method approach to edit point detection, described below.

6.6 The Video Soundtrack and Edit Point Identification

The audio track of videos can be a rich source of information regarding edit points in the video. For example, in video of dance accompanied by music, beat locations in the music could be expected to be temporally close to dance phrase transitions. Also, many percussive motions give rise to a simultaneous sound—the stamp of a dancer's foot, for example.

The audio analysis undertaken here is a simple characterisation of the audio energy associated with each frame. This is found by the mean of the samples associated with each frame after full-wave rectification of the waveform. Let $w(i)$ represent the original audio signal. The number of audio samples to each video frame is designated by $F = \frac{f_s}{F_r}$, the ratio of the audio sampling frequency f_s to the video framerate F_r. The audio energy trace, *audio*, is then found by

$$audio(n) = \frac{1}{F} \sum_{i=nF \ldots (n+1)F} |w(i)| \tag{6.39}$$

It is noted that modelling perceptual loudness of a waveform requires more complicated techniques than the approach described here [224]. For videos with musical accompaniment, beat detection could also be applied to the soundtrack [269]. The audio energy approach used here is found to assist in edit point detection in the videos assessed, and has the advantage of being applicable to videos with or without music.

Figure 6.24 shows an extract of the audio energy trace for the greenDancer sequence. The three edit points in these frames are all associated with peaks in the audio trace.

Characteristics	ROC Area
0001	0.612166
0010	0.657108
0100	0.650587
1000	0.632159
1111	0.754281
0010	0.657108

Table 6.9: Audio peak classification performance

6.6.1 Maxima in the audio trace

The optimal smoothing scheme for the *audio* trace was determined by evaluating edit point detection using all maxima in the smoothed signal. The results of this assessment are shown in appendix C.4.1. The best-performing smoothing scheme uses Savitzky-Golay filtering and relative difference gating.

6.6.2 Peaks in the audio trace

The audio energy trace is smoothed and parsed into peaks in the same way as the previously described traces. Figure 6.25 shows how peak-relative locations are distributed amongst edit points.

The ground truth criterion used for audio is that a peak is desired if

$$\exists(i).\mathcal{G}(i) \in [\zeta_{max} - 0.1\zeta_{ascentwidth}, \zeta_{max} + 0.1\zeta_{descentwidth}] \tag{6.40}$$

is suggested. In other words, peaks for which the peak maximum is close to an edit point are designated as desired.

6.6.3 Peak classification in the audio trace

The prior probability that a peak in the *audio* trace is desired was found to be 0.104. The same four peak characteristics used in previous traces are used for peak classification. The distributions found are presented in appendix C.4.2. Classification performance is shown in table 6.9. The best classifier uses all four characteristics, achieving an area under the ROC curve of 0.75.

6.7 Combined-Method Edit Point Identification

Thus far, this chapter has presented a number of disparate methods for detecting edit points. It has been established that none of these methods in isolation results in entirely satisfactory

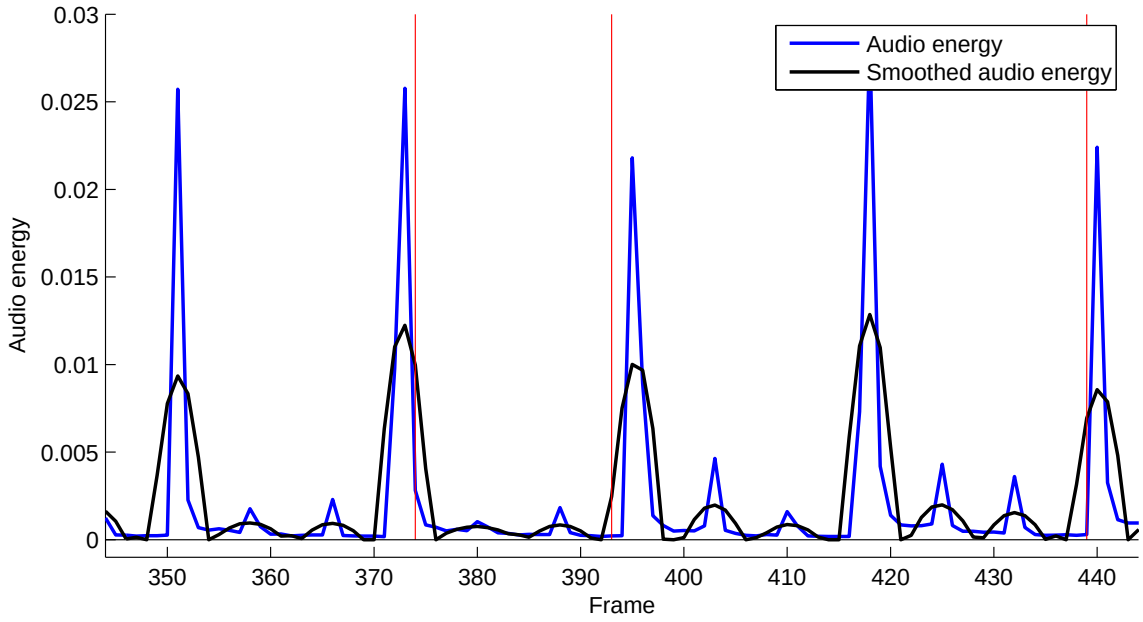


Figure 6.24: The audio energy signal over 100 frames in the *greenDancer* sequence. Ground-truth edit points are indicated by vertical red lines.

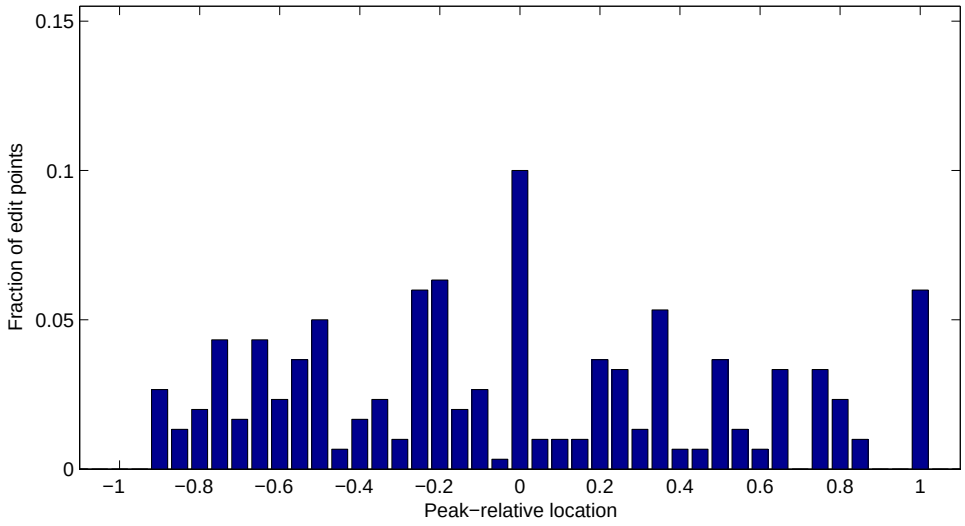


Figure 6.25: Peak-relative locations of edit points in the audio energy trace

Trace	No.	Features	W_R	W_ρ	$W_{R\rho}$
motion	0	1101	0.806	0.507	0.409
bb_{min_x}	1	1111	0.777	0.519	0.403
bb_{max_x}	2	1011	0.694	0.428	0.297
bb_{min_y}	3	1011	0.734	0.518	0.381
bb_{max_y}	4	1011	0.689	0.494	0.341
bb_{width}	5	1111	0.706	0.438	0.309
bb_{height}	6	1111	0.685	0.420	0.279
vf_M	7	1011	0.771	0.285	0.220

Trace	No.	Features	W_R	W_ρ	$W_{R\rho}$
vf_x	8	1111	0.705	0.234	0.158
vf_y	9	1111	0.698	0.254	0.178
vf_θ	10	1011	0.721	356	0.256
$vf_{\sigma^2_M}$	11	1111	0.853	0.586	0.500
$vf_{\sigma^2_x}$	12	0111	0.823	0.560	0.461
$vf_{\sigma^2_y}$	13	1111	0.807	0.580	0.468
$freq_{interlace}$	13	1111	0.669	0.331	0.222
$freq_{sharp}$	13	1000	0.598	0.076	0.046
audio	13	1011	0.75	0.105	0.078

Table 6.10: Peak features for classification, area under classifier ROC (W_R), and prior probability of a peak's being desired (W_ρ), for each of the 17 traces. $W_{R\rho}$ is the product of W_R and W_ρ.

edit point detection. In this section, it is investigated whether these methods can be combined so as to obtain better overall performance.

The preceding material has introduced a set of traces, $\mathcal{T}$, comprising the motion trace, six bounding box traces, seven traces from the motion vectors, a motion blur trace (either $freq_{sharp}$, or $freq_{interlace}$, depending on the video format), and the audio trace. Each trace t is decomposed into a set of peaks $\mathcal{P}_t$, and it has been demonstrated that in each trace, edit point location is in some degree correlated with peak location—principally, whether edit points are more likely at peak minima or peak maxima in that trace.

In this section, a scheme for converting the results of this trace analysis to a probability signal for edit points is described. A trace $t(n)$ is converted to a signal $EP_t(n)$ which reflects the probability that frame n corresponds to an edit point. These individual probability signals can then be combined to generate an overall probability signal $EP(n)$, which incorporates analysis of all the traces described for edit point detection.

6.7.1 Converting Traces to Edit Point Signals

The first stage is the transformation of each trace into a signal reflecting the probability of an edit point at each frame given that trace alone. To achieve this, the traces are parsed into peaks, and each peak is classified as desired or undesired. A kernel $K_t(n, P)$ is then added to the edit point probability signal, centred on the appropriate peak feature location—for example, at the peak minimum—and scaled according to the $C_t(p)$, the confidence of the peak classification. Thus EP_t is found by

$$EP_t(n) \propto \sum_{p \in \mathcal{P}_t} C_t(p) K_t(n, p) \tag{6.41}$$

The form of the kernel $K_t(n, p)$ is different for each trace t.

The confidence measure was introduced in 6.2.6.1, and is based on the result of classification

of the peak p using the classification system developed in this chapter. For each trace, the combination of peak characteristics yielding the best classification performance, as measured by the area under the R.O.C. curve, has been identified. These combinations are shown in table 6.10. The confidence measure is based on the difference between the class energies

$$C_t(p) = \exp(E_t(p|\zeta(p) = 0) - E_t(p|\zeta(p) = 1)) \tag{6.42}$$

where $E_t(p|\zeta(p) = 0)$ is the negative logarithm of the likelihood that peak p is undesired, as described in 6.2.6. $C_t(p)$ is greater than one if $P(p|\zeta(p) = 0) < P(p|\zeta(p) = 1)$, and less than one otherwise. Thus the kernel is amplified or attenuated according to the classification result of the peak at hand.

Where a peak characteristic has a value close to the limit of the range, it is found that the ratio of the likelihood distributions can become very high. This results in a very high confidence measure, amplifying the kernel value by several hundred times in some cases. To avoid this effect, values of C_t can be limited to be less than some maximum. The choice of a suitable limit is described in the section on assessment later in this chapter.

6.7.1.1 Edit point probability using the motion trace

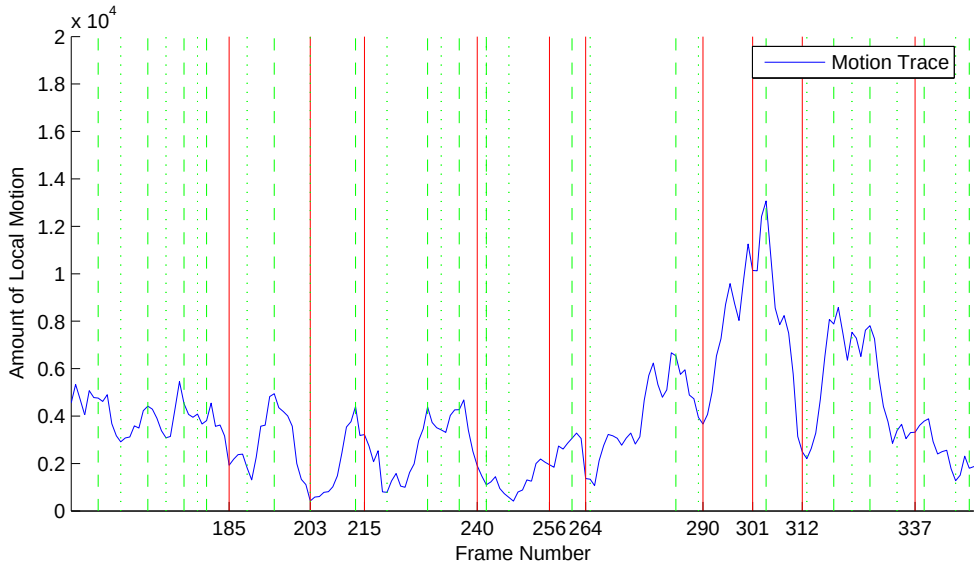
For the motion trace, edit points are expected near peak minima. The joint distribution $P(n, \frac{\zeta_{descent}(n)}{\zeta_{descentwidth}(n)})$ over edit point location relative to the peak minimum, and peak steepness, was presented in figure 6.12. The kernel for a peak in the motion trace $\zeta(p)$ is then

$$K'_0(n,p) \propto P(n - \zeta_{rightmin}(p)|\zeta_{descent}(p), \zeta_{descentwidth}(p)) \tag{6.43}$$

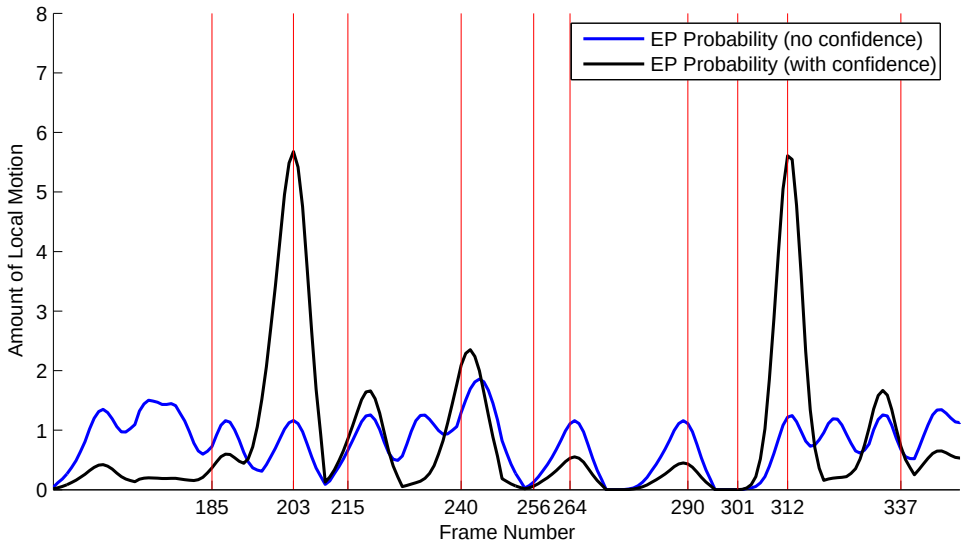
$$K_0(n,p) = \frac{K'_0(n,p)}{\max_{n'} K'_0(n',p)} \tag{6.44}$$

The kernel is normalised to have a peak value of 1.

Figure 6.26 illustrates how $EP(n)$ is calculated based on the motion trace alone. Figure 6.26 (a) presents the motion trace over a range of frames in the *greenDancer* sequence; figure 6.26 (b) then shows how $EP(n)$ is made up of kernels with their modes at the peak minima. The left tail of each kernel is longer for shallower peak descents. The figure also illustrates how scaling each kernel by the classification confidence improves the separation between false peak minima and desired peak minima for the edit points at frames 203, 240, and 312—although the peak classification has a detrimental effect as regards detection of the stops at frames 264 and 290. While not all ground truth edit points are at peaks in the $EP(n)$ signal, a strong correlation is evident.



(a) The *motion* trace over 200 frames of the *greenDancer* sequence. Peak maxima are shown by dashed green lines; peak minima are shown by dotted green lines. Red lines show the location of edit points in the ground truth.



(b) The edit point probability signal generated from the *motion* trace. Red lines show the location of edit points n the ground truth.

Figure 6.26: Edit point probability analysis using the *motion* trace

6.7.1.2 Edit point probability using the bounding box traces

In the six bounding box traces, edit points are expected near the extrema of the peak traces. The kernels are made up of three functions, based on the non-normalised Gaussian:

$$G(\mu, \sigma^2, n) = \exp\left(-\frac{1}{2\sigma^2}(n-\mu)^2\right) \tag{6.45}$$

right half-Gaussian (non-normalised):

$$G^r(\mu, \sigma^2, n) = \begin{cases} \exp\left(-\frac{1}{2\sigma^2}(n-\mu)^2\right), & x \geq 0 \\ 0 & \text{otherwise} \end{cases} \tag{6.46}$$

and left half-Gaussian (non-normalised):

$$G^l(\mu, \sigma^2, n) = \begin{cases} \exp\left(-\frac{1}{2\sigma^2}(n-\mu)^2\right), & x \leq 0 \\ 0 & \text{otherwise} \end{cases} \tag{6.47}$$

The Gaussians are not normalised as it is not desired that the confidence should increase for narrower variance. In this formulation, the kernel maximum in all cases is one, as for the *motion* trace kernel above.

The overall kernel for the bounding box traces is then

$$\begin{aligned} K_{1-6}(n,p) = &\frac{1}{3}G^r(\zeta_{leftmin}(p), \frac{\zeta_{ascentwidth}}{10}, n) \\ &+ \frac{1}{3}G(\zeta_{max}(p), \frac{\zeta_{ascentwidth} + \zeta_{descentwidth}}{20}, n) \\ &+ \frac{1}{3}G^l(\zeta_{rightmin}(p), \frac{\zeta_{descentwidth}}{10}, n) \end{aligned} \tag{6.48}$$

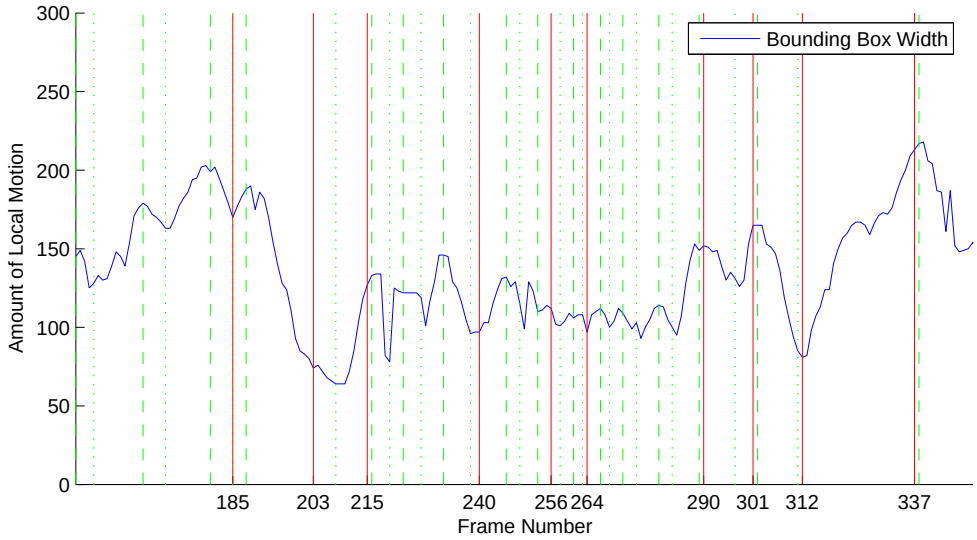
The width of the kernel increases with increasing peak width, corresponding to the ground truth criterion used in peak classification. Figure 6.27 illustrates how the edit point probability signal is derived from the bb_{width} trace. It can be seen that this trace has assigned a high probability to the ground truth edit point at frame 185, which was missed in the analysis of the *motion* trace.

6.7.1.3 Edit point probability using the vector field traces

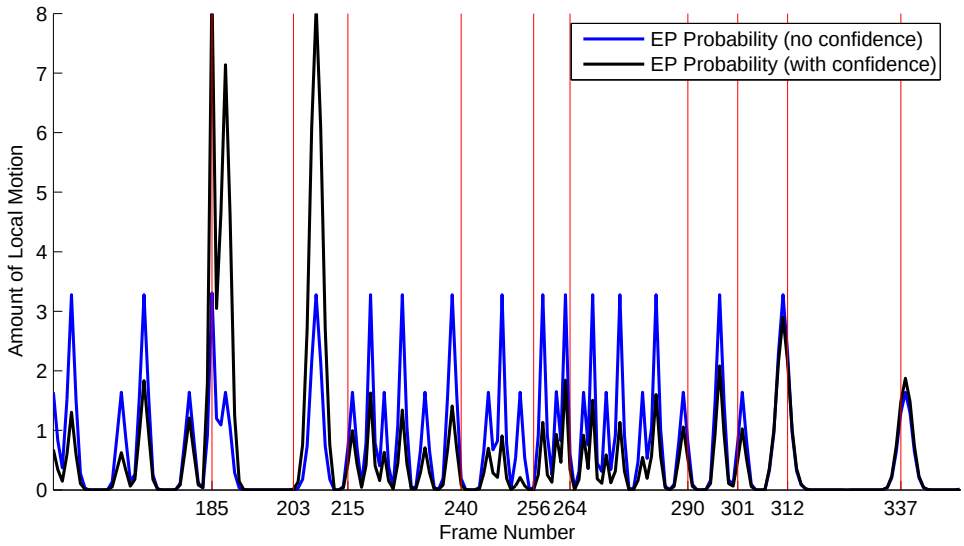
Similar kernels are employed for the vector field traces. For the vf_M, vf_x, and vf_y traces, edit points are expected near the peak minima, and so the kernel used is the left half-Gaussian

$$K_{7-9}(n,p) = G^l(\zeta_{rightmin}(p), \frac{\zeta_{descentwidth}}{10}, n) \tag{6.49}$$

where again the kernel width increases with the width of the peak. In the case of the vf_θ trace, edit points are correlated with changepoints, i.e. minima and maxima, and so the same kernel is used as for the bounding box trace—$K_{10}(n,p) = K_1(n,p)$. In the case of the vector field



(a) The bb_{width} trace over 200 frames of the *greenDancer* sequence. Peak maxima are shown by dashed green lines; peak minima are shown by dotted green lines. Red lines show the location of edit points in the ground truth.



(b) Edit point probability signal generated from the bb_{width} trace. Ground truth edit points are shown by red lines.

Figure 6.27: Edit point probability analysis using the bb_{width} trace

variance traces, $vf_{\sigma_M^2}$, $vf_{\sigma_x^2}$, and $vf_{\sigma_y^2}$, edit point location is to be biased away from the peak maximum, reflecting the cup shape visible in the distributions. The kernel used is then:

$$K_{11-13}(n,p) = \frac{1}{2}G^r(\zeta_{leftmin}(p), \frac{\zeta_{ascentwidth}}{2}, n) + \frac{1}{2}G^l(\zeta_{rightmin}(p), \frac{\zeta_{descentwidth}}{2}, n)$$

Figure 6.28 shows the edit point probability process in the case of the $vf_{\sigma_M^2}$ trace. Here many of the resulting peaks are slightly after ground truth edit points. The edit point at frame 301 has been accorded a small probability of detection, which could aid in detection of this frame in combination with the other traces.

6.7.1.4 Edit point probability using the motion blur traces

In the case of the traces reflecting motion blur, edit points are correlated with minima in the $freq_{interlace}$ trace and with maxima in the $freq_{sharp}$ trace. For interlaced footage, then, the kernel used is the left half-Gaussian

$$K_{14_i}(n,p) = G^l(\zeta_{rightmin}(p), \frac{\zeta_{descentwidth}}{10}, n) \tag{6.50}$$

and for 3:2 pulldown material, a Gaussian centred on the peak maximum is used

$$K_{14_s}(n,p) = G(\zeta_{max}(p), \frac{\zeta_{ascentwidth} + \zeta_{descentwidth}}{20}, 20) \tag{6.51}$$

Figure 6.29 shows the result of this analysis for a segment of the *greenDancer* sequence. This sequence has been derived from video footage via 3:2 pulldown, and so motion blur is assessed using the $freq_{sharp}$ trace in this case. Of note here is the high probability assigned to the edit point at frame 290, which has not featured in the previously illustrated traces.

6.7.1.5 Edit point probability using the audio trace

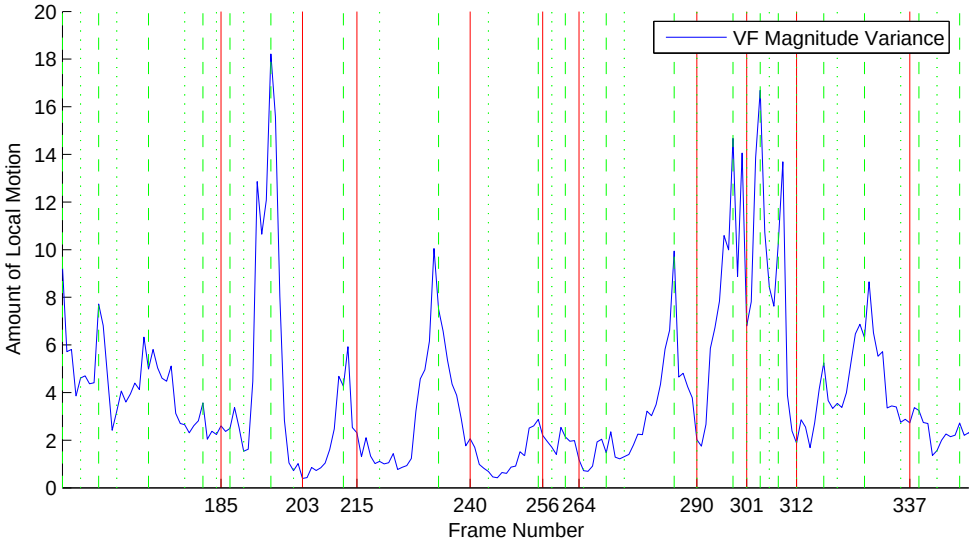
For the audio trace, edit points are expected near the peak maximum. The kernel employed is the same as that used for the $freq_{sharp}$ trace, so $K_{13} = K_{14_s}$. Figure 6.30 shows the edit point probability signal derived from the *audio* trace. While the edit points at 203, 240, and 290 are effectively detected, the high peak in probability at frame 329 is does not correspond to a defensible edit point.

6.7.2 Combining Probability Traces

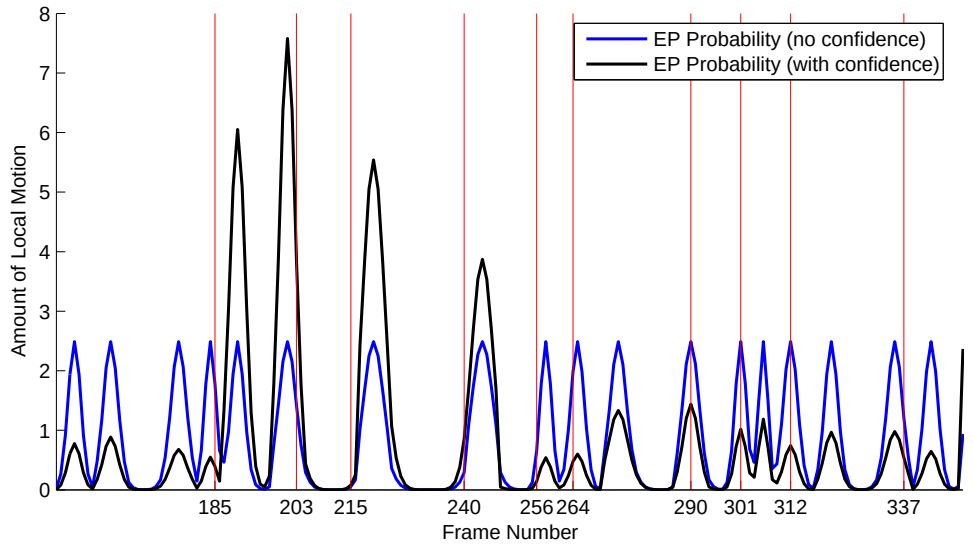
To form a combined probability signal for edit points at frame n, the weighted mean of the individual EP_t signals is then found using

$$EP(n) = \frac{\sum_{t\in\mathcal{T}} W_t EP_t(n)}{\sum_{t\in\mathcal{T}} W_t} \tag{6.52}$$

Maxima in this combined signal then correspond to detected edit points using all the traces described.

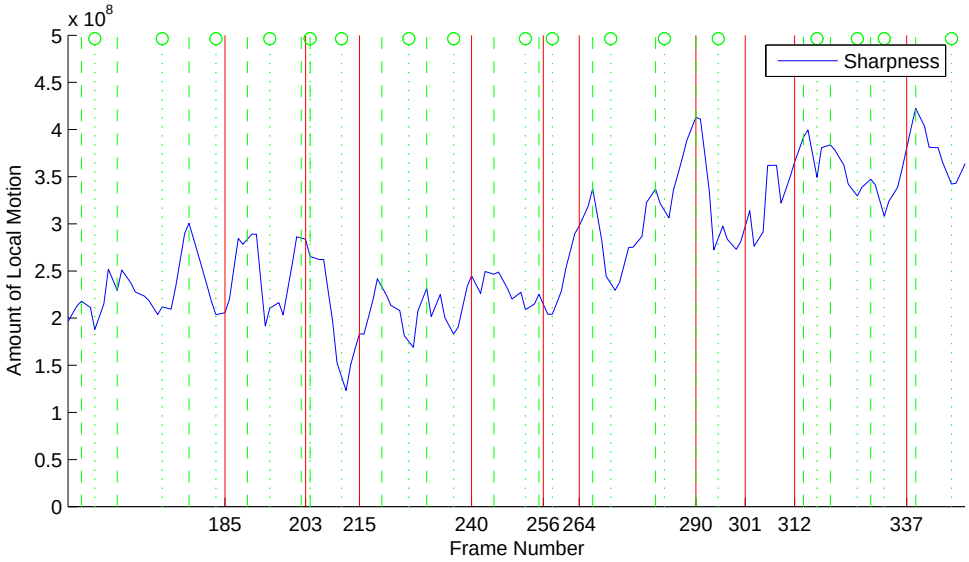


(a) The $vf_{\sigma^2_M}$ trace over 200 frames of the *greenDancer* sequence. Peak maxima are shown by dashed green lines; peak minima are shown by dotted green lines. Red lines show the location of edit points in the ground truth.

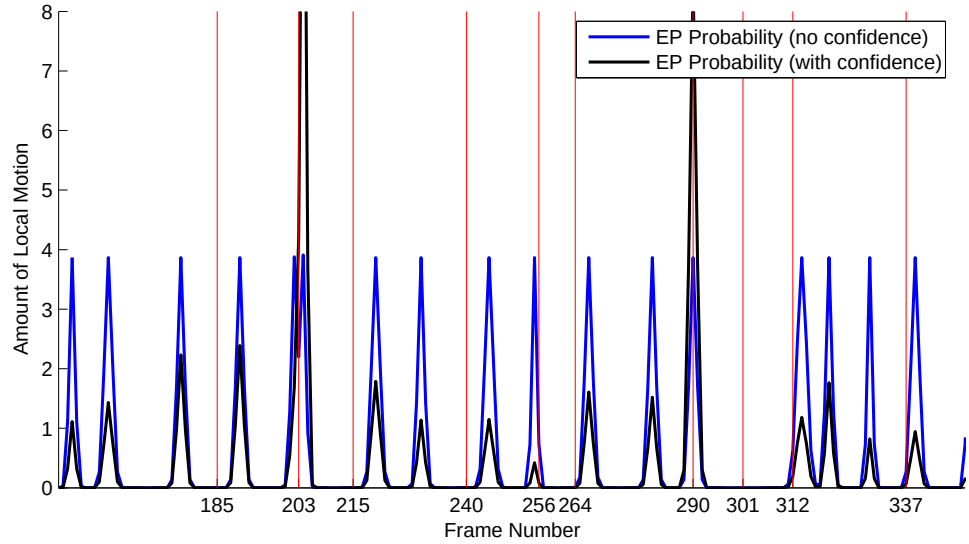


(b) Edit point probability signal generated from the $vf_{\sigma^2_M}$ trace. Vertical red lines show ground truth edit points.

Figure 6.28: Edit point probability analysis using the $vf_{\sigma^2_M}$ trace

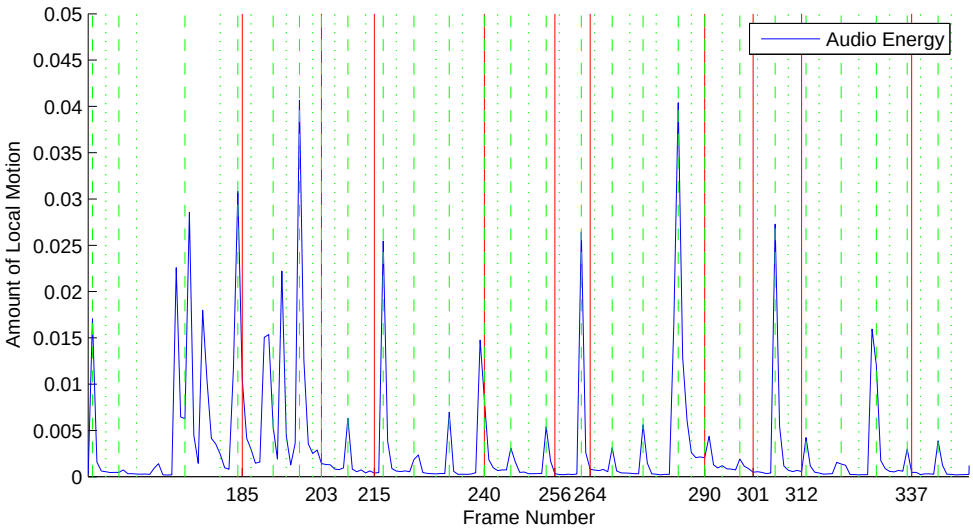


(a) The $freq_{sharp}$ trace over 200 frames of the *greenDancer* sequence. Peak maxima are shown by dashed green lines; peak minima are shown by dotted green lines. Red lines show the location of edit points in the ground truth.

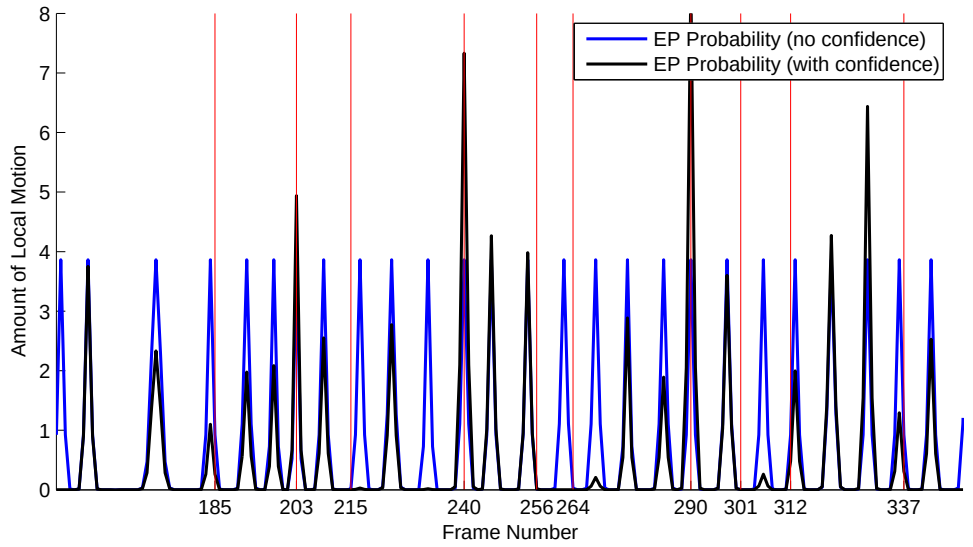


(b) Edit point probability generated by the $freq_{sharp}$ trace. Vertical red lines show ground truth edit points.

Figure 6.29: Edit point probability analysis using the $freq_{sharp}$ trace



(a) The *audio* trace over 200 frames of the *greenDancer* sequence. Peak maxima are shown by dashed green lines; peak minima are shown by dotted green lines. Red lines show the location of edit points in the ground truth.



(b) Edit point probability signal generated from the *audio* trace. Vertical red lines show the location of ground truth edit points.

Figure 6.30: Edit point probability analysis using the *audio* trace

6.7.2.1 Peaks in the combined probability signal

The locations of peak maxima in an edit point probability signal are taken as the identified edit points. The peak maxima are identified through parsing the signals into successive peaks in the same way as in the traces. Edit point signals $EP_t(n)$ arising from a single trace are parsed into peaks without any smoothing. This is because in these cases, the maxima in EP_t correspond directly to the best estimate for the location of an edit point, and so absolute fidelity in the identified peak maxima to maxima in the signal is required. For the combined $EP(n)$ signal, Savitzky-Golay filtering is applied prior to parsing into peaks. This ensures that adjacent peaks in different traces are combined to give a smooth final signal.

6.7.2.2 Weights for probability signals

A number of possibilities for the choice of the weights for each choice can be considered. The first possibility is not to use any weighting, effectively assigning each trace a weight of 1. These weights are designated $W_1(t)$.

Each trace could be weighted according to how reliably the peaks in the trace can be classified. For this scheme, the weights correspond to the area under the ROC obtained in classifier training. The intention is that the more reliably classified traces should have more influence on the derived edit point probability. These weights are designated by $W_R(t)$.

A further possibility is to use the prior probabilities for each trace as the weights. This corresponds to the intuition that the traces are weighted by how likely it is that a peak in the trace should correspond to an edit point. The weights are denoted $W_\rho(t)$.

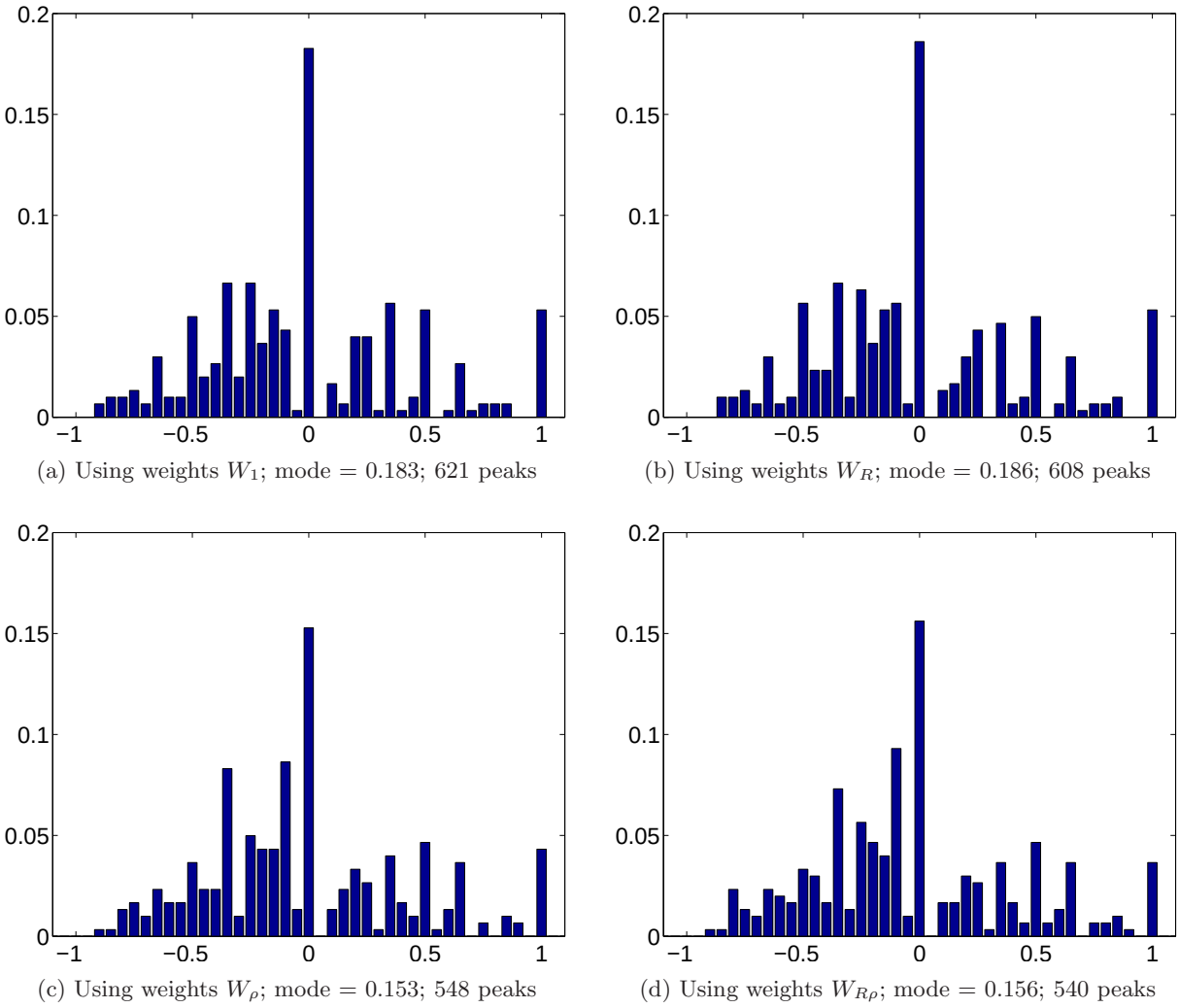
These latter two weighting schemes can be combined to form weights $W_{R\rho}(t) = W_R(t)W_\rho(t)$. These weights take into account both the prior probability that a peak corresponds to an edit point, and the reliability of peak classification in the various traces.

The values of W_R, W_ρ, and $W_{R\rho}$ for each trace are shown in table 6.10.

Irrespective of the weighting scheme used, the weights should also take into account the mutual information between related traces. For example, it has been established that there is considerable overlap in edit point detection between the bounding box traces bb_{min_x}, bb_{max_x}, and bb_{width}. To compensate for this effect, the weights for the bounding box traces are divided by 3. Similarly, the weights for the vector field traces are also divided by 3, with the exception of the weight for the vf_θ trace.

Figure 6.31 shows the distribution of peak-relative locations for edit points in the ground truth against peaks in the edit point probability traces generated using each of the three weighting schemes described. The ground truth data over the *greenDancer*, *maleDancer*, and *ballet2* sequences was combined to generate this figure. Using W_R, 18.27% of ground truth edit points are located exactly at peak maxima, compared to 15.9% and 15.6% for $W_{R\rho}$ and W_ρ respectively.

Alternatively, the trace weights can be chosen based on a subjective assignment of 'aesthetic effect' to each trace. Here, an operator analysing a particular video could assign weights to



(a) Using weights W_1; mode = 0.183; 621 peaks

(b) Using weights W_R; mode = 0.186; 608 peaks

(c) Using weights W_ρ; mode = 0.153; 548 peaks

(d) Using weights $W_{R\rho}$; mode = 0.156; 540 peaks

Figure 6.31: Location of 301 ground truth edit points from the *greenDancer*, *maleDancer*, and *ballet2* sequences, relative to peaks in the edit points probability signal. The number of peaks found in the probability signal is also given.

the individual traces manually corresponding to how important each effect is to the intended application. For example, for a flowing dance a change in vector direction would be assigned a high weight, whereas for more staccato dancing the *motion* and $freq_{sharp}$ traces could be given the largest weights. The effects of different trace weightings on edit point selection are illustrated on the accompanying DVD.

ω	*greenDancer*	*maleDancer*	*ballet2*
2^1	3.83	2.08	0.67
2^2	2.42	2.08	0.75
2^3	3.75	1.92	0.75
2^4	2.33	2.67	1.17
2^5	3.42	2.67	1.25
2^6	3.42	2.75	1.67
2^7	3.42	2.67	2.33
2^8	3.50	2.67	2.42
2^9	3.50	1.92	2.50
2^{10}	3.50	1.92	2.67

Table 6.11: Detection performance for the top twelve edit points for various limits on $C_t(p)$.

6.7.3 Assessment

6.7.3.1 Limiting kernel amplification

It was described above how the probabilistic edit point signal is found by adding together kernels, and how these kernels are amplified by the peak classification confidence C_tp. Where peak feature values are at an extreme of the range, very high $C_t(p)$ values are generated, and it is desirable to limit these values to some maximum ω. Table 6.11 shows how different values for this limit affect performance in edit point detection. The measure used is

$$\frac{1}{12}\sum_{i\in 1\ldots 12}\min_n |ep_i - \mathcal{G}(n)| \tag{6.53}$$

Here ep_i is the peak in $EP(n)$ having ith highest value. Thus the value of ω chosen is that which minimises the mean distance between the top twelve detected edit points and edit points in the ground truth. The value of 2^2 is chosen as achieving the best compromise over the three training sequences.

6.7.3.2 Numerical Assessment

Not all peaks in the $EP(n)$ signal will correspond to edit points. Thus, to extract a set of edit points from this signal, a threshold is chosen. Only peak maxima having values greater than this threshold are taken as candidate edit points. The set of estimated edit points is thus

$$ep = \{n : \exists p.\zeta_{max}(p) == n \wedge EP(n) > \hat{EP}\} \tag{6.54}$$

where $\hat{EP}$ is the threshold, and $\zeta_{max}(p)$ is the location of the maximal point of the pth peak in $EP(n)$.

$\hat{EP}$	#H	#FA	#M	P	R	M_{PR}	MD_{median}	MD_{mean}	MD_{var}
0.00	76	157	3	0.33	0.96	0.64	0.5	0.34	6.01
0.50	75	149	4	0.33	0.95	0.64	1	0.39	5.94
1.00	69	95	10	0.42	0.87	0.65	1	0.57	6.07
1.50	56	39	23	0.59	0.71	0.65	1	0.68	6.26
2.00	35	13	43	0.73	0.44	0.59	0	0.66	6.11
2.50	15	0	64	1.00	0.19	0.59	0	0.27	4.50
3.00	4	0	75	1.00	0.05	0.53	0	0.50	6.33
3.50	1	0	78	1.00	0.01	0.51	-2	-2.00	0.00
4.00	1	0	78	1.00	0.01	0.51	-2	-2.00	0.00
4.50	1	0	78	1.00	0.01	0.51	-2	-2.00	0.00
5.00	1	0	78	1.00	0.01	0.51	-2	-2.00	0.00

Table 6.12: Edit point detection using $EP(n)$ in the *greenDancer* sequence.

Tables 6.12, 6.13, and 6.14 show the performance of edit point detection for various values of $\hat{EP}$ in the three training sequences. The optimal value for $\hat{EP}$ is different for each of these sequences.

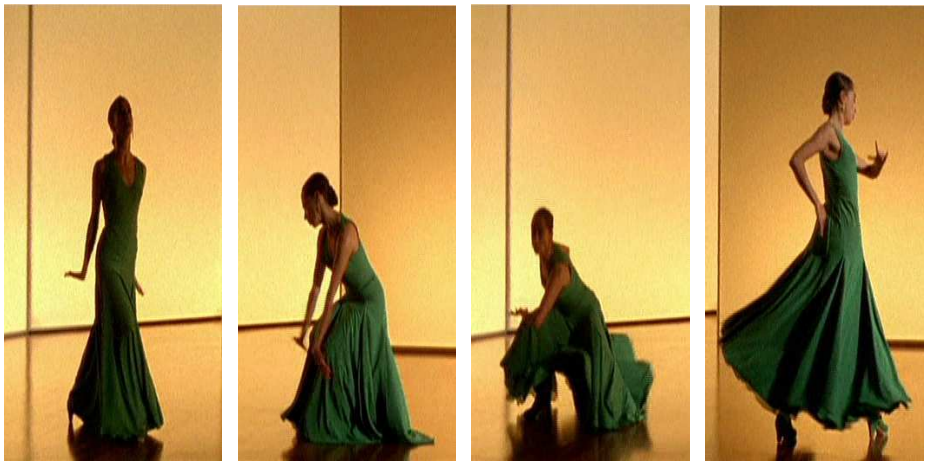
These results can be compared to 6.1, which described edit point detection performance using minima in the *motion* trace alone. Performance is considerably improved on the *maleDancer* sequence, where the maximum M_{PR} score rises from 0.66 to 0.77. On the other two training sequences, performance is about the same. This suggests that the *motion* trace is the most informative of the various traces considered. However, the combined method for edit point detection offers advantages in that the detected edit points can be ranked (by their $EP(n)$ value). Higher values of $EP(n)$ increase the precision of the set of detected edit points considerably. Furthermore, this ranking corresponds quite well to the relative significance, or visual strength, of the edit point. Lastly, with the combined method, the weights of the individual traces can be interactively adjusted to match the desired aesthetic criteria, or the characteristics of the sequence at hand. This is described in more detail in the next chapter.

6.7.3.3 Visual Assessment

Figures 6.32 and 6.33 show the twelve candidate edit points with the highest $EP(n)$ values in the *greenDancer* and *ballet2* sequences. The $EP(n)$ value and distance to the nearest ground truth edit point are shown underneath each frame.

These sequences were used in building the peak classification distributions, and so good performance is expected. This is achieved, in that all the selected edit points depict striking poses in the dance, and are detected very near ground truth edit points.

Figure 6.34 shows the twelve highest-scoring edit points detected in the *maleDancer2* se-



(a) 1537 (5.09, 2) (b) 578 (3.50, 0) (c) 689 (3.28, 4) (d) 439 (3.20, 0)

(e) 1406 (2.99, 3) (f) 773 (2.95, 1) (g) 528 (2.92, 1) (h) 201 (2.72, 2)

(i) 289 (2.69, 1) (j) 1443 (2.69, 1) (k) 849 (2.60, 0) (l) 823 (2.56, 1)

Figure 6.32: The top twelve edit points obtained from the *greenDancer* sequence. For each frame, the distance to the nearest ground truth edit point is shown in parentheses.

$\hat{EP}$	#H	#FA	#M	P	R	M_{PR}	MD_{median}	MD_{mean}	MD_{var}
0.00	98	145	1	0.40	0.95	0.68	1	0.87	3.42
0.50	98	142	1	0.41	0.95	0.68	1	0.87	3.42
1.00	98	126	1	0.44	0.95	0.69	1	0.87	3.42
1.50	97	102	2	0.49	0.94	0.71	1	0.88	3.44
2.00	96	78	3	0.55	0.93	0.74	1	0.98	3.18
2.50	90	53	8	0.63	0.87	0.75	1	0.91	3.32
3.00	83	29	15	0.74	0.81	0.77	1	0.95	3.41
3.50	75	18	22	0.81	0.73	0.77	1	0.97	3.08
4.00	67	12	30	0.85	0.65	0.75	1	1.00	3.21
4.50	55	6	42	0.90	0.53	0.72	1	1.07	3.36
5.00	42	3	59	0.93	0.41	0.67	1	1.07	3.34
5.50	32	2	70	0.94	0.31	0.63	1.5	1.25	3.29
6.00	21	0	81	1.00	0.20	0.60	1	0.86	3.93
6.50	13	0	89	1.00	0.13	0.56	0	0.92	3.91
7.00	6	0	96	1.00	0.06	0.53	1.5	1.00	5.20
7.50	3	0	100	1.00	0.03	0.51	-1	-1.00	1.00
8.00	3	0	100	1.00	0.03	0.51	-1	-1.00	1.00
8.50	2	0	101	1.00	0.02	0.51	-1.5	-1.50	0.50
9.00	1	0	102	1.00	0.01	0.50	-2	-2.00	0.00

Table 6.13: Edit point detection using $EP(n)$ in the *maleDancer* sequence.

quence. This material is from the same source as the *maleDancer* sequence, but was not used in building the peak classification distributions. Similarly, figure 6.35 shows the top twelve edit points detected in the *ballet1* sequence, which is from the same source as the *ballet2* sequence but was not used in building the classification distributions.

Performance is good on this sequence; all selected edit points depict striking poses, with the possible exception of the final image, frame 3604 in figure 6.34. It is notable that the *ballet1* sequence shows a group of dancers, while *ballet2* shows a single dancer. The good performance achieved on the *ballet1* sequence suggests that the techniques applied generalise to footage depicting multiple dancers.

The *greenMale* material depicts dance that begins with a single man, who is joined by a woman towards the end of the sequence. This footage was not used in building the distributions for peak classification. Figure 6.36 shows the top twelve edit points detected in this sequence. Performance on this sequence is again good. The highest-scoring edit point, at frame 2096, marks the end of the dance and is perhaps the most significant edit point in the sequence. Frame 2110 is a essentially a duplication of this edit point, selected due to the motion of the dress after the dancers stop.

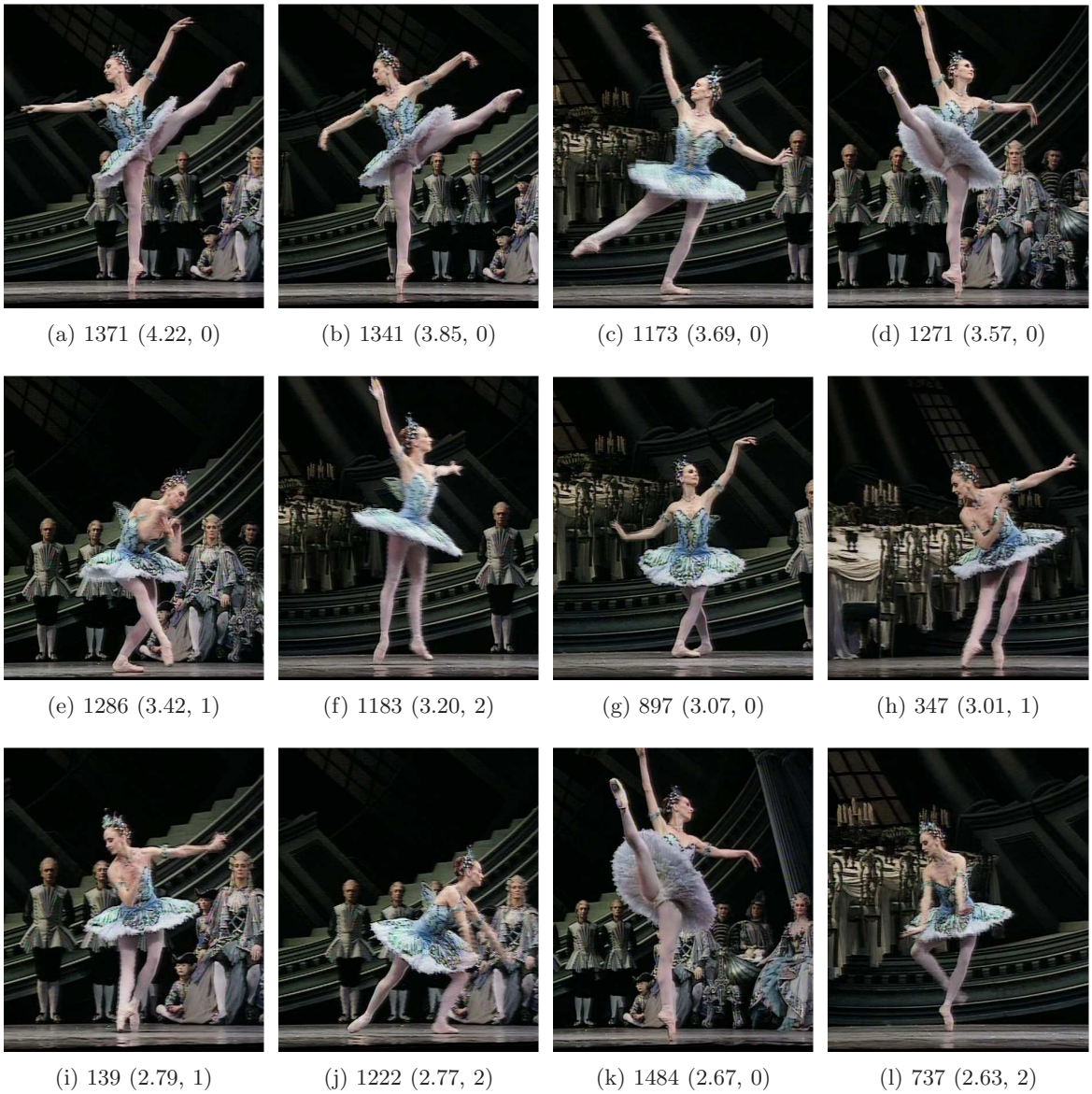
$\hat{EP}$	#H	#FA	#M	P	R	M_{PR}	MD_{median}	MD_{mean}	MD_{var}
0.00	113	79	3	0.59	0.95	0.77	0	-0.06	3.34
0.50	113	79	3	0.59	0.95	0.77	0	-0.06	3.34
1.00	113	79	3	0.59	0.95	0.77	0	-0.06	3.34
1.50	113	76	3	0.60	0.95	0.77	0	-0.06	3.34
2.00	111	67	4	0.62	0.93	0.78	0	-0.04	3.51
2.50	105	54	10	0.66	0.88	0.77	0	-0.05	3.53
3.00	100	45	14	0.69	0.84	0.76	0	-0.07	3.60
3.50	95	29	18	0.77	0.80	0.78	0	-0.17	3.63
4.00	81	17	32	0.83	0.68	0.75	0	0.01	3.39
4.50	68	10	45	0.87	0.57	0.72	0	0.09	3.75
5.00	54	8	61	0.87	0.45	0.66	0	0.04	2.75
5.50	41	6	75	0.87	0.34	0.61	0	0.00	2.65
6.00	33	2	83	0.94	0.28	0.61	0	-0.06	2.81
6.50	22	0	95	1.00	0.18	0.59	0	-0.14	2.41
7.00	12	0	107	1.00	0.10	0.55	0	-0.08	1.36
7.50	9	0	110	1.00	0.08	0.54	0	-0.56	0.53
8.00	8	0	111	1.00	0.07	0.53	0	-0.50	0.57
8.50	6	0	113	1.00	0.05	0.53	0	-0.50	0.70
9.00	5	0	114	1.00	0.04	0.52	0	-0.20	0.20
9.50	4	0	115	1.00	0.03	0.52	0	0.00	0.00
10.00	2	0	117	1.00	0.02	0.51	0	0.00	0.00
10.50	1	0	118	1.00	0.01	0.50	0	0.00	0.00
11.00	1	0	118	1.00	0.01	0.50	0	0.00	0.00

Table 6.14: Edit point detection using $EP(n)$ in the *ballet2* sequence.

6.8 Conclusion

This chapter has described a number of approaches for describing aspects of foreground motion in video, particularly dance video. Detecting discontinuities in the foreground motion can be achieved by locating extrema in these traces. Significant discontinuities in the foreground motion are shown to correspond to edit points in dance video. The edit point detection mechanism is shown to generate convincing results.

Numerically, the performance attained by the peak classification approach is not much greater than that attained by simply taking all minima in the *motion* trace. However, having a measure of edit point strength, such as that provided by probabilistic, combined-trace edit point detection, is useful for a number of applications. Several such applications are described in chapter 8.



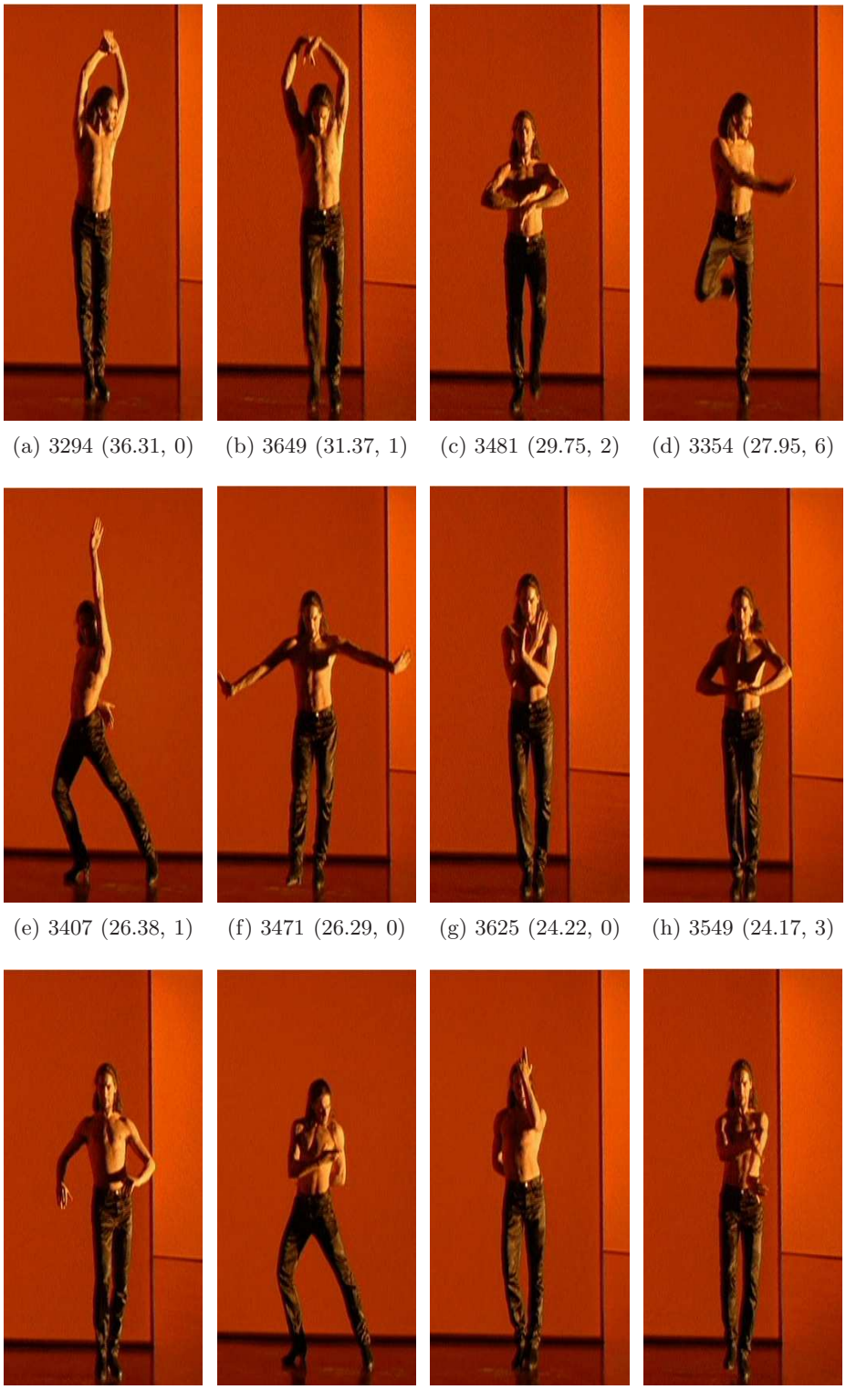
(a) 1371 (4.22, 0) (b) 1341 (3.85, 0) (c) 1173 (3.69, 0) (d) 1271 (3.57, 0)

(e) 1286 (3.42, 1) (f) 1183 (3.20, 2) (g) 897 (3.07, 0) (h) 347 (3.01, 1)

(i) 139 (2.79, 1) (j) 1222 (2.77, 2) (k) 1484 (2.67, 0) (l) 737 (2.63, 2)

Figure 6.33: The top twelve edit points obtained from the *ballet2* sequence. For each frame, the distance to the nearest ground truth edit point is shown in parentheses.

Various refinements to the methods described here are possible. Improvements to the motion detection, global motion estimation, or local motion estimation methods used would assist in edit point detection. The key area for future work is the peak classification system, and in particular the method used to parse traces into peaks. In the system as described, minima in the traces are determined using a heuristic method with fixed parameters (such as the Savitzky-Golay filter order, and the size of the median filter). A probabilistic approach to minima detection could result in improved performance for peak extraction and hence edit point detection.

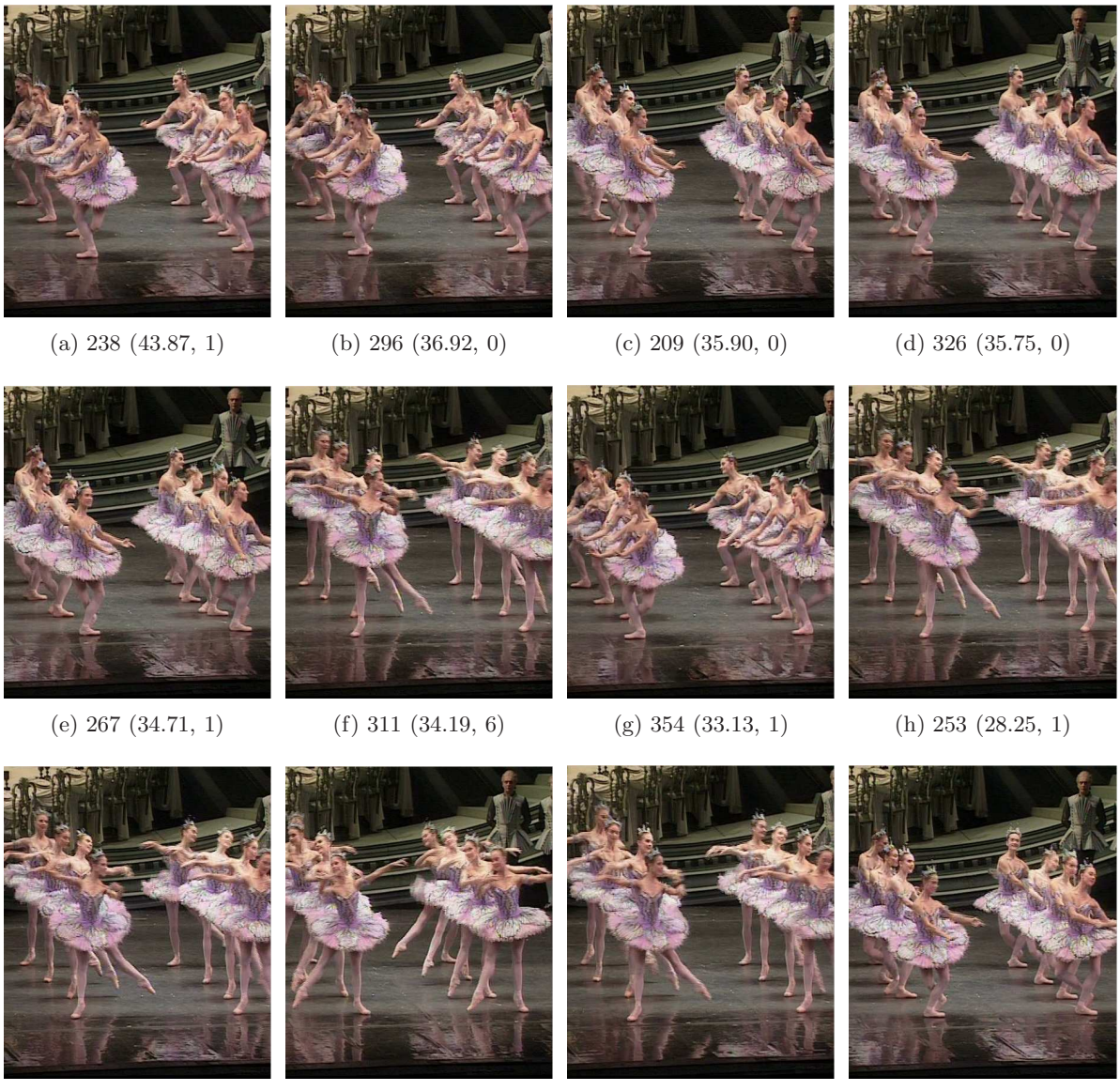


(a) 3294 (36.31, 0) (b) 3649 (31.37, 1) (c) 3481 (29.75, 2) (d) 3354 (27.95, 6)

(e) 3407 (26.38, 1) (f) 3471 (26.29, 0) (g) 3625 (24.22, 0) (h) 3549 (24.17, 3)

(i) 3571 (24.15, 1) (j) 3380 (23.51, 5) (k) 3438 (22.83, 2) (l) 3604 (22.73, 10)

Figure 6.34: The top twelve edit points obtained from the *maleDancer2* sequence.

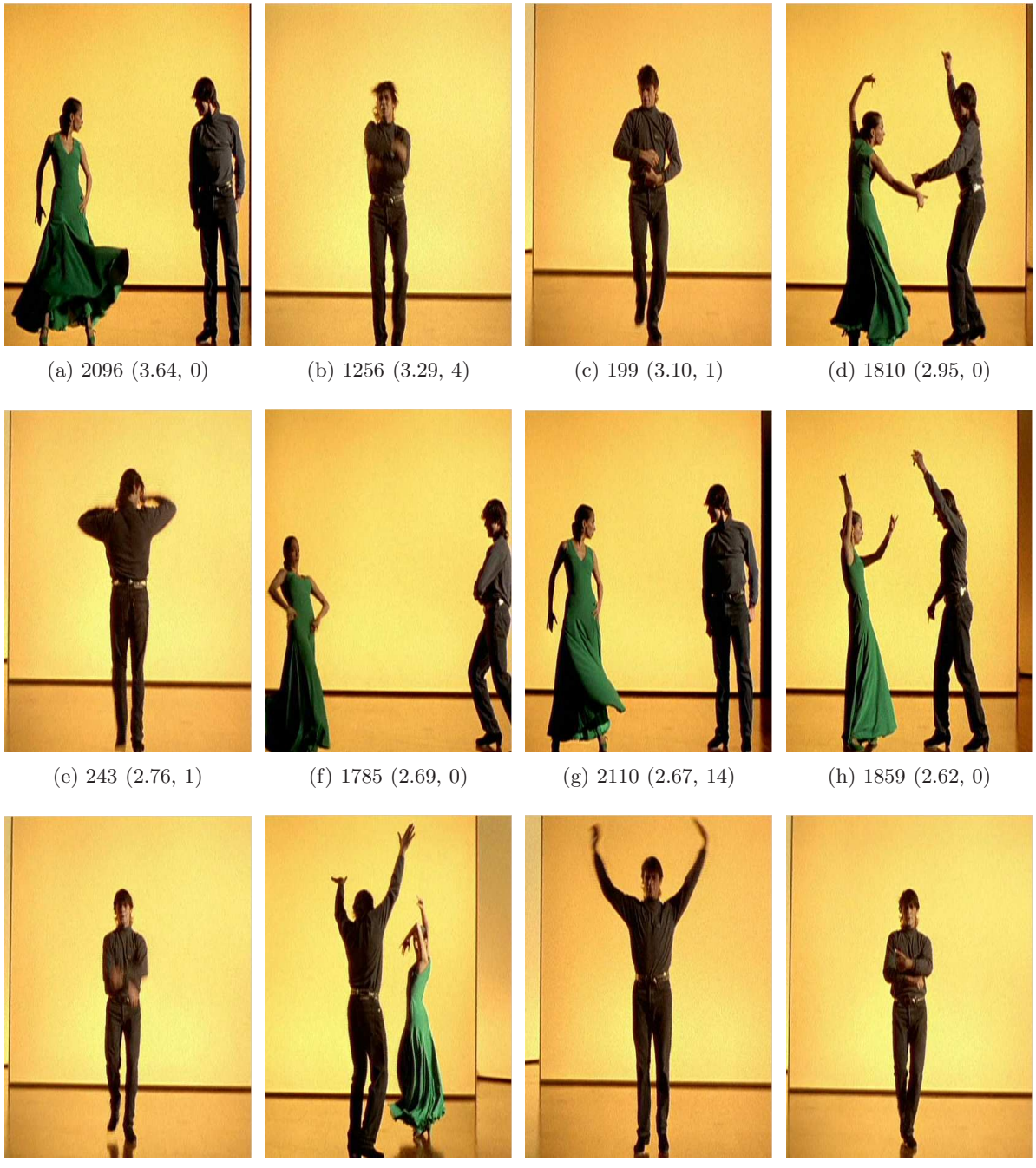


(a) 238 (43.87, 1) (b) 296 (36.92, 0) (c) 209 (35.90, 0) (d) 326 (35.75, 0)

(e) 267 (34.71, 1) (f) 311 (34.19, 6) (g) 354 (33.13, 1) (h) 253 (28.25, 1)

(i) 368 (28.18, 0) (j) 341 (27.23, 2) (k) 194 (26.74, 4) (l) 384 (26.03, 0)

Figure 6.35: The top twelve edit points obtained from the *ballet1* sequence.



(a) 2096 (3.64, 0) (b) 1256 (3.29, 4) (c) 199 (3.10, 1) (d) 1810 (2.95, 0)

(e) 243 (2.76, 1) (f) 1785 (2.69, 0) (g) 2110 (2.67, 14) (h) 1859 (2.62, 0)

(i) 645 (2.54, 1) (j) 1648 (2.53, 3) (k) 188 (2.53, 2) (l) 700 (2.53, 0)

Figure 6.36: The top twelve edit points obtained from the *greenMale* sequence.

7

Segmentation For Edit Point Detection [1]

The previous chapter considered methods for edit point identification based on information characterising each entire frame of an image sequence. In many cases, the performance of these methods is impaired because the motion of some regions of each frame is more informative regarding edit points than others. The *greenDancer* sequence includes several typical examples, such as where the motion of the arms is expressive of an edit point while the motion of the clothing is not.

To improve on performance in these cases, an attempt to segment the video into its constituent elements is undertaken. Characterising the motion of each segment individually should then enable the edit point detection process to distinguish between the dancer's clothing and her limbs, for example, and then bias edit point detection to the more relevant regions.

The video segmentation problem was introduced in chapter 2. In outline, the segmentation process involves assigning a label to each pixel in each frame of the video. The different label values correspond to different segments in the frame, and the design aim is that these segments should be meaningful. For the application described here, a Markov Random Field approach using Iterated Conditional Modes (ICM) is employed to solve the resulting Maximum *a posteriori* (MAP) problem.

This chapter continues with an overview of Markovian Random Field Segmentation as im-

[1] An account of the early stages of this work was described in "Exploiting temporal discontinuities for event detection and manipulation in video streams" by Hugh Denman, Erika Doyle, Anil Kokaram, Daire Lennon, Rozenn Dahyot and Ray Fuller, in *MIR '05: Proceedings of the 7th ACM SIGMM International Workshop on Multimedia Information Retrieval*, pages 183–192, New York, October 2005

plemented here. Subsequent sections detail how colour and motion data are used to inform the segmentation; how likely segmentation boundaries are determined via the 'edge process'; a means for assessing the confidence associated with a given motion vector; and how smoothness is encouraged in the segmentation. The following sections describe how changes in the character of the video are dealt with, be they changes in the motion of a segment or changes in the number of segments required to model the sequence. The chapter ends with a discussion of how the segmentation output is used to assist in edit point identification. Videos illustrating details and results from this chapter are included on the DVD accompanying this thesis.

7.1 Markovian Random Field Segmentation

Segmentation is carried out using the image data, I, and the local motion vectors D at each frame. The image space is divided into sites $\mathbf{x}$, and each site is assigned a label $L(\mathbf{x}) \in \mathcal{L}$. Each label has an associated model, describing the observed 'behaviour' at sites assigned this label in terms of colour distribution, motion, and location within frame space.

In probabilistic terms, the problem of segmentation is then to find the most probable label field $L(\mathbf{x})$. However, a probability distribution over an entire labelfield has very high dimensionality, typically in the many hundreds or thousands, and it is very difficult to analyse such high-dimensional probability spaces. Therefore, the segmentation is calculated iteratively on a site-by-site basis. This involves analysis of the probability distribution at each site $L(\mathbf{x})$ individually, given the label assignments at other sites $L(-\mathbf{x})$ and the image and motion data I and D:

$$P(L(\mathbf{x})|L(-\mathbf{x}), I, D, \theta) \tag{7.1}$$

θ represents the model parameters for each label. Each site is assigned the label maximising 7.1, i.e. the mode of the distribution. This technique for optimization in high-dimensionality spaces is called ICM [325].

It can generally be assumed that the probability distribution over labels at a given site is independent of label assignments at sites far away in the image space. This assumption is the essence of Markov Random Field modelling, an approach that simplifies matters by assuming that the probability distribution over labels at site $\mathbf{x}$ depends only on the labels assigned at neighbouring sites, $\{L(\mathbf{x}')|\mathbf{x}' \in \mathcal{N}(\mathbf{x})\}$:

$$P(L(\mathbf{x})|L(-\mathbf{x}), I, D, \theta) = P(L(\mathbf{x})|L(\mathcal{N}(\mathbf{x})), I, D, \theta) \tag{7.2}$$

In this work the neighbourhood $\mathcal{N}(\mathbf{x})$ includes the eight neighbours of $\mathbf{x}$ found along the horizontal, vertical, and diagonal directions:

$$\mathcal{N}(\mathbf{x}) = \left\{\mathbf{x}' : \mathbf{x}' \in \left\{\mathbf{x} + \{-1, 0, 1\}^2\right\} \setminus \mathbf{x}\right\} \tag{7.3}$$

Analysis is further simplified by factoring the probability as follows:

$$\underbrace{P(L(\mathbf{x})|L(-\mathbf{x}),I,D,\theta)}_{\text{Posterior}} \propto \underbrace{P(I,D|L(\mathbf{x}),\theta)}_{\text{Likelihood}} \underbrace{P(L(\mathbf{x})|L(-\mathbf{x}),\theta)}_{\text{Smoothness Prior}} \tag{7.4}$$

The likelihood component measures how likely the data at the site is, given a particular label assignment—i.e. how well the data accords with the model corresponding to that label. The prior incorporates the intuition that neighbouring sites should have the same label. The data features used here are the colour characteristics and local motion vector at the site. These features are assumed to be independent, and so the likelihood is factorised into two components:

$$P(I,D|L(\mathbf{x}),\theta) = \underbrace{P(I(\mathbf{x})|L(\mathbf{x}),\theta)}_{\text{Colour likelihood}} \underbrace{P(\mathbf{D}(\mathbf{x})|L(\mathbf{x}),\theta)}_{\text{Vector likelihood}} \tag{7.5}$$

Segmenting of the video at pixel granularity would require that each pixel is a site in itself. However, computing label assignments at pixel granularity is prohibitively computationally expensive. Furthermore, segments in video are usually much larger than one pixel, and so pixel granularity is not necessarily required. To improve computational performance, then, pixels are manipulated in blocks, where each block is a site. Here a blocksize of 17×17 pixels is used, for analysis of PAL video at a resolution of 720×576 pixels.

7.2 Colour Model

The colour likelihood $P(I(\mathbf{x})|L(\mathbf{x}),\theta)$ is a measure of how likely it is that the colour values at site $\mathbf{x}$ were generated by $L(\mathbf{x})$. The model for the label contains a representation of the colour distribution for that label, and the site also contains a distribution of colours. The likelihood of each label at the site is then inversely related to the distance between the label distribution and the site distribution.

7.2.1 Colourspace selection

In colour video, the colour of each pixel is represented with a three-tuple. This three-tuple represents a point in some colourspace. A wide variety of colourspaces have been developed, each suited to different applications. Digital video data is usually stored in some variant of the YUV colourspace, which lends itself to compression and transmission. However, this colourspace is not perceptually uniform: the perceived colour difference of a given numerical difference in a component value is not uniform across the range of the component. The CIE L, u*, v* colourspace was developed to have approximate perceptual uniformity throughout its gamut, and is therefore chosen to represent colour distributions here.

Each colour component is treated as independent, and so the colour likelihood is factorised into three components

$$P(I|L(\mathbf{x}),\theta) = P(I^L(\mathbf{x})|L(\mathbf{x}),\theta)P(I^{u*}(\mathbf{x})|L(\mathbf{x}),\theta)P(I^{v*}(\mathbf{x})|L(\mathbf{x}),\theta) \tag{7.6}$$

where I^k is the image data in channel $k \in \{L, u*, v*\}$.

7.2.2 Distribution distance

Various methods of assessing the difference between the distribution of colour in the block being assessed and the distribution of colour in the model can be adopted. An important criterion for the distance measure used is that partial matches should be recognised. Where a label corresponds to a moving object much larger than the site size, each site may contain only some of the colours associated with the label. These sites should have high likelihood of the corresponding label, on the strength of a partial distribution match.

To allow for the possible complexity of coloured labels, a non-parametric histogram representation is used for the colour distributions. As described in chapter3, a number of distance measures can be used to compare the histograms. The simplest measure is the bin-to-bin distance. While popular due to its conceptual simplicity, it offers a poor representation of relative difference between histograms, and is not appropriate for partial histogram matching.

The Earth Mover's Distance (EMD) [264, 265] is a histogram distance developed specifically for image retrieval, and has numerous advantages over the bin-to-bin distance for image segmentation. In this measure the histograms are considered as distributions of mass; one is designated the 'supply' distribution, and the other the 'demand' distribution. The distance measures the minimal amount of work that has to be done to transform the supply histogram into a configuration that satisfies the demand histogram. Partial distribution matching can be incorporated by not normalising the histograms; the histogram containing greater mass is then chosen as the supply histogram. Figure 7.1 presents some examples of inter-histogram differences measured by the bin-to-bin and EMD approaches, illustrating the advantages of the EMD in various cases.

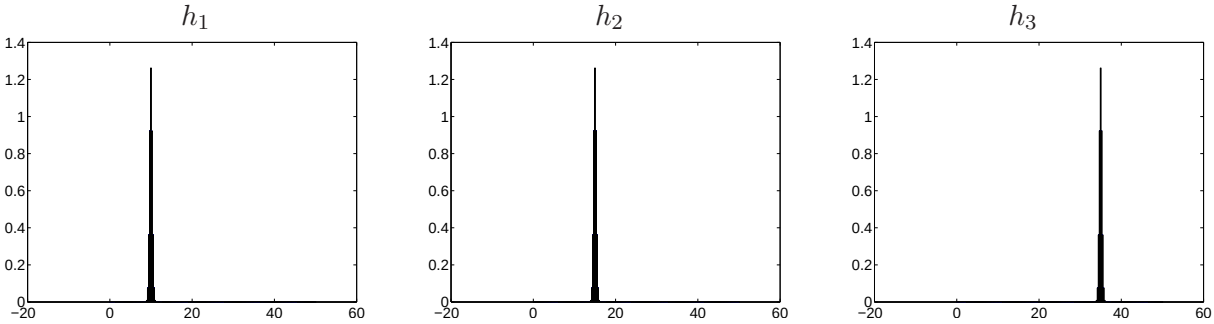
The colour model for each label consists of the three histograms (one for each channel) over all the sites assigned to the label. This work deals with images digitised to 8-bits per channel, and so a 256-bin histogram specifies the channel distribution completely. However, the EMD is a computationally expensive measure, and so histograms of 50 bins are used here. In some sequences, the channel range in a particular frame is much less than the full gamut of 0-255. Therefore the 50 bin centres are evenly spaced over the observed channel range in each frame before the colour models are found.

7.2.3 Colour likelihood

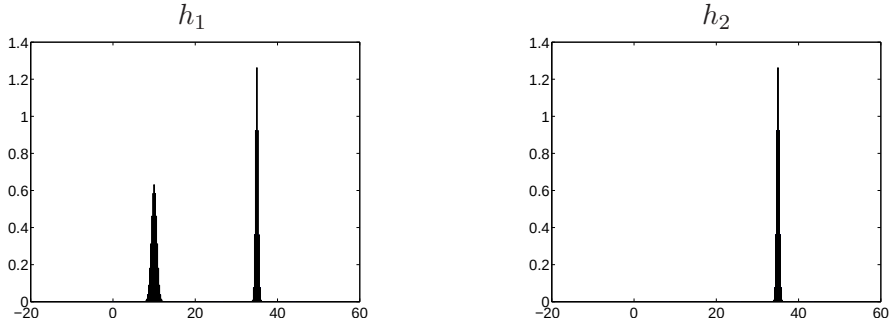
The likelihood of the colour distribution at site x, given the label assignment $L(\mathbf{x})$, is given by

$$P(I^k(\mathbf{x})|L(\mathbf{x})) \propto \exp - \left[\frac{EMD[H^k(\mathbf{x}), H^k_{L(\mathbf{x})}]}{2\hat{\sigma}_k^2} \right] \tag{7.7}$$

Here $H^k(\mathbf{x})$ is the histogram of values for channel k at site $\mathbf{x}$, and $H^k_{L(\mathbf{x})}$ is the histogram of values for channel k in the model with label $L(\mathbf{x})$. $\hat{\sigma}_k^2$ is a constant chosen to represent the expected within-class variance of the colour distance in channel k.



(a) Three histograms are shown. h_2 and h_3 have no overlap with h_1, but h_3 is further from h_1 than h_2. The bin-to-bin difference B2B(h_1, h_2) is the same as B2B(h_1, h_3), taking the value 4 in both cases. The EMD takes the 'distance of the difference' into account: EMD(h_1, h_2)=5, EMD(h_1, h_3)=25.



(b) This figure illustrates partial histogram matching. h_2 is a subset of h_1. The bin to bin distance B2B(h_1, h_2) is 2. If the histograms are normalised, B2B($h_1 - h_2$) = 100. Using the EMD, EMD(h_1, h_2) = 0. When the histograms are normalised, EMD(h_1, h_2) = 12.5. Thus the EMD is suitable for partial matching of un-normalised histograms.

Figure 7.1

7.3 The Edge Process

Adjacent sites containing dissimilar data are less likely to be members of the same segment than neighbouring sites with similar data. It is therefore expedient to measure the similarity of adjacent sites in advance. This gives rise to the *edge process* [298,325], a line field having values for each clique of blocks, describing the distance between their data. Here the edge process is estimated using the site colour content, and the EMD measure described above is used to assess the similarity of neighbouring blocks.

In many implementations a 'hard' (binary valued) edge process is used, and is iteratively optimised as part of the segmentation process. This add significantly to the complexity and computational cost of the process, however. Here a soft, scalar-valued edge process is used, such that the influence of a block on its neighbour is proportional to the similarity of their colour content. This edge process is precomputed for each frame of the video before segmentation is undertaken.

To estimate the edge process, the EMD is calculated between all pairs of neighbouring blocks

in the label field, for each of the L, u*, and v* histograms. This results in twelve difference maps:

$$D_v^k(r,c) = \text{EMD}[H^k(r,c)), H^k(r+1,c))] \tag{7.8}$$

$$D_h^k(r,c) = \text{EMD}[H^k(r,c), H^k(r,c+1))] \tag{7.9}$$

$$D_{d_1}^k(r,c) = \text{EMD}[H^k(r,c), H^k(r+1,c+1))] \tag{7.10}$$

$$D_{d_2}^k(r,c) = \text{EMD}[H^k(r,c), H^k(r-1,c+1))] \tag{7.11}$$

where $k \in \{L, u*, v*\}$ represents the three colour channels, and v, h, d_1, d_2 are the vertical, horizontal, descending diagonal, and ascending diagonal directions. $H^k(\mathbf{x})$ is the histogram of the values in channel k at site $\mathbf{x}$.

Peaks in the difference maps are assumed to correspond to edges. These are identified by finding the zero crossings in the derivative. Let $D(n)$ be the difference values in one channel along a particular orientation, for example

$$D(n) = D_v^{u*}(n, C) \tag{7.12}$$

for some fixed column C. This is essentially a first derivative signal along a column of the image. Such a signal is shown in the uppermost plot of figure 7.2. The signal $D'(n) = D(n) - D(n-1)$ is then found, shown in the middle plot of figure 7.2. This is a second derivative of the image, and thus zero crossings in this signal correspond to edges in the image [279]. The edge strength signal, $Z(n)$, is then given by

$$Z_v(n) = \begin{cases} \min(|D'(n)|, |D'(n+1)|) & \text{if } D'(n) > 0 \wedge D'(n+1) < 0 \\ 0 & \text{otherwise} \end{cases} \tag{7.13}$$

where strength of the edge is taken to be the minimum of the slopes on either side of the peak. The lowermost plot in figure 7.2 shows the edge strength associated with the detected peaks.

The final difference map values for each direction are generated by addition across the channels, for example

$$Z_v(r,c) = Z_v^L(r,c) + Z_v^{u*}(r,c) + Z_v^{v*}(r,c) \tag{7.14}$$

The difference map for for any pair of adjacent sites, $Z[(r,c),(r',c')]$ is then defined by the four oriented difference maps. To create a soft edge process, $E[(r,c),(r',c')]$, the difference values are mapped to the range $0, 1$ according to

$$E[(r,c),(r',c')] = G_{(6,10)}\left(Z[(r,c),(r',c')]\right) \tag{7.15}$$

$$G_{(\mu,\sigma^2)}(x) = 0.5\left(1 + \text{erf}\left(\frac{x-\mu}{\sqrt{2\sigma^2}}\right)\right) \tag{7.16}$$

$G_{(\mu,\sigma^2)}$ is the cumulative distribution function of a Gaussian. Here $\mu = 4$ and $\sigma^2 = 2$, and so site difference peaks of strength 4 are mapped to an edge value of 0.5. This mapping has been found to generate reasonable soft edge maps for several sequences. The mapping curve is shown in figure 7.3. Figure 7.4 illustrates the edges exceeding various threshold values in the *coconut* sequence; further examples of edge maps found by this process are presented on the accompanying DVD.

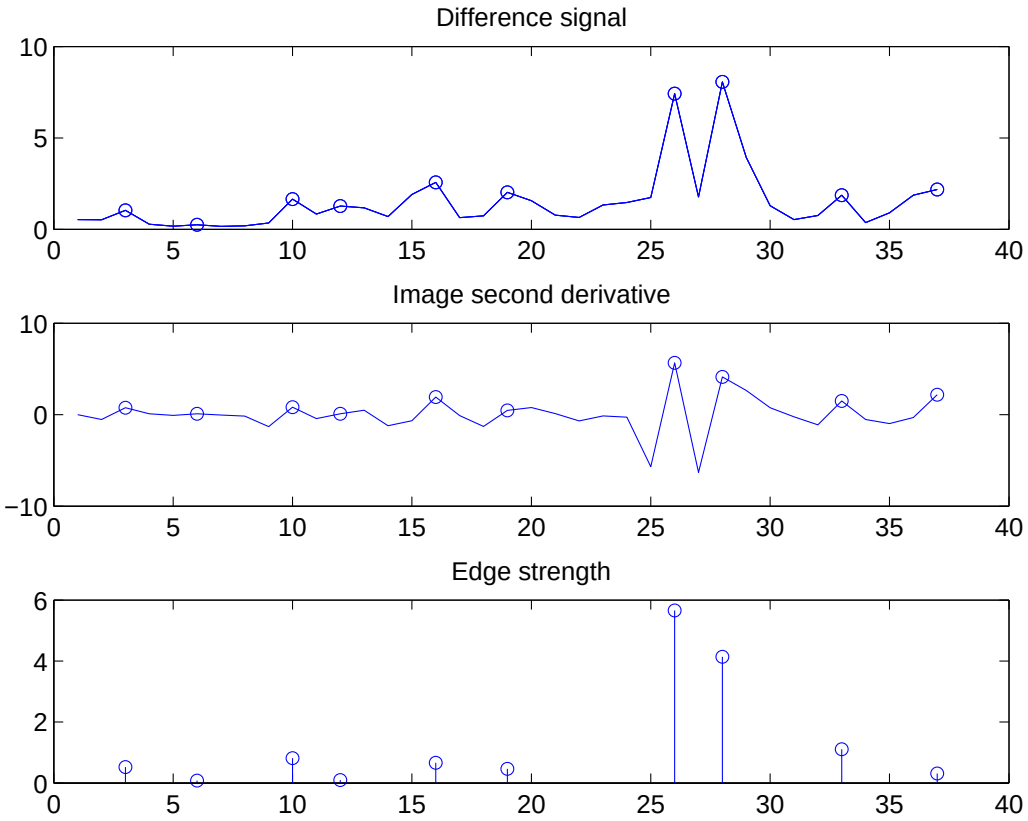


Figure 7.2: Locating peaks in the difference trace. The locations of detected peaks are marked by circles.

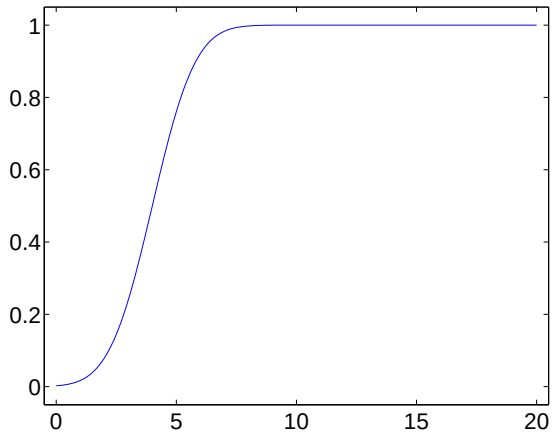


Figure 7.3: Mapping from difference values to edge strength. The curve is the cdf of a Gaussian with mean 4 and variance 10.

Figure 7.4: Edges in a frame from the *coconut* sequence, with edge thresholds of 0.1, 0.5, and 0.9.

7.4 Motion Models

The motion in the image sequence is represented by the local motion vectors. The forward and backward vectors are defined by the image sequence model

$$I_n(\mathbf{x}) = I_{n-1}(\mathbf{x} + \mathbf{D}_n^b(\mathbf{x})) \tag{7.17}$$

$$I_n(\mathbf{x}) = I_{n+1}(\mathbf{x} + \mathbf{D}_n^f(\mathbf{x})) \tag{7.18}$$

The Wiener-based local motion estimation technique described by Kokaram in [178] is used to generate the vectors here. The vector model for each label is represented using a two-dimensional Gaussian distribution; a confidence measure on each vector $C(\mathbf{x})$, described in the next section, is used to weight the contribution of the vector to the Gaussian.

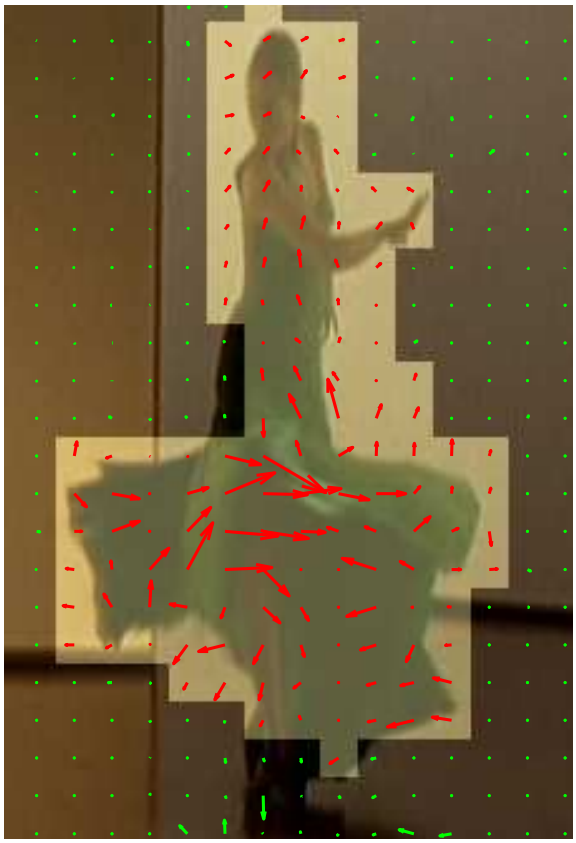
Some issues pertaining to the choice of vector parameterisation are described first, followed by a more exact description of how the vector models for each label are found, and how the likelihood of a particular vector given a vector model is calculated.

7.4.1 Cartesian / Polar vector representation

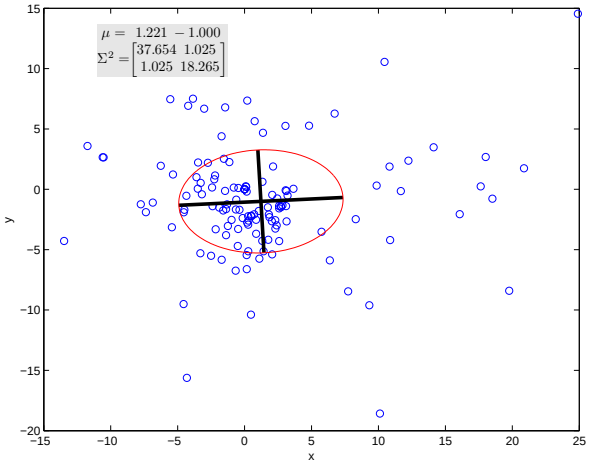
There are several ways in which motion can be parameterised for the purposes of motion-based segmentation. The most commonly used parameterisation is the Cartesian representation, $\mathbf{D}^k(\mathbf{x}) = [dx, dy]$. $k \in \{b, f\}$ denotes the direction, either backwards or forwards. This representation results in a 'strict' segmentation of the vector field, in that dissimilar vectors will have high distances. However, alternative parameterisations are more suitable in sequences where motion estimation is difficult. Consider the vectors shown in figure 7.5 (a). The highlighted region corresponds to the sites in one segment after a hypothetical vector-based segmentation. The motion of the dancer's dress is *pathological*, and results in incoherent motion vectors. The distribution for this label, then, should represent only that motion is occurring, without specifying the nature of that motion.

Figure 7.5 (b) shows the Gaussian distribution fit to the Cartesian representation of these vectors. The fitted distribution has a high variance and is essentially uninformative. In particular, due to the near-random spread of the vectors in the segment, the model mean is very close to zero, and so zero vectors in stationary parts of the scene will have a high likelihood given this model.

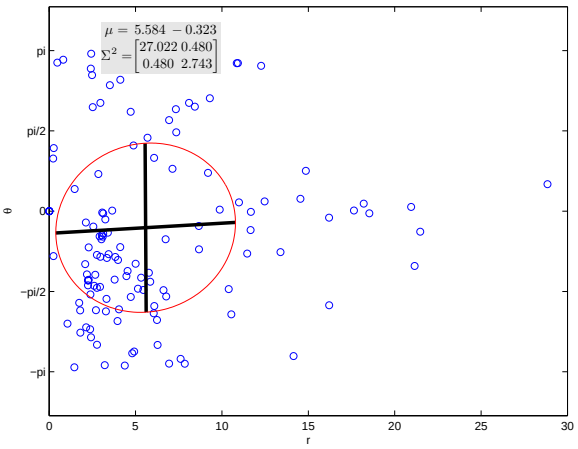
Figure 7.5 (c) shows the Gaussian distribution fit to the polar representation of these vectors. Here, the mean magnitude is 5 pixels, indicating motion, but the distribution remains quite broad, and the zero vector is about one standard deviation from mean. Figure 7.5 (d) shows the Gaussian fit to the vectors in a $[\log(r), \theta]$ parameterisation. Taking the log of the magnitude has the effect of dilating the vector space at low magnitudes, and compressing the range for larger vectors; the distance between two vectors along the $\log(r)$ axis is proportional to the relative difference in their magnitudes. In this parameterisation, the vector magnitudes must be thresholded, to avoid taking the log of 0; here a minimum magnitude threshold of 0.01 is



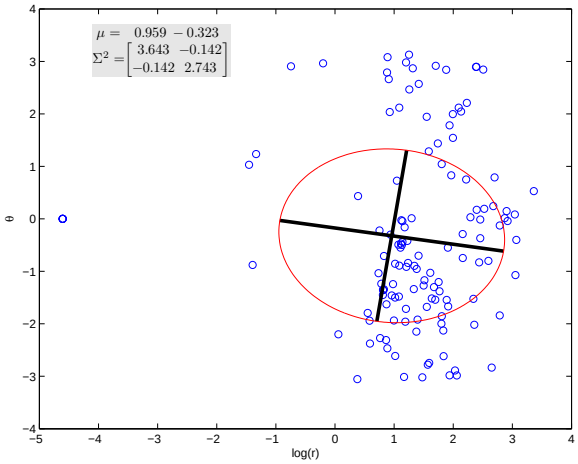
(a) Pathological motion of clothing in the *greenDancer* sequence



(b) Vector distribution with (x, y) parameterisation



(c) Vector distribution with (r, θ) parameterisation



(d) Vector distribution with $(\log(r), \theta)$ parameterisation

Figure 7.5: Vector parameterisation and pathological motion. The red circles indicate the mean and standard deviation of the Gaussian fit to each distribution.

used. With this threshold, the distance between the model mean and the zero vector is over 4 standard deviations. The $[\log(r), \theta]$ representation of the vector in direction k at site $\mathbf{x}$ is designated $\mathbf{M}^k(\mathbf{x})$.

7.4.2 Wrapping the vector angle distribution

A difficulty with vector models parameterised on vector angle is that the parameter space is non-Euclidean with respect to distance, being toroidally connected at π and $-\pi$. In practice, this results in undesirably high variance in the angle component for leftward motion (where the vector angle is close to π). Figure 7.6 (a) shows a region undergoing leftward motion in the *coconut* sequence, and the high variance in the angle parameter is illustrated in figure 7.6 (b). In figure 7.6 (c), the vector distribution is shown where the angle space has been shifted to range over $-2\pi, 0$. The variance in the angle dimension of this distribution is an order of magnitude smaller than that over the $-\pi, \pi$ range, which indicates that this range is to be preferred in this case.

Shifting the range of the vector angles in this way is denoted *wrapping the angle space*. Denote the angles of the vectors by $\mathbf{A}$, and the corresponding vector confidences by $\mathbf{C}$. The vector angles will be in the range $-\pi, \pi$. The angles after wrapping at θ, denoted $\mathbf{A}^\theta$, are given by

$$\mathbf{A}_i^\theta = \begin{cases} A_i, & A_i \leq \theta \\ A_i - 2\pi, & A_i \leq \theta \end{cases} \tag{7.19}$$

The wrap point minimising the variance of angles A, $\hat{\theta}(A)$, is then found according to

$$\hat{\theta}(A) = \underset{\theta \in (-\pi, \pi)}{\arg\min} \left[\frac{\sum_i C_i^2 (A_i^\theta - \bar{A^\theta})(A_i^\theta - \bar{A^\theta})'}{\sum_i C_i^2} \right] \tag{7.20}$$

$\bar{A^\theta}$ is the weighted mean of the angles $\mathbf{A}$, with weights given by the vector confidences $\mathbf{C}$. $\hat{\theta}$ is found using the algorithm described in [258], based on golden section search and parabolic interpolation. The vectors after wrapping at the optimal point $\hat{\theta}$ are denoted $\mathbf{M}_{\hat{\theta}}^k(\mathbf{x})$.

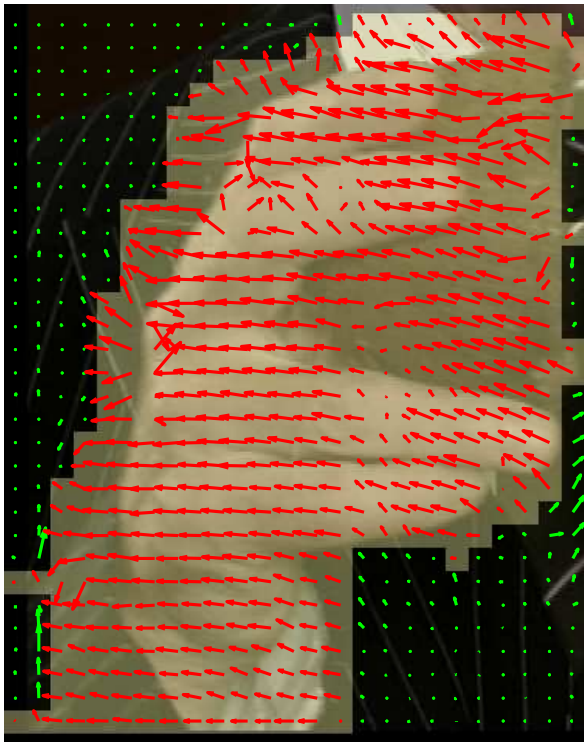
7.4.3 Vector models

The vector model for label l in direction k, using the Cartesian (x, y) parameterisation, is found by

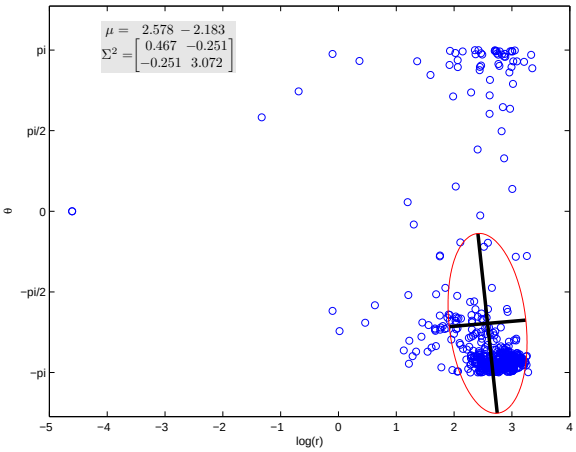
$$\boldsymbol{\mu}_l^k = \frac{\sum_{\mathbf{x} \in \{\mathbf{x} | L(\mathbf{x}) = l\}} C^k(\mathbf{x}) \mathbf{D}^k(\mathbf{x})}{\sum_{\mathbf{x} \in \{\mathbf{x} | L(\mathbf{x}) = l\}} C^k(\mathbf{x})} \tag{7.21}$$

$$\mathbf{R}_l^k = \frac{\sum_{\mathbf{x} \in \{\mathbf{x} | L(\mathbf{x}) = l\}} (C^k(\mathbf{x}))^2 (\mathbf{D}^k(\mathbf{x}) - \boldsymbol{\mu}_l^k)(\mathbf{D}^k(\mathbf{x}) - \boldsymbol{\mu}_l^k)'}{\sum_{\mathbf{x} \in \{\mathbf{x} | L(\mathbf{x}) = l\}} (C^k(\mathbf{x}))^2} \tag{7.22}$$

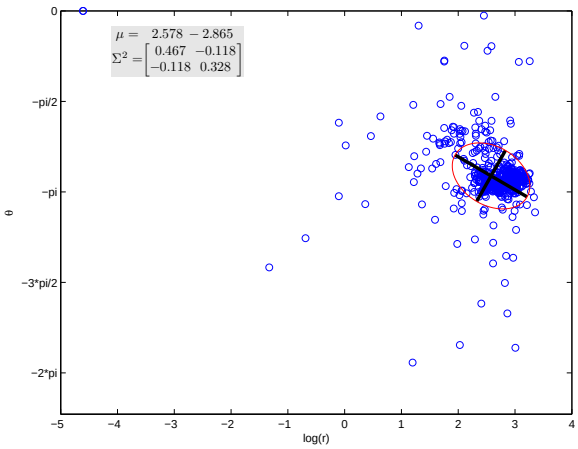
As mentioned above, the vector at each site is weighted by a confidence measure, $C^k(\mathbf{x})$, to prevent the vector models being influenced by erroneous motion estimation; the calculation of this vector confidence is addressed in the next section.



(a) Leftward motion in the *coconut* sequence



(b) Vector distribution with $(\log(r), \theta)$ parameterisation



(c) Vector distribution with $(\log(r), \theta)$ parameterisation, wrapped angle domain.

Figure 7.6: Vector parameterisation and leftward motion. The red circles indicate the mean and standard deviation of the Gaussian fit to each distribution.

For the $[log(r), \theta]$ vector representation, the vector model is found as in equations 7.21 and 7.22, with $\mathbf{D}^k(\mathbf{x})$ replaced by $\mathbf{M}^k_{\theta_w}(\mathbf{x})$.

7.4.4 Vector likelihood

For the (x, y) vector parameterisation, the vector likelihood is found by

$$P(\mathbf{D}^k(\mathbf{x})|L(\mathbf{x})) \propto \exp - \left[0.5 \left(\mathbf{D}^k(\mathbf{x}) - \boldsymbol{\mu}^k_{L(\mathbf{x})}\right)' \left(\mathbf{R}^k_{L(\mathbf{x})}\right)^{-2} \left(\mathbf{D}^k(\mathbf{x}) - \boldsymbol{\mu}^k_{L(\mathbf{x})}\right)\right] \tag{7.23}$$

where $\boldsymbol{\mu}^k_{L(\mathbf{x})}$ and $\mathbf{R}^k_{L(\mathbf{x})}$ are the mean and covariance of the Gaussian vector distribution for the model with label $L(\mathbf{x})$.

The likelihood is similar for the representations with an angle parameter:

$$P(\mathbf{M}^k(\mathbf{x})|L(\mathbf{x})) \propto \exp - \left[0.5\gamma \left(\mathbf{M}^k(\mathbf{x}) - \boldsymbol{\mu}^k_{L(\mathbf{x})}\right)' \left(\mathbf{R}^k_{L(\mathbf{x})}\right)^{-2} \gamma \left(\mathbf{M}^k(\mathbf{x}) - \boldsymbol{\mu}^k_{L(\mathbf{x})}\right)\right] \tag{7.24}$$

Here γ is a function wrapping the θ component of the difference to the range $0, \pi$

$$\gamma(\log(r), \theta) = \begin{cases} (\log(r), \theta), & \theta \in (0, \pi) \\ (\log(r), 2\pi - \theta), & \theta \in (\pi, 2\pi) \end{cases} \tag{7.25}$$

7.5 Vector confidence

Two kinds of error can arise in optic flow estimation. The more fundamental errors arise from model inadequacy with respect to the scene content. For example, occlusion and revealing effects, changes in object appearance or attitude, and illumination changes, can all confound optic flow-based estimation. However, motion measurements for those portions of a scene subject to translational motion, and hence in which the optic flow model is adequate, can also be inaccurate. One well-documented example is the aperture problem. Optic flow estimation is always conducted over some support region, usually a rectangular block. The aperture problem arises when the block is small relative to the object features in the scene. Motion estimation in a region falling wholly inside a spatially uniform moving object can be expected to return zero optic flow. Furthermore, where a motion estimation block spans a straight edge of a moving object, the estimate is only reliable perpendicular to the edge, and quite unreliable along the direction of the edge.

As the work described here is fundamentally concerned with using motion vectors for image sequence analysis, some way of assessing the confidence of the vectors is needed. The focus here is on finding errors due to model inadequacy; vectors affected by the aperture problem are in some sense the right answer for local optic flow, though they do not reflect the true motion.

7.5.1 Image Gradient Confidence

Optic flow methods most commonly exploit spatial gradients in the image sequence. In many cases, the amount of gradient in a block can indicate the confidence associated with the motion

measurement at that block. To assess the vector confidence correctly, the magnitude of the reported vector must be related to the scale of the gradient in the block. Assuming that the image sequence is somewhat corrupted by spatial noise, it can be assumed that a vector of 0.5 or 1 pixel magnitude can only be accorded a high confidence if there is gradient at high spatial frequency in the sequence. For larger magnitude vectors, lower-frequency spatial gradients are sufficient.

The first stage in this approach, then, is to compute the gradient at different scales for the support region of the vector. This is achieved by building a three-level image pyramid, through downsampling the image by factors of two and four. Only the intenstity (Y) component of the image is used here. Each level of the image pyramid is then filtered with horizontal and vertical Prewitt filters:

$$\frac{\partial}{\partial x} = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}, \quad \frac{\partial}{\partial y} = \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix} \tag{7.26}$$

The proportion of filtered pixels exceeding a threshold in the support region of the vector is found. Where I^s is the intensity image at level $s \in \{0, 1, 2\}$, define

$$G_x^s(\mathbf{x}) = \frac{\sum_{\mathbf{p} \in P(\mathbf{x})}(|\frac{\partial I^s}{\partial x}(\mathbf{p})| > \rho)}{|P(\mathbf{x})|}, \quad G_y^s(\mathbf{x}) = \frac{\sum_{\mathbf{p} \in P(\mathbf{x})}(|\frac{\partial I^s}{\partial y}(\mathbf{p})| > \rho)}{|P(\mathbf{x})|} \tag{7.27}$$

$$G^s(\mathbf{x}) = 0.5(G_x^s(\mathbf{x}) + G_y^s(\mathbf{x})) \tag{7.28}$$

G^s is then fraction of significant gradient in the region at each level. A threshold of $\rho = 12$ intensity levels is used.

The gradient-based confidence of a vector is assessed against a gradient scale according to the magnitude of the vector, as follows:

$$Grad(\mathbf{x}) = \begin{cases} G^0 & \text{if } \|D(\mathbf{x})\| < 2; \\ G^1 & \text{if } \|D(\mathbf{x})\| < 4; \\ G^2 & \text{otherwise} \end{cases} \tag{7.29}$$

The gradient score is mapped to a confidence using

$$C_g(\mathbf{x}) = G_{(0.15,0.003)}(Grad(\mathbf{x})) \tag{7.30}$$

where G is the Gaussian cdf defined in 7.16. Hence, vectors whose support region has 15% significant gradient at the relevant scale are accorded a gradient-based confidence of 0.5.

This gradient-based confidence measure does not incorporate information on the orientation of the gradient in each site. As described in chapter 2, motion can only be accurately estimated perpendicular to image gradient (the aperture effect); as described there, the numerical stability of the optic flow equation has been incorporated into local motion estimation schemes to account for this. Irani *et al.*exploit this fact for vector confidence assessment, as described in [158].

7.5.2 Vector Confidence via the Mirror Constraint

At a site $\mathbf{x}$ inside an object whose motion is being tracked by the motion estimator, the forward vector $\mathbf{D}_n^f(\mathbf{x})$ should be the mirror of the backward vector at $\mathbf{D}_{n+1}^b(\mathbf{x}+\mathbf{D}_n^f(\mathbf{x}))$. A suitable diagnostic for estimator failure, including revealing / occlusion effects, is then the *mirror constraint* [178]:

$$\mathbf{D}_n^f(\mathbf{x}) + \mathbf{D}_{n+1}^b(\mathbf{x}+\mathbf{D}_n^f(\mathbf{x})) = 0 \tag{7.31}$$

$$\mathbf{D}_n^b(\mathbf{x}) + \mathbf{D}_{n-1}^f(\mathbf{x}+\mathbf{D}_n^b(\mathbf{x})) = 0 \tag{7.32}$$

To measure the extent to which a vector violates the mirror constraint, define the forward disparity by

$$Disp_n^f(\mathbf{x}) = \frac{\left\| \mathbf{D}_n^f(\mathbf{x}) - \mathbf{D}_{n+1}^b(\mathbf{x}+\mathbf{D}_n^f(\mathbf{x})) \right\|}{\left\| \mathbf{D}_n^f(\mathbf{x}) \right\|} \tag{7.33}$$

and similarly for the backward vectors. The disparity is mapped to a confidence score using

$$C_m^f(\mathbf{x}) = 1 - G_{(0.5,0.02)}(Disp^f(\mathbf{x})) \tag{7.34}$$

where G is the Gaussian cdf defined in 7.16. Hence, vectors with a mirror disparity of 0.5 are given a confidence value of 0.5. Vectors due to revealing and occlusion generally have disparities much greater than one.

7.5.3 Vector Confidence via Vector Field Divergence

A further confidence measure is suggested by the fact that in translational motion in an image sequence, the vector field should be spatially smooth. Where multiple vectors are found to point to the same location, model inadequacy or motion estimation failure should be suspected. Figure 7.7 illustrates this in the case of tracking failure in a region of the *coconut* sequence.

This aspect of the vector field can be assessed via the vector divergence. Where (r, c) are the site co-ordinates and (u, v) are the vector components, the divergence is a measure of

$$\frac{\partial(u)}{\partial(r,c)} + \frac{\partial(v)}{\partial(r,c)}$$

This is computed as the weighted mean of the difference between the vector and its neighbours:

$$Div(\mathbf{x}) = \frac{\sum_{\mathbf{x}' \in \mathcal{N}(\mathbf{x})} w(\mathbf{x},\mathbf{x}') \left\| D(\mathbf{x}') - D(\mathbf{x}) \right\|}{\left\| D(\mathbf{x}) \right\| \sum_{\mathbf{x}' \in \mathcal{N}(\mathbf{x})} w(\mathbf{x},\mathbf{x}')} \tag{7.35}$$

where the weights, $w(\mathbf{x}, \mathbf{x}')$, account for the edge process and distance between the sites:

$$w(\mathbf{x},\mathbf{x}') = \frac{1 - E[\mathbf{x},\mathbf{x}']}{\|\mathbf{x}-\mathbf{x}'\|} \tag{7.36}$$

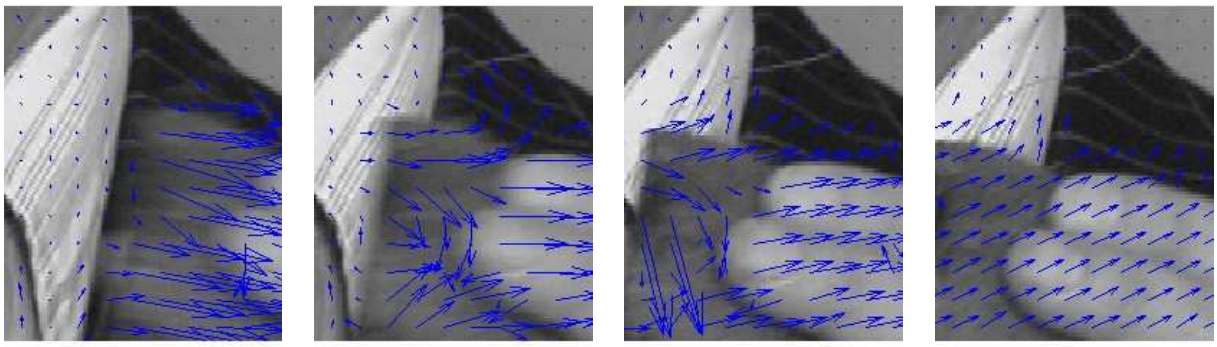


Figure 7.7: Non-unique vector destinations where tracking has failed

The divergence is mapped to a confidence score using

$$C_{\nabla}(\mathbf{x}) = 1 - G_{(0.7,0.05)}(Div(\mathbf{x})) \tag{7.37}$$

where G is the Gaussian cdf defined in 7.16. Hence, vectors with a divergence score of 0.7 are given a confidence value of 0.5.

7.5.4 Combining the Confidence Measures

The three confidence measures defined above are combined to give a confidence measure for each vector. For the backward vectors, the confidence is

$$C^b(\mathbf{x}) = G_{(1.2,0.3)}(0.5C^b_g(\mathbf{x}) + 2C^b_m(\mathbf{x}) + C^b_{\nabla}(\mathbf{x})) \tag{7.38}$$

and similarly for the forward vectors. The coefficients of each confidence component reflect their relative reliability. The function defined in 7.16 is applied with $\mu = 1.2$, $\sigma^2 = 0.3$ to map these values to the range $0, 1$.

7.6 Encouraging Smoothness

Spatially or temporally adjacent sites should in the most part have the same label; this intuition informs the spatio-temporal smoothness prior on configurations of the MRF. The smoothness likelihood is factorised into the spatial and temporal components,

$$P(L_n(\mathbf{x})|L_{n-1}, L_n(-\mathbf{x})) = P(L_n(\mathbf{x})|L_{n-1})P(L_n(\mathbf{x})|L_n(-\mathbf{x})) \tag{7.39}$$

and each component is assessed separately.

If a label l is assigned to a site $\mathbf{x}$, temporal smoothness suggests that the motion compensated location of the site in the previous frame should have been assigned the same label l, or at least

be close to a site with the label l. This is assessed using

$$S^t(\mathbf{x}) = \underset{\mathbf{x}' \in \{L_{n-1}=L_n(\mathbf{x})\}}{\arg\min} \left(\left\| \mathbf{x} + D_n^b(\mathbf{x}) - \mathbf{x}' \right\| \right) \tag{7.40}$$

which measures the minimum distance between the motion compensated location of the site $\mathbf{x}$ in frame $n-1$ and the sites having label $L(\mathbf{x})$ in the L_{n-1}. The temporal smoothness prior is then given by

$$P(L_n(\mathbf{x})|L_{n-1}) \propto exp(-\frac{S^t}{2\hat{\sigma}_t}) \tag{7.41}$$

$\hat{\sigma}_t$ is a constant chosen to representation the expected within-segment variance of the temporal distance. A value of 0.5 is used.

For spatial smoothness, define the spatial smoothness score S^s by

$$S^s(\mathbf{x}) = \frac{\sum_{\mathbf{x}' \in \mathcal{N}(\mathbf{x})} w(\mathbf{x}, \mathbf{x}')[L(\mathbf{x}') \neq L(\mathbf{x})]}{\sum_{\mathbf{x}' \in \mathcal{N}(\mathbf{x})} w(\mathbf{x}, \mathbf{x}')} \tag{7.42}$$

where the weights $w(\mathbf{x}, \mathbf{x}')$ are defined by

$$w(\mathbf{x}, \mathbf{x}') = \frac{C^l(\mathbf{x}')(1 - E[\mathbf{x}, \mathbf{x}'])}{\|\mathbf{x} - \mathbf{x}'\|} \tag{7.43}$$

These weights take into account the edge process and inter-site distance, just as those in equation 7.36. C^l is the label confidence field, introduced in the next section. The spatial smoothness prior is then evaluated using

$$p(L(\mathbf{x})|L(-\mathbf{x})) \propto exp\left(-\frac{S^s}{2\hat{\sigma}_s}\right) \tag{7.44}$$

7.7 Label Assignment Confidence

It is found in some cases that a particular site has very similar model-membership likelihoods for two or more models. This implies that the site is plausibly a member of either of these models, which in turn can be interpreted as a low confidence in the label assignment. The implications for smoothness are that sites with a low confidence should not influence their neighbours in the smoothness computation, and the incorporation of the label confidence, C^l, provides this. The label confidence is defined as

$$\frac{p(I(\mathbf{x}), D(\mathbf{x}), L_{n-1}|L(\mathbf{x}) = l_1)}{p(I(\mathbf{x}), D(\mathbf{x}), L_{n-1}|L(\mathbf{x}) = l_2)} \tag{7.45}$$

where l_1 and l_2 are the two label assignments resulting in the highest posterior probability, excluding spatial smoothness, for site $\mathbf{x}$:

$$\begin{aligned} l_1 &= \underset{l \in L_n}{\arg\max} \left(\Pr(I(\mathbf{x}), D(\mathbf{x}), L_{n-1}|L(\mathbf{x}) = l) \right) \\ l_2 &= \underset{l \in L_n, l \neq l_1}{\arg\max} \left(\Pr(I(\mathbf{x}), D(\mathbf{x}), L_{n-1}|L(\mathbf{x}) = l) \right) \end{aligned} \tag{7.46}$$

Rather than take the ratio of probabilities, the difference of the negative log probabilities is used:

$$c(\mathbf{x}) = -\log(p(I(\mathbf{x}), D(\mathbf{x}), L_{n-1}|L(\mathbf{x}) = l_2)) - -\log(p(I(\mathbf{x}), D(\mathbf{x}), L_{n-1}|L(\mathbf{x}) = l_1)) \quad (7.47)$$

These values are mapped to a confidence in the range $0, 1$

$$C^l(\mathbf{x}) = G_{(log(2)/2,0.01)}(c(\mathbf{x})) \quad (7.48)$$

A difference of $\log(2)/2$ in the negative log likelihood corresponds to a difference factor of $\sqrt{2}$ in the probabilities, which maps to a confidence of 0.5. A two-fold, or greater, difference in the likelihoods maps to a confidence value of 1.

It is possible to further impose that sites with low label confidence should be more easily influenced by their neighbours in the smoothness computation, by defining

$$S_1^s(\mathbf{x}) = C^l(\mathbf{x})^{-1} S^s(\mathbf{x}) \quad (7.49)$$

This boosts the smoothing energy at sites with low label confidence. However, where a site has low label confidence due to the high likelihood of two models, the smoothness boost can result in a third, low-likelihood model being assigned at the smoothing stage, and so S_1^s is not used.

7.8 Changes in the Motion Content

The video segmentation approach described here is essentially concerned with tracking coherent segments from one frame to the next. Segments are tracked on the assumption that the motion and colour of a segment is stable from frame to frame. However, this assumption is invalidated when objects in the scene change direction. Figure 7.8 (a) illustrates this occurrence in four frames from the *coconut* sequence. As the directions of motion of the hands change, the label assignments of both hands changes over the four frames shown. This is because the motion changes so rapidly that the vector models for the existing labels becomes 'out of date', in that they are a poor match for the new motion. This results a new label being introduced to match the new motion.

To allow for this effect, the motion model for all labels is updated at the start of processing of each frame until the model matches some of the expected sites in the frame. If the motion of an object with label l changes between frame $n-1$ and n, the corresponding sites will exhibit a poor fit to the vector model for label l. To assess this, the mean vector fit is introduced, given by

$$\hat{F}_{n,k}(l) = \frac{\sum_{\mathbf{x} \in L_n = l} -\log(P(\mathbf{M}_n^k(\mathbf{x})|L_n(\mathbf{x})))C_n^k(\mathbf{x})}{\sum_{\mathbf{x} \in L_n = l} C_n^k(\mathbf{x})} \quad (7.50)$$

which is a measure of how well the vectors fit the vector model in frame n and direction k. The vector model is that computed in equations 7.21 and 7.22, and $P(\mathbf{M}_n^k(\mathbf{x})|L_n(\mathbf{x}))$ is then given by equation 7.24.

When processing of frame n commences, the vector models and mean vector confidence for the labels in $\mathcal{L}_{n-1}$ are known. Whether the motion within a label has changed is then assessed by examining the vectors in the appropriate locations in frame n. The set of expected sites for a label l in frame n is given by

$$S_{l,n} = \{\mathbf{x} : \mathbf{x} \in L_{n-1} = l\} + \boldsymbol{\mu}^f_{n-1,l} \tag{7.51}$$

where $\boldsymbol{\mu}^f_{n-1,l}$ is the mean forward vector for label l in frame $n-1$. The approach adopted consists in assuming that most of these sites correspond to the object being tracked with label l, and increasing the variance of the model until some proportion of these sites exhibit a good fit to the model. The procedure is a sequence of iterations over

$$F^i = \left\{ -\log(P(\mathbf{M}^k_n(\mathbf{x})|L_n(\mathbf{x}) = l)) : \mathbf{x} \in S_{l,n} \right\} \tag{7.52}$$

$$F^i_p = \frac{|\{n : n \in F^i \wedge n < \hat{F}_{n-1,k}(l)\}|}{|F^i|} \tag{7.53}$$

$$(\mathbf{R}^k_{n-1})^i = 1.05(\mathbf{R}^k_{n-1})^{i-1} \tag{7.54}$$

F^i is the set of vector fits to the model in the expected sites. F^i_p is then the proportion of those sites which have a fit better than the mean for that model. The iteration is terminated when F^i_p is greater than 0.1. At each iteration, the covariance matrix for the model is scaled by a factor of 1.05.

Figure 7.8 (b) shows how increasing model covariances to account for changes in object motion at the start of processing of each frame enables the label assignment to the hands in a section of the *coconut* sequence to remain stable.

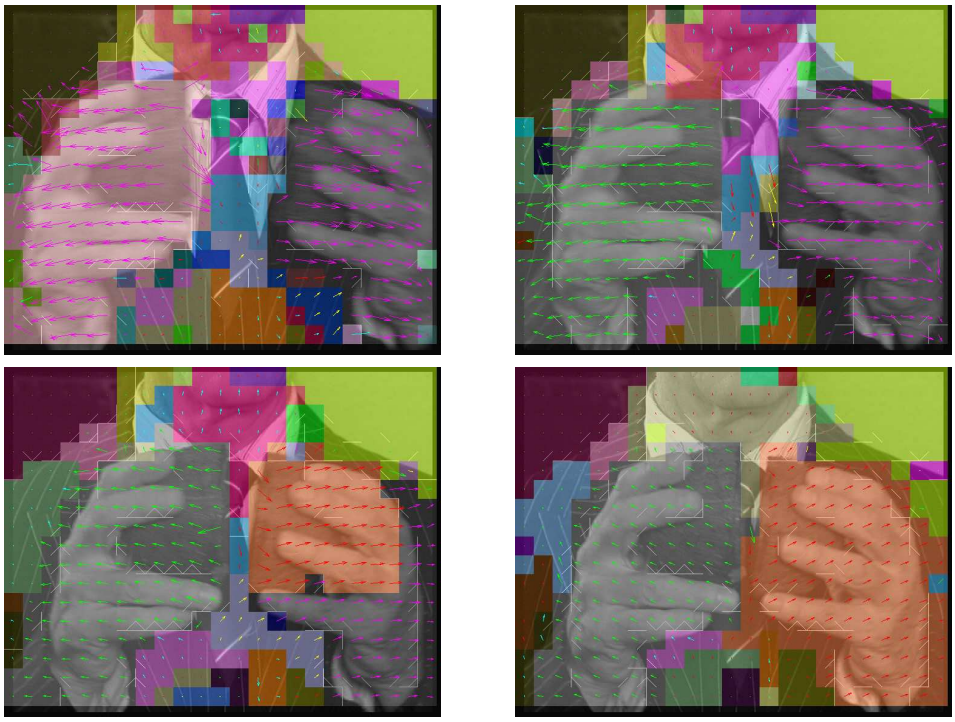
7.9 The Segmentation Process in Outline

The overall segmentation scheme can now be outlined as follows. It is desired to segment frame n, given a set of labels $\mathcal{L}_n$ and their associated models. First, the fixed energies for each label are found at each site:

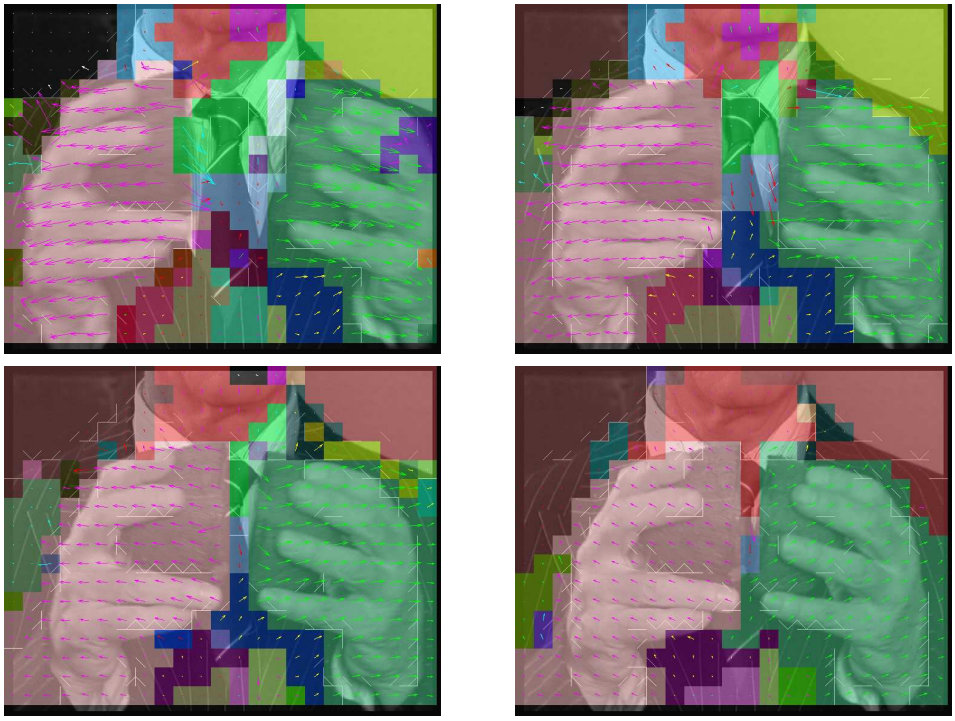
$$\begin{aligned} F_n(\mathbf{x}, l) =& C^l(\mathbf{x}) \left(-\log\left(\Pr(\mathbf{I}(\mathbf{x})|L(\mathbf{x}) = l)\right)\right) \\ & -C^l(\mathbf{x})C^b(\mathbf{x}) \log\left(\Pr(\mathbf{D}^b(\mathbf{x})|L(\mathbf{x}) = l)\right) \\ & -C^l(\mathbf{x})C^f(\mathbf{x}) \log\left(\Pr(\mathbf{D}^f(\mathbf{x})|L(\mathbf{x}) = l)\right) \\ & -\log\left(\Pr(L_n(\mathbf{x}) = l|L_{n-1})\right) \end{aligned} \tag{7.55}$$

The label field is then initialised, with each site being assigned the label having the lowest fixed energy:

$$L^0_n(\mathbf{x}) = \arg\min_{l \in \mathcal{L}_n} F(\mathbf{x}, l) \tag{7.56}$$



(a) Segmentation without adaptive vector model variance



(b) Segmentation with adaptive vector model variance

Figure 7.8: The effect of adaptive vector model variance. Each figure shows four frames, reading left to right, top to bottom.

Then the spatial smoothness of the label field is refined on a site-by-site basis. The Iterative Conditional Modes scheme [325] is employed here. Each site is updated according to

$$L_n^{i+1}(\mathbf{x}) = \arg\min_{l \in \mathcal{L}_n} F(\mathbf{x}, l) - \Lambda \log\left(p(l|L_n^i(-\mathbf{x}))\right) \tag{7.57}$$

Λ here controls the relative influence of the spatial smoothness prior; a value of 8 is used. Several iterations over the entire label field are made. The sites are visited in a chequerboard-style scan skipping over two sites in each row and column.

Once L_n has been established, the data models for each label are updated as described previously. Before the next frame can be processed, however, the label field is examined for new motions, where an existing model should be split, and model convergence, where two models exhibit the same motion and should be merged. These topics are the subject of the next section.

7.10 New Clusters and Refining the Segmentation

For this application it is not known in advance how many models are required to track the motion content of a given image sequence, and the number of models required will vary over the course of the sequence. Thus new labels must be introduced over the course of the sequence. Two approaches for the introduction of new labels are described here.

7.10.1 The No-Match Label

Wherever a site is a poor match for all existing models, it is assigned a new label. The site update equation is then

$$l' = \arg\max_{l \in \mathcal{L}} \left(\Pr(L|_{\mathbf{x}=l})\right) \tag{7.58}$$

$$L(\mathbf{x}) = \begin{cases} l', & -\log\left(\Pr(L|_{\mathbf{x}=l'})\right) < \rho \\ \max(\mathcal{L}) + 1, & \text{otherwise} \end{cases} \tag{7.59}$$

Thus all sites that are a poor match for all existing models are assigned the same new label. The threshold ρ is determined by

$$\rho = \rho_I + \rho_D(\mathbf{x}) + \rho_{S^t} \tag{7.60}$$

Here ρ is the sum of the out-of-class energies for the colour, vector, and temporal smoothness data. For the colour and temporal smoothness energies, these are constants derived from $\hat{\sigma}^2_{k \in \{L,u*,v*\}}$ and $\hat{\sigma}^2_t$:

$$\rho_I = \sum_{k \in \{L,u*,v*\}} -\log(\Pr_k(2.576\hat{\sigma}_k)) \tag{7.61}$$

$$\rho_t = -\log(\Pr_t(2.576\hat{\sigma}_t)) \tag{7.62}$$

Here k is a random variable drawn from a normal distribution with mean 0 and standard deviation $\hat{\sigma}_k$, i.e. $k \sim \mathcal{N}(0, \hat{\sigma}_k)$, and similarly $t \sim \mathcal{N}(0, \hat{\sigma}_t)$. 2.576 standard deviations corresponds to the 99% confidence interval. The idea is if that all label assignments to a site result in an EMD or temporal smoothness outside the 99% confidence interval, the site should be assigned a new label.

In the case of the vector data, the covariance of each vector model is different, and one of these must be chosen to calculate the out-of-class energy. Therefore the two labels l_1, l_2 having the least fixed energy at the site $\mathbf{x}$ are found, as in equation 7.46, and of these the vector model with a better fit for the vector at site $\mathbf{x}$ is used to calculate the out-of-class energy.

$$l' = \arg\max_{l \in \{l_1, l_2\}} -\log\left(\Pr_{\mathbf{l}'}(D^k(\mathbf{x}))\right) \tag{7.63}$$

$$\rho_{D^k} = -\log(\Pr_{\mathbf{l}'}([0, 2.576]\mathbf{R}_{l'})) \tag{7.64}$$

where $\mathbf{l}' \sim \mathcal{N}(\mathbf{0}, \mathbf{R}_{l'})$.

The 'no-match' label corresponds to new regions in the frame, but it is not known how many new regions are involved. As a first measure to resolve this, all spatially contiguous regions having the 'no match' label are assigned unique labels. The approach described below for splitting existing clusters can also be applied.

7.10.2 Splitting Existing Clusters

In the course of video segmentation it can arise that one label corresponds to two distinct motions, for example where two objects moving alongside each other (and therefore assigned the same label) split apart. To address this, a cluster splitting scheme was investigated. Clusters are split by fitting a Gaussian Mixture Model to their vectors. Either the backward or forward vectors may be used depending on which direction has the greater average confidence C^b, C^f within the cluster:

$$\frac{1}{|L = \mathbf{x}|} \sum_{\mathbf{x} \in L = \mathbf{x}} C^k(\mathbf{x}) \tag{7.65}$$

In the following, $D(\mathbf{x})$ should be understood to mean either $D^b(\mathbf{x})$ or $D^f(\mathbf{x})$, depending on the direction of higher confidence, and similarly $C(\mathbf{x})$ represents the appropriate vector confidence. Colour information is not used in cluster splitting, as a coherently moving object can often consist of distinctly coloured regions.

A Gaussian Mixture Model with N centres takes the form

$$\theta = \{\pi_c, \mu_c, \mathbf{R}_c | c = 1 \ldots N\} \tag{7.66}$$

where π_c, μ_c, and $\mathbf{R}_c$ are the mixture coefficient, centre mean, and centre covariance of component c. The mixture model is initialised using the K-means technique, and refined using the Expectation Maximisation (EM) algorithm. Clustering is applied using the Cartesian $[dx, dy]$

representation of the vectors, as the toroidally connected nature of the polar vector space makes finding the optimal set of mixture components difficult.

Consider that the cluster with label l is to be split. The procedure commences with the K-means algorithm, which determines a set of cluster centres $\mathcal{C}$ minimising the objective function

$$J = \sum_{\mathbf{c}\in\mathcal{C}} \sum_{\mathbf{x}\in L=l} \sigma_{\mathbf{c}}(\mathbf{x}) C(\mathbf{x}) \left\| D(\mathbf{x}) - \mathbf{c} \right\| \tag{7.67}$$

Above and in the following, the vector confidence $C(\mathbf{x})$ is used as a weight so that the cluster splitting is not influenced by erroneous vectors. To start the algorithm, a set of initial cluster centres $\mathcal{C}^0$ is found. The procedure then iterates over

$$\sigma_c^i(\mathbf{x}) = \begin{cases} 1, & \mathcal{C}_c = \arg\ \min_{\mathbf{c}\in\mathcal{C}^i} \left\| D(\mathbf{x}) - \mathbf{c} \right\| \} \\ 0, & \text{otherwise} \end{cases} \tag{7.68}$$

$$\mathcal{C}_c^{i+1} = \frac{\sum_{\mathbf{x}\in L=l} \sigma_c(\mathbf{x}) C(\mathbf{x}) D(\mathbf{x})}{\sum_{\mathbf{x}\in L=l} \sigma_c(\mathbf{x}) C(\mathbf{x})} \tag{7.69}$$

Here data are assigned to a cluster centre, and then the cluster centres are moved to reflect the data assignment. Iteration terminates when the cluster centres remain stationary.

The EM procedure for Gaussian mixture models is similar to K-means, with soft cluster assignments. This algorithm optimises the mixture model parameters θ. Firstly, these parameters are initialised using the results of the K-means algorithm:

$$\begin{aligned} \pi_c^0 &= \frac{\sum_{\mathbf{x}\in L=l} \sigma_c(\mathbf{x})}{\sum_{\mathbf{x}\in L=l} 1} \\ \mu_c^0 &= \frac{\sum_{\mathbf{x}\in L=l} \sigma_c(\mathbf{x}) C(\mathbf{x}) D(\mathbf{x})}{\sum_{\mathbf{x}\in L=l} \sigma_c(\mathbf{x}) C(\mathbf{x})} \\ \mathbf{R}_c^0 &= \frac{\sum_{\mathbf{x}|\sigma_c(\mathbf{x})=1} (C(\mathbf{x}))^2 (D(\mathbf{x}) - \mu_c^0)(D(\mathbf{x}) - \mu_c^0)^T}{\sum_{\mathbf{x}|\sigma_c(\mathbf{x})=1} (C(\mathbf{x}))^2} \end{aligned} \tag{7.70}$$

For initialisation, the cluster membership functions $\sigma_c(\mathbf{x})$ are assumed to be those found at the last iteration of the K-means procedure. Within the EM function, they are unknowns. The *E-step* consists of calculating the soft membership functions at each site, according to

$$h_c^i(\mathbf{x}) = \frac{P(\mathbf{D}(\mathbf{x})|\sigma_c(\mathbf{x}) = 1, \theta^i)\pi_c^i}{\sum_{c'\in\mathcal{C}^i} P(\mathbf{D}(\mathbf{x})|\sigma_{c'}(\mathbf{x}) = 1, \theta^i)\pi_{c'}^i} \tag{7.71}$$

where

$$P(\mathbf{D}(\mathbf{x})|\sigma_c(\mathbf{x}) = 1, \theta^i) = \left((2\pi)^2 |\mathbf{R}_c^i|\right)^{-1/2} \exp\left\{ -\frac{1}{2} (D(\mathbf{x}) - \mu_c^i)(\mathbf{R}_c^i)^{-1} (D(\mathbf{x}) - \mu_c^i) \right\} \tag{7.72}$$

The *M-step* then updates the model parameters, using

$$\begin{aligned}
\mu_c^{i+1} &= \frac{\sum_{\mathbf{x}|\sigma_c(\mathbf{x})=1} h_c^i(\mathbf{x}) C(\mathbf{x}) D(\mathbf{x})}{\sum_{\mathbf{x}|\sigma_c(\mathbf{x})=1} h_c^i(\mathbf{x}) C(\mathbf{x})} \\
\mathbf{R}_c^{i+1} &= \frac{\sum_{\mathbf{x}|\sigma_c(\mathbf{x})=1} h_c^i(\mathbf{x}) (C(\mathbf{x}))^2 (\mathbf{x} - \mu_c^{i+1})(\mathbf{x} - \mu_c^{i+1})^T}{\sum_{\mathbf{x}|\sigma_c(\mathbf{x})=1} h_c^i(\mathbf{x}) (C(\mathbf{x}))^2} \\
\pi_c^{i+1} &= \frac{1}{\sum_{\mathbf{x} \in L=l} 1} \sum_{\mathbf{x}|\sigma_c(\mathbf{x})=1} h_c^i(\mathbf{x})
\end{aligned} \tag{7.73}$$

When a cluster with label l has been split, the largest subcluster is assigned the existing label l, and the remaining subclusters are given new labels.

The question of how many subclusters a given cluster should be split into is an instance of the model order selection problem. Numerous approaches to this problem have been proposed, including the Akaike Information Criterion, Minimum Description Length, and the Bayes Information Criterion. A simpler approach, popular for model order selection by inspection, is V-fold cross validation (RANSAC). Here the evidence for each proposed order is assessed by repeatedly partitioning the data into a large training set (here containing 90% of the vectors) and a test set. A model of order N is fit to the vectors in the training set, and then the model fit error for the vectors in the test set is found against this model. A model of excessively low (respectively high) order will underfit (respectively overfit) the test set, resulting in a high error for the training set vectors. Thus test-set fit error against model order N follows a concave curve; the model order corresponding to minimum test-set error is chosen.

The V-fold cross-validation scheme was evaluated for choosing the cluster splitting order, but was rarely found to give good results. This is attributed to the small size of the data set (number of sites / vectors in a label, typically 50–100), which makes accurately estimating the model order difficult. The scheme finally chosen was to split clusters into three parts, and rely on cluster merging and site smoothing to discard superfluous subclusters. This is similar to the approach taken by Wang and Adelson in [321], where the first frame of the sequence is deliberately oversegmented and models that are very similar are merged. Model merging is described next.

7.10.3 Merging Clusters

A further refinement to labelfield is to consider merging similar clusters. At various stages in processing the video, oversegmentation can arise. This impacts negatively on performance, and makes interpreting the segmentation result more difficult. Cluster merging is invoked to mitigate the effects of oversegmentation.

Testing for whether two clusters should be merged is based principally on the vector distribution of the clusters. Whether the backward or forward vector distribution is used depends on the confidence associated with the vectors. Let l_1 and l_2 be labels whose models are being

assessed for merging. The direction k used for assessment is given by

$$k = \arg\max_{k' \in \{b,f\}} \frac{\sum_{\mathbf{x} \in L=l_1} C^{k'}(\mathbf{x}) + \sum_{\mathbf{x} \in L=l_2} C^{k'}(\mathbf{x})}{\sum_{\mathbf{x} \in L=l_1} 1 + \sum_{\mathbf{x} \in L=l_2} 1} \tag{7.74}$$

The vector distributions being assessed are then $\mu^k_{l_1}, \mathbf{R}^k_{l_1}$ and $\mu^k_{l_2}, \mathbf{R}^k_{l_2}$. The distribution distance used is the cross-entropy or Kullback-Leibler distance [92], which has a closed form for Gaussian distributions:

$$\|l_1, l_2\|_{KL} = \frac{1}{2}\left(\log|\mathbf{R}^{-1}_{l_1}\mathbf{R}_{l_2}| + \mathrm{Tr}(\mathbf{R}^{-1}_{l_1}\mathbf{R}_{l_2}) + (\mu_{l_1} - \mu_{l_2})K^{-1}_{l_1}(\mu_{l_1} - \mu_{l_2})^T - N\right) \tag{7.75}$$

$\mathrm{Tr}(\mathbf{R})$ is the trace of the matrix $\mathbf{R}$, equal to the sum of the elements on the diagonal. N is the dimensionality of the distributions being compared, here 2. The direction superscript k has been omitted for clarity. As the cross-entropy is an assymetric distance, the merging criterion is

$$\frac{1}{2}\left(\|l_1, l_2\|_{KL} + \|l_2, l_1\|_{KL}\right) < \rho \tag{7.76}$$

The threshold value ρ is 3 in this work.

7.10.4 Cluster histories

A flaw in the label merging approach described above is the lack of historical information. This means that, for example, where two different objects in the video collide and remain at rest adjacent to each other for a few frames, they will be merged as they exhibit similar motion—even though they are distinct objects in the video.

To prevent this happening, a historic vector distribution is incorporated into the motion model for each label. This historic distribution is designed to be a weighted mean of the vector model over time, and is updated as a linear combination of the existing historic distribution and the current vector distribution. Define the 'history factor', F, to be the relative importance of the existing historic distribution against the present vector distribution. Then the weights for the linear combination are found by

$$a_1 = \frac{F}{1+F},\ a_2 = \frac{1}{1+F} \tag{7.77}$$

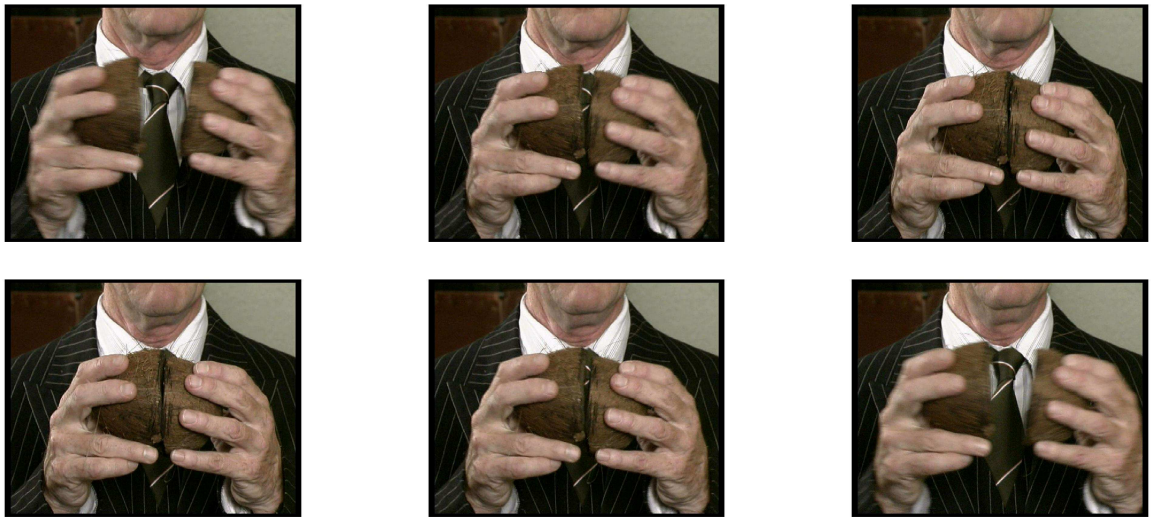
Then the historic vector distribution can be updated using

$$\boldsymbol{\mu}_t = a_1\boldsymbol{\mu}_{t-1} + a_2\boldsymbol{\mu} \tag{7.78}$$

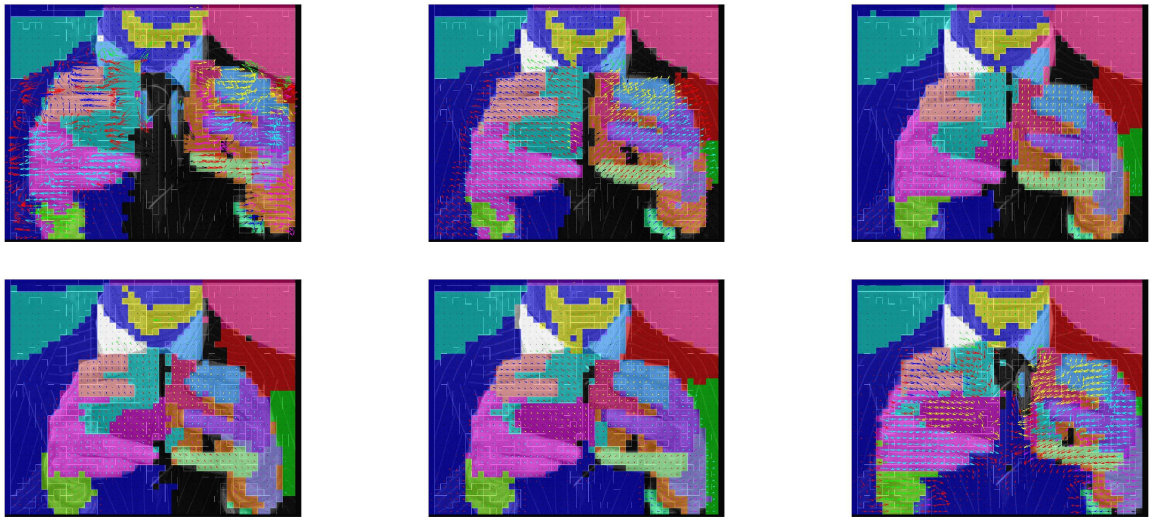
$$\mathbf{R}_t = a_1^2\mathbf{R}_{t-1} + a_2^2\mathbf{R} \tag{7.79}$$

where $(\boldsymbol{\mu}_t, \mathbf{R}_t)$ is the updated historic model for frame t, $(\boldsymbol{\mu}_{t-1}, \mathbf{R}_{t-1})$ is the historic model for the previous frame, and $(\boldsymbol{\mu}, \mathbf{R})$ is the vector model computed for vectors in the current frame. The historic vector model for a new label is initialised to be the same as the current vector model for the label.

The 'history factor' F can be chosen to be a fixed constant (e.g. 10). Alternatively, the 'age' of the label (the number of frames since the label was first introduced) can be used, such that the historic vector distribution is updated more slowly for older labels.



(a) Six frames from the *coconut* sequence, showing a coconut shell being beaten.



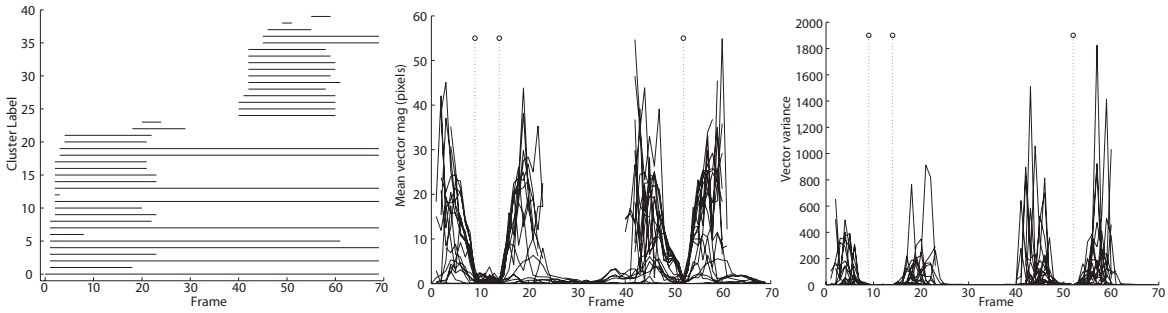
(b) Motion clusters (in colour) and motion vectors for the frames above.

Figure 7.9: Clusters in the *coconut* sequence.

7.11 Segmentation Assessment and Analysis

The goal of the video segmentation algorithm described above is to assist in the detection of edit points. Some of the segments introduced should persistently and consistently track an edit-point generating object in the video—for example, a dancer, or the arm of a dancer. An analysis of the motion within each segment, using the motion vector and motion blur analysis techniques developed in chapter 6, would then reveal the edit points within each segment.

Figure 7.9 shows the segments detected in a number of frames in the *coconut* sequence. The



(a) Cluster lifetimes. (b) Vector magnitude within each cluster. (c) Vector variance within each cluster.

Figure 7.10: Cluster analysis in the *coconut* sequence. Ground truth edit points are shown by dotted lines.

desired edit point here corresponds to the collision of the two shells, reflected in the motion of the hands and the shells. These portions of the video are consistently and persistently tracked across the edit point.

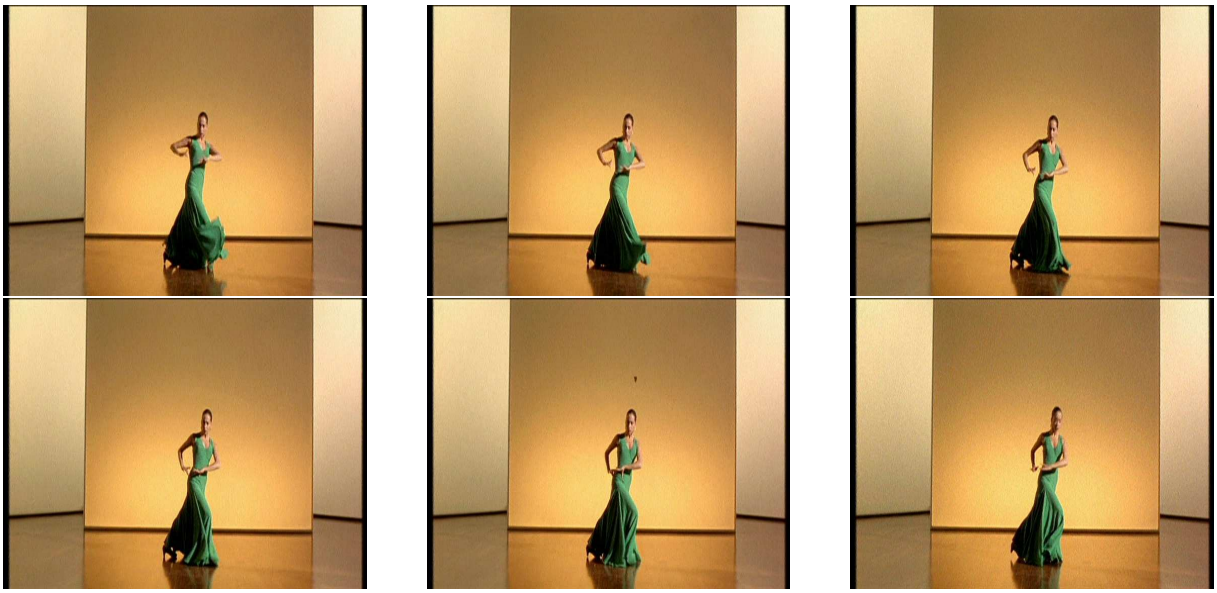
A summary of the segments created over the course of the video is shown in figure 7.10 (a). The two major episodes in this video are when the coconut shells are clapped: they are introduced from either side of the frame, banged together in the middle of the frame, and then moved outside of the frame area. This occurs twice in the sequence. These two episodes are clearly visible in this graph, in the segments with labels 1–21 (frames 1–23), and the segments with labels 24–37 (frames 40–62).

The edit points in the sequence are detected by analysis of the motion of the segments. Figures 7.10 (b) and (c) show the mean vector magnitude and vector variance traces for each cluster in the sequence. The edit points in the sequence, shown in vertical dotted lines in the figure, clearly correspond to minima in the traces.

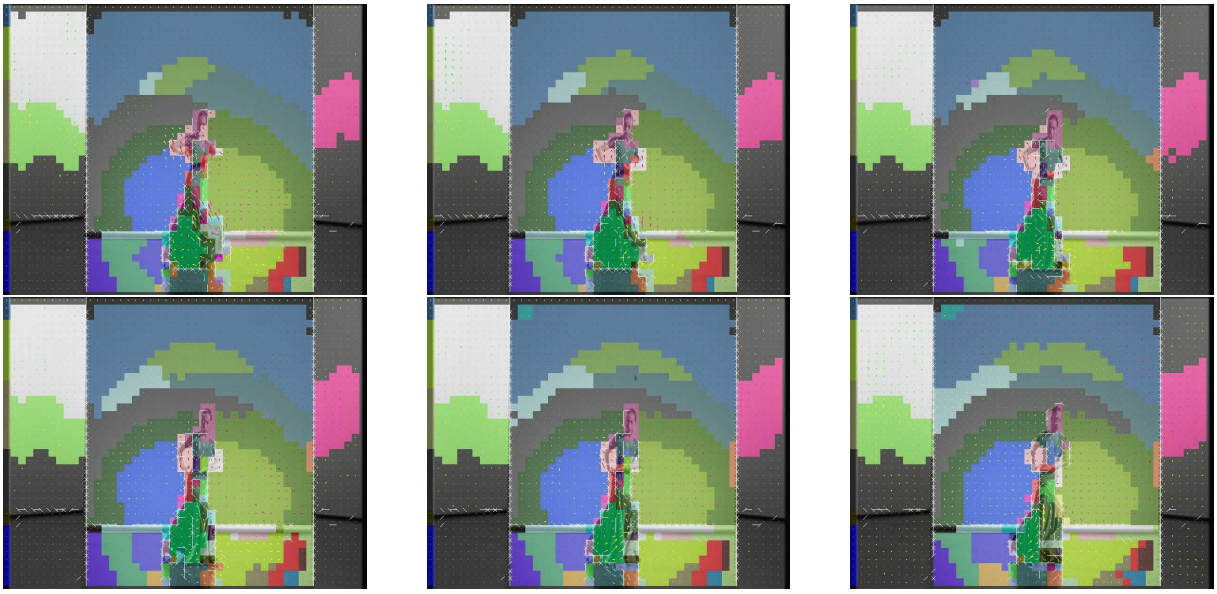
Segmentation results for more complicated sequences are more difficult to analyse for edit points. Figure 7.11 (b) shows a segmentation result from the *greenDancer* sequence. Frame 92 is an edit point, generated principally by the movement of the dancer's arms. While the dancer's right arm is tracked quite well in these frames, maintag a coherent cluster over the left arm is more difficult. Some oversegmentation of the background is visible; this is due principally to a change in the scene illumination at the start of the sequence.

Figure 7.11 (c) shows a segmentation of the same frames generated using a 9×9 blocksize, rather than 17×17. This smaller blocksize makes it easier to track smaller features in the video, including both the dancer's arms. Here the dancer's left arm is reasonably well-tracked by a green segment over the frames shown. While segmentation at this resolution improves the performance in tracking smaller features, the computational cost involved is very high.

Segmentation results over complete sequences are presented on the DVD accompanying this

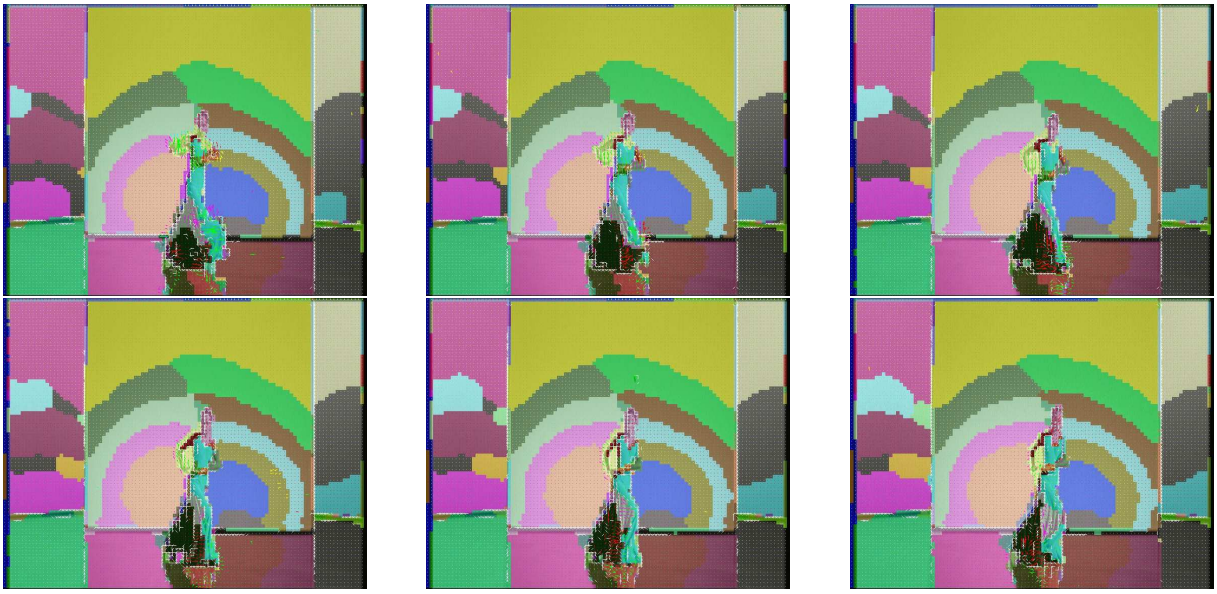


(a) Frames 84, 86, 88, 90, 92, and 94 from the *greenDancer* sequence.



(b) Segmentation result for the frames shown above.

Figure 7.11: Clusters in the *greenDancer* sequence.



(c) Segmentation result for the frames shown in figure 7.11 (a) with a 9×9 blocksize.

Figure 7.11: Clusters in the *greenDancer* sequence, with a 9×9 blocksize.

thesis. While these results do show that the segmentation technique described here performs reasonably well for motion tracking over short intervals, it is rare that a single object in the video is tracked by a single label for more than a few tens of frames. This is for a number of reasons.

Firstly, the segmentation algorithm described here involves a number of parameters. These include Λ, controlling the influence of the spatial smoothness prior; $\hat{\sigma}_k^2, k \in \{L, u*, v*, s\}$, the expected within-class variance for the colour EMD measures and temporal smoothness; the weights used to compute vector confidence; the measure used to compute the label confidence $C^l(\mathbf{x})$; parameters controlling the 'vector variance growing' approach for coping with motion changes, described in section 7.8; and parameters controlling splitting and merging, described in section 7.10.

Each of these parameters has a qualitative effect on the nature of the segmentation. The high computational cost of the algorithm makes meaningful exploration of the parameter space difficult; the present MATLAB implementation requires several days to process a sequence of several hundred frames. In particular, as the sequence is processed, the number of labels active can grow rapidly, impacting severely on the speed. High values for the spatial smoothness influence parameter, Λ, and expected within-class variance parameters $\hat{\sigma}_k^2$, reduce the likelihood of oversegmentation, but in this case smaller features in the video, such as the dancer's arms, are more likely to be merged with the background.

The cluster merging technique described in section 7.10 was introduced to counter the effects of oversegmentation and keep computation time reasonable. However, as described in that

section, merging on the basis of motion similarity within a single frame will cause distinct, temporarily adjacent objects to be merged, and so cluster merging must incorporate some notion of historic similarity of motion. This similarity of motion should incorporate at least twenty-five frames, as a pause in the motion of an object can easily last for a second. Reliably characterising the motion of a segment over a sliding temporal window of this size is difficult, in that the Gaussian used for the historic distribution can become uninformative. The necessity of incorporating a relatively long-term history to deal with object collisions is one aspect in which the segmentation problem addressed here differs from those more commonly addressed, and little work addressing this issue has appeared.

A second difficulty with the segmentation problem here is that the most informative motions in the video are those that are most difficult to track. An edit point is generally generated by a sudden change in the motion of an object in a video; this is precisely the most difficult circumstance to maintain segment coherence. The temporal smoothness prior is motion compensated, which supposes that motion is being accurately estimated for the segment; this assumption is most likely to be violated at the edit points. This behaviour suggests that in some cases, the appearance or disappearance of clusters—cluster 'churn'—will correspond to the most difficult motion discontinuities in the sequence, which in turn may correspond to edit points. However, the correspondence is too slight to be dependably exploited.

The best exploitation of a video segmentation for edit point detection is possible only when a single segment tracks the object generating the edit point with temporal consistency around the time of the edit point. When this can be achieved, analysis of the motion within the segment should reveal the edit point. The question of how to select particular segments for edit point detection has not been addressed here; the chief focus has been the development of a sufficiently general segmentation system. However, a review of the segmentation results generated suggest a number of approaches. Firstly, segments of very short temporal duration are generally due to noise in the segmentation (caused, for example, by noise in the vector field), and these can be disregarded. Segments not exhibiting motion in their lifetime correspond to video background, and can also be disregarded. Motion coherence within a segment is an indicator of relevance—for example, where a limb is sweeping through space, its motion will be smooth, whereas the motion of clothing is generally more chaotic. Finally, a number of methods for skin detection in video have been described [280]. These methods could be used to bias analysis towards those segments containing skin, which are most likely to correspond to the movements of the dancer.

7.12 Conclusion

This chapter has described a general approach to video segmentation. While reasonable segmentation performance has been achieved, obtaining a result suitable for edit point detection has proven difficult, partly because of the inherent difficulty of the task and the kind of video considered, and partly because of the high parameter count and computational cost of the

algorithm.

The question arises of whether further improvements within this scheme could yield much better results (e.g. more robust motion estimation, incorporating explicit occlusion / uncovering modelling, adjustment of the various parameters, non-parametric representation of vector distributions), or whether a different kind of segmentation approach entirely is called for. For example, a temporally smooth segmentation on the basis of motion / no motion could be sufficient to target sharpness and vector magnitude analysis on the most informative regions of the video; this is a simplification of the present approach. Alternatively, higher level modelling could be incorporated, for example explicit models of *human* motion, which introduces constraints on the spatial relationship of arms, legs, head, etc. [113]. This could help segmentation performance in the specific area of dance video, at the cost of generality.

Video segmentation remains one of the most interesting (and challenging) of video analysis tasks. However, the difficulty and computational cost of segmentation, coupled with the effectiveness of the simpler low-level techniques described in the previous chapter, suggest that at the present these low-level techniques constitute the most appropriate approach to edit point detection.

8

Applications of Edit Point Detection

Chapter 6 of this thesis described an approach to the detection of edit points in dance video. In this chapter, some applications exploiting the detected edit points are described.

8.1 Interactive Keyframe Selection

Chapter 6 described a method for combined analysis of a number of low-level traces. The results of the analysis of each trace are combined using a weight assigned to each trace. The combined analysis can then be used to generate a display of the top N edit points, i.e. the frames having the highest N values of $EP(n)$.

In that chapter, the weights used for each trace corresponded to the area under the ROC curve for peak classification in that trace. An alternative approach is to allow interactive adjustment of the weights, with a display showing the top N edit points being updated in real time in response to the selection. This offers a means of browsing keyframes, or significant phrase transitions in a dance, using reasonably intuitive concepts such as sharpness and the spatial extent of the foreground.

Rather than develop such a system in software, a number of videos have been prepared which illustrate how this application would work if implemented. Figure 8.1 shows some still images from one such video, demonstrating edit point selection in the *maleDancer2* sequence.

As described in chapter 6, the traces used for edit point detection fall into five types: the *motion* trace; the foreground bounding box traces; traces resulting from analysis of the local motion vector field; the sharpness traces ; and the audio energy trace. Each of these trace groups

is assigned a weight over the range 10^{-2} to 10^{2}. The values used are superimposed on the image selection graphically as blue vertical lines, which correspond from left to right to the groups listed above.

Each of figures 8.1 (a)-(e) shows the top twelve edit points detected in the sequence for a given choice of weights. In figure 8.1 (a) all the weights are equal, at 10^{0}. Figures 8.1 (b) and (c) show how weighting the *motion* trace by 10^{-2} and 10^{2} respectively affects the resulting set of edit points. Adjustments to this weight affect how much influence the reduction in the amount of local motion has on edit point detection, and as such it is difficult to see the effect of changing this weight using still images.

Figure 8.1 (d) illustrates the effect of setting the weight applied to the bounding box traces to 10^{2}. Here the edit point detection criterion becomes that the width or height bounding box should be at a local minimum or maximum. The edit points detected all depict poses in which the dancer's hands are outstretched, creating a local maximum in the bounding box height or width, or poses in which the dancer's hands cross in front of his body, creating a local minimum.

In figure 8.1 (e) shows the edit points selected when a weight of 10^{2} is applied to the $freq_{sharp}$ trace. Here, none of the selected frames exhibit any motion blur, as expected.

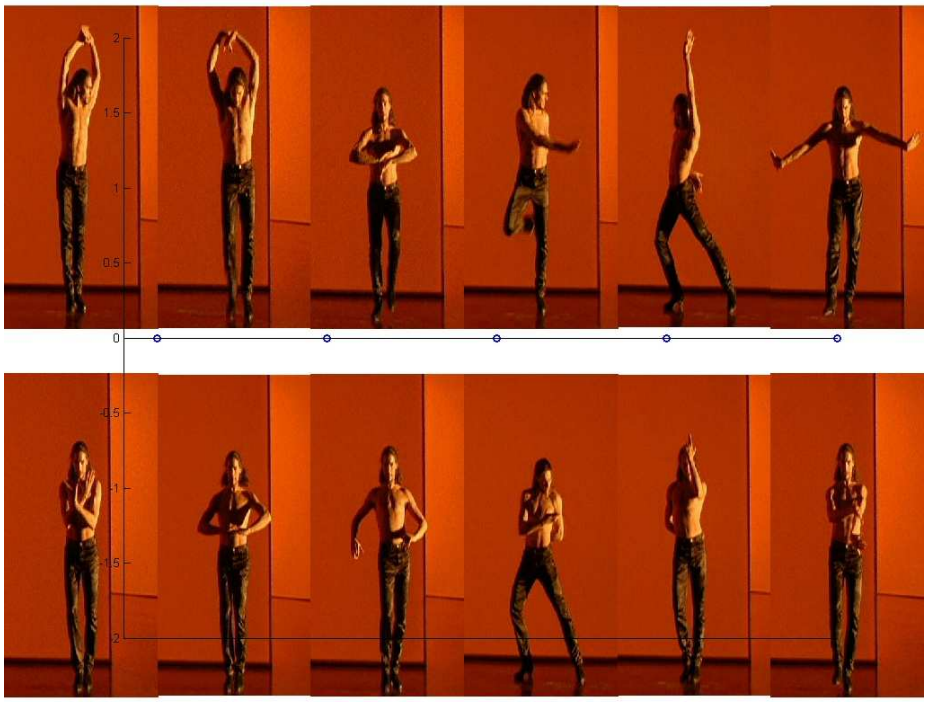
8.2 Motion Phrase Image

The edit points detected in dance video should correspond to the start and end points of phrases in dance video. This creates the possibility of using the Motion History Image (MHI) [33] technique described for snooker shot summaries in chapter 5 to create summary images representing each dance phrase, and hence a summary of the dance video itself. The effect is somewhat similar to the 'salient stills' process developed by Teodosio *et al.* [189, 215, 216, 300]. However, that work does not address the selection of frames from which to generate the summary image. By contrast, in this work the use of edit point detection means that the summary images created correspond to meaningful episodes (dance phrases) in the video.

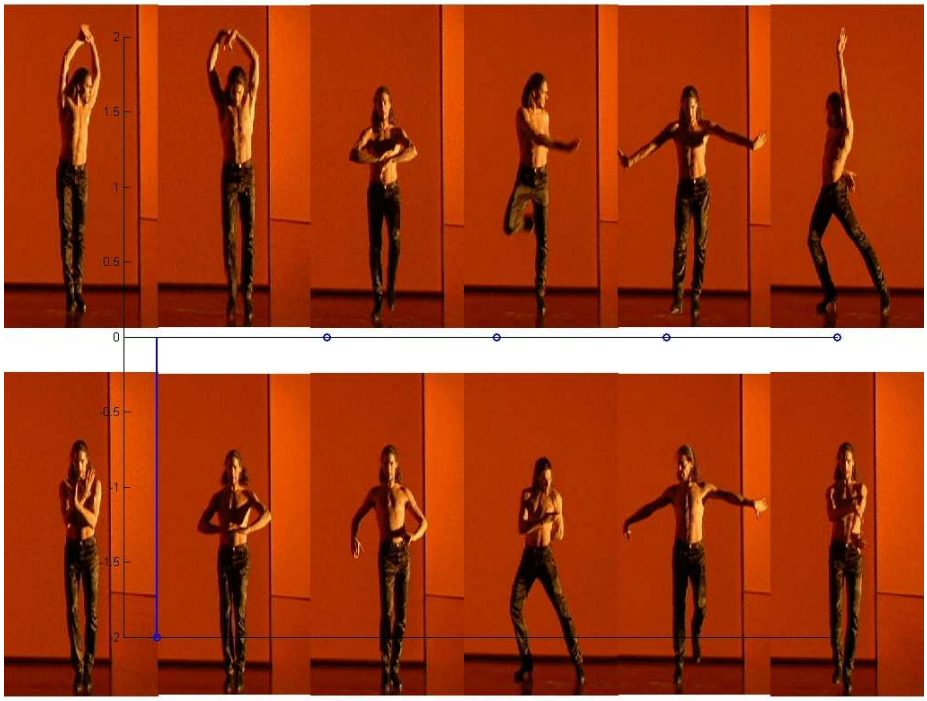
8.2.1 MHI generation with global motion

Consider that two successive edit point frames have been identified at times T_1 and T_2. The first stage in generating a summary image for this dance phrase consists in finding the MHI for this section of the video. In the case of snooker footage, the most informative shots were filmed with a static camera. However, in dance video, some camera motion is common, and thus compensation for global motion must be introduced. Figure 8.2 illustrates the essential algorithm used to generate motion compensated MHIs.

The algorithm operates on each pair of frames $\{i-1, i\}$ for $i \in T_1 + 1 \ldots T_2$. For each pair, the global motion estimation technique described in chapter 2 is used to find the global motion parameters between the two frames. These parameters are then used to find the Displaced

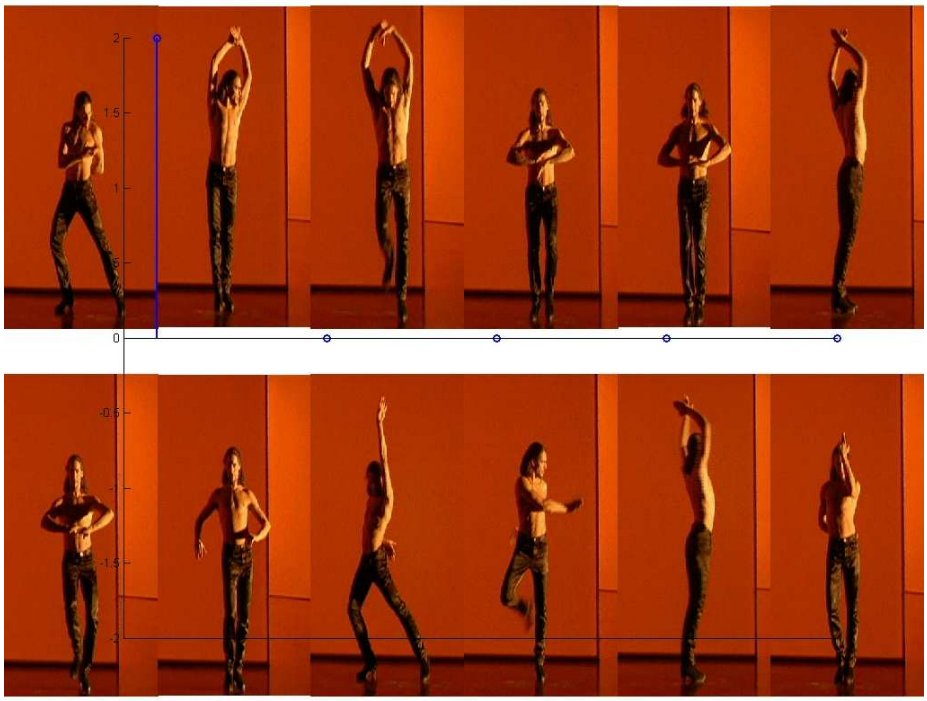


(a) All traces weighted at 10^0.



(b) *motion* weight at 10^{-2}.

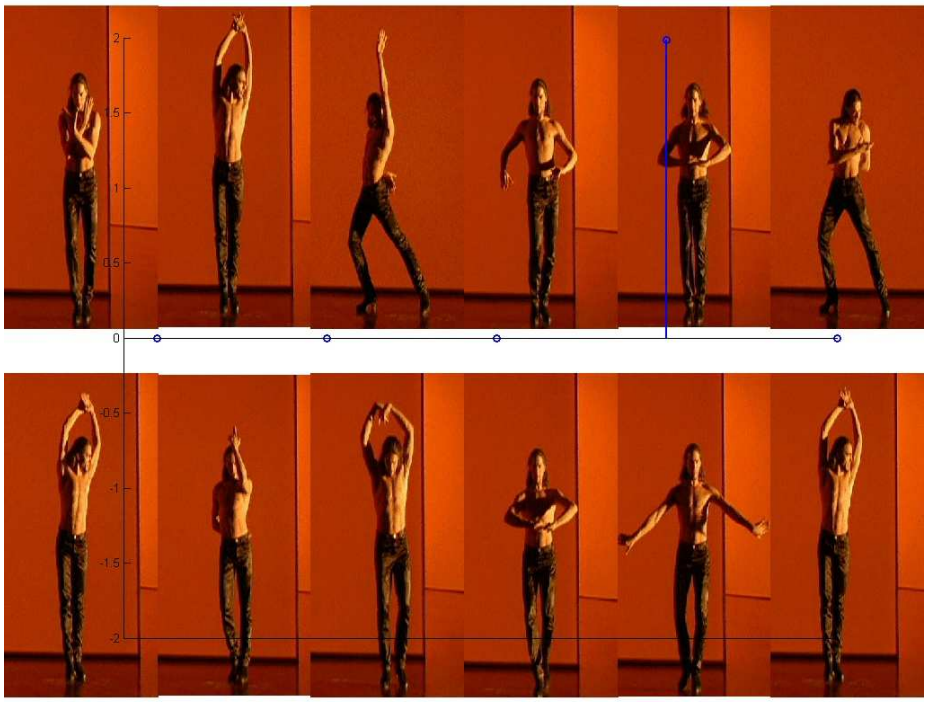
Figure 8.1: Interactive keyframe selection



(c) *motion* weight at 10^2.

(d) Bounding box traces weights at 10^2.

Figure 8.1: Interactive keyframe selection



(e) $freq_{sharp}$ trace weighted at 10^2

Figure 8.1: Interactive keyframe selection

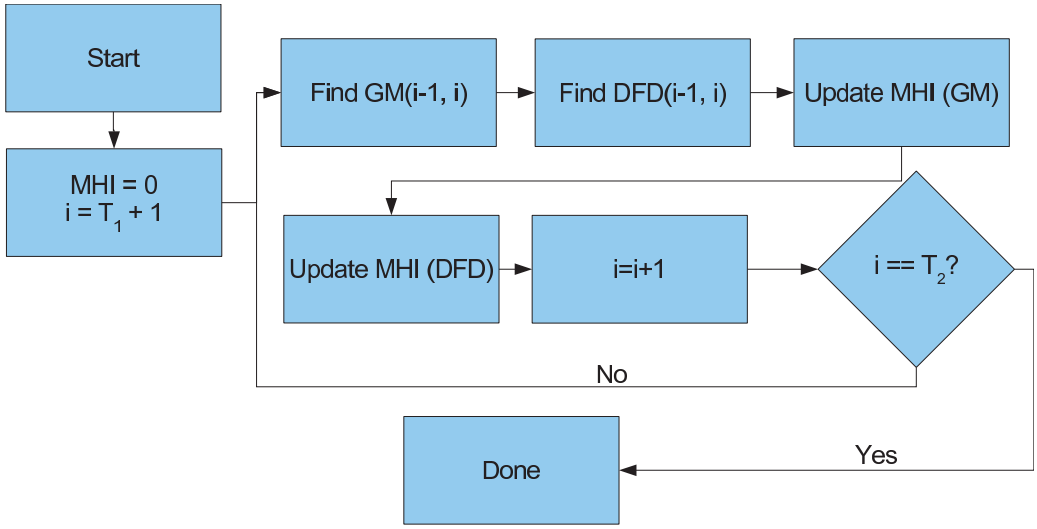
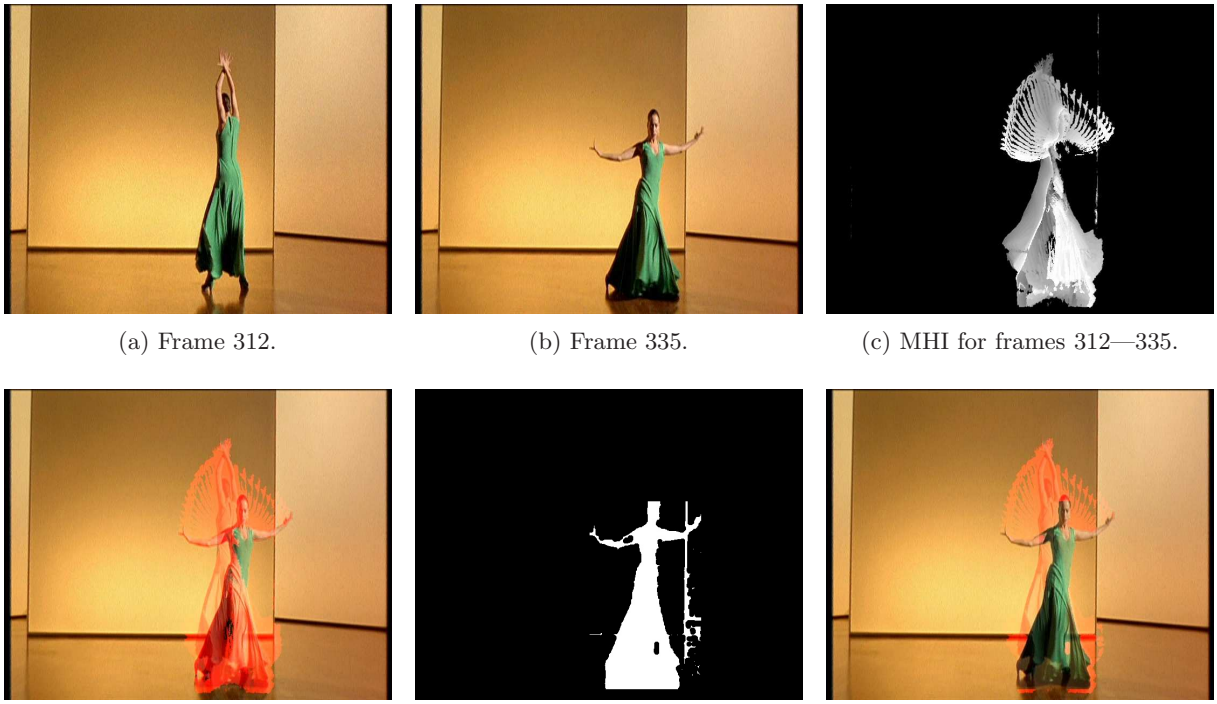


Figure 8.2: MHI generation with global motion compensation.



(a) Frame 312. (b) Frame 335. (c) MHI for frames 312—335.

(d) MHI overlaid on composite. (e) Foreground map. (f) Refined MHI overlaid on composite.

Figure 8.3: Dance summary image generation using the MHI.

Frame Difference (DFD) between the two frames. The parameters are also used to compensate the existing MHI to align it with the present frame. The DFD image is then used to update the MHI, according to

$$\mathrm{MHI}(\mathbf{x},t) = \begin{cases} 255 & \text{if } \mathrm{DFD}(\mathbf{x}) > \tau_m \\ \max(0, \mathrm{MHI}(\mathbf{x}, t-1) - (255-\upsilon)/(T_2 - T_1 + 1)) & \text{otherwise} \end{cases} \tag{8.1}$$

where υ is the value to be assigned to the oldest motion regions in the MHI; $\upsilon = 64$ is used here. Figures 8.3 (a) - (c) illustrate the start and end frames of a dance phrase and the corresponding MHI.

Figure 8.3 (d) shows the result of overlaying the MHI on a composite image, generated according to

$$C^R(\mathbf{x}) = 0.3I^R_{T_1}(\mathbf{x}) + 0.7I^R_{T_2}(\mathbf{x}) + MHI(\mathbf{x}) \tag{8.2}$$

$$C^G(\mathbf{x}) = 0.3I^G_{T_1}(\mathbf{x}) + 0.7I^G_{T_2}(\mathbf{x}) \tag{8.3}$$

$$C^B(\mathbf{x}) = 0.3I^B_{T_1}(\mathbf{x}) + 0.7I^B_{T_2}(\mathbf{x})$$

where $C^{\{R,G,B\}}$ are the red, green, and blue channels of the composite image, and $I^{\{R,G,B\}}_T$ are the channels of the video image at time T.

8.2.2 Finding the foreground map of a video frame

While this image does convey the nature of the phrase represented, the overlaid MHI obscures the image of the dancer, reducing the aesthetic effect.

To mitigate this problem, an explicit foreground map can be found for frame T_2, and the MHI set to zero in the identified foreground regions. The bounding box of the foreground region in frame T_2 is known as a result of the refined global motion estimation described in chapter 2. It is then necessary to find a frame in which the foreground bounding box does not overlap with the foreground box of frame T_2. The bounding box co-ordinates of frame t are denoted BBt_t, BBb_t, BBl_t, BBr_t for the top, bottom, left, and right edges. These co-ordinates must be transformed to account for any global motion between frames t and T_2, using

$$\left[BBt'_t, BBl'_t, 1\right] = \left[BBt_t, BBl_t, 1\right] \mathbf{M}_{t,T_2} \tag{8.4}$$

$$\left[BBb'_t, BBr'_t, 1\right] = \left[BBb_t, BBr_t, 1\right] \mathbf{M}_{t,T_2} \tag{8.5}$$

where $\mathbf{M}_{t,T_2}$ collects the global motion parameters transforming frame t to frame T_2.

The bounding box of frame t is non-overlapping with that of frame T_2 if

$$\left(BBb'_t > BBt_{T_2}\right) \vee \left(BBt'_t < BBb_{T_2}\right) \vee \left(BBl'_t > BBr_{T_2}\right) \vee \left(BBr'_t < BBl_{T_2}\right) \tag{8.6}$$

Frames $T_2 \pm 1, T_2 \pm 2, \ldots$ are evaluated to find the frame closest to T_2 satisfying 8.6. When such a frame t' is found, the global motion parameters $\mathbf{M}_{t',T_2}$ are used to transform $I_{t'}$ to $I'_{t'}$, aligned with frame T_2. The foreground map F_{T_2} is given by

$$F_{T_2}(\mathbf{x}) = \begin{cases} 1 & \text{if } |I'_{t'}(\mathbf{x}) - I_{T_2}(\mathbf{x})| > \tau \wedge \mathbf{x} \in \mathcal{B}_{T_2} \\ 0 & \text{otherwise} \end{cases} \tag{8.7}$$

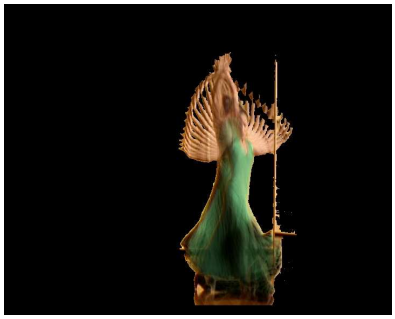
where $\mathcal{B}_{T_2}$ is the set of sites inside the foreground bounding box of frame T_2. To fill in any small holes in the map, an image closing operation is applied to $F(\mathbf{x})$, using a circular structuring element five pixels in diameter.

Figure 8.3 (e) shows a foreground map generated by this method. Some background detail has been detected as foreground here. This is due to the difficulty of long-range motion estimation in this sequence. Figure 8.3 (f) shows how setting the MHI to zero where the foreground map is nonzero improves the appearance of the dance phrase image.

8.2.3 Artificial rear-curtain effect

Using the MHI for dance phrase summary images requires explicit motion detection, and therefore relies on on the determination of some detection threshold . An alternative approach which avoids explicit motion detection is to composite all the frames comprising the dance phrase together, after compensating for global motion. The aim here is to emulate the effect produced by rear-curtain sync flash photography [125]. Here the camera flash is set to fire just before the shutter closes, rather than firing when the shutter opens. This produces a motion blur effect

(a) An example of rear-curtain-sync photography [223].



(b) Composite of foreground regions.

(c) Final summary image.

Figure 8.4: The artifical rear-curtain effect summary image for a dance phrase.

leading into a sharp final position. The effect is commonly used in photography of dance and other moving subjects. An example of a photograph produced using rear-curtain sync is shown in figure 8.4 (a).

To emulate this effect, it is desired to make a composite of all the frames in the dance phrase. Simply averaging the frames will not work, however. This is because a dance phrase may consist of thirty or forty frames, with most of the foreground region non-overlapping over these frames. At a site $\mathbf{x}$ containing foreground data in only one frame t', for example, the value at $\mathbf{x}$ averaged over all frames will be $\left((N-1)\mathbf{B}(\mathbf{x}) + \mathbf{F}_{\mathbf{t}'}(\mathbf{x})\right)/N$, where $N = T_2 - T_1 + 1$ is the length of the dance phrase. For dance phrases of moderate length, the site $\mathbf{x}$ will be effectively background, and the motion blur effect will be lost.

Rather than composite all the frames in the dance phrase, only the foreground regions are composited. The foreground maps for each frame $f \in \{T_1 \ldots T_2\}$ are found as described above. Define the *foreground count image* $C(\mathbf{x})$ by

$$C(\mathbf{x}) = \max\left(1, \sum_{t \in T_1 \ldots T_2} F_t(\mathbf{x})\right) \tag{8.8}$$

The foreground composite image is then

$$\mathbf{J}(\mathbf{x}) = \frac{\left(\sum_{t \in T_1 \ldots T_2} (I_t(\mathbf{x}) F_t(\mathbf{x}))\right)}{C(\mathbf{x})} \tag{8.9}$$

Figure 8.4 (b) shows such a foreground composite image.

To ensure that the final pose of the dance phrase is clearly visible in the final motion phrase image, the foreground region of frame T_2 is superimposed on the foreground composite. Those regions of the frame not included in the foreground composite are filled in with the background

of frame T_2. Thus the final image, $\mathbf{K}(\mathbf{x})$, is given by

$$\mathbf{K}(\mathbf{x}) = \begin{cases} \mathbf{J}(\mathbf{x}) & \text{if } \exists t.F_t(\mathbf{x}) = 1 \wedge F_{T_2}(\mathbf{x}) = 0 \\ \mathbf{I}_{T_2}(\mathbf{x}) & \text{otherwise} \end{cases} \tag{8.10}$$

Figure 8.4 (c) shows an example of the final foreground composite.

Figure 8.5 illustrates twelve successive phrases from the *greenDancer* sequence summarised by this method. While the general nature of the dance is preserved, it is clear that in a number of cases the extraction of the foreground has been less than exact. This is due to the difficulty of long-range global motion estimation in this sequence, and also due to the fact that a single fixed threshold for foreground / background segmentation is not always optimal. More sophisticated methods for addressing these problems have been presented, which could be expected to improve the quality of the resulting images considerably [181, 221]. Incorporation of these methods into the summary generation mechanism is left for future work.

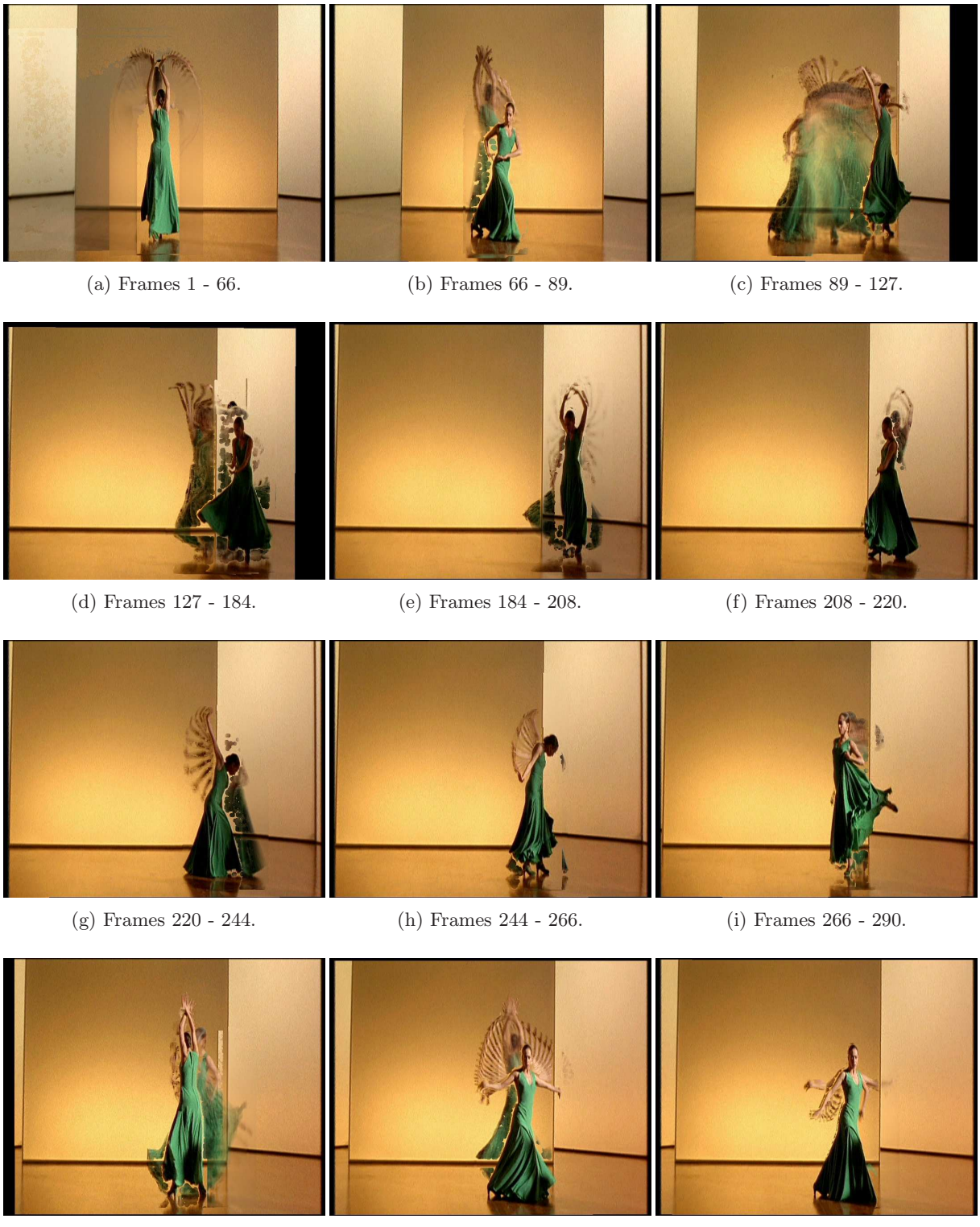
8.3 Frame Synchronisation

The edit point detection techniques described have focused particularly on analysis of dance footage; often this dance is performed to musical accompaniment. Given a new source of music and a means of identifying the *beat times* in this music, the video can be adaptively retimed so as to bring the edit point frames into alignment with these beat times. The effect in the new video is of the same dance being performed to a different tune.

8.3.1 Related Work

Automatic music video creation is an application area beginning to gain ground in the research community at large. However, most of the previously published work is geared towards music-synchronized video editing, in particular to condense large bodies of less-than-compelling home movie footage into a more watchable (and much shorter) music video. One of the earlier publications in this vein describes work done at FX Palo Alto Laboratory [111]. Here, the quality of the user's home video is assessed via an over-exposure metric and camera stability, and temporally segmented into 'clips'—contiguous regions of high quality. Changepoints in the audio track are then localised using a self-similarity measure based on the STFT, and transitions between clips are timed to coincide with these changepoints.

Xian-Sheng Hua *et al.*at Microsoft Research Asia have described similar systems in several publications [152–154]. These are also essentially automated editing systems, in which, for example, the frequency of clip transitions (shot changes) is related to the tempo of the accompanying music. The system described in [153] also uses content analysis to match scenes with high motion content to fast music, and to ensure that similar scenes in the video are matched with similar passages in the music.



(a) Frames 1 - 66. (b) Frames 66 - 89. (c) Frames 89 - 127.

(d) Frames 127 - 184. (e) Frames 184 - 208. (f) Frames 208 - 220.

(g) Frames 220 - 244. (h) Frames 244 - 266. (i) Frames 266 - 290.

(j) Frames 290 - 312. (k) Frames 312 - 343. (l) Frames 343 - 356.

Figure 8.5: Summary images for the first twelve dance phrases in the *greenDancer* sequence.

Similar ideas are beginning to appear in commercially available non-linear editing systems. Muvee Technologies [227] provide a rule-based system to emulate various video editing styles when applied to a collection of home video footage, including a 'music video' style in which transitions and effects are synchronised with the beat; YesVideo [337] offer a similar technology in their analogue to DVD video transfer service. Pinnacle's Studio 9 product [253] incorporates a 'SmartMovie' feature which automatically arranges a selection of clips in sympathy with a soundtrack—again, transitions and effects are automatically synchronised to the beat of the chosen music. As these are commercially released systems, details of the algorithms they employ are not readily available.

While this system targets a similar general application area, the approach to music video generation is quite different. Rather than focusing on editing, the concern here is with synchronising one specific clip to a given piece of music, based on deeper information in the content. One commercially available product similar in spirit is Microsoft's 'Plus! Digital Media Edition' [220], which includes an application called 'Plus! Dancers'. This uses data acquired from live dancers via motion capture techniques to animate a rendered character such that it dances to in time to the music that the user is playing. Again, it is difficult to assess this product quantitatively, as the algorithm employed is not public; however it is of interest as a computer graphics approach to the same aesthetic end result that we seek using video processing techniques.

Other related work includes that of Bregler *et al.* [45] and that of Ezzat *et al.* [102], on synchronising video lip movements to recorded speech. Also of interest is the work of Suzuki *et al.* [290,291] on 'Multimedia Montage', describing various systems for, and theoretical approaches to, video editing for audio-visual synchronisation, using structures informed by music theory.

8.3.2 Beat detection

The first stage in achieving this resynchronisation effect is the detection of the beat times in the new music. Where video is being prepared to accompany a piece under composition, the exact beat times may be available, for example as MIDI data. For existing music, *beat detection* must be applied. A number of methods for beat detection have been described in the literature [83, 123, 124, 270]. For this work, the algorithm first described by Scheirer [269] is used. The audio signal is split into six frequency bands, and each bandpass signal is fed into a bank of 100 tuned resonators. The resonance frequency showing the strongest response over all six frequency bands is then chosen as corresponding to the tempo of the music. This approach was chosen as it is suitable for on-line beat detection in real-time, is readily implemented, and generates good results for most music signals with a steady beat.

8.3.3 Retiming the dance phrase

Given the edit points in the input video, and the beat points in the new piece of music, the idea is to warp the timeline such that visual edit points coincide with the audio beats in the music.

The number of frames of the output video to be generated to the time of the next beat in the new music track, designated F_{BP}, is therefore known. The number of input video frames to the next edit point F_{EP} is also known, having been found using the techniques described above. Figure 8.6 illustrates the situation.

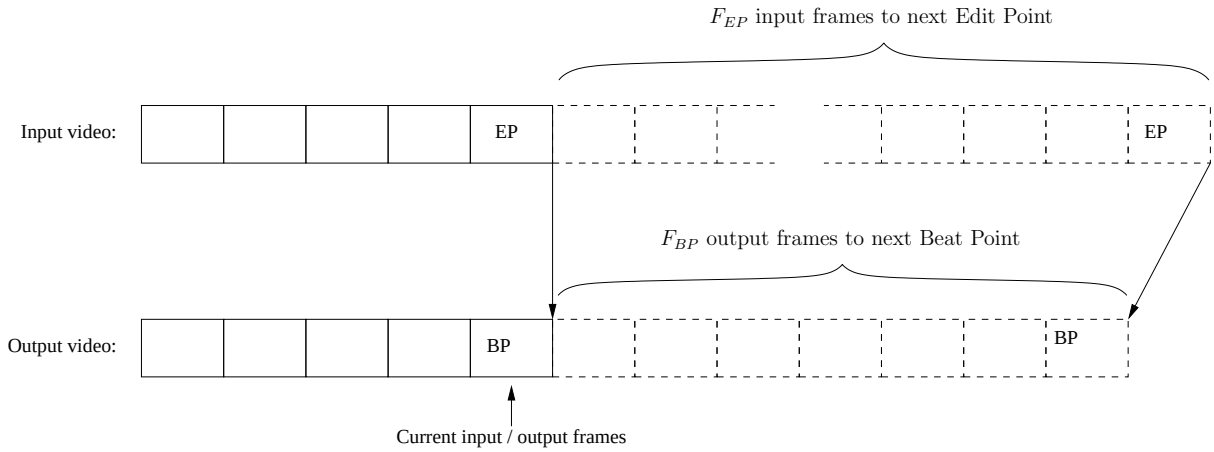


Figure 8.6: Synchronising edit points with beat points

Consider that an edit point E_1 has been emitted to coincide with a beat at time B_1. The next edit point is at time E_2, and the next beat point at time B_2. The frames $E_1 \ldots E_2$ in the input video must be mapped to the frames $B_1 \ldots B_2$ in the output video. One of three strategies can be employed to achieve this. Using linear retiming, output frames $O(B_1 + n)$ for $n \in \{1 \ldots B_2 - B_1\}$ are copied from input frames $I(n')$ according to

$$O(B_1 + n) = I\left(E_1 + n\frac{E_2 - E_1}{B_2 - B1}\right) \tag{8.11}$$

This scheme is illustrated in figure 8.7 (a).

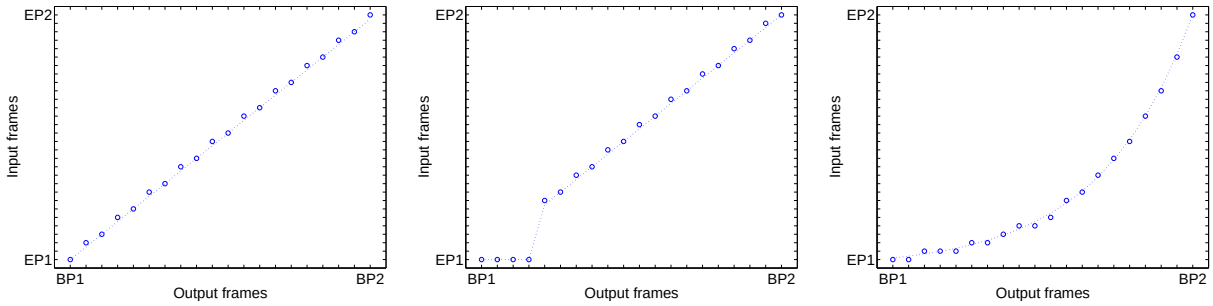
The perceived strength of a synchronised event can be increased by repeating the edit point frame for a small number, τ_e, of output frames, and resuming input processing τ_e frames after the edit point. This accentuation effect is referred to as *stop and skip*, and is governed by

$$O(B_1 + n) = \begin{cases} I(E_1) & \text{if } n < \tau_e \\ I(E_1 + n\frac{E_2 - E_1}{B_2 - B1}) & \text{otherwise} \end{cases} \tag{8.12}$$

Figure 8.7 (b) shows the mapping from input frames to output frames with this approach, where $\tau_e = 4$.

Rather than using linear retiming, an exponential scheme can be employed. Here input frames are mapped to output frames by

$$O(B_1 + n) = I\left(E_1 + \exp\left(n \log\left(\frac{E_2 - E_1}{B_2 - B1}\right)\right)\right) \tag{8.13}$$



(a) Linear retiming.

(b) Linear retiming with a pause to emphasise the edit point (*stop and skip*).

(c) Exponential retiming.

Figure 8.7: Retiming strategies for dance phrase video. The dashed line shows the ideal retiming. The blue circles show the frames actually emitted.

as shown in figure 8.7 (c). This introduces a very strong acceleration towards the edit point, and a sudden deceleration afterward, which emphasises the edit point very strongly and introduces a markedly artificial feel to the motion of the dance.

Regardless of the retiming approach used, some threshold on the maximum permitted degree of video compression and dilation is required. Retiming is only applied if $\frac{1}{3} \leq \frac{E_2-E_1}{B_2-B1} \leq 3$. If $\frac{E_2-E_1}{B_2-B1} < \frac{1}{3}$, the next edit point is skipped. When $\frac{E_2-E_1}{B_2-B1} > 3$, input frames are output sequentially until $\frac{E_2-E_1}{B_2-B1} < 3$.

No temporal interpolation is used when retiming the video; wherever an input frame at a non-integer offset n is required, $I(\lfloor n + 0.5 \rfloor)$ is used. A smoother effect could be achieved by using a more sophisticated retiming algorithm.

8.3.4 Final points

The length of the generated video can be increased by introducing a looping effect. When looping is activated, frames are taken from the input video in reverse order, synchronising to the previous edit point rather than the next. After an edit point frame has been emitted synchronised with a beat frame, looping may be activated with some probability P_l. A value of $P_l = 0.4$ is found to work well in most sequences, in the sense that the length of the dance is increased but the effect of the repetitions introduced is not tedious.

A number of examples of videos generated using this fully automated application are provided on the accompanying DVD. They show that despite the numerically moderate performance of the edit point detection process, the resulting effect is convincing. The technique has been implemented in a real-time application in which video is synchronised to a live incoming music stream. This application has proven popular for generating visuals synchronised to a DJ in nightclubs.

8.4 Conclusion

In this chapter, a number of applications of edit point detection in dance video have been described. In each case the implementation described is effectively at the proof-of-concept level. However, the promise of these techniques for high-level interaction with dance video is clear. The dance re-synchronisation technique in particular has potential in music video creation, in personal media players, and in the real-time projection of video to accompany live performance.

9 Conclusion

The central thrust of this work has been to show how a number of diverse video processing problems are conceptually concerned with discontinuity detection. In all cases, the essential approach is to obtain some targeted description of the video, and then to detect discontinuities in this description. The application-specific aspect is the description of the video, which determines the generality and semantic level of the events encompassed.

The first application of discontinuity analysis described is shot change detection. Here the description of the video is in terms of frame-to-frame similarity. Discontinuities in the dissimilarity signal then inform shot change detection. The bulk of chapter 3 was concerned with the varieties of frame similarity measure—or ways of describing frame similarity—that have been deployed for this problem. It was described how increasingly researchers are focusing on the detection of discontinuities in the similarity description, as being of equal importance to the similarity measure itself. The Bayesian paradigm has been particularly fruitful in this regard. Chapter 4 then described refinements for cut detection in difficult sequences and dissolve detection.

The work on analysis of snooker video has a similar process of description and discontinuity detection. In particular, the event detection algorithm based on region monitoring has sympathies with shot change detection, being based on similarity analysis within a particular region of the frame. Explicit extraction of ball tracks is also an instance of targeted video description, and it was outlined how the discontinuities in these tracks are informative regarding events in the snooker game.

The automatic detection of edit points in dance video is an entirely new application area in

digital video processing. The treatment presented here encompasses the fusion of a number of frame-level descriptions of motion, along with an investigation of video segmentation for edit point detection. A particular success of the low level approach is that the same framework for discontinuity detection, based on Bayesian classification of peak characteristics, was found effective for traces generated by a variety of analysis techniques. The examples of resynchronised dance video on the accompanying DVD demonstrate that compelling effects can be realised through exploitation of these edit points.

Motion based segmentation of dance sequences was considered as a diagnostic tool for making the percussive movement detector more robust. This work differs from previous approaches to segmentation in that ensuring that the extracted segments are smooth and contiguous over a long period of time is a particular concern here. What makes this difficult is that objects are modelled in a low level, piecewise manner, rather than using any explicit modelling, for example describing the human body. Several ideas for encouraging this temporal consistency have been introduced here. The idea of using magnitude and angle to parameterise motion clusters is also new to this work. It has been shown that with this parameterisation, segments containing pathological motion can be isolated, which would otherwise be confused with stationary background.

9.1 Issues

Perfect shot change detection remains an elusive goal, and the area continues to attract research attention. The problem of effects used in shot changes in particular remains a challenge. Further investigation of the dissolve modelling approach described here is warranted—in particular, a more complete determination of the statistics of the α-curves described there, through automated synthesis of a large number of dissolves, would be of use. Another area worthy of investigation is whether the spatial evolution of the error map arising in this algorithm could be analysed to inform dissolve detection.

The work in chapter 5 presents techniques that constitute a set of primitives for snooker analysis. As described in that chapter, other researchers have demonstrated how those tools can be generalised for analysis of other sports, and integrated into systems for sports analysis and retrieval. One aspect not yet addressed is an evaluation of how the 'ball pot' and 'near miss' event detection system, based on region monitoring, could be applied to sports with similar events such as basketball and golf.

Chapter 6 takes a very broad view of low-level edit point detection, and each of the frame analysis primitives employed there could be explored in more depth. More sophisticated approaches to motion detection, global and local motion estimation, and motion blur detection, would all improve edit point detection in this framework. One possible weak point of the framework itself is the 'hard' nature of the initial parsing of each trace into peaks; robustness could be perhaps be improved by using a softer, probabilistic approach to peak extraction and the detection of minima and maxima in the traces. Analysis of a larger corpus of varied material would

be informative, in particular in assessing how the edit point detection framework generalises to non-dance footage.

The work on video segmentation described in chapter 7 establishes that maintaining temporal coherence in dance footage segmentation is a difficult undertaking, due chiefly to the large displacements and non-translational motion typical of dance video. A key requirement here is to avoid or recover from over-segmentation to improve the processing speed; this suggests that further investigation into the incorporation of model merging and motion histories with each model is warranted. A multi-resolution approach could also be useful in this regard.

It is interesting that the segmentation output did not markedly improve the 'percussive visual beat' system. This is simply because of the difficulty of extracting semantic meaning without the application of some model connecting the low level features to the semantics. It would be useful therefore to consider connecting the various segments within a model of the human form and thus inferring further information about the motion explicitly as human dance, for example.

The various applications shown in chapter 8 remain at the prototype stage. Of particular interest here is the nature of the most appropriate user interface for video resynchronisation. The exploitation of higher level music analysis (i.e. rhythm analysis as opposed to tempo analysis) is also worthy of investigation.

9.2 Final remarks

All of the work conducted in this thesis was performed using real data. As such it highlights that motion analysis must move toward acknowledging the real phenomena that cause algorithms to fail, such as motion blur. This work has shown that using relatively low level tools the presence of blur can be connected to some semantics concerned with movement. More generally, this approach of explicitly acknowledging the hard problems confounding motion analysis, such as pathological motion, transparent motion, and shot changes involving visual effects, provides a rich body of future work.

A

Derivation of the optimal alpha value for dissolve modelling

In chapter 4, a method for characterising a dissolve region in video using an α-curve was described. The α-value for frame t given two template frames I_{T_0}, I_{T_1} is governed by

$$p(\alpha|I_t) \propto \exp(-\sum_{\mathbf{x}} \left[I_t(\mathbf{x}) - I_{M(\alpha)}(\mathbf{x})\right]^2) \tag{A.1}$$

The maximum-likelihood value of alpha given I_t can be estimated by solving

$$\frac{d}{d\alpha} p(\alpha|I_t) = 0 \tag{A.2}$$

This is equivalent to solving

$$\frac{d}{d\alpha} \sum_{\mathbf{x}} \left[I_t(\mathbf{x}) - I_{M(\alpha)}(\mathbf{x})\right]^2 = 0 \tag{A.3}$$

Differentiating yields

$$2 \sum_{\mathbf{x}} \left[\left(I_t(\mathbf{x}) - I_{M(\alpha)}(\mathbf{x})\right)\left(-I_{T_0}(\mathbf{x}) + I_{T_1}(\mathbf{x})\right)\right] = 0 \tag{A.4}$$

Let $\nabla_{T_0,T_1}(\mathbf{x}) = I_{T_0}(\mathbf{x}) - I_{T_1}(\mathbf{x})$. Then

$$\sum_{\mathbf{x}} \left[-\nabla_{T_0,T_1}(\mathbf{x}) I_t(\mathbf{x}) + \nabla_{T_0,T_1}(\mathbf{x}) \alpha I_{T_0}(\mathbf{x}) + \nabla_{T_0,T_1}(\mathbf{x})(1-\alpha) I_{T_1}(\mathbf{x})\right] = 0 \tag{A.5}$$

$$\sum_{\mathbf{x}} \left[-\nabla_{T_0,T_1}(\mathbf{x}) + \nabla_{T_0,T_1}(\mathbf{x}) \alpha I_{T_0}(\mathbf{x}) + \nabla_{T_0,T_1}(\mathbf{x}) I_{T_1}(\mathbf{x}) - \alpha \nabla_{T_0,T_1}(\mathbf{x}) I_{T_1}(\mathbf{x})\right] = 0 \tag{A.6}$$

$$\sum_{\mathbf{x}} \left[\alpha \nabla_{T_0,T_1}(\mathbf{x})(I_{T_0}(\mathbf{x}) - I_{T_1}(\mathbf{x})) - \nabla_{T_0,T_1}(\mathbf{x}) I_{T_1}(\mathbf{x}) + \nabla_{T_0,T_1}(\mathbf{x}) I_{T_1}(\mathbf{x})\right] = 0 \tag{A.7}$$

Rearranging:

$$\alpha \sum_{\mathbf{x}} [\nabla_{T_0,T_1}(\mathbf{x})(I_{T_0}(\mathbf{x}) - I_{T_1}(\mathbf{x}))] = \sum_{\mathbf{x}} [\nabla_{T_0,T_1}(\mathbf{x})I_{T_1}(\mathbf{x}) - \nabla_{T_0,T_1}(\mathbf{x})I_{T_1}(\mathbf{x})] \tag{A.8}$$

which yields a closed form solution for α:

$$\alpha = \frac{\sum_{\mathbf{x}} [\nabla_{T_0,T_1}(\mathbf{x})I_t(\mathbf{x}) - \nabla_{T_0,T_1}(\mathbf{x})I_{T_1}(\mathbf{x})]}{\sum_{\mathbf{x}} \left[\nabla^2_{T_0,T_1}(\mathbf{x})\right]} \tag{A.9}$$

B

Sequence Sources

The *greenDancer*, *maleDancer*, *maleDancer2*, and *greenMale* sequences come from the DVD 'Flamenco', directed by Carlos Saura, released by New Yorker Video in 1994.

The *ballet2* and *ballet1* sequences are from the DVD release of 'The Sleeping Beauty', performed by the Birmingham Royal Ballet, released in 2003.

The *coconut* sequence is from the DVD 'Monty Python and The Holy Grail (Special Edition)', released by Sony Pictures in 2001.

C Results

C.1 The Foreground Bounding Box

C.1.1 Minima Detection

	#H	#FA	#M	P	R	M_{PR}	MD_{median}	MD_{mean}	MD_{var}
Sequence Training									
No smoothing									
M	242	527	59	0.31	0.80	0.56	1	0.55	5.62
M_s	180	213	121	0.46	0.60	0.53	0	0.26	8.33
M_r	243	520	58	0.32	0.81	0.56	0	0.44	5.93
M_{rs}	179	204	122	0.47	0.59	0.53	0	-0.02	8.79
With smoothing									
$\hat{M}$	202	291	99	0.41	0.67	0.54	0	-0.03	7.17
$\hat{M}_s$	167	180	134	0.48	0.55	0.52	0	-0.33	8.74
$\hat{M}_r$	201	275	100	0.42	0.67	0.55	0	0.04	8.19
$\hat{M}_{rs}$	161	168	140	0.49	0.53	0.51	0	-0.07	9.55

Table C.1: Left

	#H	#FA	#M	P	R	M_{PR}	MD_{median}	MD_{mean}	MD_{var}
Sequence Training									
No smoothing									
M	249	596	52	0.29	0.83	0.56	1	0.42	4.61
M_s	186	238	115	0.44	0.62	0.53	1	0.53	6.89
M_r	245	597	56	0.29	0.81	0.55	1	0.40	4.49
M_{rs}	195	238	106	0.45	0.65	0.55	0	0.44	6.67
With smoothing									
$\hat{M}$	205	286	96	0.42	0.68	0.55	1	0.56	6.25
$\hat{M}_s$	176	164	125	0.52	0.58	0.55	1	0.36	7.74
$\hat{M}_r$	210	302	91	0.41	0.70	0.55	1	0.55	6.10
$\hat{M}_{rs}$	178	167	123	0.52	0.59	0.55	1	0.61	6.93

Table C.2: Right

	#H	#FA	#M	P	R	M_{PR}	MD_{median}	MD_{mean}	MD_{var}
Sequence Training									
No smoothing									
M	264	675	37	0.28	0.88	0.58	0	0.29	4.38
M_s	197	246	104	0.44	0.65	0.55	0	0.02	7.52
M_r	262	638	39	0.29	0.87	0.58	0	0.29	4.75
M_{rs}	181	219	120	0.45	0.60	0.53	0	-0.02	8.03
With smoothing									
$\hat{M}$	261	479	40	0.35	0.87	0.61	0	-0.22	5.84
$\hat{M}_s$	216	239	85	0.47	0.72	0.60	-1	-0.63	6.73
$\hat{M}_r$	251	447	50	0.36	0.83	0.60	0	-0.10	6.68
$\hat{M}_{rs}$	206	214	95	0.49	0.68	0.59	-1	-0.26	8.64

Table C.3: Bottom

	#H	#FA	#M	P	R	M_{PR}	MD_{median}	MD_{mean}	MD_{var}
Sequence Training									
No smoothing									
M	252	589	49	0.30	0.84	0.57	0	0.06	3.84
M_s	197	212	104	0.48	0.65	0.57	0	-0.09	5.26
M_r	248	563	53	0.31	0.82	0.56	0	0.01	3.72
M_{rs}	192	193	109	0.50	0.64	0.57	0	-0.17	5.03
With smoothing									
$\hat{M}$	227	263	74	0.46	0.75	0.61	0	0.42	5.87
$\hat{M}_s$	190	141	111	0.57	0.63	0.60	0	0.27	6.48
$\hat{M}_r$	231	256	70	0.47	0.77	0.62	0	0.30	5.83
$\hat{M}_{rs}$	195	139	106	0.58	0.65	0.62	0	0.16	6.58

Table C.4: Top

	#H	#FA	#M	P	R	M_{PR}	MD_{median}	MD_{mean}	MD_{var}
Sequence Training									
No smoothing									
M	265	647	36	0.29	0.88	0.59	1	0.52	4.10
M_s	214	255	87	0.46	0.71	0.58	1	0.64	6.64
M_r	265	637	36	0.29	0.88	0.59	0	0.46	4.02
M_{rs}	214	255	87	0.46	0.71	0.58	1	0.56	6.29
With smoothing									
$\hat{M}$	221	318	80	0.41	0.73	0.57	1	0.57	6.81
$\hat{M}_s$	180	195	121	0.48	0.60	0.54	1	0.65	7.33
$\hat{M}_r$	235	309	66	0.43	0.78	0.61	1	0.52	6.38
$\hat{M}_{rs}$	192	179	109	0.52	0.64	0.58	1	0.86	7.10

Table C.5: Width

	#H	#FA	#M	P	R	M_{PR}	MD_{median}	MD_{mean}	MD_{var}
Sequence Training									
No smoothing									
M	287	798	14	0.26	0.95	0.61	1	0.45	2.61
M_s	235	297	66	0.44	0.78	0.61	0	0.29	5.47
M_r	287	732	14	0.28	0.95	0.62	0	0.33	2.60
M_{rs}	234	262	67	0.47	0.78	0.62	0	0.12	5.29
With smoothing									
$\hat{M}$	257	367	44	0.41	0.85	0.63	1	0.49	4.95
$\hat{M}_s$	215	198	86	0.52	0.71	0.62	1	0.47	6.48
$\hat{M}_r$	267	371	34	0.42	0.89	0.65	1	0.39	4.74
$\hat{M}_{rs}$	228	192	73	0.54	0.76	0.65	0	0.14	6.03

Table C.6: Height

C.1.2 Peak Classification

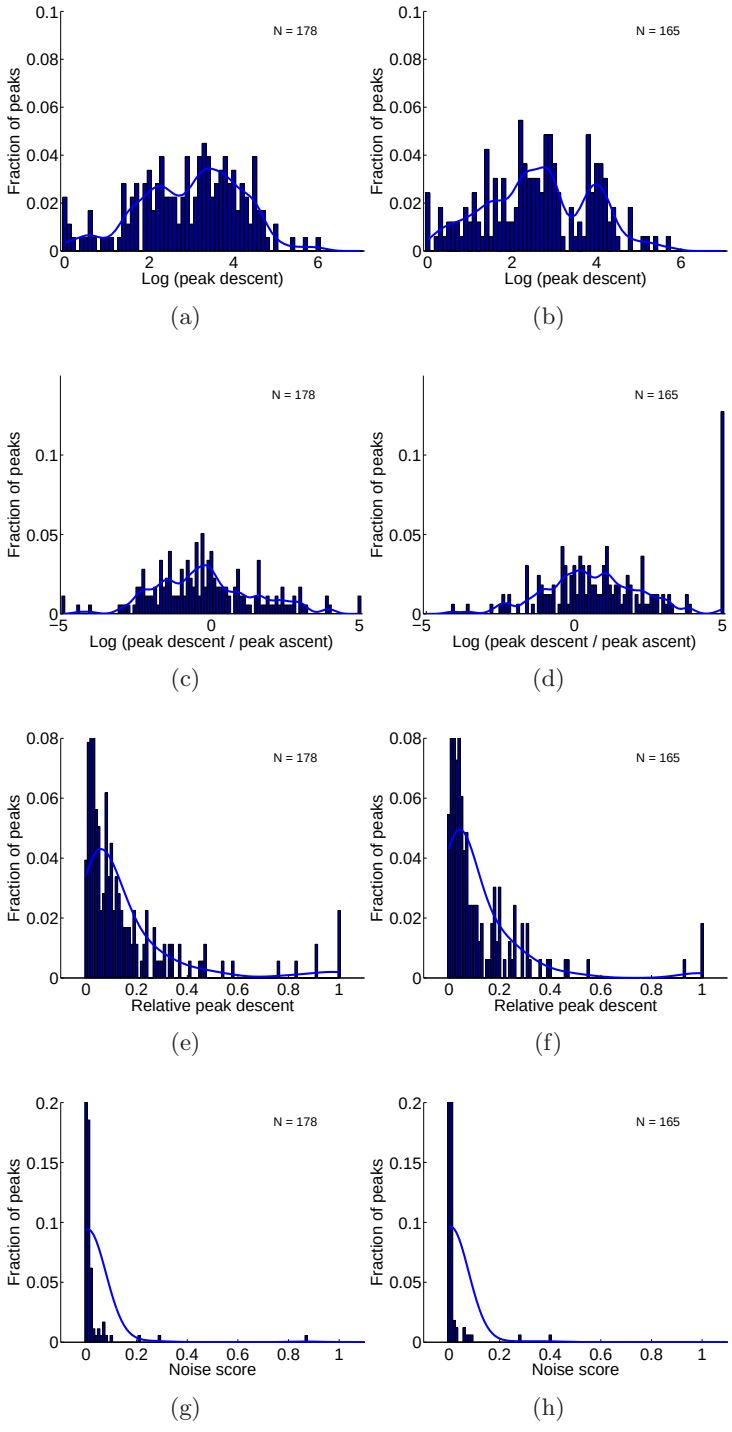


Figure C.1: Peak statistics for the bb_{min_x} trace, from the *greenDancer*, *maleDancer*, and *ballet2* sequences combined. Desired peaks, corresponding to motion events, are shown on the left, while the figures on the right illustrate false alarms. The Parzen estimations of the densities are also shown.

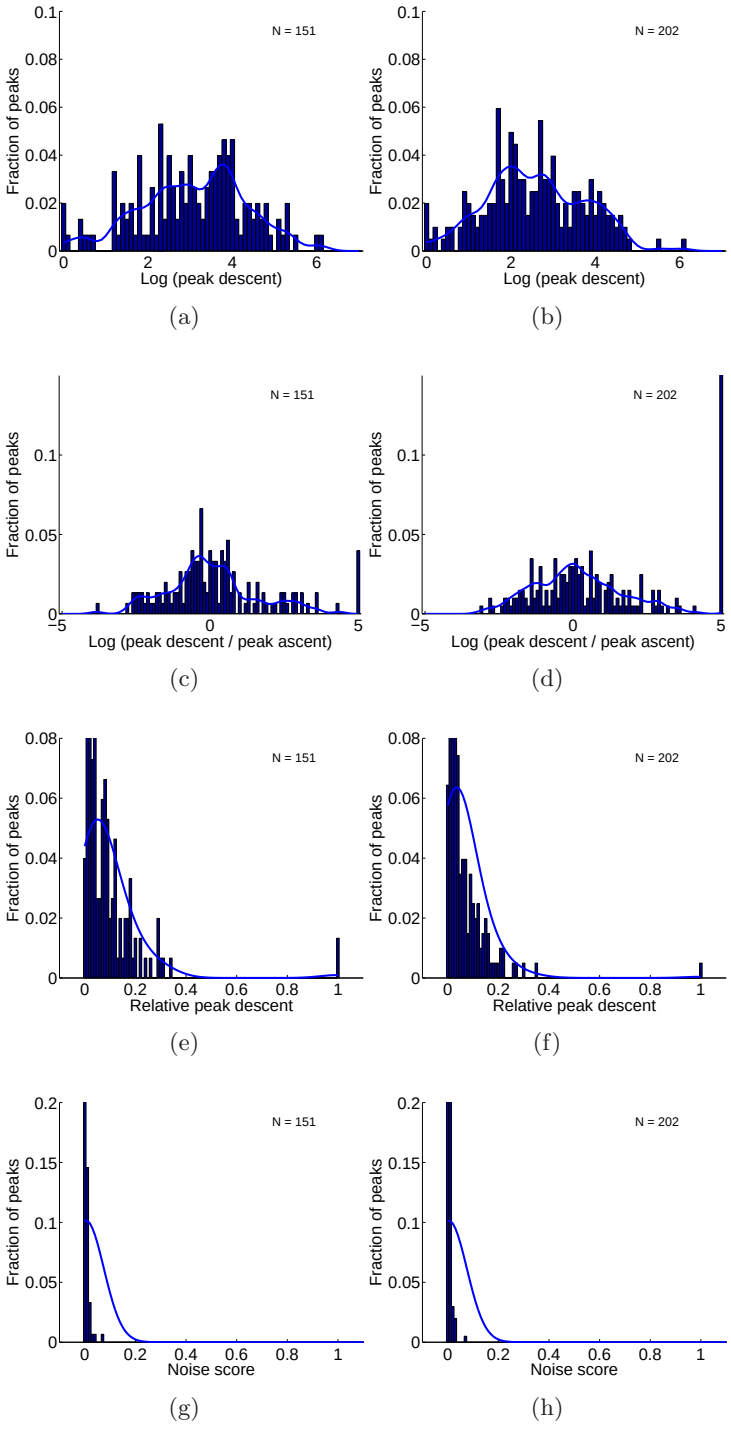


Figure C.2: Peak statistics for the bb_{max_x} trace, from the *greenDancer*, *maleDancer*, and *ballet2* sequences combined. Desired peaks, corresponding to motion events, are shown on the left, while the figures on the right illustrate false alarms. The Parzen estimations of the densities are also shown.

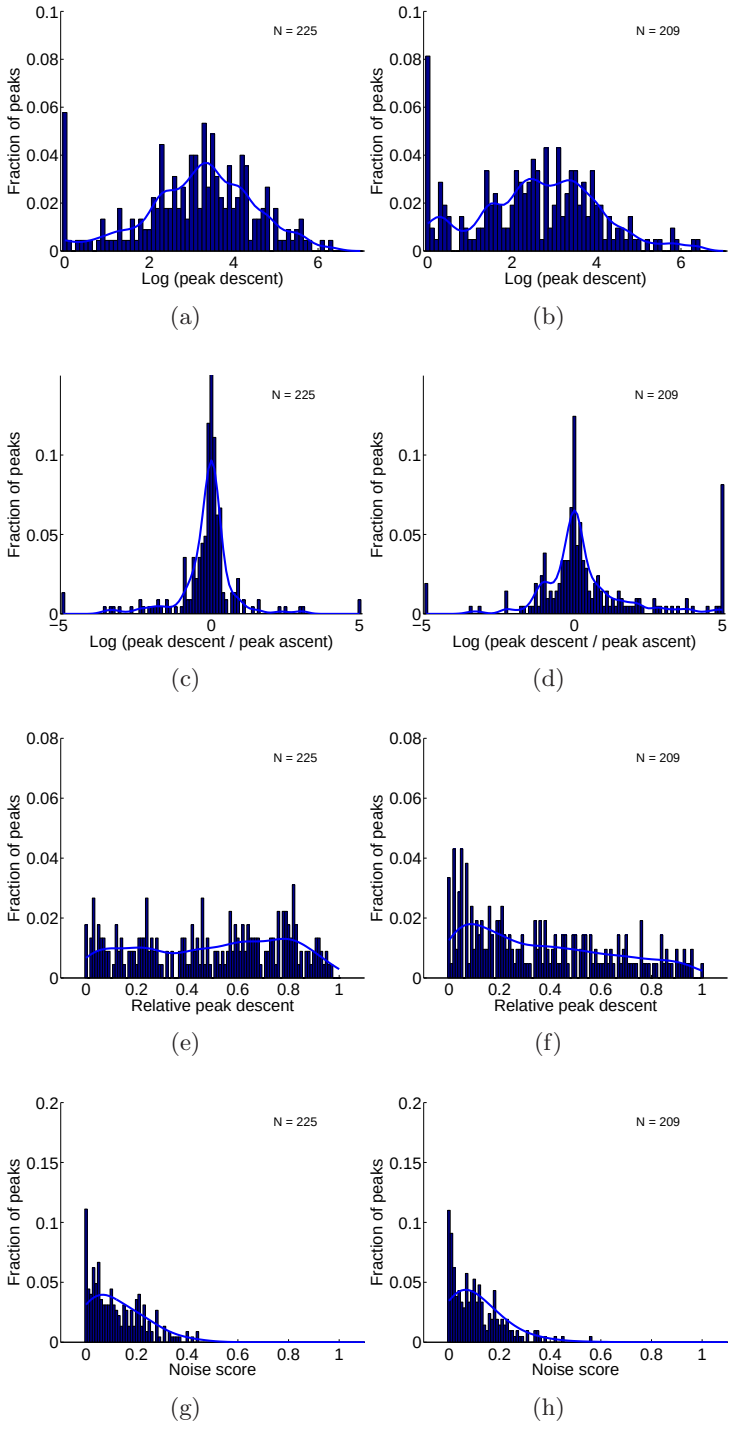


Figure C.3: Peak statistics for the bb_{min_y} trace, from the *greenDancer*, *maleDancer*, and *ballet2* sequences combined. Desired peaks, corresponding to motion events, are shown on the left, while the figures on the right illustrate false alarms. The Parzen estimations of the densities are also shown.

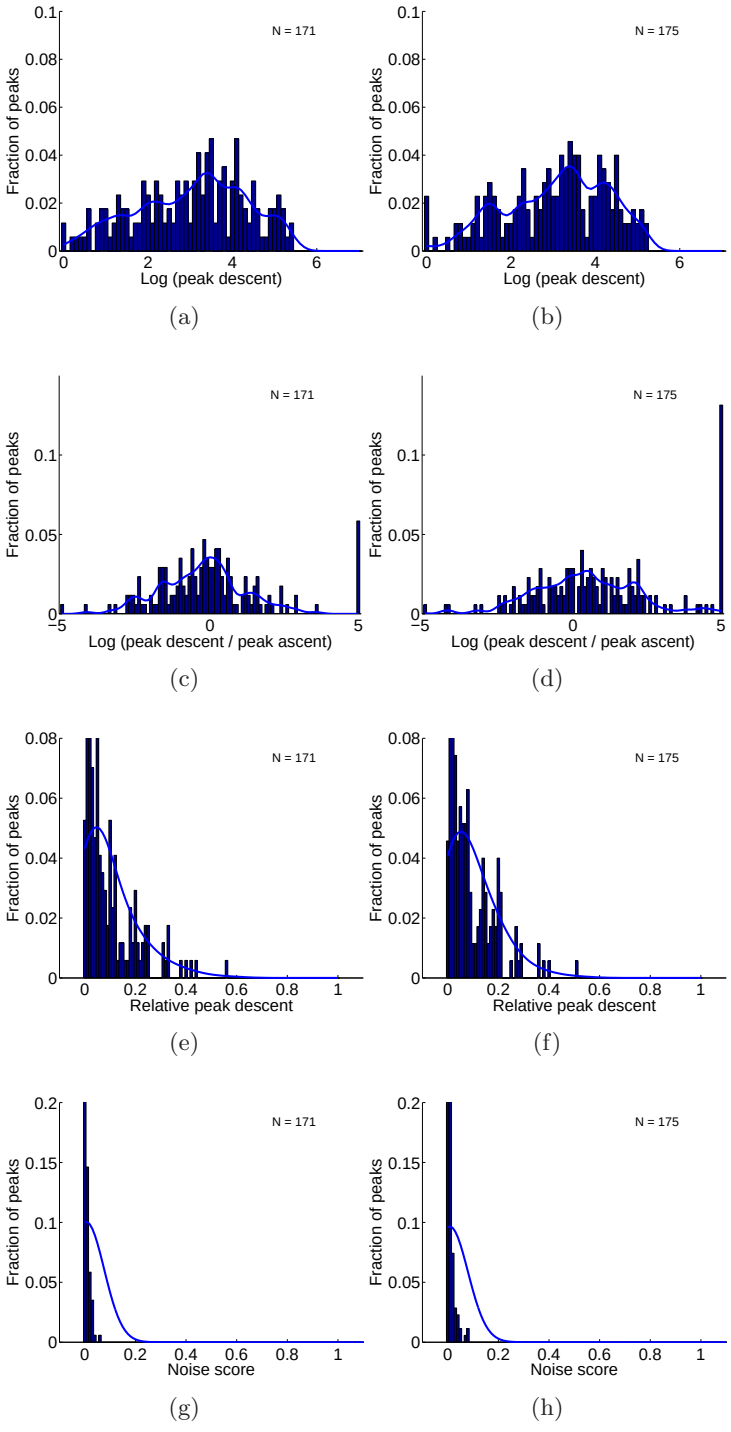


Figure C.4: Peak statistics for the bb_{max_y} trace, from the *greenDancer*, *maleDancer*, and *ballet2* sequences combined. Desired peaks, corresponding to motion events, are shown on the left, while the figures on the right illustrate false alarms. The Parzen estimations of the densities are also shown.

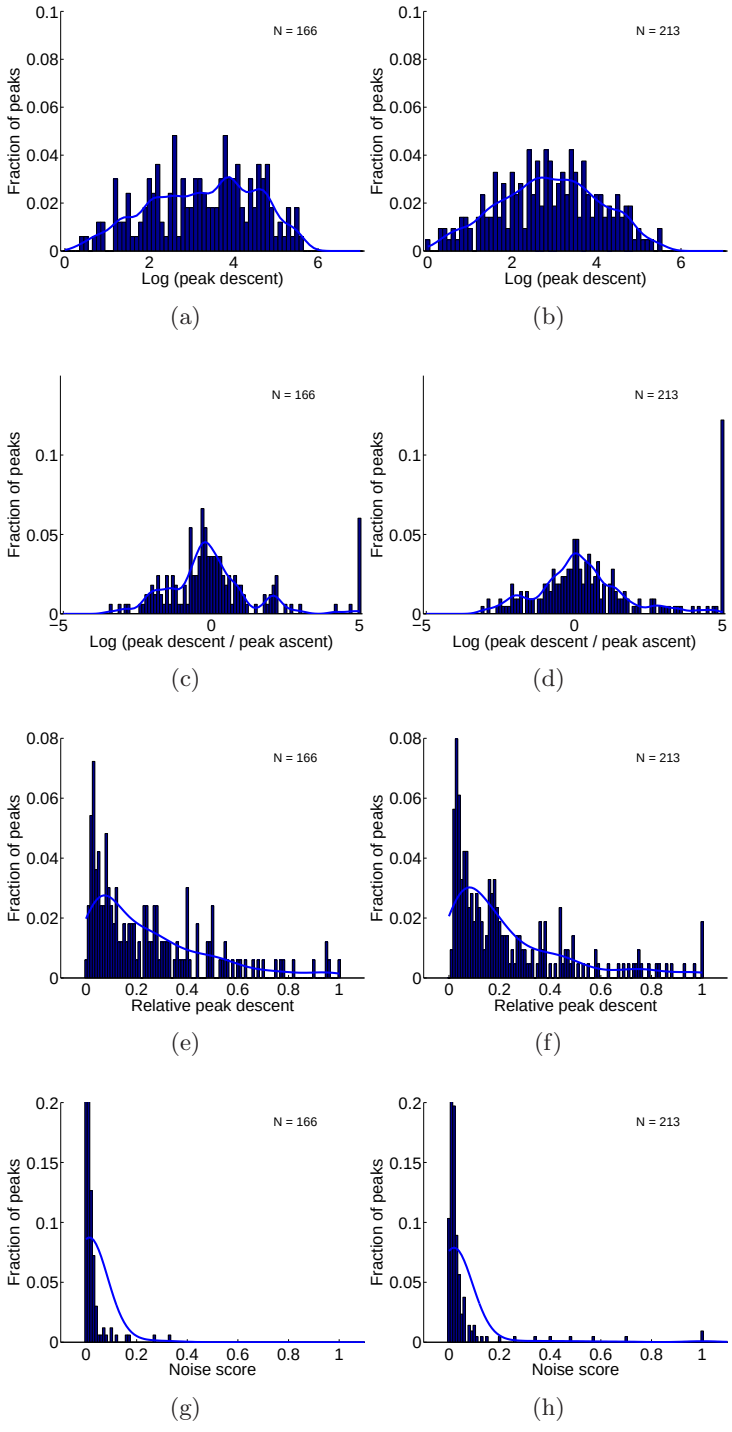


Figure C.5: Peak statistics for the bb_{width} trace, from the *greenDancer*, *maleDancer*, and *ballet2* sequences combined. Desired peaks, corresponding to motion events, are shown on the left, while the figures on the right illustrate false alarms. The Parzen estimations of the densities are also shown.

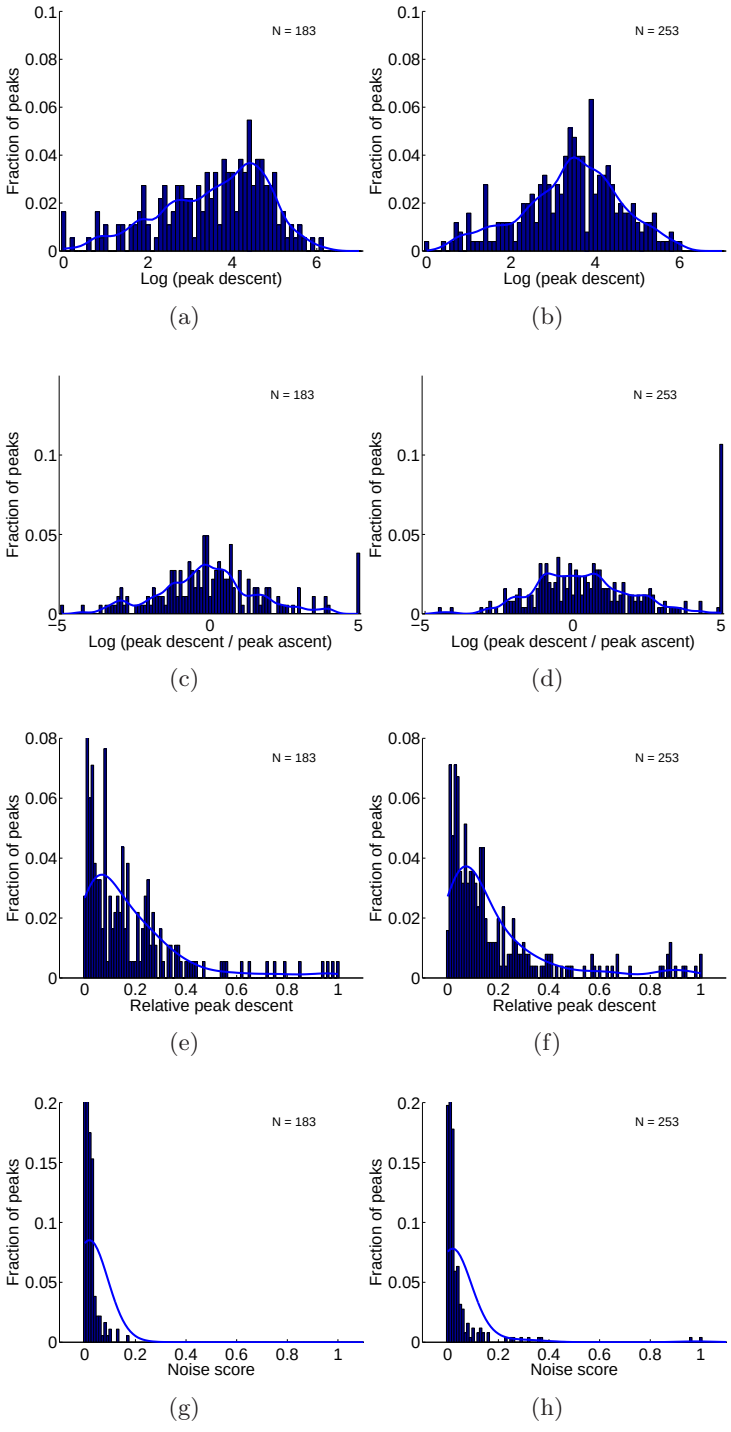


Figure C.6: Peak statistics for the bb_{height} trace, from the *greenDancer*, *maleDancer*, and *ballet2* sequences combined. Desired peaks, corresponding to motion events, are shown on the left, while the figures on the right illustrate false alarms. The Parzen estimations of the densities are also shown.

C.2 Vector Field

C.2.1 Minima Detection

	#H	#FA	#M	P	R	M_{PR}	MD_{median}	MD_{mean}	MD_{var}
Sequence Training									
No smoothing									
M	297	1121	4	0.21	0.99	0.60	1	0.43	1.87
M_s	240	278	61	0.46	0.80	0.63	1	0.70	4.99
M_r	298	1068	3	0.22	0.99	0.60	0	0.27	2.05
M_{rs}	256	309	45	0.45	0.85	0.65	0	0.25	4.75
With smoothing									
$\hat{M}$	274	661	27	0.29	0.91	0.60	1	0.64	3.19
$\hat{M}_s$	247	238	54	0.51	0.82	0.66	1	0.62	4.68
$\hat{M}_r$	274	636	27	0.30	0.91	0.61	0	0.47	3.35
$\hat{M}_{rs}$	246	200	55	0.55	0.82	0.68	1	0.38	4.59

Table C.7: vf_M

	#H	#FA	#M	P	R	M_{PR}	MD_{median}	MD_{mean}	MD_{var}
Sequence Training									
No smoothing									
M	293	983	8	0.23	0.97	0.60	1	0.44	2.34
M_s	260	383	41	0.40	0.86	0.63	1	0.56	4.12
M_r	294	910	7	0.24	0.98	0.61	0	0.12	2.26
M_{rs}	263	305	38	0.46	0.87	0.67	0	0.24	4.07
With smoothing									
$\hat{M}$	271	455	30	0.37	0.90	0.64	1	0.70	3.40
$\hat{M}_s$	244	240	57	0.50	0.81	0.66	1	0.50	4.60
$\hat{M}_r$	275	422	26	0.39	0.91	0.65	1	0.49	3.93
$\hat{M}_{rs}$	245	216	56	0.53	0.81	0.67	1	0.40	4.97

Table C.8: $vf_{\sigma^2_M}$

	#H	#FA	#M	P	R	M_{PR}	MD_{median}	MD_{mean}	MD_{var}
Sequence Training									
No smoothing									
M	294	1261	7	0.19	0.98	0.58	0	0.33	2.30
M_s	258	539	43	0.32	0.86	0.59	0	0.28	5.84
M_r	293	1174	8	0.20	0.97	0.59	0	0.27	2.57
M_{rs}	257	455	44	0.36	0.85	0.61	0	0.11	5.77
With smoothing									
$\hat{M}$	270	628	31	0.30	0.90	0.60	0.5	0.30	4.96
$\hat{M}_s$	246	341	55	0.42	0.82	0.62	1	0.43	6.35
$\hat{M}_r$	266	602	35	0.31	0.88	0.60	0	0.33	4.58
$\hat{M}_{rs}$	246	305	55	0.45	0.82	0.63	0.5	0.38	6.06

Table C.9: vf_x

	#H	#FA	#M	P	R	M_{PR}	MD_{median}	MD_{mean}	MD_{var}
Sequence Training									
No smoothing									
M	297	1072	4	0.22	0.99	0.60	0	0.44	2.31
M_s	255	382	46	0.40	0.85	0.62	1	0.56	4.70
M_r	298	1009	3	0.23	0.99	0.61	0	0.24	2.01
M_{rs}	252	314	49	0.45	0.84	0.64	0	0.03	4.38
With smoothing									
$\hat{M}$	273	537	28	0.34	0.91	0.62	1	0.68	3.69
$\hat{M}_s$	239	257	62	0.48	0.79	0.64	1	0.65	4.40
$\hat{M}_r$	280	514	21	0.35	0.93	0.64	1	0.32	3.85
$\hat{M}_{rs}$	245	225	56	0.52	0.81	0.67	1	0.48	4.79

Table C.10: $vf_{\sigma_x^2}$

	#H	#FA	#M	P	R	M_{PR}	MD_{median}	MD_{mean}	MD_{var}
Sequence Training									
No smoothing									
M	299	1195	2	0.20	0.99	0.60	0	0.46	1.99
M_s	277	515	24	0.35	0.92	0.64	0	0.28	4.72
M_r	300	1112	1	0.21	1.00	0.60	0	0.30	1.96
M_{rs}	270	430	31	0.39	0.90	0.64	0	0.23	4.91
With smoothing									
$\hat{M}$	286	578	15	0.33	0.95	0.64	1	0.50	4.33
$\hat{M}_s$	265	321	36	0.45	0.88	0.67	1	0.48	5.33
$\hat{M}_r$	283	561	18	0.34	0.94	0.64	0	0.18	4.05
$\hat{M}_{rs}$	261	288	40	0.48	0.87	0.67	0	0.22	5.79

Table C.11: vf_y

	#H	#FA	#M	P	R	M_{PR}	MD_{median}	MD_{mean}	MD_{var}
Sequence Training									
No smoothing									
M	296	1069	5	0.22	0.98	0.60	1	0.35	1.99
M_s	268	398	33	0.40	0.89	0.65	1	0.30	4.71
M_r	298	992	3	0.23	0.99	0.61	0	0.07	2.15
M_{rs}	260	340	41	0.43	0.86	0.65	0	-0.03	4.28
With smoothing									
$\hat{M}$	275	516	26	0.35	0.91	0.63	1	0.64	3.09
$\hat{M}_s$	246	246	55	0.50	0.82	0.66	1	0.71	4.41
$\hat{M}_r$	278	478	23	0.37	0.92	0.65	0	0.27	3.56
$\hat{M}_{rs}$	257	227	44	0.53	0.85	0.69	0	0.32	4.60

Table C.12: $vf_{\sigma_y^2}$

	#H	#FA	#M	P	R	M_{PR}	MD_{median}	MD_{mean}	MD_{var}
Sequence Training									
No smoothing									
M	301	1456	0	0.17	1.00	0.59	0	0.26	1.58
M_s	280	637	21	0.31	0.93	0.62	0	0.11	5.15
M_r	301	1364	0	0.18	1.00	0.59	0	0.28	1.94
M_{rs}	273	543	28	0.33	0.91	0.62	0	0.28	5.87
With smoothing									
$\hat{M}$	290	762	11	0.28	0.96	0.62	0	0.04	4.54
$\hat{M}_s$	261	383	40	0.41	0.87	0.64	0	-0.20	6.51
$\hat{M}_r$	294	722	7	0.29	0.98	0.63	0	0.19	4.97
$\hat{M}_{rs}$	246	352	55	0.41	0.82	0.61	0	-0.00	7.18

Table C.13: vf_θ

C.2.2 Peak Classification Distributions

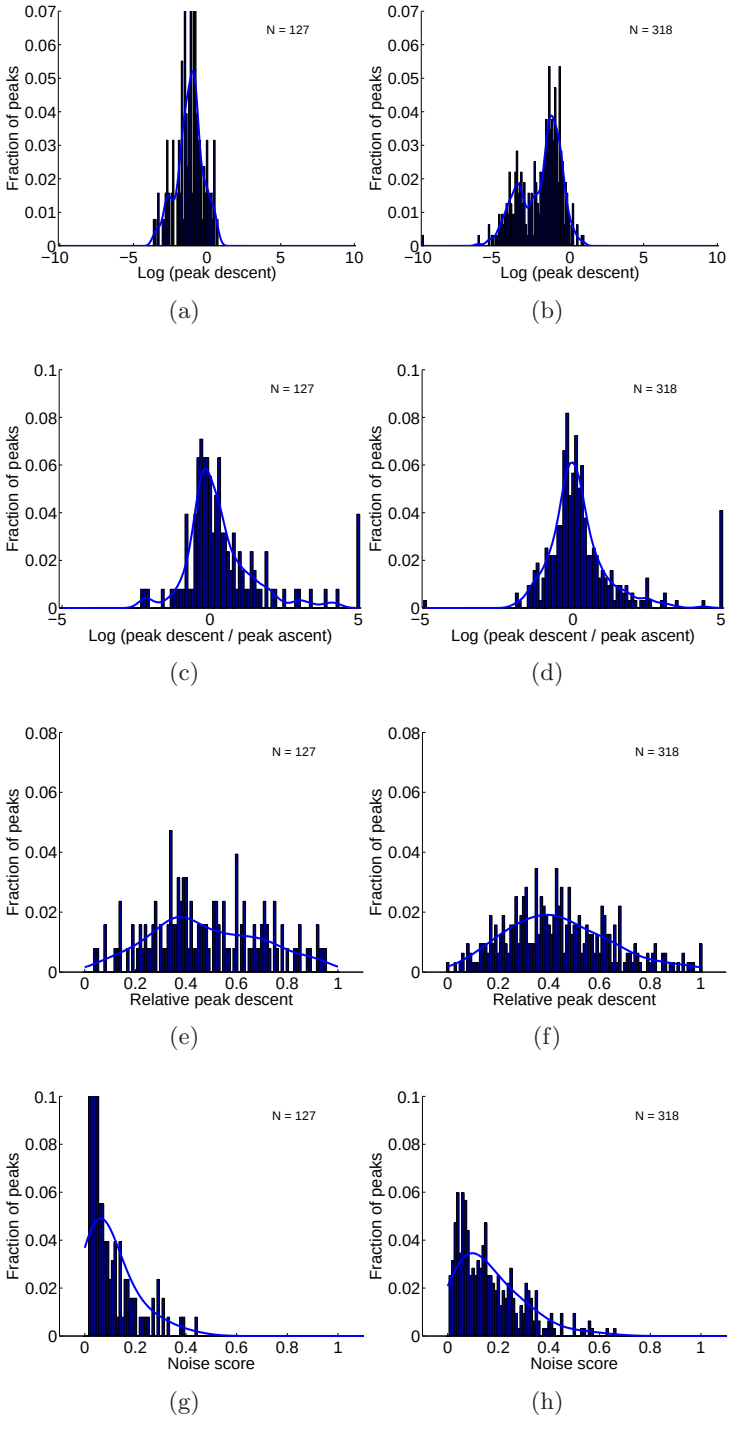


Figure C.7: Peak statistics for the vf_M trace, from the *greenDancer*, *maleDancer*, and *ballet2* sequences combined. Desired peaks, corresponding to motion events, are shown on the left, while the figures on the right illustrate false alarms. The Parzen estimations of the densities are also shown.

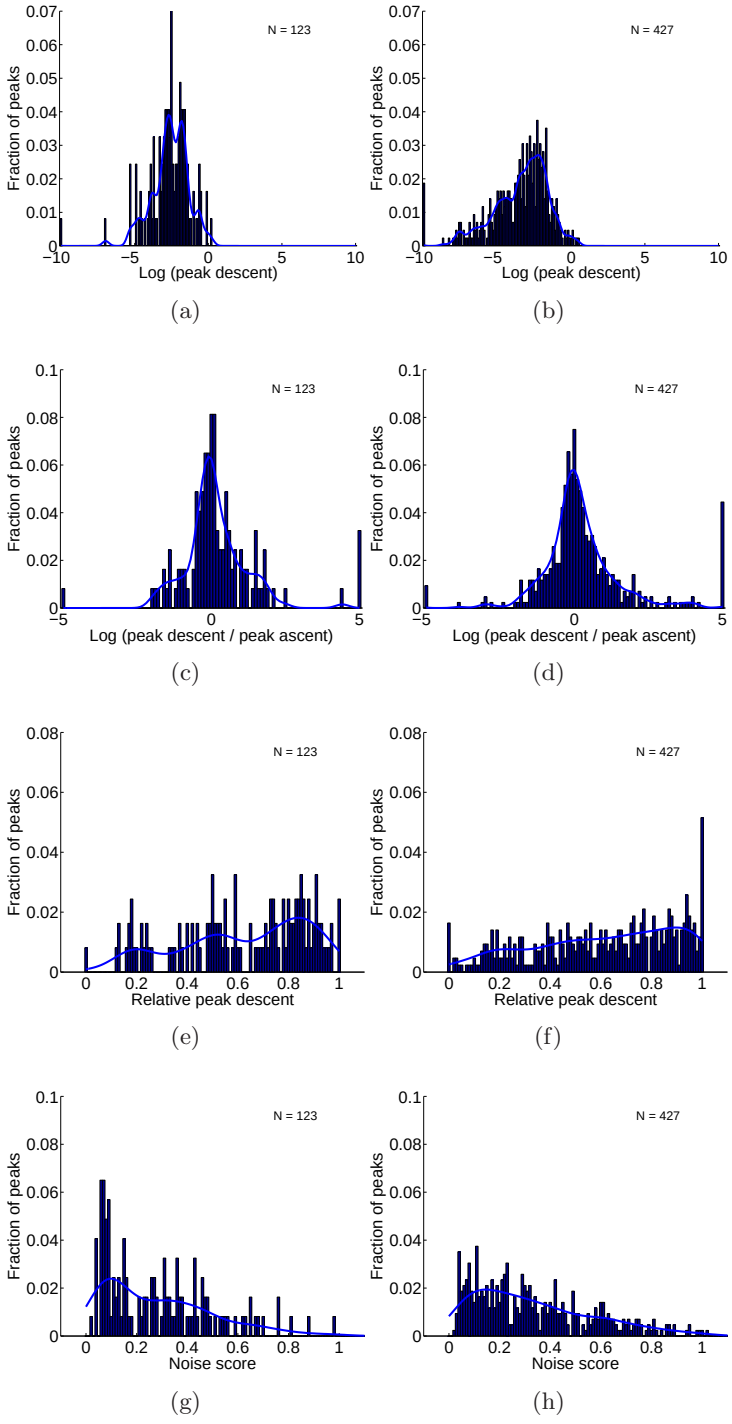


Figure C.8: Peak statistics for the vf_x trace, from the *greenDancer*, *maleDancer*, and *ballet2* sequences combined. Desired peaks, corresponding to motion events, are shown on the left, while the figures on the right illustrate false alarms. The Parzen estimations of the densities are also shown.

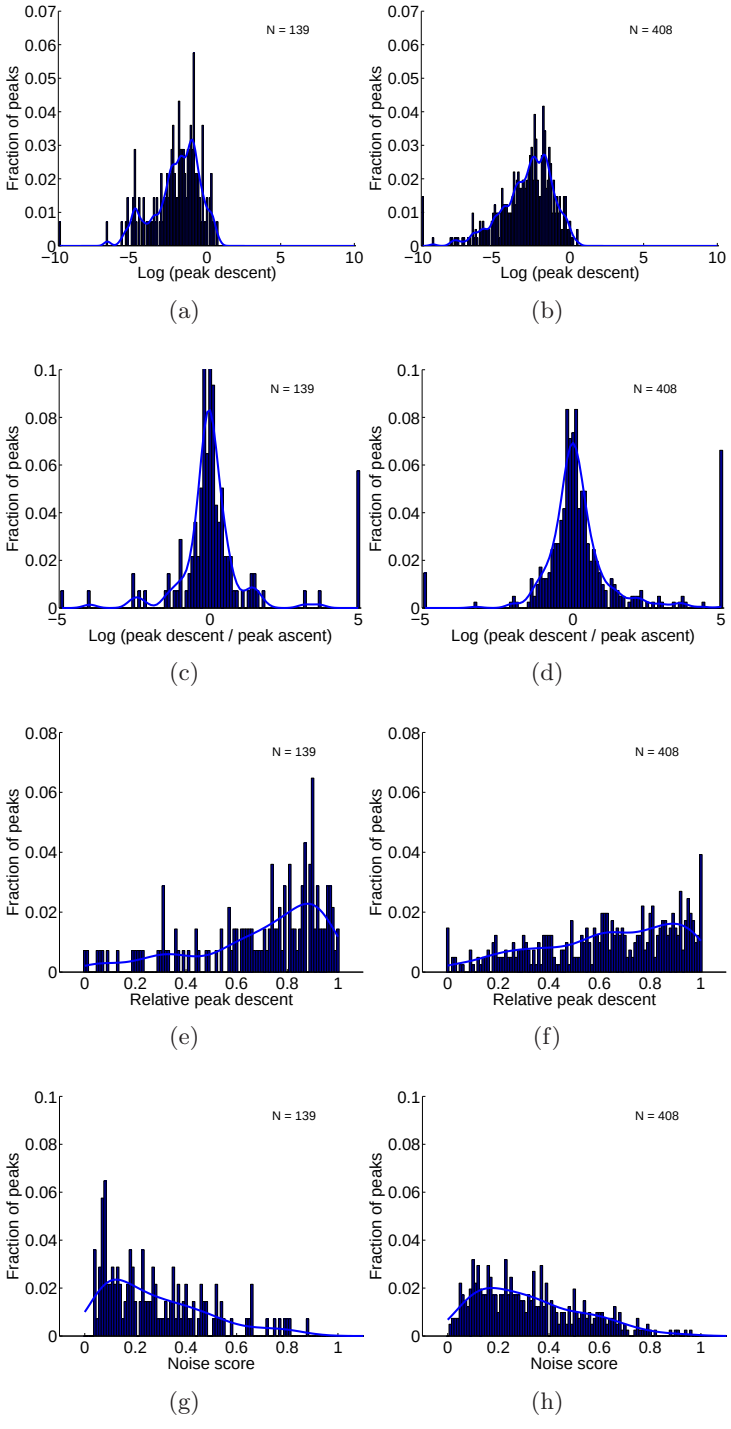


Figure C.9: Peak statistics for the vf_y trace, from the *greenDancer*, *maleDancer*, and *ballet2* sequences combined. Desired peaks, corresponding to motion events, are shown on the left, while the figures on the right illustrate false alarms. The Parzen estimations of the densities are also shown.

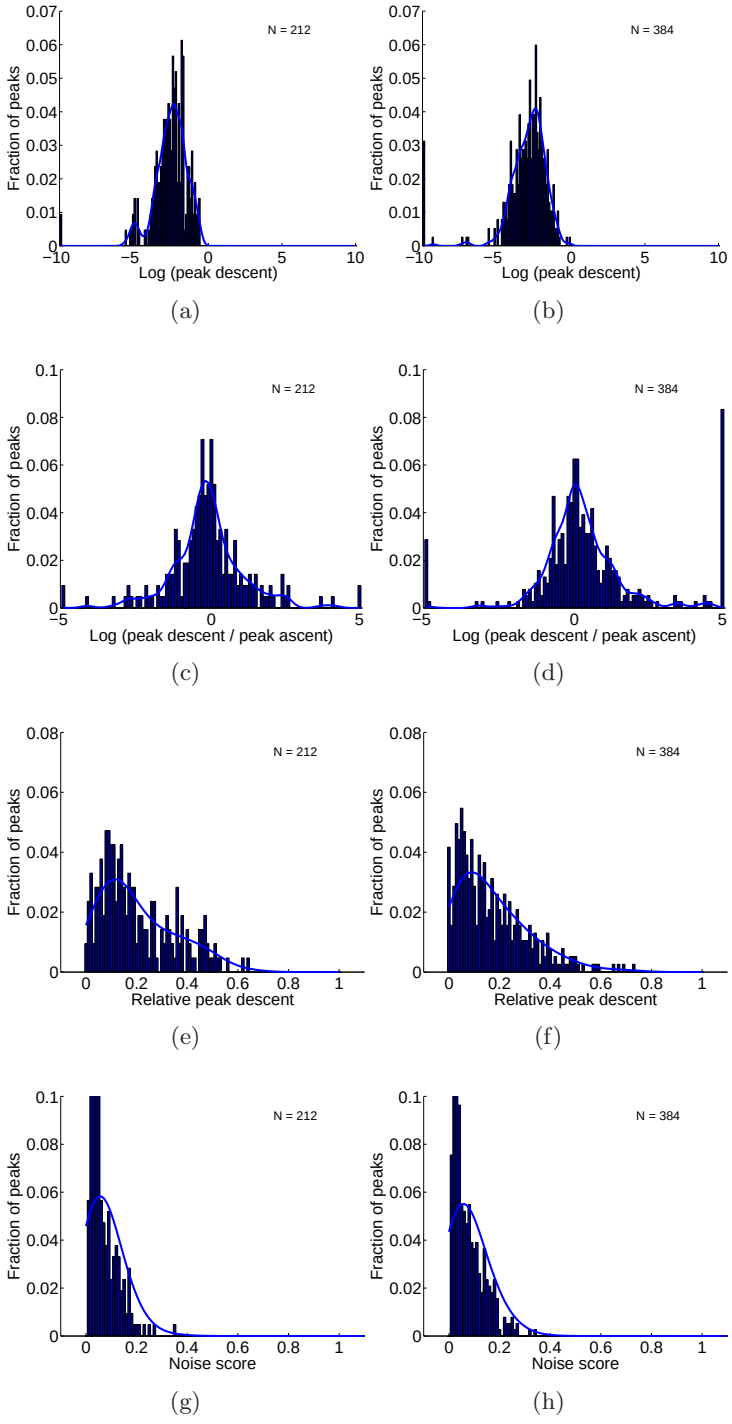


Figure C.10: Peak statistics for the vf_θ trace, from the *greenDancer*, *maleDancer*, and *ballet2* sequences combined. Desired peaks, corresponding to motion events, are shown on the left, while the figures on the right illustrate false alarms. The Parzen estimations of the densities are also shown.

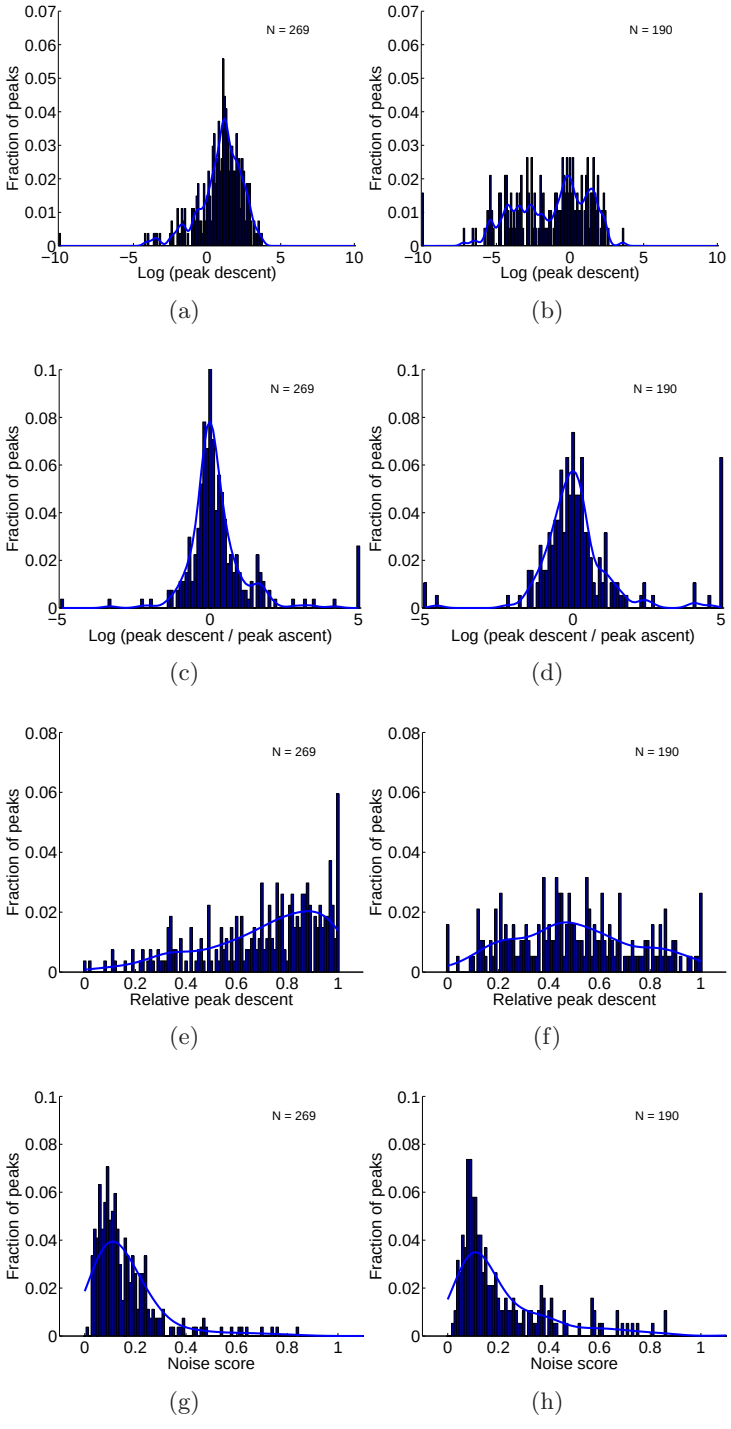


Figure C.11: Peak statistics for the $vf_{\sigma_M^2}$ trace, from the *greenDancer*, *maleDancer*, and *ballet2* sequences combined. Desired peaks, corresponding to motion events, are shown on the left, while the figures on the right illustrate false alarms. The Parzen estimations of the densities are also shown.

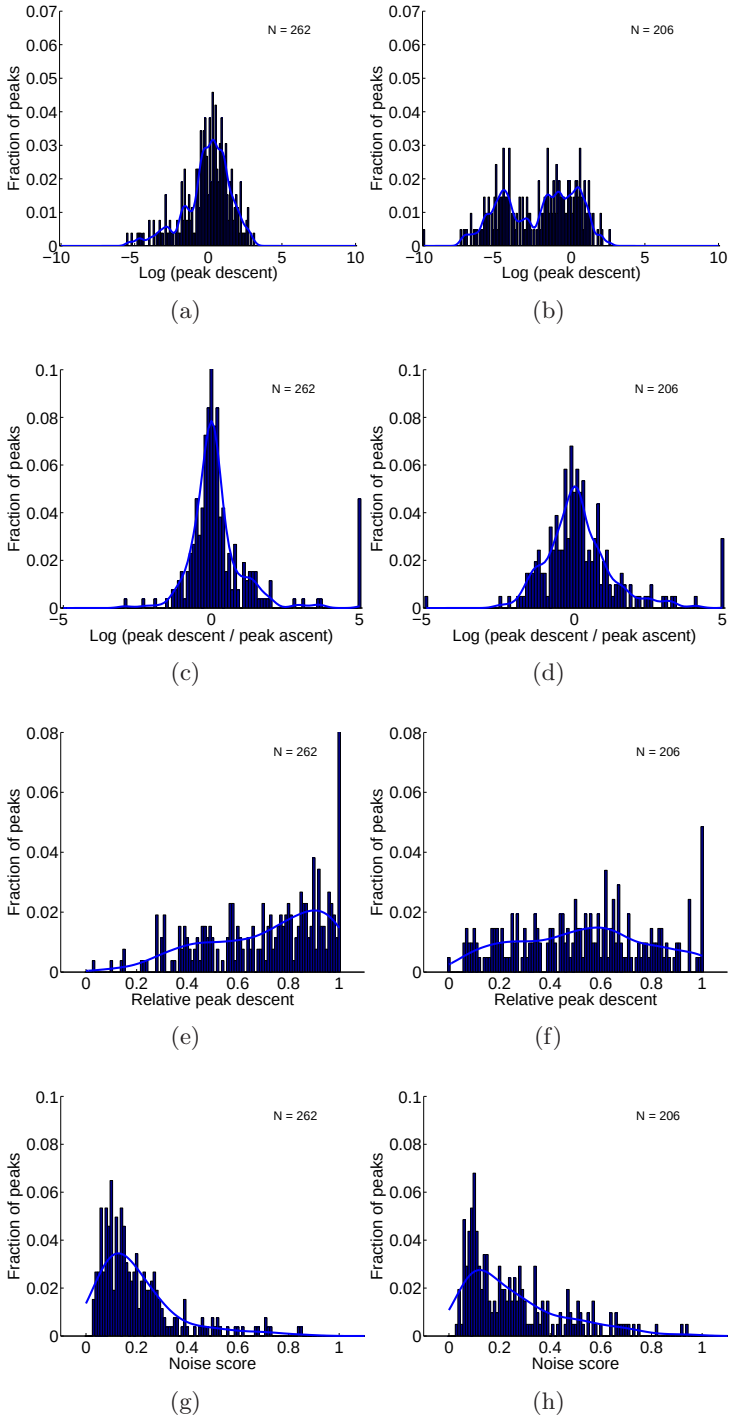


Figure C.12: Peak statistics for $vf_{\sigma_x^2}$ trace, from the *greenDancer*, *maleDancer*, and *ballet2* sequences combined. Desired peaks, corresponding to motion events, are shown on the left, while the figures on the right illustrate false alarms. The Parzen estimations of the densities are also shown.

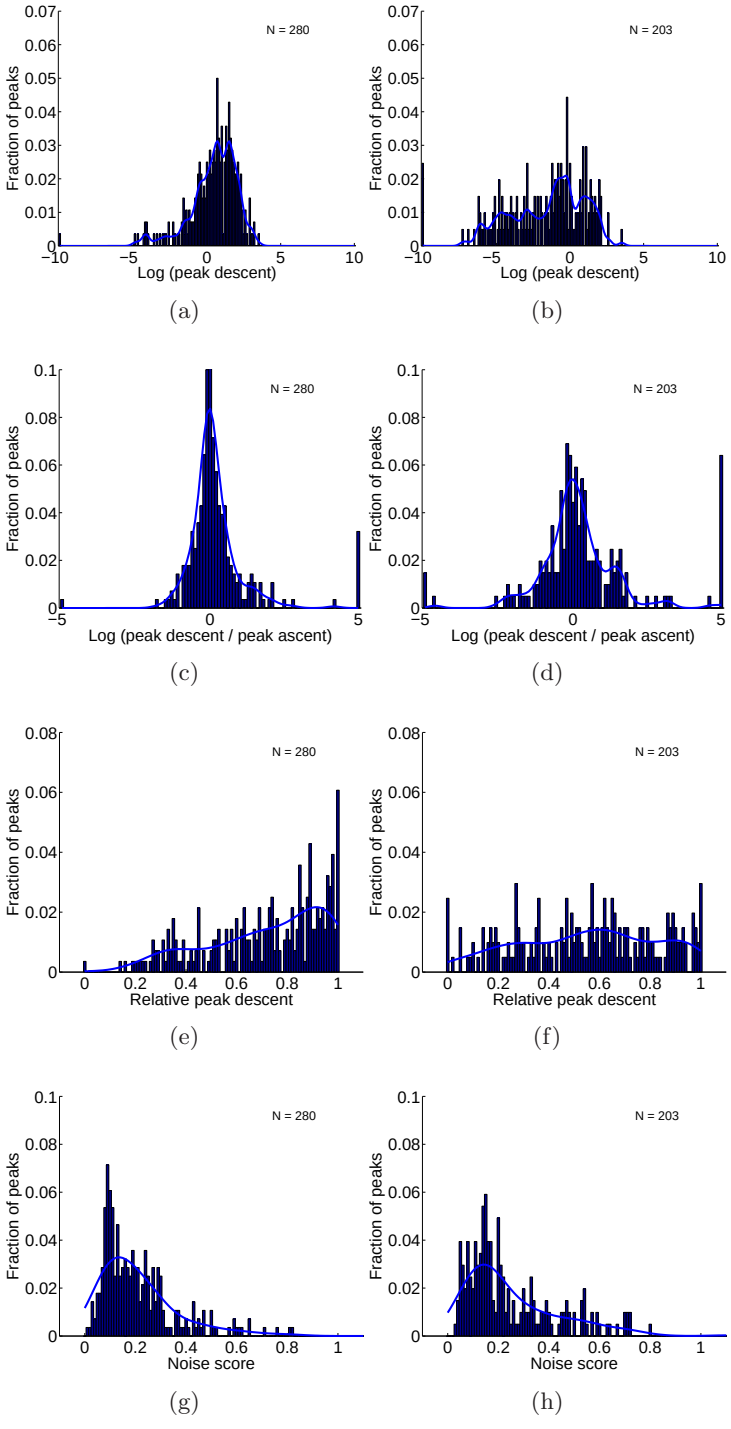


Figure C.13: Peak statistics for the $vf_{\sigma_y^2}$ trace, from the *greenDancer*, *maleDancer*, and *ballet2* sequences combined. Desired peaks, corresponding to motion events, are shown on the left, while the figures on the right illustrate false alarms. The Parzen estimations of the densities are also shown.

C.3 Sharpness

C.3.1 Minima Detection

	#H	#FA	#M	P	R	M_{PR}	MD_{median}	MD_{mean}	MD_{var}
Sequence Training									
No smoothing									
M	168	540	14	0.24	0.92	0.58	0	0.24	5.38
M_s	112	240	70	0.32	0.62	0.47	0	-0.05	8.21
M_r	159	542	23	0.23	0.87	0.55	0	-0.06	5.21
M_{rs}	112	247	70	0.31	0.62	0.46	0	0.29	8.12
With smoothing									
$\hat{M}$	128	304	54	0.30	0.70	0.50	0	-0.25	7.01
$\hat{M}_s$	117	224	65	0.34	0.64	0.49	0	-0.42	7.78
$\hat{M}_r$	128	319	54	0.29	0.70	0.49	0	-0.23	6.81
$\hat{M}_{rs}$	103	228	79	0.31	0.57	0.44	0	0.07	7.22

Table C.14: SumFFT

	#H	#FA	#M	P	R	M_{PR}	MD_{median}	MD_{mean}	MD_{var}
Sequence Training									
No smoothing									
M	106	451	13	0.19	0.89	0.54	0	0.41	1.56
M_s	102	280	17	0.27	0.86	0.56	1	0.31	4.02
M_r	106	408	13	0.21	0.89	0.55	0	0.48	2.21
M_{rs}	99	228	20	0.30	0.83	0.57	0	0.33	5.41
With smoothing									
$\hat{M}$	93	139	26	0.40	0.78	0.59	1	0.20	6.49
$\hat{M}_s$	73	78	46	0.48	0.61	0.55	0	-0.33	7.64
$\hat{M}_r$	91	155	28	0.37	0.76	0.57	1	0.34	6.36
$\hat{M}_{rs}$	78	81	41	0.49	0.66	0.57	0	-0.04	8.06

Table C.15: Interlace

C.3.2 Peak Classification

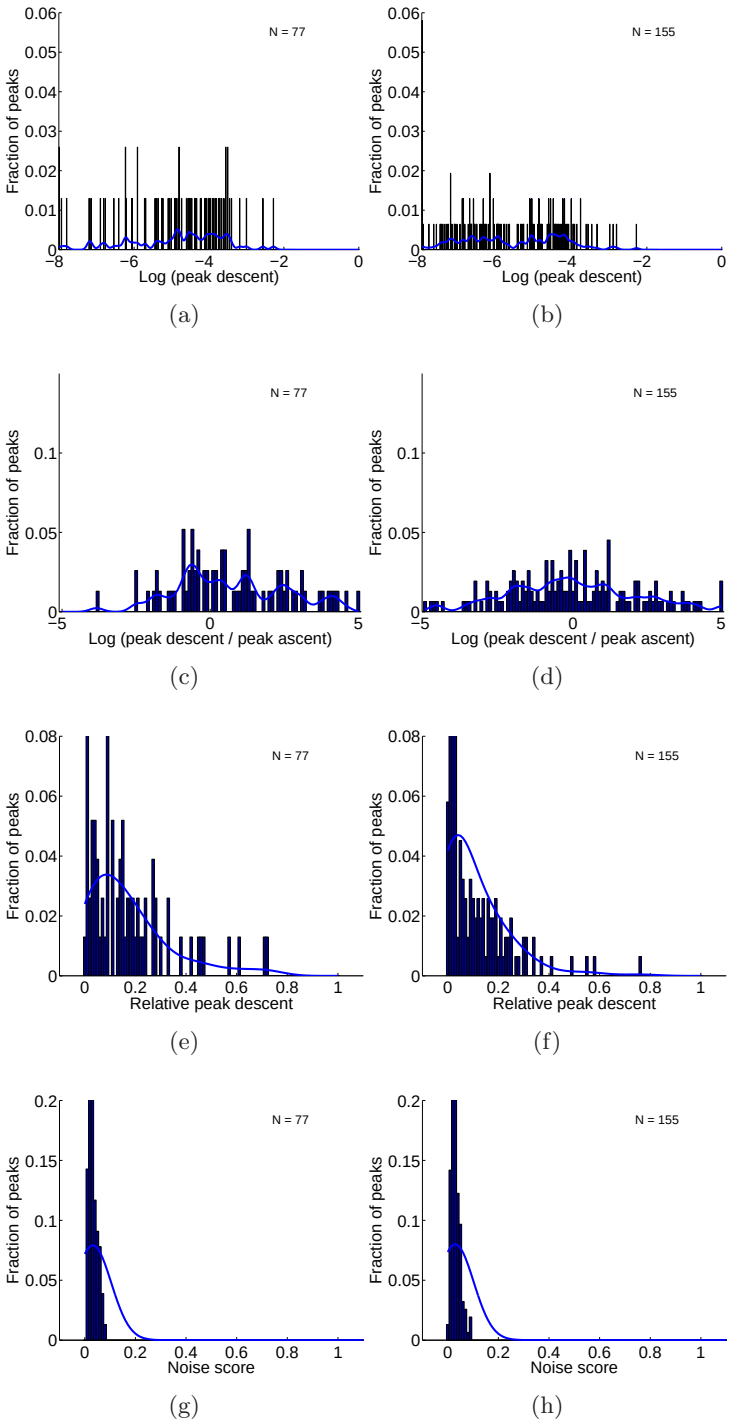


Figure C.14: Peak statistics for the $freq_{interlace}$ trace, from the *ballet2* sequence. Desired peaks, corresponding to motion events, are shown on the left, while the figures on the right illustrate false alarms. The Parzen estimations of the densities are also shown.

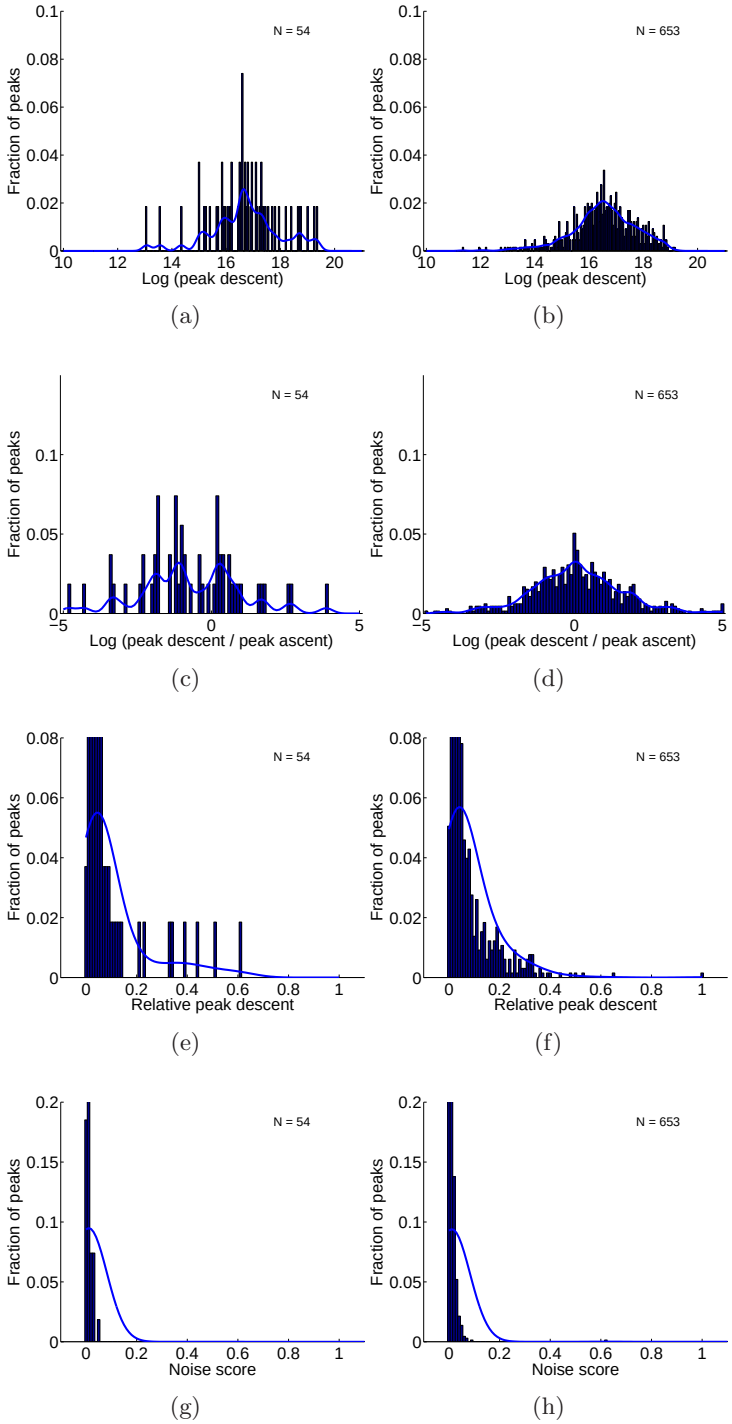


Figure C.15: Peak statistics for the $freq_{sharp}$ trace, from the *greenDancer*, and *maleDancer* sequences combined. Desired peaks, corresponding to motion events, are shown on the left, while the figures on the right illustrate false alarms. The Parzen estimations of the densities are also shown.

C.4 Audio

C.4.1 Maxima Detection

	#H	#FA	#M	P	R	M_{PR}	MD_{median}	MD_{mean}	MD_{var}
Sequence Training									
No smoothing									
M	287	1084	14	0.21	0.95	0.58	0	0.18	3.32
M_s	186	503	115	0.27	0.62	0.44	0	0.18	5.14
M_r	285	1051	16	0.21	0.95	0.58	0	0.09	3.98
M_{rs}	211	499	90	0.30	0.70	0.50	0	-0.08	5.60
With smoothing									
$\hat{M}$	247	560	54	0.31	0.82	0.56	0	-0.04	5.16
$\hat{M}_s$	206	379	95	0.35	0.68	0.52	0	-0.11	5.97
$\hat{M}_r$	250	583	51	0.30	0.83	0.57	0	0.06	5.11
$\hat{M}_{rs}$	192	374	109	0.34	0.64	0.49	0	-0.03	6.01

Table C.16: Audio

C.4.2 Peak Classification

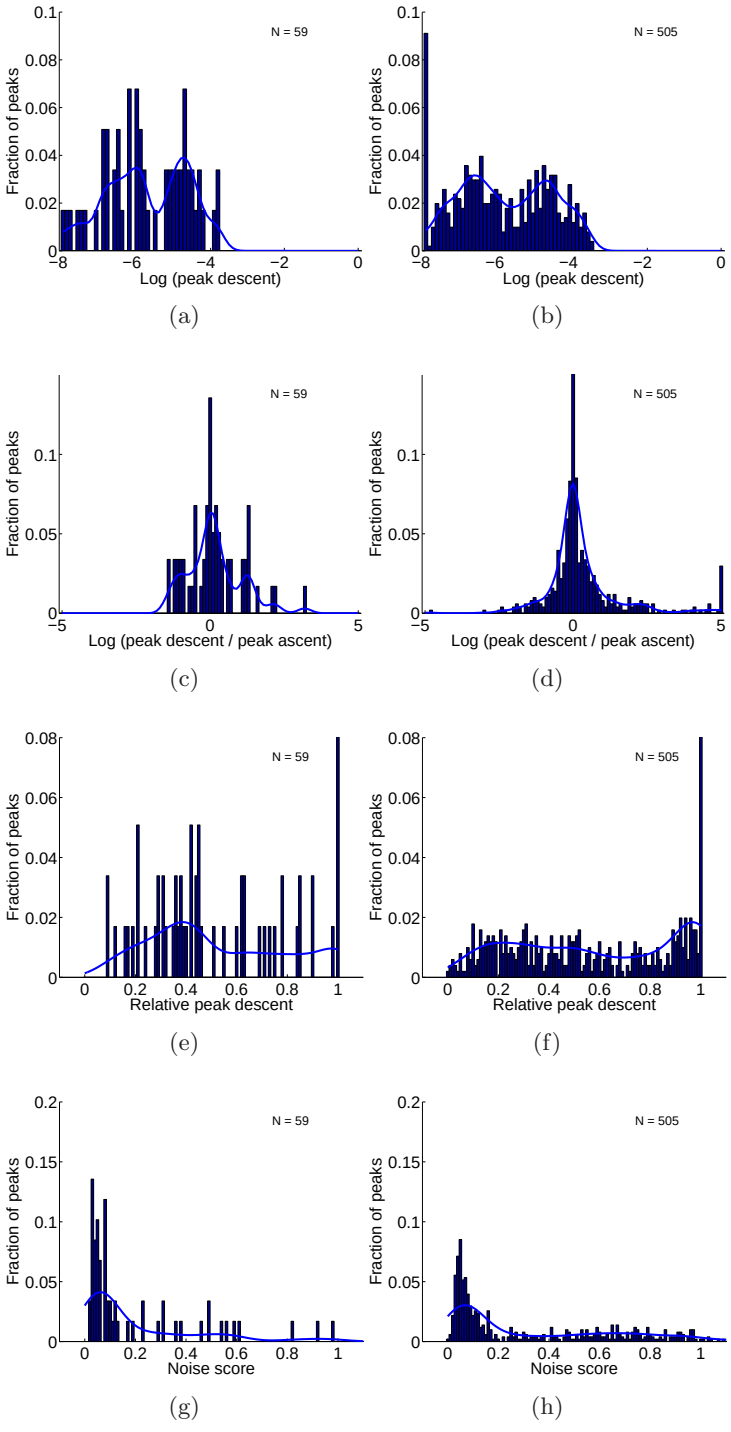


Figure C.16: Peak statistics for the *audio* trace, from the *greenDancer*, and *maleDancer* sequences combined. Desired peaks, corresponding to motion events, are shown on the left, while the figures on the right illustrate false alarms. The Parzen estimations of the densities are also shown.

Bibliography

[1] R. M. A. Ekin, A. Murat Tekalp. Automatic soccer video analysis and summarization. *IEEE Trans. Image Process.*, 12(7):796–807, 2003.

[2] I. Abdelqader, S. Rajah, W. Snyder, and G. Bilbro. Energy minimisation approach to motion estimation. *Signal Processing*, 28:291–309, September 1992.

[3] G. Adiv. Inherent ambiguities in recovering 3-d motion and structure from a noisy flow field. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(5):477–489, May 1989.

[4] D. Adjeroh, M. Lee, and C. Orji. Techniques for fast partitioning of compressed and uncompressed video. *Multimedia Tools and Applications*, 4(2):225–243, March 1997.

[5] G. Ahanger and T. Little. A survey of technologies for parsing and indexing digital video. *Journal of Video Communication and Image Representation*, 7(1):28–43, March 1996.

[6] P. Aigrain and P. Joly. The automatic real-time analysis of film editing and transition effects and its applications. *Computer and Graphics*, 18(1):93–103, 1994.

[7] A. Akutsu, Y. Tonomura, H. Hashimoto, and Y. Ohba. Video indexing using motion vectors. In P. Maragos, editor, *Proceedings of SPIE Conference on Visual Communications and Image Processing*, pages 1522–1530, November 1992.

[8] A. M. Alattar. Detecting and compressing dissolve regions in video sequences with a dvi multimedia image compression algorithm. In *IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 13–16, 1993.

[9] A. M. Alattar. Wipe scene change detector for segmenting uncompressed video sequences. In *Proc., Int. Symp. on Circuits and Systems Volume IV-Image and Video Processing*, pages 249–252, 1998.

[10] A. Albiol and V. Naranjo. Low complexity cut detection in the presence of flicker. In *Proc. IEEE ICIP '00*, volume III, pages 957–960, 2000.

[11] E. Ardizzone, G. Gioiello, M. L. Cascia, and D. Molinelli. A realtime neural approach to scene cut detection. In *SPIE Storage and Retrieval for Image and Video Databases IV*, 1996.

[12] F. Arman, A. Hsu, and M.-Y. Chiu. Feature management for large video databases. In *Proceedings of IS&T/SPIE Conference on Storage and Retrieval for Image and Video Databases I*, volume 1908, pages 2–12, 1993.

[13] F. Arman, A. Hsu, and M.-Y. Chiu. Image processing on compressed data for large video databases. In *Proceedings of First ACM Internation Conference on Multimedia*, pages 267–272, August 1993.

[14] S. Ayer and H. S. Sawhney. Layered representation of motion video using robust maximum likelihood estimation of mixture models and mdl encoding. In *Proceedings of the International Conference on Computer Vision*, pages 777–784, 1995.

[15] T. Bae, S. Jin, and Y. Ro. Video segmentation using hidden markov model with multimodal features. In *3rd International Conference on Image and Video Retrieval (CIVR '04)*, pages 401–409, 2004.

[16] R. Baillergeon. Representing the existence and the location of hidden objects: Object permanence in 6 and 8 month old infants. *Cognition*, 23:21–24, 1986.

[17] O. Bao and L. Guan. Scene change detection using dc coefficients. In *Proc. IEEE ICIP '02*, pages II: 421–424, 2002.

[18] M. Basseville. Detecting changes in signals and systems - a survey. *Automatica*, 1.24(3):309–326, 1988.

[19] J. Bergen, P. Burt, R. Hingori, and S. Peleg. Computing two motions from three frames. In *Proceedings of the Third International Conference on Computer Vision*, pages 27–32, December 1990.

[20] L. Bergen and F. Meyer. A novel approach to depth ordering in monocular image sequences. *CVPR*, pages 36–541, June 2000.

[21] J. Bescos. Real-time shot change detection over online mpeg-2 video. *IEEE Transactions on Circuits and Systems for Video Technology*, 14(4):475–484, April 2004.

[22] J. Bescos, G. Cisneros, and J. Menendez. Multidimensional comparison of shot detection algorithms. In *Proc. IEEE ICIP '02*, pages II: 401–404, 2002.

[23] J. Bescos, J. Martinez, J. Cabrera, J. Menendez, and G. Cisneros. Gradual shot transition detection based on multidimensional clustering. In *4th IEEE Southwest Symposium on Image Analysis and Interpretation*, pages 53–57, April 2000.

[24] J. Bescos, J. Menendez, G. Cisneros, J. Cabrera, and J. Martinez. A unified approach to gradual shot transition detection. In *Proc. IEEE ICIP '00*, pages Vol III: 949–952, 2000.

[25] J. Bescos, A. Movilla, J. Menendez, and G. Cisneros. Real time temporal segmentation of mpeg video. In *Proc. IEEE ICIP '02*, pages II: 409–412, 2002.

[26] F. N. Bezerra. A longest common subsequence approach to detect cut and wipe video transitions. In *Computer Graphics and Image Processing, XVII Brazilian Symposium on (SIBGRAPI'04)*, pages 154–160, 2004.

[27] J. Biemond, D. E. Boekee, L. Looijenga, and R. Plompen. A pel-recursive wiener based displacement estimation algorithm. *Signal Processing*, 13:399–412, 1987.

[28] M. Bierling. Displacement estimation by hierarchical block-matching. In *Proceedings of Visual Communications and Image Processing*, volume SPIE vol. 1001, pages 942–951, 1988.

[29] M. J. Black. Combining intensity and motion for incremental segmentation and tracking over long image sequences. In *European Conference on Computer Vision*, pages 485–493, 1992.

[30] M. J. Black and P. Anandan. Robust dynamic motion estimation over time. In *Proceedings of the IEEE Conference on Visual Pattern Recognition*, pages 296–302, 1991.

[31] M. J. Black and P. Anandan. A framework for the robust estimation of optical flow. In *Proceedings of Fourth International Conference on Computer Vision*, pages 231–236, May 1993.

[32] A. Bobick and J. Davis. Real-time recognition of activity using temporal templates. In *IEEE Workshop on Applications of Computer Vision*, pages 39–42, December 1996.

[33] A. Bobick and J. Davis. The recognition of human movement using temporal templates. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(3), March 2000.

[34] G. Boccignone, A. Chianese, V. Moscato, and A. Picariello. Using attention for video segmentation. In *19th International Conference on Pattern Recognition (ICPR'04)*, pages III: 838–841, 2004.

[35] G. Boccignone, A. Chianese, V. Moscato, and A. Picariello. Foveated shot detection for video segmentation. *IEEE Transactions on Circuits and Systems for Video Technology*, 15(3):365–377, March 2005.

[36] G. Boccignone and G. Percannella. Automated threshold selection for the detection of dissolves in mpeg videos. In *IEEE International Conference on Multimedia and Expo (ICME '00)*, page WP5, 2000.

[37] J. Boreczky and L. Rowe. Comparison of video shot boundary detection techniques. *Journal of Electronic Imaging*, 5(2):122–128, April 1996.

[38] J. Boreczky and L. Rowe. Comparison of video shot boundary detection techniques. *Proceedings of SPIE*, 2670:170–179, 1996.

[39] J. S. Boreczky and L. D. Wilcox. A hidden Markov model framework for video segmentation using audio and video features. In *Proc. Int. Conf. Acoustics, Speech, and Signal Processing (ICASSP-98)*, pages 3741–3744, May 1998.

[40] L. Böröczky, J. N. Driessen, and J. Biemond. Adaptive algorithms for pel-recursive displacement estimation. In *Proceedings SPIE VCIP*, pages 1210–1221, 1990.

[41] G. D. Borshukov, G. Bozdagi, Y. Altunbasak, and A. M. Tekalp. Motion segmentation by multistage affine classification. *IEEE Transactions on Image Processing*, 6(11):1591–1594, 1997.

[42] P. Bouthemy and F. Ganansia. Video partitioning and camera motion characterization for content-based video indexing. In *Proc. 3rd IEEE Int. Conf. Image Processing*, Lausanne, Switzerland, September 1996.

[43] P. Bouthemy, M. Gelgon, and F. Ganansia. A unified approach to shot change detection and camera motion characterization. *IEEE Transactions on Circuits and Systems for Video Technology*, 9(7):1030, October 1999.

[44] R. N. Bracewell, K.-Y. Chang, A. K. Jha, and Y.-H. Wang. Affine theorem for two-dimensional Fourier transform. *Electronics Letters*, 29(3), 1993.

[45] C. Bregler, M. Covell, and M. Slaney. Video rewrite: driving visual speech with audio. In *SIGGRAPH '97: Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, pages 353–360, New York, NY, USA, 1997. ACM Press/Addison-Wesley Publishing Co.

[46] M. Brunig and W. Niehsen. Fast full-search block matching. *IEEE Transactions on Circuits and Systems for Video Technology*, 11(2):241–247, February 2001.

[47] P. Burt and E. Adelson. The Laplacian pyramid as a compact image code. *IEEE Transactions on Communications*, 31:532–540, April 1983.

[48] T. Butz and J. Thiran. Shot boundary detection with mutual information. In *Proc. IEEE ICIP '01*, pages III: 422–425, 2001.

[49] J. M. C. Fowlkes, S. Belongie. Efficient spatiotemporal grouping using the nystrm method. In *CVPR*, Dec 2001.

[50] C. Cafforio and F. Rocca. Methods for measuring small displacements of television images. *IEEE transactions on Information Theory*, 22:573–579, 1976.

[51] J. Calic and E. Izquierdo. Temporal segmentation of mpeg video streams. *EURASIP Journal on Applied Signal Processing*, 2002(6):561–565, June 2002.

[52] H. M. L. C.C. Shih, H.R. Tyan. Shot change detection based on the reynolds transport theorem. In *Advances in Multimedia Information Processing - PCM 2001 : Second IEEE Pacific Rim Conference on Multimedia*, volume 2195, page 819, October 2001.

[53] Z. Cernekova, C. Kotropoulos, N. Nikolaidis, and I. Pitas. Video shot segmentation using fusion of svd and mutual information features. In *IEEE International Symposium on Circuits and Systems, 2005 (ISCAS 2005)*, volume 4, pages 3849 – 3852, May 2005.

[54] Z. Cernekova, C. Kotropoulos, and I. Pitas. Video shot boundary detection using singular value decomposition. In *Proc. of 4th European Workshop on Image Analysis for Multimedia Interactive Services (WIAMIS-2003)*, April 2003.

[55] Z. Cernekova, C. Nikou, and I. Pitas. Shot detection in video sequences using entropy-based metrics. In *Proc. IEEE ICIP '02*, pages III: 421–424, 2002.

[56] Z. Cernekova, I. Pitas, and C. Nikou. Information theory-based shot cut/fade detection and video summarization. *IEEE Transactions on Circuits and Systems for Video Technology*, 16(1):82–91, January 2006.

[57] M. Chang, M. Sezan, , and A. Tekalp. An algorithm for simultaneous motion estimation and segmentation. In *IEEE Int. Conf. on Acoustics, Speech and Signal Processing*, volume V, pages 221–224, April 1994.

[58] M. M. Chang, A. M. Tekalp, and M. I. Sezan. Simultaneous motion estimation and segmentation. *IEEE Transactions on Image Processing*, 6(9):1326–1333, 1997.

[59] L. Cheong. Scene-based shot change detection and comparative evaluation. *Computer Vision and Image Understanding*, 79(2):224–235, August 2000.

[60] L. Cheong and H. Huo. Shot change detection using scene-based constraint. *Multimedia Tools and Applications*, 14(2):175–186, June 2001.

[61] S.-Y. Chien, S.-Y. Ma, L.-G. Chen, K. N. Ngan, T. Sikora, and S. Ming-Ting. An efficient video segmentation algorithm for real-time mpeg-4 camera system. In *SPIE Visual communications and image processing*, volume 4067-3, pages 1087–1098, June 2000.

[62] J. W. Choi, M. G. Kang, and K. T. Park. An algorithm to extract camera-shaking degree and noise variance in the peak-trace domain. *IEEE Transactions on Consumer Electronics*, 44:1159–1168, August 1998.

[63] W. Christmas, J. Kittler, D. Koubaroulis, B. Levienaise-Obadia, and K. Messer. Generation of semantic cues for sports video annotation. In *Oulu International Workshop on Image Retrieval*, 2001.

[64] W.-T. Chu and J.-L. Wu. Development of realistic applications based on explicit event detection in broadcasting baseball videos. In *Proceedings of International Multimedia Modeling Conference*, pages 12–19, 2006.

[65] T. Chua, Y. Lin, and M. Kankanhalli. A general framework for video segmentation based on temporal multi-resolution analysis. In *Proc. International Workshop on Advanced Image Technology (IWAIT 2000)*, pages 119–124, January 2000.

[66] M. Chung. A scene boundary detection method. In *Proc. IEEE ICIP '00*, pages Vol III: 933–936, 2000.

[67] D. Comaniciu. An algorithm for data-driven bandwidth selection. *IEEE Trans. Pattern Anal. Mach. Intell.*, 25(2):281–288, 2003.

[68] J. Corridoni and A. D. Bimbo. Film semantic analysis. In *Computer Architectures for Machine Perception, 1995. Proceedings. (CAMP '95)*, pages 202–209, Como, Italy, 1995. IEEE Computer Society.

[69] C. Costaces, N. Nikoladis, and I. Pitas. Video shot detection and condensed representation. *IEEE Signal Processing Magazine*, March 2006.

[70] M. Covell and S. Ahmad. Analysis-by-synthesis dissolve detection. In *Proc. IEEE ICIP '02*, pages II: 425–428, 2002.

[71] A. Crawford, H. Denman, F. Kelly, F. Pitié, and A. Kokaram. Gradient based dominant motion estimation with integral projections for real time video stabilisation. In *IEEE International Conference on Image Processing (ICIP'04)*, October 2004.

[72] C. Croarkin and P. Tobias, editors. *NIST/SEMATECH e-Handbook of Statistical Methods*. U.S. National Institute of Standards and Technology, 2003.

[73] T. Darrell and A. Pentland. Robust estimation of a multi-layered motion representation. In *Proceedings of the IEEE Workshop on Visual Motion*, pages 173–178, October 1991.

[74] Data Project Srl. `http://www.dataproject.com/`.

[75] G. Davenport, T. G. A. Smith, and N. Pincever. Cinematic primitives for multimedia. *IEEE Computer Grpahics and Applications*, pages 67–74, July 1991.

[76] G. de Haan, P. Biezen, H. Huijgen, and O. A. Ojo. True-motion estimation with 3-d recursive search block matching. *IEEE Transactions on Circuits and Systems for Video Technology*, 3(5):368–379, October 1993.

[77] G. de Haan and P. W. A. C. Biezen. Sub-pixel motion estimation with 3-d recursive search block-matching. *Signal Processing: Image Communciation*, 6:229–239, 1994.

[78] G. de Haan and P. W. A. C. Biezen. An efficient true-motion estimator using candidate vectors from a parametric motion model. *IEEE Transactions on Circuits and Systems for Video Technology*, 8(1):85–91, February 1998.

[79] D. DeMenthon. Spatio-temporal segmentation of video by hierarchical mean shift analysis. In *SMVP 2002 (Statistical Methods in Video Processing Workshop)*, June 2002.

[80] H. Denman and A. Kokaram. A multiscale approach to shot change detection. In *The Irish Machine Vision and Image Processing Conference (IMVIP '04)*, September 2004.

[81] H. Denman, N. Rea, and A. Kokaram. Content based analysis for video from snooker broadcasts. In *International Conference on Image and Video Retrieval (CIVR)*, July 2002.

[82] R. Depommier and E. Dubois. Motion estimation with detection of occlusion areas. In *Proceedings of the IEEE Internation Conference on Acoustics, Speech, and Signal Processing (ICASSP '92)*, volume 3, pages 269–273, April 1992.

[83] S. Dixon. Automatic extraction of tempo and beat from expressive performances. *Journal of New Music Research*, 30(1):39—58, 2001.

[84] P. M. Dorin Comaniciu. Mean shift analysis and applications. In *ICCV*, pages 1197–1203, 1999.

[85] P. M. Dorin Comaniciu. Mean shift: A robust approach toward feature space analysis. *IEEE Trans. Pattern Anal. Mach. Intell.*, 24(5):603–619, 2002.

[86] P. M. Dorin Comaniciu, Visvanathan Ramesh. Real-time tracking of non-rigid objects using mean shift. In *CVPR*, pages 2142–2149, 2000.

[87] P. M. Dorin Comaniciu, Visvanathan Ramesh. The variable bandwidth mean shift and data-driven scale selection. In *ICCV*, pages 438–445, 2001.

[88] V. R. Dorin Comaniciu. Mean shift and optimal prediction for efficient object tracking. In *ICIP*, 2000.

[89] M. Drew, J. Wei, and Z. Li. Illumination-invariant image retrieval and video segmentation. *Pattern Recognition*, 32(8):1369–1388, August 1999.

[90] J. Driessen, L. Boroczky, and J. Biemond. Pel-recursive motion field estimation from image sequences. *Visual Communication and Image Representation*, 2:259–280, 1991.

[91] L.-Y. Duan, M. Xu, Q. Tian, C.-S. Xu, and J. Jin. A unified framework for semantic shot classification in sports video. *IEEE Transactions on Multimedia*, 7:1066 – 1083, December 2005.

[92] R. Duda, P. Hart, and D. Stork. *Pattern Classification*. Wiley Interscience, 2nd edition, 2000.

[93] F. Dufaux and J. Konrad. Efficient, robust, and fast global motion estimation for video coding. *IEEE Transaction on Image Processing*, 9(3):497–501, March 2000.

[94] F. Dufaux and F. Moscheni. Motion estimation techniques for digital tv: a review and a new contribution. *Proceedings of the IEEE*, 83(6):858–876, June 1995.

[95] S. Efstratiadis and A. Katsagellos. A model based, pel-recursive motion estimation algorithm. In *Proceedings IEEE ICASSP*, pages 1973–1976, 1990.

[96] A. Ekin, A. Tekalp, and R. Mehrotra. Automatic soccer video analysis and summarization. *IEEE Transactions on Image Processing*, 12:796–807, July 2003.

[97] D. P. Elias and N. G. Kingsbury. The recovery of a near optimal layer representation for an entire image sequence. In *The IEEE International Conference on Image Processing*, pages 735–738, 1997.

[98] W. Enkelmann. Investigations of multigrid algorithms for the estimation of optical flow fields in image sequences. *Computer Vision, Graphics and Image Processing*, 43:150–177, 1988.

[99] Y. M. Erkam, M. I. Sezan, and A. T. Erdem. A hierarchical phase-correlation method for motion estimation. In *Proceedings Conference on Information Science and Systems*, pages 419–424, 1991.

[100] R. Ewerth and B. Freisleben. Frame difference normalization: an approach to reduce error rates of cut detection algorithms for mpeg videos. In *Proc. IEEE ICIP '03*, pages II: 1009–1012, 2003.

[101] R. Ewerth and B. Freisleben. Improving cut detection in mpeg videos by gop-oriented frame difference normalization. In *19th International Conference on Pattern Recognition (ICPR'04)*, pages II: 807–810, 2004.

[102] T. Ezzat, G. Geiger, and T. Poggio. Trainable videorealistic speech animation. In *SIGGRAPH '02: Proceedings of the 29th annual conference on Computer graphics and interactive techniques*, pages 388–398, New York, NY, USA, 2002. ACM Press.

[103] A. Ferman and A. M. Tekalp. Efficient filtering and clustering for temporal video segmentation and visual summarization. *Journal of Visual Communication and Image Representation*, 9(4):338–351, 1998.

[104] W. Fernando, C. Canagarajah, and D. Bull. Fade and dissolve detection in uncompressed and compressed video sequences. In *Proc. IEEE ICIP '99*, volume 3, pages 299–303, October 1999.

[105] W. Fernando, C. Canagararajah, and D. Bull. Fade-in and fade-out detection in video sequences using histograms. In *The 2000 IEEE International Symposium on Circuits and Systems, 2000*, pages 709 – 712, May 2000.

[106] W. Fernando and K. Loo. Abrupt and gradual scene transition detection in mpeg-4 compressed video sequences using texture and macroblock information. In *Proc. IEEE ICIP '04*, volume 3, pages 1589–1592, October 2004.

[107] W. A. C. Fernando, C. N. Canagarajah, and D. R. Bull. Dfd based scene segmentation for h.263 video sequences. In *Proc., Int. Symp. on Circuits and Systems, Volume IV-Image and Video Processing, Multimedia and Communications*, pages 520–523, 1999.

[108] W. A. C. Fernando, C. N. Canagarajah, and D. R. Bull. Sudden scene change detection in mpeg-1 video sequences. In *Proc., IEEE Int. Workshop on Multimedia Signal Processing*, pages 259–264, 1999.

[109] W. A. C. Fernando, C. N. Canagarajah, and D. R. Bull. Video segmentation and classification for content based storage and retrieval using motion vectors. In *SPIE Storage and Retrieval for Image and Video Databases VII*, pages 687–698, 1999.

[110] A. W. Fitzgibbon. Stochastic rigidity: Image registration for nowhere-static scenes. In *Proceedings International Conference on Computer Vision*, volume 1, pages 662–670, 2001.

[111] J. Foote, M. Cooper, and A. Girgensohn. Creating music videos using automatic media analysis. In *MULTIMEDIA '02: Proceedings of the tenth ACM international conference on Multimedia*, pages 553–560, New York, NY, USA, 2002. ACM Press.

[112] R. M. Ford, C. Robson, D. Temple, and M. Gerlach. Metrics for shot boundary detection in digital video sequences. *ACM Multimedia Syst.*, 8(1):37–46, 2000.

[113] H. Fujiyoshi and A. Lipton. Real-time human motion analysis by image skeletonization. In *Proc. of the Workshop on Application of Computer Vision*, pages 15–21, October 1998.

[114] K. Fukunaga and L. Hostetler. The estimation of the gradient of a density function, with applications in pattern recognition. *IEEE Transactions in Information Theory*, 21:32–40, 1975.

[115] X. Gao and X. Tang. Unsupervised and model-free news video segmentation. In *Proceedings of the IEEE Workshop on Content-based Access of Image and Video Libraries (CBAIVL '01)*, page 58, 2001.

[116] U. Gargi, R. Kasturi, and S. Strayer. Performance characterization of video-shot-change detection methods. *IEEE Transactions on Circuits and Systems for Video Technology*, 10(1):1–13, February 2000.

[117] U. Gargi, S. Oswald, D. Cosiba, S. Devadiga, , and R. Kasturi. Evaluation of video sequence indexing and hierarchical video indexing. In *SPIE/IS&T Proceedings on Storage and Retrieval in Image and Video Databases - III*, volume 2420, pages 144–151, February 1995.

[118] S. Geman and D. Geman. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6(6):269–273, 1984.

[119] M. Ghanbari. The cross-search algorithm for motion estimation. *IEEE Transactions on Communications*, 38(7):950–953, July 1990.

[120] J.-L. Godard. À bout de souffle, 1960.

[121] Y. Gong and X. Liu. Video shot segmentation and classification. In *15th International Conference on Pattern Recognition (ICPR'00)*, volume 1, pages 860–863, 2000.

[122] Y. Gong, L. T. Sin, C. H. Chuan, H. Zhang, and M. Sakauchi. Automatic parsing of tv soccer programs. In *Multimedia Computing and Systems, 1995., Proceedings of the International Conference on*, pages 167 –174, 1995.

[123] M. Goto. An audio-based real-time beat tracking system for music with or without drum-sounds. *Journal of New Music Research*, 30(2):159—171, June 2001.

[124] F. Gouyon, A. Klapuri, S. Dixon, M. Alonso, G. Tzanetakis, C. Uhle, and P. Cano. An experimental comparison of audio tempo induction algorithms. *IEEE Transactions on Audio, Speech and Language Processing*, 14(5):1832—1844, 2006.

[125] T. Grimm and M. Grimm. *Basic Book Of Photography.* Plume, fifth edition, 2003.

[126] S. Guimaraes, M. Couprie, A. de Albuquerque Araujo, and N. Leite. Video fade detection by discrete line identification. In *17th International Conference on Pattern Recognition (ICPR'02)*, volume 2, pages 1013–1016, 2002.

[127] S. Guimaraes, M. Couprie, A. de Albuquerque Araujo, and N. Leite. Video segmentation based on 2d image analysis. *Pattern Recognition Letters*, 24(7):947–957, April 2003.

[128] S. Guimaraes, A. de Albuquerque Araujo, M. Couprie, and N. Leite. An approach to detect video transitions based on mathematical morphology. In *Proc. IEEE ICIP '03*, volume 2, pages 1021–1024, 2003.

[129] S. J. F. Guimaraes, M. Couprie, A. D. A. Araujo, and N. J. Leite. A method for cut detection based on visual rhythm. In *SIBGRAPI '01: Proceedings of the XIV Brazilian Symposium on Computer Graphics and Image Processing (SIBGRAPI'01)*, page 297, Washington, DC, USA, 2001. IEEE Computer Society.

[130] B. Gunsel, A. Ferman, and A. Tekalp. Temporal video segmentation using unsupervised clustering and semantic object tracking. *Journal of Electronic Imaging*, 7(3):592–604, July 1998.

[131] M. Hagiwara, M. Abe, and M. Kawamata. Shot change detection and camerawork estimation for old film restoration. In *Proceedings of IEEE International Symposium on Intelligent Signal Processing and Communication Systems*, number 1 in 17, pages 370–374, Nov 2004.

[132] A. Hampapur, R. Jain, and T. Weymouth. Digital video segmentation. In *ACM Multimedia '94 Proceedings*, pages 357–364. ACM Press, 1994.

[133] A. Hampapur, R. Jain, and T. Weymouth. Production model based digital video segmentation. *Journal of Multimedia Tools and Applications*, 1(1):1–38, March 1995.

[134] K. Han and A. Tewfik. Minimization of the spurious shot boundaries using principal components decomposition and progressive nonlinear filter. In *Proc. IEEE ICIP '99*, volume 4, pages 147–151, 1999.

[135] A. Hanjalic. Shot-boundary detection: unraveled and resolved? *IEEE Transactions on Circuits and Systems for Video Technology*, 12(2):90–105, February 2002.

[136] A. Hanjalic and L.-Q. Xu. Affective video content representation and modelling. *IEEE Transactions on Multimedia*, 7(1), February 2005.

[137] Y. Haoran, D. Rajan, and C. Den. A unified approach to detection of shot boundaries and subshots in compressed video. In *Proc. IEEE ICIP '03*, pages II: 1005–1008, 2003.

[138] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, second edition, 2004.

[139] S. Hayward. *Cinema Studies: The Key Concepts*. Routledge, third edition, 2006.

[140] F. Heitz and P. Bouthemy. Motion estimation and segmentation using a global bayesian approach. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 2305–2308, 1990.

[141] F. Heitz and P. Bouthemy. Multimodal estimation of discontinuous optical flow using markov random fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(12):1217–1232, 1993.

[142] W. Heng and K. Ngan. Integrated shot boundary detection using object-based technique. In *Proc. IEEE ICIP '99*, volume 3, pages 289–293, 1999.

[143] W. Heng and K. Ngan. An object-based shot boundary detection using edge tracing and tracking. *Journal of Video Communication and Image Representation*, 12(3):217–239, September 2001.

[144] W. Heng and K. Ngan. High accuracy flashlight scene determination for shot boundary detection. *Signal Processing: Image Communication*, 18(3):203–219, March 2003.

[145] W. Heng, K. Ngan, and M. Lee. Comparison of mpeg domain elements for low-level shot boundary detection. *Real-Time Imaging*, 5(5):341–358, October 1999.

[146] W. Hesseler and S. Eickeler. Mpeg-2 compressed-domain algorithms for video analysis. *EURASIP Journal on Applied Signal Processing*, 2006(Article ID 56940):1–11, 2006.

[147] L. Hill and T. Vlachos. On the estimation of global motion using phase correlation for broadcast applications. In *Proceedings IEE International Conference on Image Processing and its Applications*, volume 2, pages 721–725, July 1999.

[148] B. Horn and B. Schunck. Determining optical flow. *Artificial Intelligence*, 17:185–203, 1981.

[149] B. K. P. Horn. *Robot Vision*. M.I.T. Press, 1986.

[150] P. Hsu and H. Harashima. Spatiotemporal representation of dynamic objects. In *Proceedings IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 93)*, pages 14–19, June 1993.

[151] P. Hsu and H. Harashima. Detecting scene changes and activities in video databases. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP-94)*, volume 5, pages 14–19, April 1994.

[152] X.-S. Hua, L. Lu, and H.-J. Zhang. AVE - automated home video editing. In *ACM Multimedia (ACM MM 2003)*, pages 490–497, Berkeley, CA, USA, November 2003.

[153] X.-S. Hua, L. Lu, and H.-J. Zhang. Automatic music video generation based on temporal pattern analysis. In *ACM Multimedia (ACM MM 2004)*, pages 472–475, New York, USA, October 2004.

[154] X.-S. Hua, L. Lu, and H.-J. Zhang. Optimization-based automated home video editing system. *IEEE Transactions on Circuits and Systems for Video Technology*, 14(5):572–582, May 2004.

[155] C. Huang and B. Liao. A robust scene-change detection method for video segmentation. *IEEE Transactions on Circuits and Systems for Video Technology*, 11(12):1281–1288, December 2001.

[156] P. J. Huber. *Robust Statistics*. Wiley, 1981.

[157] M. Irani and S. Peleg. Image sequence enhancement using multiple motions analysis. In *Proceedings of the IEEE Conference on Computer Viusion and Pattern Recognition*, pages 216–221, June 1992.

[158] M. Irani, B. Rousso, and S. Peleg. Computing occluding and transparent motions. *International Journal of Computer Vision (IJCV)*, 12(1):5–16, February 1994.

[159] M. Isard and A. Blake. Condensation – conditional density propagation for visual tracking. *International Journal of Computer Vision*, 29, 1998.

[160] S.-L. Iu. Robust estimation of motion vector fields with discontinuity and occlusion using local outliers rejection. In *SPIE*, volume 2094, pages 588–599, 1993.

[161] 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 at Buffalo, 1978.

[162] A. Jepson and M. Black. Mixture models for optical flow computation. In *Conference on Computer Vision and Pattern Recognition (CVPR '93)*, pages 760–761, 1993.

[163] I. Jermyn and H. Ishikawa. Globally optimal regions and boundaries as minimum ratio weight cycles. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23:1075–1088, October 2001.

[164] R. H. Kallenberger and G. D. Cvjetnicanin. *Film into Video: A Guide to Merging the Technologies*. Focal Press, 1994.

[165] R. Kasturi and R. Jain. Dynamic vision. In *Computer Vision: Principles*, pages 469–480. IEE Computer Society Press, 1991.

[166] R. Kasturi, S. H. Strayer, U. Gargi, and S. Anti. An evaluation of color histogram based methods in video indexing. Technical Report CSE-96-053, Department of Computer Science and Engineering, Pennsylvania State University, September 1996.

[167] F. Kelly. *Fast Probabilistic Inference and GPU Video Processing*. PhD thesis, University of Dublin, Trinity College, 2005.

[168] S. Khan and M. Shah. Object based segmentation of video using color motion and spatial information. *Computer Vision and Pattern Recognition*, 2:746–751, 2001.

[169] E. Kijak, G. Gravier, P. Gros, L. Oisel, and F. Bimbot. Hmm based structuring of tennis videos using visual and audio cues. In *International Conference on Multimedia and Expo (ICME '03)*, volume 3, pages 309–312, July 2003.

[170] T. Kikukawa and S. Kawafuchi. Development of an automatic summary editing system for the audio-visual resources. *Transactions on Electronics and Information*, J75-A:204–212, 1992.

[171] H. Kim, S. J. Park, W. M. Kim, and M. H. Song. Processing of partial video data for detection of wipes. In *SPIE Storage and Retrieval for Image and Video Databases VII*, pages 280–289, 1999.

[172] J. Kim and R. H. Park. A fast feature-based block matching algorithm using integral projections. *IEEE Journal on Selected areas in Communications*, 10(5), June 1992.

[173] S. Kim and R. Park. A novel approach to scene change detection using a cross entropy. In *Proc. IEEE ICIP '00*, pages Vol III: 937–940, 2000.

[174] Y. T. Kiyotaka Otsuji and Y. Ohba. Video browsing using brightness data. In *Proc. of SPIE, VCIP'91*, volume 1606, pages 980–989, 1991.

[175] V. Kobla, D. DeMenthon, and D. Doermann. Detection of slow-motion replay sequences for identifying sports videos. In *IEEE 3rd Workshop on Multimedia Signal Processing*, pages 135 –140, 1999.

[176] V. Kobla, D. Doermann, K.-I. Lin, and F. C. Compressed domain video indexing techniques using dct and motion vector information in mpeg video. In *Proceedings of SPIE conference on Storage and Retrieval for Image and Video Databases V*, volume 3022, pages 200–211, February 1997.

[177] T. Koga, K. Iinmua, A. Hirano, Y. Iijima, and T. Ishiguro. Motion-compensated interframe coding for video conferencing. In *Proceedings NTC'81 (IEEE)*, pages G.5.3.1–G.5.3.4, 1981.

[178] A. Kokaram. *Motion Picture Restoration*. Springer-Verlag, ISBN: 3540760407, May 1998.

[179] A. Kokaram. Practical, unified, motion and missing data treatment in degraded video. *Journal of Mathematical Imaging and Vision*, 20:163–177, 2004.

[180] A. Kokaram, R. Bornard, A. Rares, D. Sidorov, J. H. Chenot, L. Laborelli, and J. Biemond. Digital restoration systems: Coping with reality. *SMPTE Motion Imaging Journal*, pages 225–231, July/August 2003.

[181] A. Kokaram, B. Collis, and S. Robinson. Automated rig removal with bayesian motion interpolation. *IEE Proceedings on Vision, Image and Signal Processing*, 152:407—414, August 2005.

[182] A. Kokaram and P. Delacourt. A new global motion estimation algorithm and its application to retrieval in sports events. In *2001 IEEE International Workshop on Multimedia Signal Processing (MMSP 2001)*, October 2001.

[183] A. Kokaram, N. Rea, R. Dahyot, A. M. Tekalp, P. Bouthemy, P. Gros, and I. Sezan. Browsing sports video. *IEEE Signal Processing Magazine: Special Issue on Semantic Retrieval of Multimedia*, March 2006.

[184] J. Konrad and E. Dubois. Bayesian estimation of motion vector fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(9), September 1992.

[185] I. Koprinska and S. Carrato. Temporal video segmentation: A survey. *Signal Processing: Image Communication*, 16(5):477–500, January 2001.

[186] C. D. Kuglin and D. C. Hines. The phase correlation image alignment method. In *Proceedings of IEEE International Conference on Cybernetics and Society*, pages 163–165, 1975.

[187] M. P. Kumar, P. Torr, and A. Zisserman. Learning layered motion segmentations of video. In *Proceedings of the Tenth IEEE International Conference on Computer Vision (ICCV'05)*, volume 1, pages 33–40, 2005.

[188] S. Kumar, M. Biswas, and T. Q. Nquyen. Global motion estimation in frequency and spatial domain. In *Proceedings of IEEE International Conference Acoustics, Speech, and Signal Processing (ICASSP '04)*, volume 3, pages 333–336, May 2004.

[189] W. B. Laura Teodosio. Salient video stills: content and context preserved. In *Proceedings of the first ACM international conference on Multimedia*, pages 39—46, August 1993.

[190] P. Leconte. La fille sur le pont, 1999.

[191] I. H. Lee and R. H. Park. A fast block matching algorithm using integral projections. In *Proceedings of IEEE TENCON '87 Conference*, pages 18.3.1–18.3.5, August 1987.

[192] J. H. Lee and J. B. Ra. Block motion estimation based on selective integral projections. In *IEEE ICIP*, volume I, pages 689–693, 2002.

[193] M. C. Lee and D. A. Adjeroh. Indexing and retrieval in visual databases via colour ratio histograms. In *Proceedings, 1st International Conference on Visual Information Retrieval*, pages 309–316, 1996.

[194] S. Lee, Y. Kim, and S. Choi. Fast scene change detection using direct feature extraction from mpeg compressed videos. *IEEE Transactions on Multimedia*, 2:240–254, December 2000.

[195] S. Lefevre, J. Holler, and N. Vincent. A review of real-time segmentation of uncompressed video sequences for content-based search and retrieval. *Real-Time Imaging*, 9(1):73–98, February 2003.

[196] R. Leser. Applied video and game analysis in professional soccer. In *World Conference of Performance Analysis of Sport VII*, August 2006.

[197] Y. Leung, J.-S. Zhang, and Z.-B. Xu. Clustering by scale-space filtering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(12):1396–1410, 2000.

[198] A. Levite. Music video for 'Mutescreamer' (Beans), warp records, 2003, 1999.

[199] W. Li and E. Salari. Successive elimination algorithm for motion estimation. *IEEE Transactions on Image Processing*, 4(1):105–107, January 1995.

[200] Z. Li. Spatio-temporal joint probability images for video segmentation. In *Proc. IEEE ICIP '00*, pages Vol II: 295–298, 2000.

[201] R. Lienhart. Comparison of automatic shot boundary detection algorithms. In *Proc. SPIE Image and Video Processing VII*, volume 3656, pages 290–301, January 1999.

[202] R. Lienhart. Reliable transition detection in videos: a survey and practitioner's guide. *International Journal of Image and Graphics*, 1(3):469–486, 2001.

[203] R. Lienhart and A. Zaccarin. A system for reliable dissolve detection in videos. In *Proc. IEEE ICIP '01*, pages III: 406–409, 2001.

[204] X. Liu and T. Chen. Shot boundary detection using temporal statistics modeling. In *IEEE International Conference on Acoustics, Speech, and Signal Processing, 2002 (ICASSP '02)*, volume 4, pages 3389—3392, May 2002.

[205] Y. Liu, W. Wang, W. Gao, and W. Zeng. A novel compressed domain shot segmentation algorithm on H.264/AVC. In *Proc. IEEE ICIP '04*, pages IV: 2235–2238, 2004.

[206] LRS Sports Software Ltd. `http://www.lrs.com/LRSSports/`.

[207] H. Lu and Y. Tan. An effective post-refinement method for shot boundary detection. In *Proc. IEEE ICIP '03*, pages II: 1013–1016, 2003.

[208] H. Lu and Y. Tan. An effective post-refinement method for shot boundary detection. *IEEE Transactions on Circuits and Systems for Video Technology*, 15(11):1407–1421, November 2005.

[209] L. Lucchese. Estimating affine transformation in the frequency domain. In *Proceedings IEEE International Conference on Image Proceedings (ICIP '01)*, volume 2, pages 7–10, October 2001.

[210] G. Lupatini, C. Saraceno, , and R. Leonardi. Scene break detection: a comparison. In *Int. Workshop on Research Issues in Data Engineering*, pages 34–41, 1998.

[211] A. Machi and M. Tripiciano. Video shot detection and characterisation in semi-automatic digital video restoration. In *15th International Conference on Pattern Recognition (ICPR'00)*, pages Vol I: 855–859, 2000.

[212] J. F. A. Magarey. *Motion estimation using complex wavelets*. PhD thesis, Cambridge University, 1997.

[213] S. Mann and R. W. Picard. Video orbits of the projective group: a simple approach to featureless estimation of parameters. *IEEE Transactions on Image Processing*, 6(9), September 1997.

[214] D. M. Martinez. *Model-based motion estimation and its application to restoration and interpolation of motion pictures*. PhD thesis, Massachusetts Institute of Technology, 1986.

[215] M. Massey. The new cronophotography: novel applications of salient stills. In *Proceedings of the fourth ACM international conference on Multimedia*, pages 401—402, November 1996.

[216] M. Massey and W. Bender. Salient stills: process and practice. *IBM Systems Journal*, 35(3-4):557—573, 1996.

[217] Match Analysis Inc. `http://www.matchanalysis.com/`.

[218] K. Meier, T. Ngan. Video segmentation for content-based coding. *Circuits and Systems for Video Technology, IEEE Transactions on*, 9:1190–1203, Dec 1999.

[219] H. Meng, Y. Juan, and S. Chang. Scene change detection in a mpeg compressed video sequence. In *Proceedings of SPIE Conference on Digital Video Compression: Algorithms and Technologies*, volume 2419, pages 14–25, February 1995.

[220] Microsoft Plus! Digital Media Edition. `http://www.microsoft.com/windows/plus/dme/dmehome.asp`.

[221] J. Migdal and W. E. L. Grimson. Background subtraction using markov thresholds. In *WACV-MOTION '05: Proceedings of the IEEE Workshop on Motion and Video Computing (WACV/MOTION'05) - Volume 2*, pages 58–65, Washington, DC, USA, 2005. IEEE Computer Society.

[222] P. Milanfar. A model of the effect of image motion in the radon transform domain. *IEEE Transactions on Image Processing*, 8(9):1276–1281, 1999.

[223] K. Minolta. *Dynax 5D Guide Book*. Konica Minolta Photo Imaging, Inc, 2005.

[224] B. Moore and B. Glasberg. A revision of zwicker's loudness model. *Acta Acustica united with Acustica*, 82:335–345, 1996.

[225] MPEG-4. *Coding of moving pictures and audio*. ISO/IEC JTC1/SC29/WG11 14496-2:Visual, 1998.

[226] D. Murray and B. Buxton. Scene segmentation from visual motion using global optimization. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 9(2):220–228, February 1987.

[227] muvee Technologies Pte Ltd. `http://www.muvee.com/`.

[228] A. Nagasaka and Y. Tanaka. Automatic scene-change detection method for video works. In *Proc. 40th National Conf. Information Processing Society of Japan*, volume 1Q-5, 1990.

[229] A. Nagasaka and Y. Tanaka. Automatic video indexing and full-video search for object appearances. In *Proceedings of the IFIP TC2/WG 2.6 Second Working Conference on Visual Database Systems II*, pages 113–127. North-Holland, 1992.

[230] H. Nagel. Image sequences: ten (octal) years; from phenomenology towards a theoretical foundation. In *IEEE ICASSP*, pages 1174–1185, 1986.

[231] H. Nagel. On a constraint equation for the estimation of displacement rates in image sequences. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11:13–30, January 1989.

[232] H.-H. Nagel. On the estimation of optic flow: Relations between different approaches and some new results. *Artificial Intelligence*, 33:299–324, 1987.

[233] H. H. Nagel and W. Enkelmann. An investigation of smoothness constraints for the estimation of displacement vector fields from image sequence. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(5):565–593, September 1986.

[234] S. Nayar and M. Ben-Ezra. Motion-based motion deblurring. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26:689–698, June 2004.

[235] J. A. Nelder and R. Mead. A simplex method for function minimization. *Computer Journal*, 7:308–313, 1965.

[236] A. Netravali and J. Robbins. Motion-compensated television coding: Part 1. *The Bell System Technical Journal*, pages 59:1735–1745, November 1980.

[237] C. Ngo, T. Pong, and R. Chin. Detection of gradual transitions through temporal slice analysis. In *Proceedings IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 99)*, pages I: 36–41, 1999.

[238] C. Ngo, T. Pong, and R. Chin. Video partitioning by temporal slice coherency. *IEEE Transactions on Circuits and Systems for Video Technology*, 11(8):941–953, August 2001.

[239] C. Ngo, T. Pong, and H. Zhang. Motion-based video representation for scene change detection. In *15th International Conference on Pattern Recognition (ICPR'00)*, pages Vol I: 827–830, 2000.

[240] C. Ngo, T. Pong, and H. Zhang. Motion-based video representation for scene change detection. *Int. J. of Computer Vision*, 50(2):127–142, November 2002.

[241] N.Rea, R. Dahyot, and A. Kokaram. Classification and representation of semantic content in broadcast tennis videos. In *IEEE International Conference on Image Processing (ICIP'05)*, September 2005.

[242] K. Nummiaro, E. Koller-Meier, and L. V. Gool. An adaptive color-based particle filter. *Image and Vision Computing*, 21:99–110, 2003.

[243] A. L. Nuno Vasconcelos. Empirical bayesian motion segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(2):217–221, February 2001.

[244] J.-M. Odobez and P. Bouthemy. Robust multiresolution estimation of parametric motion models. *Journal of Visual Communication and Image Representation*, 6(4):348–365, December 1995.

[245] K. Otsuji and Y. Tonomura. Projection detecting filter for video cut detection. In *ACM Multimedia '93 Proceedings*, pages 251–257, 1993.

[246] K. Otsuji and Y. Tonomura. Projection detecting filter for video cut detection. *Multimedia Systems*, pages 205 –210, March 1994.

[247] M. Park, R. Park, and S. Lee. Efficient shot boundary detection for action movies using blockwise motion-based features. In *International Symposium on Visual Computing (ISVC '05)*, pages 478–485, 2005.

[248] N. Patel and I. Sethi. Compressed video processing for cut detection. *IEE Proceedings - Vision, Image, and Signal Processing*, 143(5):315–323, October 1996.

[249] S. Pei and Y. Chou. Efficient and effective wipe detection in mpeg compressed video based on the macroblock information. In *Proc. IEEE ICIP '00*, volume 3, pages 953–956, 2000.

[250] S.-C. Pei and Y.-Z. Chou. Efficient mpeg compressed video analysis using macroblock type information. *IEEE Transactions on Multimedia*, 1(4):321–333, December 1999.

[251] A. Petrovic, O. D. Escoda, and P. Vandergheynst. Multiresolution segmentation of natural images: From linear to non-linear scale-space representations. *IEEE Transactions on Image Processing*, 13(8):1104–1114, August 2004.

[252] G. Pingali, Y. Jean, and I. Carlbom. Real time tracking for enhanced tennis broadcasts. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 260 – 265, June 1998.

[253] Pinnacle Systems ltd. `http://www.pinnaclesys.com/`.

[254] F. Pitié, R. Dahyot, F. Kelly, and A. Kokaram. A new robust technique for stabilizing brightness fluctuations in image sequences. In *2nd Workshop on Statistical Methods in Video Processing (ECCV 2004)*, pages 153–164, May 2004.

[255] T. Poggio and V. T. an C. Koch. Computational vision and regularization theory. *Nature*, pages 314–319, 1985.

[256] S. Porter, M. Mirmehdi, and B. Thomas. Video cut detection using frequency domain correlation. In *15th International Conference on Pattern Recognition (ICPR'00)*, volume 3, pages 413–416, 2000.

[257] M. C. Potter and J. F. Foss. *Fluid Dynamics*. Great Lakes Press, 1982.

[258] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical Recipes in C*. Cambridge University Press, second edition, 1995.

[259] N. Rea. *High-level Event Detection in Broadcast Sports Video*. PhD thesis, University of Dublin, Trinity College, 2004.

[260] N. Rea, R. Dahyot, and A. Kokaram. Semantic event detection in sports through motion understanding. In *3rd International Conference on Image and Video Retrieval (CIVR 04)*, July 2004.

[261] B. S. Reddy and B. N. Chatterji. An FFT-based technique for translation, rotation, and scale-invariant image registration. *IEEE Transactions on Image Processing*, 5(8), August 1996.

[262] W. Ren and S. Singh. Automatic video segmentation using machine learning. In *Proceeding of 5th International Conference on Intelligent Data Engineering and Automated Learning (IDEAL 2004)*, pages 285–293, 2004.

[263] O. Robles, P. Toharia, A. Rodri'guez, and L. Pastor. Using adaptive thresholds for automatic video cut detection. In *Proceedings of TRECVID 2004*, Nov 2004.

[264] Y. Rubner, C. Tomasi, and L. Guibas. The earth mover's distance as a metric for image retrieval. Technical Report STAN-CS-TN-98-86, Computer Science Department, Stanford University, September 1998.

[265] Y. Rubner, C. Tomasi, and L. Guibas. The earth mover's distance as a metric for image retrieval. *International Journal of Computer Vision*, 40(2):99–121, 2000.

[266] Y. Rui, A. Gupta, and A. Acero. Automatically extracting highlights for tv baseball programs. In *Proc. ACM Multimedia*, pages 105 –115, October 2000.

[267] D. Saur, Y.-P. Tan, S. Kulkarni, and P. Ramadge. Automated analysis and annotation of basketball video. In *SPIE Storage and Retrieval for Still Image and Video Databases V*, volume 3022, pages 176–187, 1997.

[268] A. Savitzky and M. J. E. Golay. Smoothing and differentiation of data by simplified least squares procedures. *Analytical Chemistry*, 36:1627–1639, 1964.

[269] E. D. Scheirer. Tempo and beat analysis of acoustic musical signals. *Journal of the Acoustic Society of America*, 103(1):588—601, 1998.

[270] W. A. Sethares, R. D. Morris, and J. C. Sethares. Beat tracking of audio signals using low level audio features. *IEEE Trans. On Speech and Audio Processing*, 13(2), March 2005.

[271] A. Shahraray. Scene change detection and content-based sampling of video sequences. *Digital Video Compression: Algortithms and Technologies*, 2419:2–13, 1995.

[272] C. Shannon and W. Weaver. *The Mathematical Theory of Communication*. University of Illinois Press, 1949.

[273] J. Shi and J. Malik. Motion segmentation and tracking using normalized cuts. In *Int. Conf. Computer Vision*, pages 1154–1160, Jan 1998.

[274] H.-C. Shih and C.-L. Huang. MSN: statistical understanding of broadcasted baseball video using multi-level semantic network. *IEEE Transactions on Broadcasting*, 51:449 – 459, December 2005.

[275] A. Smeaton and P. Over. The TREC-2002 video track report. In *The Eleventh Text REtrieval Conference (TREC 2002)*, pages 69–85, March 2003.

[276] Snell & Wilcox. Archangel Ph.C Image Restoration System. `http://www.snellwilcox.com/`.

[277] B. Song and J. Ra. Automatic shot change detection algorithm using multi-stage clustering for mpeg-compressed videos. *Journal of Video Communication and Image Representation*, 12(3):364–385, September 2001.

[278] B. C. Song and J. B. Ra. Fast edge map extraction from mpeg compressed video data for video parsing. In *Proc. SPIE Storage and Retrieval for Image and Video Databases VII*, volume 3656, pages 710–721. SPIE, Jan 1999.

[279] M. Sonka, V. Jlavac, and R. Boyle. *Image Processing, Analysis, and Machine Vision*. Chapman and Hall, 1st edition, 1993.

[280] M. Soriano, S. Huovinen, B. Martinkauppi, and M. Laaksonen. Skin detection in video under changing illumination conditions. In *Proc. 15th International Conference on Pattern Recognition*, pages 839–842, 2000.

[281] E. S. Spelke. Principles of object segregation. *Cognitive Science*, 14:29–56, 1990.

[282] E. S. Spelke, K. Breinlinger, K. Jacobson, and A. Phillips. Gestalt relations and object perception: A developmental study. *Perception*, 22(12):1483–1501, 1993.

[283] SportsCad (Seaside Software). `http://www.sportscad.com/`.

[284] Sportstec International Ltd. `http://www.allsportstec.com/`.

[285] E. Steinbach, P. Eisert, and B. Girod. Motion-based analysis and segmentation of image sequences using 3-d scene models. *Signal Process.*, 66(2):233–247, 1998.

[286] C. Stiller. Object-based estimation of dense motion fields. *IEEE Transactions on Image Processing*, 6(2):234–250, February 1997.

[287] C. Stiller and B. H. Hurtgen. Combined displacement estimation and segmentation in image sequences. In *Proceedings of the SPIE / EUROPTO Conference on Video Communications and PACS for Medical Applications*, volume SPIE 1977, pages 276–287, 1993.

[288] C. Stiller and J. Konrad. Estimating motion in image sequences. *IEEE Signal Processing Magazine*, 16(4):70–91, July 1999.

[289] G. Sudhir, J. C. M. Lee, and A. K. Jain. Automatic classification of tennis video for high-level content-based retrieval. In *Proceedings of the 1998 International Workshop on Content-Based Access of Image and Video Databases (CAIVD '98)*, January 1998.

[290] R. Suzuki, Y. Iwadate, and M. Mihoh. Image wave: A study on image synchronization. In *MULTIMEDIA '99: Proceedings of the seventh ACM international conference on Multimedia*, pages 179–182, Orlando, Florida, United States, November 1999. ACM Press.

[291] R. Suzuki, Y. Iwadate, R. Nakatsu, and M. Mihoh. Multimedia montage—counterpoint synthesis of movies. *Multimedia Tools and Applications*, 11(3):311–331, August 2000.

[292] S. J. Swain and D. H. Ballard. Color indexing. *International Journal of Computer Vision*, 7(1):11–32, 1991.

[293] D. Swanberg, C. Shu, and R. Jain. Knowledge-guided parsing in video databases. *Proc. SPIE Conf. on Storage and Retrieval of Image and Video Databases*, 1908:13–24, February 1993.

[294] K. Sze, K. Lam, and G. Qiu. Scene cut detection using the colored pattern appearance model. In *Proc. IEEE ICIP '03*, pages II: 1017–1020, 2003.

[295] R. Szeliski and S. B. Kang. Shape ambiguities in structure from motion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(5):506–512, May 1997.

[296] Y.-P. Tan, J. Nagamani, and H. Lu. Modified Kolmogorov-Smirnov metric for shot boundary detection. *Electronics Letters*, 39(18):1313–1315, September 2003.

[297] D. Tancharoen, S. Jitapunkul, P. Kittipanya-Ngam, N. Siritaranukul, K. N. Ngan, T. Sikora, and S. Ming-Ting. Video segmentation based on adaptive combination of multiple features according to mpeg-4. In *SPIE Visual communications and image processing*, volume 4067-3, pages 1099–1106, June 2000.

[298] A. M. Tekalp. *Digital Video Processing*. Prentice-Hall, 1995.

[299] A. M. Tekalp, Y. Altunbasak, and P. E. Eren. Region based parametric motion segmentation using color information. *Journal of Graphical Models and Image Processing*, 60(1):13–23, January 1998.

[300] L. Teodosio and W. Bender. Salient stills. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMCCAP)*, 1:16—36, February 2005.

[301] G. A. Thomas. Television motion measurement for DATV and other applications. Technical Report BBC–RD–1987–11, BBC Research Department, 1987.

[302] W. B. Thompson. Combining motion and contrast for segmentation. *IEEE Trans. Pattern Analysis and Machine Intelligence*, pages 543–549, Nov 1980.

[303] Y. Tonomura. Video handling based one structured information for hypermedia systems. In *Proceedings of the ACM Int'l Conf. on Multimedia Information Systems*, pages 333–344, 1991.

[304] P. H. S. Torr, R. Szeliski, and P. Anandan. An integrated Bayesian approach to layer extraction from image sequences. In *Proceedings of the International Conference on Computer Vision*, pages 983–990, 1999.

[305] V. Tovinkere and R. J. Qian. Detecting semantic events in soccer games: Towards a complete solution. In *Proc. ICME 2001*, August 2001.

[306] B. Truong. Improved fade and dissolve detection for reliable video segmentation. In *Proc. IEEE ICIP '00*, volume 3, pages 961–964, 2000.

[307] B. T. Truong, C. Dorai, and S. Venkatesh. New enhancements to cut, fade, and dissolve detection processes in video segmentation. In *MULTIMEDIA '00: Proceedings of the eighth ACM international conference on Multimedia*, pages 219–227, New York, NY, USA, 2000. ACM Press.

[308] H. Ueda, T. Miyatake, and S. Yoshizawa. Impact: An interactive natural-motion picture dedicated multimedia authoring system. In *CHI'91 Coference Proceedings*, pages 343–350, 1991.

[309] R. Urquhart. Graph theoretical clustering based on limited neighborhood sets. *Pattern Recognition*, 15(3):173–187, 1982.

[310] A. Vadivel, M. Mohan, S. Sural, and A. Majumdar. Object level frame comparison for video shot detection. In *Proceedings of the IEEE Workshop on Motion and Video Computing (WACV/MOTION'05)*, volume 2, pages 235–240, 2005.

[311] N. Vasconcelos and A. Lippman. Empirical Bayesian EM-based motion segmentation. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, pages 527–532, 1997.

[312] N. Vasconcelos and A. Lippman. Bayesian video shot segmentation. In *Proceedings of Neural Information Processing Systems 13*, 2000.

[313] N. Vasconcelos and A. Lippman. Statistical models of video structure for content analysis and characterization. *IEEE Transactions on Image Processing*, 9(1), January 2000.

[314] J. Vega-Riveros and K. Jabbour. Review of motion estimation techniques. In *Communications, Speech and Vision, IEE Proceedings I*, volume 136, pages 397–404, Dec 1989.

[315] T. Vlachos. Cut detection in video sequences using phase correlation. *IEEE Signal Processing Letters*, 7(7):173–175, July 2000.

[316] H. Wang, A. Divakaran, A. Vetro, S. F. Chang, and H. Sun. Survey of compressed-domain features used in audio-visual indexing and analysis. *Journal of Visual Communication and Image Representation*, 14(2):50–183, June 2003.

[317] J. Wang and N. Parameswaran. Analyzing tennis tactics from broadcasting tennis video clips. In *Proceedings of the 11th International Multimedia Modelling Conference (MMM 2005)*, pages 102 – 106, January 2005.

[318] J. Wang, B. Thiesson, Y. Xu, and M. Cohen. Image and video segmentation by anisotropic kernel mean shift. In *ECCV*, volume 2, pages 238–249, 2004.

[319] J. Wang, Y. Xu, H.-Y. Shum, and M. Cohen. Video tooning. In *ACM Trans. on Graphics (Proc. of SIGGAPH2004)*, volume 23-3, pages 574–583, 2004.

[320] J. Y. A. Wang and E. H. Adelson. Layered representation for motion analysis. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 361–366, June 1993.

[321] J. Y. A. Wang and E. H. Adelson. Representing moving images with layers. *The IEEE Transactions on Image Processing Special Issue: Image Sequence Compression*, 3(5):625–638, September 1994.

[322] J. Z. Wang, J. Li, R. M. Gray, and G. Wiederhold. Unsupervised multiresolution segmentation for images with low depth of field. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(1):85–90, 2001.

[323] M. Wetheimer. *A Sourcebook of Gestalt Psychology*, chapter Laws of organisation in perceptual forms, pages 71–88. Harcourt, Brace, and Company, 1938.

[324] A. Whitehead, P. Bose, and R. Laganiere. Feature based cut detection with automatic threshold selection. In *3rd International Conference on Image and Video Retrieval (CIVR '04)*, pages 410–418, July 2004.

[325] G. Winkler. *Image Analysis, Random Fields, and Dynamic Monte Carlo Methods: A Mathematical Introduction*. Springer, 1995.

[326] S. Wu, W. Lin, L. Jiang, W. Xiong, L. Chen, and S. H. Ong. An objective out-of-focus blur measurement. In *Fifth International Conference on Information, Communications and Signal Processing*, pages 334–338, December 2005.

[327] Z. Wu and R. Leahy. An optimal graph theoretic approach to data clustering: Theory and its application to image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11:1101–1113, November 1993.

[328] L. Xie and S.-F. Chang. Structure analysis of soccer video with hidden markov models. In *IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2002.

[329] W. Xiong and J. Lee. Efficient scene change detection and camera motion annotation for video classification. *Computer Vision and Image Understanding*, 71(2):166–181, August 1998.

[330] W. Xiong, J. Lee, and M. C. Ip. Net comparison: a fast and effective method for classifying image sequences. In *Proc. SPIE Conf. Storage and Retrieval for Image and Video Databases III*, volume 2420, pages 318–328, 1995.

[331] M. Xu, L.-T. Chia, H. Yi, and D. Rajan. Affective content detection in sitcom using subtitle and audio. In *Multi-Media Modelling Conference Proceedings, 2006 12th International*, pages 129–134, January 2006.

[332] M. Xu, L.-Y. Duan, C. Xu, M. Kankanhalli, and Q. Tian. Event detection in basketball video using multiple modalities. In *Fourth International Conference on Information, Communications and Signal Processing*, volume 3, pages 1526 – 1530, December 2003.

[333] W. Y. Smoothness in layers: Motion segmentation using nonparametric mixture estimation. In *Proceedings of IEEE conference on Computer Vision and Pattern Recognition*, pages 520–527, 1997.

[334] M.-H. Yang. *Hand Gesture Recognition and Face Detection in Image*. PhD thesis, University of Illinois at Urbana-Champange, 2000.

[335] M. Yazdi and A. Zaccarin. Scene break detection and classification using a block-wise difference method. In *Proc. IEEE ICIP '01*, pages III: 394–397, 2001.

[336] B. Yeo and B. Liu. Rapid scene analysis on compressed video. *IEEE Transactions on Circuits and Systems for Video Technology*, 5(6):533–544, December 1995.

[337] YesVideo Incorporated. `http://www.yesvideo.com/`.

[338] G. Young and R. Chellappa. Statistical analysis of inherent ambiguities in recovering 3-d motion from a noisy flow field. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(10):995–1013, October 1992.

[339] D. Yow, B.L.Yeo, M. Yeung, and G. Liu. Analysis and presentation of soccer highlights from digital video. In *Proc. ACCV, 1995*, December 1995.

[340] H. Yu, G. Bozdagi, and S. Harrington. Feature based hierarchical video segmentation. In *Proc. IEEE ICIP '97*, volume 2, pages 498–501, October 1997.

[341] H. Yu and W. Wolf. A multi-resolution video segmentation scheme for wipe transition identification. In *Proc., Int. Conf. on Acoustics, Speech, and Signal Processing*, pages 2965–2968, 1998.

[342] Y. Yusoff, W. Christmas, and J. Kittler. A study on automatic shot change detection. In *In Third European Conf. on Multimedia Applications, Services and Techniques*, volume 1425, pages 177–189. Springer, 1998.

[343] Y. Yusoff, W. Christmas, and J. Kittler. Video shot cut detection using adaptive thresholding. In *Proc. of The Eleventh British Machine Vision Conference (BMVC '00)*, pages 362–371, September 2000.

[344] R. Zabih, J. Miller, and K. Mai. A feature-based algorithm for detecting and classifying scene breaks. In *Proceedings of the third ACM international conference on Multimedia*, pages 189–200, 1995.

[345] R. Zabih, J. Miller, and K. Mai. A feature-based algorithm for detecting and classification of production effects. *ACM Multimedia Systems*, 7(1):119–128, January 1999.

[346] A. Zaccarin and B. Liu. Fast algorithms for block motion estimation. In *Proceedings of IEEE ICASSP*, volume 3, pages 449–452, 1992.

[347] C. Zahn. Graph-theoretic methods for detecting and describing gestalt clusters. *IEEE Transactions on Computing*, 20:68–86, 1971.

[348] D. Zhang and G. Lu. Segmentation of moving objects in image sequence: A review. *Circuits Systems Signal Processing*, 20(3):143–183, 2001.

[349] H. Zhang, A. Kankanhalli, and S. W. Smoliar. Automatic partitioning of full-motion video. *Multimedia Systems Journal*, 1(1):10–28, 1993.

[350] J. Zhang and J. Hanauer. The mean field theory for image motion estimation. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 5, pages 197–200, 1993.

[351] Z. Zhang. Parameter estimation techniques: A tutorial with application to conic fitting. *Image and Vision Computing Journal*, 15(1):59–76, 1997.

[352] D. Zhong and S.-F. Chang. Structure analysis of sports video using domain models. In *IEEE International Conference on Multimedia and Exposition*, August 2001.