3D Reconstruction of Sporting Activities from Multi-Camera Video

J.J.M. Kilner

Submitted for the Degree of
Doctor of Philosophy
from the
University of Surrey

Centre for Vision, Speech and Signal Processing
Faculty of Engineering and Physical Sciences
University of Surrey
Guildford, Surrey GU2 7XH, U.K.

May 2010

© J.J.M. Kilner 2010
Summary

3D Reconstruction using multiple cameras is a technique often employed in the studio production environment. The outdoor sports broadcast environment is also a multi-camera environment and this thesis attempts to apply and adapt studio techniques to the unique requirements of this environment, including increased errors in calibration and matting, variations in camera configuration and wide-baseline camera arrangements.

This thesis examines 3D reconstruction and activity recognition as applied to the domain of field sports such as football and rugby. Video is captured from a number of cameras arranged around the pitch, and processed using computer vision techniques to generate a 3D scene representation for rendering or analysis. The applications considered in this thesis are free-viewpoint video and the synthetic representation of segments of a match.

The thesis presents an analysis of the 3D reconstruction errors in the outdoor sports broadcast environment. Also presented are several Shape-From-Silhouette techniques which are shown to provide improved scene reconstruction in this environment. A dual-mode deformable model is presented that refines the reconstruction using stereo information from neighbouring cameras, and simultaneously optimises silhouette extraction in a manner that is more robust to the calibration and reconstruction errors typical of the outdoor sports broadcast scenario.

This thesis also presents an action matching technique to extract temporally consistent, synchronised pose information for each player present in the scene. An analysis of matching scores shows that the symmetric Kullback-Leibler divergence between shape histograms is a suitable score for measuring the difference between noisy visual hull reconstructions. Players are automatically segmented in 3D and a hidden Markov model is used to match extracted shape histograms against a library of exemplar poses. Various extensions to the scheme are presented and evaluated and it is shown that good results can be achieved using a combination of action matching and key-pose detection.

Key words: 3D Synthesis, Free-Viewpoint Video, Human Action Recognition, Pose Recognition.
Acknowledgements

This work was supported by the DTI technology programme project ‘iview: Free-viewpoint video for interactive entertainment production’ TP/3/DSM/6/1/15515 and EPSRC Grant EP/D033926, led by BBC Research and Development. More information can be found on the website at http://www.bbc.co.uk/rd/iview. I would like to thank the project partners for providing the sports footage and discussion of the research reported in this thesis.

Special thanks also go to all the members of CVSSP who have provided me with discussion, help and coffee when I needed them.

There are also a number of individuals to whom I must give particular thanks; Prof. Adrian Hilton, whose guidance and support has been invaluable over the course of this project, Dr. Jon Starck, without whom I might never have been able to turn my dream of pursuing a PhD into a reality, and my parents John and Carmen, and all my family, for their understanding and support.

And finally, special thanks go to my wife Holly who has supported me 100% since I first had the idea of handing in my notice and going back to university. Your help, support and feedback have been invaluable. I couldn’t have done this without you.
Contents

Mathematical Notation ix
  0.1 Symbolic Conventions ................................................................. ix
  0.2 Iverson Bracket Notation .............................................................. ix
  0.3 Representation of Multi-camera Data ........................................... ix
  0.4 Groups and Indicator Functions .................................................. x

Glossary of Abbreviations xi

1 Introduction 1
  1.1 Motivation ...................................................................................... 2
  1.2 Research Contributions ................................................................... 7
  1.3 Outline ............................................................................................ 7
  1.4 List of Publications ......................................................................... 8

2 Literature Review 11
  2.1 Free-viewpoint Video ................................................................... 11
    2.1.1 Applications in Sports ............................................................. 12
    2.1.2 Basic Principles ....................................................................... 13
    2.1.3 A Taxonomy of NVS Techniques ............................................. 15
    2.1.4 Applications to the Outdoor Sports Broadcast Environment ... 24
  2.2 3D Reconstruction Refinement ...................................................... 25
    2.2.1 Matting Techniques ................................................................ 26
    2.2.2 Shape Optimisation ................................................................. 27
  2.3 Modelling Human Activities .......................................................... 28
    2.3.1 Human Activity Detection ....................................................... 28
    2.3.2 Pose Detection ........................................................................ 30
  2.4 Summary .......................................................................................... 33
3 Error-Tolerant 3D Reconstruction 35
  3.1 Introduction ................................................................. 35
  3.2 Sources of Error .......................................................... 36  
    3.2.1 Calibration ........................................................... 36
    3.2.2 Matting ................................................................. 36
  3.3 Effects of Errors ............................................................ 37
  3.4 A Taxonomy of Errors .................................................. 40  
    3.4.1 Errors in Shape and Appearance .......................... 40
    3.4.2 Relationship to Real Errors ................................. 44
  3.5 Quantitative Evaluation ............................................... 45  
    3.5.1 Objective Quality Assessment ............................ 45
    3.5.2 Quality Metrics .................................................. 46
  3.6 Error-tolerant Visual Hull ............................................. 49  
    3.6.1 Visual Hull Techniques .......................................... 50
    3.6.2 Tri-state Visual Hull .............................................. 51
    3.6.3 Conservative Visual Hull ..................................... 53
    3.6.4 Consensus Hull .................................................. 54
    3.6.5 Combined Technique ........................................... 56
    3.6.6 Comparison with Probabilistic Hull ....................... 57
    3.6.7 3D Clean-up ....................................................... 60
  3.7 Quantitative Analysis of Techniques ............................. 60
  3.8 Summary ................................................................. 74

4 Shape Optimisation 77
  4.1 Introduction ............................................................... 77
  4.2 Surface Refinement for Free-viewpoint Video ................ 78  
    4.2.1 Free-viewpoint Video Rendering .......................... 78
    4.2.2 Surface Refinement .............................................. 80
  4.3 Dual-mode Deformable Model ...................................... 80
    4.3.1 Initialisation ....................................................... 82
    4.3.2 Fitting ............................................................... 88
  4.4 Results ................................................................. 88
  4.5 Discussion ............................................................... 91
  4.6 Summary ............................................................... 94
## Contents

<table>
<thead>
<tr>
<th>Chapter</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td><strong>3D Action Matching</strong></td>
<td>97</td>
</tr>
<tr>
<td></td>
<td>5.1 Introduction</td>
<td>97</td>
</tr>
<tr>
<td></td>
<td>5.2 Player Tracking</td>
<td>101</td>
</tr>
<tr>
<td></td>
<td>5.3 Shape Similarity</td>
<td>106</td>
</tr>
<tr>
<td></td>
<td>5.3.1 Feature Vector</td>
<td>106</td>
</tr>
<tr>
<td></td>
<td>5.3.2 Similarity Metric</td>
<td>107</td>
</tr>
<tr>
<td></td>
<td>5.3.3 Evaluation</td>
<td>108</td>
</tr>
<tr>
<td></td>
<td>5.4 Action Matching</td>
<td>114</td>
</tr>
<tr>
<td></td>
<td>5.4.1 Hidden Markov Model</td>
<td>114</td>
</tr>
<tr>
<td></td>
<td>5.5 Results</td>
<td>120</td>
</tr>
<tr>
<td></td>
<td>5.6 Discussion</td>
<td>122</td>
</tr>
<tr>
<td></td>
<td>5.7 Summary</td>
<td>126</td>
</tr>
<tr>
<td>6</td>
<td><strong>Action Matching: Extensions and Enhancements</strong></td>
<td>127</td>
</tr>
<tr>
<td></td>
<td>6.1 Introduction</td>
<td>127</td>
</tr>
<tr>
<td></td>
<td>6.2 Key-pose Detection</td>
<td>128</td>
</tr>
<tr>
<td></td>
<td>6.2.1 Results</td>
<td>131</td>
</tr>
<tr>
<td></td>
<td>6.3 Animation Summarisation</td>
<td>135</td>
</tr>
<tr>
<td></td>
<td>6.3.1 Results</td>
<td>138</td>
</tr>
<tr>
<td></td>
<td>6.4 Hierarchical Action Matching</td>
<td>143</td>
</tr>
<tr>
<td></td>
<td>6.4.1 Hierarchical Markov Model</td>
<td>146</td>
</tr>
<tr>
<td></td>
<td>6.4.2 Results</td>
<td>150</td>
</tr>
<tr>
<td></td>
<td>6.5 Summary</td>
<td>155</td>
</tr>
<tr>
<td>7</td>
<td><strong>Conclusions and Further Work</strong></td>
<td>157</td>
</tr>
<tr>
<td></td>
<td>7.1 Conclusions</td>
<td>157</td>
</tr>
<tr>
<td></td>
<td>7.2 Further Work</td>
<td>159</td>
</tr>
<tr>
<td>A</td>
<td><strong>Sports Capture Data Sets</strong></td>
<td>163</td>
</tr>
<tr>
<td></td>
<td>A.1 Synthetic</td>
<td>164</td>
</tr>
<tr>
<td></td>
<td>A.2 Synthetic Error</td>
<td>164</td>
</tr>
<tr>
<td></td>
<td>A.3 Football 1</td>
<td>164</td>
</tr>
<tr>
<td></td>
<td>A.4 Football 2</td>
<td>164</td>
</tr>
</tbody>
</table>
### Contents

A.5 Football 3 ................................................................. 164  
A.6 Rugby 1 ................................................................. 165  
A.7 Rugby 2 ................................................................. 165  
A.8 Sample Images ...................................................... 165  

**B** Motion Capture Library  
B.1 Sources .............................................................. 173  
B.2 Action Library ....................................................... 173  

**C** View-dependent Texturing Algorithm and Application  
C.1 The Workbench ..................................................... 179  

Bibliography ............................................................. 183
Mathematical Notation

0.1 Symbolic Conventions

The following typographical conventions are used throughout the text:

- \( \mathbb{R} \) the set of real numbers
- \( \mathcal{X} \) a set (in calligraphic capitals)
- \( \mathbf{x} \) a vector (in bold type)
- \( \hat{x} \) a unit vector in the direction of \( x \)

0.2 Iverson Bracket Notation

Iverson bracket notation will be used to define functions which may return 0 or 1, such that:

\[
f(x) = [x \geq 0]
\]

is exactly equivalent to the expression:

\[
f(x) = \begin{cases} 
1 & x \geq 0 \\
0 & x < 0 
\end{cases}
\]

0.3 Representation of Multi-camera Data

A multi-camera video \( \mathcal{S} \) consists of an ordered set of videos \( \{\mathcal{V}_0, \mathcal{V}_1, \ldots, \mathcal{V}_m\} \)

A video consists of an ordered set of images: \( \mathcal{V} = \{I_0, I_1, \ldots, I_m\} \)

An image consists of an ordered set of pixels: \( \mathcal{J} = \{p_0, p_1, \ldots, p_m\} \)

A pixel is a tuple consisting of a location and a colour value: \( p = (x, c) \).

A pixel location is a vector in \( \mathbb{R}^2 \). The location of a pixel is indicated by \( l(p) = x \)

A pixel colour is a vector in \( \mathbb{R}^3 \) on the interval \((0 : 1, 0 : 1, 0 : 1)\). The colour of a pixel is indicated by \( c(p) = c \)

Pixels will generally be indicated by symbols \( p \) or \( q \).
0.4 Groups and Indicator Functions

Sets of pixels and voxels will often be defined in terms of an indicator function. The same symbol is used to represent both the set and the indicator function for that set. So if $\mathcal{X}$ is a set of pixels, then $\mathcal{X}(p)$ is a function over the set of all pixels, such that:

$$X(p) = [p \in \mathcal{X}]$$

Similarly, if $\mathcal{Y}(v)$ is a function over a set of voxels that returns a scalar value, then the set $\mathcal{Y}$ is defined by:

$$v \in \mathcal{Y} \iff \mathcal{Y}(v) \neq 0$$
$$v \not\in \mathcal{Y} \iff \mathcal{Y}(v) = 0$$
Glossary of Abbreviations

CCH - consensus conservative hull (page 56)
CH - consensus hull (page 55)
CVH - conservative visual hull (page 54)
FVV - free-viewpoint video (page 11)
GLSL - OpenGL shading language [34]
HD - high definition
HMM - hidden Markov model (page 100)
HVS - human visual system
IBR - image based rendering (page 15)
KLD - Kullback Leibler divergence (page 107)
NVS - novel view synthesis (page 11)
OpenGL - Open Graphics Library [34]
PSNR - peak signal to noise ratio (page 46)
RGB - red green and blue
ROC - receiver operating characteristic (page 113)
SD - standard definition
SFS - shape-from-silhouette (page 35)
SIFT - scale-invariant feature transform [66]
VH - visual hull (page 21)
VIF - visual information fidelity (page 48)
Chapter 1

Introduction

The use of 3D data in the entertainment industry ranges from providing special effects in films and computer games, to rendering statistical charts for news programmes. The rendering of high quality 3D graphics is an industrialised process, with advanced rendering hardware available on any high street in the form of graphics cards or games consoles.

While the display of 3D graphics is a fully automated process, the generation of the content to be displayed is a complex, manual process often requiring large teams of skilled artists. The field of computer vision and specifically the topic of 3D reconstruction is attempting to bring data acquisition to the same level of automation as has been obtained with capture and display, by attempting to extract all the information required to generate a 3D model from a set of images.

This thesis applies techniques from computer vision to images obtained at multi-camera outdoor sports broadcasts in an attempt to generate a 3D representation of the recorded sporting events. Several techniques are presented and solutions suitable for use in this environment are evaluated, considering the general problems of natural image matting, error-tolerant reconstruction and human action recognition.
Figure 1.1: *Image of a football match taken using Skycam, a computer-controlled camera suspended from cables. (Image courtesy of BBC R\&D.)*

1.1 Motivation

The recording and broadcasting of sports footage is a multi-billion dollar international industry, with rights to live coverage of a single season of Premiership football costing around £1.7 billion [25]. There is intense competition between broadcasters seeking to improve market share, increase advertising and sponsorship revenue and improve their chances of securing the lucrative rights to broadcast the major leagues and competitions. The actual sporting event itself is out of the hands of the broadcast companies, so competition between broadcasters focuses on how they can add value to their programmes through expert commentary or the development of new technologies.

The use of innovative technology in the sports broadcast arena is fuelled by two main objectives; giving the viewer the best possible view of the action on the pitch, and enhancing the footage with annotations.

In attempting to provide novel views of the match, the broadcaster aims to give the viewer access to viewpoints they could never experience even from the best seats in the stadium. This objective has driven a great deal of development in the field of
1.1. Motivation

camera technology, including high-speed cameras which provide slow-motion footage of the match, robotic cameras such as "Skycam" (a robotic camera suspended by wires over the pitch, see Figure 1.1) and miniature "hardened" cameras which can be placed inside cricket stumps or on goalposts without interfering with play or being damaged by the inevitable impacts sustained during a match.

The use of annotations encompasses a wide range of aims, from allowing expert commentators to highlight interesting tactical and strategic events during a game, to showing statistics or projecting sponsors' logos onto the pitch (see Figure 1.2). These technologies began with the invention of the Telestrator, a device for drawing lines over still frames of footage, allowing experts to highlight lines of sight etc. (see Figure 1.3), but have now developed into complex augmented reality applications blending real and virtual elements, allowing perspective-correct annotations to be displayed on the pitch and even permitting players to be moved around the screen.

There are, however, limitations to progress on these fronts using purely traditional engineering. New camera equipment is expensive to develop, and constraints on deployment (the equipment must not interfere with the sportsmen or obstruct the view of paying spectators) mean that there is a limit to what can be done with physical cameras. Also, multiple cameras are expensive and so are the operators required to man them, placing financial constraints on the amount that camera technology can be

Figure 1.2: An example of the more advanced techniques possible using systems such as Piero, where translucent team logos are projected onto the pitch in the view from a moving camera. (Image courtesy of BBC R&D.)
used to augment coverage of anything but the most high profile sporting events. Even with large numbers of cameras, it is impossible to predict the exact location on the pitch where the most interesting events of the game will take place, so broadcasters often find themselves in the position where the recorded view of an event does not clearly show what happened.

Similarly, while perspective-corrected annotations for a single image can be easily created by hand, generating and maintaining perspective-correct annotations across a video sequence with a moving camera, or across multiple views of a scene, is a laborious task and quickly becomes prohibitively expensive and slow without extensive automation.

It is into this environment that the field of computer vision is starting to make an appearance. From pioneering work such as Takeo Kanade's Eye-Vision being used at the Superbowl[101] to the Piero system developed by the BBC/Red Bee Media[7](see Figure 1.4), the Hawkeye system used extensively in tennis and cricket[67](see Figure 1.5), and more recent commercial systems such as LiberoVision[31], computer vision systems are bringing a new versatility and depth to the enhancement of sports coverage.

By interpreting images from the multiple cameras viewing an event as the projections of a three-dimensional world into the image planes of a set of cameras, computer vision techniques can be used to model the 3D world described by the set of images.
1.1. Motivation

Figure 1.4: The Piero system is extensively used by sports broadcasters to enhance coverage of sporting events with perspective-corrected annotations. (Image courtesy of BBC R&D.)

Figure 1.5: The Hawkeye system has found extensive use in both cricket and tennis coverage giving advanced statistics and match analysis using computer vision techniques. (Image courtesy of Hawkeye Ltd.)
Chapter 1. Introduction

Figure 1.6: The Virtual Replay system used by BBC Sport to allow viewers to interactively explore on-pitch events. This consists of a synthetic model of the action generated by hand to match the recorded footage. (Image courtesy of BBC Sport.)

This description can then be used to generate new views outside the range of the original cameras, and to blend real and synthetic data seamlessly, providing both novel viewpoints and sophisticated annotations.

Computer vision techniques can also be used to automatically interpret the action on the pitch by providing a high level description of the players and activities recorded. Use of pose detection and activity models can simplify the currently manual process of generating a synthetic representation of the match. This can then be used to generate a virtual replay in which sophisticated annotations can be displayed, and events may be altered to present "what if?" scenarios, similar to the output of the Virtual Replay system used by BBC Sport (see Figure 1.6).

While the field of computer vision contains much work in the area of 3D reconstruction from images, this has mainly focused on the studio environment. It is an attempt to adapt these techniques to the needs of the outdoor sports broadcast environment that provides the motivation for the work presented in this thesis. This thesis is based on research carried out as part of the iview project, funded by the DTI Technology Programme, in collaboration with the BBC, Snell & Wilcox and Hawkeye.

This thesis will present several novel enhancements in the field of 3D reconstruction as
applied to the sports broadcast scenario, culminating in the introduction of the first fully automatic system for the simultaneous recovery of pose for multiple people in a sports environment.

1.2 Research Contributions

The following research contributions are presented in this thesis:

- A technique for the quantitative analysis of reconstruction errors that is suitable for use in the outdoor sports broadcast reconstruction environment.
- Several extensions to shape-from-silhouette suitable for use in the presence of matting and calibration errors typical of multi-view reconstruction in the outdoor sports broadcast environment.
- A deformable model technique that is less prone to the parameterisation issues common in the outdoor sports environment.
- An evaluation of shape histogram matching scores for pose matching in noisy environments.
- Several extensions to pose matching via shape histograms for use in noisy environments.
- A full system capable of estimating the pose of multiple people simultaneously in the outdoor sports broadcast environment.

1.3 Outline

The first half of the thesis concentrates on the retrieval of 3D geometry from images. Chapter 2 consists of a review of the field of 3D reconstruction, with particular reference to the outdoor sports environment. Chapter 3 introduces an analysis of the errors in the reconstruction pipeline and Chapter 4 applies the results of this analysis to a shape optimisation strategy.
Chapter 1. Introduction

The second half of the thesis concentrates on pose estimation from images in the outdoor sports domain. Chapter 5 presents a novel scheme for extracting pose information from a 3D reconstruction and Chapter 6 proposes some extensions to this framework. Chapter 7 concludes the thesis and presents some notes on further work required on this topic.

1.4 List of Publications

The following publications were derived either directly or indirectly from work carried out in pursuit of this thesis.

2010

Free-viewpoint Video for TV Sport Production

2009

3D Action Matching with Key-pose Detection
[J. Kilner, J.-Y. Guillemaut and A.Hilton]
Proceedings of the Twelfth IEEE International Conference on Computer Vision: IEEE Workshop on Search in 3D and Video

Summarised Hierarchical Markov Models for Speed Invariant Action Matching
[J. Kilner, J.-Y. Guillemaut and A.Hilton]
Proceedings of the Twelfth IEEE International Conference on Computer Vision: The Second IEEE International Workshop on Tracking Humans for the Evaluation of their Motion in Image Sequences

Robust Graph-cut Scene Segmentation and Reconstruction for Free-viewpoint Video of Complex Dynamic Scenes
[J.-Y. Guillemaut, J.Kilner and A.Hilton]
Proceedings of the Twelfth IEEE International Conference on Computer Vision
1.4. List of Publications

Objective Quality Assessment in Free-viewpoint Video Production
Signal Processing: Image Communication, Volume 24, Issues 1-2, Pages 3-16
2008

Objective Quality Assessment in Free-viewpoint Video Production
[J. Starck, J. Kilner and A. Hilton]
Proceedings of 3DTV Conference 2008
2007

Dynamic Feathering: Minimising Blending Artefacts in View-dependent Rendering
Proceedings of the Fourth European Conference On Visual Media Production

A Robust Free-viewpoint Video System for Sport Scenes
[O. Grau, G. Thomas, A. Hilton, J. Kilner and J. Starck]
Proceedings of 3DTV Conference 2007

A Free-viewpoint Video System for Visualization of Sport Scenes
[O. Grau, A. Hilton, J. Kilner, G. Miller and J. Starck]
SMPTE Motion Imaging Journal, Volume 116, Issues 5-6, Pages 213-219

A Bayesian Framework for Simultaneous Matting and 3D Reconstruction
Proceedings of the Sixth International Conference on 3-D Digital Imaging and Modeling

Dual-mode Deformable Models for Free-viewpoint Video of Sports Events
Proceedings of the Sixth International Conference on 3-D Digital Imaging and Modeling
2006

A Comparative Study of Free-viewpoint Video Techniques for Sports Events
[J. Kilner, J. Starck and A. Hilton]
Proceedings of the Third European Conference on Visual Media Production
A Free-viewpoint Video System for Visualisation of Sport Scenes


Proceedings of the International Broadcasting Convention
Chapter 2

Literature Review

This chapter will introduce the basic principles of free-viewpoint video (FVV) and its application to the outdoor sports broadcast environment. It will then go on to examine state-of-the-art techniques for 3D reconstruction refinement and introduce the topic of action recognition and pose detection.

2.1 Free-viewpoint Video

When traditional fixed-viewpoint video of an event is rendered, the only viewpoint available for playback is that of the camera that recorded the event. FVV attempts to break this restriction by allowing the specification of the viewpoint at the point of rendering rather than at the point of recording.

FVV is simply an extension into the time domain of the older and more studied problem of novel view synthesis (NVS). NVS covers a broad set of computer vision techniques which have been developed to attempt to solve the problem of generating a novel view from a set of measurements of a scene (typically a set of images). Thus NVS is fundamentally concerned with two things; determining correspondences between images and interpolating or extrapolating from these correspondences to form a new image. Where NVS deals with images, FVV deals with image sequences, and so FVV typically consists of a frame-by-frame application of an image-based NVS technique to sequences of images from multiple cameras.
2.1.1 Applications in Sports

Sports often rely on the movement of contestants within a delineated playing area. Cameras are set up to try to get a direct view of the boundaries of these areas, as action there is often crucial to the game. However, as broadcasters typically have a limited number of cameras, they must try to pre-empt the locations of interesting events. If they get this wrong they risk missing out on crucial footage and wasting a camera that could have provided another angle for general coverage.

Similarly, a great deal of tactical interest in sport relies on the spatial configuration of the players, such as the relationship that determines whether an individual is offside in football. These relationships are often very difficult to determine accurately unless a camera is specifically placed to highlight the desired configuration. As the configuration of the players may become interesting at an arbitrary time and location in the playing area, it is impossible to place cameras to guarantee a good viewpoint. As a result it can often be very difficult, if not impossible, to work out the spatial relationship of the contestants from the broadcast coverage of a sporting event.

Finally, with broadcasters competing for share of the large audiences that big sporting events command, there is a demand for special effects such as the “Matrix-style” camera transitions used during the Super Bowl[101]. Given these considerations, and the fact that the location of cameras may be further constrained by the shape of the arena the event takes place in, crowd seating, etc., it can be seen that an FVV system suitable for use in sporting events is highly desirable, as it allows broadcasters to generate the output they want without having to pre-empt the correct camera locations for every possible eventuality.

Advances in miniaturisation and wireless technology have allowed some novel solutions to this problem, including suspending cameras from wires and fitting micro-cameras to various bits of sporting apparatus, however none of these provide the required flexibility and most fall short in terms of quality. Most of the current techniques for FVV are designed around a multi-camera studio environment with controlled lighting and well-calibrated static cameras, meaning they also fail to perform at an acceptable quality in the context of outdoor sports coverage with unconstrained illumination and poor
2.1. Free-viewpoint Video

2.1.2 Basic Principles

The basic principal underlying all NVS techniques is that of the plenoptic function, introduced in the commonly seen 5-dimensional form by McMillan and Bishop[70]. This states that a single function describes all possible images of a scene:

\[ c = p(x, y, z, \theta, \phi) \]  

(2.1)

where \( x, y, z \) is a position in \( \mathbb{R}^3 \), a direction is given by \( \theta, \phi \) as azimuth and elevation, \( c \) is the colour viewed from that location in the specified direction and \( p \) is the plenoptic function (see Figure 2.1). An image is then a discrete sample of \( p \) for a given value of \( x, y, z \), with each pixel being the integral over a small range of \( \theta \) and \( \phi \). The problem of NVS can then be phrased as an attempt to generate a continuous representation of
Figure 2.2: A set of pixels related by being measurements of light emitted by a patch \( s \) on surface \( S \). \( p \) is a real pixel on real image \( J \) and \( p' \) is a synthesised pixel on synthetic image \( J' \).

\( S \) is the set of real images.

the entire plenoptic function from a small set of discrete samples. By adding a term \( t \) to indicate the time of sampling, the 6-dimensional plenoptic function can be obtained to describe dynamic scenes.

\[ c = p(x, y, z, \theta, \phi, t) \] (2.2)

In this way, the problem of NVS can be trivially extended to FVV.

A second principle that is often relied upon is that of image formation by objects; specifically that a region of an image is formed by light reflecting off a patch \( s \) on a surface \( S \), as shown in Figure 2.2. By determining the properties of \( s \), and with a knowledge of the camera calibration and geometry, the pixels relating to \( s \) can be determined across a set of images \( G \). \( S \) is often referred to as the “scene geometry”.

Often two simplifying assumptions are made; 1) that \( S \) is locally smooth and 2) that the reflectance characteristics of \( S \) are Lambertian i.e. that the reflectance characteristics are isotropic.
2.1.3 A Taxonomy of NVS Techniques

NVS techniques are often divided into image-based rendering (IBR) techniques and model-based techniques, the distinction being that an IBR technique does not maintain any representation of the scene other than the source images, whereas a model-based technique generates a model of the scene from the images and uses that to render new views. This distinction does not necessarily tell us much about the underlying logical structure of the algorithm, in fact many techniques can be implemented in both image-based and model-based manners, such as estimation of the visual hull[69, 74]. Therefore, this taxonomy will instead look at the criteria for building the sets of correlated pixels that are used to synthesise new images.

Any FVV technique consists of three steps:

1. Determine some correspondence between pixels in the various input images.

2. Describe a relationship between the novel image to be generated and the original images.

3. Calculate novel image values based on previous steps.

For example, a technique may first determine the pixels in each image that show a particular point on a surface using the known camera and scene geometry. It may then use these pixels to generate a view-dependent model of the appearance of the known surface. Finally, it may blend these pixels dependent on parameters such as the location of the desired synthetic view. This taxonomy will focus on the criteria used in the first step to determine correlation between pixels in different images.

Correlation between pixels can be sought in one of two domains; the pixel value domain or the geometric domain. When seeking correlation in the pixel value domain, the only consideration is the pixel values and their local structure. Techniques that seek correlation in this way look for pixels with similar values and similar local structure. When seeking correlation in the geometric domain, the only consideration is the geometric relationship between pixels. Techniques that seek correlation in this way apply geometric projections from image to image to build up sets of correlated pixels.
Geometric relationships can vary from those dependent solely on the projective relationship between the cameras (epipolar geometry) to those that also rely on the scene structure (scene geometry). Techniques reliant mainly on epipolar geometry are making the basic assumption that the plenoptic function is locally smooth and so interpolation between similar rays is a meaningful operation. Techniques relying on scene geometry assume that a 2D manifold representing the geometry of the scene can be obtained with sufficient accuracy.

In fact, almost all techniques are some combination of these extremes, either using an approximate geometry to constrain the search for correspondence in the pixel value domain, or using correspondence in the pixel value domain to aid the refinement of scene geometry. Figure 2.3 shows a selection of techniques, plotted to demonstrate how different interpretations of the image data are used to synthesise a novel view.
2.1. Free-viewpoint Video

Correspondence from Pixel Values

The purest examples of techniques deriving correspondence from pixel values are image-feature-based techniques such as those used for object detection and for bootstrapping camera calibration. These look for distinctive features in an image and describe them in such a way that they can be matched against similar features in other images. A commonly-used example is the SIFT feature descriptor introduced by Lowe [66]. Obviously techniques for bootstrapping the calibration process cannot rely on the camera geometry, and the sparse features generated do not generally rely on the image geometry either.

Optic flow also relies on calculating correspondence purely from pixel values. Using nothing but the value of a pixel and its relationship to its neighbours, optic flow seeks to determine correspondences between pixels in two images. This is generally used for the purposes of temporal interpolation rather than spatial interpolation. Also, these techniques rely on the image geometry to some extent, as they use spatial coherence within the image domain to filter their results.

Another technique involving only information from the pixel value domain is the formation of image mosaics as used by Szeliski and Shum[97]. Mosaicing is the stitching together of the edges of several images so that a larger, panoramic image can be constructed. New images can then be generated by sampling a section of the larger mosaic. This technique works well for images that are taken with widely varying viewing angles but the same view location (i.e. rotating a camera about a fixed pivot). However, the novel image must also share or be close to this location, a constraint which is not satisfied in the general NVS scenario.

Correspondence from Pixel Values and Epipolar Geometry

Epipolar geometry is the relationship between a pair of cameras where a point in the image of one camera projects to a line in the image of another camera (see Figure 2.4). This constraint is used extensively in stereo matching techniques as it drastically reduces the complexity of the problem. Also, the constraint can be accurately evaluated
for a pair of cameras without requiring full calibration of the pair (so called “weak calibration”). A full description of the geometric relationships between cameras can be found in Hartley and Zisserman [37].

The fundamental assumption of all stereo techniques is that a surface patch at a certain distance (or depth) from a pair of cameras will appear similar in both cameras, although at a different location in the images. Stereo techniques then involve searching a pair of images to find the optimum matches between image patches. This search is constrained by the epipolar relationship between the images, and the distance in image space between the matching patches is referred to as a disparity. Disparity is negatively correlated with the depth of the surface to which the matching pixels relate. In this way a stereo technique produces a set of per-pixel depth values which best explain the scene. A selection of techniques is compared and contrasted by Scharstein et al. [87].

As stereo looks at the local structure around a pixel, it must assume that the local structure is invariant between images. In order for this to hold true, occlusions and projective distortion must be minimised. Also, for pixel values to be consistent between images, a Lambertian reflectance model is assumed. In order to meet these requirements, a stereo
2.1. Free-viewpoint Video

data set is typically captured along a narrow baseline, although some attempt is made
to correct for intensity variation by using normalised[87] or illumination-invariant[41]
matching scores, and more recent stereo-based techniques attempt to explicitly model
occlusions[96]. All stereo techniques require the presence of some non-uniformity in the
appearance of a surface in order to make unique matches.

The depth images generated using stereo techniques can be projected into 3D using the
camera geometry to generate a model which can be rendered using standard computer
graphics techniques to produce the desired novel viewpoint. Alternatively, the new
view can be parameterised as a linear combination of the original views, and pixels
may simply be interpolated based on disparity as in Werner et al. [105].

Stereo has some specific limitations which affect its usefulness in the field of outdoor
sports reconstruction. Firstly, stereo techniques are not easily extended from two cam­
ers to n cameras. In order to relate multiple images, some knowledge of the camera
gometry must be introduced, as in the techniques considered by Seitz et al. [88]. Sec­
ondly, it requires some overlap between images - for example, images of different faces
of a cube cannot be used with a stereo technique, even if the camera geometry is known,
as there are no common image patches to find. Finally, it requires local structure to be
compared between images; data sets with images at greatly varying resolutions, where
sampling artifacts and aliasing are major considerations, will not produce good results.

The epipolar constraint can be extended to three images using the trifocal tensor. This
leads to a set of techniques known as trifocal transfer, which are contrasted with simple
epipo lar techniques by Reid and Connor[83]. These techniques suffer from the same
problems as traditional stereo, namely poor handling of occlusions and non-Lambertian
surfaces.

Ray casting techniques as developed by Irani et al. [46] use epipolar constraints along
with the ordering constraint to find surfaces in an implicit geometric model. This
technique works well with unoccluded images of several objects with distinct colours,
but does not handle inter-object occlusion well. The technique also relies on multiple
correspondences in order to be robust and therefore requires a large number of cameras
(10 or more) along a narrow baseline. A Bayesian extension has been proposed by
Fitzgibbon et al. [28] which also uses image priors to attempt to resolve ambiguities in reconstruction. This works well, but still requires a large number of cameras viewing each point to be effective. Both techniques have only been demonstrated with narrow baseline capture.

**Correspondence from Pixel Values and Scene Geometry**

This category consists of techniques that attempt to build up an explicit model of the scene geometry using the assumption that this geometry exhibits Lambertian reflectance characteristics and is locally smooth. This assumption can then be used to find pixels of similar colours in all cameras and relate them to a point in space of that same colour. With reference to Figure 2.2, the assumption is that $S$ is smooth and $p$ is the same colour in all images in $M$.

A technique similar to the ray casting technique is that of space carving, as introduced by Kutulakos and Seitz[61]. This technique uses a voxel-based approach to solve the photo-consistency problem in the general case, using an explicit geometric model. It determines a surface known as the photo hull, which is the maximal shape that encloses all photo-consistent reconstructions. This technique relies on a Lambertian surface model and is sensitive to camera calibration errors. Also, in regions where the images are not photo-consistent, the volume will be over-carved. Even with photo-consistent images, the technique only generates a surface that encloses the true surface. Finally, like any voxel-based technique, it is sensitive to aliasing due to the resolution of the voxel grid, and sampling error due to the size of the voxel grid in relation to the object.

A simple extension of stereo is to posit a scene geometry which consists of a set of planar surfaces parallel to the image plane of the camera. Reconstruction is then a case of optimising the depth and occupancy of these planes to maximise stereo matching. An example of this technique is given by Connor and Reid[13, 84] who give each layer an aspect (an appearance from a certain direction) and an occupancy. The presence of an aspect per source image allows non Lambertian surface effects to be represented, and the use of occupancy (rather than simply visibility) allows the optimisation of layers over time to apply temporal consistency constraints to the problem and use a "last
2.1. Free-viewpoint Video

seen” approach to fill holes in the synthesised view. This technique relies on dense correspondence to identify the implicit scene geometry and has only been applied to pairs of cameras. Similar work in reconstructing multiple depth layers from monocular image sequences can be found in work by Jojic and Frey [48] who introduce deformable “flexible sprites” to describe scene elements, and in work by Torr et al. [100] who present a Bayesian framework for decomposing a scene into a piecewise planar representation. A related multi-camera technique that attempts to deal correctly with occupancy and surface colour at boundaries between layers can be found in Zitnick et al. [108]. This technique requires a high number of cameras arranged along a narrow baseline, but is of note as it has been specifically developed for FVV and produces high quality results.

A significant technique in this field is shape-from-silhouette, which attempts to estimate the visual hull as described by Laurentini [63]. The visual hull (VH) is the maximal volume that is consistent with the silhouettes generated by the source images. This technique can be seen as defining a subspace of $\mathbb{R}^3$ within which all interesting correspondences must occur. An image is segmented into regions of interest $F$ (foreground) and regions to be discarded $B$ (background). It therefore follows that any volume of $\mathbb{R}^3$ that is of interest to the reconstruction must project to $F$ in all cameras. This technique is extremely powerful as it can set an upper bound on the region to be reconstructed, even in the absence of any overlap between the input images, and is often used as a primary step in the NVS pipeline.

Calculating the visual hull can be done in a volumetric fashion using a voxel grid, as shown by Moezzi et al. [74], or in an image-based manner by projecting intervals from the desired ray onto the source silhouettes, as shown by Matusik et al. [69]. Visual hulls (like photo hulls) are very sensitive to camera calibration errors, as incorrect epipolar geometry can lead to a large decrease in correspondence and hence a large decrease in the volume of the visual hull. To counter this, probabilistic volumetric frameworks have been used, such as that developed by Franco et al. [29], which treat voxel occupancy as a likelihood rather than a hard constraint.

All visual hull techniques are susceptible to masking and camera calibration errors. Due to the global nature of the hull, a single bad mask or badly calibrated camera
can cull large sections of the model. As such, carefully calibrated environments\cite{74} or complex methods to account for the errors\cite{29} are required. Even in the presence of correctly calibrated cameras, errors are still possible, as volumes in the scene may be occluded in all cameras without being occupied. These issues can cause phantom volumes and protrusions from the surface. Such problems are often removed by using stereo refinement of the surface.

Recent work on stereo refinement of the visual hull includes Hernandez and Schmitt\cite{40} and Starck et al. \cite{92}, who use energy minimisation over deformable surfaces to generate a globally optimised solution. Also Miller et al. \cite{72}, who use iterative per-pixel refinement for a view-dependent solution. These techniques can provide high quality results, but rely on good image segmentation for the generation of an accurate visual hull, and like all stereo techniques rely on Lambertian assumptions and require a non-uniform surface in order to find matches. However, the use of a visual hull as an initial approximation of the scene geometry significantly reduces the requirement for narrow baseline cameras.

**Correspondence from Epipolar Geometry**

Techniques which rely solely on epipolar geometry include lightfield rendering, as presented by Levoy and Hanrahan\cite{64}. This uses a geometric interpretation of the relationship between the source images to obtain correspondence between pixels, however it does not contain any information about the specifics of the scene geometry. This technique relies heavily on dense sampling, and the synthetic views that can be produced are constrained to lie close to the original camera locations.

A similar technique developed by Shum and He is that of concentric mosaics\cite{91}. This samples the plenoptic function using a slit camera that travels on a plane in concentric circles. While this technique uses only one camera, it produces restricted viewing angles for synthesised images and requires a specialist camera to be placed at the centre of the scene. Also, the technique for capture does not lend itself to FVV, as frames cannot be captured at a high enough rate.
2.1. Free-viewpoint Video

Correspondence from Approximate Scene Geometry

An extension of the lightfield to include a very simple proxy geometry is the lumigraph as presented by Gortler et al. [33]. This technique uses a bounding box to approximate the scene geometry and hence place some local support restrictions on the scene rays generated by the source images.

Reconstruction using view-dependent texture mapping and proxy geometry is demonstrated by Debevec et al. [17]. However, the proxy requires user-specified correspondences, limiting this technique to simple geometric surfaces. Also, where features are not modelled by the user they are represented as changes in the surface of the object. This can work effectively when these features are considerably smaller than the modelled geometry, but as the two approach each other in scale then errors grow and misalignments between different views become apparent. More recently, this technique has been unified with the lumigraph work by Buehler et al. [10], allowing a family of techniques that vary in their dependence on dense sampling and a priori knowledge of the scene geometry.

Correspondence from Exact Scene Geometry

If the camera geometry and the surface geometry of the object of interest are known precisely, an accurate model of the entire plenoptic function can be obtained with just a few images. Techniques for determining surface geometry include laser range scanning, where a laser pulse is reflected off an object to measure the distance to the surface at various points, and structured light techniques, where a known pattern is shone on an object and surface structure is determined by the observed distortion of the pattern. These techniques can produce high quality geometry which can then be rendered using a texture, or multiple textures, captured at the same time. These techniques however require fine control of the illumination, precisely-calibrated cameras and light sources, and specific custom equipment. Generating video-rate capture with these techniques is difficult as they require multiple illumination sources which must be interlaced or otherwise combined, such as in recent work presented by Xu and Aliaga[107]
2.1.4 Applications to the Outdoor Sports Broadcast Environment

Most NVS techniques are focused on generating video from an object in a studio environment. However some work has been applied specifically to capture of outdoor sporting events. The most famous application of FVV in the sporting arena is the Eye Vision project of Kanade et al. [101]. This involved driving 30 robotically-operated cameras off of one human-operated master camera so that they were all viewing the same section of the pitch, allowing a time-frozen rotation of the viewpoint around a point in the scene. As this technique involves switching between cameras, the viewpoint change is jerky and the nature of the technique means that a human operator must choose the point of interest at recording time. Similarly, while the viewpoint is mobile, no synthetic views are used and hence the viewpoint is not truly free. The integration of "Virtualized Reality" techniques[50] into this framework would generate a true FVV system, although still constrained to focus on the point chosen by the human operator at the point of recording.

Inamoto and Saito[44] have developed a system for augmented reality display of football footage. Their system breaks down images into a field region, background region and dynamic region. The background region is modelled using a panorama generated by image mosaicing, and the field region is generated by a homographic projection of all source images onto a plane. Both background and field are rendered using images from an empty stadium. Players are then segmented and projected onto the ground plane. Epipolar correspondence is used to morph the images between views, a technique which requires images from a narrow baseline but only weak camera calibration. Players are then rendered as billboards against the background and field regions, or superimposed on a scale model in an augmented reality environment. While this technique works in simple environments, the lack of a true 3D model means that internal occlusions and player-player occlusions are not handled. The use of billboards and image morphing also means that synthetic views are restricted to being close to the original views, and the use of static backgrounds limits the realism of the generated images – the pitch will not change accurately as the game progresses, there are no shadows, and lighting is static over the course of a game.
2.2. 3D Reconstruction Refinement

The multiple layers technique described in Connor and Reid[13] has also been applied to a football match, but suffers from the previously mentioned limitations of that technique, namely requirement for a narrow baseline and limited movement of the viewing angle. Finally, Koyama et al. [58] have proposed a simplified geometric representation using planar billboards for real-time view synthesis. This requires complex equipment and a birds-eye-view camera to accurately locate players. The system also has several limitations – it cannot accurately reconstruct the ball (as it relies on all objects being on the ground plane) and it cannot resolve player-player occlusions.

Billboards are a piecewise planar representation of the scene geometry. A more sophisticated technique that has been applied in this field is the piecewise planar+depth representation which allows local variation in the geometry of the billboard, based on the refinement of a per-camera depth image. This technique was applied to the sports environment by Guillemaut et al. [36].

Commercial systems which can perform limited NVS in the sports domain are already available. The Piero[7] system developed by the BBC and Red Bee Media allows the semi-automatic generation of 2D player billboards which can then be rendered to virtual viewpoints to generate novel synthetic views. As the stadium is represented by a synthetic model and the players are represented by flat 2D billboards, the realism of these virtual views is quite limited. A recent and more sophisticated product is the Liberovision[31] system. This uses a more advanced billboard technique which includes a surface displacement to model the appearance of players more faithfully. Liberovision can use this model to transition between pairs of cameras while rendering the background from the source images, providing relatively realistic synthetic views. However, both systems currently only operate on static images and are unable to generate FVV.

2.2 3D Reconstruction Refinement

Even in the constrained environment of the 3D capture studio, the shape generated by 3D reconstruction can often be of insufficient quality. As such, it is common to refine both the inputs and the outputs of the reconstruction process as part of the full
reconstruction pipeline. Common refinements include matting refinement and shape optimisation.

2.2.1 Matting Techniques

Many of the techniques mentioned rely on accurate segmentation of foreground regions in the source images, either for use in the reconstruction algorithm or for use as texture maps in the final rendering. The generation of maps which identify pixels as either foreground, background or mixed is known as matting or keying. The simplest matting technique is that of chroma-keying. This involves partitioning pixels into foreground and background sets based on the colour space distance between a pixel and the estimated background and foreground colours. This technique does not perform well other than in very constrained environments (such as in a blue-screen studio) and will fail if foreground and background colours are similar in any region of the image. Another technique is that of background subtraction, where a pixel is compared to the pixel value from a known background image. If the value is similar then it is background, if not then it is foreground. This also requires a constrained environment, as it will fail if the background changes (due to gross illumination changes or genuine movement of the background) or if background and foreground colours are similar in a specific region.

In many applications, only a single image is available with an unconstrained background (natural image matting) and as a result most modern techniques start with a user-supplied coarse segmentation and refine this. The segmentation is typically initialised as a trimap, which is a map assigning the pixels in an image into three groups – known foreground, known background and “unknown”. The boundary is then found using information from the foreground and background regions. These include using local image statistics (as in Chuang et al.’s Bayesian matting[12]), using a smoothing function and the image gradient (as in Sun et al.’s Poisson matting[95]) or energy minimisation using graph cuts (as in Rother et al. [86] and more recently in Juan and Keriven[49]). While these techniques can give good results, they all rely on user intervention, with techniques such as Poisson matting being particularly sensitive to the trimap specification. Recently Hasinoff et al. [38] have attempted to use 3D scene
information to perform matting. This approach relies on narrow baseline stereo, but generates sub-pixel mattes without user intervention.

### 2.2.2 Shape Optimisation

Snakes were introduced by Kass et al. [51] as an algorithm for extracting a contour from an image. They have two key properties; 1) a physical simulation combines an internal regularisation force with an external data force to fit a smooth contour to the data, 2) the shape of the snake determines the region of the image that influences the snake. An iterative approach then allows the image to update the shape of the snake and then uses the shape of that snake to determine the region of the image that is examined. In 3D, snakes can be implemented with an implicit geometric representation such as level sets[89], or with an explicit geometric representation such as elastic deformable models[98]. Deformable models are particularly attractive when the surface to be reconstructed is small compared to the reconstruction volume and a high resolution result is desired, as the computation cost is proportional only to the number of surface elements, whereas in level sets it is proportional to the number of volume elements. Another discriminant between the two types of technique is that level sets can change their topology whereas deformable models cannot.

Snakes using deformable models are an attempt to use physical modelling to combine multiple data cues in a smooth way. The deformable model is initialised with a shape known to be close to the final solution. This surface is then modelled as an elastic object acted on by many springs, fields or other physical constraints. The simulation of the object’s elasticity provides the internal regularisation force, and the physical constraints are used to generate the external forces. A physical simulation is then used to determine the movement of the model and, at some termination point, the final shape of the model is recorded. Deformable models have recently been used as a successful framework for combining stereo and silhouette constraints in order to refine 3D geometry [40, 92] and these techniques are amongst the highest quality techniques currently in use[88].
2.3 Modelling Human Activities

While direct 3D reconstruction attempts to explain the appearance of a scene in terms of a generic model of the plenoptic function, an alternative approach is to make use of high level scene knowledge in order to determine the constituent parts of the scene. In sports reconstruction, as in many fields where computer vision is of interest, the objects being reconstructed are humans, and human action recognition and tracking is an extensive and well studied field.

2.3.1 Human Activity Detection

The field of human action recognition has applications in monitoring the elderly or infirm, detecting abnormal behaviour in security applications, and interpreting body language and sign language. In the general case, human activity detection consists of analysing video sequences of humans performing certain activities and comparing them to some activity model in order to generate a high level description of the recorded activity.

Typical work in this field attempts to detect discrete, pre-determined actions within an image sequence. In order to interpret the image sequence, either some a priori assumptions about the structure of the data are required, or the data is first transformed to a lower dimensional representation, e.g. through feature detection.

As the aim is to detect events in both space and time, appropriate features must contain a temporal as well as a spatial element. An example of the features used in human activity detection techniques are space-time-interest-points introduced by Laptev[62]. To generate these features a sequence of images is stacked along the time axis so that it becomes a single 3-dimensional structure with axes $x, y$ and $t$ ($x$ and $y$ being the original image coordinates and $t$ being time). Corners and edges detected in this 3D space correspond to events in the video, and these detected events can be used to recognise human activities such as walking, running, boxing etc.[62]. Space-time-interest-points have also been used to detect other activities such as mouse behaviour and facial expressions, as in work by Dollar et al. [21].
2.3. Modelling Human Activities

Rather than representing an activity as a set of discontinuous features, a sequence of segmented images can be concatenated along the temporal axis to form space-time shapes as introduced by Gorelick et al. [32]. Plate-like and stick-like structures are detected within the shape in order to perform human motion classification.

An alternative to these discrete space-time features and shapes are motion templates based on local distributions of optical flow as introduced in work by Efros et al. [23]. This technique is of note as it has been successfully applied to real sporting footage, matching recorded images against a database of templates for activity labelling and image replacement. Similar work by Robertson and Reid [85] combines motion templates with spatio-temporal descriptors and interprets the sequence using a hidden Markov model to generate a high level description of a tennis match. Other work on low resolution sports footage includes that published by Lu and Little, who used histograms of orientated gradients to perform action recognition on low resolution soccer and hockey video sequences [68].

Action recognition using 3-dimensional sequences is a less studied area. An example of work in this field is the work on Motion History Volumes introduced by Weinland et al. [104]. This technique generates a shift- and rotation-invariant descriptor of pre-segmented human activities by taking the Fourier magnitudes of a 3-dimensional motion signature defined in cylindrical co-ordinates around the central axis of the subject. A low-dimensional feature is generated from the descriptor and used to compare recorded motions against a pre-labelled library of motions.

While these techniques can accurately represent the changing type of activity, they rarely model the precise state of the activity in question. So for example, the model may recognise that a subject is waving their arm up and down, but it will not be able to determine precisely where the subject’s arm is at any specific frame. This does not meet the requirements of those seeking to generate a synthetic representation of a scene, as it offers no way to relate the observed activity at a specific time back to the motion model in a manner suitable for synthesis. A notable exception to this is the voice puppetry system introduced by Brand [8] which relates an input Markov model in one domain (voice recognition) to an output model in another domain (video animation) through...
Chapter 2. Literature Review

Figure 2.5: Diagram showing a human skeletal model in “rest” pose (left) and a frame from a jogging animation (right). The model consists of bones and joints which are arranged hierarchically. Each joint consists of a fixed translation in the parent joint’s co-ordinate space and a rotation of the joint’s own co-ordinate space. In this way, the rotation of a joint does not affect the joint itself but rather the placement of all child joints.

an entropy-minimising cross-training technique.

2.3.2 Pose Detection

An alternative set of methods which do attempt to explicitly model the state of the human subject are those concerned with pose detection. The pose of a human body can be described in terms of a set of angles which describe the rotations at the joints of a rigid skeletal system (see Figure 2.5). A sequence of rotations along with a translation of the root node is thus sufficient to fully describe the pose of a human at any given moment in time. Systems of this type are commonly used both for calculating inverse and forward kinematics in the field of computer graphics, and for calculating human pose from images in both marker-based and marker-less motion capture systems. The field of image based motion capture is itself a significant area and a review of the work in this field can be found in Moeslund et al. [73].

Pose detection is typically employed with the aim of generating high-level descriptions
of the detected motion for applications such as action recognition for surveillance[4]. In these systems, a synthetic human body model is manipulated and compared to the observed images (which may or may not be pre-segmented). The pose of the model is then optimised to best fit the observed images. Pose detection may be performed on a single image or a sequence of images from one or more cameras.

There is a substantial body of work on 2D pose recognition. Recently Dimitrijevic et al. used Bayesian templates to recognise walking poses in natural scenes[20], Ferrari et al. used progressive search space reduction to estimate body pose in TV and film data[26], and Gammeter et al. used a statistical model of human pose to refine a pedestrian tracking system[30].

There also exist techniques which attempt to determine 3-dimensional human pose from a single viewpoint. Agarwal and Triggs [1] use silhouettes generated from background subtraction to generate histogram based shape-context descriptors. Nonlinear regression combined with a dynamic model is then used to infer 3-dimensional pose. Training is carried out using simple human models and motion capture data. More recent work by Guan et al. [35] optimises an initial manual pose estimate to jointly estimate pose, body shape, lighting and albedo using both shading and edge cues in the image. While this technique can provide highly accurate results from monocular images, it can not operate in the presence of loose clothing as this affects the shading and edge cues required for the optimisation.

All monocular work however is limited in the cases of severe occlusions, clutter and restricted fields of view. If something is not visible from the given viewpoint, there is no way to recover the missing information. One way to overcome this issue is to combine images from multiple calibrated cameras. Detailed pose recovery from multiple calibrated cameras is generally confined to studio systems which are often referred to as markerless motion capture systems. A variety of techniques exist varying in their use of body model and optimisation algorithm. Techniques include the PoseCut system presented by Bray et al. [9] which combines a simple stick figure human model with conjugate gradient descent to jointly discover the best pose and segmentation of a subject. Other work includes state-of-the-art systems such as those of Balan et al. [5]
and Deutscher and Reid [19] which model the body as a system of articulated cylinders and use particle filter based methods to both estimate the initial pose and to track the pose throughout a sequence. Finally, more sophisticated body models may be used, such as the SCAPE model used by Balan et al. [11]. In their technique both body parameters and pose parameters are optimised using a simplex algorithm alternating between pose and body-shape estimation. With this technique body pose may be retrieved in the presence of confounding factors such as loose clothing.

A notable technique in the field of pose estimation from natural scenes is the recent work by Hasler et al. [39] which combines video from multiple moving camcorders to generate a 3-dimensional reconstruction. Cameras are synchronised using the audio streams of the various videos, and structure from motion is used to generate multiple 3D models which are then merged using a point-matching technique. Finally human silhouettes are extracted in a combined pose optimisation and segmentation algorithm. The system requires multiple moving cameras for the structure-from-motion technique to work, however it produces accurate results on a single subject in natural, outdoor scenes.

Less work has been done on matching human pose in 3-dimensional data. Huang et al. performed 3D shape matching for 3D video summarisation of studio data [42] however frames from a sequence were only compared against other frames from the same sequence and no pose information was extracted. More general techniques use an articulated model and the iterative closest point algorithm (ICP) to determine pose information by fitting to the visual hull captured in a studio environment. Work by Mundermann et al. [75] uses a generic repository of articulated models and selects the most appropriate using the height and volume of the subject, while more recent work by Corazza et al. [14] uses a subject-specific model derived from laser scans.

The search space for the pose parameters of the human body is a high dimensional space, and the manifold described by the objective function is often complex with many local minima. As such, many techniques rely on a constrained environment, simple unambiguous poses or good initialisation in order to simplify the problem. Although highly successful in the studio, such techniques fail in the presence of the calibration
and matting errors typical of an outdoor broadcast scenario. Most pose detection techniques are only demonstrated on single subjects, where the subject is large within the image and is framed against a simple background. Pose recovery in video of outdoor team sports, which are cluttered, noisy, and contain multiple small subjects, remains an open and challenging problem.

2.4 Summary

A set of images can be taken as a set of samples of the 5-dimensional plenoptic function. The extension from these samples to a continuous model of the plenoptic function is the key business of 3D-reconstruction. To this end, it relies on the assumption of a Lambertian scene geometry, however ambiguity in camera views and errors in calibration and matting mean that a scene geometry proxy that relies on balancing a set of input costs via a deformable model often yields a far superior result than directly calculating the geometry from the views. For a semantic model of the scene, techniques such as pose detection use a hierarchical, jointed model to represent a human and attempt to best fit this to the observed data. Action recognition techniques often perform well in natural environments as the combination of a high level dynamic model with simple features increases the robustness of the technique with regards to noise in the input data, however they are unable to provide a full specification of the subject’s pose over a sequence. Pose detection techniques provide a full specification of the subject’s pose over a sequence, but due to their reliance on dense information from the input images and the complex and ambiguous nature of pose recovery from camera images, they are susceptible to errors in the data.
Chapter 3

Error-Tolerant 3D Reconstruction

3.1 Introduction

While many types of 3D reconstruction were described in the previous chapter, by far the most prevalent pipeline for wide-baseline multi-camera reconstruction consists of the following four stages:

1. Camera calibration
2. Image segmentation
3. Visual hull calculation
4. Surface refinement

This chapter will examine errors in the first three stages of this pipeline and their effect on the calculation of the visual hull, with a particular focus on issues arising in the outdoor sports broadcast domain. An analysis of the resultant errors in reconstruction will be derived and several variants of shape-from-silhouette (SFS) will be presented. The chapter will conclude with a quantitative analysis of these techniques on real and simulated data from sporting events.
This chapter is based upon previous work published by Kilner et al. The qualitative analysis was introduced in "A Comparative Study of Free-viewpoint Video Techniques for Sports Events" [56] and expanded on in "Objective Quality Assessment in Free-viewpoint Video Production" [55]. The conservative visual hull was introduced in "Dual-mode Deformable Models for Free-viewpoint Video of Sports Events" [57] and the consensus hull was introduced in "3D Action Matching with Key-pose Detection" [53].

3.2 Sources of Error

An outdoor FVV technique must work with cameras arranged along a relatively wide baseline and must reconstruct objects that are relatively small and far away. In an outdoor scene with unconstrained illumination and moving cameras, calibration and matting are likely to be inaccurate. Therefore the key areas of research in transferring techniques from the studio to an external environment are improving the quality of matting and calibration, and improving the robustness of the reconstruction techniques.

3.2.1 Calibration

Accurate camera calibration is required for all wide-baseline FVV techniques, however this is not easily available in the outdoor sports environment. Camera positions are sensitive to change over the course of the event due to factors such as the deformation of the stadium under changes in load or vibration from crowd movements. The cameras will also change viewing direction as the operator moves the camera to follow action on the field. Finally, intrinsic calibration will vary as the camera is zoomed in and out to follow action in the game. With all these variables, calibration prior to capture is not possible.

3.2.2 Matting

Unlike a studio environment, images captured at live outdoor sporting events have unconstrained lighting and complex multi-coloured backgrounds (they are "natural images"). As much of each image consists of players against grass, colour segmentation
3.3. Effects of Errors

Figure 3.1: Comparison of the mattes available in a sports environment (left) with that available in the studio environment with controlled lighting and background (right).

Techniques can be used, taking green as the background colour. However, there remain many problems such as white lines and muddy patches on the pitch, players wearing green or black, shadows, and areas with non-uniform backgrounds such as advertisement boards, the crowd or the goal. Background subtraction can also be used, but this can fail in shadows and regions where the player is the same colour as the background (for example, a player in a white shirt in front of a white line).

3.3 Effects of Errors

In the standard model of multi-view geometry, a camera $C$ is a mapping between the $\mathbb{R}^2$ image domain and $\mathbb{R}^3$. An image which has been segmented into foreground and background is a labelling on $\mathbb{R}^2$, defining a set of pixels $\mathcal{F}$ which consists of all the foreground pixels, and a set of pixels $\mathcal{E}$ consisting of the entirety of $\mathbb{R}^2$ that is external to the image.

Shape-from-silhouette techniques then map this labelling from $\mathbb{R}^2$ to $\mathbb{R}^3$, using the multi-camera geometry of the scene to define $\mathcal{O}$, a set of occupied voxels in $\mathbb{R}^3$. A voxel $v$ is a volume element in $\mathbb{R}^3$. The centre of $v$ is mapped by camera $C$ to the pixel $p$ in image $\mathcal{I}$. To avoid sampling issues, a voxel size is chosen such that the image of $v$ in $\mathcal{I}$ is smaller than a pixel, and thus the centre point can be taken as a representative sample of the entire voxel. This mapping is repeated for all cameras present and a voxel is considered occupied if it maps to $\mathcal{F}$ or $\mathcal{E}$ for all cameras. The set of occupied voxels $\mathcal{O}$
Chapter 3. Error-Tolerant 3D Reconstruction

is then given by:

$$\mathcal{O}(v) = g(v) \prod_{C,v \rightarrow p} \mathcal{F}(p) + \mathcal{E}(p)$$

(3.1)

where $g(v)$ is some function that indicates whether a voxel is suitable for inclusion in the reconstruction. A typical implementation of $g$ would be:

$$g(v) = \left[ \sum_{C,v \rightarrow p} \mathcal{F}(p) \geq \alpha \right]$$

(3.2)

where $\alpha$ is the minimum number of cameras required to produce an acceptable reconstruction (typically around 3). Thus a voxel is a member of $\mathcal{O}$ if and only if it projects to $\mathcal{F}$ or $\mathcal{E}$ in all cameras, and projects to $\mathcal{F}$ in at least $\alpha$ cameras. The visual hull is then the boundary of $\mathcal{O}$.

In a sparsely populated scene such as an outdoor sporting event, errors in calibration will tend to underestimate the visual hull. This is because membership of $\mathcal{O}$ relies on the projection of $v$ to a pixel in $\mathcal{F}$ in all cameras. As $\mathcal{F}$ is small relative to the total image size, the likelihood of a random error erroneously mapping $v$ to $p \in \mathcal{F}$ in all cameras is low, however the likelihood of erroneously mapping $v$ to $p \notin \mathcal{F}$ in one or more cameras is relatively high. This error is compounded by the fact that the error is
3.3. Effects of Errors

Figure 3.3: Reconstruction of an object (circle with diagonal hatchings) from three images using SFS techniques, left: with no calibration error, right: with calibration error. The resultant visual hull in each case is shown as the shaded area formed by the intersection of the three silhouette projection cones.

additive i.e. the more cameras you have, the more likely a single camera will generate an error and therefore the worse quality the reconstruction will be. Due to the large distances between cameras relative to the size of the objects to be reconstructed, a small error in camera calibration parameters (of the order of 1%) can mean that no reconstruction at all is generated.

The visual hull is typically refined using shape estimation which attempts to solve some cost function defined on a manifold $S$ in $\mathbb{R}^3$. This cost function is designed such that it is lowest when the reprojection of the manifold back into the source cameras satisfies some consistency constraints. However, when the error in this mapping is great there is no guarantee that the cost function has a non-trivial minimum, or at any rate has one that is related to the desired surface. An incorrect $O$ will also give a poor estimate of the region within which the optimisation is to be performed, as well as providing poor initialisation for the optimisation. In this way, errors in calibration can have a double effect on the errors in the scene compared to segmentation, errors that are further compounded by the fact that the camera transformation is non-linear due to spherical distortion terms. The complex relationship of the errors in this system quickly yields an intractable solution and so direct estimation of error likelihood is not feasible. Therefore a different approach is taken and rather than looking at the errors in the
estimated parameters, the resultant errors in the rendered frames are considered.

### 3.4 A Taxonomy of Errors

The FVV synthesis pipeline consists of the set of steps shown in Figure 3.4. In the capture and matting process, a video sequence is captured using a video camera and processed so as to extract mattes of the images in the sequence. A matte is a single per-pixel value, stored as either a monochrome bitmap or an extra channel in the image. The per-pixel value indicates how much of that pixel is foreground and how much is background, with intermediate values indicating a pixel that is partially foreground and partially background. During reconstruction, the images, mattes and camera calibrations from all cameras are combined and a scene reconstruction is generated. During rendering, the reconstruction is combined with the specified virtual viewpoint and a synthetic video sequence is generated.

The errors inherent in this process can be classified by considering the ways in which a synthetic video sequence generated for a given viewpoint can differ from the video that would have been captured by a real camera placed at that same viewpoint. In the following discussion the frame from the synthetic video sequence will be referred to as \( J' \) and the corresponding image from the real camera will be referred to as \( J \). Similarly \( \mathcal{R}' \) is a region in \( J' \) which corresponds to the region \( \mathcal{R} \) in \( J \).

#### 3.4.1 Errors in Shape and Appearance

Errors in shape are errors where \( J' \) is missing a foreground element that is present in \( J \), or \( J' \) contains an extraneous foreground element that was not present in \( J \) (see Figure 3.6). Some real-world examples are missing limbs or double images as shown in Figure 3.5 a) and b). In both cases, pixels in \( J' \) have been incorrectly classified as either foreground or background. This error can also be considered an error of completeness, where the reproduction of the foreground is incomplete or overcomplete.

Errors in appearance occur when a region of correct shape \( \mathcal{R}' \) contains different pixel values to \( \mathcal{R} \). This error can occur in two ways. Firstly, the surface that is rendered to
3.4. A Taxonomy of Errors

Figure 3.4: The FVV synthesis pipeline.
Chapter 3. Error-Tolerant 3D Reconstruction

Figure 3.5: A comparison of synthetic images (left) to their corresponding ground truths (right). a) Shows an incomplete synthetic image, b) shows a synthetic image where the player is incorrectly rendered twice, c) shows a player incorrectly rendered to a foreground region, and d) shows a blurred player.

Figure 3.6: Diagram showing errors in shape - the original scene is shown on the left and a reconstruction is shown on the right. Region A demonstrates a region where an additional element is present that was not in the original scene, and region B shows a region where the reconstruction is missing an element from the original scene.
3.4. A Taxonomy of Errors

Figure 3.7: Diagram showing errors in appearance - the original scene is shown on the left and a reconstruction is shown on the right. Region A demonstrates a region where the incorrect surface is rendered (possibly due to an incorrect occlusion calculation), and region B shows an area where a distortion of the correct surface is rendered.

$\mathcal{R}'$ may not be the surface whose image appears in $\mathcal{R}$. Alternatively, the correct surface may be rendered, but the reconstruction technique may incorrectly synthesise the view of the surface, so $\mathcal{R}'$ will be a distorted version of $\mathcal{R}$ (see Figure 3.7). Some real-world examples are the rendering of surfaces in incorrect locations due to incorrect modelling of occlusions, and blurred rendering of surfaces, as shown in Figure 3.5 c) and d). In both cases, pixel values in the generated image are inconsistent with the correct pixel values.

Table 3.1 summarises these classifications. This taxonomy covers all possible errors in the domain of synthesised foreground images. If the entirety of $\mathcal{F}'$ is treated as a region of correct shape then this analysis reduces to a comparison of pixel values across the image. This relates the classification to standard image quality measurements, which are measures of appearance across the entire image. It is the distinction between foreground and background which allows us to also consider shape in this analysis.
Table 3.1: Classification of errors in foreground synthesis, comparing a region of the ground-truth image $\mathcal{R}'$ to a corresponding region of the synthetic image $\mathcal{R}$. $\alpha$ and $\beta$ denote different elements within the scene.

<table>
<thead>
<tr>
<th>Error</th>
<th>Image in $\mathcal{R}'$</th>
<th>Image in $\mathcal{R}$</th>
<th>Classification</th>
</tr>
</thead>
<tbody>
<tr>
<td>missing foreground</td>
<td>present</td>
<td>absent</td>
<td>error in shape</td>
</tr>
<tr>
<td>extraneous foreground</td>
<td>absent</td>
<td>present</td>
<td>error in shape</td>
</tr>
<tr>
<td>none</td>
<td>present</td>
<td>present</td>
<td>correct shape</td>
</tr>
<tr>
<td>incorrect image</td>
<td>image of $\beta$</td>
<td>image of $\alpha$</td>
<td>error in appearance</td>
</tr>
<tr>
<td>distorted image</td>
<td>distorted image of $\alpha$</td>
<td>image of $\alpha$</td>
<td>error in appearance</td>
</tr>
<tr>
<td>none</td>
<td>image of $\alpha$</td>
<td>image of $\alpha$</td>
<td>correct view synthesis</td>
</tr>
</tbody>
</table>

3.4.2 Relationship to Real Errors

Errors in rendering are caused by four geometric issues which relate directly to the taxonomy of resultant errors. A reconstruction can be overcomplete, meaning that it contains surfaces which were not present in the original scene. These surfaces will lead to extraneous foreground in the rendered scene. Overcomplete reconstructions occur where the projection cones of silhouettes overlap, but where no object was originally present. This type of error is referred to as a phantom volume, or (if it is conjoined with the real surface) a phantom protrusion. In the presence of clutter or concave surfaces, errors such as this can occur even if calibration and matting are free from error.

Geometry can also be truncated. This will lead to missing foreground in the reconstructed images. This error can be caused by errors in matting or calibration. If matting and calibration are free from errors then the visual hull is guaranteed to be free from these errors.

An important source of error comes from the calculation of occlusions. If occlusions are not correctly modelled, or the geometry used to calculate the occlusions is insufficiently detailed, then the appearance that is rendered to a surface may differ drastically from that of the true surface. If the reconstruction is incomplete along the line of sight of the source camera, then the appearance of an item closer to the camera will be incorrectly
rendered on the geometry. Similarly, an extraneous piece of geometry may be rendered with the appearance of an object behind it. This type of error will lead to a gross error in appearance in the rendered image.

Finally, parallax errors in the geometry will cause the projective relationship between the source camera and the target camera to be incorrect and hence the image to be distorted. If the surface is locally smooth but not at the correct location then the result of this is to shift the image in the target camera. If the surface is not locally smooth then the result will be to distort the image. This error will also generate errors in the appearance of the rendered image, however these errors will typically only displace the resultant pixels a small distance from their correct location in the rendered image.

3.5 Quantitative Evaluation

3.5.1 Objective Quality Assessment

In image-based reconstruction, geometric accuracy has been evaluated using ground-truth 3D shape. Seitz et al. [88] present a comprehensive framework to compare reconstruction techniques against 3D geometry acquired from a laser stripe scanner. Recent work in free-viewpoint production has demonstrated that with current camera hardware, geometric accuracy is insufficient to represent the detailed geometry of a scene, and that where display resolution reflects camera resolution, image-based rendering is required to achieve sub-pixel accuracy to minimise visual artifacts in view synthesis[93]. An evaluation of free-viewpoint video should therefore target the accuracy, or quality, of view synthesis, rather than ground-truth accuracy in geometric reconstruction.

The problem of defining video quality metrics has received significant interest in the image processing community to assess degradations introduced by video acquisition, processing, coding, transmission and display [78]. There is also active research studying the degradation introduced by watermarking schemes [16]. An overview of the field can be found in Eckert and Bradley [22]. Research into image quality assessment can be broken down into two broad categories; those attempting to model the human visual system (HVS), and those using more direct pixel fidelity criteria.
There has been much work focusing on HVS-based measures of the fidelity of an image. Examples include measuring mutual information in the wavelet domain [90], contrast perception modelling [79] and modelling the contrast gain control of the HVS [106]. However, HVS techniques do not necessarily reflect the true complexity of the visual system and objective measurement of perception remains an open research problem.

Pixel-wise fidelity metrics such as mean square error (MSE) and peak signal to noise ratio (PSNR) remain widely adopted as simple, well-understood measures of fidelity, despite a poor correlation with visual quality [103]. An overview of several measures and their performance can be found in Eskicioglu and Fischer [24], while a statistical analysis of various techniques encompassing both pixel metrics and HVS-based metrics can be found in Avcibas et al. [3].

Objective evaluation should ideally provide simple, repeatable quality measures that afford a clear physical interpretation tailored to perceived visual quality. However, the lack of effective standard measures is testament to the difficulties of achieving this in the general case, and so a more fruitful approach is to use domain-specific measures to target the quality assessment of particular types of image.

3.5.2 Quality Metrics

Metrics to measure errors in appearance and shape can now be derived. In the following discussion, \( p \) and \( q \) are pixels in the real image \( I \) while \( p' \) and \( q' \) refer to pixels in the synthesised image \( I' \).

A serious problem that affects any attempt to evaluate the quality of a reconstruction technique is the provision of ground-truth images. If a synthetic data set is used, a genuine ground-truth can be provided. However, in using a synthetic data set, we are no longer testing the technique on the kind of data we truly wish to evaluate. On the other hand, if real data is used then invariably the ground-truth itself must involve some estimated parameters. This can be illustrated by considering a real image \( I \) with calibration \( C \). As direct measurement of \( C \) is not possible, only the estimated calibration \( C' \) is available. This means that the synthetic image \( I' \) is generated with calibration \( C' \) rather than \( C \). In the studio environment, where calibration can be
3.5. Quantitative Evaluation

determined accurately, the approximation $C \approx C'$ is valid, and so the difference between $J$ and $J'$ is due purely to the difference between the real world and the reconstruction. However, if $C$ and $C'$ differ significantly (as they do in the case where only approximate calibration is available), then at least some of the difference between $J$ and $J'$ is due to the difference in the viewpoint of the two images.

An $r$-shuffle is a perturbation of an image such that if $J'$ is an $r$-shuffle of $J$, then every pixel $p \in J$ will be transformed to a pixel $p' \in J'$, such that $||l(p') - l(p)||_2 < r$ [60] (where $l(p)$ gives the co-ordinates of pixel $p$). By modelling the differences between $I$ and $I'$ caused by errors in $C'$ as an $r$-shuffle, the minimum difference between $J$ and $J'$ due to reconstruction error can be estimated. In this way, a lower bound on the distortion introduced by a particular reconstruction technique can be set.

The $r$-neighbourhood $N_r$ of a pixel $p$ on the image $J$ is defined such that for some other pixel $q$

$$q \in N_r(p) \iff ||l(q) - l(p)||_2 < r. \quad (3.3)$$

Image quality metrics in terms of errors in shape and appearance that are independent of the error in the ground-truth calibration can now be defined. To measure errors in shape we simply look for the difference between the ground truth and the synthetic image in terms of pixels marked as foreground. Detecting errors in appearance is simply a case of looking at the pixel values in the regions free from errors in shape.

First a function $h$ is defined that indicates whether any pixel $p$ within the $r$-neighbourhood of a pixel $p'$ is foreground:

$$h(p') = \max_{p \in N_r(p')} \mathcal{F}(p) \quad (3.4)$$

Then the shape matching function $s$ can be defined.

$$s(J') = \sum_{p' \in J'} \frac{\mathcal{F}(p')h(p')}{\sum_{p'' \in J'} \max(\mathcal{F}(p''), h(p''))} \quad (3.5)$$
When $\tau$ is taken as 0, this is simply the Jaccard index of the foreground regions in $J$ and $J'$ (i.e. the ratio of $F'$ to $F \cap F'$).

Another indicator function $\tilde{h}$ can be defined to indicate whether any pixel $p$ within the $\tau$-neighbourhood of a pixel $p'$ has a colour value within a certain tolerance $\tau$ of the value at $p'$:

$$\tilde{h}(p') = \left[ \min_{p \in N_\tau(p')} (\|c(p') - c(p)\|_2) < \tau \right]$$  \hspace{1cm} (3.6)

where $c(p)$ is the colour of pixel $p$ represented as an RGB value. Summing over the entire image and normalising by the common foreground region gives the appearance score.

$$a(J') = \frac{\sum_{p' \in J'} F(p')h(p')\tilde{h}(p')}{\sum_{p' \in J'} \max(F(p'),\tilde{h}(p'))}$$  \hspace{1cm} (3.7)

Thus the appearance score is the ratio of the number of pixels rendered with correct shape and appearance, to the number of pixels rendered with correct shape.

These measures are compared against the PSNR which is given by:

$$\text{PSNR}(J') = 20 \log_{10} \left( \frac{K \sqrt{|J'|}}{\sqrt{\sum_{p' \in J'} \|c(p') - c(p)\|_2}} \right)$$  \hspace{1cm} (3.8)

where $K$ is $\|c_{\text{max}} - c_{\text{min}}\|_2^2$, $c_{\text{max}}$ and $c_{\text{min}}$ being the minimum and maximum possible values returned by $c(p)$.

Table 3.2 shows a comparison of these measures against the visual information fidelity measure (VIF) [90] of visual quality in an image which was chosen as a baseline full-reference quality metric. The comparison was carried out on several test images, some consisting of filtered versions of an original image and others on reconstructions of the scene. It should be noted that for this test the “original” used was a hand-matted image, thus the reconstruction scores are particularly low as they include errors from matting. It can be seen that measures are in broad agreement, justifying the use of
Table 3.2: Comparison of evaluation techniques on a single frame. Median and Blur are the original image after applying a median filter and a Gaussian blur respectively. VH and BB are rendered reconstructions using a view-dependent textured visual hull technique and a billboarding technique respectively.

<table>
<thead>
<tr>
<th>Image</th>
<th>Shape</th>
<th>Appearance</th>
<th>PSNR</th>
<th>VIF</th>
</tr>
</thead>
<tbody>
<tr>
<td>Original</td>
<td>1</td>
<td>1</td>
<td>inf</td>
<td>1</td>
</tr>
<tr>
<td>Median</td>
<td>0.99</td>
<td>0.98</td>
<td>17.35</td>
<td>0.36</td>
</tr>
<tr>
<td>Blur</td>
<td>0.75</td>
<td>0.96</td>
<td>16.64</td>
<td>0.32</td>
</tr>
<tr>
<td>VH</td>
<td>0.86</td>
<td>0.99</td>
<td>14.02</td>
<td>0.22</td>
</tr>
<tr>
<td>BB</td>
<td>0.81</td>
<td>0.95</td>
<td>11.03</td>
<td>0.17</td>
</tr>
<tr>
<td>Blank</td>
<td>0</td>
<td>0</td>
<td>6.00</td>
<td>0</td>
</tr>
</tbody>
</table>

these measures in the comparison, however the shape and appearance figures give more information than VIF or PSNR as to the nature of the reconstruction. For example, all scores agree that the blur transformation produces a worse effect than the median filter, but the shape and appearance scores correctly indicate that the median filter preserves the shape of the image, which the blur does not.

3.6 Error-tolerant Visual Hull

The typical 3D reconstruction pipeline begins with an estimation of the visual hull. By definition, any visual hull calculation is an SFS technique. Even those probabilistic models which do not carry out any prior segmentation involve an implicit model of the silhouette images which are calculated as part of the visual hull recovery.

The attempt to reconstruct a visual hull from noisy mattes has received considerable interest in the past. However, this work has typically focused on errors in the silhouettes, such as occlusions and false positives (typically shadows)[29], or errors in quantisation [76]. Little work has been done on robustness to gross calibration error, which is a challenging topic and typically unnecessary in the studio environment.

A camera system in the standard model of 3D reconstruction consists of a set of mea-
surveysments (the images themselves) and some estimated parameters (the camera calibration). While errors in the measurements are small and typically consist of normally distributed fluctuations in the recorded values, the errors in the camera calibration may be large and subject to systematic bias. Due to the complex nature of these errors, directly modelling them is not feasible, therefore a preferable alternative is to follow the same reasoning as was used in the quantitative evaluation and seek to model the resultant errors rather than than the estimation errors. In this way, the errors in the system can be more accurately modelled while still obtaining a computationally tractable and useful result.

To this end, three main sources of error in the system must be considered:

- Sensor error in the silhouette extraction phase
- Mapping error in camera calibration
- Gross error due to occlusions or gross calibration error

### 3.6.1 Visual Hull Techniques

All techniques presented in this chapter are built upon a common framework. In order to estimate the visual hull over the large areas involved with sports reconstruction while maintaining the required level of detail, an octree-based algorithm is used. This divides the volume to be reconstructed hierarchically into progressively smaller voxels. Each voxel is reprojected back into the source images and the silhouette overlap in each camera is calculated to determine whether the node requires subdivision (fully or partially occupied), or whether it should be discarded (empty). At the finest level of division, voxel occupancy $O(v)$ (where $0 \leq O(v) \leq 1$) is calculated and the marching cubes algorithm[65] is used to extract a triangle mesh representation of the $O(v) = 0.5$ iso-surface.

To repeat the equations presented earlier in this chapter, the basic visual hull calculation is governed by the following equation for the determination of voxel occupancy $O(v)$:

$$O(v) = g(v) \prod_{C:v\rightarrow p} T(p) + E(p),$$

(3.9)
3.6. Error-tolerant Visual Hull

3.6.2 Tri-state Visual Hull

The first error to model is sensor error. If silhouette extraction is performed using standard techniques, then non-recoverable errors occur where incorrect matting leads to erosion or inflation of the visual hull. This problem can be mitigated by introducing the concept of a third silhouette state, so that the silhouettes define two sets of pixels, \( \mathcal{F} \) definitely foreground and \( \mathcal{M} \) potentially foreground. These can be generated by simply looking at the likely causes of matting errors. The first error is the misclassification of mixed pixels around the boundary of a player matte. The second error is the misclassification of pixels around background regions which fall into the foreground colour distribution.

Both of these regions of uncertainty can then be accounted for in the mattes - the first by dilating the silhouettes with a single pixel border of potential foreground, and the second by identifying regions which are classified throughout the entire sequence as foreground and marking them as only potentially foreground. An example of this is

\[
g(v) = \sum_{C:v=p} \mathcal{F}(p) \geq \alpha.
\]  

(3.10)

Figure 3.8: Reconstruction of an object (circle with diagonal hatchings) using SFS techniques, left: with no calibration error, centre: with matting error and right: tri state visual hull with matting error. In each case, the reconstructed areas are shown as shaded. Indeterminate regions are shown as grey on the image plane. Regions that only project to indeterminate areas of the matte are not reconstructed and so several 'phantom volumes' are eliminated.

with \( g \) given by:
Figure 3.9: The sum of all mattes in an image sequence from one camera. It can be clearly seen that some pixels are labelled as foreground for the entire sequence. Thus, for these pixels the labelling of “foreground” is actually indeterminate.
3.6. Error-tolerant Visual Hull

Figure 3.10: Reconstruction of an object (circle with diagonal hatchings) using SFS techniques, left: with no calibration error, centre: with calibration error and right: conservative visual hull with calibration error. In each case the reconstructed area is shown as shaded. Increasing the width of the silhouette projection cones increases the area of overlap and so reduces truncation.

shown in Figure 3.9, where the fully white pixels are classified as foreground throughout the entire sequence.

Once these regions are classified, we can take advantage of the fact that a region that is uncertain in one camera view is likely to be unambiguous in another view. This can be done by modifying the visual hull formulation to allow volumes which reproject to a mixture of $\mathcal{F}, \mathcal{M}$ and $\mathcal{E}$ to be included in the reconstruction.

$$\mathcal{O}(v) = g(v) \prod_{C: v \rightarrow p} (\mathcal{F}(p) + \mathcal{M}(p) + \mathcal{E}(p))$$ (3.11)

as $g$ is still expressed in terms of $(F)$, at least $\alpha$ reprojections to $\mathcal{F}$ must occur, otherwise the volume is discarded. In this way, $O$ consists of voxels which do not project to background in any images and project to foreground in at least $\alpha$ images. The tri-state visual hull can help with gross errors in matting, such as logos printed on the pitch and areas where the player is silhouetted against the crowd. Background subtraction in the matting process can also help with these problems but requires hard decisions to be made which cannot then be corrected by information from other views.

3.6.3 Conservative Visual Hull

The second error to be considered is error from camera calibration. Using the same model as was used for the quantitative evaluation performed in the previous section, it
is assumed that the error in the image reprojection can be modelled with an $r$-shuffle. Determining the maximum possible occupancy value for a voxel within this modelled distortion sets an upper bound on the true shape to be recovered. The shape generated by this technique is referred to as the conservative visual hull (CVH - shown in light grey in Figure 3.10). The benefit of using a CVH in this way is that it provides a more complete reconstruction, as shown in Figure 3.11. The disadvantage of a CVH is that it is only weakly related to the true surface; if a large enough value of $r$ is used the CVH will contain the true surface, but no other guarantees are given.

\begin{equation}
\mathcal{O}(v) = g(v) \prod_{C_{im}=p} \max_{q \in \mathcal{N}(p)} \max \mathcal{F}(q) + \mathcal{E}(p)
\end{equation}

### 3.6.4 Consensus Hull

The final type of error to consider is occasional gross error. This can occur when matting fails completely due to occlusion of the image by a background object, or towards the edge of an image when radial distortion parameters are incorrectly estimated. Here, the fact that the problem is over-specified, and that accidental correspondence between cameras is rare, can be exploited.

If we relax the constraint that voxels in $\mathcal{O}$ must map to pixels in $\mathcal{S}$ in all cameras, a
3.6. Error-tolerant Visual Hull

Figure 3.12: Reconstruction of an object (circle with diagonal hatchings) using SFS techniques, left: with no calibration error, centre: with calibration error and right: consensus hull with calibration error. It should be noted that the particularly low number of cameras creates a significant distortion to the shape in this example. This distortion is reduced as more cameras are used.

Figure 3.13: Comparison of visual hull (left) and consensus hull (right) for the reconstruction of a person. Note truncation is greatly reduced although some detail is lost.

The consensus hull can be calculated, where the consensus hull is an iso-surface of labelling consensus, as shown in Figure 3.12. As the level of required consensus is reduced from 100% (equivalent to the visual hull), the consensus hull (CH) quickly generates a lot of phantom volumes and noise, but at 80% – 90% shape is retained while truncation is significantly reduced as shown in Figure 3.13.

\[ 0(v) = g(v)H\left(\sum_{C_{uv}=p} F(p) + E(p) - \beta\right) \]

(3.13)

where \( \beta \) is the required number of cameras to provide a reconstruction and \( H \) is the...
Chapter 3. Error-Tolerant 3D Reconstruction

Heaviside step function:

\[ H(x) = [x \geq 0] \]  

(3.14)

3.6.5 Combined Technique

All three of these parameterisations can be combined into the following occupancy test:

\[ \Theta(v) = g(v)H\left( \left( \sum_{q \in \mathcal{N}_p} \max_{q \in \mathcal{N}_p} \mathcal{F}(q) + \mathcal{M}(q) + \mathcal{E}(p) \right) - \beta \right) \]

(3.15)

This combined technique is referred to as the conservative consensus hull or CCH. For this work a CCH was used where a 90% consensus hull was calculated using a 1 pixel reprojection error tolerance. This generates a complete reconstruction of the scene at the cost of some loss of accuracy. However, the loss of accuracy is considerably less than if the CVH is used alone, which requires a reprojection threshold of up to 4 pixels in order to generate a complete reconstruction.

Taking these values together allows us to parameterise the reconstruction in an explicit and unambiguous way, for example: reconstruct all voxels that reproject to within 1 pixel of an occupied pixel in 90% of the cameras, and where at least 3 views are unambiguous. With parameter settings like this, it is possible to tailor the settings to specific environmental conditions in a meaningful way.

Calculation of the CCH is not suitable as a technique for accurate surface reconstruction, but properly configured it provides sufficient guarantees of completeness to be used to initialise a refinement technique. Due to the weaker constraints for agreement between the original input images, the CCH is more susceptible to phantom volumes and to errors from pixel noise in the silhouettes. However with some simple domain knowledge (no surfaces exist below pitch level, no very small objects etc.), effective clean-up is possible.
3.6. Comparison with Probabilistic Hull

The work presented by Franco and Boyer[29] frames the SFS problem as a Bayesian reasoning problem. It attempts to determine the occupancy likelihood for every voxel in a region of space by considering the joint likelihood of an image pixel being marked as foreground and the likelihood that the pixel's state is related to the occupancy of the voxel.

This can be considered as a probabilistic integration function:

\[ \Psi_o(v) = \prod_{C: v \rightarrow p \text{exp} \rightarrow q} \sum_{\Psi_f(q) \Psi_q(v)} \]

\[ \Theta(v) = \left[ \Psi_o(v) > \gamma \right] \]

where \( e \) is a calibration error function mapping \( p \) to a nearby pixel \( q \), \( \Psi_o \) is the probability of a voxel being occupied, \( \Psi_f \) is the probability of a pixel being foreground (modelled in terms of the background colour distribution combined with false detection and sensor error likelihoods), and \( \Psi_q \) is the probability that the pixel measurement \( q \) relates to a voxel (modelled as a flat distribution over a window of pixels around the voxel reprojection).

Although this technique incorporates a model for calibration errors and occlusions, these models are simple uniform distributions which account for the possibility of an error, rather than actually calculating the probability of a specific error. A full model of occlusion likelihood would relate every voxel to every other voxel and to every pixel in every image and thus quickly render computation intractable. Properly accounting for calibration errors with their complex non-linear characteristics would further complicate matters.

Another confounding factor with such probabilistic methods is that they give a volume of maximal likelihood. This volume contains all points whose likelihood of falling within the real measured volume exceeds some threshold \( \gamma \).

One way to interpret \( \gamma \) is in terms of the space of all reconstructions that are possible given the measurements provided and all possible permutations of the estimated pa-
rameters within the known error bands. In this case, a point within the reconstructed volume generated with $\gamma = 0.8$ would lie within 80% of all possible reconstructions. This model is effective so long as errors are small and randomly distributed about the mean. However, in the presence of systematic errors or gross bias, it may be that the true surface only lies within a small fraction of these possible reconstructions, and in order to extract the relevant surface a value of $\gamma$ has to be chosen that is so low as to make the extracted surface grossly distorted.

There are similarities between the probabilistic visual hull and the work presented in this chapter such that the following comparison can be made; the r-shuffle of the CVH is analogous to the window size used to calculate $\Psi_q$, the tri-state visual hull can be considered a thresholding on the false alarm and detector rate probabilities used to calculate $\Psi_f$, and the consensus hull parameter $\beta$ is analogous to the probability threshold $\gamma$. However, there are some distinct differences which make the presented technique more suitable for use in the outdoor sports broadcast environment.

Firstly, the per-pixel reliability score is based on the behaviour of the pixel rather than a model of the colour distribution. In this way, the system can correctly recover from errors in the colour model, which can become confused by scene elements such as the crowd and animated billboards. Secondly, the r-shuffle is a much weaker constraint on the reprojection of errors than the model proposed in the probabilistic work. As such, errors that do not fit well to that model (such as a reduction in size due to incorrect estimation of zoom parameters) are not as heavily penalised as they are in the probabilistic model. Finally, as the consensus level is a threshold on the voting algorithm, it can recover from a gross error in one or more cameras. In a probabilistic framework, the likelihood of occupancy of a voxel is dominated by the contribution from the least favourable camera. In order to maintain a high probability for voxels where only one camera has “failed” either a low threshold for the probability iso-surface must be set or a high probability for sensor failure must be maintained, both of which will tend to produce a noisy, over-inflated mesh.

The result of these differences is that the probabilistic methods are harder to parameterise such that a single set of parameters will produce a good reconstruction over the
Figure 3.14: Comparison of conservative consensus hull (left) and probabilistic hull (right) for the reconstruction of a cluttered scene. Both images are close ups from the same scene. Both techniques suffer from some phantom volumes, however the probabilistic hull suffers from greater phantom volumes (bottom row), while at the same time being less complete than the conservative consensus hull (top row).
entire field of play. This is shown in Figure 3.14 where the probabilistic hull is less complete than the CCH in some areas of the scene, and has larger phantom volumes in other areas.

3.6.7 3D Clean-up

After calculation of voxel occupancy, a final clean-up of 0 can be performed. This can make use of various constraints on the reconstruction that are imposed by the reconstruction scenario. For example, volumes outside of the field of play can be immediately discounted, as can those on or beneath the pitch. If the ball can be tracked, then volumes which are entirely above a certain height (say 1.5m) can be discounted as well. More usefully, temporal consistency checks can be applied and volumes which come into and go out of existence, as well as volumes which stay above or below a certain height for the duration of the clip, can be removed. Finally, simple size filters may remove volumes under a certain threshold (typically the size of the ball). In this way, many matting errors which are difficult to disambiguate in 2D can become trivial to repair in the 3D domain. A typical result is shown in Figure 3.15.

3.7 Quantitative Analysis of Techniques

The different SFS techniques are now evaluated using the comparison metric introduced earlier in this chapter. Various data sets of football and rugby matches with between 6 and 14 cameras were used to perform the evaluation (a full description of the data sets can be found in Appendix A). Approximately 20 frames were evenly sampled from each data set for comparison. A camera was held back from each data set, and the remaining cameras were used to generate a reconstruction with each of the various techniques. The reconstructions were then rendered from the point of view of the held-back camera, using the view-dependent rendering algorithm described in Appendix C.

The resultant images were then compared against the real image recorded from the held-back camera. The real images were segmented using the same automatic technique used in the reconstruction process. These ground-truth images were then hand edited
3.7. Quantitative Analysis of Techniques

Figure 3.15: An example of the CCH before and after 3D cleanup. Cleanup has been performed by considering the axis-aligned bounding-cubes of all objects and removing those objects which meet any of the following criteria: volume less than $0.5 \text{m}^3$, does not extend below 1m in y-axis, does not extend above 0.7m in y-axis, has any one dimension less than 0.2m or has all dimensions greater than 1.0m. Note that only disconnected meshes are cleaned up, hence the phantom protrusion on the player near the centre of the image is not removed.

<table>
<thead>
<tr>
<th>Technique</th>
<th>Shape</th>
<th>Appearance</th>
</tr>
</thead>
<tbody>
<tr>
<td>visual hull</td>
<td>0.801</td>
<td>0.950</td>
</tr>
<tr>
<td>tri-state visual hull</td>
<td>0.801</td>
<td>0.950</td>
</tr>
<tr>
<td>conservative visual hull</td>
<td>0.894</td>
<td>0.941</td>
</tr>
<tr>
<td>consensus hull</td>
<td>0.901</td>
<td>0.955</td>
</tr>
<tr>
<td>conservative consensus hull</td>
<td>0.868</td>
<td>0.879</td>
</tr>
<tr>
<td>probabilistic hull</td>
<td><strong>0.915</strong></td>
<td>0.942</td>
</tr>
</tbody>
</table>

Table 3.3: Mean appearance and shape scores across both synthetic data sets for the techniques shown.
to remove gross errors such as undesirable regions of the stadium and pitch lines. The results of the comparison are presented in Table 3.3, which shows the technique as applied to synthetic data, and Table 3.4, which shows the technique as applied to real data. For both tables a value of \( r = 2 \) was chosen as an estimate of the calibration error in the ground-truth.

Further results are presented in Figure 3.16 and Figure 3.17, which plot each technique against the size of the \( r \)-shuffle used in the analysis. Finally, some example images are compared with the ground-truth for each data set. These are presented in Figures 3.18 to 3.24.

Several factors mean that even a perfect reconstruction would fail to return perfect scores in this analysis. These are:

- Errors in the ground-truth mattes
- Limits to camera visibility (a player captured only in the ground-truth camera will not be reconstructed)
- Matting errors in the other cameras which will affect rendering even if the reconstructed geometry is good

These factors affect all techniques equally and so do not impact on the relative values attained by the different techniques.
3.7. Quantitative Analysis of Techniques

Figure 3.16: Graphs displaying the shape score for each technique and data set plotted against the size (in pixels) of the r-shuffle used to calculate the score.
Figure 3.17: Graphs displaying the appearance score for each technique and data set plotted against the size (in pixels) of the r-shuffle used to calculate the score.
Figure 3.18: Rendered images compared to the ground-truth for the Synthetic data set.
Figure 3.19: Rendered images compared to the ground-truth for the Synthetic Error data set.
Figure 3.20: Rendered images compared to the ground-truth for the Football 1 data set.
Figure 3.21: Rendered images compared to the ground-truth for the Football 2 data set.
Figure 3.22: Rendered images compared to the ground-truth for the Football 3 data set.
Figure 3.23: Rendered images compared to the ground-truth for the Rugby 1 data set.
Figure 3.24: Rendered images compared to the ground-truth for the Rugby 2 data set.
The results presented in Table 3.3 show that in idealised conditions with small errors in calibration and no errors in matting, the probabilistic hull provides the reconstruction closest in shape to the original, while only introducing a slight degradation in shape. Similarly, the consensus hull maintains the appearance of the reconstruction, although it also introduces slight distortions to the shape of the reconstruction.

The analysis carried out on real data, as shown in Table 3.4, demonstrates that the CVH provides the best compromise between maximising the shape score and preserving the appearance score. This is borne out by examination of the results shown in Figures 3.18 to 3.24. However, the CVH still suffers from truncations, and if a more complete reconstruction is required then the CCH provides the best balance between avoiding truncations and not introducing too many additional phantom volumes, giving the best overall shape of any reconstruction technique. It should be noted that the variation in the appearance of the different techniques is minor - all techniques preserve the general appearance of the reconstruction, but introduce local distortions which negatively affect the visual quality of the result. The major difference between the techniques is apparent in the shape scores, which clearly group the techniques into those with little error tolerance (standard and tri-state visual hull), those with moderate error tolerance (consensus hull) and those with high error tolerance (conservative, conservative consensus and probabilistic hull).

An examination of the results broken down by data set gives further insight into the strengths and limitations of each of the techniques. It should be noted that the camera configuration for the Rugby 1 and Rugby 2 data sets (and hence also for the synthetic data sets which use the Rugby 1 camera calibration) provide the most challenging view interpolation, with a large distance between the virtual and real cameras. It is for this reason that the Rugby 1 data set particularly separates the techniques in the appearance score - the challenging combination of a cluttered area around the scrum and the unfavourable camera geometry increase the size of rendering artefacts.

A second point to note is that the Football 2 data set has only five cameras that view the entire scene and is therefore the data set that most closely approximates what can be achieved in the outdoor sports broadcast environment without the provision of any
additional specialist cameras. As a consequence, the analysis of this data set gives markedly worse results than the other techniques, and severe truncations can be seen in Figure 3.21.

The shape scores shown in Figure 3.16 are the most telling as to the general correctness of the different techniques. The results for the synthetic data show that no technique gives perfect results. In fact, with no errors in the system, the plain visual hull outperforms all the variants. However, as error is introduced into the data set, the error-tolerant techniques begin to outperform the visual hull. It is worth noting that in the Football 1 data set, which consists of many cameras with similar viewpoints, the consensus hull significantly outperforms the visual hull. However, in the Football 2 and Football 3 data sets this technique performs less well, as the assumption of redundancy in the input data breaks down.

Overall, it can be seen that the CVH, probabilistic visual hull and CCH all perform to similar levels. An exception is the Football 3 data set, where the resilience to complex matting errors elevates the CCH above the other techniques (similarly, for this data set the tri-state visual hull significantly outperforms the visual hull).

Finally, the shapes of these plots can also tell us something about the nature of the errors in the different techniques. As $r$ is increased the score will rise quickly, with small local errors falling within the modelled error. Larger errors will also gradually be accounted for by the error model, but this happens slowly as protrusions or voids are eroded one pixel at a time. Thus, a reconstruction technique with widespread but small local errors (such as shifting the entire image a pixel to the right) will give a plot that increases sharply with $r$. Alternatively, a technique with gross errors will increase slowly with $r$. Typically any reconstruction will contain a mixture of local and gross errors, and the shape of the plot can reveal the prevalence of these errors in a reconstruction. For example, in the Football 1 data set, the visual hull suffers mainly from gross errors, while the CCH reduces these gross errors but introduces more local error. In the Football 2 data set, all techniques suffer mainly from gross errors (in this case significant truncations) and so the curves are flatter for this data set than the others.
Figure 3.25: Three views of a cube which has been textured with the view-dependent appearance of a blue sphere. The left and right views are from the viewpoint of the projecting cameras and hence the appearance is correct. The central view is interpolated between the two source views. The difference in shape between the cube and the sphere, as well as the fact that the cube is larger than the sphere it is replacing, produce a gap between the appearance projected from the right and left cameras.

It can be seen from the graphs shown in Figure 3.17 that there is much less difference between the techniques in appearance than there is in shape. The differences are most marked in the Rugby 1 data set, due to the more challenging view interpolation. It can be seen here that the visual hull performs better at preserving appearance than any of the error-tolerant techniques. This is due to the view dependent texturing algorithm used. As can be seen in Figure 3.24, the extreme angle between the viewpoint and the reconstruction means that the inflation of the volume caused by the error-tolerant techniques separates the texture mapped from the nearest cameras such that they no longer align. This can be so extreme as to create a gap between the texture mapped from the different cameras (seen as a dark band through the centre of some players). This effect is explained in Figure 3.25.

3.8 Summary

In this chapter, an explicit model of the resultant errors in reconstruction is presented, based on a taxonomy of all possible errors. Errors are divided into errors in shape and errors in appearance, and a quantitative analysis based on this error model is presented.

Several modifications to the visual hull algorithm are then presented. The tri-state
3.8. Summary

visual hull reduces errors due to matting, the conservative visual hull reduces errors due to inaccuracies in calibration estimation, and the consensus hull reduces errors due to gross failure in calibration and matting. The combination of these techniques, the conservative consensus hull (CCH), is introduced and contrasted with an existing probabilistic visual hull algorithm. The chapter concludes with a quantitative analysis of the techniques using the derived shape and appearance measures.

The techniques presented in this chapter can be seen as attempts to exploit the global consistency of a scene representation. The next chapter considers the improvements to reconstruction that can be attained by considering local consistency between smaller sets of cameras.
Chapter 3. Error-Tolerant 3D Reconstruction
Chapter 4

Shape Optimisation

4.1 Introduction

The previous chapter demonstrated how shape recovery using shape from silhouette techniques can be improved by modelling the various errors in the 3D reconstruction pipeline. This chapter examines a technique for refining the reconstruction produced when only a few, low-resolution cameras are used to reconstruct a scene. While visual-hull-based techniques can generate a good estimate of scene geometry in this scenario, they are not detailed enough to provide artifact-free rendering of free-viewpoint video (FVV). In fact, as shown in Figure 4.1, even in the absence of matting and calibration errors the camera geometry is such that the visual hull is a poor estimate of the scene geometry. This makes high quality rendering from the visual hull alone impossible without the addition of many more cameras. Therefore, in order to generate a surface of sufficient fidelity to accurately reproduce views of a complex scene, some form of shape refinement must be employed.

This chapter proposes that techniques that combine shape-from-silhouette with refinement using snakes are particularly suitable for this type of reconstruction. A novel technique is proposed that uses a deformable model to combine multi-view segmentation with shape and stereo optimisation across the 3D reconstruction of a scene. The technique uses only video from a set of standard calibrated cameras, can handle
multiple self-occluding objects and is error-tolerant with regards to initial scene segmentation and poor camera calibration. The novel elements of the technique are the formulation of the silhouette term, which avoids the need for an initial high quality image segmentation, and the two-phase use of the deformable model which reduces the dependence on parameterisation.

The deformable model formulation presented in this chapter was first published by Kilner et al. in the paper “Dual-mode deformable models for free-viewpoint video of sports events” [57] and the analysis is an expansion of that presented by Kilner et al. in “Objective quality assessment in free-viewpoint video production” [55].

4.2 Surface Refinement for Free-viewpoint Video

4.2.1 Free-viewpoint Video Rendering

Two classes of data are used in the field of rendering for FVV; global and view-dependent. Global data is invariant with the virtual camera position and orientation, whereas view-dependent data relies on the virtual camera location. When rendering the 3D reconstruction of the scene, the above distinction leads to three different classes of rendering technique; global geometry and texture, global geometry and view-
dependent texture, and view-dependent geometry and texture. FVV techniques make use of view-dependent rendering to overcome inaccuracies in scene reconstructions generated from a limited number of camera views. View synthesis is performed either by sampling appearance from a subset of cameras near a virtual view in order to minimise correspondence error [18], or by deriving a view-dependent geometry to optimise correspondence in the camera set used in rendering [36].

Rendering with global geometry and texture is a standard technique in computer graphics. However, with the errors inherent in the outdoor sports broadcast environment, it is not possible to generate a single, textured geometry suitable for artifact-free rendering in all views. As such, this class of technique is not considered.

The term “view-dependent texturing” was introduced by Debevec et al. [18] and refers to the technique of blending together multiple texture maps, based on the orientation of the virtual camera relative to the surfaces of the mesh. Often, multiple close textures are chosen and these are then blended on the surface, with their blend weights dependent on the angular distance between the viewing rays of the virtual camera and the viewing rays of the original cameras that generated the source images. Applying a view-dependent texture to a static mesh provides a technique that uses global geometry and view-dependent texture. The visual hull techniques evaluated in the previous chapter fall into this class of rendering.

Finally, you can modify both the geometry and the texture based on the viewing angle. This extends the concept of view-dependent texturing into the geometry domain and entails choosing the geometry to display (along with the texture to apply) based on the virtual camera location. This is the rendering strategy adopted by various view-dependent surface techniques which select the most relevant meshes and combine them along with their relevant images to generate a virtual view [71, 36]. Another example of this class of technique is where the geometry itself is computed based on the location of the virtual camera. In its simplest form this gives us one version of billboarding where a single polygon is rotated to face the virtual camera and textured using the various camera images.
4.2.2 Surface Refinement

Surface refinement describes those techniques where an initial surface geometry is optimised to improve its correspondence with the source images of the scene. Unlike surface recovery techniques, surface refinement techniques are often iterative in nature and require initialisation with an approximate surface. As such, these techniques can incorporate more complex constraints such as self-occlusion and surface normal constraints[94]. The disadvantages of these techniques are high computational cost, the need for an approximate initialisation of the surface and complex parameterisation.

Surface refinement using active contours (or snakes) as introduced in Chapter 2, is a commonly used technique. Such methods seek to balance data driven cues as to the true surface's location against a priori knowledge of the surface properties of the object (such as smoothness constraints). Deformable models are typically used in the studio environment to generate a global geometry which can then be used with view-dependent texturing for high quality view synthesis[93].

These techniques rely on the alignment of cues in all input cameras, i.e. that a surface exists which will project onto refinement cues in all cameras. In the presence of calibration errors, this may not be true or it may be that a surface which reprojects to refinement cues in all cameras is not a good approximation of the scene geometry.

4.3 Dual-mode Deformable Model

A novel technique which uses surface refinement techniques to generate view-dependent geometry is now presented. A separate geometry is generated for each of the recorded viewpoints using stereo information for the adjacent cameras in the dataset. Unlike previous techniques, the various view-dependent geometries share a common connectivity, making it possible to blend geometry between views.

As previously mentioned, the presence of errors in calibration and matting mean that there may be no globally optimal solution to the reconstruction of the scene. However, by optimising solely for one view, we can calculate a geometry that is refined for
just that view. In this way, we can improve the accuracy of shape and appearance without sacrificing completeness. By optimising for multiple shapes from the same global initialisation, we can generate a set of view-dependent meshes with constant connectivity. The requirement for a consistent connectivity means that a deformable-model-based snake is used rather than a level-set-based technique.

Snakes suffer from two problems. Firstly, there is the question of locality - how far from the snake’s current position do you look for data to drive the snake? Too small a band, and the model will be too constrained by the initialisation, possibly unable to consider a deformation large enough to bring the surface into agreement with the data, or trapped in a local minimum that happens to be closer to the initialisation surface than the true surface. Too large a band, and the contextual information from the initialisation is lost and the model may evolve towards undesirable solutions.

Secondly, there is the question of parameterisation - how do you balance data forces against regularisation, and how do you balance various data forces against one another? Although both data and regularisation costs are united in the deformable model framework, there is no pre-defined relationship between these costs. Therefore, while solving for the data forces or the internal forces in isolation is a simple mathematical exercise, combining these terms requires arbitrary scaling parameters to be applied.

These two problems are of particular relevance in this application. As the initialisation error is of similar scale to the features in the data, a band large enough to take a poorly initialised surface from its initial position to the correct position is also large enough to take a well-initialised surface and move it to an undesirable position. The scaling of costs is even more problematic. Due to noise in the image and the variable lighting conditions, a set of values which balance to find the correct surface in one section of the image will not balance in another section, meaning it is not possible to use one set of parameters for the entire image.

To avoid these problems, a dual-mode deformable model is proposed within which each set of forces is given equal space, but where they are not balanced directly against each other. This algorithm uses the same deformable model in each of two modes, differing only in the parameterisation of the model. The first mode is a “search” mode which
addresses the issue of banding by performing a search through configuration space, seeking out a consistent set of data points for the reconstruction. Discarding weak data points and over-smoothing are not problems as the algorithm is searching for data in this mode. The second mode is a "fitting" mode where the deformable model is used in a more conventional manner, albeit with a much weaker regularisation force as we have already discarded the major outliers in the data.

### 4.3.1 Initialisation

Taking a "key" view with index $i$, the deformable model is optimised to reduce the energy for that view $E_i$ over the surface $S$ of the model. $S$ is represented by a simple triangle mesh composed of vertices $v$ and edges $e$. The initial estimate of $S$ is initialised with the mesh generated by a visual-hull-based reconstruction technique.

$E_i$ is composed of a data driven energy term $D_i$ and an internal elastic energy term $U$. These are combined using a weighting term $\beta$ such that:

$$E_i(S) = (1 - \beta)D_i(S) + \beta U(S). \quad (4.1)$$

$U(S)$ is the elastic energy of the mesh:

$$U(S) = \sum_{e \in S} l(e)^2 \kappa \quad (4.2)$$

where $l(e)$ is the length of edge $e$ and $\kappa$ is a stiffness constant.

$D_i(S)$ is a term expressing the data fitness of the surface. It is expressed as a per-vertex energy in terms of a vertex $v$'s most desirable local position $v'$:

$$D_i(S) = \sum_{v \in S} || v - v' ||^2 \quad (4.3)$$

$v'$ is calculated by maximising a per-vertex data score. This score is a silhouette fitness score for edge generating vertices, and a stereo score for all other vertices.

To calculate $v'$ we first determine the set of edge generators in the neighbourhood $W$ of $i$. A vertex is considered an edge generator when viewed from a camera with index
4.3. Dual-mode Deformable Model

$j$, by calculating the value $P_j(v)$, which is a measure of how perpendicular the unit vertex normal $\mathbf{n}$ is to a unit vector along the camera viewing direction $\mathbf{a}_j$:

$$P_j(v) = (1 - |\mathbf{n} \cdot \mathbf{a}_j|)^2$$  \hspace{1cm} (4.4)

By considering the set of cameras $W = \{i - 1, i, i + 1\}$ we can classify each vertex as being in $\mathcal{K}_i$, the group of edge generators for the view $i$:

$$\mathcal{K}_i(v) = \{i = \arg\max_{x \in W} P_x(v) \text{ and } P_i(v) \geq \lambda\}$$  \hspace{1cm} (4.5)

$\lambda$ controls the thickness of the strip of edge generators that is considered, and a value of $\lambda = 0.8$ was used. $v'$ can now be given as the location which maximises the data fitness term $F$, which is expressed in terms of a silhouette fitness score $G$ and a stereo score $T$:

$$v' = v + \delta \mathbf{n}$$  \hspace{1cm} (4.6)

$$F(v') = \begin{cases} G(v') & v \in \mathcal{K}_i \\ T(v') & v \notin \mathcal{K}_i \end{cases}$$  \hspace{1cm} (4.7)

$$\delta = \arg\max_{\delta} F(v + \delta \mathbf{n}) \quad r \geq \delta \geq -r$$  \hspace{1cm} (4.8)

$\delta$ is determined by sampling along $\mathbf{n}$ within some range $r$ to maximise $F(v')$. $v'$ is therefore the projection of the strongest local data cue onto the line $v + \delta \mathbf{n}$. $F$ is thresholded to suppress noise in regions of low image structure, and it may be that for some $v$ no value of $F(v')$ exceeds this threshold. In this case, it is said that no valid local data cue exists for $v$ and values of $\delta = 0$ and $v = v'$ are therefore assigned.

$G$ is a term representing the silhouette matching score for the projection of $v'$ into image $i$, and $T$ is a term representing the stereo matching score for regions around the projection of $v'$ into the most appropriate cameras in $W$. Similar terms are used in other deformable-model-based work, such as that by Hernandez et al. [40].

$T$ calculates the correlation between the reprojections of the source images onto a surface placed at the location $v'$. We consider a pixel $p$, the projection of $v'$ into $J_i$. 
The normalised cross correlation function \( \text{ncc} \) takes a 3x3 pixel window centred around \( p \), and reprojects each pixel in this window into \( J_j, j \in W \). The reprojection is done using the planar geometry determined by the location \( v' \) and \( n \), the normal to the surface at \( v \). This geometry is shown in Figure 4.2.

The reprojection of a pixel \( q \) in \( J_i \) to image \( J_j \) yields a pixel \( q' \). By considering the values of \( q \) and \( q' \) obtained through the reprojection of all pixels in the 3-neighbourhood of \( p \) \( (N_3(p)) \), the normalised cross correlation can be formulated as:

\[
\text{ncc}(i, j, v') = \frac{1}{8} \sum_{q \in N_3(p)} \frac{(c(q) - \mu)(c(q') - \mu')}{\sigma \sigma'}
\]

(4.10)

where \( \mu \) and \( \sigma \) are the mean and variance over the sampled values of \( c(q) \) and \( \mu' \) and \( \sigma' \) are the mean and variance over the sampled values of \( c(q') \). \( T(v') \) can then be formulated as the average of the correlation of all cameras in the set of neighbours \( W \). The correlation is made symmetric by averaging \( \text{ncc}(i, j, v') \) and \( \text{ncc}(j, i, v') \) to account for sampling differences in the input cameras.

\[
T(v') = \frac{1}{|W| - 1} \sum_{j \in W, j \neq i} \frac{\text{ncc}(i, j, v') \text{ncc}(j, i, v')}{2}
\]

(4.11)

While the formulation of \( T \) is similar to the stereo score used in other deformable model formulations\[40\], we do not have accurate silhouettes to use for the normal formulation of our silhouette score \( G \). This problem is addressed by formulating the silhouette energy in terms of image gradients. It can be noted that silhouette shape can be determined from the gradient of a matte \( (\nabla \alpha) \), as the silhouette boundary occurs where \( \nabla \alpha \) is maximised. This allows a formulation of the silhouette energy in terms of the image gradient \( (\nabla J) \), using the following approximation introduced by Sun et al. \[95\]:

\[
J = \alpha \mathcal{F} + (1 - \alpha) \mathcal{B}
\]

(4.12)

\[
\nabla J = (\mathcal{F} - \mathcal{B}) \nabla \alpha + \alpha \nabla \mathcal{F} + (1 - \alpha) \nabla \mathcal{B}
\]

(4.13)

\[
\nabla \alpha \approx \frac{1}{\mathcal{F} - \mathcal{B}} \nabla J
\]

(4.14)
4.3. Dual-mode Deformable Model

Figure 4.2: The calculation of the stereo score using normalised cross correlation between a window of 3x3 pixels around the projection of $v'$ into $I_i$ and the corresponding region after reprojection into image $I_j$. Projection is done using the geometry defined by the cameras $C_i$ and $C_j$ and the plane defined by the location $v'$ and direction $\vec{n}$.

Equation 4.12 is the classical matting equation in terms of an image $J$, its foreground $F$ and background $B$, and Equation 4.13 is its first derivative. If $F$ and $B$ are taken to remain constant (or nearly constant such that $\nabla F$ and $\nabla B$ are small), then this yields Equation 4.14 which states that the rate of change of $\alpha$ is proportional to the image gradient. As high $\nabla \alpha$ occurs at silhouette boundaries, it can be seen that in regions where $F$ and $B$ are constant, silhouette boundaries coincide with regions of high $\nabla J$ and hence silhouette shape can be determined without explicitly calculating $\alpha$.

Thus we can define $G$ as simply:

$$G(v') = |\text{sobel}(p)|, C_i : v' \mapsto p$$  \hfill (4.15)

where sobel is the standard Sobel (discrete image gradient) operator centred at pixel $p$ where $p$ is the projection of $v'$ into image $I_i$.

Both $G$ and $T$ are thresholded to avoid noise. Due to the nature of the snake algorithm used, these thresholds are not particularly sensitive and can be set by inspection from the average edge intensity and standard deviation in the source images.
An optimisation over the mesh to minimise $E_i(S)$ is performed using conjugate gradient descent. The step length in conjugate gradient descent is defined by performing a line search using the back-tracking algorithm.

The deformable model is initially used as a banded search through the solution space. Due to the conservative nature of the initialisation, the desired solution is taken to lie within or near to the initialisation surface. The surface is therefore allowed to collapse under the internal elastic energy of the mesh. As it collapses, it searches for regions that agree with the input data.

As initialisation is poor, a value for $\beta$ is chosen (typically 0.1), so that regularisation effects dominate during this phase. This allows the model to evolve in a smooth fashion, collapsing from its initial state to a smaller shape. As $D_i(S)$ is not zero (even though it is small), the strongest data terms affect this evolution, such that the collapsed mesh represents the strongest features of the object. If the mesh was simply treated as a normal snake with strong regularisation, this would pull the surface away from the relevant image regions as shown in Figure 4.3. In order to avoid this, $v'$ is not re-evaluated at every iteration.

If $v'$ was never re-evaluated, the model would fix onto any edges that happened to fall near the initialisation, and this is equally undesirable. Hence, a technique of selective updates is used to determine when to update $v'$ for a given vertex. If $N_v^\alpha$ is the $\alpha$-neighbourhood of $v$ (i.e. all vertices which can be connected to $v$ traversing $\alpha$ or less edges), then the local support $L(v)$ is measured as:
4.3. Dual-mode Deformable Model

Figure 4.4: Local support for vertex location cues. The vertices in the deformable model are shown as grey dots, and their most desirable locations as light grey squares. The elastic energy (represented as arrows) attempts to draw the vertices inwards. Vertices with active location cues are shown being attracted towards their most desirable locations. Vertices $v_1$ and $v_2$ have strong local support. Therefore $v'_1$ and $v'_2$ are not re-evaluated, and so $v_1$ and $v_2$ are held in place. Vertex $v_3$ is on the edge of a region of local support. This vertex will re-evaluate its most desirable location $v'_3$ at every iteration. However, as it is not being strongly pulled away from its location, $v'_3$ will remain stationary. $v_6$ however has no local support. As the vertex and its neighbours are being strongly pulled away from that location, $v'_6$ will be re-evaluated and the vertex may latch onto another location cue.

$$L(v) = \frac{|Q_v^c|}{|N^c_v|}$$

(4.16)

where $Q_v^c$ is the subset of $N^c_v$ only containing those vertices with a valid current data cue. For these experiments $\alpha = 3$ was used. If $L(v) > \alpha$ then $v'$ is not updated, but if not, then a new value of $v'$ is calculated. If the vertex has not moved far from its previous position, then recalculation will yield the same value of $v'$. If however $v$ is pulled away from its previous position, then the previous value of $v'$ will fall out of the considered range and a new value will be calculated (see Figure 4.4).

In this way, the local support of the data fitness of the model is used to determine the
update. If a section of the model is driven by some data cues but cannot be integrated continuously with the rest of the model, then the model will update itself to discard the anomalous data cues and to seek new cues that are more consistent with the rest of the model. This does not compromise the model's ability to jump gaps in the data, as it is only where the regularisation is attempting to move vertices far from their previous positions that cues are discarded - if the cues form a smooth whole, then the model will not discard them. Thus data cues are only discarded if they are inconsistent with a smooth shape incorporating the majority of data cues. The process is terminated once variation between iterations falls below a certain level.

4.3.2 Fitting

Having selected the most appropriate values of \( \mathbf{v}' \) to use for the model, the data term is now allowed to dominate by relaxing the regularisation (\( \beta = 0.01 \)). The deformable model is re-initialised with the original vertex positions, but maintains the values of \( \mathbf{v}' \) discovered through the first phase. Returning the vertices to their original positions allows the model to better fit to detail that may have been passed over during the search phase.

Another known problem of snake-like techniques is that they will not move into concavities that are larger than the search band. In order to determine the correct final shape an "exploratory" or "ballooning" force must be used. Our exploratory force operates on those vertices that are part of \( \mathcal{X}_i \) but for which \( G(\mathbf{v}') \) is less than the specified threshold. This force is modelled by simply moving the relevant vertices inwards along the normal direction. This allows the deformable model a chance to find edges further inside the shape if they exist, but allows the deformable model's internal energy to pull the vertices back out to the surface if no such edges exist.

4.4 Results

An evaluation of the technique was performed using the odd numbered cameras from the Football 1 data set, providing a set of sparse, low resolution cameras. The recon-
4.4. Results

Figure 4.5: Three frames from the Football 1 sequence, left: CVH, right: dual-mode snakes.

Figure 4.6: Left and right: original images used in reconstruction, centre: reconstructed view.
Table 4.1: Analysis of foreground reconstruction on a variety of techniques. BB = billboards, VH = visual hull, VH w BM = visual hull with Bayesian mattes, CVH = conservative visual hull, and DMS = dual-mode snakes. Scores are shape, completeness, appearance and combined appearance with completeness.

<table>
<thead>
<tr>
<th>Technique</th>
<th>Shape</th>
<th>Completeness</th>
<th>Appearance</th>
<th>Combined</th>
</tr>
</thead>
<tbody>
<tr>
<td>BB</td>
<td>0.71</td>
<td>0.86</td>
<td>0.78</td>
<td>0.67</td>
</tr>
<tr>
<td>VH</td>
<td>0.77</td>
<td>0.83</td>
<td>0.94</td>
<td>0.78</td>
</tr>
<tr>
<td>VH w BM</td>
<td>0.75</td>
<td>0.80</td>
<td>0.94</td>
<td>0.75</td>
</tr>
<tr>
<td>CVH</td>
<td>0.27</td>
<td>0.98</td>
<td>0.84</td>
<td>0.82</td>
</tr>
<tr>
<td>DMS</td>
<td>0.56</td>
<td>0.95</td>
<td>0.91</td>
<td>0.85</td>
</tr>
</tbody>
</table>

The technique was compared against a number of alternative techniques: billboards[44], visual hull[63], visual hull using refined mattes, and the CVH described in the previous chapter. The refined mattes were generated by applying a dilation operation to the original segmentation to generate a tri-map, and then using Bayesian matting techniques[95] to refine the matting. A completeness score was also included in the evaluation. This is identical to the shape score, except it disregards regions only marked foreground in the synthetic view. In this way, it only penalises a reconstruction for missing elements compared to the ground-truth. Finally the Completeness and Appearance scores are combined by multiplying them together to give a score which takes into account both the accuracy of reconstruction and the amount of the scene reconstructed.
4.5. Discussion

Images from a set of frames from the input sequence were rendered and Table 4.1 shows the results of comparison against the hand-matted ground-truth images. Figure 4.5 shows example images from various frames. Figure 4.9 shows magnified views of a region where refinement has improved the reconstruction.

In addition to this comparative analysis, a full analysis of the dual-mode snakes technique against all data sets was performed following the methodology described in the previous chapter. This allowed a direct comparison against the results presented for the various visual hull based techniques. These results are presented in Figures 4.7 and 4.8. As can be seen, with improved visual hull estimation techniques, higher resolution images and improved matting, most of the benefits of the technique are negated, and the slight truncations introduced by the technique are exacerbated, leading to a reduction in the accuracy of the reconstructed shape without a significant corresponding increase in reconstruction accuracy. As such, this technique is only appropriate for situations where low quality approximate geometry is available.

4.5 Discussion

It can be seen that the dual-mode snakes technique improves the appearance of the reconstruction compared to directly rendering the CVH, when the CVH is parameterised to provide a fully complete reconstruction of the reduced Football 1 data set. The proposed technique improves the appearance of the reconstruction without significantly degrading the shape of the reconstruction.

Despite the dual-mode approach, the final shape is still strongly affected by the initialisation. The model can cope well with inaccurate placement of a roughly correct shape, but is unable to recover from gross errors in initialisation which are consistent with the source data (such as the inclusion of pitch lines in the initial mattes - as shown in Figure 4.9).

Concavities remain a problem for this technique, as can be seen in Figure 4.9. This is because the low resolution of the images and the presence of shadows and image bleeding can provide a strong edge across the mouth of the concavity that prevents the
Figure 4.7: Plot of shape score against $r$ for deformable model and CCH. Synthetic data is not considered in this analysis as rendering fails for several frames.
Figure 4.8: Plot of appearance score against $r$ for deformable model and CCH. Synthetic data is not considered in this analysis as rendering fails for several frames.
Figure 4.9: Reconstructions of the view from a camera between cameras 5 and 7. Top: crop from full render, bottom: detail. Left to right: ground-truth, billboards, visual hull, CVH, and dual-mode snakes.

surface from evolving inwards.

While the technique compares favourably with simple reconstruction techniques applied to poor input data, as the quality of the input data is increased and more sophisticated reconstruction techniques are applied, the improvements provided by the technique decrease. However, the reconstructions provided by the technique are still smoother than the hull techniques presented in the previous chapter and this can improve rendering in some situations. This can be seen in the slightly enhanced appearance scores when low values of $r$ are used in the evaluation. However, these effects are small and do not make a significant difference to the overall visual quality of the result.

4.6 Summary

This chapter demonstrates a technique that combines simultaneous multi-view shape extraction with stereo refinement to generate a view-dependent optimisation of an initial scene reconstruction. The dual-mode snake technique presented in this chapter shows improvements in the completeness of the reconstruction compared to the conservative visual hull when initialised with poor data. The technique is shown to improve both the shape and the appearance of the reconstruction compared to the conservative visual hull.

The technique is still susceptible to poor initial segmentation and to clutter in particularly noisy parts of the image (such as viewing the goalkeeper through the goal net),
which can lead to poor reconstruction even when clear views of the objects exist from other directions. Further work is required to improve the performance of the technique in these situations.

While this chapter demonstrates that some improvement to free-viewpoint synthesis can be made using local optimisation of reconstructed shape, the complexity of the scene limits the usefulness of this approach. In the next chapter, an attempt is made to make use of high level knowledge of the scene's constituent elements - humans - and exploit the a priori knowledge of their temporal coherence and dynamics to extract a high level scene model.
Chapter 5

3D Action Matching

5.1 Introduction

While it is possible to use reconstructed surfaces to render FVV as shown in the previous chapters, these techniques are severely limited by the errors in the camera calibration and the low quality of the input images. The previous chapter demonstrated that simultaneously exploiting global scene knowledge (through initialisation from the visual hull) and local knowledge (in the form of stereo refinement from a subset of cameras) can improve reconstruction quality. However, this is ignoring a significant body of knowledge about the scene - namely that the majority of the objects being reconstructed are humans engaging in sporting activities. This semantic knowledge can be exploited by processing the scene in terms of the human beings on the pitch, rather than as a generic surface. This allows us to generate a synthetic model of the recorded events, which in turn may be used to aid reconstruction refinement or directly as an alternative representation of the footage.

In this chapter, a technique is proposed to automatically generate synthetic proxies representing players on a pitch. Proxies performing an appropriate action are selected from a motion capture library and synchronised to the original action. These synthetic proxies can be used directly for analysis, as avatars in a virtual view of the event, or as priors for further scene reconstruction.
Figure 5.1: Images depicting the first two stages of processing. Top: the original image, bottom: a segmented "key" separating the players from the background.
Figure 5.2: Images depicting the second two stages of processing. Top: a 3D reconstruction of the players, bottom: the final synthetic proxies of the players.
Multi-camera video is recorded and a robust SFS technique is used to generate a 3D scene representation. This allows the fusion of multiple moving, zooming cameras, each of which may have been unsuitable for use on its own. Direct pose retrieval in this environment is severely challenged by the ambiguity in the relationships between the cameras. Therefore a hidden Markov model (HMM) is used to match both the pose and the action dynamics to a pre-defined library of human motion capture. The full process is illustrated in Figures 5.1 and 5.2.

When working with multiple cameras, there are two possible approaches to pose estimation. Either monocular pose estimators can be run on each view and combined in 3D to generate a result, or the various images can first be combined in 3D using the camera geometry, allowing pose estimation to be performed directly in 3D. When working in separate monocular views, care must be taken to handle all issues of varying viewpoints and image resolutions correctly. Similarly, as cameras come into and out of occlusion, disambiguating cues must be explicitly propagated from camera to camera, and disagreement between pose estimators working on different views must be handled in a coherent manner. If tracking is performed in 3D however, these tasks of sensor fusion and normalisation have already been carried out by the metric calibration of the cameras.

The technique presented in this chapter aims to discover the individual players in the scene and match them to synthetic representations drawn from an appropriate library of activities. As such, the process is divided into two stages - tracking and action matching. The aim of player tracking is to identify a temporally coherent 3D representation of each player, and the aim of action matching is to identify the pose of each player at every time instance. This chapter will start by describing the tracking technique used in this process. It will then describe the similarity measure used to match the player shapes against the recorded library of motions, and will conclude with a description of the action matching technique, together with the results of the application of the technique to real and synthetic data. The work presented in this chapter was previously published in the paper “3D Action Matching with Key-Pose Detection” by Kilner et al. [53].
5.2 Player Tracking

The CCH technique introduced in Chapter 3 is applied to the silhouettes and calibrations obtained from the images of the event that is being reconstructed. As no pre-processing is performed to segment or track individual players in the source images, this step produces a single discontinuous triangle mesh that represents all the foreground objects in the scene for a single frame. Thus, the first step of player tracking is to perform a connected components analysis of this mesh to divide it up into separate sub-meshes. Most of these sub-meshes will represent individual objects such as players, but some will be phantom volumes (a part of the visual hull that does not correspond to any real-world object), and in some cases player meshes will be joined together (as shown in Figure 5.3). It is therefore important that the tracking algorithm can handle both these types of erroneous input data.

The desired player tracks are extracted using the Viterbi algorithm[102]. First, a trellis is constructed with each vertex representing a connected component, or sub-mesh,
Figure 5.4: The trellis structure used to determine paths. Each node in the graph represents a sub-mesh from the reconstruction at time \( t \). Nodes are connected by edges, each of which has a cost determined by the sub-mesh similarity. Some nodes are not connected by an edge as they are sufficiently different to discount the possibility of a track (illustrated by dashed lines in the diagram). The Viterbi algorithm is then used to find the path through the trellis from time \( t = 0 \) to \( t = N \) which accumulates least end-to-end cost.

Extracted from the full mesh (i.e. each vertex represents a player, except for those players whose reconstructions have become connected due to their close proximity to each other, where a single vertex may represent multiple players). A row in the trellis therefore represents all sub-meshes in one frame of the reconstruction. To generate the trellis, the maximum number of sub-meshes per frame is calculated over the entire sequence, and this is taken to be the number of vertices in each row. The edges connecting vertices in subsequent rows represent the distance between sub-meshes in subsequent frames as shown in Figure 5.4. The cost associated with an edge connecting sub-meshes \( m_1 \) and \( m_2 \) is calculated using the following formula:

\[
\begin{align*}
V_{\text{sim}} &= 1 - (|v_1 - v_2| / \max(v_1, v_2)), \\
C_{\text{sim}} &= 1 - (|c_1 - c_2| / (|c_1| + |c_2|)), \\
O_{\text{sim}} &= O(bb_1, bb_2) / \max(v_1, v_2), \\
S_{\text{im}} &= (V_{\text{sim}} + C_{\text{sim}} + O_{\text{sim}}) / 3,
\end{align*}
\]

where \( v_x \) is the volume of \( m_x \), \( c_x \) is the centroid of \( m_x \), \( bb_x \) is the bounding box around
5.2. Player Tracking

Figure 5.5: Track showing the cost of successive paths extracted from the Viterbi trellis. The few tracks that are of indeterminate cost are generally partial tracks (where a player enters or exits the reconstruction area during the course of the reconstructed period).

$m_x$ and $O$ is a function returning the volume of the overlap between two bounding boxes. To reduce the complexity of the problem, steps which would produce a change in volume of more than 1000% are discounted, as are those which would produce a motion of more than 1m (the fastest recorded soccer strike was 50m/s[15] and footage is captured at 50fps). Where a frame does not contain enough sub-meshes to fully populate a row of the trellis, steps that lead to the unused vertices are also discounted.

The shortest path through the trellis is calculated using the Viterbi algorithm and the meshes associated with the nodes along this path are stored as a player track. Once a track has been generated, the minimum volume of the shape over the sequence is subtracted from the volume associated with each mesh on the path. In this way, multiple paths can share a single component, but the algorithm can still assess whether all objects in the scene have been sufficiently explained. The edge costs in the trellis are re-calculated and then subsequent tracks are extracted from the scene. The algorithm terminates when the score associated with the extraction of a path from the scene drops as shown in Figure 5.5. In practice, this is found by terminating the algorithm when the path score drops to under 1/4 of the maximum path score. The tracks produced by the algorithm are illustrated in Figures 5.6 and 5.7.

Trellis construction cannot take into account the dynamics of the extracted tracks (as the dynamics themselves depend on the path taken through the trellis), and as a result the algorithm can fail when two players collide or pass close to each other and
Figure 5.6: An example of the player tracks generated by the tracking algorithm over the course of the Rugby 1 data set (see Appendix A). Dashed and dotted lines are simply to differentiate tracks of the same colour. Note that where paths cross and multiple tracks share the same component (which consists of multiple players joined together), the tracks are distorted towards the median point of the two players.
5.2. Player Tracking

Figure 5.7: Examples of player tracks used as input to the shape matching algorithm. Note that connected components are not split, leading to some conjoined player meshes. In the top example the joined players (highlighted in red) do not affect the correct generation of the track. In the bottom example the more extreme case caused by the conjunction of three players and several phantom volumes results in tracks crossing over for two players.
their tracks become confused. The algorithm could be extended to include the colour model for the player re-projections, or modified to calculate the dynamics as the Viterbi algorithm is executed. However, in practice this is found to be a minor cause of error in the system.

5.3 Shape Similarity

Pose estimation attempts to recover the configuration of a human body from a set of observations. The presented technique determines the similarity between the 3D shape obtained from a multi-view reconstruction and a set of exemplar poses contained within a library. The best match is used as an estimate of the observed pose.

To determine the similarity between a candidate pose and the real data requires a measure of similarity between two different 3D shapes. To measure similarity, both a feature vector and a distance metric are required. These must be chosen such that they maximise the distance between shapes derived from differing poses, but minimise the distance between shapes derived from the same pose. The measure should be robust to the amount of reconstruction error expected in the system. In this section, a shape similarity metric is evaluated and shown to be robust to the errors in large-scale, multi-view reconstruction.

5.3.1 Feature Vector

Various shape descriptors have been proposed for 3D shape matching, including spin images[47], spherical harmonics[52] and shape histograms[2]. Huang et al. [43] investigated the use of these measures for matching 3D videos of people in the studio environment and proposed a volumetric shape histogram which is robust to small differences in shape while retaining rotational information.

For this work, volumetric shape histograms using a spherical co-ordinate system are used. The surface $S$ is isotropically scaled such that it lies within a sphere of radius 1 located at the origin and re-oriented per-frame such that the direction of motion of
5.3. Shape Similarity

the centroid is always along the Z axis. $S$ is then represented by an implicit function $V$ where $V(x) = 1$ when $x$ lies within $S$ and $V(x) = 0$ when $x$ is outside of $S$.

The shape histogram $H$ is then obtained as:

$$H_{i,j,k} = \sum_x V(x)B(i,j,k,x), \quad (5.5)$$

where $B$ is the bin-membership function:

$$B(i,j,k,x) = \begin{cases} 
1 & (i\delta r < r_x < (i + 1)\delta r) \\
1 & (j\delta \theta < \theta_x < (j + 1)\delta \theta) \\
1 & (k\delta \phi < \phi_x < (k + 1)\delta \phi) \\
0 & \text{otherwise}
\end{cases} \quad (5.6)$$

with $x$ expressed in spherical co-ordinates $(r_x, \theta_x, \phi_x)$ and with histogram quantisation steps $(\delta r, \delta \theta, \delta \phi)$.

5.3.2 Similarity Metric

In prior work on shape histograms, the Euclidean distance[43] and quadratic distance (introduced in the original work on shape histograms by Ankerst et al. [2]) were used. These measures perform well in the shape matching environments for which they were demonstrated (matching studio-captured 3D data and synthetic protein models), however both measures perform poorly in the presence of noisy data such as that generated in the outdoor sports reconstruction environment. As such, an evaluation of multiple metrics was performed to determine the most appropriate technique for use.

Several distance measures were evaluated against ground-truth to determine the most suitable matching score. The measures considered were the histogram intersection $I$, Euclidean distance $E$, quadratic distance $Q$, Mahalanobis distance $M$, chi-squared measures $\chi^2_1$ and $\chi^2_2$ and the Kullback Leibler Divergence (KLD) $K[59]$. These measures
are given by the following equations:

\[
I(h, e) = \sum_i \min(h_i, e_i), \tag{5.7}
\]

\[
E(h, e) = \sqrt{(h - e)(h - e)^T}, \tag{5.8}
\]

\[
Q(h, e) = \sqrt{(h - e)Q(h - e)^T}, \tag{5.9}
\]

\[
M(h, e) = \sqrt{(h - e)\Sigma(h - e)^T}, \tag{5.10}
\]

\[
\chi^21(h, e) = \sum_i (h_i - e_i)^2/e_i, \tag{5.11}
\]

\[
\chi^22(h, e) = \sum_i (h_i - e_i)^2/(h_i + e_i), \tag{5.12}
\]

\[
K(h, e) = \sum_i h_i \log(h_i/e_i), \tag{5.13}
\]

where \( h \) and \( e \) are histograms of the same dimensions and \( i \) is an index over all histogram bins, \( Q \) is the quadratic distance matrix which encodes the Euclidean distance between the centres of the histogram bins and \( \Sigma \) is the covariance matrix of all the data. For those measures which are asymmetric (such as \( K, \chi^21 \) and \( \chi^22 \)), a symmetric measure was used:

\[
X_{sym}(a, b) = 0.5(X(a, b) + X(b, a)). \tag{5.14}
\]

As all shape histograms are calculated in a metric space, the Euclidean measure was calculated without any prior normalisation. This was to avoid those artefacts which alter the total volume of the shape (such as protrusions and truncations) having an undue effect on the matching score. Instead, probabilistic measures were introduced which perform normalisation in a more sophisticated manner.

### 5.3.3 Evaluation

Several tests were carried out to determine the most suitable measure for use in the sports scenario. The tests were designed to simulate the kinds of actions and noise that are typical of the target data. To this end, a motion capture library of 16 motion sequences was used. Further details of the motion capture library used can be found in Appendix B.
5.3. Shape Similarity

The library of motion-captured animations was used to generate animated meshes from which a library of shape histograms \( L = \{L_0 \ldots L_{|L|}\} \) was generated. From this library, a sub-sequence \( S = \{S_0 \ldots S_{|S|}\} \) was chosen to use for comparison. A sequence of shape histograms with synthetic noise added \( N = \{N_0 \ldots N_{|S|}\} \) was then generated from the same source meshes as \( S \).

To generate \( N \), each mesh was distorted using the following technique. The mesh was voxelised and the voxelisation \( v_{\text{orig}} \) was dilated by \( \alpha \) pixels to give voxel set \( v \). \( v_{\text{orig}} \) was then dilated by \( \beta \) pixels and \( v \) was subtracted to give voxel set \( v_2 \) - a shell around \( v \). Then \( \theta \) of the voxels in \( v \) were randomly set to zero, and \( 1.0 - \phi \) of the voxels in \( v_2 \) were randomly set to zero. Finally, \( v \) and \( v_2 \) were combined to give the new noisy voxelisation. Thus varying amounts of noise can be generated by altering parameters \((\alpha, \beta, \theta, \phi)\), simulating expansion of the visual hull by the conservative hull calculation \((\alpha)\), errors in hull calculation caused by matting \((\theta)\), and the presence of phantom volumes and clutter \((\beta \text{ and } \phi)\).

A qualitative evaluation was then performed by generating a set of similarity matrices (a matrix \( \Psi \) where \( \Psi_{A,B,C}(i,j) = A(B_i,C_j) \) for a given measure \( A \) and sequences \( B \) and \( C \) ). \( \Psi_{X,S,N} \) was generated for each measure \( X \) and was then compared to a ground-truth self-similarity matrix \( \Psi_{E,S,S} \) (i.e. comparing \( S \) to itself using the Euclidean distance \( E \) defined in Equation 5.8). As both \( S \) and \( N \) are derived from the same source meshes, the ideal measure should generate a matrix displaying similar structure to the ground-truth, with particular importance being attached to the main diagonal feature which corresponds to the distance of each frame from itself. These results allowed a comparison of the different measures as shown in Figures 5.8 and 5.9. It can be seen in these diagrams that with the addition of large amounts of noise to the system many of the similarity metrics start to display a strong row structure. This is indicating that, in terms of the chosen measure, the difference between the each of the noisy data samples becomes much greater than the difference between the clean samples against which they are being compared. In this way the similarity score becomes simply a property of the noisy sample \( i \) and so the matrix degrades to a sequence of rows. These diagrams show that the measure which best preserves the original structure in the presence of this noise is the KLD.
Figure 5.8: Similarity matrices demonstrating the performance of various similarity measures when comparing a noisy sequence to the original clean sequence. A good measure should preserve the structure shown in the ground-truth, particularly the strong diagonal feature.
5.3. Shape Similarity

Figure 5.9: Similarity matrices continued.
Chapter 5. 3D Action Matching

Figure 5.10: ROC curves for various 3D shape similarity measures in the presence of multi-view reconstruction errors. More discriminative techniques produce a curve which tends towards the upper left corner, less discriminative techniques tend towards a straight line from the bottom left to the upper right corner.

A quantitative evaluation was also performed. A similarity matrix $\Psi$ can be converted into a Boolean classification function $C^\Psi(i,j) = \Psi(i,j) \leq \tau$. A ground-truth classification $G = C^{0.2\sigma}_{\Psi_{E,S,L}}$ is then generated where $\sigma$ is the standard deviation of $\Psi_{E,S,L}$. If $C^{0}_{\Psi_{E,S,L}}$ was used as the ground-truth then only exact matches would be considered. However, it is not this behaviour which requires evaluation, but rather the ability of a distance to correctly discriminate library poses which are "close" to the query pose from those which are "far". By using a threshold of $0.2\sigma$, $G$ contains the matches between frames in $S$ and $L$ which fall within an acceptable threshold.

Each measure $X$ was then used to compare $N$ with $L$, and a set of classification functions $F^X_C = C^X_{\Psi_{X,N,L}}$ were generated. Comparing $F^X_C$ with $G$ gives a set of true positives (where $F^X_C(i,j) = true = G(i,j)$) and false positives (where $F^X_C(i,j) = true \neq G(i,j)$). By varying $\kappa$ from $\min(\Psi_{X,N,L})$ to $\max(\Psi_{X,N,L})$, a receiver operating characteristic
5.3. Shape Similarity

Figure 5.11: ROC curves showing the behaviour of the KLD measure as both the angular resolution and the number of shells are altered. For these ROC curves the “Light Distortion” parameters from Figure 5.10 were used.

(ROC) curve for each technique $X$ can be generated as shown in Figure 5.10.

The ROC curves agree with the qualitative analysis that, in the presence of multi-view reconstruction errors, the KLD provides the most appropriate measure of shape histogram similarity.

Figure 5.11 shows how the matching accuracy changes with the dimensionality of the shape histogram. The computational costs and memory requirements grow linearly with the size of the histograms. It can be seen that increasing the number of histogram shells only yields a small improvement, while larger gains can be made by increasing the angular resolution. A 5 by 6 by 12 histogram was used in this work, as that was empirically determined to capture pose information in sufficient detail while minimising computational costs.
5.4 Action Matching

The KLD allows shape to be matched between noisy data and a clean exemplar. However, ambiguities in the recovered signal and variation between the detail of the recorded action and the library motions mean that frame-by-frame matching generates poor results. In order to resolve these ambiguities, an HMM is used to match the library to the target sequence, taking into account the shape and dynamics of the sequence. The resulting synthetic sequence is then generated by concatenating animation segments from the library to match the shape and dynamics of the target sequence.

5.4.1 Hidden Markov Model

Pose interpolation is achieved by modelling the evolution of the motion as a Markov process represented in the standard way using a state transition matrix $T$ and an emission matrix $E$. Each frame of each animation in the library $L$ is a state in the model. The original structure of the animations is encoded into the state transition matrix $T$:

$$T(i, j) = \eta(i, j)\rho(i, j) - \alpha(1 - \eta(i, j))Z(i, j).$$

$$Z(i, j) = \frac{-\sqrt{K(L_i, L_j)} + \min_{k,l} \sqrt{K(L_k, L_l)}}{\max_{k,l} \sqrt{K(L_k, L_l)} + \min_{k,l} \sqrt{K(L_k, L_l)}}$$

$\rho$ is a function that returns a maximal value when $j = i+1$ and hence favours the natural playback of a library animation. However, non-zero values for $j = i$ and $j = i+2$ allow the animation to repeat or skip frames to match animations at different rates. $\eta$ is 1 if $i$ and $j$ are in the same animation (zero otherwise) and $\alpha$ is a small value that controls the rate of switching between animations. $\alpha = 0.1$ was empirically determined to be a suitable value - higher values tend to make the model choose a single animation when it would be better to switch, while lower values tend to encourage the model to switch animations excessively. The emission matrix is then defined by:

$$E(i, j) = -\sqrt{K(D_i, L_j)}$$
5.5. Results

Figure 5.12: Crop of players showing every 2nd frame over a rugby sequence. At each time frame, a crop from the original image set is shown on the left, a rendering of the CCH is shown in the centre, and the resultant synthetic model is shown on the right. The top four rows were generated by simply selecting the closest pose from the library, while the bottom four rows were generated using the full action matching scheme.

In this way the likelihood of an observation being related to a specific library state is determined by the KLD between the shape histograms of the observation and the library state, with low values of $K(D_i, L_j)$ yielding a high emission probability and vice-versa. $T$ and $E$ are normalised appropriately and the optimal state sequence is calculated using the Viterbi algorithm, maximising the product of the transition and emission probabilities for each frame in the sequence.
Figure 5.13: Crop of players showing every 5th frame over a football sequence. At each time frame, a crop from the original image set is shown on the left, a rendering of the CCH is shown in the centre, and the resultant synthetic model is shown on the right. The top four rows were generated by simply selecting the closest pose from the library, while the bottom four rows were generated using the full action matching scheme.
Figure 5.14: Crop of players showing every 10th frame over a rugby sequence. At each time frame, a crop from the original image set is shown on the left, a rendering of the CCH is shown in the centre, and the resultant synthetic model is shown on the right. The top six rows were generated by simply selecting the closest pose from the library, while the bottom six rows were generated using the full action matching scheme.
Figure 5.15: Source images and action matched synthetic models from the Rugby 1 data set.
Figure 5.16: Source images and action matched synthetic models continued.
5.5 Results

The technique was evaluated with the data sets presented in Appendix A, which consist of multiple recordings of both rugby and football data. The data was captured using a mixture of moving broadcast cameras and "locked off" static cameras which were added to the recording set-up specifically for use in this reconstruction.

The synthetic library matched to the recorded video is specified in Appendix B. Neither the motion capture nor the model are specific to this domain. It should be noted that domain-specific motion capture would possibly improve the quality of the matching, and use of properly rigged football/rugby player models would greatly improve the visual quality of the results.

An example frame is shown in Figure 5.15 with further results shown in Figures 5.12, 5.13 and 5.14. Figure 5.12 shows a sequence where the simple pose matching scheme works relatively well. The reconstruction is not too distorted and so modelling the dynamics simply adds some smoothness to the reconstruction. Figures 5.13 and 5.14 show sequences where the simple pose matching scheme fails due to significant errors in reconstruction. The technique tries to select poses which explain both the underlying structure due to the player's pose and the distortion caused by calibration error, resulting in the selection of incorrect poses for the majority of the sequence. Modelling of the dynamics through the action matching scheme greatly improves the result, as it weakens the influence of the distortions which tend to vary over much longer time-frames than the dynamics of the underlying activity.

For comparison, a result generated with a state-of-the-art pose tracking system[4] is shown in Figure 5.17. This system uses an annealed particle filter to fit a human body model (consisting of a hierarchy of articulated rectangles) to the silhouettes generated from the input cameras. A hypothesised pose is rendered into each of the input cameras and silhouette overlap is combined with a distance-map based score to evaluate the quality of the pose. The particle filter is then used to optimise this score and to track the pose over the sequence. However, the calibration and matting errors in the sports data mean that the technique fails to converge on a solution and quickly drifts away from a reasonable pose estimate.
Figure 5.17: Pose estimation for a single player. The top row shows the original images, the middle row shows a state-of-the-art pose estimation technique[4] and the bottom row shows the proposed 3D action matching technique. Due to calibration and matting errors, the traditional pose estimation technique is unable to recover the pose from the images.
A quantitative analysis of the results was also performed. Due to matting errors and occlusions in the original images, standard recall/precision measurements comparing the pixels in a virtual view against the pixels in the original mattes are not very meaningful. The lack of a ground-truth also precludes a direct quantitative analysis of the technique. Instead, every tenth frame of each sequence was examined and each player assessed as either a "hit" or a "miss". Players were considered an exact match if the pose was of an appropriate action and was as accurate as possible given the limitations of the library. Close matches and matches where left/right ambiguity resulted in the selection of a mirror image of the correct pose were also considered hits. Anything else was considered a miss, including poses which were close to correct but came from an inappropriate action. The rationale for these distinctions is that in an application for semi-automatic pose recovery, exact hits would be those results which require no manual intervention, "close" and "mirror" poses would require some slight manual intervention, and "misses" would require fully manual pose recovery.

To generate these results an application was created which presents the user with cropped images of a player together with a synthetic pose which they would then classify. The order of presentation of the different techniques was randomised to avoid bias in the grading. Example poses, together with their classifications, are shown in Figure 5.18. The results of the analysis are shown in Table 5.1 which compares simple pose matching and action matching.

5.6 Discussion

The results presented in the previous section show that in many cases the technique can successfully recover pose information simultaneously for multiple humans. The system executes on an Intel Core2Duo 6300 1.8GHz and is implemented using Python and NumPy. It can process 59 frames consisting of 26 players in 420 seconds, yielding an average processing time of 0.27 seconds per player per frame. When taking the CVH calculation and track splitting into account this increases to 1.18 seconds per player per frame. These times compare favourably with other pose matching systems such as the Balan et al. technique[4] used as a benchmark, which requires up to 5 seconds per
Figure 5.18: Examples of recovered poses and their classifications. a) Poses which are classified as “misses” due to their deviation from the true pose. b) Poses classified as “close” to the true pose. c) Poses classified as “mirror” versions of the true pose. d) Poses classified as correct representations of the true pose.
### Table 5.1: Evaluation of the generated pose estimates for shape matching and action matching.

**Pose Only**

<table>
<thead>
<tr>
<th>Sequence Name</th>
<th>Matches</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Exact</td>
<td>Near</td>
<td>Mirror</td>
<td>Hit</td>
<td>Miss</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Synthetic</td>
<td>38.5%</td>
<td>20.0%</td>
<td>1.5%</td>
<td>60.0%</td>
<td>40.0%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Synthetic Error</td>
<td>38.5%</td>
<td>12.3%</td>
<td>2.3%</td>
<td>53.1%</td>
<td>46.9%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Rugby 1</td>
<td>17.2%</td>
<td>11.3%</td>
<td>5.9%</td>
<td>34.5%</td>
<td>65.5%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Rugby 2</td>
<td>16.3%</td>
<td>17.0%</td>
<td>13.3%</td>
<td>46.7%</td>
<td>53.3%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Football 1</td>
<td>11.5%</td>
<td>10.6%</td>
<td>4.6%</td>
<td>26.7%</td>
<td>73.3%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Football 2</td>
<td>9.9%</td>
<td>9.4%</td>
<td>6.4%</td>
<td>25.7%</td>
<td>74.3%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Football 3</td>
<td>18.7%</td>
<td>11.5%</td>
<td>4.9%</td>
<td>35.2%</td>
<td>64.8%</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td>19.5%</td>
<td>12.5%</td>
<td>5.6%</td>
<td>37.6%</td>
<td>62.4%</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Action Matching**

<table>
<thead>
<tr>
<th>Sequence Name</th>
<th>Matches</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Exact</td>
<td>Near</td>
<td>Mirror</td>
<td>Hit</td>
<td>Miss</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Synthetic</td>
<td>80.0%</td>
<td>10.8%</td>
<td>0.0%</td>
<td>90.8%</td>
<td>9.2%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Synthetic Error</td>
<td>69.2%</td>
<td>20.0%</td>
<td>0.0%</td>
<td>89.2%</td>
<td>10.8%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Rugby 1</td>
<td>25.1%</td>
<td>13.5%</td>
<td>12.6%</td>
<td>51.2%</td>
<td>48.8%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Rugby 2</td>
<td>31.1%</td>
<td>13.3%</td>
<td>8.1%</td>
<td>52.6%</td>
<td>47.4%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Football 1</td>
<td>12.0%</td>
<td>11.5%</td>
<td>5.5%</td>
<td>29.0%</td>
<td>71.0%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Football 2</td>
<td>36.8%</td>
<td>9.9%</td>
<td>11.7%</td>
<td>58.5%</td>
<td>41.5%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Football 3</td>
<td>28.0%</td>
<td>7.1%</td>
<td>4.9%</td>
<td>40.1%</td>
<td>59.9%</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td>34.9%</td>
<td>12.3%</td>
<td>7.5%</td>
<td>54.6%</td>
<td>45.4%</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*Table 5.1* depicts the evaluation of the generated pose estimates for shape matching and action matching. **Exact matches** are matches where it is reasonable to conclude that no closer match exists in the library and the action is appropriate. **Near matches** are matches where the action is appropriate but the pose is not exactly the same, and **mirror matches** are where the left/right ambiguity has been resolved incorrectly.*
5.6. Discussion

Despite the fact that most of the animation in the Synthetic data set is identical to animations in the synthetic library, the simple pose matching performs poorly when applied to this data set. This is explained by the fact that even with perfect calibration and mattes, the taking of the visual hull with sparse, low resolution images and scene clutter, introduces significant reconstruction artefacts. Modelling the dynamics of the scene virtually eliminates these effects, as can be seen by the particularly high scores for the action matching technique in the “Synthetic” and “Synthetic Error” data sets. The errors that persist are mainly due to two complex animations present in the synthetic data set that were excluded from the library - a rugby tackle and a turning kick.

Results are particularly poor for the Football 1 data set. This data set is an extended sequence of play around the goal area, with multiple changes in the direction of play. As such, motion consists of many small, unusual motions, such as single steps sideways or backwards followed by a sudden change in motion. In addition to the challenging nature of the dynamics in the scene, the low resolution images, poor matting, and crowding mean that the initial reconstruction is amongst the poorest of all data sets, making it particularly challenging.

It should be noted that many of the failures common to both the “pose only” and “action matching” techniques are due to players performing actions which are not in the pose library. In many such cases, the system selects a pose close to the correct pose, suggesting that if the library contained more appropriate motions they would be selected. Examples of actions that caused failure in this way include walking sideways and kicking. Another cause of failure is excessive clutter. The system can perform well in the presence of occlusions in several cameras, but in cases where multiple players are occluded for long periods of time (creating a single 3D volume for multiple players), the system fails. While the current track generation algorithm can compute separate paths through these volumes (and hence generate separate shape histograms for each player), improved pre-processing could fully separate these volumes into individual players and hence improve results. Roughly 1/3 of failures were due to actions not being represented in the library, while a further 1/3 were due to excessive clutter in
the scene. Finally, left/right ambiguity was an issue in many cases (and indeed is often hard to resolve visually from the original images). In these cases, the system will occasionally switch modes as it changes from incorrect to correct representation of the left/rightness. Switching in this way either introduces an additional cycle or doubles the length of a cycle.

5.7 Summary

The technique presented in this chapter uses the symmetric Kullback Leibler divergence of shape histograms as input to a hidden Markov model which allows the retrieval of pose information from video of multiple people playing field sports such as football or rugby. The technique can make use of multiple noisy and disjoint input images to extract player pose information, even when the input data contains calibration and matting errors. The action library used is independent of camera pose and player appearance, allowing a single library to be used in all scenarios. The next chapter introduces a number of extensions to this technique which further improve the performance of the action matching algorithm.
Chapter 6

Action Matching: Extensions and Enhancements

6.1 Introduction

The previous chapter introduced the action matching framework for automatic pose estimation of multiple subjects in the outdoor sports broadcast environment. This technique is robust to noise and relatively computationally inexpensive, but is limited with regards to the number of actions that can be recognised.

This chapter introduces a number of extensions to the action matching framework which improve the matching performance of the technique as well as addressing issues with the scalability and generality of the motion library used for matching.

The extensions introduced are:

- a technique for key-pose detection to improve the matching of periodic activities
- a technique for animation summarisation that addresses issues of rate-dependence in the exemplar library, while generalising key-pose detection to non-periodic activities and reducing the computational complexity of the action matching technique
Chapter 6. Action Matching: Extensions and Enhancements

- a hierarchical Markov model which addresses the scalability of the model with respect to the exemplar library size

The work in this chapter is based on the papers “3D Action Matching with Key-pose Detection” [53] and “Summarised Hierarchical Markov Models for Speed Invariant Action Matching” [54].

6.2 Key-pose Detection

One important feature of action matching is the synchronisation of the synthesised action to the original activity. With cyclic activities this synchronisation can drift out of phase over time due to ambiguities in the input sequence. The majority of cyclic activities in the outdoor sports environment are ambulatory motions such as walking, jogging and running. Prior work has demonstrated that “scissor poses” are a highly characteristic feature in 2D video of people walking. Ramanan et al. [82] use rectangular chamfer template edge masks in a tree pictorial structure to detect pedestrians in images. The detector is templated on the scissor pose as this is found to be a pose distinctive to pedestrian targets, even when engaging in sporting activities such as ice skating and pitching a baseball. Similar work by Dimitrijevic et al. [20] uses chamfer distances from 2D templates in a Bayesian spatio-temporal pedestrian detection scheme. Templates are generated by rendering several 2D views of a 3D synthetic human model, animated using scissor poses derived from motion capture data.

Therefore, to improve the synchronisation of the action matching framework to the input data, a key-pose detection stage is added to the framework to detect “scissor poses” and aid synchronisation of the output animation to these detected events (or “detector hits” as they are known). The pose detector compares shape histograms of the left scissor pose \( Y_l \), the right scissor pose \( Y_r \) (see Figure 6.1), and the mean pose of the library \( Y_m = \sum_{i} L_{yi} / |L| \) to the input sequence of histograms \( D = \{D_0 \ldots D_{|D|} \} \) (where \( L = \{L_0 \ldots L_{|L|} \} \) is the library of shape histograms as defined in the previous chapter). These values are combined to generate a matching score \( u \) as given by:
6.2. Key-Pose Detection

Figure 6.1: Exemplar left and right scissor poses.

$$u(t) = 2K(D_t, Y_m) - K(D_t, Y_l) - K(D_t, Y_r)$$  \hspace{1cm} (6.1)$$

where $K$ is the Kullback-Leibler divergence introduced in the previous chapter. $Y_m$ provides a correction for changes in the matching score due to non-pose-related factors, such as changes in the volume of the CVH or gross errors including the truncation of limbs. This combined signal is then smoothed and the local maxima are taken as the set of detector hits $\mathcal{H}$. Example graphs comparing the signal to the recorded sequence

Figure 6.2: Example matching scores from real data with corresponding images from the recorded sequence. The grey vertical lines are the detected scissor poses.
Figure 6.3: Example matching scores from real data with corresponding images from the recorded sequence. The grey vertical lines are the detected scissor poses. In this example it can be seen that where the motion is ambiguous near the beginning of the sequence the detector generates some false positives.

can be seen in Figures 6.2 and 6.3. It is important to note when looking at these graphs that it is only the local variation in $u$ that is of interest. The absolute value of $u$ will naturally vary over time as the pose of the player is closer or further away from the target exemplars (despite normalisation by the mean score, some variation in the overall scores persists due to differences in the position of the upper limbs, inclination of the torso etc.). Large errors in the data will cause miss-fires in the detector, and sections where motion is not as distinct will fail to produce clear peaks causing the detector to miss scissor poses.

Due to the probability of false and missing detections, the detector hits are incorporated as soft constraints into the HMM used in the action matching scheme. These constraints are encoded by the addition of a “boosting” term to the calculation of the emission matrix $E$ (boosting is used in its simplest meaning in this context and has nothing to do with the machine learning technique of the same name).

\[
E(i, j) = -b(i, j) \sqrt{K(D_i, L_j)}
\]

(6.2)

where $b$ is given by:

\[
b(i, j) = \begin{cases} 
1.2 & (i \in \mathcal{H}, j \in \mathcal{H}_L) \\
1.1 & (i \pm 1 \in \mathcal{H}, j \pm 1 \in \mathcal{H}_L) \\
1 & (\text{otherwise}).
\end{cases}
\]

(6.3)

where $\mathcal{H}_L$ are those poses in the library manually identified as being scissor poses.
6.2. Key-pose Detection

Figure 6.4: A crop from the emission matrix overlaid with the optimal path obtained by the Viterbi algorithm using the unmodified HMM (left) and the HMM boosted with the key-pose detector (right). Circles are coloured red to highlight output states corresponding to frames marked as scissor poses in the synthetic library. The effect of the key-pose detector can be seen at frames 125 and 140 (frame number is displayed along the top of the graph). Without boosting, the model will tend to favour playback of the animation at its natural rate, key-pose detection can overcome this and improves the synchronisation to the original action.

In this way \( b \) boosts the emission probability of scissor pose states when a scissor pose is detected in the observation sequence. The detector hits \( \mathcal{H} \) encourage the model to fit to the detected poses and hence improve the synchronisation of running and walking actions. The result of this boosting process on the HMM can be seen in Figure 6.4.

6.2.1 Results

A quantitative analysis of the results is shown in Table 6.1 which demonstrates the improvement in matching using this technique over the simple action matching scheme presented in the previous chapter. Figures 6.5, 6.6, 6.7 and 6.8 compare several sequences, both with and without the key-pose matching, demonstrating the improvement in synchronisation of the synthetic results. As the technique exploits both forwards and backwards temporal consistency in the data set, the synchronisation is weakest near the start and end of the sequence. This can be seen in Figure 6.6 where the technique looses synchronisation near the end of the sequence. Also, when the motion consists of many short, complex activities chained together, as in the sequence shown in Figure 6.8, the technique fails to produce good results.
### Table 6.1: Evaluation of the generated pose estimates. The column headed “Baseline” gives the corresponding scores for the original action matching technique presented in Chapter 5.

<table>
<thead>
<tr>
<th>Sequence Name</th>
<th>Exact</th>
<th>Near</th>
<th>Mirror</th>
<th>Hit</th>
<th>Miss</th>
<th>Baseline Hit</th>
<th>Baseline Miss</th>
</tr>
</thead>
<tbody>
<tr>
<td>Synthetic</td>
<td>83.1%</td>
<td>8.5%</td>
<td>0.8%</td>
<td>92.3%</td>
<td>7.7%</td>
<td>90.8%</td>
<td>9.2%</td>
</tr>
<tr>
<td>Synthetic Error</td>
<td>76.9%</td>
<td>13.1%</td>
<td>0.0%</td>
<td>90.0%</td>
<td>10.0%</td>
<td>89.2%</td>
<td>10.8%</td>
</tr>
<tr>
<td>Rugby 1</td>
<td>51.5%</td>
<td>13.1%</td>
<td>3.2%</td>
<td>67.7%</td>
<td>32.3%</td>
<td>51.2%</td>
<td>48.8%</td>
</tr>
<tr>
<td>Rugby 2</td>
<td>43.0%</td>
<td>18.5%</td>
<td>3.7%</td>
<td>65.2%</td>
<td>34.8%</td>
<td>52.6%</td>
<td>47.4%</td>
</tr>
<tr>
<td>Football 1</td>
<td>18.0%</td>
<td>15.7%</td>
<td>8.3%</td>
<td>41.9%</td>
<td>58.1%</td>
<td>29.0%</td>
<td>71.0%</td>
</tr>
<tr>
<td>Football 2</td>
<td>49.1%</td>
<td>9.4%</td>
<td>5.8%</td>
<td>64.3%</td>
<td>35.7%</td>
<td>58.5%</td>
<td>41.5%</td>
</tr>
<tr>
<td>Football 3</td>
<td>34.6%</td>
<td>11.0%</td>
<td>4.9%</td>
<td>50.5%</td>
<td>49.5%</td>
<td>40.1%</td>
<td>59.9%</td>
</tr>
<tr>
<td>Total</td>
<td>48.2%</td>
<td>12.8%</td>
<td>4.1%</td>
<td>65.1%</td>
<td>34.9%</td>
<td>54.6%</td>
<td>45.4%</td>
</tr>
</tbody>
</table>

Figure 6.5: Crop of players showing every 10th frame over a football sequence. At each time frame, a crop from the original image set is shown on the left, the result of the simple action matching scheme is shown in the centre, and the result of the boosted action matching scheme is shown on the right.
6.2. Key-Pose Detection

By detecting the major periodic signal in the captured shape data, this technique improves the synchronisation of the model to the recorded action. For example, the pure action matching technique can sometimes become "confused" by the left/right ambiguity inherent in the data. By encoding the requirement that the completion of a full animation cycle must pass through two scissor poses, key-pose detection improves the robustness of the technique to these ambiguities. The real benefit of this technique is seen in the resulting video sequences, which are smoother and better synchronised with the original activity.

There are, however, still a significant number of errors in the synthesised representations. These errors are mainly caused by the problems that affect the plain animation matching technique - namely activities that are not represented in the motion library and poor reconstruction of the 3D shape. The errors introduced by erroneous scissor pose detections are negligible compared to the errors caused by the underlying technique. However, misdetections may become more significant if the recorded actions do not consist mainly of ambulatory motions.
Figure 6.7: Crop of players showing every 20th frame over a rugby sequence. At each time frame, a crop from the original image set is shown on the left, the result of the simple action matching scheme is shown in the centre, and the result of the boosted action matching scheme is shown on the right.
6.3 Animation Summarisation

This technique is not particularly extensible. Attempts to introduce exemplars for other poses such as jumping and kicking produce poor results, as these poses are not sufficiently distinctive to generate an unambiguous set of detections. As such, this technique is limited to use with cyclic ambulatory motions such as running and walking.

6.3 Animation Summarisation

The addition of a key-pose detector to the action matching scheme has been shown to improve the performance of the technique in matching football and rugby data to an exemplar library. However, this is not a general solution (it specifically targets cyclic ambulatory activities) and does not scale well to the addition of extra key-poses. Therefore, a more general approach is required to extract key features for use in matching. Animation summarisation provides such a general technique and also helps significantly reduce the complexity of the matching.

As presented in the previous chapter, action matching attempts to model an input sequence in terms of a set of library actions. These actions are represented as states within an HMM. The simple approach used is to represent each frame of the sequence as a separate state in the model, however this causes two problems.
Firstly, many of the states are similar. As the likelihood of an observation originating from a state is related to the distance between feature vectors, many states that are close to a given observation will lead to similar observation likelihoods and the model becomes ambiguous.

Secondly, the structure of an animation (the sequential ordering of the frames) is encoded in the transition function \( T(i, j) \), which gives the probability of transitioning from state \( i \) to state \( j \). In order to encode the sequential nature of an animation, \( T \) should return a high value when \( j \) follows directly from \( i \), and a low value otherwise. However, strictly enforcing this constraint (e.g. \( T(i, i + 1) = 1 \) otherwise \( T(i, j) = 0 \)) also rigidly encodes the recording rate. So if the library contains a cyclic run animation with a period of 20 frames, the model will only match this to someone running with a period of 20 frames. Weakening the constraints (i.e. \( T(i, i + 1) = 1 - \delta \) otherwise \( T(i, j) = \delta \)) allows for repeating and skipping frames. The general technique of matching two similar sequences recorded at different rates is known as time-warping. Linear programming techniques are often used to perform time-warping of simple non-repeating structures, however they can not deal with arbitrarily looping structures. Hidden Markov models are often used to perform time-warping in such cases, however they are limited by the exponential accumulation of self-transition probabilities (the probability of remaining in state \( i \) for one time step is \( T(i, i) \), for two steps is \( T(i, i)^2 \), for three is \( T(i, i)^3 \) and so on). In this way small self transition probabilities quickly become vanishingly small, and large self-transition probabilities dominate the structure of the model. This high sensitivity to self-transitions can easily lead to the model settling on degenerate solutions involving excessive skipping or repeating of frames. Finding a balance between these two extremes will depend on the specifics of the data being analysed, the particular distribution of activities being observed, and the rates of those activities, which is information not typically available a priori.

One solution to these problems is to break the animation down into a discontinuous, high level representation that seeks to capture only the salient features of the action. By representing an action in these terms, the tight coupling to the recording rate is broken and the number of similar states is greatly reduced.
6.3. Animation Summarisation

Figure 6.9: An animation summary. The bottom row is the original animation. The top row shows the key-frames in the summarised representation. The grey boxes indicate the frames of the animation represented by each key-frame. The blue bars above each frame are a plot of the distortion introduced by representing the frame with the respective key-frame.

Summarisation is the process of representing an animation by a subset of its original frames, known as key-frames. This summary is chosen so as to be both a minimal representation of the animation (i.e. as few key-frames as possible) and to be a complete representation (i.e. all salient parts of the animation must feature in the summary). Key-frame extraction for video summarisation has long been studied in the field of video analysis and retrieval - a review of the state-of-the-art can be found in Barbieri et al. [6]. As data has progressed from 2D to 3D, the concept of summarisation has also been adapted to 3D in work carried out by Huang et al. [42].

A summarised animation represents the entire animation by a sub-set of its frames known as the key-frames (as shown in Figure 6.9). If an animation consists of \( n \) frames \( f_0 \ldots f_{n-1} \), then the summary consists of a set of key-frames \( \nu \) (\( 0 < |\nu| \leq n \)) and a mapping \( \mu \) between original frames and key-frames (\( \mu : i \mapsto \nu, 0 \leq i < n \)). Values for \( \nu \) and \( \mu \) are chosen which minimise the rate \( r \) and the distortion \( d \) of the summary where:

\[
\begin{align*}
\text{rate } r & \propto |\nu| \\
\text{distortion } d & = \sum_{i=0}^{n} X(f_i, f_{\mu(i)})
\end{align*}
\]

with \( X \) being some distance metric which measures the difference between a frame \( f_i \) and a key-frame \( f_{\mu(i)} \). This minimises both the number of key-frames and the difference between the key-frame representation and the full representation. This process can be seen as a specialised form of clustering where clusters are constrained to only contain adjacent frames.
Figure 6.10: Three example summaries. 20 key frames are shown. For the first two rows this represents approximately 100 frames of animation. However, as the third player is much more stationary, the third row represents approximately 180 frames. As can be seen, the representation is greatly reduced in size while keeping the salient features of the activity.

The proposed technique uses the KLD between shape histograms as the distance metric $X$, and measures the rate as $r = |\nu|$. A brute force search is then performed to determine the optimal values for $\nu$ and $\mu$ such that:

$$\nu, \mu = \arg\min_{\nu, \mu} (|\nu| \sum_{i=0}^{n} K(f_i, f_{\mu(t)}))$$ (6.6)

Each key-frame then represents an interval in the animation, and a direct mapping between intervals in the library and recorded sequences can be established. This 3D animation summarisation technique is a variant of that introduced in [42] which used the euclidean distance as the distance metric.

An HMM is then constructed as in the standard action matching technique presented in the previous chapter, except model states and observations now correspond only to key-frames instead of to each recorded frame. In this way, a smaller HMM which allows for variations in the rate of motion can be constructed, while retaining the temporal relationships in the data.

6.3.1 Results

A quantitative analysis of the results for this technique can be seen in Table 6.2. While performing better than simple pose matching, these results are significantly worse than for the plain action matching scheme. However, it should be noted that the reduction
6.3. Animation Summarisation

<table>
<thead>
<tr>
<th>Sequence Name</th>
<th>Matches</th>
<th>Baseline</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Exact</td>
<td>Near</td>
</tr>
<tr>
<td>Synthetic</td>
<td>40.0%</td>
<td>20.0%</td>
</tr>
<tr>
<td>Synthetic Error</td>
<td>33.8%</td>
<td>23.1%</td>
</tr>
<tr>
<td>Rugby 1</td>
<td>26.6%</td>
<td>13.8%</td>
</tr>
<tr>
<td>Rugby 2</td>
<td>32.6%</td>
<td>16.3%</td>
</tr>
<tr>
<td>Football 1</td>
<td>8.3%</td>
<td>5.1%</td>
</tr>
<tr>
<td>Football 2</td>
<td>27.5%</td>
<td>9.4%</td>
</tr>
<tr>
<td>Football 3</td>
<td>23.1%</td>
<td>7.1%</td>
</tr>
<tr>
<td>Total</td>
<td>25.9%</td>
<td>12.7%</td>
</tr>
</tbody>
</table>

Table 6.2: Evaluation of the generated pose estimates. The column headed “Baseline” gives the corresponding scores for the original action matching technique presented in Chapter 5.

in accuracy varies greatly between data sets, and while in the Synthetic data set performance is reduced by almost 30%, in the Rugby 2 data set performance is only reduced by 2.2%.

On an Intel Core2Duo 6300 1.8GHz processor the standard key-pose enhanced action matching algorithm takes 1020 seconds to process the Rugby 2 data set, while the summarised action matching algorithm takes 567 seconds.

This technique reduces the computational cost of the algorithm significantly while also generalising the key-pose detection strategy to non-cyclic and non-ambulatory motions. However, the lower temporal granularity has two detrimental effects on the system. Firstly, the output sequence consists mainly of interpolation between key-frames and hence is a “smoothed” version of the target action. While this can be more visually appealing, it does reduce the precision of the representation as matching is only enforced at key-frames. Secondly, the technique can show an increased sensitivity to noise. Particularly noisy frames will tend to be extracted as key-frames due to their dissimilarity to the surrounding frames, and the erroneous detection caused by matching against these noisy frames will affect the reconstruction of all frames represented by the key-frame.

Some examples are shown in Figures 6.11, 6.12 and 6.13. In Figures 6.11 and 6.12
Figure 6.11: Crop of players showing every 2nd frame over a rugby sequence. At each time frame, a crop from the original image set is shown on the left, the result of the simple action matching scheme is shown in the centre, and the result of the summarised action matching scheme is shown on the right.
Figure 6.12: Crop of players showing every 2nd frame over a rugby sequence. At each time frame, a crop from the original image set is shown on the left, the result of the simple action matching scheme is shown in the centre, and the result of the summarised action matching scheme is shown on the right.
Figure 6.13: Crop of players showing every 5th frame over a football sequence. At each time frame, a crop from the original image set is shown on the left, the result of the simple action matching scheme is shown in the centre, and the result of the summarised action matching scheme is shown on the right.
6.4 Hierarchical Action Matching

the more general representation of the dynamics of the activity allows the technique to correctly synchronise to the animation where the action matching technique fails. However, in Figure 6.13 the lack of detail in the motion representation allows the system to erroneously select an animation facing in the wrong direction. A small number of these catastrophic failures contribute greatly to the reduced accuracy of the technique when compared to the boosted action matching technique.

On the whole, this technique is best suited to scenarios where data is essentially clean, but where a more general approach (i.e. not limited to ambulatory motions) is required. In the outdoor sports broadcast environment considered here, the benefit is minimal compared to the standard action matching technique with key-pose detection. However, the performance improvement is significant, which is an important consideration if the exemplar action library is to be extended.

6.4 Hierarchical Action Matching

While animation summarisation drastically reduces the complexity of the action matching problem, the number of states to be considered still grows with each animation added to the action library. With a traditional hidden Markov model (HMM), the number of state transitions considered grows with the square of the total number of key-frames in the library - a computational and memory cost which can quickly become very large as the number of animations considered grows. This section introduces a variant on the HMM which addresses this shortcoming.

The HMM has been extended in many ways to allow it to deal with large numbers of states and to exploit the high level relationships between states. Layered HMMs[77], stochastic context-free grammars [80] and Hierarchical HMMs[27] have all been proposed as ways of structuring Markov models to exploit the inherent structure of the system being modelled.

All of these techniques work by breaking up the model into a layered representation of the system, rather than a single monolithic state machine. The layers in these machines can typically be considered as “child” layers, which encode the relationship between
the observed data and the simplest symbolic representation of the data, and “parent” layers, which encode the higher level structure of the model. In these models typically a child model is directly associated with one and only one parent state.

A problem for layered structures such as these is re-initialisation of the model for the child layers. The parent layer decides which child model to activate at any given time, based on the responses of the children to the data and its own internal state transition model. As such, it must be able to perform a meaningful comparison between the likelihoods of different child models. However, state probabilities are accumulated over time as each model in the child layer derives its own best explanation of the observations. If a model is never re-initialised then models in the child layer are being compared based on their ability to explain the entire observation sequence. For extended sequences of observations this is obviously not desirable behaviour.

This problem affects the application of HMMs to the recognition of continuous actions. While an individual child model can be chosen at any point (either by looking ahead at the sequence or by consuming pre-segmented sections of the observation sequence), dynamically changing between child models is affected by the re-initialisation problem. This problem can be summarised thus: a child model must explain the current state and a set of previous states, while a re-initialised child model only has to explain the current state. Comparing against re-initialised models will almost always favour the re-initialised model. So the problem is choosing a strategy that re-initialises the child models sufficiently frequently to stop the model getting stuck in one state, but not so frequently that the model becomes unstable.

This issue is typically solved by quantising or clustering events at the lowest time granularity and re-initialising each child model after the cluster of observations is processed, as in work by Oliver et al. [77]. Parent models then consume the relative likelihoods of each child as a new observation vector, working at a different temporal granularity as shown in Figure 6.14. This introduces temporal quantisation artefacts as each level of models is limited by the temporal granularity at which it operates. An alternative approach is that employed by Hierarchical HMMs introduced by Fine et al. [27], where control flows up and down the hierarchy, with each level yielding to a higher level as it
6.4. Hierarchical Action Matching

Figure 6.14: A Layered HMM. The parent layer (shown as a green block) "consumes" observations (shown as green circles) emitted by the child layer (shown as red blocks). The child layer consumes the raw data observations (shown as red circles) which are segmented into groups. The parent layer and child layers operate at different temporal granularities.

Figure 6.15: A Hierarchical HMM. The parent layer (shown as a green block) "consumes" observations (shown as green circles) emitted by the child layer (shown as red blocks). The child layer consumes the raw data observations (shown as red circles). Control flow is indicated by black arrows. Upon selecting a child model, the parent layer yields control to the child layer which then consumes observations until it reaches a production state, at which point it yields control to the parent layer which selects the next child model.
reaches a production state, as shown in Figure 6.15. While this type of model can avoid
temporal quantisation artefacts in some scenarios, it cannot model arbitrary transitions
between looping behaviours, as a child model cannot be preempted by another child
model - it must yield on reaching a production state. Similarly, if the parent model
chooses an incorrect child model it has no way to recover from this mistake until the
child model reaches a production state - and there is no guarantee that at that time
the model will be in a situation to unambiguously select the correct state.

This work uses a novel formulation similar to the Layered Markov Models used in the
work of Oliver et al. [77]. Unlike previous work, our formulation allows all levels of the
model to work at the lowest level of temporal granularity, and does not limit action
transitions to any specific states of the modelled behaviours.

6.4.1 Hierarchical Markov Model

To model the activity in the input sequence, a Markov model is constructed. Each state
in the model represents a key-frame from the library sequences and each key-frame in
the input sequence is an observation to be explained by the model. State transition
probabilities represent constraints on transitions between frames.

In order to handle arbitrary re-initialisation, a novel type of hierarchical Markov model
is introduced. Each parent state represents two child models, one with a re-initialised
set of initial state probabilities and one with a set of initial state probabilities taken
from the previous time step.

This problem is solved by representing the system through a single parent and multiple
child models. The parent model represents the progression of the sequence in terms
of the most appropriate action to represent the sequence at any one time. Each child
model matches the action it represents to the observed sequence on a frame-by-frame
basis.

Both child and parent models are represented in the standard manner as a state tran-
sition probability matrix $T$, an emission probability matrix $E$ and an initial state
probability vector $\pi$, and the maximum likelihood state sequence is calculated using
the Viterbi algorithm[81].
6.4. Hierarchical Action Matching

The system is represented by a single high level model and multiple children models. Each state in the parent model corresponds to a child model (only the children of two parent states are shown in this diagram). Each child model is evaluated twice - once with re-initialisation at every time step (blue) and once retaining state from time step to time step (red). Colour coding of the transitions in the parent model show that the output of the blue evaluation is used when the parent model transitions between states over time, and the red evaluation is used when the parent model remains in the same state.

At each time \( t \), the window of observations \((t, t+1)\) is evaluated using each child model. The child model for action \( n \) is evaluated in two ways - once with \( \pi \) representing a uniform distribution over all states to give the re-initialised likelihood \( P_n^\text{r}(t) \), and once using \( \pi \) calculated from the evaluation of the child model at \( t-1 \), giving the continuous likelihood \( P_n^\text{c}(t) \) (see Figure 6.16). In both cases, \( T(i, j) = 1 \) if \( \text{modulom}(j - i) = 1 \), otherwise \( T(i, j) = 0 \) (where \( m \) is the number of states in the model), and \( E(i, j) = -\sqrt{K(i,j)} \) (\( K \) being the KLD described in Chapter 5). \( E \) is then normalised such that for each row \( E(i) \), \( \min(E(i)) = 0 \) and \( \max(E(i)) = 1 \). In order to avoid weighting towards the re-initialised model, \( \pi \) is always normalised so that all likelihoods sum to 1.

The parent model is then evaluated to determine the child model that will be active at any given time. A transition probability matrix is used such that \( T(i, i) = 0.8 \) and \( T(i, j) = \frac{0.2}{m-1} \) (where \( m \) is the number of states in the model). This distribution is used to avoid over-fitting the model to the relatively small amount of data available.
With larger data sets these likelihoods could be learnt using Baum-Welch[81] or similar techniques.

\[ E(i, i) = P_f(t) \] and \[ E(i, j) = P_t(t). \] This means that if the parent model changes action, then the observation probability is calculated using the re-initialised child model, and if the model stays on the same action, then the child model is used without re-initialisation. In this way a path through the parent model can be calculated, allowing for the re-initialisation of any child model at any time and for the correct simultaneous evaluation of multiple possible paths through the model, without the need for an up-front segmentation of the observation sequence into actions.

At this point, the prior state \( Q^*_n(t-1) \) is known for the maximum-likelihood state sequence that ends in state \( n \) at time \( t \). If the prior state was a different state (i.e. \( Q^*_n(t-1) \neq n \)), then we preserve the final distribution of the re-initialised child model. Otherwise (i.e. \( Q^*_n(t-1) = n \)) we preserve the final state of the continuous model. This preserved state is then used to initialise the continuous model in the next time step.

With this model, it is also easy to maintain the child model state sequence \( Q_n \) for each child model \( n \) that accompanies the high level state sequence \( Q \). At each time \( t \), the state sequences \( Q'_n(t) \) and \( Q''_n(t) \) are obtained from the child model (where \( Q'_n(t) \) is a single state at time \( t \), while \( Q''_n(t) \) will contain multiple previous states). If the parent model calculates that the maximum likelihood path through \( n \) at \( t \) is a transition from another state, then \( Q'_n(t) \) is appended to \( Q_n \). If the parent model indicates the maximum likelihood path through the model at \( t \) is a self-transition from the same state, then \( Q''_n(t) \) overwrites the previous entries in \( Q_n \). In this way, the final output state sequence can be calculated simply as \( Q_n(t) \) with \( n = Q(t) \).

An alternative way of looking at these models is via a standard graphical model representation of the relationship between the various probability distributions on the observed and hidden variables in the system. These are shown in Figure 6.17 where it can be seen that the models vary principally in the conditional relationships involving \( p_t \) (i.e. what parts of the model affect when the parent model can change state) and \( q_{t-1} \) (i.e. when the child model re-initialises itself). Figure 6.17 a) shows a simple arrangement where the child state is conditional only on the parent state and the child
6.4. Hierarchical Action Matching

Figure 6.17: Various hierarchical HMMs showing the different relationships between random variables in the models. $p_t$ and $p_{t-1}$ are the discrete probability distributions over the state variable for the parent model at times $t$ and $t-1$, while $q_t$ and $q_{t-1}$ are the discrete probability distributions over the state variable for the child model at times $t$ and $t-1$. There are $n$ child models however for clarity only one is shown (the structure within the blue plate is repeated $n$ times). $x_t$ is the continuous distribution over the observation space at time $t$. a) A simple hierarchical HMM, b) a Hierarchical HMM [27], c) a Layered HMM [77], d) the proposed model.
Chapter 6. Action Matching: Extensions and Enhancements

Table 6.3: Evaluation of the generated pose estimates. The column headed “Baseline” gives the corresponding scores for the original action matching technique presented in Chapter 5.

<table>
<thead>
<tr>
<th>Sequence Name</th>
<th>Matches Exact</th>
<th>Matches Near</th>
<th>Matches Mirror</th>
<th>Baseline Hit</th>
<th>Baseline Miss</th>
</tr>
</thead>
<tbody>
<tr>
<td>Synthetic</td>
<td>40.0%</td>
<td>19.2%</td>
<td>3.8%</td>
<td>63.1%</td>
<td>36.9%</td>
</tr>
<tr>
<td>Synthetic Error</td>
<td>31.5%</td>
<td>20.0%</td>
<td>4.6%</td>
<td>56.2%</td>
<td>43.8%</td>
</tr>
<tr>
<td>Rugby 1</td>
<td>43.8%</td>
<td>12.6%</td>
<td>2.7%</td>
<td>59.1%</td>
<td>40.9%</td>
</tr>
<tr>
<td>Rugby 2</td>
<td>41.5%</td>
<td>12.6%</td>
<td>3.0%</td>
<td>57.0%</td>
<td>43.0%</td>
</tr>
<tr>
<td>Football 1</td>
<td>13.4%</td>
<td>12.4%</td>
<td>3.2%</td>
<td>29.0%</td>
<td>71.0%</td>
</tr>
<tr>
<td>Football 2</td>
<td>40.9%</td>
<td>9.9%</td>
<td>1.8%</td>
<td>52.6%</td>
<td>47.4%</td>
</tr>
<tr>
<td>Football 3</td>
<td>20.3%</td>
<td>11.0%</td>
<td>0.0%</td>
<td>31.3%</td>
<td>68.7%</td>
</tr>
<tr>
<td>Total</td>
<td>33.8%</td>
<td>13.3%</td>
<td>2.6%</td>
<td>49.7%</td>
<td>50.3%</td>
</tr>
</tbody>
</table>

6.4.2 Results

A quantitative analysis of the results is shown in Table 6.3. The technique evaluated uses key-pose boosting, as presented earlier in the chapter, within the hierarchical framework. Processing on an Intel T7300 2GHz Core2Duo processor takes 503 seconds to process the Rugby 1 data set, while the flat HMM takes 1020 seconds. The
6.4. Hierarchical Action Matching

Implementation of the hierarchical technique contains much more python code than the flat action matching technique, therefore it is expected that an implementation in C++ would display a further significant improvement in performance. Thus, this technique allows an improvement in the computational complexity of the action matching algorithm, without degrading the matching performance as much as the animation summarisation technique.

A second major benefit of this technique is that the independence of the models at the child layer allows their calculation in parallel, thus greatly increasing the scalability of the model. Finally, with sufficient data, the models may be improved by replacing the a priori transition probabilities with learnt distributions. This model separates the intra-animation transitions from the inter-animation transitions, allowing both to be learnt separately, and thus removing the requirement to re-train the entire model should a new activity be added to an existing model.
Figure 6.18: Crop of players showing every 2nd frame over a rugby sequence. At each time frame, a crop from the original image set is shown on the left, the result of the simple action matching scheme is shown in the centre, and the result of the hierarchical action matching scheme is shown on the right.
Figure 6.19: Crop of players showing every 2nd frame over a rugby sequence. At each time frame, a crop from the original image set is shown on the left, the result of the simple action matching scheme is shown in the centre, and the result of the hierarchical action matching scheme is shown on the right.
Figure 6.20: Crop of players showing every 10th frame over a football sequence. At each time frame, a crop from the original image set is shown on the left, the result of the simple action matching scheme is shown in the centre, and the result of the hierarchical action matching scheme is shown on the right.
6.5 Summary

Despite the overall degradation in matching compared to the non-hierarchical boosted action matching technique, the technique does improve the matching in several instances, as shown in Figures 6.18 and 6.19. However, there are more instances where the model fails when compared to the original technique, as shown in Figure 6.20. These failures are caused by the lack of constraints on inter-animation transitions. In the flat model, state transition probabilities between states in different animations are related to the similarities of the two states. In the hierarchical model, this constraint cannot be enforced without recalculating the higher level transition matrix at every time step. The removal of this constraint allows the hierarchical model to transition at less appropriate times. However, part of the reason that this limitation has such an effect is that the system is trying to fit the observed data from a small selection of exemplars. Expanding the library (as is possible with the hierarchical model) would potentially reduce the errors generated in this way.

6.5 Summary

This chapter has introduced several enhancements to the action matching system proposed in Chapter 5. Key-pose detection improves the synchronisation of the synthesised animation to the source animation. Animation summarisation reduces the computational complexity of the problem and generalises it to non-ambulatory activities, but introduces some degradation in the action matching, and a hierarchical Markov model improves the scalability of the technique. These enhancements address some of the weaknesses of the initial system, improving the robustness, generality and scalability of the action matching scheme.
Chapter 7

Conclusions and Further Work

The introduction of 3D reconstruction techniques to the outdoor broadcast environment is a challenging subject, but one with many applications to the broadcast of sporting events. The work presented in this thesis has explored several avenues for the automatic generation of 3D content from multi-camera footage of sporting events.

7.1 Conclusions

In Chapter 3, a quantitative technique for the analysis of reconstruction errors was presented. Due to the difficulty of directly estimating errors in the multi-camera system, the resultant errors in the system were modelled instead - a technique which was developed to produce a number of error-tolerant modifications to the visual hull algorithm. These reconstruction techniques were compared against a standard and a probabilistic visual hull technique in the field of free-viewpoint video rendering. None of the techniques provide a sufficiently detailed reconstruction of the scene for use in the production of high quality video, however they provide reconstructions of the scene with varying degrees of accuracy and completeness.

Due to the camera geometry involved in the outdoor sports broadcast environment (multiple cameras arranged along a wide baseline, far away from the recorded events), it would not be possible to generate a high quality reconstruction of the scene using
simple visual hull techniques alone. In the presence of errors, not only is the ability to generate a scene reconstruction compromised, but also the ability to use any resulting geometry to accurately determine pixel correspondences between the original images. This severely limits the abilities of a free-viewpoint video system built upon such data. In order to generate high quality reconstructions of this data, either the global nature of the reconstruction or the realism of the reconstruction must be sacrificed.

In Chapter 4, a local shape optimisation technique was introduced which converted the global reconstruction generated using shape-from-silhouette into a scene reconstruction optimised for views in the vicinity of one of the original cameras. The dual-mode deformable model introduced in Chapter 4 first performs a search for the best local data cues and then smoothly deforms an initial reconstruction to fit these cues. This technique was shown to improve the quality of reconstruction when low quality, sparse, single-definition cameras were used. However, this technique did not greatly improve the quality of reconstruction when higher quality images were used.

Global and local inter-camera correspondence is limited by calibration accuracy, but the high level structure of the scene (humans engaging in sporting activities) and the scene dynamics are less affected by these errors. Chapter 5 introduces the action matching scheme as an attempt to leverage the high level and dynamic scene structure to generate synthetic scene representations.

A technique for segmenting a single scene reconstruction was presented as well as an analysis of various shape matching techniques. The symmetric Kullback Leibler divergence was proposed as the most appropriate distance measure to compare shape histograms of player reconstructions against a library of 3D synthetic animations. A Markov-model-based scheme was introduced in order to model the dynamics of the recorded activities, and the scheme was evaluated against multiple recordings of football and rugby.

While the action matching scheme performs well in a challenging environment, the noisy and inaccurate nature of the reconstructions means that failures often occur. Chapter 6 introduces a key-pose boosting enhancement to the action matching scheme which is shown to significantly improve the matching ability of the technique. Anima-
tion summarisation is also introduced as a way to improve action matching without restricting the technique to ambulatory activities. Finally, a hierarchical Markov model is proposed as a way to overcome the limitations inherent in the flat model proposed in the basic action matching scheme. However, this is shown to have a negative impact on matching accuracy.

The system proposed produces fully automatic, accurate pose recovery for a substantial number of the total subjects present in the data sets used for evaluation. The technique greatly automates what has, up until now, been a manual process in the preparation of synthetic representations of sporting events. However, there are still many failure cases and there remains substantial room for improvement in the technique.

7.2 Further Work

While the work presented in this thesis applies several reconstruction techniques to sports footage, much work remains to be done in order to successfully bring free viewpoint video to the outdoor broadcast environment. Techniques which combine a global multi-camera reconstruction with local image refinement have been shown to be capable of high quality scene reconstruction [36] and further work in this direction is needed to evaluate additional avenues of enquiry. It is possible that the simple formulation of the dual-mode deformable model was appropriate for use with rough initialisation and low resolution images, but for better initialisation with higher quality images, a more sophisticated regularisation term is required. Similarly, more advanced handling of the calibration errors, possibly incorporating some calibration tolerance in the stereo matching energy, could improve the quality of the results.

The action matching technique as presented would also bear significant further enquiry. A larger library of domain-specific activities, including sharp turns, sideways motions and kicks should be assembled and applied to footage to determine the effect on matching. Investigation should examine whether multiple similar examples improve the matching quality and how the model behaves as a larger variety of body poses are introduced - does increasing the flexibility of the model in this way lead to an improved
ability to match to more representative activities, or does it simply under-constrain the model to over-fit to the noise in the inputs?

Another area for investigation should be the application of learnt dynamics rather than the simple a priori dynamical model presented in this thesis. This would encapsulate both training the response of the model to individual activities and the distribution of activity transitions in the data (for example, the "jog to run" transition is more likely than the "jog to jump" transition). Again, the challenge here would be to avoid over-fitting to a small training set.

The effect of extending the action matching scheme to consider other factors such as an appearance model should also be considered. Velocity and angular rotation as well as the facing direction (possibly recovered using face detection) could be incorporated into the technique, but methods to balance the relative importance of these cues must also be incorporated.

While there are multiple routes to improving the action matching technique, in the long term more fruitful avenues could be explored through the combination of action matching with other techniques. In pursuit of more robust and accurate pose recovery, action matching could be used to initialise particle-filter-based techniques to deliver fully automatic, fully generic pose detection in the outdoor sports broadcast environment.

Action matching provides a rough approximation of the human pose that generated the recorded images, however the fine detail of the pose and surface effects such as cloth motion and muscle deformation are lost. By combining deformation models such as the one suggested in Chapter 4 with the temporally consistent priors generated by the action matching scheme, it may be possible to improve shape reconstruction, as a temporally consistent framework is generated within which reconstruction cues from multiple frames of multiple cameras can be combined. Also, the techniques presented in this thesis have to over-estimate the reconstructed shape in order to avoid truncations due to errors, thus these reconstructions can be viewed as an upper bound on the shape to be reconstructed. As it is known that a human generated the shapes for which reconstruction is being attempted a lower bound on the reconstruction can be
generated using the shape derived from the action matching technique (for example, the reconstruction must have exactly two arms, two legs and one head). Thus, the techniques may be combined to set both an upper and lower bound for an optimisation, such as a deformable model, to recover the correct shape. In this way yet more of the available scene knowledge may be leveraged to take another step towards fully automatic high-quality scene reconstruction of outdoor broadcasting events.
Appendix A

Sports Capture Data Sets

This thesis makes use of seven data sets. This appendix contains a brief description and visualisation of each data set. For all data apart from the synthetic data sets, camera calibration was obtained from natural image features using the technique described by Thomas et al. [99], and images were segmented into foreground and background using a combination of chroma-keying (with the green as the background colour) and background subtraction.

Table A.1 summarises the data sets used in this thesis:

<table>
<thead>
<tr>
<th>Name</th>
<th>Format</th>
<th>Calibration Errors</th>
<th>Matting Errors</th>
<th>Compression Errors</th>
<th>Sequence Length</th>
<th>Frame Rate</th>
<th>No. of Cameras</th>
</tr>
</thead>
<tbody>
<tr>
<td>Synthetic</td>
<td>HD</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>100</td>
<td>50fps</td>
<td>10</td>
</tr>
<tr>
<td>Synthetic Error</td>
<td>HD</td>
<td>✓</td>
<td>X</td>
<td>X</td>
<td>100</td>
<td>50fps</td>
<td>10</td>
</tr>
<tr>
<td>Football 1</td>
<td>SD</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>307</td>
<td>50fps</td>
<td>14</td>
</tr>
<tr>
<td>Football 2</td>
<td>HD</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>185</td>
<td>50fps</td>
<td>6</td>
</tr>
<tr>
<td>Football 3</td>
<td>HD</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>120</td>
<td>50fps</td>
<td>6</td>
</tr>
<tr>
<td>Rugby 1</td>
<td>HD</td>
<td>✓</td>
<td>✓</td>
<td>X</td>
<td>286</td>
<td>50fps</td>
<td>10</td>
</tr>
<tr>
<td>Rugby 2</td>
<td>HD</td>
<td>✓</td>
<td>✓</td>
<td>X</td>
<td>49</td>
<td>50fps</td>
<td>10</td>
</tr>
</tbody>
</table>

Table A.1: Summary of the data sets used in this thesis.
Appendix A. Sports Capture Data Sets

A.1 Synthetic

This data set consists of renders of textured human models generated using the camera locations from the Rugby 1 data set. The mattes for this data set are also generated synthetically, so this data set contains no matting errors. It is 100 frames long and uses 10 cameras.

A.2 Synthetic Error

This data set consists of the images and mattes from the Synthetic data set, however the camera calibration parameters have been perturbed slightly to simulate the effect of camera calibration error.

A.3 Football 1

This data set consists of SD images of a football match. The cameras were arranged around one quarter of the pitch, focusing on one half of the pitch only. The data set is 307 frames long and contains 14 cameras.

A.4 Football 2

This data set consists of HD images of a football match. The cameras used were the standard broadcast cameras plus one extra static camera. The images were captured from the broadcast replay system, so some compression and sharpening artefacts are present in the source images. The data set is 185 frames long and uses 6 cameras.

A.5 Football 3

This data was captured during the same football match as the Football 2 data set, but at a different time. The data set is 120 frames long and uses 6 cameras.
A.6 Rugby 1

This data set consists of HD images of a rugby match. The cameras used were five of the standard broadcast cameras and five extra static cameras. The images were captured directly from the cameras and are uncompressed, although some sharpening artefacts are present in the source images. The data set is 286 frames long and uses 10 cameras.

A.7 Rugby 2

This data was captured during the same rugby match as the Rugby 1 data set, but at a different time. The data set is 49 frames long and uses 10 cameras.

A.8 Sample Images

Figures A.1 to A.7 show example images from all cameras and a schematic of the position of the cameras. In the camera layout diagram for each data set, static cameras are rendered blue, moving cameras are rendered green, and the camera removed for “leave one out” tests is rendered red.
Figure A.1: *The Synthetic data set.*
Figure A.2: The Synthetic Error data set.
Figure A.3: The Football 1 data set.
Figure A.4: The Football 2 data set.
Figure A.5: The Football 3 data set.
A.8. Sample Images

Figure A.6: *The Rugby 1 data set.*
Figure A.7: The Rugby 2 data set.
Appendix B

Motion Capture Library

For this work, a standard library of motion captured human activities was used to generate synthetic data sets, and as exemplars for use in shape matching schemes. The libraries used are detailed in this appendix.

B.1 Sources

The motion capture data used comes from two sources; the commercially available Mega MoCap library[45] (MM) and additional data from an in-house motion capture session (SM).

B.2 Action Library

An articulated human model, along with a library of motion capture, was used as the basis for motion synthesis. A simple generic human model was skinned to a standard human skeleton, and this was animated using motion capture from the MM and SM motion capture databases. The animations used are presented in Table B.1. Some animations were truncated from the supplied sequence to extract a single loop. These animations were chosen as a simple representative sample of the motions to be recognised in the recorded data.

The following figures show every third frame of each animation:
### Appendix B. Motion Capture Library

Table B.1: The animations in the motion library used in the action matching scheme.

<table>
<thead>
<tr>
<th>Animation Name</th>
<th>Source</th>
<th>File Name</th>
<th>Start Frame</th>
<th>End Frame</th>
<th>Loop?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jog</td>
<td>MM</td>
<td>Runs/Jog.bvh</td>
<td>0</td>
<td>22</td>
<td>✓</td>
</tr>
<tr>
<td>Jump</td>
<td>MM</td>
<td>SkipRope.bvh</td>
<td>67</td>
<td>103</td>
<td>X</td>
</tr>
<tr>
<td>Jump2</td>
<td>SM</td>
<td>MISC/JUMPING1.BVH</td>
<td>38</td>
<td>63</td>
<td>✓</td>
</tr>
<tr>
<td>Run</td>
<td>MM</td>
<td>Runs/RunFast.bvh</td>
<td>0</td>
<td>19</td>
<td>✓</td>
</tr>
<tr>
<td>RunMed</td>
<td>MM</td>
<td>Runs/RunMed.bvh</td>
<td>0</td>
<td>13</td>
<td>✓</td>
</tr>
<tr>
<td>RunSlow</td>
<td>MM</td>
<td>Runs/RunPaceSlow.bvh</td>
<td>0</td>
<td>22</td>
<td>✓</td>
</tr>
<tr>
<td>RunSlow2</td>
<td>MM</td>
<td>Runs/RunPaceSlow2.bvh</td>
<td>0</td>
<td>22</td>
<td>✓</td>
</tr>
<tr>
<td>RunTurnLeft</td>
<td>MM</td>
<td>Runs/RunHardTurnLeft.bvh</td>
<td>3</td>
<td>27</td>
<td>X</td>
</tr>
<tr>
<td>RunTurnRight</td>
<td>MM</td>
<td>Runs/RunHardTurnRight.bvh</td>
<td>0</td>
<td>14</td>
<td>X</td>
</tr>
<tr>
<td>Skip</td>
<td>MM</td>
<td>Walks/Skip.bvh</td>
<td>0</td>
<td>32</td>
<td>✓</td>
</tr>
<tr>
<td>Sprint</td>
<td>MM</td>
<td>Runs/RunFastHardShort.bvh</td>
<td>0</td>
<td>15</td>
<td>✓</td>
</tr>
<tr>
<td>Sprint2</td>
<td>SM</td>
<td>RUNS/SPRINT.BVH</td>
<td>0</td>
<td>18</td>
<td>✓</td>
</tr>
<tr>
<td>Sprint3</td>
<td>MM</td>
<td>Runs/RunFastHard2.bvh</td>
<td>0</td>
<td>16</td>
<td>✓</td>
</tr>
<tr>
<td>Standing</td>
<td>SM</td>
<td>calibration/CALIB3.BVH</td>
<td>0</td>
<td>49</td>
<td>✓</td>
</tr>
<tr>
<td>Walk</td>
<td>MM</td>
<td>Walks/Walk.bvh</td>
<td>0</td>
<td>29</td>
<td>✓</td>
</tr>
<tr>
<td>WalkBackwards</td>
<td>MM</td>
<td>Walks/WalkBackwards.bvh</td>
<td>24</td>
<td>64</td>
<td>✓</td>
</tr>
</tbody>
</table>

Figure B.1: Jog.
B.2. Action Library

Figure B.2: Jump.

Figure B.3: Jump2.

Figure B.4: Run.

Figure B.5: RunMed.
Figure B.6: RunSlow.

Figure B.7: RunSlow2.

Figure B.8: RunTurnLeft.

Figure B.9: RunTurnRight.
Figure B.10: *Skip.*

Figure B.11: *Sprint.*

Figure B.12: *Sprint2.*

Figure B.13: *Sprint3.*
Figure B.14: *Standing.*

Figure B.15: *Walk.*

Figure B.16: *WalkBackwards.*
Appendix C

View-dependent Texturing Algorithm and Application

C.1 The Workbench

Figure C.1: A screenshot of the workbench showing debug overlay with camera locations and simulated stadium.

A real-time view-dependent rendering application was developed, allowing the visualisation of various reconstruction techniques with a state-of-the-art view-dependent rendering engine. This technique was used for rendering all reconstruction techniques presented in this thesis. Example screenshots are shown in Figures C.1 and C.2.
Figure C.2: A close-up showing players rendered against a blue background.

Figure C.3: Diagram illustrating the relationship of surface element $p$, virtual viewpoint $v$ and source camera eye-point $e_i$, used to calculate texture blend weights for view-dependent texturing.
The engine is OpenGL-based[34] and uses a GLSL fragment shader to perform radial-distortion-corrected view-dependent rendering, with full occlusion calculation. The technique uses a shadow mapping pass to calculate the visibility of the scene geometry in each of the source cameras. The technique then chooses the nearest two cameras using the following formula, based on the angle between a ray from the surface patch to the camera and a ray from the surface patch to the desired viewpoint (see Figure C.3):

\[
d_i = e_i - p
\]

\[
a = v - p
\]

\[
s_i(p) = (\hat{d}_i \cdot \hat{n})(\hat{d}_i \cdot \hat{a})
\]

where \(p\) is the surface point being rendered, \(e_i\) is the location of the eye-point for camera \(i\), \(v\) is the location of the virtual camera, and \(n\) is the surface normal at \(p\). The source images are then sampled by projecting \(p\) into the two cameras which maximise \(s_i(p)\). These two samples are then blended based on the ratio of the scores of their respective cameras. The choice of cameras to blend is made on a per-output-pixel basis and so a single triangle may be textured with multiple source images over its entire extent. The code also disregards cameras where the projection of \(p\) lies close to the image edge, to avoid sharp transitions at visibility boundaries.

It is worth noting that while real-time rendering at up to 50fps is easily achieved with this renderer, the strain of loading multiple HD images far exceeds the computational strain of rendering a single frame. As a result, real-time playback of a sequence is not possible and rates of approximately 0.5 fps are achieved. With improved caching, for example loading image data as a video sequence to be decompressed at run-time, higher frame rates may be achieved.
Bibliography


[77] N. Oliver, E. Horvitz, and A. Garg. Layered representations for human activity


the 31st Annual Conference on Computer Graphics and Interactive Techniques,