A Structure from Motion solution to Head Pose Recovery for Model-Based Video Coding

by

Jonathan Michael Heathcote

Submitted in fulfillment of the academic requirements
for the degree of Master of Science in Engineering
in the School of Electrical, Electronic and Computer Engineering
at the University of Kwa-Zulu Natal, Durban, South Africa

October 2005

To my parents, Mike and Heather.

This document was created in LaTeX

Abstract

Current hybrid coders such as H.261/263/264 or MPEG-1/-2 cannot always offer high quality-to-compression ratios for video transfer over the (low-bandwidth) wireless channels typical of handheld devices (such as smartphones and PDAs). Often these devices are utilised in videophone and teleconferencing scenarios, where the subjects of interest in the scene are peoples faces. In these cases, an alternative coding scheme known as Model-Based Video Coding (MBVC) can be employed. MBVC systems for face scenes utilise geometrically and photorealistically accurate computer graphic models to represent head and shoulder views of people in a scene. High compression ratios are achieved at the encoder by extracting and transmitting only the parameters which represent the explicit shape and motion changes occurring on the face in the scene. With some *a priori* knowledge (such as the MPEG-4 standard for facial animation parameters), the transmitted parameters can be used at the decoder to accurately animate the graphical model and a synthesised version of the scene (originally appearing at the encoder) can be output.

Primary components for facial re-animation at the decoder are a set of local and global motion parameters extracted from the video sequence appearing at the encoder. Local motion describes the changes in facial expression occurring on the face. Global motion describes the three-dimensional motion of the entire head as a rigid object. Extraction of this three-dimensional global motion is often called head tracking.

This thesis focuses on the tracking of rigid head pose in a monocular video sequence. The system framework utilises the recursive Structure from Motion (SfM) method of Azarbayejani and Pentland. Integral to the SfM solution are a large number of manu-

ally selected two-dimensional feature points, which are tracked throughout the sequence using an efficient image registration technique. The trajectories of the feature points are simultaneously processed by an extended Kalman filter (EKF) to stably recover camera geometry and the rigid three-dimensional structure and pose of the head. To improve estimation accuracy and stability, adaptive estimation is harnessed within the Kalman filter by dynamically varying the noise associated with each of the feature measurements.

A closed loop approach is used to constrain feature tracking in each frame. The Kalman filter's estimate of motion and structure of the face are used to predict the trajectory of the features, thereby constraining the search space for the next frame in the video sequence. Further robustness in feature tracking is achieved through the integration of a linear appearance basis to accommodate variations in illumination or changes in aspect on the face.

Synthetic experiments are performed for both the SfM and the feature tracking algorithm. The accuracy of the SfM solution is evaluated against synthetic ground truth. Further experimentation demonstrates the stability of the framework to significant noise corruption on arriving measurement data. The accuracy of obtained pixel measurements in the feature tracking algorithm is also evaluated against known ground truth. Additional experiments confirm feature tracking stability despite significant changes in target appearance.

Experiments with real video sequences illustrate robustness of the complete head tracker to partial occlusions on the face. The SfM solution (including two-dimensional tracking) runs near real time at 12 Hz. The limits of Pitch, Yaw and Roll (rotational) recovery are 45°, 45° and 90° respectively. Large translational recovery (especially depth) is also demonstrated. The estimated motion trajectories are validated against (publically available) ground truth motion captured using a commercial magnetic orientation tracking system. Rigid re-animation of an overlayed wireframe face model is further used as a visually subjective analysis technique. These combined results serve to confirm the suitability of the proposed head tracker as the global (rigid) motion estimator in an MBVC system.

Preface

The research work presented in this thesis was performed by Jonathan Michael Heathcote, under the supervision of Mr Bashan Naidoo and Mr Stephen McDonald, at the University of KwaZulu-Natal's School of Electrical, Electronic and Computer Engineering. This work has been generously sponsored by Thales Advanced Engineering and Armscor. The financial assistance of the Department of Labour (DoL) towards this research is also acknowledged.

Aspects of this thesis were presented by the author at PRASA 2003, SATNAC 2003/2004 and MICSSA 2003/2005.

The entire dissertation, unless otherwise indicated as a reference, is the students own original work and has not been submitted, in whole or in part, to any other university for degree purposes.

Signed:

Name:

Date:

Acknowledgements

I wish to thank my supervisor, Mr Bashan Naidoo, for his friendly manner and trust in my ability to pursue a research avenue that would bring me educational fulfillment. Similar thanks must go to Mr Stephen McDonald, who has acted as co-supervisor during this work and invested time in proof reading this thesis. Many thanks must go to my sponsors, Thales and Armscor, for financing this research and the numerous (highly enjoyable) conferences I have been fortunate enough to attend. Specific thanks go to Mr Peter Handley and Mrs Franzette Vorster, for their involvement in this research from the beginning.

To my good friend Dan Carsky, with whom I have shared an office during this whole process, my thanks for the many insightful conversations and friendship during this process. To the rest of my postgraduate colleagues and members of staff I have been fortunate enough to get to know, thank you for the friendship, support and assistance you have given me. Stefan Scriba must be noted for encouraging me to compile this document using LaTeX, saving me from much aggravation had I pursued the conventional route.

I have received useful advice and assistance from researchers outside of South Africa. Many thanks must go to Dr. Simon Baker, Dr. Jacob Ström, Dr. Jorgen Ahlberg and Dr. Tony Jebara for their input, as I know they are very busy people. I would also like to thank Dr. Marco La Cascia and Dr. Stan Sclaroff, for compiling an online database of head-and-shoulder video sequences with corresponding motion ground truth. Without this, validating results in this thesis would have been difficult. In a similar vein, I appreciate all the time my good friend Colin Crompton spent sitting for me as a head model.

Finally, I would like to thank my parents and close friends. Without their support and

encouragement over the past months, I may have lost the focus I needed to complete this undertaking. Overall, it has been a highly enjoyable time for me that I will always remember fondly.

Contents

Abstract ii

Preface iv

Acknowledgements v

Contents vii

List of Figures xiii

List of Tables xviii

List of Acronyms xix

1 Introduction 1

1.1 Thesis Overview 1

1.2 Very Low Bit Rate Video Coding 3

1.3 Model-Based Video Coding 5

1.4 Advancement of Model-Based Coding Schemes 6

1.5 A MBVC System for Faces 8

1.5.1 MBVC System Components 9

1.6 Problem Definition 10

1.7 A Note on Accuracy 11

1.8 Contributions and Publications 12

2 Related Work **14**

2.1 Introduction 14

2.2 Optical Flow Based Methods 15

2.3 Active Appearance Techniques 17

2.4 Template Techniques 18

2.5 Kalman Filtering Methods 19

2.6 Proposed System Choices 21

3 Structure from Motion **24**

3.1 Introduction 24

3.2 A Recursive SfM Solution 26

3.3 System Dynamics 27

3.4 Geometry and Representations 28

3.4.1 Camera Model 29

3.4.2 Structure Model 31

3.4.3 Translation and Rotation 32

3.4.4 Coordinate Frame Transformation 38

3.4.5 Dealing with Scale 38

3.5 Kalman Filtering 39

3.5.1 Linear Kalman Filtering 40

3.5.2 Extended Kalman Filtering 41

3.5.3 Optimality of the EKF 43

3.6 EKF for SfM 44

3.6.1 State Vector Parameterisation 44

3.6.2 SfM Dynamics Solution 44

3.6.3 Calculating the Jacobian 47

3.7 EKF Synthetic Experiments 49

3.7.1 Generating the Data Set 50

3.7.2 Motion Estimates - Low Noise 52

3.7.3 Motion Estimates - High Noise 54

3.7.4 Structure and Camera Parameters 55

3.8 Conclusions on the SfM Technique 59

4 Visual Tracking **61**

4.1 Introduction 61

4.2 Tracking Techniques 62

4.2.1 Tracking via Image Flow 62

4.2.2 Tracking via Feature Correspondence 63

4.3 Region-based Tracking 64

4.3.1 Lucas-Kanade Tracking 66

4.3.2 Deriving the Lucas-Kanade Algorithm 67

4.4 An Efficient Tracking Algorithm 70

4.4.1 Inverse Compositional Image Alignment 71

4.5 Modeling Appearance Variation 75

4.5.1 Linear Appearance Variation 76

4.6 Motion Model Selection 78

4.7 Alignment and Appearance Change Experiments 82

4.7.1 Pixel Alignment Error 83

4.7.2 Parameter Convergence 84

4.7.3 Adaptive Appearance Modeling 86

4.8 2D Tracking Conclusions 91

5 Tracking System Implementation **92**

5.1 Schematic Overview 93

5.2 2D Tracker Initialisation 94

5.2.1 Including an Appearance Basis 97

5.3 Kalman Filter Initialisation 98

5.3.1 Initial Structure 99

5.3.2 Model Coordinates 100

5.3.3 Calculating Intersections 101

5.3.4 Camera and Motion Parameters 104

5.3.5 Noise Statistics 104

5.4 Measurement Noise Covariance 105

5.4.1 Major Error Case 106

5.4.2 Minor Error Case 107

5.5 Controlling Tracking Using Structure 108

5.5.1 Affine Parameter Estimates 109

5.6 System Design Conclusions 112

6 System Evaluation **113**

6.1 Robustness to Occlusion 115

6.2 Structure and Camera Convergence 118

6.3 Pose Estimation 120

6.3.1 Example Sequences 121

6.3.2 Example Sequences Discussion 127

6.3.3 Ground Truth Validation 129

6.3.4 Scope of Recovery 132

6.4 Tracking Failure 132

6.5 Tracking Performance 136

6.6 Evaluation Conclusion 137

7 Conclusions and Future Work **139**

7.1 Conclusions 139

7.2 Future Work 142

A Kalman Filtering **144**

A.1 Linear Kalman Filtering 144

A.2 Calculating Kalman Gain 145

B Synthetic Experiments **147**

B.1 Motion Estimates: Low Noise 147

B.2 Motion Estimates: High Noise 151

B.3 Structure and Camera Parameter Estimates: High Noise 155

B.4 Structure Estimates: Structural Evolution 156

C Linear Appearance Modeling **157**

C.1 IC Algorithm with Linear Appearance Variation 157

D Head Tracking Results **161**

D.1 Example Sequence - Glasses 162

D.2 Ground Truth Sequence 165

List of Figures

1.1 Schematic thesis overview. Numbered stages correspond to associated chapter numbers. 2

1.2 Basic concept of a Model-Based Video Coding system. 5

1.3 A MBVC system for face communication. 8

1.4 Rigid 3D animation parameters. 11

3.1 A standard Structure from Motion setup showing relative camera to object motion. 24

3.2 Camera Projection Models. 30

3.3 Extended Kalman filter equations for state prediction, and state correction via weighted measurement. 43

3.4 Measurement model prediction process. 46

3.5 Basis of SfM Experiments. Structure and motion recovery from pixel measurements. 50

3.6 Initial condition and groundtruth structure points. 56

3.7 Convergence of the 21 structural points. 57

3.8 Convergence of the camera parameter β. 58

3.9 Evolution of structure points to a hemispherical structure. 59

4.1 Template selection and template alignment. 65

4.2 The 9 online steps of the Lucas-Kanade image alignment algorithm. 69

4.3 Offline and online steps of the Inverse Compositional algorithm. 73

4.4 Schematic overview of the Inverse Compositional algorithm. Steps (3–6) are performed once as an *offline* pre-computation. The *online* algorithm consists of: image warping (Step 1), image differencing (Step 2), image dot products (Step 7), multiplication with the inverse of the Hessian (Step 8), and the update to the warp (Step 9). 74

4.5 Fundamental set of 2D planar transformations 78

4.6 Template, Figure 4.6(a), and Target, 4.6(b), frames used in the the evaluation of the IC image alignment algorithm. 83

4.7 Corner pixel alignment errors. 84

4.8 Incremental motion parameter updates. 85

4.9 Typical set of training images for an appearance basis. 87

4.10 Transition from template frame at t_0 to target frame at t_N under significant appearance change. 88

4.11 Template, target and appearance corrected regions. 88

4.12 Experiments comparing the IC algorithm. Template selection in 4.12(a). False/inaccurate alignment in 4.12(b) with no included appearance basis. Accurate alignment in 4.12(c) when an appearance basis is included. 90

5.1 Schematic overview of the tracking system. 94

5.2 Template windows selected around the eyes, corners of the nose and the mouth. 95

5.3 The four corners of each template are used as 2D tracking measurements. . 95

5.4 Tracking images demonstrating which template windows are most reliable under different out-of-plane rotations. 97

5.5 Candide face model [1] manually aligned to the face for initial structure estimates. 99

5.6 Ray tracing method for the estimation of depth. 101

5.7 Location of surrounding model vertices for each template corner. 102

5.8 Initial structure estimates on the Candide face model. 103

5.9 Badly matched two-dimensional features can be constrained using knowledge of the three-dimensional structure of the object in the scene. Bad (darker coloured) features are corrected to the two-dimensional projections of the structure. 109

6.1 Camera and magnetic tracker coordinate systems. 115

6.2 Occlusion sequences. In $(a), (c)$ *and* (e), the measurement covariance R_k is kept constant and the system diverges when occlusions occur. In $(b), (d)$ *and* (f), the covariance R_k is dynamically varying, so the system maintains stability. 117

6.3 Convergence of the structure parameters α. 119

6.4 Convergence of the camera parameter β. 120

6.5 Sequence demonstrating tracking over a wide range of rotations. 122

6.6 Pose estimates for the rotation sequence. 123

6.7 Quaternion estimates for the rotation sequence. 124

6.8 Sequence demonstrating tracking over a wide range of translations. 125

6.9 Motion estimates for the translation sequence. 126

6.10 Quaternion estimates for the translation sequence. 127

6.11 Frames from the Rotation sequence where the pose is frontal. 128

6.12 Ground truth tracking sequence. 130

6.13 Ground truth sequence motion estimates. 131

6.14 Tracking failure due to full occlusion. 133

6.15 Tracking failure caused by very high inter-frame motion. 134

6.16 Tracking failure due to large out-of-plane rotation. 135

B.1 Quaternion estimates for q_0 with corresponding estimation error. 147

B.2 Quaternion estimates for q_1, q_2 and q_3 with corresponding estimation errors. 148

B.3 Euler angles (converted from the estimated rotation quaternion) with corresponding estimation errors. 149

B.4 Estimated translation parameters with corresponding estimation errors. . . 150

B.5 Quaternion estimates for q_0 and q_1 with corresponding estimation errors using high noise measurements. 151

B.6 Quaternion estimates for q_2 and q_3 with corresponding estimation errors using high noise measurements. 152

B.7 Euler angles (converted from the estimated rotation quaternion) with corresponding estimation errors using high noise measurements. 153

B.8 Estimated translation parameters with corresponding estimation errors using high noise measurements. 154

B.9 Convergence of the 21 structural points using highly corrupted measurement data. 155

B.10 Convergence of the camera parameter β using highly corrupted measurement data. 155

B.11 Evolution of structure points to a hemispherical structure. 156

D.1 Quaternion estimates for the sequence with the user wearing glasses. 162

D.2 Sequence demonstrating tracking with an irregular face structure where the user is wearing glasses. 163

D.3 Motion estimates for the sequence with the user wearing glasses. 164

D.4 Ground truth tracking sequence. 165

D.5 Ground truth sequence motion estimates. 166

List of Tables

3.1 Rotation and translation estimation errors - Low Noise. 54

3.2 Rotation and translation estimation errors - High Noise. 55

5.1 Table of normalised correlation ranges with associated standard deviation. . 107

List of Acronyms

2D	Two-dimensional
3D	Three-dimensional
AAM	Active Appearance Model
AFT	Adaptive Feedback Tracker
AU	Action Unit
CCD	Charge Coupled Device
COP	Center Of Projection
DFFS	Distance from Face Space
EKF	Extended Kalman Filter
FACS	Facial Action Coding System
FAPs	Facial Animation Parameters
fps	Frames Per Second
HCI	Human Computer Interface
IA	Inverse Additive
IC	Inverse Compositional
IEEE	Institute of Electrical and Electronic Engineers
IRLS	Iteratively Re-weighted Least Squares
KF	Kalman Filter
L2	Least Squares
MBVC	Model-Based Video Coding
MPEG	Moving Picture Experts Group
NMSE	Normalised Mean Square Error

PCA	Principle Component Analysis
PDA	Personal Digital Assistant
RMS	Root Mean Square
ROC	Region of Convergence
SfM	Structure from Motion
SGI	Silicon Graphics Institute
SSD	Sum of Squared Difference

Chapter 1

Introduction

1.1 Thesis Overview

Achieving high quality video transmission under the high bandwidth constraints of hand-held devices (such as videophones) is becoming an ever more desirable capability. A popular approach to achieving such low bit rate video transfer is Model-Based Video Coding (MBVC), where a computer model is used to represent objects in a scene and only iterative model updates are transmitted for each frame of the video sequence. The work presented in this thesis deals with the tracking and estimation of the motion parameters required to rigidly animate a model of a human head in a MBVC system for faces.

Figure 1.1 provides a schematic overview of important processing modules and information flow within the proposed head tracking system. Attached to each significant stage are tags indicating the chapter number where that stage of the process is discussed in this thesis.

Section 1.2 of this chapter expands the discussion on very low bit-rate video, with a brief conceptual overview of model-based video coding systems presented in Sections 1.3 to 1.5. The contributions of this thesis are contextualised in Section 1.6, along with an explanation of necessary outputs that meet the demands of the head tracking component of a model-based coding system.

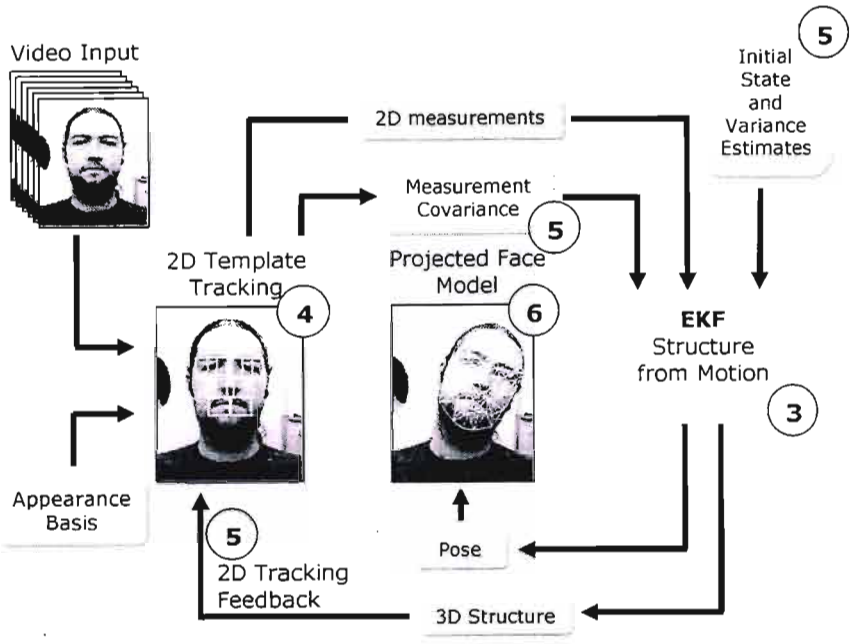


Figure 1.1: Schematic thesis overview. Numbered stages correspond to associated chapter numbers.

Head tracking is a current research topic with many proposed applications. These range from Human Computer Interfaces [2] to Model-Based Coding schemes [3] and Avatar animation [4]. In Chapter 2 a number of the techniques proposed in current literature to solving the head pose estimation problem are reviewed with an expansion on the proposals of this work in relation to these methods. Chapter 3 introduces the topics of Structure from Motion (SfM) and Kalman Filtering (KF), which constitute the foundational framework of the proposed three-dimensional (3D) head tracking system. The remainder of the chapter details necessary system models and parameter representations and concludes with a presentation of results obtained on a synthetic data set.

Vital data measurements in the proposed head tracker are pixel coordinates used in the SfM algorithm to estimate 3D head motion. These pixel measurements arrive from a two-dimensional (2D) tracking module presented in Chapter 4. The chapter begins with an introduction to the different 2D tracking techniques presented in the literature and then covers an efficient tracking algorithm employed in this work. The complete tracking system is presented in Chapter 5, where feedback techniques employed to improve robustness and

stability of the head tracking system are expanded upon. Robustness in the thesis is defined in terms of the systems ability to handle significant disturbances, or "outliers", that do not obey the assumed model of input data. This definition focuses on the idea of a breakdown point (or worst case scenario) where some smallest number of outliers can cause the system estimator to produce arbitrarily bad results. The system is evaluated on real data in Chapter 6, in terms of computational performance, robustness to outliers and numerical accuracy (relative to groundtruth motion data), confirming the applicability of the approach as a global motion parameter estimator in a model-based coding system for faces.

1.2 Very Low Bit Rate Video Coding

Throughout our everyday lives, the human vision system is constantly bombarded with a myriad of images. These images are almost instantaneously interpreted without the conscious mind even realising that some form of information processing has been taking place.[5]. Some of the more interesting objects and scenes people come across are those involving human faces. As well as the incredible ability to identify individuals amongst a large number of faces, a vast amount of interpretable information can be gleaned from just a passing glance of a person. A persons mood or attitude can be ascertained from even the slightest change in the facial mimic. Faces are of primary interest to humans as they reflect information integral to identification and communication between other humans.

From a technical perspective, much research work has been devoted towards creating convincing synthetic facial animation, a task that is complicated by an acute human sensitivity towards small errors in the facial mimic. Even small deviations from what is naturally expected can result in artificial looking animations. In recent years however, much development has occurred in facial modeling and rendering with highly realistic facial animation now available in many applications. One example of this is the offline rendering of virtual characters in movie productions, a familiar application to most people. Interestingly however, the ever increasing computational and graphical power available now facilitates the realtime implementation of facial animation techniques on small handheld devices such

as personal digital assistants (PDAs) or smart phones. Furthermore, these techniques are not limited to the production of synthetic graphics content but can also contribute to improving the quality and efficiency in communication applications [5].

It is often the case that handheld devices are connected to the Internet over low-bandwidth wireless channels. Facilitating the use of video-phone applications or the streaming of video clips under such low bit-rate constraints requires high compression ratios of the raw video data. Established coding standards for efficiently compressing video data include hybrid video codecs such as H.261/263/264 or MPEG-1/-2. These coding schemes use block based motion compensated prediction in combination with residual coding to achieve compression ratios of several $100 : 1$ at acceptable visual quality.

In situations where much higher compression is required, more sophisticated scene models can be utilised which exploit some *a priori* knowledge of the scene that can be made available to both encoder and decoder. This relatively new and rapidly evolving technique is known as model-based video coding (MBVC), [6, 7, 8, 9, 10], where three-dimensional computer models are used to describe objects in a scene. The graphical model representations are transmitted only once, with all following scene changes described by motion, deformation, or lighting parameters [5]. Given that only a few update parameters need to be encoded for each frame of the video sequence, extremely low bit-rates can be achieved.

Transmission of large numbers of 3D models in order to describe complex scenes is unrealistic. Thus restricting the number of models implies the optimal use of model-based coding for applications such as video-phones, video-conferencing and streaming of video content like news casts [5]. These scenarios typically comprise one or more people in the foreground with the background remaining relatively static or of limited interest. The scene can thus be adequately represented by 3D head-and-shoulder models that are animated in order to show motion and facial expressions.

1.3 Model-Based Video Coding

Model-based coding schemes, [6, 7, 8, 9, 10], utilise three-dimensional computer models to define the appearance of objects in the scene, streaming only high level motion, deformation and lighting information to represent the dynamic changes in the sequence [5]. In the specific case of head-and-shoulder video sequences, the basic concept of a model-based coder is depicted in Figure 1.2.

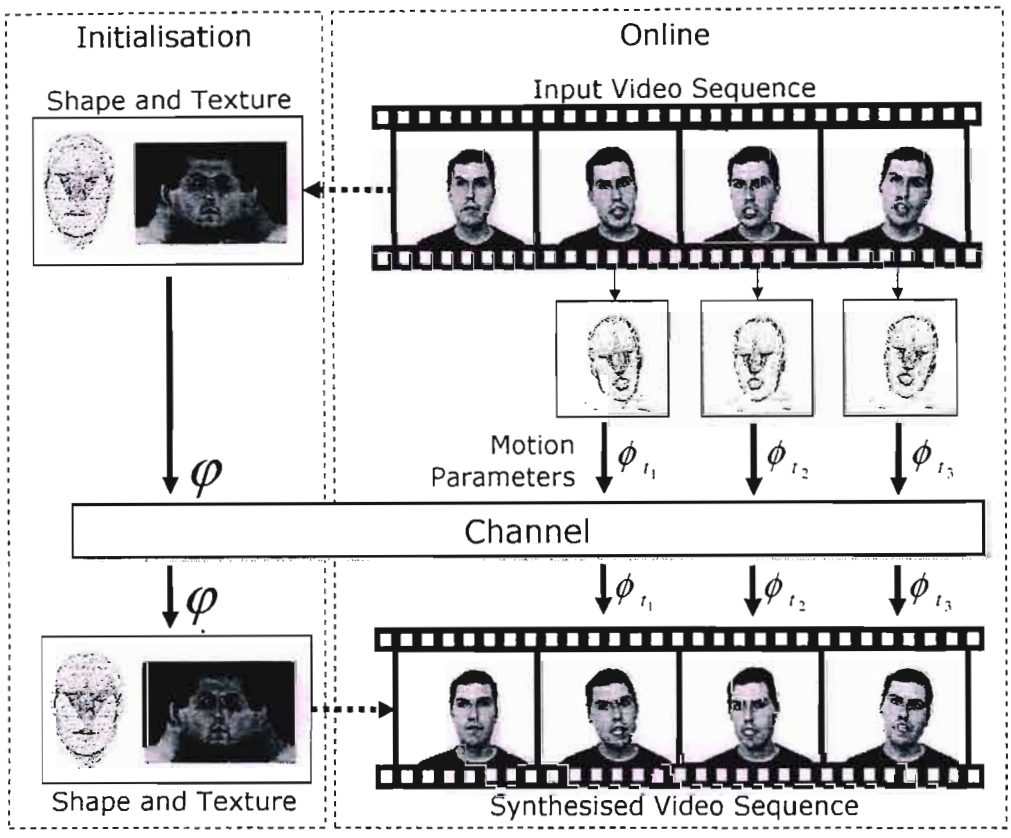


Figure 1.2: Basic concept of a Model-Based Video Coding system.

An input video sequence is captured, for instance, from a camera viewing a person participating in a virtual conference scenario. During an initialisation phase, the first frames are used to fit or create a 3D head model, specifying shape and texture information for the person in the scene. The model specific data is encoded and transmitted only once (unless it has already been stored at the decoder in a previous session). Online, the encoder then

analyzes the video sequence and estimates the 3D motion of the subject in the scene. This motion, which is further discussed in Section 1.5, typically includes information describing both global head motion and local facial expression change. This motion data is then streamed over the channel. At the decoder, the 3D head model is deformed according to the received motion information and new frames are synthesized using computer graphics techniques. Since only a few parameters have to be transmitted for each frame of the sequence, bit rates of about 1 kbit/s can be achieved [11].

Achieving such low bit-rates makes MBVC a very desirable coding scheme for face-to-face communication systems. For this reason, Section 1.4 presents some of the major developments in face analysis and model-based coding of faces. Thereafter, Section 1.5 gives a more detailed review of the required model animation parameters.

1.4 Advancement of Model-Based Coding Schemes

In face-to-face communications, such as a video phone or a teleconference scenario, the subjects to be coded in the scene are mainly people's faces. The development of a Facial Action Coding System (FACS) by Ekman and Friesen [12] in 1978 formed a very important precursor in the evolution of model-based coding systems for these scenarios. Their work focused on the investigation into the relationship between emotion and facial expression. As a part thereof, the FACS was developed as a way to quantitatively measure the intensity of facial actions. Each minimal facial action was called an *Action Unit* (AU) and with the combination of about 50 Action Units they found that it should be possible to analyse every facial expression. These Action Units have subsequently become popular as parameters for animation of human faces and the FACS was one of the most widely used systems until the MPEG-4 definition [13] of Facial Animation Parameters (FAPs).

In 1983, Forcheimer and Fahlander [14] presented the first modern scheme for model-based video coding of faces. This was followed by work published in [15], in which Action Units were estimated from video sequences using a simple wireframe face model (known as Candide [1]) that was suitable for real-time animation. Candide is still a popular model but

current facial animators typically use more advanced models. One of the drawbacks with these original systems was their inability to portray realistic skin tones and facial features, primarily relying on unlifelike skin coloured polygons. The problem was soon rectified in 1987, when Aizawa [16] and subsequently Welsh [9] introduced texture-mapping into the model systhesis aspect of the system. Texture-mapping techniques created the potential for photo realistic model-based coding. The developments in the works of Welsh and Aizawa also made the concept of model-based coding more popular because it convinced people that good quality face image communication could be achieved using these techniques.

The next major step in the evolution of model-based coding was the introduction of analysis-by-synthesis coding [17]. Here, the encoder contains a feedback loop holding a copy of the decoder, enabling it to use the synthesized image and compare it to the original image. The transmitted parameters can then be optimised at the encoder by minimizing the residual image (difference between the real and synthesised image). A further benefit is the transmission of the residual image as a means of improving image quality at the decoder. These developments paved the way for robust video analysis.

Thus the early nineties saw an increase in model-based coding and face animation as research topics. A survey on many of these can be found in [18]. Intensive research has been done into such aspects as face detection [19], face tracking [20, 21, 22, 23] and facial expression analysis [24, 25, 11]. At the same time, face models have become increasingly advanced, some consisting of thousands of polygons and having advanced motion patterns to facilitate the synthesis of natural looking animation. In 1999, the international standard MPEG-4 [13] was ratified, part of which includes definitions for the representation and coding of Facial Animation Parameters (FAPs). The quantity of research into model-based coding and face analysis is now vast, with the possibility of new media forms making it an enticing avenue for industrial and commercial development. Although the intended application in this work is low bandwidth video transmission for handheld devices and teleconferencing, much research is currently directed towards facial animation in movies, computer games, and Web based technologies such as avatar animation.

1.5 A MBVC System for Faces

This study focuses on model-based compression in videophone or teleconferencing applications with emphasis on face-to-face communication. This permits a breakdown of scene information into two primary parameter sets. These are referred to as global and local motion information [26], which are generally considered sufficient to describe frame-to-frame head motion and local deformation of facial features of the subject being viewed. Global motion information describes the 3D pose of the head in each frame with a set of rotations and translations. Local motion information is defined by the set of facial expression parameters for the face, which, since the release of the MPEG-4 standard [13], are known as *facial animation parameters* (FAPs).

A typical model-based scheme for face coding is shown in Figure 1.3. Along with common knowledge, the figure highlights the various processes required at the encoder and decoder to anaylse and then synthesise face images. The parameters transmitted over the channel include global and local motion information as well as any model or background updates.

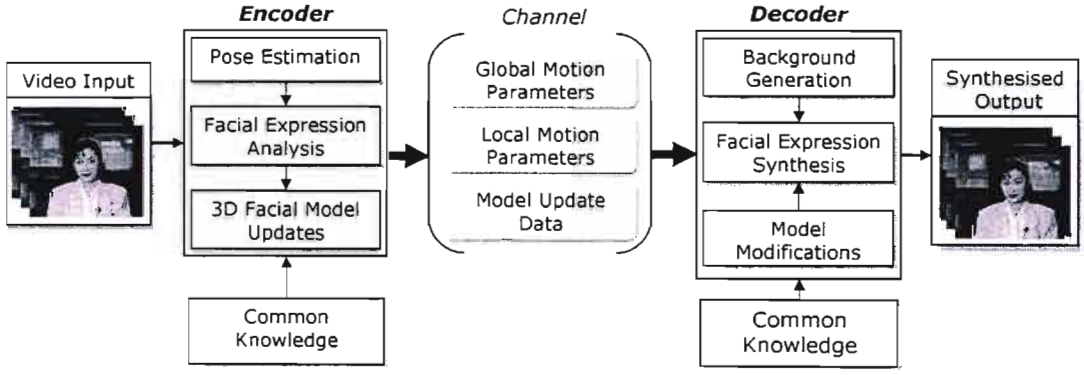


Figure 1.3: A MBVC system for face communication.

This type of model-based coder consists of three parts: common knowledge, an encoder and a decoder. Common knowledge provides a basis for the analysis-synthesis aspect of the facial images and includes a 3D facial model and general knowledge for facial expression analysis, i.e. the set of FAPs. The 3D facial model consists of the 3D shape and the texture of the person's face to be coded. Global and local motion parameters are estimated at the

encoder and the facial model is updated from the current input image. These parameters are encoded and transmitted over the channel. The transmitted motion parameters (global and local) are used at the decoder to modify the facial model and synthesise a new face image. The decoder may also generate the background of the output images locally or synthesise the background using the background data from the first frame of the video sequence. The transmitted update data may include the corrected depth and updated textures for the 3D facial model, as well as some newly exposed background data, which are now used for more precise analysis of the scene and improving the quality of the synthesised image.

1.5.1 MBVC System Components

With the advent of advanced and highly realistic computer graphics and a set of standards for facial animation described by MPEG-4 [13], the decoder aspect of a model-based coding system is essentially a solved problem. In the encoder however, several key components can be identified that need further development.

- ***Face detection*** is necessary to know *if* there is a face in the image and *where* it is located.
- ***Model initialisation*** is required to place the facial model correctly in three dimensions and may also change the shape of the three-dimensional model in order to make it conform to the geometry of the face.
- ***Facial texture coding*** is needed to compress and send the texture of the face to the decoder.
- ***Head tracking*** (global motion tracking) ensures that the model mimics the three-dimensional rigid body motion of the head.
- ***Facial gesture tracking*** (local motion tracking) extracts facial expression information, such as smiles and eye blinks, in the form of FAPs.

- ***Model refinement*** continuously updates the fitted three-dimensional head model by adding texture and changing the shape.

Each of these components presents a unique set of problems that need to be addressed in a realistic model-based face coding system. In this thesis, focus is placed on a solution to one of these problems; that of estimating the global motion parameters (head tracking) describing the three-dimensional pose of the head in each frame. This topic alone has received much attention in current literature. An overview of many of the proposed estimation techniques is provided in Chapter 2. At this point however, Section 1.6 defines the set of necessary global motion parameters and what they represent.

1.6 Problem Definition

This thesis proposes a solution to the head tracking problem in terms of estimating a set of three-dimensional motion parameters. These parameters describe the rigid, three-dimensional movement of the head in a close-range video scene. In this regard, a definition of the proposed task is beneficial:

DEFINITION *From a monocular image sequence of a moving head, estimate the three-dimensional pose (position and orientation) of the head, assuming that tracking begins from some known initial pose, i.e. the head is in a frontally facing position and occupies some minimum area of the image.*

A *monocular* image sequence implies that the video is captured from a single camera, not from a *stereo* camera rig. The aim of this thesis is to achieve some of the goals of video telephony over handheld devices (a typical model-based coding application) for which two-camera systems are an undesirable expense.

The desired outputs of head tracking are three-dimensional pose parameters, explicitly defined as a set of rotations and translations relative to some chosen reference frame, an issue that is further addressed in Chapter 3. The three rotations required are the Pitch, Yaw and Roll of the head with corresponding translations along the x, y and z axes

respectively. This relationship is easily interpreted from Figure 1.4.

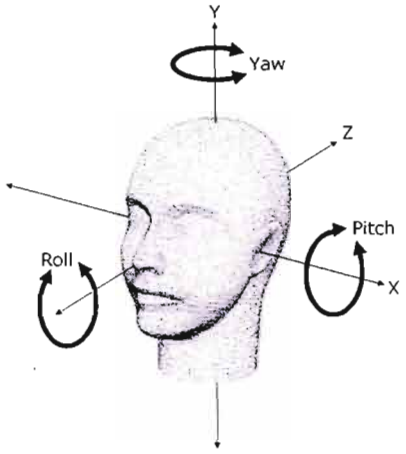


Figure 1.4: Rigid 3D animation parameters.

1.7 A Note on Accuracy

In terms of accuracy, a note must be made regarding the fidelity of images synthesised at the receiver of a model-based coding system. In traditional coding schemes, a conventional measure of fidelity is to find the normalised mean square error (NMSE) between the received image and the original at the transmitter. In such cases, there typically exists a direct relation between the NMSE measure and the subjective visual quality of the image. In model-based coding systems, this measure of synthesised image fidelity may not be applicable, as presented in an argument by Welsh in [9]. Below a similar argument is presented (based on that in [9]) in view of the goal of estimating global motion parameters in this thesis.

If one considers a sequence involving motion; say for instance a sequence of a person moving their head from side to side or up and down, the primary requirement for success at the decoder would be to display a synthesised sequence where the person is moving their head from side to side or up and down. If the precise speed, angle and position of the head displayed do not *exactly* match the original sequence, it may not really matter to the person viewing the received sequence as there is no noticeable defect in the generated

image. If the NMSE was taken between the synthesised and original frames however, the difference might be significant.

In a model-based coding system, the model of the person that is to be synthesised is animated using the parameters generated during analysis of the original scene. If inaccuracies in parameter estimation occur during the analysis, they may not, in certain circumstances, be a critical issue, implying that the traditional measure of fidelity may no longer be appropriate. The quality of the received image may be judged subjectively on its ability to deliver social cues and meaning.

This argument is presented with specific reference to the evaluation of results presented in Chapter 6. In many head tracking systems, [27, 28, 29, 30], the accuracy of the estimated pose parameters is compared against a set of ground truth motion from a commercial magnetic tracking system (ground truth data refers to a set of system measurements acquired from an authenticated source). Although similar ground truth testing is performed in Chapter 6, it is proposed that results can also be evaluated from a visually subjective level on their ability mimic the motion exhibited in the original video sequence, which is in agreement with the fidelity argument above. The objective of this work is not to achieve exact image reproduction, but rather in reproducing the same human perception of the facial activity occurring in the original video sequence.

In this thesis, mimicry is demonstrated via the animation of a "fitted" wireframe head model (Candide [1]) which is projected over the original image sequence. It is argued that the coherent motion of the face model with the apparent rigid motion of the face in the sequence further confirms the authenticity of the estimated motion and meets the criteria for rigid face animation in a MBVC system.

1.8 Contributions and Publications

This work has been the subject of several local conference proceedings publications, which include [31, 32, 33, 34, 35]. In these works, the interaction between the Structure from Motion and 2D tracking solutions (which form the foundations of this thesis) have been

progressively refined. The final tracking system is presented in [35] and employs several modifications to the tracking systems presented in the earlier publications. Novel contributions include the use of an affine motion model in the 2D tracking aspect of the system and the addition of an appearance basis. These provide an accurate, yet efficient revision to the original tracking systems proposed in publications such as [32] and [33], and are the topic of discussion in Chapter 4. Another variation this work employs is taken from ideas originally proposed in [22], where the concept of harnessing multiple points on tracking templates has been utilised to increase the number of measurements available at any time. This modification is discussed in Chapter 5.

Chapter 2

Related Work

2.1 Introduction

Head tracking and pose estimation is a well researched, but far from solved problem in the computer vision community. It is a popular topic with application in Model-Based coding schemes [3], Human-Computer Interfaces [36], video teleconferencing [37] and avatar animation [4]. The number of proposed solutions is vast and the degree of motion recovery varies. A good comparison of accuracy and level of recovery of a number of different tracking methods is presented in [23]. Systems are categorized by their algorithm characteristics, ranging from: colour blob, motion blob, depth blob, edge, feature, template and optic flow. The first three types refer to systems that track colour, motion or depth clusters, respectively. Systems are also differentiated according to their recovery algorithms, which are similarly classified. In [23], two interesting conclusions are drawn. Firstly, certain cues, like colour and motion are useful for positional tracking and recovery since they can be computed quickly, but lack the precision and robustness necessary for full 3D tracking. Secondly, 3D tracking is predominantly based on either feature or template algorithms, with template techniques providing better robustness and feature-based methods displaying better computational efficiency.

Some of the fundamental evaluation criteria for head tracking systems are their robustness

to occlusion and illumination variation and the ability to recover large motions, all within real-time computational constraints [23]. A real-time system is expected to process each image is less than 40 ms, meaning that the system runs at 25 fps.

The primary difficulty in recovering motion is experienced when out-of-plane rotations (Pitch and Yaw) occur, where the face is no longer parallel with the image plane. Most systems are able to handle some subset of the above problems, but greater levels of robustness tend to come at the expense of further computational cost. In the following sections, a review of works that propose solutions to the problem of 3D head motion recovery is presented, along with their proposals for illumination and occlusion handling. These systems include optical flow based methods, active appearance methods, template approaches and Kalman filtering techniques. In closing, Section 2.6 proposes a solution based on tracking salient features coupled with an Extended Kalman Filter.

2.2 Optical Flow Based Methods

Much work on pose estimation has been produced using optical flow based methods. The principal idea of optical flow is to generate a motion field from the individual velocity vectors for each point in an image. The flow field can then be analysed in some appropriate fashion to recover the necessary dynamics of a scene. An overview of the different optical flow techniques is presented in Section 4.2.1 of Chapter 4.

One of the most commonly cited optical flow methods for pose estimation is that of Black and Yacoob [38]. They developed a regularised optical-flow method that uses an eight parameter 2D affine model of flow. Regularising the flow field implies constraining the flow computation using some suitable model, in this case the parametric affine model. The implementation yields good results for pose estimation in a facial expression recognition system. Unfortunately the use of a plane-like model limits accurate tracking to medium-size head motions and the method fails for very large rotations. In [38], about 2 minutes of processing time is required per frame on a Digital Alpha 3000 machine.

Much research, [21, 39, 40, 6, 41] identified the need for better fitting head models for reg-

ularising the optical flow, such that larger head motions could be handled. Basu et al. [21] use an ellipsoid model instead of a planar model. Initially optical flow is computed independently of face position and orientation using a gradient-based method. Then the rigid motion of the ellipsoidal mesh model that best accounts for the observed flow is interpreted as the motion of the head, thereby regularising the flow computation. The performance against the planar model of [38] is compared and the ellipsoidal model is found to give better results in terms of its ability to conform to the various orientations of the head. Unfortunately, the method's strengths are also its weaknesses. It manages to cope well with large head rotations since it does not rely on any fixed features. For the same reason, it has no means to ground the model to the face, thus the error accumulates and the ellipsoidal mesh used to regularise the optical flow slowly drifts off the face. Also, the method is computationally expensive, requiring approximately 30 seconds per frame on an SGI workstation.

Other work which modifies the model fitting approach to regularise the motion field has been performed by Zhang and Kambhamettu [39]. They use an Extended Superquadric model as a better fit than the ellipsoidal head model. They also use a motion segmentation algorithm to provide robustness against partial occlusion. The occluded areas are detected and discarded as noise. A further post regularisation method based on edge flows (motion field of edge points) is also employed to reduce error accumulation, making it more stable over long occlusion sequences. Like the other systems however, the system does not run anywhere near real-time.

In [40], Roivainen uses the three-dimensional head model, Candide [1], to parameterize the optical flow. Li, Roivainen and Forchheimer [6] then extend this work in three ways. Firstly, they set up a nonrigid affine motion model in which they can parameterize the knowledge about facial shape and expression. Secondly, they introduce motion prediction so that temporal motion information as well as spatial motion is used. Lastly, they use a rendered version of the head to provide a level of feedback, which greatly improves the performance of the system. Without the feedback stage, the errors in the motion estimation stage accumulate, and the tracker is susceptible to drift as in the systems of [38, 21].

DeCarlo and Metaxas [41] use a complex polygonal head model constructed from anthropological data. They extract optical flow at some feature points and regularize it by the movement of the model. The measurements are then stabilized in a Kalman filter framework. Again, using optical flow leads to a similar error accumulation as in [21] and [38], but the DeCarlo et al. system employs additional face edge information to prevent divergence. As in the case of [6], this work is also extended to extract face shape and facial expressions. The system handles large rotations well and runs at about 2.5 seconds per frame on a 175 MHz R10000 SGI O2 workstation.

Finally, Harville et al. [42] use depth measurements from a stereo camera rig to extend the optical flow brightness constraint with a depth constraint. This increases robustness for large rotations out of the image plane and for translation in depth, but the system carries the cost of requiring stereo input.

2.3 Active Appearance Techniques

In [43], Cootes et al. analyse images of faces using statistical models of the shape and grey-level appearance of the face. The appearance model, given a good enough training set, can generalize to almost any valid example of the face represented, potentially giving a full photo-realistic approximation [44]. Fitting an Active Appearance Model (AAM) to an image consists of minimising the error between the input image and the closest model instance; i.e. solving a nonlinear optimisation problem. Applications include face and expression recognition [45] or face pose estimation and tracking [27, 46].

Even though Active Appearance Models are two-dimensional, they can cope with a certain amount of out-of-plane rotation by deforming the associated shape modes. By switching between several two-dimensional models, a wide span of orientations can be covered.

In [47], Xiao et al. show how AAMs can be used to model 3D phenomena in faces. They use a non-rigid structure-from-motion algorithm on tracked feature points to construct 3D shape modes that correspond to an associated 2D AAM. Conversely, this leads to its use as a means of constraining the AAM so that only model instances that can *also* be

generated with the 3D shape modes are generated. This could then be used to fit AAMs across pose, illumination and occlusion constraints.

Cascia et al.[27] use a three-dimensional version of the Active Appearance Method. Instead of using two-dimensional PCA, they use a three-dimensional texture mapped cylindrical model for the head. Head pose tracking is formulated as a registration of the incoming face image with the cylinder's texture mapped images under different face poses. To solve the registration problem in the presence of lighting variation and head motion, the residual error of registration is modeled as a linear combination of texture warping templates and an orthogonal illumination basis (where the illumination basis is person independent). The method achieves near real-time performance, running at 15 frames per second on a SGI O2 graphic workstation

Dornaika and Ahlberg [46] also use Candide [1] for face tracking with Active Appearance Models. The shape parameters used in this work allow for both local and global motion estimation.

Zivkovic et al. [48] use a generic 3D model, texture mapping it with the initial frame of the sequence. A limited number of significant points on the model are used to improve performance, and an adaptive appearance changes model is used that iteratively updates as new images in the scene arrive. The appearance model is then constrained to a neighbourhood of the initial gray values in the starting frame. This is different from the general Active Appearance methods discussed thus far, because there is no *a priori* appearance basis for the model when tracking begins. Because the algorithm is so heavily reliant on the initial image however, the system works best for small motions of the head.

2.4 Template Techniques

Some methods do not fall directly into the above two categories. The work of Xiao et al.[29] is one, which is categorised here as a template based method. Their system uses a cylindrical head model with a reference template from an initial frame which is able to fully recover the full 3D motion of the head. The method achieves robustness via

a combination of three techniques. Firstly, it uses iteratively re-weighted least squares (IRLS) in conjunction with the image gradient to accommodate for non-rigid motion and occlusion. Secondly, a dynamic template update is performed which helps to diminish the effects of self-occlusion and illumination changes. Thirdly, a re-registration of the current image to an initial reference template is performed whenever head pose is close to that in the reference. The method has been shown to be robust and give accurate results under various testing conditions with a wide range of motion recovery, as well as running near frame-rate at around 15 fps.

2.5 Kalman Filtering Methods

Kalman filtering has been a popular method of data fusing in many tracking algorithms as it provides an efficient method of smoothing estimates of system parameters, or integrating measurements of the system in a recursive manner for online parameter estimation [49]. Using a Kalman filter for data smoothing is usually a secondary stage used to constrain the dynamic changes that can occur in a series of system estimates that have already been gathered from some other estimation module. More often however, the filter is used to infer state information of a system based on a predictive, corrective process, where the corrective step.updates state estimates using measurements of the system. A more extensive discussion on Kalman filtering is presented in Section 3.5 of Chapter 3. Standard references include [50, 51].

Cordea et al. [52] present a "$2\frac{1}{2}$"D tracking method for the recovery of the three-dimensional position and 2D orientation of a head moving in the image plane. The method uses a 2D elliptical head model and region and edge-based matching algorithms with a linear Kalman filter for fusing measurements of the system and maintaining optimal estimates of system state.

Zhu et al. [30] propose a system that performs 2D face detection as well as 3D pose tracking in one. A linear Kalman filter is used along with face appearance and a Sum of Squared Difference (SSD) matching criterion to constrain the 3D face pose in each frame.

SSD matching entails minimising a cost function in order to estimate some unknown parameters; in this case the motion parameters that bring a projected face view image and the detected face view image into alignment. Eye detection is also performed using IR illumination, which is subsequently used to further constrain the possible 3D pose solutions, reducing the possibility of drift.

An extension to the linear Kalman filter is the extended Kalman filter (EKF) for handling non-linear dynamics. Azarbayejani and Pentland [28] reformulate the Structure from Motion (SfM) problem as a recursive parameter estimation problem that is solved using an EKF. Their representations are used in this thesis and are covered in detail in Chapter 3. One of the applications presented in their work [28] is head tracking. A small number of features on the head are tracked using a 2D correlation type matching algorithm, and the location of these features are used to infer the three-dimensional pose of the head, as well as the three-dimensional locations of the features as they move along their trajectories.

Much work subsequently extended the foundations laid by Azarbayejani and Pentland with their extended Kalman filter solution. Jebara and Pentland [22] build on the work of [28] for head pose estimation with an automatic initialization procedure that extracts significant feature windows around the eyes, sides of the nose and corners of the mouth. These windows are used as templates in an efficient tracking technique [53] that allows in-plane rotation and scaling of the image patch. The extra degrees of freedom on the patch (as compared to a purely translational model) mean that two feature points can be used for each template, thus decreasing the number of templates needed for the system to be observable. The three-dimensional structure that is obtained from the filter is constrained via a projection onto an eigenspace of depth maps taken from Cyberware [54] head scans. The regularized structure is then used as a starting point for the feature tracker in the next frame by projecting the three-dimensional structure onto the image plane. Therefore, if a template is ever lost during tracking, it has a good chance of being found again using the better starting estimate from the projected structure. The errors in the feature measurements are fed to the Kalman filter framework where a reliability weighting can be integrated with each measurement so that bad features may be trusted less than good ones. The algorithm runs at 30 Hz (real-time) on an SGI O2 200 MHz

machine.

A number of systems have since emerged which harness various elements from the work of Jebara and Pentland [22] and use the Structure from Motion framework of Azarbayejani and Pentland [28]. Cordea et al.[4] use the framework for head pose estimation in virtual reality type applications. Strom et al. [20] combine the approach with a texture-mapped version of the head model, Candide [1], to constrain tracking from frame-to-frame. Feature points are selected from the texture-mapped model, and tracked via a normalised correlation technique. Once the EKF has estimated the pose, the face in the scene can be warped back to a frontal position for eigenspace coding of the mouth. The system runs at 30 Hz. Calvagno et al. [55] also use the Candide model and have included non-rigid motion estimation for facial expression recovery.

From the proliferation of work specifically using the above SfM framework, it is evident that the technique is a popular method for head pose recovery. Besides exhibiting the highest frame rate, the method is easily modified to include robustness to occlusion. The system proposed in this thesis is largely based on this SfM formulation and the feedback techniques proposed by Jebara et al. [22]. Section 2.6 outlines the proposed system.

2.6 Proposed System Choices

The aforementioned methods can be evaluated on several criteria that are crucial to any head pose estimation system [23]. These include range of recovery, occlusion handling, robustness to varying illumination and computational performance. The foremost of these is the range of recoverable motion.

From the literature, [22, 29, 48], out-of-plane rotations (Pitch and Yaw) of between 35° to 45° are considered large, thus this range will be used to distinguish good pose estimators from poorer ones. Another difficult parameter to estimate is depth, as there is no directly recoverable depth information from a single view.

The optical flow based methods which incorporate a three-dimensional model to regularise

the flow field, [21, 39, 41, 6], achieve these pose estimation goals for rotation. A strength of the optical flow approach is the use of the whole face in the estimation process, not relying on salient features which can become lost and cause some systems to fail. This strength however, is also a weakness, as the model is not always effectively tied to the head and the flow field eventually diverges. Optical flow based methods are also computationally expensive as they need to gather measurements from a large number of points in order to achieve robustness. Furthermore, given that optical flow can only be measured accurately in regions that contain edges, it seems wasteful to include optical flow from edge-free regions into the calculation.

The appearance modeling techniques, [27, 46], template based approach ,[29], and feature based Kalman filtering approaches, [22, 30], all achieve acceptable range of estimation for 3D pose. At the time the work in this thesis began however, the systems in [29, 30, 46] were unpublished, so they could not be considered as methods for solving the pose estimation problem.

To deal with varying illumination, the system of Cascia et al. [27] employs an orthogonal illumination basis, but the system does not have any facility for handling occlusions. Conversely, the system of Jebara et al. [22] is designed to handle partial occlusions, but does not incorporate any provision for general appearance variation. In terms of performance however, only the Adaptive Feedback Tracker (AFT) from Jebara and Pentland [22] achieves full frame rate (i.e., 25 Hz) even though the system by Cascia et al.[27] comes close with 15 Hz. Performance benefits come from the use of a point configuration based on individual features and an efficient 2D tracking algorithm. Partial occlusion is handled within the Kalman filter framework, where bad (high noise covariance) feature measurements from the 2D tracker can be neglected until they become visible again. A coherent structural estimate from the extended Kalman filter is also used to add an extra level of control over the feature tracking process. The combination of these estimation and control mechanisms in this approach make it an attractive solution to the head pose estimation problem.

In this thesis, a head tracker similar to that of Jebara and Pentland (with some exten-

sions) is proposed. Similar to the concepts of appearance modeling used in the Active Appearance methods described in Section 2.3, general appearance variation is integrated into the 2D tracking technique by extending it with a small appearance basis. This basis is made up of a number of sample images of the object (namely the head) under different appearance conditions. The basis is then integrated into the 2D tracking model and contributes towards tracking accuracy when lighting conditions or aspect, due to pose or facial expression, vary in the scene. This addition essentially harnesses the strengths of the system from Cascia et al.[27]. Within current head tracking literature, this extension has not been proposed for any head pose estimator using a SfM framework.

The original head tracker of [22] uses a 2D similarity transform for the feature tracking motion model. This allows for in-plane rotations and scaling, but cannot sufficiently model the affine or projective transformations that can occur when the head undergoes out-of-plane rotations. In this thesis the use of an affine motion model for 2D feature tracking is proposed. An affine model is able to deform a template region to most orientations of moving facial features, so long as the region is visible, albeit warped, in the image. The use of a higher order motion model for the template, as argued in [22], means that more points from each template can be harnessed, thus decreasing the number of templates needed for the system to be observable. The 2D tracking method and its extension with an appearance basis is covered in Chapter 4. The extended Kalman filter is covered in Chapter 3 and a complete overview of the system is presented in Chapter 5.

Chapter 3

Structure from Motion

3.1 Introduction

Structure from Motion is a fundamental problem in computer vision and has been the subject of large volumes of research in the vision community [56, 49, 57, 58]. While much progress has been made into the theoretical aspects of the area, engineering practical solutions is a less developed science [56].

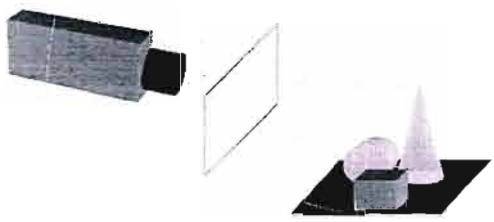


Figure 3.1: A standard Structure from Motion setup showing relative camera to object motion.

A typical SfM setup is shown in Figure 3.1. The aim of any SfM algorithm is to derive 3D motion and structure of an object from observed features on the surface of that object. Depending on the nature of the features observed (two- or three-dimensional, points or lines on the surface of the object etc.) different formulations and algorithms are utilised.

All algorithms rely on several simplifying assumptions about the scene being viewed. One key assumption is that objects in the scene are moving rigidly (or equivalently, only the camera is moving with respect to the stationary object). An additional simplification is that there exists a module which pre-processes the camera's images to consistently locate, extract and label features in the scene. In each frame, the features are detected and associated to their corresponding instantiations in the other frames. These (usually noisy and error-prone) measurements are the inputs to the SfM problem.

As discussed in [56], the choice of SfM algorithm is restricted to the type and range of the data available. Techniques are generally compared under two primary methodologies. Early two-frame techniques dominated much of the initial SfM research, until multi-frame methods became more popular in the mid to late 1980s. Under a multi-frame approach, batch processing techniques have been popular and have been shown to give good results, but sometimes available measurements are causally restricted and a recursive or "fusing" approach has to be used.

So called fusion approaches update the SfM solution as new image data is obtained. A popular way to do this is with variants of the Kalman filter [59, 28]. Oliensis [56] describes the benefits and drawbacks of fusion versus batch approaches. He makes the point that a fusion solution is only as robust as the local updates made to the SfM solution while a batch solution can use the whole sequence of data to adjust the global solution; inevitably leading to better results. A critique on currently popular SfM algorithms can be found in [56], and a review on the SfM task can be found in [58, 49].

The feature data available in this work takes the form of 2D pixel measurements from a tracking module, as covered in Chapter 4. The images in a model-based video coding system arrive sequentially, so an online solution is favourable in order to process data as it arrives. This means that a recursive solution should be considered, which can use past

and present measurements to give an optimal solution for the camera/object motion and structure.

3.2 A Recursive SfM Solution

A recursive SfM solution is applied when multiple images are not available in batch form but arrive in a temporally incremental way, frame by frame. One way of formulating such a multi-image solution is by repeating a 2-frame estimate such as the relative-orientation problem proposed by Horn [60]. In such a solution each new pair of frames is used to compute an estimate of camera and object structure. Oliensis et al. [61] and Soatto et al. [62] sequentially compute such 2-frame estimates and post-process the output with a smoothing Kalman filter (KF). Here, the measurement vectors and the state vectors are the same so the KF is linear, completely observable and hence does not have any linearization problems [49]. In fact, the KF is only acting as a smoothing filter. It is not really being used in its full capacity as a state estimator where the measurements can be nonlinearly inverted to obtain state information while keeping track of the system's complex internal dynamics.

Extended Kalman Filters (EKFs) deal with nonlinearity explicitly and can be applied to nonlinearly uncover motion and structure instead of smooth the output of 2-frame techniques. EKF frameworks have been the focus of much attention and have been utilized on image sequences by Broida et al. [63], Matthies et al. [64] and Young et al. [65]. A seminal paper by Broida, Chandrashekhar and Chellappa features a nonlinear EKF for recovering state information [59]. It does not rely on 2-frame techniques but rather folds the estimation into the Kalman filtering equations. The filter is used to (nonlinearly) invert the feature measurements to gather motion and structure information.

The EKF based approach presented in this thesis is built on the recursive formulation proposed by Azarbayejani and Pentland in [28]. Emphasis is placed on casting the SfM problem into a dynamic system framework, utilising some important representational modifications that contribute to the implementation being accurate and stable. In addition, the system integrates the temporal information over the sequence of images in a proba-

bilistic framework within the Kalman filter. Therefore, it is robust to errors in 2D feature tracking, as discussed in Sections 5.4 and 5.5 of Chapter 5. The approach introduces a unique parameterisation of well known geometric concepts and representations. In so doing, the SfM solution not only recovers the full 3D metric structure and motion of the camera, but is also able to recover the camera focal length, an important parameter which is normally assumed to be fixed and/or known in other SfM formulations.

3.3 System Dynamics

In order to formulate an appropriate model, some fundamental constraints on the dynamics involved in the system need to be established. It can be logically assumed that cameras do not teleport erratically around the scene, or conversely, the objects do not move about too suddenly. These bodies are governed by physical dynamics and, as a result, it makes sense to constrain the possible configurations of the camera to undergo smooth temporal changes over a causal time sequence.

Referring to Gelb [50], a typical time varying dynamic system (from time step k to $k+1$) is represented as:

$$\begin{aligned} \bar{\mathbf{x}}_{k+1} &= f_k\left(\bar{\mathbf{x}}_k\right) + \bar{\mathbf{w}}_k, \qquad \bar{\mathbf{w}}_k \sim N\left(\bar{0}, Q_k\right), \\ \bar{\mathbf{z}}_k &= h_k\left(\bar{\mathbf{x}}_k\right) + \bar{\mathbf{v}}_k, \qquad \bar{\mathbf{v}}_k \sim N\left(\bar{0}, R_k\right), \end{aligned} \tag{3.1}$$

Here, $\bar{\mathbf{z}}$ represents an observation vector at each moment in time as a concatenation of measurements of the system. The observations are caused by the dynamic changes of a set of internal state parameters, $\bar{\mathbf{x}}$. It is not possible to directly measure $\bar{\mathbf{x}}$, but one can observe the effect of any changes through the measurement vector $\bar{\mathbf{z}}$. In this SfM formulation, the measurement vector will be a set of 2D feature coordinates and the state vector will contain the parameters which describe the structure and motion of the object through the scene.

The dynamics function $f_k(\cdot)$, measurement function $h_k(\cdot)$ and the noise statistics Q_k and

R_k are assumed to be known. In this thesis, the mapping from $\bar{\mathbf{x}}$ to $\bar{\mathbf{z}}$ via $h_k(\cdot)$ is a nonlinear function which projects three-dimensional coordinates in the camera frame to two-dimensional coordinates on the image plane, corrupted by some noise $\bar{\mathbf{v}}_k$. Here, the noise is represented as an additive Gaussian process with zero-mean and time varying covariance R_k. The matrix R_k probabilistically encodes the accuracy of the measurements (2D feature coordinates). By entering large covariance values for feature measurements, this property of R_k can be used to represent features that are missing in certain frames, a focus of Section 5.4.

In addition, the dynamics of the internal state are constrained. The 3D structure, 3D motion and camera geometry do not vary haphazardly but are linearly dependant (via $f_k(\cdot)$) on their previous values at the past time interval plus Gaussian noise. The noise process is additive with zero-mean and covariance Q_k. Following the formulation of Azarbayejani and Pentland [28], it can be assumed that the motion of the camera through the scene is not known *a priori* and thus, $f_k(\cdot)$ can be set to identity. Therefore, the internal state varies only through some Gaussian random noise process. This can be seen as a 'random walk' type of internal state space [50]. In other words, the vector $\bar{\mathbf{x}}$ can be seen to vary randomly but smoothly with small deltas from its past values.

The camera model and the representation of the three-dimensional points are very important in a recursive SfM algorithm. Azarbayejani and Pentland [28] propose a parameterisation that is different from the one commonly used in the literature, discussed in Section 3.4.1. These new parameterisations have been shown to provide a dramatic increase in filter stability (damping and convergence) compared with recursive SfM implementations such as [59]. Therefore, Section 3.4 presents this reformulation of the generic state space parameterisation as it is proposed in [28].

3.4 Geometry and Representations

To develop a dynamic system with robustness and numerical stability for SfM geometry calculations, the SfM problem needs to be interpreted in an appropriate representation.

In [28] a stable and accurate recursive solution to this SfM problem has been presented. This representation has been popular with many authors. Model-Based coding systems in [55, 22, 66, 4] use the representation (or many components thereof) for robust head-pose estimation, and [67] have used the technique for an augmented reality type application. The representation has also recently been included in the industry leading 3D animation software product, "Maya" from Alias software [68], for the recovery of camera position, focal length, camera motion, perspective and 3D scene geometry in live video sequences. The primary aspects of the state space parameterisation based on the representations proposed in [28] are covered in Sections 3.4.1 to 3.4.3

3.4.1 Camera Model

Conventionally the camera model is represented by a perspective projection [69] as indicated in Figure 3.2(a). The model consists of an image plane and a 3D point called the center of projection (COP), where the COP is considered the origin of the *camera frame*. The distance between the camera frame and the COP is called the focal length f. The perspective projection equation, which represents the projection of a 3D point in the camera frame onto the image plane, is given by Equation (3.2),

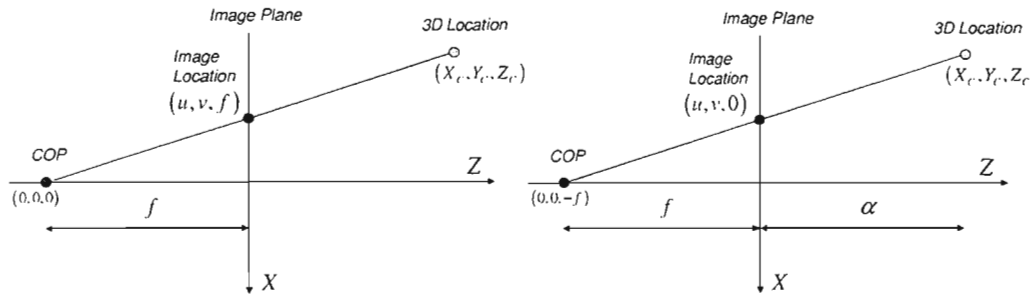
$$\begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} X_C \\ Y_C \end{pmatrix} \frac{f}{Z_C} \tag{3.2}$$

where (u, v) are the coordinates of the point in the 2D image plane, and (X_C, Y_C, Z_C) are the 3D coordinates of the point in the camera frame.

In [28], the standard camera model is modified by moving the origin to the image plane using,

$$\begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} X_C \\ Y_C \end{pmatrix} \frac{1}{1 + Z_C\beta} \tag{3.3}$$

where $\beta = 1/f$ is an alternative parameterisation of the focal length. This camera model



(a) Perspective Projection (b) Central Projection

Figure 3.2: Camera Projection Models.

has long been used in the photogrammetry community and has also been adopted by Szeliski and Kang [57] in their nonlinear least squares formulation of the structure from motion problem. This new representation can be seen in Figure 3.2(b).

As discussed in [28], the choice of an image centered coordinate system means that any change in focal length alters the imaging geometry, i.e. the relative distances between the projected points in the image plane changes. In contrast however, changing f in Equation (3.2) will only change the scale of the image coordinates, and the imaging geometry will essentially be coded into the depths Z_C. The use of β instead of f means that the imaging geometry can be altered independently of the depth value Z_C.

Another important benefit of the Central Projection model, Equation (3.3) and Figure 3.2(b), is that it does not suffer from the usual numerical ill-conditioning that occurs when focal length becomes large, i.e. the orthographic case. Under a standard perspective model, when the focal length $f \rightarrow \infty$, the perspective projection equations no longer suitably model image formation. In this case, an orthographic projection model is generally adopted, where all points project orthogonally onto the image plane, as in Equation (3.4).

$$\begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} X_C \\ Y_C \end{pmatrix} \tag{3.4}$$

With the central projection model however, it is possible to represent both cases. The model can span the range of a perspective projection, but also the special orthographic

case. This flexibility is not common in traditional computer vision approaches where perspective and orthographic projection are treated independently. Although this work does not explicitly require the modeling of the orthographic case, it is worthwhile to note the merits of this modification. More appropriate examples of where this flexibility becomes significant are during the reconstruction of scene geometry, such as demonstrated in [28], or the production of augmented reality, discussed in [67].

With any camera model, it is clear that knowledge of the focal length is required. This is not always known in computer vision scenarios, and is often a limiting factor. With the SfM formulation of Azarbayejani and Pentland [28], the inverse focal length parameter β is modeled as one of the unknowns in the system to be estimated, meaning that the inverse focal length β becomes the first state parameter of interest in this system. A note must be made here that other camera *intrinsic* parameters are assumed to be known *a priori* (calibrated), such as camera skew, aspect ratio and the position of the principal point.

3.4.2 Structure Model

In this system there are N feature points that are tracked over F frames in an image sequence. Traditionally, the 3D structure of an object may be thought of as the true Cartesian coordinates (X, Y, Z) of each of these 3D points which are then projected onto the image plane as 2D coordinates (u, v). Azarbayejani et. al [28] argue that this parameterization is not compact (number of parameters) or stable, as there are 3 unknowns per point being tracked, resulting in the concatenation of $3 \times N$ dimensions to the state vector. Another problem is that this representation in no way encodes the rigidity of the object being tracked.

A more compact representation of the 3D location is given by Equation (3.5).

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = \begin{pmatrix} (1+\alpha\beta)u^0 \\ (1+\alpha\beta)v^0 \\ \alpha \end{pmatrix} \tag{3.5}$$

The position of a feature point can now be represented by its image coordinates in the first frame (u^0, v^0), the inverse focal length parameter β and its unknown depth α, as shown in Figure 3.2(b). The point is restricted to lie on the line that emanates from the center of projection and goes through the point in the image plane. If the depth α is known, the Cartesian coordinates (X, Y, Z) in the first frame can be computed from u^0, v^0, β and α using Equation (3.5). It must be noted that this first frame condition assumes that the camera and object frame are aligned. Thus the 3D point (X, Y, Z) in the object frame is essentially in the same position in the camera frame (X_C, Y_C, Z_C), i.e. the relative rotation and translation between the two reference frames in the initial frame is zero.

Using this representation, the three-dimensional location of a point can be parameterized by the single parameter α, since the image coordinates (u^0, v^0) in the first frame are known. For N feature points plus the required camera parameter β, the state vector now has $N+1$ degrees of freedom. In a normal Cartesian representation, $3N+1$ states are needed. This is the case in the seminal work of Broida et al. [59], whose Kalman filter formulation is underconstrained, and good estimates for structure are needed to achieve convergence. In fact, it is stated in [59] that the method is only suitable for pose estimation after a batch procedure had been used for initialisation. With the structure parameterization used by Azarbayejani and Pentland [28] there are $N+1$ degrees of freedom compared to $2N$ measurements, such that when $N > 1$, the problem is overconstrained and thus uniquely solvable in each time step. This argument is further developed with the addition of six motion parameters for rotation and translation. With these state parameters included, the system becomes overconstrained when $2N$ measurements outnumber $N+7$ degrees of freedom, which occurs when $N > 7$.

3.4.3 Translation and Rotation

1) Translation Model: Translational motion, defined as the 3D location of the object reference frame relative to the current camera frame, is most often parameterized as (t_x, t_y, t_z) . The t_x and t_y components represent motions parallel to the image plane, and

the t_z component corresponds to the depth of the object along the optical axis. For this reason, the sensitivity of motion in the image plane to t_x and t_y motion will be similar to each other. The sensitivity to t_z motion on the other hand will vary, depending upon the focal length of the imaging geometry.

Use of standard video camera focal lengths results in very low sensitivity to t_z motion in the image plane compared to motion in the t_x and t_y directions [28]. This sensitivity difference increases as focal lengths increase, until the limiting orthographic case is reached where there is zero image plane sensitivity to t_z motion. For this reason, Azarbayejani and Pentland [28] propose the coupling of the inverse focal length parameter β with the translational component t_z, to form $t_z\beta$.

If one considers the sensitivity of the u (for example) image coordinate to $t_z\beta$ given by:

$$\frac{\partial u}{\partial t_z \beta} = \frac{-X_C}{(1 + Z_C\beta)^2} \tag{3.6}$$

it is observed that this does not approach zero in the limiting case of $f \rightarrow \infty\,(\beta \rightarrow 0)$, whereas in the normal parameterisation for t_z, $f \rightarrow \infty$ causes the sensitivity to approach zero, given by:

$$\frac{\partial u}{\partial t_z} = \frac{-X_C\beta}{(1 + Z_C\beta)^2} \tag{3.7}$$

Equation (3.6) implies that $t_z\beta$ remains observable from the measurements even with long focal lengths and translation is thus parameterised as:

$$\mathbf{T} = (t_x, t_y, t_z\beta) \tag{3.8}$$

The actual translation t_z can then be recovered post estimation by dividing out the inverse focal length β from $t_z\beta$.

*2) **Rotation Model:***

In describing a rotation, the two primary components are an axis and an angle. The rotation consists of moving the object through the given angle while keeping the axis fixed. The important characteristic of the axis is its direction, which is represented by a unit vector. Thus a rotation can be defined as a pair, $(\theta, \mathbf{u})$, in which θ is an angle and $\mathbf{u}$ is a unit vector. Thus in a three dimensional coordinate system, a rotation can be defined for each of the X, Y and Z axes.

The most widely known representation of such rotations is that of Euler angles [70]. Euler angles can be used to represent any arbitrary three dimensional rotation by specifying a sequence of rotations about the X, Y and Z axes by, for instance, the angles ϕ, θ, ψ respectively. The result of the rotation is dependant on the order in which the rotations occur, representing a non-commutative relationship. For example, a rotation about the X, Y then Z axis by the corresponding angles would result in a 3D rotation matrix $\mathbf{R}_{xyz}$ of the form

$$\mathbf{R}_{xyz} = r_x(\phi)\, r_y(\theta)\, r_z(\psi) \tag{3.9}$$

where

$$r_x(\phi) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(\phi) & \sin(\phi) \\ 0 & -\sin(\phi) & \cos(\phi) \end{pmatrix} \tag{3.10}$$

$$r_y(\theta) = \begin{pmatrix} \cos(\theta) & 0 & \sin(\theta) \\ 0 & 1 & 0 \\ -\sin(\theta) & 0 & \cos(\theta) \end{pmatrix} \tag{3.11}$$

$$r_z(\psi) = \begin{pmatrix} \cos(\psi) & \sin(\psi) & 0 \\ -\sin(\psi) & \cos(\psi) & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{3.12}$$

This would not be the same as a rotation of the form, $\mathbf{R}_{zyx} = r_z(\psi)\, r_y(\theta)\, r_x(\phi)$.

One of the most important drawbacks of the Euler parameterization is the existence of so-called *gimbal lock* singularities [71]. Because of the mutually independent characteristics of Euler angles, rotations break down when the second Euler angle becomes 90 degrees, resulting in a loss of one degree of freedom. Different formulations of the Euler angles in the rotation matrix do not remove this singularity [71].

An alternative representation which avoids any of these singular or special configurations is provided by *quaternions* [72]. Much work in the computer vision and graphics communities and more specifically SfM research such as [28, 57, 59], has adopted a unit quaternion to represent rotation.

A quaternion can be seen as a four-component number consisting of one scalar part and three orthogonal parts. Originally conceived by Hamilton, quaternions are more accurately perceived as an extension of complex numbers to $\mathbf{R}^4$ with three imaginary parts. Formally, a quaternion can be defined as

$$\mathbf{q} = q_0 + q_1 i + q_2 j + q_3 k \tag{3.13}$$

where each of the q_i is a real number, and i, j and k are orthogonal imaginary unit vectors.

From the above definition, the class of quaternions can be seen to include scalars, ordinary complex numbers, and three-element spatial vectors. A better understanding of quaternions and the relevant algebraic definitions and properties can be found in [72]. Two important definitions for rotation are quaternion multiplication and conjugation.

The conjugate of a quaternion is found in much the same way as a standard complex number. Hence

$$\mathbf{q}' = q_0 - q_1 i - q_2 j - q_3 k \tag{3.14}$$

In considering multiplication, a quaternion can also be represented as a scalar for the coefficient of 1 and a vector for the coefficients of the imaginary terms. This can be

written as:

$$\begin{aligned} \mathbf{q} &= \begin{bmatrix} q_0 & q_1 & q_2 & q_3 \end{bmatrix} = (s, \mathbf{v}) \\ s &= q_0 \\ \mathbf{v} &= \begin{bmatrix} q_1 & q_2 & q_3 \end{bmatrix} \end{aligned} \tag{3.15}$$

The product of two quaternions in terms of the $(s, \mathbf{v})$ representation using standard vector products is then

$$\begin{aligned} \mathbf{q}_1 &= (s_1, \mathbf{v}_1) \\ \mathbf{q}_2 &= (s_2, \mathbf{v}_2) \\ \mathbf{q}_1 * \mathbf{q}_2 &= (s_1 s_2 - \mathbf{v}_1 \cdot \mathbf{v}_2, s_1 \mathbf{v}_2 + s_2 \mathbf{v}_1 + \mathbf{v}_1 \times \mathbf{v}_2) \end{aligned} \tag{3.16}$$

The result of these two definitions is that a rotation in three dimensions can be computed about a unit vector, $\mathbf{u}$, by the angle θ, using a unit quaternion

$$\mathbf{q} = (s, \mathbf{v}) = \left(\cos \frac{\theta}{2}, \mathbf{u} \sin \frac{\theta}{2} \right) \tag{3.17}$$

This can be performed by representing a point $\mathbf{p}$ in space by the quaternion $\mathbf{P} = (0, \mathbf{p})$. The desired rotation about the point, using quaternion multiplication, is computed as

$$\begin{aligned} \mathbf{P} &= (0, \mathbf{p}) \\ \mathbf{P}_{rot} &= \mathbf{q} * \mathbf{P} * \mathbf{q}' \end{aligned} \tag{3.18}$$

This rotates the point $\mathbf{P} = (0, \mathbf{p})$ about the unit vector $\mathbf{u}$ by angle θ. For more details on quaternion rotation, the reader is referred to [72].

It now becomes possible to develop a three-dimensional rotation using a unit quaternion. For this derivation, the reader is referred to [72]. The rotation matrix, given in Equation (3.19), thus represents the relative rotation between the camera and object reference frames.

$$R(\mathbf{q}) = \begin{bmatrix} q_0^2 + q_1^2 - q_2^2 - q_3^2 & 2(q_1q_2 - q_0q_3) & 2(q_1q_3 + q_0q_2) \\ 2(q_1q_2 + q_0q_3) & q_0^2 - q_1^2 + q_2^2 - q_3^2 & 2(q_2q_3 - q_0q_1) \\ 2(q_1q_3 - q_0q_2) & 2(q_2q_3 + q_0q_1) & q_0^2 - q_1^2 - q_2^2 + q_3^2 \end{bmatrix} \tag{3.19}$$

The introduction of the quaternion based rotation representation presents a problem. its four elements only have three degrees of freedom due to the normality constraint required to keep $\|\mathbf{q}\| = 1$ [72]. Thus, all four components cannot be estimated independently; a nonlinear constrained minimisation is required to recover the quaternion directly. Since the method used in this thesis utilises an EKF, which relies on a linearisation at each step, the nonlinear normality constraint cannot be rigidly enforced within the EKF computational framework.

For this reason the authors of [28] suggest the use of a three-parameter incremental rotation representation in order to estimate inter-frame rotation at each step of the EKF. Incremental Euler angles centered about zero do not over parameterize rotation, are approximately independent and can therefore be used reliably in a system linearisation. The unit quaternion is then maintained outside of the Kalman filter framework.

Because quaternions rotate vectors by multiplication, each time step implements an update to the global rotation quaternion using Equation (3.20) in a quaternion multiplication with the current unit quaternion (composed of the incremental Euler angles $(\omega_x, \omega_y, \omega_z)$), as in Equation (3.21).

$$\begin{aligned} \delta\mathbf{q} &= \left(\sqrt{1-\varepsilon}, \omega_X/2, \omega_Y/2, \omega_Z/2\right) \\ \varepsilon &= \left(\omega_X^2 + \omega_Y^2 + \omega_Z^2\right)/4 \end{aligned} \tag{3.20}$$

$$\mathbf{q}_{k+1} = \mathbf{q}_k * \delta\mathbf{q} \tag{3.21}$$

3.4.4 Coordinate Frame Transformation

The translation and incremental rotation parameters add another 6 parameters to the system state vector. As mentioned in Section 3.4.2, the inclusion of these new parameters raises the degrees of freedom in the system to $N + 7$.

Combining the global rotation matrix $R(\mathbf{q})$ in Equation (3.19) with the translation vector $(t_x, t_y, t_z\beta)$, a coordinate frame transformation equation between the object and camera reference frame can be written as:

$$\begin{pmatrix} X_C \\ Y_C \\ \beta Z_C \end{pmatrix} = \begin{bmatrix} 1 & & \\ & 1 & \\ & & \beta \end{bmatrix} R(\mathbf{q}) \begin{pmatrix} X \\ Y \\ Z \end{pmatrix} + \begin{pmatrix} t_x \\ t_y \\ \beta t_z \end{pmatrix} \tag{3.22}$$

The combination of the above representations for the camera, structure, translation and rotation leads to an explicit representation of an object to image transformation, which becomes the measurement transfer function, $h_k(\cdot)$ of Equation (3.1), relating state space parameters to available measurements of the system. This will be further dealt with in Section 3.6.2.

3.4.5 Dealing with Scale

In SfM methods specific to single camera systems, a well known issue arises concerning the recovery of shape and translational motion [49] which are subject to arbitrary scaling (rotation is not subject to this scaling as it does not require knowledge of any true lengths in the scene). This scale factor is not directly recoverable either. Some methods attempt to deal with this unpredictable effect by fixing one of the lengths associated with the motion or structure. In [59] and [61], a component of the translational motion is fixed at some finite value. This however generally leads to further numerical ill-conditioning when the translation is zero (or very small). Also, because motion is generally dynamic, the scale is changing in every frame and this hinders rather than contributes to stability (damping and convergence). In [28], a novel and elegant method of dealing with the unknown scale

is tied into the EKF framework, the foundation of this SfM formulation. By fixing one of the shape parameters α, a uniform scale can be achieved for all the structure and motion parameters over the video sequence. This is achieved by setting the initial error covariance on an arbitrarily chosen structure point, say α_0, to zero, which fixes that parameter at its initial value. All the other structure parameters then automatically scale themselves to accommodate this constraint.

3.5 Kalman Filtering

Section 3.4 covered the necessary parameter representations required to describe camera to image transformations and relative camera to object motion. These parameters will constitute the set of internal states $\bar{\mathbf{x}}$ of the system that need to be recovered from observations of the system $\bar{\mathbf{z}}$ in an optimal estimation step of the EKF.

The Kalman filter was first introduced in the seminal paper by R.E Kalman [73], which describes a recursive solution to the discrete-data linear filtering problem. Since that time, technological demands and advances in computational complexity have led to much new research and many applications for the Kalman filter. The "filter" is actually a set of mathematical equations that provide an efficient computational solution of the least-squares method. The filter is very powerful, in that it supports estimations of past, present, and even future states, and is able to do this even when the precise nature of the modeled system is unknown.

In Section 3.5.1, a summary of the standard linear Kalman filtering equations is presented. For the unfamiliar reader, a derivation of these equations with meanings of relevant parameters can be found in Appendix A. Section 3.5.2 expands on the extended Kalman filter and Section 3.6 covers the derivation of a system specific measurement model as well as the measurement model linearisation required in each iteration of the extended Kalman filter.

3.5.1 Linear Kalman Filtering

Following references such as [50] or [51], the fundamental objective of the linear Kalman filter is to optimally estimate a quantity $\bar{\mathbf{x}}_k$ that changes dynamically according to the linear system:

$$\bar{\mathbf{x}}_k = F_{k-1}\bar{\mathbf{x}}_{k-1} + \bar{\mathbf{w}}_{k-1} \qquad \bar{\mathbf{w}}_k \sim N\left(\bar{0}, Q_k\right), \tag{3.23}$$

but can only be measured by:

$$\bar{\mathbf{z}}_k = H_k\bar{\mathbf{x}}_k + \bar{\mathbf{v}}_k \qquad \bar{\mathbf{v}}_k \sim N\left(\bar{0}, R_k\right), \tag{3.24}$$

This is the previously discussed topic of Section 3.3 as well as Appendix A. The explicit definitions of both H_k and F_k will be discussed in Section 3.6.2.

Some initial conditions of the system must be known. These include an initial estimate of the state vector and an initial error covariance, defined by:

$$\mathbf{x}_0^+ = E\left[\bar{\mathbf{x}}_0\right] \qquad P_0^+ = E\left[\left(\bar{\mathbf{x}}_0 - \hat{\mathbf{x}}_0^+\right)\left(\bar{\mathbf{x}}_0 - \hat{\mathbf{x}}_0^+\right)^T\right] \tag{3.25}$$

where $E[\cdot]$ is the expectation of the random variable [50].

An *a priori* estimate of the state, $\hat{\mathbf{x}}_k^-$, and error covariance matrix, P_k^-, can then be extrapolated to the next time step k by:

$$\hat{\mathbf{x}}_k^- = F_{k-1}\hat{\mathbf{x}}_{k-1}^+ \tag{3.26}$$

$$P_k^- = F_{k-1}P_{k-1}^+F_{k-1}^T + Q_{k-1} \tag{3.27}$$

The Kalman gain K_k and the error covariance P_k^+ are updated as:

$$K_k = P_k^- H_k^T \left[H_k P_k^- H_k^T + R_k \right]^{-1} \tag{3.28}$$

$$P_k^+ = (I - K_k H_k) P_k^- \tag{3.29}$$

Finally the measurement $\bar{\mathbf{z}}_k$ can be used to update the *a posteriori* estimate of the state vector using:

$$\hat{\mathbf{x}}_k^+ = \hat{\mathbf{x}}_k^- + K_k \left(\bar{\mathbf{z}}_k - H_k \hat{\mathbf{x}}_k^- \right) \tag{3.30}$$

where the difference $(\bar{\mathbf{z}}_k - H_k \hat{\mathbf{x}}_k^-)$ is called the *innovation* (or *residual*). The innovation reflects the discrepency between the predicted measurement $H_k \hat{\mathbf{x}}_k^-$ and the actual measurement $\bar{\mathbf{z}}_k$. Thus, an innovation of zero means that the two are in complete agreement.

The algorithm then continues by iterating Equation (3.26) $\rightarrow$ (3.30).

3.5.2 Extended Kalman Filtering

It is not possible to express all problems with the linear dynamics in Equation (3.23) and (3.24). This section extends the optimal estimation techniques of the linear Kalman filter to the more general case of nonlinear systems. Again, following Gelb [50], such a system can be described by

$$\begin{aligned} \bar{\mathbf{x}}_k &= f_{k-1} \left(\bar{\mathbf{x}}_{k-1} \right) + \bar{\mathbf{w}}_{k-1}, \qquad \bar{\mathbf{w}}_k \sim N \left(\bar{0}, Q_k \right), \\ \bar{\mathbf{z}}_k &= h_k \left(\bar{\mathbf{x}}_k \right) + \bar{\mathbf{v}}_k, \qquad \bar{\mathbf{v}}_k \sim N \left(\bar{0}, R_k \right), \end{aligned} \tag{3.31}$$

Here $f_k(\cdot)$ and $h_k(\cdot)$ can both be non-linear, time-varying functions of the state $\bar{\mathbf{x}}_k$. The initial conditions, as for the linear system, are given as:

$$\mathbf{x}_0^+ = E \left[\bar{\mathbf{x}}_0 \right] \qquad P_0^+ = E \left[\left(\bar{\mathbf{x}}_0 - \hat{\mathbf{x}}_0^+ \right) \left(\bar{\mathbf{x}}_0 - \hat{\mathbf{x}}_0^+ \right)^T \right] \tag{3.32}$$

The extrapolation of the *a priori* state estimate is performed using the non-linear equation

$$\hat{\mathbf{x}}_k^- = f_{k-1}\left(\hat{\mathbf{x}}_{k-1}^+\right) \tag{3.33}$$

Propagation of the *a priori* error covariance matrix however, must be approximated using a truncated Taylor series expansion such that,

$$P_k^- = F_{k-1}\left(\mathbf{x}_{k-1}^+\right) P_{k-1}^+ F_{k-1}\left(\mathbf{x}_{k-1}^+\right)^T + Q_{k-1} \tag{3.34}$$

where

$$F_{k-1}\left(\mathbf{x}_{k-1}^+\right) = \left.\frac{\partial f_{k-1}\left(\bar{\mathbf{x}}\right)}{\partial \bar{\mathbf{x}}}\right|_{\bar{\mathbf{x}}=\mathbf{x}_{k-1}^+} \tag{3.35}$$

is the linear term of the Taylor expansion of $f_{k-1}\left(\bar{\mathbf{x}}_{k-1}\right)$.

The update equations which account for the measurement data are acquired in a similar fashion. The measurement transfer function $h_k\left(\bar{\mathbf{x}}\right)$ is linearised about the current best *a priori* estimate of the state $\hat{\mathbf{x}}_k^-$ from Equation (3.33) where

$$H_k\left(\mathbf{x}_k^-\right) = \left.\frac{\partial h_k\left(\bar{\mathbf{x}}\right)}{\partial \bar{\mathbf{x}}}\right|_{\bar{\mathbf{x}}=\mathbf{x}_k^-} \tag{3.36}$$

The update of the Kalman gain K_k and the error covariance P_k^+ can then be written as

$$K_k = P_k^- H_k\left(\hat{\mathbf{x}}_k^-\right)^T \left[H_k\left(\hat{\mathbf{x}}_k^-\right) P_k^- H_k\left(\hat{\mathbf{x}}_k^-\right)^T + R_k\right]^{-1} \tag{3.37}$$

$$P_k^+ = \left(I - K_k H_k\left(\hat{\mathbf{x}}_k^-\right)\right) P_k^- \tag{3.38}$$

Finally, the non-linear measurement function $h_k\left(\bar{\mathbf{x}}\right)$ is used together with the measurement $\bar{\mathbf{z}}_k$ to update the state vector,

$$\hat{\mathbf{x}}_k^+ = \hat{\mathbf{x}}_k^- + K_k\left(\bar{\mathbf{z}}_k - h_k\left(\hat{\mathbf{x}}_k^-\right)\right) \tag{3.39}$$

The EKF is then characterised by the iteration of the time and measurement update equations. Figure 3.3 gives a graphical representation of these stages.

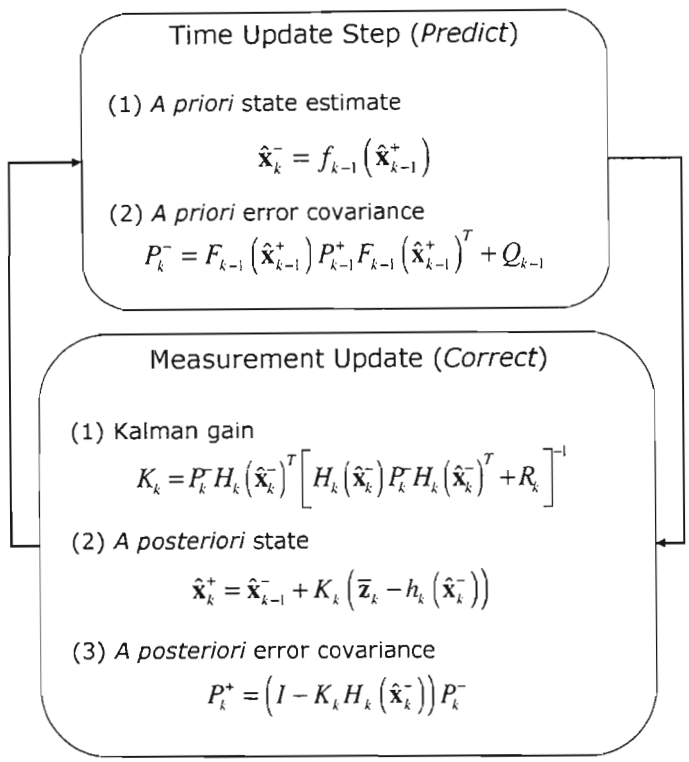


Figure 3.3: Extended Kalman filter equations for state prediction, and state correction via weighted measurement.

3.5.3 Optimality of the EKF

The linear Kalman filter provides an optimal solution to the linear problem of Equations (3.23) and (3.24). The extended Kalman filter however, since it is based on approximations, cannot claim optimality [50]. In fact, there is no guarantee that Extended Kalman filtering will produce an estimate that is close to the truly optimal solution of the non-linear system in Equation (3.31). For example, the Gaussian characteristics of the

noise are not preserved when filtered through non-linear filters. Fortunately, the method has been shown to work well in many practical applications, as reported by Gelb et al. [50].

3.6 EKF for SfM

Thus far the principles and process of parameter estimation through Kalman filtering have been discussed. A number of state parameters representing the structure and motion dynamics in the system have been identified along with relevant coordinate transformation equations. In Section 3.6.1 the parameterisation of the state vector, $\bar{\mathbf{x}}$, is discussed based on the identified system parameters. Section 3.6.2 covers the specific process and measurement models. Because the measurement model of the system is nonlinear, Section 3.6.3 deals with the linearisation step that is required in the extended Kalman filter algorithm.

3.6.1 State Vector Parameterisation

The state vector of this SfM formulation, $\bar{\mathbf{x}}$, is composed of a set of 6 motion parameters (3 translations, 3 rotations), a camera parameter defined by the inverse focal length, and N structure parameters, giving a total of $N+7$ state parameters shown in Equation (3.40).

$$\bar{\mathbf{x}} = (t_X, t_Y, t_Z\beta, \omega_X, \omega_Y, \omega_Z, \beta, \alpha_1, ..., \alpha_N) \tag{3.40}$$

These are the parameters that are to be recovered using the EKF and observation measurements of the system. As long as the number of measurements $2N$ outnumber the number of states $N + 7$, the system is over constrained. The acquisition of these measurements is the topic of Chapter 4.

3.6.2 SfM Dynamics Solution

Although the fundamentals of the solution proposed thus far are based on the work of Azarbayejani et al. [28], no explicit solution for the Kalman filter measurement model

$h_k(\bar{\mathbf{x}})$ or the required linearisation thereof is presented in [28]. The focus of this section therefore, will be on the necessary equation substitutions and model linearisations required for the EKF. Mention is also made on the choice of process model, $f_k(\bar{\mathbf{x}})$, which *is* a topic of discussion in [28] as well as Section 3.3 of this thesis.

For convenience the non-linear dynamics model of Equation (3.31) is repeated here:

$$\begin{aligned} \bar{\mathbf{x}}_k &= f_{k-1}\left(\bar{\mathbf{x}}_{k-1}\right) + \bar{\mathbf{w}}_{k-1}, \qquad \bar{\mathbf{w}}_k \sim N\left(\bar{0}, Q_k\right), \\ \bar{\mathbf{z}}_k &= h_k\left(\bar{\mathbf{x}}_k\right) + \bar{\mathbf{v}}_k, \qquad \bar{\mathbf{v}}_k \sim N\left(\bar{0}, R_k\right), \end{aligned} \tag{3.41}$$

Azarbayejani et al.[28] assume that one cannot know the motion of the camera through the scene *a priori*. As a consequence the system process model $f_k(\bar{\mathbf{x}})$ is set to an identity transform:

$$f_k(\bar{\mathbf{x}}) = I\bar{\mathbf{x}} \tag{3.42}$$

meaning that the internal state only varies due to the noise associated with the process $\bar{\mathbf{w}}_k$. This is known as a *random walk* type process [50]. As a result, the primary estimation step in this system is dictated by the measurement update stage of the Kalman filter.

The measurement model $h_k(\bar{\mathbf{x}})$ is a non-linear transfer function relating the state parameters to image plane coordinates. It is composed of a combination of the non-linear camera model, Equation (3.3), the structure representation in Equation (3.5) and the reference frame transformation in Equation (3.22). This function provides predictions of image coordinates that are used in the measurement update step of Equation (3.39).

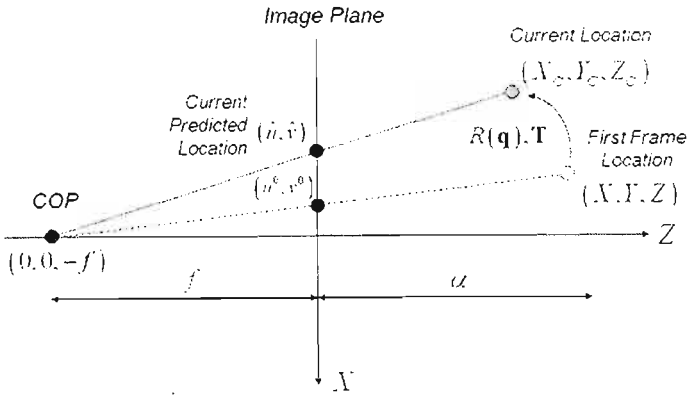


Figure 3.4: Measurement model prediction process.

The prediction process is shown in Figure 3.4. The purpose of the prediction is to compute estimates of image coordinates:

$$h_k(\bar{\mathbf{x}}) = \bar{\mathbf{y}} = (\hat{u}, \hat{v})^T \tag{3.43}$$

for each of the structure points, α, using the current *a priori* estimate of system state. The Kalman filter then uses these estimates to update the *a posteriori* state estimate. This is done by weighting the difference between the image coordinate estimates $\bar{\mathbf{y}}$ and the available measurements $\bar{\mathbf{z}}$, using Equation 3.39.

The estimated image coordinates are calculated by first finding the cartesian coordinates, (X, Y, Z), of the structure points in the first frame of the sequence. This involves substitution of the image coordinates in the initial frame (u^0, v^0) and the depth estimate of each point, α, into Equation (3.5). These coordinates are then rotated and translated using Equation (3.22) to (X_C, Y_C, Z_C) in the camera frame. Finally, they are projected onto the image plane using Equation (3.3) to provide an estimate, $(\hat{u}, \hat{v})$, of the image location of the feature point. The full set of point projections are then compared against the measurement vector $\bar{\mathbf{z}}$.

To perform the linearisation of the measurement model, an explicit equation for $h_k(\bar{\mathbf{x}})$

based on the substitutions mentioned above is required. The substitution is started by assuming, temporarily, that the 3×3 rotation matrix $R(\mathbf{q})$ can be written as:

$$\begin{aligned} R(\mathbf{q}) &= \begin{bmatrix} q_0^2 + q_1^2 - q_2^2 - q_3^2 & 2(q_1 q_2 - q_0 q_3) & 2(q_1 q_3 + q_0 q_2) \\ 2(q_1 q_2 + q_0 q_3) & q_0^2 - q_1^2 + q_2^2 - q_3^2 & 2(q_2 q_3 - q_0 q_1) \\ 2(q_1 q_3 - q_0 q_2) & 2(q_2 q_3 + q_0 q_1) & q_0^2 - q_1^2 - q_2^2 + q_3^2 \end{bmatrix} \\ &= \begin{bmatrix} r_{xx} & r_{xy} & r_{xz} \\ r_{yx} & r_{yy} & r_{yz} \\ r_{zx} & r_{zy} & r_{zz} \end{bmatrix} \end{aligned} \tag{3.44}$$

Because $h_k(\bar{\mathbf{x}})$ defines two measurement estimates $(\hat{u}_i, \hat{v}_i)$ for each of the N structural points, one can define $h_{k_i}(\bar{\mathbf{x}})$ at time step k for each of the i structure points as

$$\begin{aligned} h_{k_i}(\bar{\mathbf{x}}) &= \begin{pmatrix} \hat{u}_i \\ \hat{v}_i \end{pmatrix} \\ &= \begin{cases} \frac{t_x + r_{xx}(1+\alpha_i\beta)u^0 + r_{xy}(1+\alpha_i\beta)v^0 + r_{xz}\alpha_i}{1 + t_z\beta + r_{zx}(1+\alpha_i\beta)u^0\beta + r_{zy}(1+\alpha_i\beta)v^0\beta + r_{zz}\alpha_i\beta} \\ \frac{t_y + r_{yx}(1+\alpha_i\beta)u^0 + r_{yy}(1+\alpha_i\beta)v^0 + r_{yz}\alpha_i}{1 + t_z\beta + r_{zx}(1+\alpha_i\beta)u^0\beta + r_{zy}(1+\alpha_i\beta)v^0\beta + r_{zz}\alpha_i\beta} \end{cases} \\ &= \begin{cases} \frac{A}{1+C} \\ \frac{B}{1+C} \end{cases} \end{aligned} \tag{3.45}$$

such that $h_k(\bar{\mathbf{x}})$ at time step k is a $2N$ vector of projection estimates for each of the N structure points, i.e.:

$$h_k(\bar{\mathbf{x}}) = (h_{k_1}, h_{k_2}, ..., h_{k_N})^T \tag{3.46}$$

3.6.3 Calculating the Jacobian

In updating the Kalman gain of Equation (3.37) and the *a posteriori* error covariance of Equation (3.38), a linearised version of $h(\bar{\mathbf{x}})$ as a Taylor expansion about the current best *a priori* estimate of the state, is required.

Repeating Equation (3.36):

$$H_k\left(\mathbf{x}_k^-\right) = \left.\frac{\partial h_k\left(\bar{\mathbf{x}}\right)}{\partial \bar{\mathbf{x}}}\right|_{\bar{\mathbf{x}}=\mathbf{x}_k^-} \tag{3.47}$$

Equation (3.47) represents a $2N \times M$ *Jacobian* matrix of partial derivatives of $h\left(\bar{\mathbf{x}}\right)$ for each of the internal states, where $M = 7 + N$, the total number of states in $\bar{\mathbf{x}}$. This can hence be expanded as:

$$H_k\left(\mathbf{x}_k^-\right) = \begin{bmatrix} \frac{\partial h_{k_1}}{\partial x_1} & \cdots & \frac{\partial h_{k_1}}{\partial x_M} \\ \vdots & \ddots & \vdots \\ \frac{\partial h_{k_N}}{\partial x_1} & \cdots & \frac{\partial h_{k_N}}{\partial x_M} \end{bmatrix} \tag{3.48}$$

The partial derivatives $\frac{\partial h}{\partial x}$ over the N structure points for each of the M states, are given by:

$$\frac{\partial h_i}{\partial t_x} = \begin{cases} \frac{1}{1+C} \\ 0 \end{cases} \tag{3.49}$$

$$\frac{\partial h_i}{\partial t_y} = \begin{cases} 0 \\ \frac{1}{1+C} \end{cases} \tag{3.50}$$

$$\frac{\partial h_i}{\partial \beta t_z} = \begin{cases} \frac{-A}{(1+C)^2} \\ \frac{-B}{(1+C)^2} \end{cases} \tag{3.51}$$

$$\frac{\partial h_i}{\partial \beta} = \begin{cases} \frac{\left(r_{xx}u^0+r_{xy}v^0\right)\alpha_i}{(1+C)} - \frac{A\ \frac{t_z\beta}{\beta}+(1+\alpha_i\beta)\left(r_{zx}u^0+r_{zy}v^0\right)+\left(r_{zx}u^0+r_{zy}v^0\right)\beta\alpha_i+r_{zz}\alpha}{(1+C)^2} \\ \frac{\left(r_{yx}u^0+r_{yy}v^0\right)\alpha_i}{(1+C)} - \frac{B\ \frac{t_z\beta}{\beta}+(1+\alpha_i\beta)\left(r_{zx}u^0+r_{zy}v^0\right)+\left(r_{zx}u^0+r_{zy}v^0\right)\beta\alpha_i+r_{zz}\alpha}{(1+C)^2} \end{cases} \tag{3.52}$$

$$\frac{\partial h_i}{\partial \alpha_i} = \begin{cases} \frac{\left(r_{xx}\beta u^0+r_{xy}\beta v^0+r_{xz}\right)}{(1+C)} - \frac{A\left(r_{zx}u^0\beta^2+r_{zy}v^0\beta^2+r_{zz}\beta\right)}{(1+C)^2} \\ \frac{\left(r_{yx}\beta u^0+r_{yy}\beta v^0+r_{yz}\right)}{(1+C)} - \frac{B\left(r_{zx}u^0\beta^2+r_{zy}v^0\beta^2+r_{zz}\beta\right)}{(1+C)^2} \end{cases} \tag{3.53}$$

$$\frac{\partial h_i}{\partial \omega_x} = \begin{cases} \frac{(1+\alpha\beta)v^0 r_{xz} - \alpha r_{xy}}{1+C} - \frac{A\beta\left((1+\alpha\beta)v^0 r_{zz} - \alpha r_{zy}\right)}{(1+C)^2} \\ \frac{(1+\alpha\beta)v^0 r_{yz} - \alpha r_{yy}}{1+C} - \frac{B\beta\left((1+\alpha\beta)v^0 r_{zz} - \alpha r_{zy}\right)}{(1+C)^2} \end{cases} \tag{3.54}$$

$$\frac{\partial h_i}{\partial \omega_y} = \begin{cases} \frac{-(1+\alpha\beta)u^0 r_{xz} + \alpha r_{xx}}{1+C} - \frac{A\beta\left(-(1+\alpha\beta)u^0 r_{zz} + \alpha r_{zx}\right)}{(1+C)^2} \\ \frac{-(1+\alpha\beta)u^0 r_{yz} + \alpha r_{yx}}{1+C} - \frac{B\beta\left((1+\alpha\beta)u^0 r_{zz} - \alpha r_{zx}\right)}{(1+C)^2} \end{cases} \tag{3.55}$$

$$\frac{\partial h_i}{\partial \omega_z} = \begin{cases} \frac{(1+\alpha\beta)u^0 r_{xy} - (1+\alpha\beta)v^0 r_{xx}}{1+C} - \frac{A\beta\left((1+\alpha\beta)u^0 r_{zy} - (1+\alpha\beta)v^0 r_{zx}\right)}{(1+C)^2} \\ \frac{(1+\alpha\beta)u^0 r_{yy} - (1+\alpha\beta)v^0 r_{yx}}{1+C} - \frac{B\beta\left((1+\alpha\beta)u^0 r_{zy} - (1+\alpha\beta)v^0 r_{zx}\right)}{(1+C)^2} \end{cases} \tag{3.56}$$

These are the terms of the Jacobian matrix $H_k\left(\mathbf{x}_k^-\right)$, which represents the linear term of a Taylor expansion of the measurement transfer function $h\left(\bar{\mathbf{x}}\right)$ about the *a priori* state estimate.

All the models necessary to implement an EKF for SfM estimation have now been covered. With a set of measurement data in the form of 2D observations of a rigidly moving object, it is now possible to recover the state parameters of the vector in Equation (3.40).

3.7 EKF Synthetic Experiments

An important means of evaluating the performance of a Structure from Motion algorithm is via synthetic data testing, such as that in [59, 57, 28]. These experiments are justified at this stage in order to establish proof of principle before proceeding to new tasks. With synthetic experiments, it is possible to control every aspect of the system and compare performance under varying input and control parameters. In the experimental results discussed in this section, noise levels, initial state estimates and parameter variances constitute the variable constraints.

In Section 3.7.1, the generation of a set of measurement data is discussed which will be fed into the SfM algorithm as a set of tracked pixel coordinates. Section 3.7.2 presents motion estimation results that are typical of the quality of results one could expect when

there is little noise associated with the measurements. The alternative, when noise levels significantly corrupt measurements, is discussed in Section 3.7.3. The recovery of object structure and the camera internal geometry is presented in Section 3.7.4, with conclusions on the experiments and the SfM technique discussed in Section 3.8.

3.7.1 Generating the Data Set

The principle idea of synthetic experimentation is to create a set of temporally varying pixel coordinates which correspond to the projections of some salient features on a moving rigid object over time. These pixel coordinates are then fed as measurements to the SfM framework, from which the algorithm must estimate the original focal length, motion and structure of the salient points on the target object, as depicted in Figure 3.5.

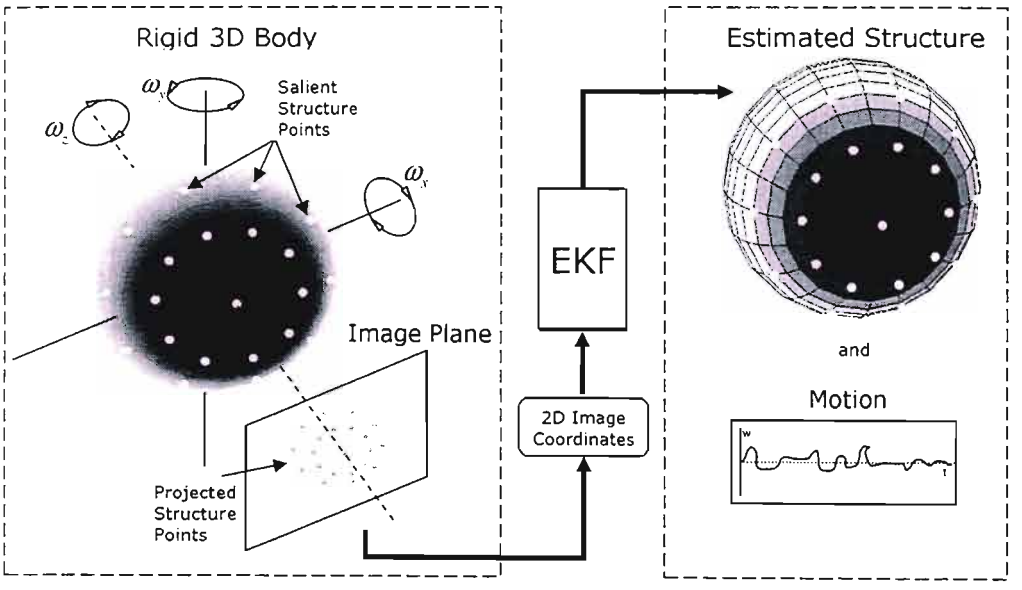


Figure 3.5: Basis of SfM Experiments. Structure and motion recovery from pixel measurements.

To project features onto the image plane, a virtual camera is needed. The camera is chosen as a perspective projection with a focal length of $f = 5mm$. This focal length creates a camera representative of the real camera used in Chapter 6. The true structure consists of 21 points on the surface of a frontally facing hemisphere, as shown in Figure 3.5. The

simulated motion trajectory is taken from the estimated motion in a real video sequence of a persons head. Typically, synthetic testing on SfM algorithms utilise uniform, continuous or periodic type motion [59, 28, 57]. In order to demonstrate the accuracy of the algorithm to the recovery of realistic motion, the experiments performed here deviate from this norm, without loosing any of the significance typical of results obtained using standard motion representations. This is seen from the quality of results presented in Sections 3.7.2 to 3.7.4.

With the ground truth motion, the simulated measurement set is constructed using a three step process of:

1. Computing the (temporally changing) spatial three-dimensional coordinates of the feature points for a specified number of frames using the known ground truth set of motion parameters (rotations and translations), under the assumption that the object is starting from rest in a frontal position.
2. Calculating the image locations of the feature points via a perspective projection of the points onto the image plane.
3. Corrupting the captured image plane data with zero-mean, Gaussian distributed noise.

The generated data set depicts measurements at the image plane in the same units of length as the camera and object frame, i.e. in millimeters. When using real video sequences, pixel data must be converted from coordinates on the CCD to image plane data using the known intrinsic parameters of the camera (principle point and relative pixel size).

Zero-mean uniformly distributed pixel noise over the interval $[-n, n]$ is added to the coordinates of each measurement. In this work, two sets of experiments are performed for varying noise. In Section 3.7.2, $n = 1$ and in Section 3.7.3, $n = 6$. Because the image plane is based on a canvas of $(512, 512)$ pixels and the projected object constitutes one third the size of the image, these noise levels correspond to 1.5% and 7.5% of the object size.

Except for the type of motion chosen, this method of obtaining a set of measurement data

is characteristic of synthetic experimentation on SfM algorithms, such as that performed in [28, 63, 59]. The results are intended to confirm the ability of the EKF to converge onto accurate results for inverse focal length, rotation, translation and structure, while remaining stable once convergence has been achieved. The quality of results and the speed of convergence are very much dependant on the accuracy of the measurements (in terms of noise corruption) and the initial conditions the filter starts with [28]. The initial conditions and convergence are discussed for each of the recovered parameters in the subsections below. The initial conditions are similar to those used in the sequences with real video data presented in Chapter 6.

3.7.2 Motion Estimates - Low Noise

Motion parameters are of primary concern in this work, as these are the rigid animation parameters which dictate the pose of the synthesised model in a MBVC system. Under the assumption, which permeates throughout this thesis, that the object is initialised in a frontal position, some intuitive initial conditions on the motion parameters can be assumed.

At time t_0, the object is constrained to start from rest, or conversely, the camera is starting from rest, such that the initial motion on the camera is zero. Under this assertion, a low initial error covariance must be chosen for each of the 6 motion parameters, as a good initial estimate of object motion is available. Setting this parameter to zero however, would cause the filter to assume that all the motion parameters start at their optimal estimate. Consequently, they would not change with the arrival of new measurements, as seen from the contribution of the error covariance to the calculation of the Kalman gain in Equation 3.37. For this reason, a sufficiently low but unharmful initial error covariance of $P_0 = 1$ is chosen for the motion parameters $(t_x, t_y, t_z\beta, w_x, w_y, w_z)$. This value will change as new measurements of the system arrive. The unit quaternion $\mathbf{q}$ is maintained outside of the EKF framework, and is initialised to $\mathbf{q} = (1, 0, 0, 0)$ because of the zero initial motion.

The choice of process noise covariance for the motion parameters is again fairly intuitive. Previously discussed in Section 3.6.2, the process transfer function is defined by an identity

transform, giving rise to a random walk type process which varies only by the noise associated with it. The state change must remain smooth, so small variance values are chosen that will constrain the variation at each step. Values are chosen as an approximation of the amount of motion the object being viewed might vary by in each frame. In a typical video sequence of head motion captured at 30 fps, the head may undergo up to 200 mm of translational motion along each cartesian axis per second. This is nearly $\Delta t = 7\ mm$ of motion per frame. The system covariance for each translational parameter is consequently set to $Q_t = 50\ mm^2$. A similar argument is used for rotation. A rotation of up to 90° per second is assumed. Using the same appraoch, this translates to a system covariance of $Q_r = 0.0025\ rad^2$.

Figures for motion estimates and corresponding errors can be found in Section B.1 of Appendix B. These include plots of the estimated quaternion components, Euler angles (converted from the rotation quaternion) and translation. Each graph plots state variables as the ordinate versus frame number as the abscissa.

With low noise corruption on the synthetic measurements, the motion estimates exhibit fast and accurate convergence. Fast convergence is considered to be coherent recovery against the ground truth in under 100 frames. Such convergence rates have been achieved in [28]. This convergence period is more easily recognised in the camera and structure estimates of Section 3.7.4.

Over the convergence period of approximately 100 frames, estimation errors are fairly large, as the system has not yet converged onto a good structure etimate. Once structural convergence has been achieved, errors decay to some low mean and motion estimates remain consistent and stable against the ground truth. Table 3.1 provides some quantative analysis of these errors for each of the motion parameters. In the table, m represents the error mean and σ the root mean square (RMS) error over the sequence. Over each of the three degrees of freedom associated with each type of motion, these values constitute an average RMS error of $\delta t \sim 3\ mm$ for translation and $\delta\omega \sim 0.3°$ for rotation.

The accurate estimation of depth motion t_z is a noteable feature of the algorithm. The errors for depth are of the same order as those for t_x and t_y. Considering that this is a

recursive estimation algorithm operating on single views, this accuracy of depth recovery becomes an asset to the system that is normally difficult to achieve [49].

Table 3.1: Rotation and translation estimation errors - Low Noise.

parameter	m	σ	unit
ω_x	-0.0035	0.0054	rad
ω_y	-0.0030	0.0065	rad
ω_y	0.0013	0.0026	rad
t_x	-1.555	3.133	mm
t_y	1.005	3.242	mm
t_z	2.370	2.262	mm

3.7.3 Motion Estimates - High Noise

The accuracy of convergence and tracking stability (convergence and damping) is very much dependant on the reliability of the measurement data to the estimation module. Increasing noise levels in the measurements being fed to the SfM framework will inevitably result in a degradation of the quality of outputs. The results in this section are presented as proof of the algorithm being able to cope with high noise cases. The generation of the data set follows that of the measurements used in Section 3.7.2, but the final measurements are corrupted with pixel noise over the interval $[-6, 6]$. This constitutes up to 7.5% of the total object size, as discussed in Section 3.7.1. As with synthetic experimentation performed in [28], these are significantly high noise levels, yet the SfM algorithm still manages to achieve reasonable motion estimates, as shown by Table 3.2.

Figures for motion estimates and corresponding errors can be found in Section B.2 of Appendix B. These plots, along with the error statistics in Table 3.2, demonstrate the degradation in performance of the algorithm with high levels of noise as compared with those in Section 3.7.2. However, the errors are still low enough to deem the pose estimates suitably reflective of the ground truth motion occuring. The average RMS error over

translation is now $\delta t \sim 28\ mm$ and for rotation is $\delta\omega \sim 3°$.

Table 3.2: Rotation and translation estimation errors - High Noise.

parameter	m	σ	unit
ω_x	0.0102	0.0778	rad
ω_y	-0.0316	0.0562	rad
ω_z	0.0085	0.0176	rad
t_x	-15.91	28.96	mm
t_y	-4.300	35.38	mm
t_z	-14.56	19.01	mm

The degradation of measurement data is a significant issue for this SfM algorithm. Without proper modeling of the variance on the measurement data, the extended Kalman filter finds it increasingly difficult to recover the necessary state parameters. Azarbayejani et al. [28] provide a more extensive discussion on the sensitivity and rate of convergence of this formulation to varying degrees of measurement noise. For the purposes of this thesis, these results are presented as a boundary within which the system must remain. The results show that under both conditions (low and high noise), the algorithm performs well as a motion estimator. In Section 5.4, a control mechanism that limits the possibility of using bad measurements is implemented, such that most measurements are of the low noise quality. However, the results in this section validate the ability of the system to remain robust and stable in the case of poor measurements being fed to the framework.

3.7.4 Structure and Camera Parameters

The aim of structure estimation is to achieve accurate estimates of the three-dimensional locations of salient feature points on the surface of the synthetic hemisphere. Because of the structure parameterisation only a single parameter, α, for each of the points is estimated. The three-dimensional coordinates can then be extracted using Equation (3.5).

For the synthetic experiments, the initial condition on the structure is that all points lie in a plane parallel to the image. For simplicity, the common plane is placed at the depth of the point on the hemisphere closest to the camera. This structure point is denoted α_0. With reference to the discussion on scale in Section 3.4.5, the error covariance, $P_{0_{\alpha_0}}$ is set to zero, fixing that parameter at the initial depth estimate. The Kalman filter then scales all other structure parameters around this value.

The initial error covariance on the other structure parameters, $\alpha_1, ..., \alpha_N$, is based on the maximum error between the known structure and the initial planar structure condition. Shown in Figure 3.6, the maximum depth difference occurs for the points on the base of the hemisphere. These points are essentially 1 radius away from the plane on the ground truth hemispherere. The radius is 100 mm, such that the error covariance $P_{0_\alpha} \sim 10000\ mm^2$.

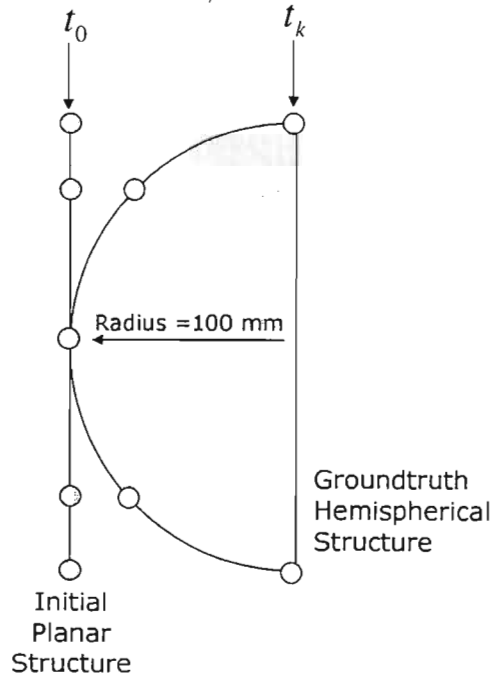


Figure 3.6: Initial condition and groundtruth structure points.

The camera parameter of interest is the inverse focal length β. Other camera parameters (intrinsic) are assumed to be calibrated in a real system based on this SfM framework. For the synthetic experiments, an initial camera focal length estimate of $f = 2.5\ mm$ is used. Consequently, the camera parameter β is initialised to $\beta = 0.4$. This is sufficiently

different from the known value, $f = 5\ mm$ (or $\beta = 0.2$), to gauge the speed and ability of the filter to converge and recover an accurate estimate. The initial error covariance P_{0_β} is chosen based on the difference between the ground truth and initial estimate. This equates to $P_{0_\beta} \sim 0.04\ mm^2$.

Process noise covariance for structure and the camera parameter are chosen in a similar fashion to those for motion. Accurate motion recovery cannot occur before structure and the camera parameter have converged, thus it is crucial that convergence is not hindered by setting the process noise too high. For this reason, the inverse focal length is limited to changing by $\Delta\beta = 0.0002\ mm^{-1}$ (1/1000 of the ground truth value of $\beta = 0.2$) and structure by $\Delta\alpha = 0.1\ mm$ (1/1000 of the radius of the hemisphere) per frame. These translate to $Q_\beta \sim 4E{-}8\ mm^{-2}$ for the camera parameter and $Q_\alpha \sim 0.01\ mm^2$.

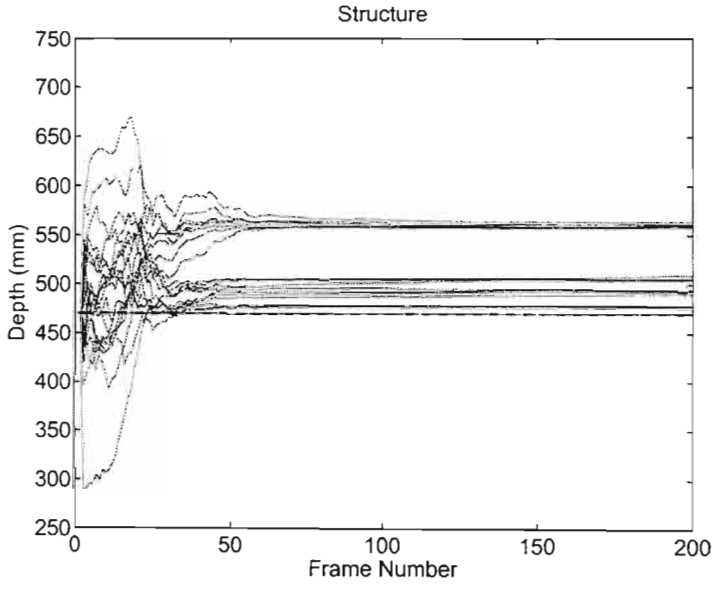


Figure 3.7: Convergence of the 21 structural points.

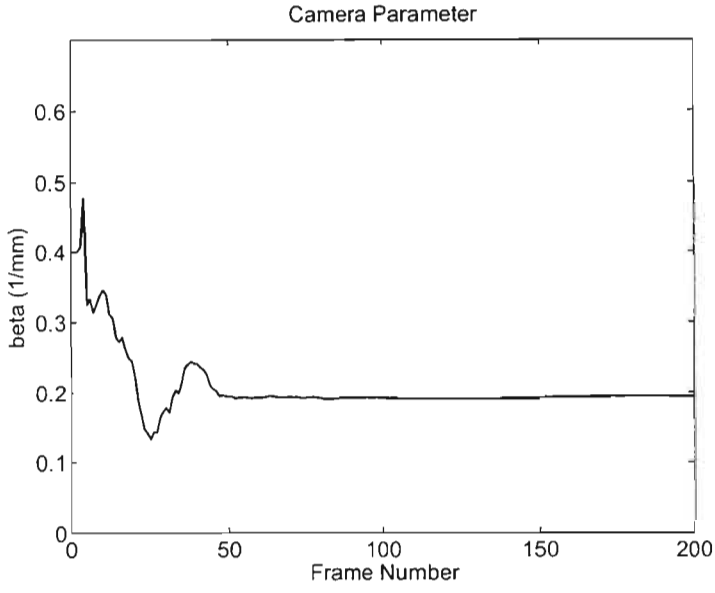
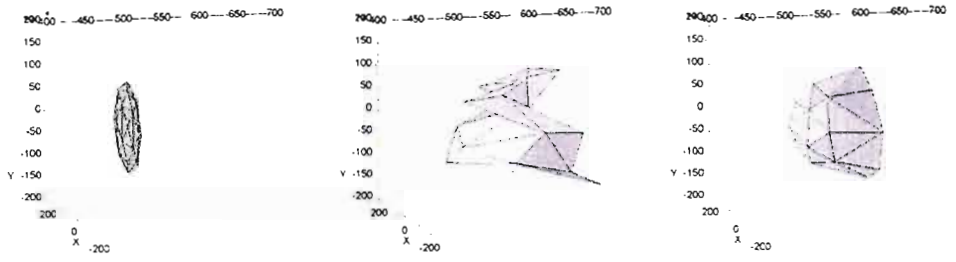


Figure 3.8: Convergence of the camera parameter β.

Figures 3.7 and 3.8 show results for structure and camera parameter estimation using the initialisation parameters already discussed in this section. Both parameters demonstrate fast and accurate convergence onto the ground truth in under 100 frames. As with the motion estimates, convergence rates and accuracy degenerate with increased noise corruption on the measurement data. Figures in Section B.3 of Appendix B show this accuracy deteriorating for the high noise case that was presented in Section 3.7.3. For a detailed analysis, the reader is referred to [28].

The three-dimensional location of the salient features is obtained using the structure estimates shown in Figure 3.7 and Equation (3.5). A few images representing the evolution of the three-dimensional structure from the planar initial estimate to the converged hemispherical surface are shown in Figure 3.9. This provides a more recognisable depiction of structural convergence than the single α parameterisation in Figure 3.7. Section B.4 of Appendix B provides further keyframes of this sequence providing a more detailed illustration of the structural evolution.



(a) Frame 1 (b) Frame 20 (c) Frame 80

Figure 3.9: Evolution of structure points to a hemispherical structure.

A discussion in [28] concerning depth reversal of structure estimates when all points are initialised in a common plane, suggests the introduction of a mild convexity bias. In the full tracking system presented in Chapter 5, the filter is encouraged to converge onto a stable and correctly convex structure using initial estimates from a three-dimensional face model. This is also a more efficient way of bootstrapping the Kalman filter and is further discussed in Section 5.3.1.

3.8 Conclusions on the SfM Technique

In this chapter a popular method of achieving online Structure from Motion has been discussed. The method is numerically stable due to a number of novel representations, such as object structure, translation and the central projection camera model. These parameterisations were originally proposed by Azarbayejani and Pentland in [28]. Since that time, the basis of the SfM technique has been used in a number of works [22, 20, 4, 55, 67].

Section 3.7 of this chapter presents a typical synthetic evaluation of the technique, with analysis of performance on two different levels of corrupted measurement data. The algorithm performs well for both sets of synthetic measurements, confirming the suitability and accuracy of the technique to the proposed head tracking problem. The use of a real motion trajectory, which deviates slightly from standard synthetic experimentation, also serves to confirm this suitability. Timing of the synthetic experiments showed that the

Kalman filter achieves full frame rates of 26 Hz using 21 (pre-processed) features. Bearing in mind that the implementation used for evaluation in this chapter is not optimised, the estimation module still runs within the real-time processing constraints discussed in Section 2.6.

A consequent topic arising from the estimation results for low and high noise levels is that of correctly modeling the reliability and accuracy of arriving measurement data to the algorithm. In a real system, measurements arrive from a two-dimensional tracking technique, for which the quality cannot always be guaranteed. Points on the object may undergo changes in aspect beyond the span of the 2D tracking method. Object points may become occluded or illumination variation may change the structural appearance of features so much that they cannot be accurately tracked. Thus a logical development in the framework would be to vary the covariance associated with arriving measurements such that accurate measurements contribute to the estimation process more than corrupted ones. This is the topic of Section 5.4, where occlusion of points and large inaccuracies on measurement data can be modeled in the measurement covariance matrix R_k when calculating the Kalman gain of Equation (3.37). This provides a varying indication on the reliability of measurements such that only good measurements are used in the estimation procedure.

In order for the proposed SfM framework to be effective with real video sequences, a method of tracking the two-dimensional location of salient points is required. Chapter 4 discusses an efficient 2D tracking algorithm that will provide these necessary measurements. Another set of experiments using ground truth data is used to demonstrate the accuracy that can be achieved using the proposed technique. This also serves to validate the choice of 2D tracking method for measurement acquisition.

Chapter 4

Visual Tracking

4.1 Introduction

Following the definition given in Chapter 3, a video sequence can be made up of temporal changes in motion of an object as it moves in front of a camera. Conversely, this could happen as the camera moves about the object. The projection of these moving objects onto the image plane provides strong visual cues for understanding the structure of a scene, as well as the relative 3D motions of the objects in the scene i.e. the inference of properties of a 3D world. The two primary problems facing motion analysis are the issues of *correspondence*, and *reconstruction*. Chapter 3 deals with the *reconstruction* problem, under the assumption that the SfM algorithm is provided with a set of 2D feature measurements. These measurements are assumed to be the 2D projections of salient 3D features on the object while it moves in front of the camera. This then identifies the *correspondence* problem; knowing which elements of the current frame correspond to the same elements of the previous frame of the sequence. In this sense, precise feature tracking is essential for the accurate recovery of three-dimensional motion and structure.

4.2 Tracking Techniques

There are two fundamentally distinct approaches that exist for tracking objects of interest through a series of images. The first derives an optical flow field (or dense motion field) for the sequence, and then analyses the structure of the flow field to infer the relevant structure and motion information. This is further addressed in Section 4.2.1. Some examples using this tracking technique were discussed in Chapter 2. Full frame algorithms like optical flow tend to lead to data intensive processing which is performed off-line or which is accelerated using specialised hardware [74]. The second approach is based on the correspondence of discrete features on an object in one image with those features in a subsequent image. Local feature tracking has found wide applicability in the vision and robotics literature, especially for the Structure from Motion task. Classic examples of features in computer vision are corners [75], lines [76, 77], deformable contours [78] or regions [79]. The following sections outline these two fundamental tracking methods.

4.2.1 Tracking via Image Flow

Image flow, or optical flow, assigns to each point in the image plane a two-dimensional (2D) velocity vector that is the projection of the apparent three-dimensional (3D) velocity of a scene point onto the image plane [80]. This flow field is then analysed to recover information about the dynamic scene and, in the case of object tracking, to make inferences about the motion of 3D objects in the scene.

It is well known in vision literature that true velocity can be recovered with local measurements only in areas with sufficient local intensity variation [80]. Thus, computation of image flow usually occurs in two steps: Using the image intensity distribution in small neighbourhoods with sufficient variation to compute local information about velocity, then propagating this information into neighbouring areas of the image to recover the full image flow.

There are three major approaches to recovering image flow from local information [81]: Correlation-based methods, intensity-based differential methods, and spatiotemporal en-

ergy methods. Barron et al. [82] compared the performance of many of these algorithms and their combinations. Each of these techniques with relevant references is briefly summarised below.

Correlation methods search the area surrounding each pixel in an image for a pixel in the next image with similar local neighborhood structure. The pixel in the new image with the most similar local neighborhood structure is taken to be the new location of the pixel.

Differential (or Gradient) based methods compute velocity from first-order derivatives of image intensity, or from filtered versions of the image. Horn and Schunk [83] are usually credited with pioneering this approach.

The *spatiotemporal energy* approach makes some assumptions about smoothness of the flow field in space and time, and uses 3D (space-time) Gabor filters to estimate the power spectrum of a moving pattern [81]. Simoncelli et al. [84] and Heeger [85] have presented spatiotemporal techniques for the computation of optical flow. Fleet and Jepson [86] use the phase-space response of similar filters to compute optic flow. Weber and Malik [87] have refined this approach, and give a systematic analysis of the error sources in the technique. Again, the reader is referred to [82] for an in-depth review of these different approaches.

4.2.2 Tracking via Feature Correspondence

An alternative to computing the motion of each pixel in each frame of a sequence is to only find correspondences between significant points in successive frames. These significant points are called features. A feature can be any easily observable characteristic of an object being tracked. Commonly, edges [77, 88], corners [78, 89], and areas of high visual contrast [79] are used as features in tracking research.

There are two main advantages to correspondence-based feature tracking over the image-flow approach described above. Since the velocity of only a few points in the image is measured, the computational burden is reduced significantly. Also, the ability to choose salient features increases the probability of achieving proper feature velocity measure-

ments. This can be done by choosing feature points with significant local image structure. On a frontally facing face, the eye, nose and mouth regions contain the highest levels of such structural information.

4.3 Region-based Tracking

A common special case of feature tracking occurs when the features to be tracked are regions in an image. Region-based tracking considers matching a stored reference window to a region of the current image. The region is either a region taken from the scene itself, or a presupplied target template. It is assumed throughout this discussion that the surface patch corresponding to the region of interest is roughly planar and that its projection is relatively small compared to the image as a whole, so that perspective effects are minimal [74]. In the context of a face, the eye and mouth regions can be considered approximately planar, thus being appropriate regions for region based tracking. Until significant out-of-plane rotations occur, template regions selected around the nose also perform well, as shown in Chapter 6.

The motion of the template patch is calculated through an image registration technique. At the most basic level, image registration (or alignment) consists of moving and possibly deforming a template to minimize the difference between the (deformed) template and an image. This concept is further clarified in Figure 4.1. In many cases the template is a cropped sub-region of another image, but it could also be a complex model of image appearance, such as an Active Appearance Model (AAM) [43].

The way in which the template can move and deform depends upon the application in question and can vary from a simple translation, as in the original Lucas-Kanade tracking algorithm [79], to a complex mesh-based alignment, as is the case for AAMs. The difference between the template and the image is usually taken to be the *sum of squared difference* (SSD), but can be more complex and (for example) take into account illumination variation [53], or even more general appearance variations [90]. Image registration is normally performed using gradient descent on the registration parameters [91], but linear

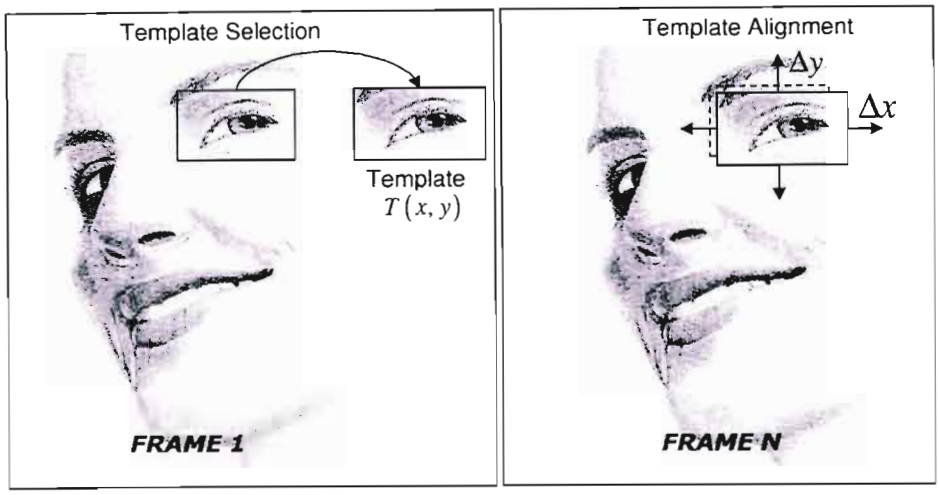


Figure 4.1: Template selection and template alignment.

regression has also been proposed as a way to estimate a linear approximation to the parameter updates in the gradient descent algorithm [90].

In general, registration techniques rely on the intensity of a pixel region remaining relatively constant throughout a sequence of alignments, thereby minimising the possibility of misalignment due to large template-to-target region error. This is called the *image constancy* criterion [69] and is a fundamental requirement of tracking techniques employing registration with no allowance for appearance change.

The first image registration technique was that of Lucas-Kanade [79], whose gradient descent type alignment algorithm is derived under the assumption that the image constancy rule holds. In reality, the minimisation of error approach that the method employs means that the algorithm can handle some small intensity changes due to small illumination variation or shadowing. It is preferable however, when these situations occur, to implement a more general algorithm that does not suffer as greatly from this inherent susceptibility to error that is present in most *real* systems. Section 4.5 discusses such a modification that incorporates some prior knowledge of object appearance. Before this is done however, Section 4.4 presents a reformulation of the original Lucas-Kanade approach that reduces the computational burden of performing an iterative alignment process for each frame of a tracking sequence. An introductory derivation of the Lucas-Kanade technique is covered

in Section 4.3.1, as it is a necessary foundation for the extensions and modifications used.

4.3.1 Lucas-Kanade Tracking

The Lucas-Kanade algorithm [79] is the original image alignment algorithm. The goal of Lucas-Kanade is to align a template image $T(\mathbf{x})$ to an input image $I(\mathbf{x})$, where $\mathbf{x} = (x, y)^T$ is a column vector containing the pixel coordinates. If the Lucas-Kanade algorithm is being used to compute optical flow or to track an image patch from time $t = 1$ to time $t = 2$, the template $T(\mathbf{x})$ is an extracted sub-region of the image at $t = 1$ and $I(\mathbf{x})$ is the image at $t = 2$.

Over time, the relative motion between the target object and the camera causes the image of the target to shift and to deform. The image motion of the target region on the object can be modeled by a parametric motion model $\mathbf{W}(\mathbf{x}; \mathbf{p})$, where $\mathbf{p}$ is a vector of motion parameters parameterised as $\mathbf{p} = (p_1, p_2, \ldots, p_n)^T$. The function $\mathbf{W}(\mathbf{x}; \mathbf{p})$ defines a warp that takes the pixel $\mathbf{x}$ in the coordinate frame of the template T and maps it to the sub-pixel location $\mathbf{W}(\mathbf{x}; \mathbf{p})$ in the coordinate frame of the image I. Hence, the recovery of the motion parameter vector $\mathbf{p}$, for each image in the tracking sequence, can be considered as "tracking the object".

The motion parameter vector of the target region can be estimated by minimising the sum of squared error between two images; namely the template and the image warped back onto the coordinate frame of the template:

$$\sum_{\mathbf{x} \in R} \left(I\left(\mathbf{W}\left(\mathbf{x}; \mathbf{p}\right)\right) - T(\mathbf{x}) \right)^2 \tag{4.1}$$

The warping of I back to compute $I(\mathbf{W}(\mathbf{x}; \mathbf{p}))$ requires a (bilinear) interpolation of the image at the sub-pixel locations $\mathbf{W}(\mathbf{x}; \mathbf{p})$. The minimization in Equation (4.1) is performed with respect to the motion parameter vector $\mathbf{p}$, with the sum performed over all of the pixels $\mathbf{x} \in R$ in the template image $T(\mathbf{x})$ (here $R = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N)$ is the set of N image locations which define the template region). Minimising this expression is

a non-linear optimization task even if $\mathbf{W}(\mathbf{x};\mathbf{p})$ is linear in $\mathbf{p}$ as the pixel values are, in general, non-linear in $\mathbf{x}$. In fact, the pixel values $I(\mathbf{x})$ are essentially un-related to the pixel coordinates $\mathbf{x}$. The optimisation of the expression in Equation (4.1) relies on the assumption that a current estimate of $\mathbf{p}$ is known and that iterative increments to the parameters $\Delta\mathbf{p}$ can be solved; i.e. the following expression is (approximately) minimised:

$$\sum_{\mathbf{x}\in R} \left(I\left(\mathbf{W}\left(\mathbf{x};\mathbf{p}+\Delta\mathbf{p}\right)\right) - T(\mathbf{x})\right)^2 \tag{4.2}$$

with respect to $\Delta\mathbf{p}$, and then the motion parameters are updated:

$$\mathbf{p} \leftarrow \mathbf{p} + \Delta\mathbf{p} \tag{4.3}$$

These two steps are iterated until the estimates of the parameters $\mathbf{p}$ converge. Typically, the test for convergence is whether some norm of the vector $\Delta\mathbf{p}$ is below a threshold ϵ i.e. $\|\Delta\mathbf{p}\| \leq \epsilon$.

4.3.2 Deriving the Lucas-Kanade Algorithm

Standard derivations of the original Lucas-Kanade algorithm can be found in [79] or [92] from which much of the discussion and notation in this section and Section 4.4.1 derives. The derivation begins by assuming that the non-linear expression in Equation (4.1) can be linearised by performing a first order Taylor expansion on the approximation of Equation (4.2), the result of which is:

$$\sum_{\mathbf{x}\in R} \left(I\left(\mathbf{W}\left(\mathbf{x};\mathbf{p}\right)\right) + \nabla I \frac{\partial \mathbf{W}}{\partial \mathbf{p}} \Delta\mathbf{p} - T\left(\mathbf{x}\right)\right)^2 \tag{4.4}$$

This expression can be broken down into the following components. The term

$$\nabla I = \left(\frac{\partial I}{\partial x}, \frac{\partial I}{\partial y}\right) \tag{4.5}$$

is the *gradient* of the image I evaluated at $\mathbf{W}(\mathbf{x};\mathbf{p})$. This implies that ∇I is computed in the coordinate frame of I and then warped back onto the coordinate frame of T using the current estimate of the warp $\mathbf{W}(\mathbf{x};\mathbf{p})$. Again, the matrix of partial derivatives in $\frac{\partial \mathbf{W}}{\partial \mathbf{p}}$ represents the *Jacobian* of the warp. So, assuming $\mathbf{W}(\mathbf{x};\mathbf{p}) = (W_x(\mathbf{x};\mathbf{p}), W_y(\mathbf{x};\mathbf{p}))^T$ one can write:

$$\frac{\partial \mathbf{W}}{\partial \mathbf{p}} = \begin{pmatrix} \frac{\partial W_x}{\partial p_1} & \frac{\partial W_x}{\partial p_2} & \cdots & \frac{\partial W_x}{\partial p_n} \\ \frac{\partial W_y}{\partial p_1} & \frac{\partial W_y}{\partial p_2} & \cdots & \frac{\partial W_y}{\partial p_n} \end{pmatrix} \tag{4.6}$$

Baker et.al [92] name the parameter $\nabla I \frac{\partial \mathbf{W}}{\partial \mathbf{p}}$ the *steepest descent* images, a term that will be referred to later in this chapter. Equation (4.4) represents a least squares problem [79]. A closed form solution to this problem is now derived.

Taking the partial derivative of the expression in Equation (4.4) with respect to $\Delta\mathbf{p}$ results in:

$$\sum_{\mathbf{x}\in R} \left(\nabla I \frac{\partial \mathbf{W}}{\partial \mathbf{p}}\right)^{\mathrm{T}} \left(I(\mathbf{W}(\mathbf{x};\mathbf{p})) + \nabla I \frac{\partial \mathbf{W}}{\partial \mathbf{p}} \Delta\mathbf{p} - T(\mathbf{x})\right) \tag{4.7}$$

Setting this expression equal to zero and solving for $\Delta\mathbf{p}$ results in a closed form solution for Equation (4.4) written as:

$$\Delta\mathbf{p} = H^{-1} \sum_{\mathbf{x}\in R} \left(\nabla I \frac{\partial \mathbf{W}}{\partial \mathbf{p}}\right)^{\mathrm{T}} (T(\mathbf{x}) - I(\mathbf{W}(\mathbf{x};\mathbf{p}))) \tag{4.8}$$

where H is a *Hessian* matrix of the form:

$$H = \sum_{\mathbf{x}\in R} \left(\nabla I \frac{\partial \mathbf{W}}{\partial \mathbf{p}}\right)^{\mathrm{T}} \left(\nabla I \frac{\partial \mathbf{W}}{\partial \mathbf{p}}\right) \tag{4.9}$$

Looking at the expression for $\Delta\mathbf{p}$, the term $\sum_{\mathbf{x}\in R} \left(\nabla I \frac{\partial \mathbf{W}}{\partial \mathbf{p}}\right)^{\mathrm{T}} (T(\mathbf{x}) - I(\mathbf{W}(\mathbf{x};\mathbf{p})))$, is called (by Baker et.al [92]) the *steepest descent parameter update*. These terms will be used for ease of reference to certain expressions. Thus the optimal choice of the motion parameter

updates, $\Delta\mathbf{p}$, are a combination of the steepest descent parameter updates multiplied by the inverse of the Hessian matrix in Equation (4.9). The system is nonsingular if the image gradients in the template region are not all collinear [53]. The matching algorithm then consists of iteratively applying Equations (4.8) and (4.3). The 9 steps performed in each of these iterations are shown in Figure 4.2.

The Lucas-Kanade Algorithm

Iterate:

(1) Warp I with $\mathbf{W}(\mathbf{x};\mathbf{p})$ to compute $I(\mathbf{W}(\mathbf{x};\mathbf{p}))$

(2) Compute the error image $T(\mathbf{x}) - I(\mathbf{W}(\mathbf{x};\mathbf{p}))$

(3) Warp the gradient ∇I with $\mathbf{W}(\mathbf{x};\mathbf{p})$

(4) Evaluate the Jacobian $\frac{\partial \mathbf{W}}{\partial \mathbf{p}}$

(5) Compute the steepest descent images $\nabla I \frac{\partial \mathbf{W}}{\partial \mathbf{p}}$

(6) Compute the Hessian matrix using Equation (4.9)

(7) Find the steepest descent updates $\sum_{\mathbf{x}\in R} \left(\nabla I \frac{\partial \mathbf{W}}{\partial \mathbf{p}}\right)^{\mathrm{T}} \left(T(\mathbf{x}) - I(\mathbf{W}(\mathbf{x};\mathbf{p}))\right)$

(8) Compute $\Delta\mathbf{p}$ using Equation (4.8)

(9) Update the parameters $\mathbf{p} \leftarrow \mathbf{p} + \Delta\mathbf{p}$

until $\|\Delta\mathbf{p}\| \leq \epsilon$

Figure 4.2: The 9 online steps of the Lucas-Kanade image alignment algorithm.

The main problem with the original Lucas-Kanade algorithm is the number of computationally expensive steps required when iterating over Equations (4.8) and (4.3). Section 4.4 presents some improvements to the original algorithm that contribute to the efficiency and reduction of the computational overhead of the method.

4.4 An Efficient Tracking Algorithm

In [91], Baker et.al distinguish between two different gradient descent type image alignment algorithms based on the Lucas-Kanade algorithm. The first they call an *additive* approach, because there is an additive increment to the warp parameters $\mathbf{p}$. The original Lucas-Kanade algorithm utilises an additive update. The second approach has been called a *compositional* approach, where instead an incremental warp is "composed" with the current warp. The concept of composition will be expanded upon later in this chapter in Section 4.6. Within these different approaches, authors in [53, 93, 91] have proposed improvements to the original Lucas-Kanade alignment algorithm which produce more efficient solutions. One key feature that these authors point out is the large computational cost of re-evaluating the Hessian in every iteration.

If the Hessian were constant, it could be pre-computed and then re-used and the number of computationally expensive steps in the algorithm could be significantly reduced. One approach, used by Shum et al. [93], is to only update the Hessian every few iterations and provide an efficient approximate to the Hessian. The problem with these approximations is the difficulty in evaluating the tradeoff between accuracy and computational efficiency [91]. A better approach would be to reformulate the image alignment problem such that the solution is the same, but the Hessian remains constant.

In this vein, Hager et al. [53] propose an efficient algorithm based on an additive formulation. The key to efficiency in their derivation is to apply a change of variables that *inverts* the role of the image and the template, hence being dubbed an *inverse additive* approach by [91]. To do this the Jacobian of the change of variables takes on a modified form, allowing for the offline computation of certain parameters. This modification unfortunately leads to a restriction in the type of warp $\mathbf{W}(\mathbf{x};\mathbf{p})$ the algorithm can use. They present 2D translational, 2D similarity and 2D affine warps. Baker et al. [91] propose an equivalently efficient but more versatile solution, known as an *inverse compositional* alignment algorithm. Their solution can be applied to any set of warps that form a group and provides the framework for the 2D tracker used in this work. The modifications to the Lucas-Kanade algorithm that give rise to the Inverse Compositional solution are

performed in Section 4.4.1.

4.4.1 Inverse Compositional Image Alignment

As originally performed in [53], the key to efficiency is in switching the role of the template and the image. This can be done using either an additive or a compositional approach. The *Inverse Compositional* (IC) algorithm of Baker and Mathews [91] tries to minimise:

$$\sum_{\mathbf{x}\in R} \left[T\left(\mathbf{W}\left(\mathbf{x};\Delta\mathbf{p}\right)\right) - I\left(\mathbf{W}\left(\mathbf{x};\mathbf{p}\right)\right)\right]^2 \tag{4.10}$$

with respect to $\Delta\mathbf{p}$ and then updates the warp as:

$$\mathbf{W}\left(\mathbf{x};\mathbf{p}\right) \leftarrow \mathbf{W}\left(\mathbf{x};\mathbf{p}\right) \circ \mathbf{W}\left(\mathbf{x};\Delta\mathbf{p}\right)^{-1} \tag{4.11}$$

Note that the incremental warp $\mathbf{W}\left(\mathbf{x};\Delta\mathbf{p}\right)$ is *inverted* before it is composed with the current estimate, $\mathbf{W}\left(\mathbf{x};\mathbf{p}\right)$. The warp parameters $\mathbf{p}$ can then be extracted from the warp $\mathbf{W}\left(\mathbf{x};\mathbf{p}\right)$, depending on what form the warp takes. The choice of this warping model will be the topic of Section 4.6.

Again, performing a first order Taylor expansion on Equation (4.10) results in:

$$\sum_{\mathbf{x}\in R} \left[T\left(\mathbf{W}\left(\mathbf{x};0\right)\right) + \nabla T\frac{\partial\mathbf{W}}{\partial\mathbf{p}}\Delta\mathbf{p} - I\left(\mathbf{W}\left(\mathbf{x};\mathbf{p}\right)\right)\right]^2 \tag{4.12}$$

Here, the assumption is made [91] that $\mathbf{W}\left(\mathbf{x};\mathbf{0}\right)$ is an identity warp; i.e. $\mathbf{W}\left(\mathbf{x};\mathbf{0}\right) = \mathbf{x}$. The solution to this least squares problem can be written as:

$$\begin{aligned}\Delta\mathbf{p} &= -H^{-1}\sum_{\mathbf{x}\in R}\left(\nabla T\frac{\partial\mathbf{W}}{\partial\mathbf{p}}\right)^T\left(I\left(\mathbf{W}\left(\mathbf{x};\mathbf{p}\right)\right) - T\left(\mathbf{x}\right)\right)\\ &= -H^{-1}\sum_{\mathbf{x}\in R} SD\left(\mathbf{x}\right)^T E\left(\mathbf{x}\right)\end{aligned} \tag{4.13}$$

$E(\mathbf{x})$ is the error image and $SD(\mathbf{x})^T$ are again the *steepest-descent images*, where:

$$SD(\mathbf{x}) = \nabla T \frac{\partial \mathbf{W}}{\partial \mathbf{p}} \tag{4.14}$$

H is still the Hessian matrix but with I replaced by T, such that:

$$H = \sum_{\mathbf{x} \in R} \left(\nabla T \frac{\partial \mathbf{W}}{\partial \mathbf{p}} \right)^{\mathrm{T}} \left(\nabla T \frac{\partial \mathbf{W}}{\partial \mathbf{p}} \right) \tag{4.15}$$

with the Jacobian $\frac{\partial \mathbf{W}}{\partial \mathbf{p}}$ being evaluated at $(\mathbf{x}; \mathbf{0})$. Since there is nothing in the Hessian that depends on $\mathbf{p}$, it is constant across iterations and can be pre-computed.

This makes the IC algorithm a far more computationally efficient solution compared to the original Lucas-Kanade algorithm, as many of the time consuming steps of the algorithm are now an offline pre-computational step. The only extra cost comes from the inversion of $\mathbf{W}(\mathbf{x}; \mathbf{\Delta p})$ and its composition with $\mathbf{W}(\mathbf{x}; \mathbf{p})$. These steps are quite involved, but as shown by Baker et al. [91], their computational overhead is minimal. An overview of the steps involved in the algorithm are shown in Figure 4.3. The numbering of each step follows that of the original Lucas-Kanade algorithm, as an indication of which steps have become pre-computational tasks. A more insightful view of the necessary steps is shown in a schematic representation in Figure 4.4.

Section 4.5 presents a brief discussion on the problems that appearance variation can cause in a tracking system, with proposals for a minor modification to the IC algorithm which accommodates for realistic appearance change.

The Inverse Compositional Algorithm

Pre-compute:

(3) Evaluate the gradient ∇T of the template $T(\mathbf{x})$

(4) Evaluate the Jacobian $\frac{\partial \mathbf{W}}{\partial \mathbf{p}}$ at $(\mathbf{x}; \mathbf{0})$

(5) Compute the steepest descent images $\nabla T \frac{\partial \mathbf{W}}{\partial \mathbf{p}}$

(6) Compute the Hessian matrix using Equation (4.15)

Iterate:

(1) Warp I with $\mathbf{W}(\mathbf{x}; \mathbf{p})$ to compute $I(\mathbf{W}(\mathbf{x}; \mathbf{p}))$

(2) Compute the error image $I(\mathbf{W}(\mathbf{x}; \mathbf{p})) - T(\mathbf{x})$

(7) Find the steepest descent updates $\sum_{\mathbf{x} \in R} \left(\nabla I \frac{\partial \mathbf{W}}{\partial \mathbf{p}} \right)^{\mathrm{T}} \left(I\left(\mathbf{W}\left(\mathbf{x}; \mathbf{p}\right)\right) - T\left(\mathbf{x}\right) \right)$

(8) Compute $\Delta \mathbf{p}$ using Equation (4.13)

(9) Update the warp $\mathbf{W}(\mathbf{x}; \mathbf{p}) \leftarrow \mathbf{W}(\mathbf{x}; \mathbf{p}) \circ \mathbf{W}(\mathbf{x}; \Delta \mathbf{p})^{-1}$

until $\|\Delta \mathbf{p}\| \leq \epsilon$

Figure 4.3: Offline and online steps of the Inverse Compositional algorithm.

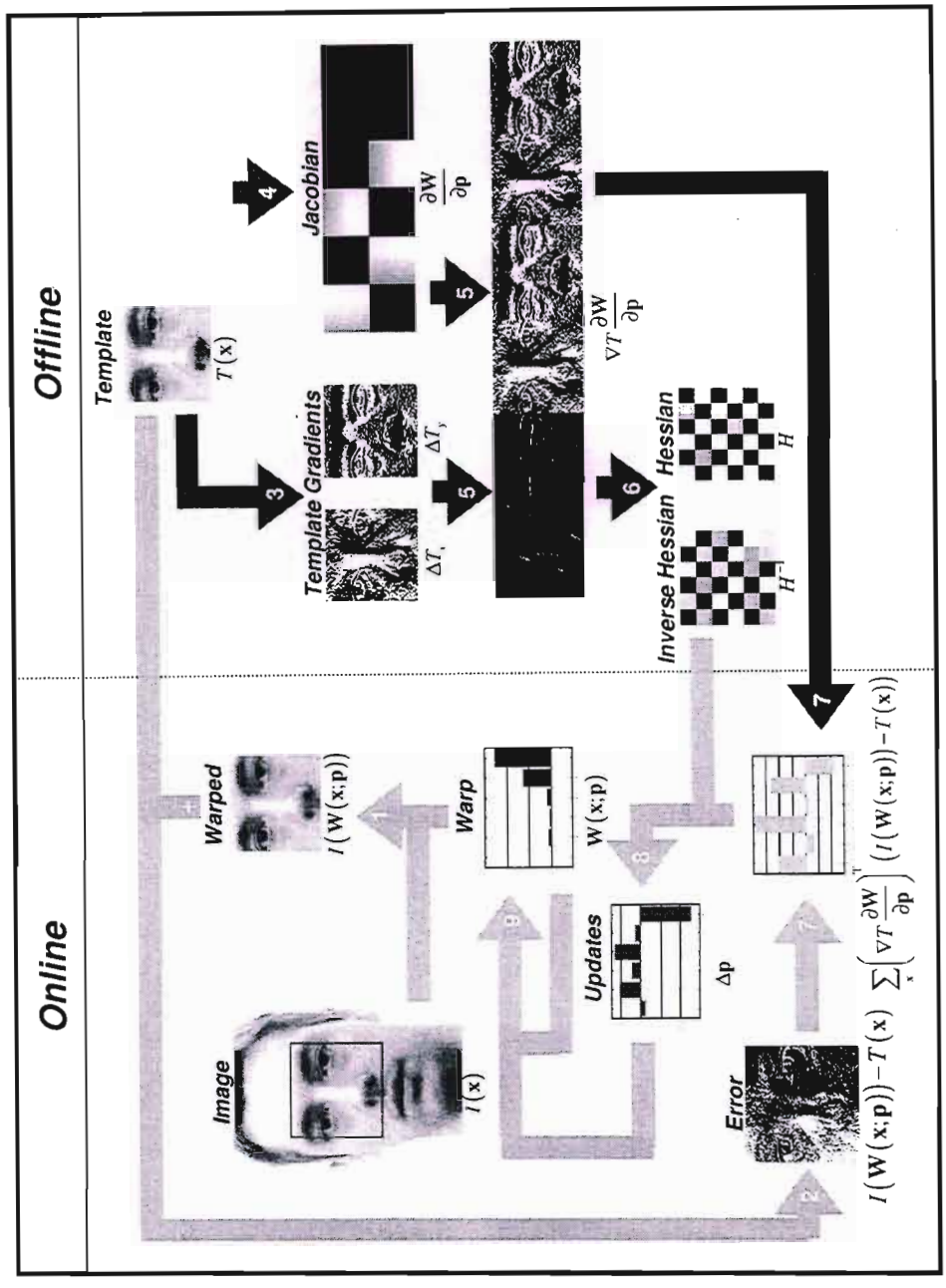


Figure 4.4: Schematic overview of the Inverse Compositional algorithm. Steps (3–6) are performed once as an *offline* pre-computation. The *online* algorithm consists of: image warping (Step 1), image differencing (Step 2), image dot products (Step 7), multiplication with the inverse of the Hessian (Step 8), and the update to the warp (Step 9).

4.5 Modeling Appearance Variation

The Inverse Compositional image alignment algorithm has been shown, in [91], to be an accurate and efficient image registration technique, highly suitable for region based tracking under the *image constancy assumption* [69]. In real-world applications however, one of the major challenges that all visual tracking algorithms face is their ability to cope with changes in the appearance of the target during tracking. These appearance changes can be caused by a variation in the illumination, an occlusion or a change in the aspect of the target itself caused by a change of pose or, for example, in the case of face tracking, by a change of facial expression. Tracking algorithms try to accommodate these variations by modeling target appearance in various ways. Some works have used texture [94], colour [95] or both [96]. Others have employed 3D models [27].

Other popular methods are based on linear subspace models [97, 53], or shape and texture [43]. In an effort to accommodate such anticipated variations in the proposed face tracking system, this thesis adopts a subspace modeling extension to the IC algorithm, originally presented by Baker et al. [91]. They describe an augmentation of the IC tracking algorithm with a linear appearance subspace, which they have applied to their work on Active Appearance models [98].

Section 4.5.1 begins with a brief discussion on modeling linear appearance variation, the purpose being the incorporation of an appearance basis into the IC algorithm. In an effort to maintain the efficiency of the algorithm however, Baker et al.[91] perform a further derivation of the algorithm that incorporates such an appearance basis but does not add any extra computational burden to the online steps of the algorithm. Because the final solution to the algorithm very closely resembles the IC algorithm without an appearance basis, only the solution to this derivation is presented in Section 4.5.1. For the interested reader, the full derivation is presented in Section C.1 of Appendix C

4.5.1 Linear Appearance Variation

The Inverse Compositional algorithm of Section 4.4 assumes that there is a template $T(\mathbf{x})$ present in the input image $I(\mathbf{x})$, albeit warped by $\mathbf{W}(\mathbf{x};\mathbf{p})$. When there is some form of appearance variation however, the template image does not explicitly occur. In this case, it is a better choice to modify the assumption such that:

$$T(\mathbf{x}) + \sum_{i=1}^{m} \lambda_i A_i(\mathbf{x}) \tag{4.16}$$

appears in the input image (appropriately warped) instead of $T(\mathbf{x})$.

Here, $A_i, i = 1, \ldots, m$, represent a set of known appearance images of the target, captured offline under different appearance conditions. For the face, this could be different illumination conditions, facial expressions or pose. The coefficients $\lambda_i, i = 1, \ldots, m$ represent a set of unknown appearance parameters which are used to weight each of the appearance images. With a linear combination of some number, N, of appearance parameters λ_N acting on an orthonormal appearance basis A_N, any arbitrary illumination variation [53] or more general appearance variation [98, 97] can be modeled. The appearance images, A_i, can be orthonormalised via a Gram-Schmidt process which can be found in [99]. Section 4.7.3 presents experimental results demonstrating this appearance modeling capability.

Modification of the inverse compositional approach begins by assuming that the expression of Equation (4.16) appears in the input image $I(\mathbf{x})$, and that the original minimisation of Equation (4.1) becomes

$$\sum_{\mathbf{x}\in R} \left(T(\mathbf{x}) + \sum_{i=1}^{m} \lambda_i A_i(\mathbf{x}) - I(\mathbf{W}(\mathbf{x};\mathbf{p})) \right)^2 \tag{4.17}$$

The inverse compositional algorithm including appearance variation is then derived from Equation (4.17) in a two stage least squares optimisation, the details of which are presented in Section C.1 of Appendix C. The notable result of this minimisation is that the addition of an appearance basis does not drastically change the form of the original inverse

compositional algorithm and in fact, all new additions to the algorithm are calculations that can be performed offline.

The steps of the algorithm remain the same as those outlined in Figure 4.3, except that the calculation of the steepest descent images is slightly modified. The expression for calculating them is given by:

$$SD(\mathbf{x}) = \nabla T \frac{\partial \mathbf{W}}{\partial \mathbf{p}}(\mathbf{x};\mathbf{0}) - \sum_{i=1}^{m} \left(\sum_{\mathbf{y}} A_i(\mathbf{y}) \nabla T \frac{\partial \mathbf{W}}{\partial \mathbf{p}}(\mathbf{y};\mathbf{0}) \right) A_i(\mathbf{x}) \tag{4.18}$$

from which the Hessian can again be computed as:

$$H = \sum_{\mathbf{x}} SD(\mathbf{x})^T SD(\mathbf{x}) \tag{4.19}$$

This does add some degree of computational complexity to the pre-computational steps of the algorithm, but the online algorithm remains just as efficient as the original inverse computational solution. This is a significant result considering the increase in robustness the addition of an appearance basis contributes to the algorithm. If required, a post-computational step may be added in order to find the appearance parameters λ_i, using:

$$\lambda_i = \sum_{\mathbf{x} \in R} A_i(\mathbf{x}) \left(I(\mathbf{W}(\mathbf{x};\mathbf{p})) - T(\mathbf{x}) \right) \tag{4.20}$$

The IC algorithm with an appearance basis now constitutes the measurement acquisition module for the SfM algorithm in Chapter 3, providing accurate 2D coordinates of salient features moving in the scene. Before the algorithm can be implemented, a motion model which defines the type of warp the template region can undergo must be selected. This is discussed in Section 4.6. Using this motion model, Section 4.7 presents a brief assessment of the algorithm in terms of alignment accuracy and tracking under varying appearance.

4.6 Motion Model Selection

It has been established in the discussions on the IC algorithm that all warps $\mathbf{W}(\mathbf{x};\mathbf{p})$ are parameterised by a motion vector $\mathbf{p}$. This vector can constitute any number of parameters to form different warping functions. The most simple warp is a 2D translation. This was the first motion model in the original registration technique presented by Lucas and Kanade [79]. From there authors have extended the Lucas-Kanade technique to include rotations and scaling (similarity transform), or full affine transformations [100, 101]. The Inverse Compositional algorithm of Baker and Mathews [91] has been extended to handle all of the above models, as well as homographies and 3D rotations of the template area. Figure 4.5 gives an overview of the forms of the different transforms.

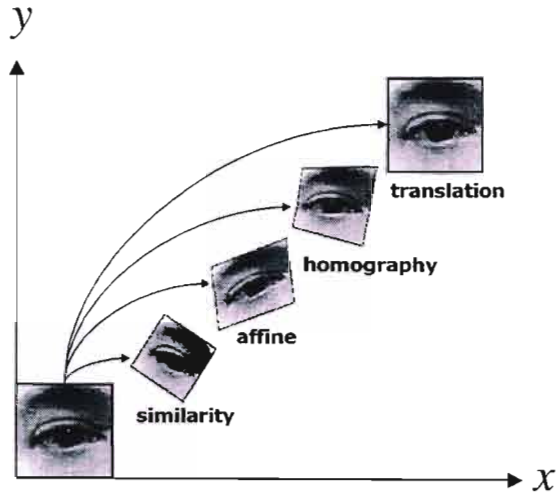


Figure 4.5: Fundamental set of 2D planar transformations

The region of interest in this thesis is a roughly planar face region containing a number of structurally rich areas (eyes, mouth and nose) which are suitable for tracking. Although the human head is close to ellipsoidal in shape, the facial surfaces are close enough to planar that an image registration technique would be suitable for recovering in plane and most medium sized out of plane motions that can occur. The algorithm would typically lose the feature being tracked if it undergoes excessive changes that are beyond the span of

the 2D motion model and the span of the integrated appearance basis. Fortunately, within the SfM algorithm of Chapter 3, the EKF includes the ability to probabilistically encode the accuracy of the feature matches in the measurement covariance matrix. This means that it is possible to constrain the individual tracking regions based on the understanding of the global motion of the object being tracked. Thus losing a feature because the motion model of the 2D tracker cannot account for the complexity of the warp, does not mean that the feature is lost forever. This will be discussed further in Sections 5.4 and 5.5

The motion model used in this implementation of the Inverse Compositional algorithm is an *affine* model. An *affine* model is the simplest motion model that can account for 2D translations, rotations, and shearing [69]. Additional parameters which define a more accurate motion model (such as a homography) can be added at the cost of extra computational burden. The affine model however, accurately describes the in-plane motion of the facial region and is able to maintain good estimates of medium-sized out-of-plane motion to the point of significant appearance variation or occlusion. This model also adds 2 extra degrees of freedom compared with the similarity transform used in the head tracker of Jebara et al. [22]. In their system extra points could be harnessed from templates by using the better motion model (as opposed to the traditional translational model [79]). The use of an affine model further extends this capability.

The affine model is parameterised by a motion parameter vector

$$\mathbf{p}=(p_1,p_2,p_3,p_4,p_5,p_6)^T \tag{4.21}$$

where the warping function $\mathbf{W}(\mathbf{x};\mathbf{p})$ transforms the source image location $\mathbf{x}=(x,y)^T$ to the new location $\mathbf{x}'=(x',y')^T$ via Equation (4.22):

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \mathbf{W}(\mathbf{x};\mathbf{p}) = \begin{pmatrix} (1+p_1) & p_3 & p_5 \\ p_2 & (1+p_4) & p_6 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \tag{4.22}$$

The scaling s, shearing γ, rotation θ and translation (t_x,t_y) parameters are encoded into

the structure of the affine transform as:

$$\begin{pmatrix} (1+p_1) & p_3 & p_5 \\ p_2 & (1+p_4) & p_6 \end{pmatrix} = \begin{pmatrix} s+\gamma\theta & -s\theta+\gamma & t_x \\ s\theta & s & t_y \end{pmatrix} \tag{4.23}$$

where it has been assumed that the rotation angle θ will be small such that $\sin(\theta) \simeq \theta$ [74].

The Jacobian matrix $\frac{\partial \mathbf{W}}{\partial \mathbf{p}}$ with respect to the motion parameters $\mathbf{p}$ is given by:

$$\frac{\partial \mathbf{W}}{\partial \mathbf{p}} = \begin{pmatrix} x & 0 & y & 0 & 1 & 0 \\ 0 & x & 0 & y & 0 & 1 \end{pmatrix} \tag{4.24}$$

The update of the warp parameters is the composition of two different warps based on the current estimate $\mathbf{p}$ and the incremental parameters $\Delta\mathbf{p}$. A standard compositional warp between a warp $\mathbf{W}(\mathbf{x};\mathbf{p})$ and an incremental warp $\mathbf{W}(\mathbf{x};\Delta\mathbf{p})$ can be written [92] as:

$$\mathbf{W}(\mathbf{x};\mathbf{p}) \circ \mathbf{W}(\mathbf{x};\Delta\mathbf{p}) \equiv \mathbf{W}(\mathbf{W}(\mathbf{x};\Delta\mathbf{p});\mathbf{p}) \tag{4.25}$$

Using the affine warp of Equation (4.22), the new warp is then:

$$\mathbf{W}(\mathbf{x};\mathbf{p}) \circ \mathbf{W}(\mathbf{x};\Delta\mathbf{p}) =$$

$$\begin{pmatrix} (1+p_1)\cdot((1+p_1)\cdot x + \Delta p_3 \cdot y + p_5) + p_3 \cdot (\Delta p_2 \cdot x + (1+p_4)\cdot y + \Delta p_6) + p_5 \\ p_2 \cdot ((1+\Delta p_1)\cdot x + \Delta p_3 \cdot y + \Delta p_5) + (1+p_4)\cdot(\Delta p_2 \cdot x + (1+p_4)\cdot y + \Delta p_6) + p_6 \end{pmatrix}$$

$$\tag{4.26}$$

A simpler way of calculating this expression is to re-write the affine warping function $\mathbf{W}(\mathbf{x};\mathbf{p})$ of Equation (4.22) as:

$$\mathbf{W}(\mathbf{x};\mathbf{p}) = \begin{pmatrix} (1+p_1) & p_3 & p_5 \\ p_2 & (1+p_4) & p_6 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = A(\mathbf{p}) \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \tag{4.27}$$

where $A(\mathbf{p})$ is now a homogenous form of the affine transformation in terms of the motion parameters $\mathbf{p}$. The composition of the two warps can then be expressed by:

$$\mathbf{W}(\mathbf{x};\mathbf{p}) \circ \mathbf{W}(\mathbf{x};\Delta\mathbf{p}) = A(\mathbf{p})A(\mathbf{\Delta p}) \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \tag{4.28}$$

The new warp formed by $A(\mathbf{p})A(\mathbf{\Delta p})$ will retain the structure of the affine warp in Equation 4.22, so the new values of the motion vector $\mathbf{p}$ can be obtained where:

$$\mathbf{p} = \begin{pmatrix} p_1 + \Delta p_1 + p_1 \cdot \Delta p_1 + p_3 \cdot \Delta p_2 \\ p_2 + \Delta p_2 + p_2 \cdot \Delta p_1 + p_4 \cdot \Delta p_2 \\ p_3 + \Delta p_3 + p_1 \cdot \Delta p_3 + p_3 \cdot \Delta p_4 \\ p_4 + \Delta p_4 + p_2 \cdot \Delta p_3 + p_4 \cdot \Delta p_4 \\ p_5 + \Delta p_5 + p_1 \cdot \Delta p_5 + p_3 \cdot \Delta p_6 \\ p_6 + \Delta p_6 + p_2 \cdot \Delta p_5 + p_4 \cdot \Delta p_6 \end{pmatrix} \tag{4.29}$$

With the inverse compositional algorithm however, the inverse of the incremental warp is composed with the current warp, as in Equation 4.11. In a similar fashion to the composition of the two warps in Equation (4.28), the update to the motion parameters in the inverse compositional algorithm can be found using:

$$\mathbf{W}(\mathbf{x};\mathbf{p}) \circ \mathbf{W}(\mathbf{x};\mathbf{\Delta p})^{-1} = A(\mathbf{p})A(\mathbf{\Delta p})^{-1} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \tag{4.30}$$

so that the new motion parameter vector $\mathbf{p}$ can be found from $A(\mathbf{p})A(\mathbf{\Delta p})^{-1}$

This simple warp update completes the derivation of an efficient tracking algorithm employing an affine motion model. Using this motion model in the modified IC algorithm, Section 4.7 presents a brief set of experiments to evaluate the accuracy and suitability of the matching method.

4.7 Alignment and Appearance Change Experiments

In the evaluation which follows, two types of experiments are performed. The first transforms a selected image region using a ground truth set of affine parameters. This process provides an initial image from which a template can be chosen and a target image, with which to demonstrate the ability of the inverse compositional algorithm to quickly converge to estimates of the known affine parameters. The images used in this manual transformation are shown in Figure 4.6. In the final tracking system the IC algorithm is used to track selected image regions and thereby provide optimal estimates of the location of corners for each of the template regions. These experiments consequently concentrate on the accuracy of recovery of the corner locations for each template. Thereafter, the convergence of the incremental, $\Delta\mathbf{p}$, and affine motion estimates $\mathbf{p}$ are discussed.

The second experiment demonstrates the ability of the modified inverse compositional algorithm to handle moderate appearance variation. The experiments show how a relatively small number of appearance images can be used to model the transition from a known template to an unknown image region with fairly significant amounts of illumination and expression variation.

(a) (b)

Figure 4.6: Template, Figure 4.6(a), and Target, 4.6(b), frames used in the the evaluation of the IC image alignment algorithm.

4.7.1 Pixel Alignment Error

A significant result in terms of evaluating alignment accuracy is the decay of the RMS error between the known location of the transformed corners and the iteratively updated estimates available by transforming the original template corners using the updated warp $\mathbf{W}(\mathbf{x};\mathbf{p})$. The error is expected to tend to zero for an alignment to be declared good. In situations where the appearance of the target changes, one cannot guarantee as good a match. In this experiment however, the image constancy assumption is held, so the alignment is expected to be accurate.

The result of a 20 iteration cycle of the algorithm is shown in Figure 4.7. When the appearance of the target region remains relatively constant and only small iterative transformations of the region occur, the experiments performed (as well as results presented in [92]) confirm that convergence is typically achieved within $5 - 15$ iterations. The results shown in Figure 4.7 demonstrate the convergence of the RMS point error. Despite a relatively large transformation of the image patch, the resulting RMS point error is close to zero for each of the corners. This indicates that subpixel accuracy has been achieved.

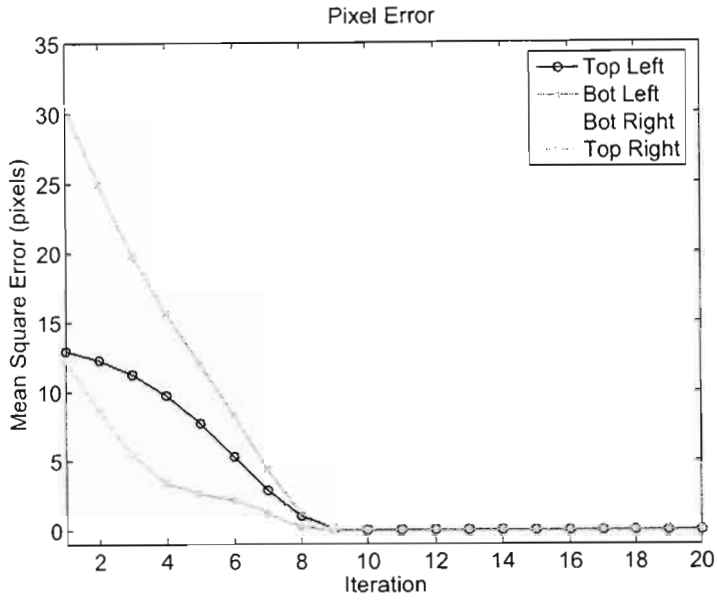


Figure 4.7: Corner pixel alignment errors.

4.7.2 Parameter Convergence

This section evaluates the speed and accuracy of motion parameter convergence. With this objective in mind, both the incremental parameter updates $\Delta\mathbf{p}$ as well as the final set of warp parameters $\mathbf{p}$ are assessed. The ground truth affine parameters which transform the template image in Figure 4.6(a) to the warped region in Figure 4.6(b) are given as a vector in Equation (4.31):

$$\mathbf{p} = \begin{pmatrix} 0.06509 & -0.19199 & 0.39199 & 1.10000 & 3.000 & -3.000 \end{pmatrix}^T \tag{4.31}$$

This parameter vector equates to a rotation of 10° anti clockwise, a scaling $s = 1.1$, and a sheer of $\gamma = 0.2$. This would be considered a fairly large inter-frame warp in a real tracking scenario, but is used here to demonstrate the accuracy of the alignment algorithm.

As a test for convergence at each iteration, the norm of the incremental motion parameter vector $\Delta\mathbf{p}$ is found. When the norm is below some threshold, the template is assumed

converged. In Figure 4.8(b) and 4.8(d), each of the incremental changes eventually tends to zero. Re-iterating, this is only possible because the image constancy assumption holds and a close to perfect match can be found. In the ideal case then, that norm would be some value very close to zero. In a practical scenario however, it makes better sense to assume some small variability is always present, and set the norm at some small value ϵ.

$$\mathbf{p} = \begin{pmatrix} 0.06498 & -0.19199 & 0.39189 & 1.10012 & 3.001 & -2.998 \end{pmatrix}^T \tag{4.32}$$

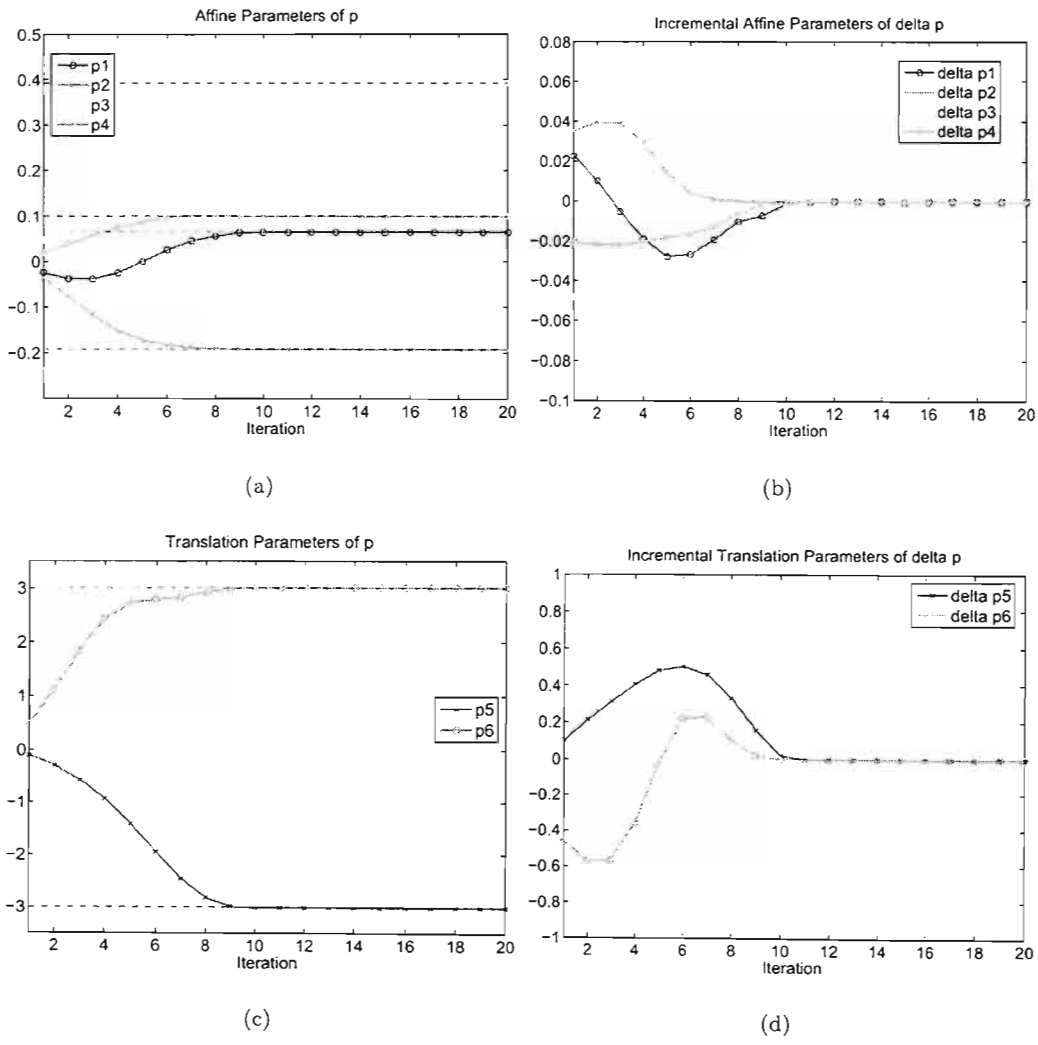


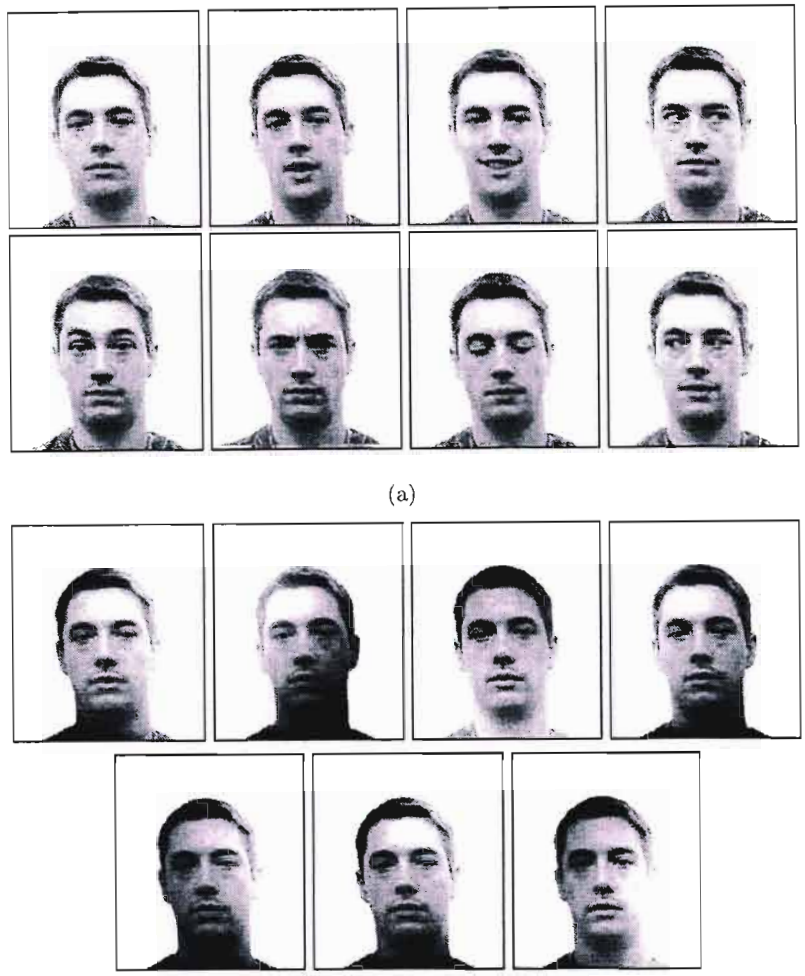
Figure 4.8: Incremental motion parameter updates.

Iterative updates to the motion parameter estimates are given in Figures 4.8(a) and 4.8(c), demonstrating accurate convergence within 10 iterations. The estimated parameter vector is given in Equation (4.32). In a real tracking system, where incremental motion is generally much smaller, convergence is typically achieved sooner; an average of 5 iterations is often sufficient. When appearance changes occur however, this number may increase up to around 15 iterations before a reliable match can be declared (assuming that the template does not diverge and the track fails).

4.7.3 Adaptive Appearance Modeling

The modification of the inverse compositional algorithm to handling appearance changes requires that a number of basis images A_i are integrated using Equation (4.18) into the steepest descent images in the pre-computational steps of the algorithm. The basis images are chosen dependant on the types of appearance change that could occur during tracking. In the head tracking system proposed in this thesis, both illumination and expression change may occur, thus the appearance basis must include training images which exhibit these variations.

A typical set of appearance images for face tracking are shown in Figure 4.9. For each new person, a similar training set would be required. This could be implemented as an automated process in a complete system, but is currently an offline acquisition involving manual face alignment with the template frame. In the experiments performed in this section, the two parts of the 15 image appearance basis in Figure 4.9 include expression images in Figure 4.9(a) and illumination images in Figure 4.9(b). In Figure 4.9(a), the template image is usually selected as the first appearance image [102]. The next 7 images depict examples of smiling, blinking, frowning, surprise and gaze direction. The illumination images are instances of various lighting directions that can occur. Each of the appearance images must be orthonormalised before it can be integrated into the appearance basis. This can be done using Gram-Schmidt orthonormalisation, found in [99].



(a)

(b)

Figure 4.9: Typical set of training images for an appearance basis.

Figure 4.10 depicts a standard tracking scenario, similar to the template to target transition in Figure 4.6. In this instance however, the transition from the template frame at t_0 to a frame at t_N involves significant variation in both expression and illumination. Also, the image at frame t_N does not exist in the appearance basis shown in Figure 4.9. Yet, as can be seen from the image at t_N in Figure 4.10, an accurate estimate of the target region is acquired by offsetting the original template with a linear combination of the images in the appearance basis. This is more easily interpreted from the images in Figure 4.11.

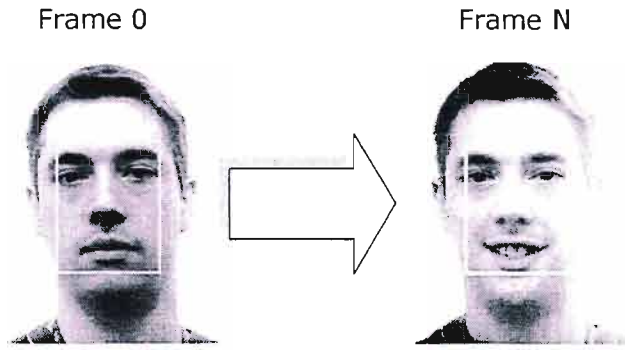


Figure 4.10: Transition from template frame at t_0 to target frame at t_N under significant appearance change.

The images in Figure 4.11(a) depict the pixel regions enclosed in the template $T(\mathbf{x})$, and target $I(\mathbf{W}(\mathbf{x};\mathbf{p}))$, regions of Figure 4.10. Comparison of these images with the appearance corrected images in Figure 4.11(b) reveal that the appearance estimates are very close to the actual regions in each image.

(a)

(b)

Figure 4.11: Template, target and appearance corrected regions.

The left image in Figure 4.11(b) represents the target region with aspect and illumination correction, given by:

$$I(\mathbf{W}(\mathbf{x};\mathbf{p})) - \sum_{i=1}^{15} \lambda_i A_i \tag{4.33}$$

which very closely resembles the original template $T(\mathbf{x})$ in Figure 4.11(a). Similarly, the right hand image in Figure 4.11(b) represents $T(\mathbf{x}) + \sum_{i=1}^{15} \lambda_i A_i$, which is very close to the target region $I(\mathbf{W}(\mathbf{x};\mathbf{p}))$ in Figure 4.11(a).

The final set of experiments present a comparison between the IC algorithm *with* an appearance basis and the IC algorithm *without*, shown in Figure 4.12. The first column of images in Figure 4.12(a) represent the template regions selected at t_0. The second column in Figure 4.12(b) depicts the transition to the target region of the IC algorithm without illumination compensation. As can be seen from each of the results, the target region has either a poor correspondence with the original template region or fails entirely. When an appearance basis is included, the column of alignment results in Figure 4.12(c) confirm that the transition from template to target remains stable and accurate. For each of the tracking examples, the final column in Figure 4.12(d) presents the original template, $T(\mathbf{x})$, target, $I(\mathbf{W}(\mathbf{x};\mathbf{p}))$, and illumination corrected ($T(\mathbf{x}) + \sum_{i=1}^{15} \lambda_i A_i$) regions respectively. As in Figure 4.11(b), the illumination corrected templates very closely resemble the tracked target region.

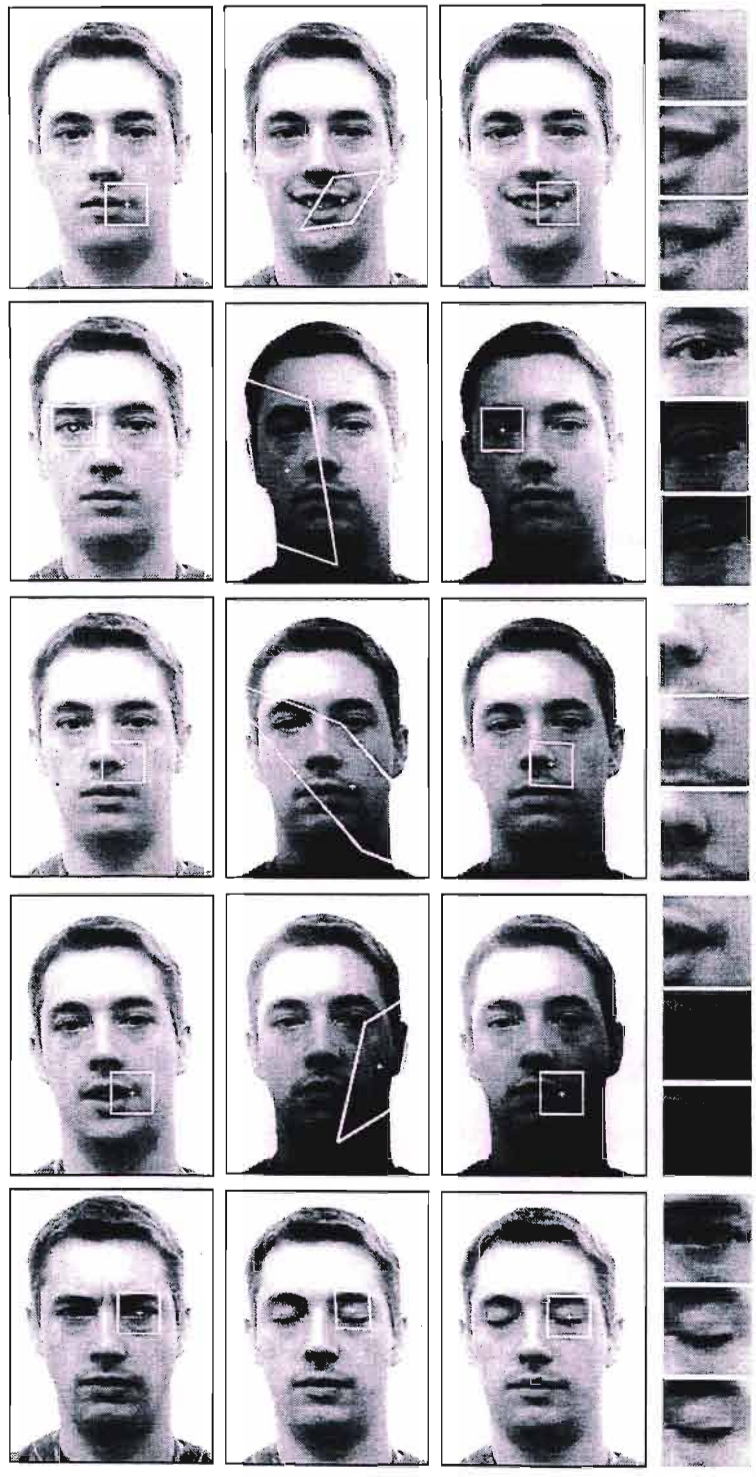


Figure 4.12: Experiments comparing the IC algorithm. Template selection in 4.12(a). False/inaccurate alignment in 4.12(b) with no included appearance basis. Accurate alignment in 4.12(c) when an appearance basis is included.

4.8 2D Tracking Conclusions

In this chapter an efficient region tracking algorithm has been presented which is robust to changes in both aspect and illumination. The region tracking approach is highly suitable for face tracking, as the face has a close-to planar surface with several structurally rich areas. Feature tracking methods are also preferable when computational burden must be minimised, as far fewer pixel locations must be used as compared with optical flow based techniques.

Experiments performed in Section 4.7.1 demonstrate that subpixel alignment accuracy can be achieved, making the algorithm a good choice as the measurement acquisition stage for the SfM algorithm. Section 4.7.3 furthers the tracking algorithm through integration of a set of appearance images. The appearance basis used in the experiments of Section 4.7.3 is typical of an appearance basis used for face tracking in this thesis. It is shown that despite significant changes in aspect and illumination, the algorithm remains robust and accurate. This is advantageous in a realistic head tracking scenario where expression change and illumination are likely to vary.

In Chapter 5, final components and initialisation of the head tracking system are discussed. This includes a feedback process from the SfM solution to further constrain 2D tracking as well as a correlation based evaluation of the quality of pixel measurements being fed into the SfM algorithm.

Chapter 5

Tracking System Implementation

Chapter 2 presented a brief outline of the proposed head tracking system. The discussion describes the use of a robust and efficient tracking method (the IC algorithm introduced in Chapter 4) used to provide pixel measurements required by a Kalman filter based pose estimator (discussed in Chapter 3). This approach is primarily built on the Structure from Motion based head tracking system of Jebara et al. [22]. This thesis extends the original concepts of Jebara et al. [22] by using a higher degree of freedom motion model in the 2D tracking algorithm such that a greater number of structure points may be harnessed from each tracking template. Further developments include a different 2D tracking algorithm which incorporates a linear appearance basis to improve 2D tracking robustness.

The challenge in this section lies in getting the Kalman filter and the IC tracking algorithm to work in harmony such that optimal pose estimates can be achieved, at the same time minimising the possibility of tracking failure. This chapter begins with a brief overview of the complete tracking system in Section 5.1. Thereafter, the subsequent sections present a more in depth look at the major implementation details highlighted in the system overview. Section 5.2 begins with a discussion on initialisation of the template tracking algorithm and the selection of an appearance basis. Next, Section 5.3 covers the selection of initial estimates and noise statistics for the Kalman filter, with emphasis in Section 5.4 on modeling a dynamically varying measurement covariance to improve track-

ing stability and occlusion handling. Finally, Section 5.5 describes the use of structural knowledge in constraining template tracking, then Section 5.6 finishes the chapter with some concluding remarks on the complete system design.

5.1 Schematic Overview

Figure 5.1 presents a schematic overview of the complete tracking system. The primary objective of the head tracking system is to optimally estimate head pose and structure using 2D observations from the 2D feature tracker. With this objective in mind, the important aspects are now highlighted.

Each step of the system begins by tracking corresponding features in the incoming frames of the video sequence. Within the 2D tracking algorithm, template regions must be selected and an appearance basis for the person being tracked must be created. This is discussed in Section 5.2.

At each frame, the 2D feature measurements are then fed to the EKF framework for pose estimation. In order to indicate the quality of measurements to the EKF, Section 5.4 discusses a reliability measure in the form of a dynamically varying covariance on each feature. This is also fed to the Kalman filter framework. On initialisation however, the Kalman filter requires initial estimates of system state and noise statistics. The choice of these values is covered in Section 5.3.

The outputs of the system are 3D pose estimates (which have been used to animate a projection of the Candide [1] face model) and an estimate of the structure of the head being tracked. These results are the focus of Chapter 6. As a form of feedback, Section 5.5 discusses the use of structural estimates to constrain 2D feature tracking in each frame, minimising the possibility of tracking failure due to occlusion.

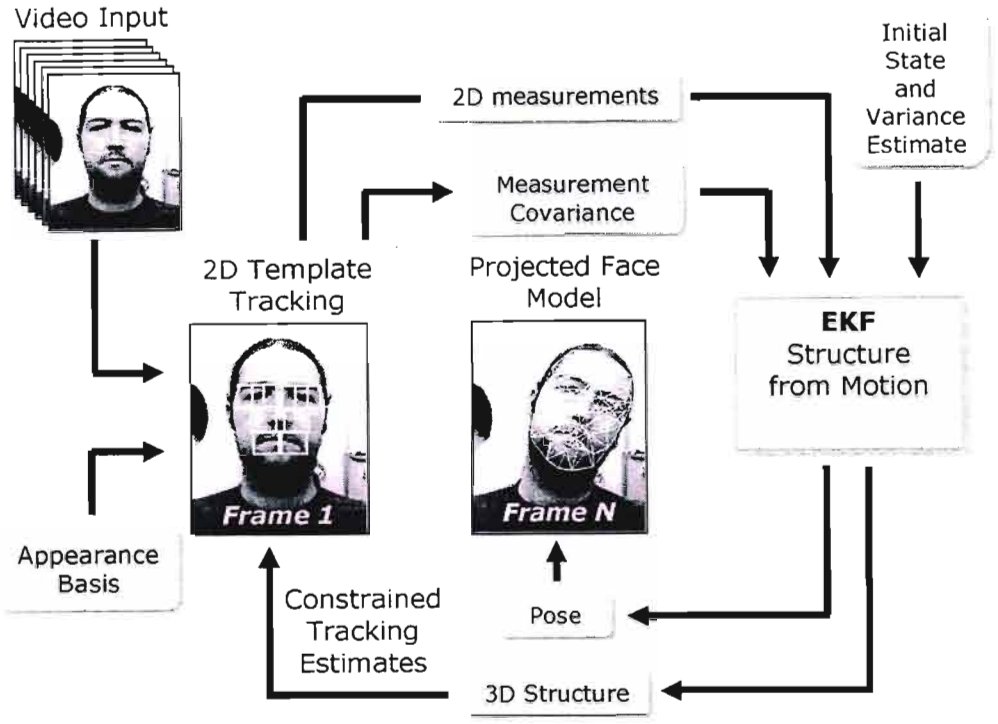


Figure 5.1: Schematic overview of the tracking system.

5.2 2D Tracker Initialisation

When a registration based tracking method is used to track region based features, the best choice of window regions are in areas of high local intensity variation [80]. On the face, these areas are the eye, nose, mouth and brow regions, as shown in Figure 5.2.

In each frame, the 2D tracking algorithm deforms the windows using a specific motion model, in order to align them with the transformed target region. In [22], Jebara et al. use a 2D similarity transform (rotation, translation and scaling) to model the motion of the template. Using this motion model, it is argued in [22] that the extra degrees of freedom enable the use of points on the template window other than the traditional center point. Instead, opposing template corners are used as measurements to the EKF, as the 2D motion model sufficiently represents the motion of the face occurring in the scene. This thesis proposes an extension to this concept, as discussed in Section 4.6, where an affine model of 2D template motion is used. The additional degrees of freedom once again

Figure 5.2: Template windows selected around the eyes, corners of the nose and the mouth.

establish the option of using more points on the template region.

The reason for increasing the number of available measurements from each template window comes from the direct influence the quantity of available measurements has on the convergence of the Kalman filter. The more measurements the Kalman filter can use, the more overconstrained the system becomes, thus the more likely it is that an accurate state solution will be recovered. In this thesis, the method of using corner points is extended to using all four corners of the deforming template region, as shown in Figure 5.3.

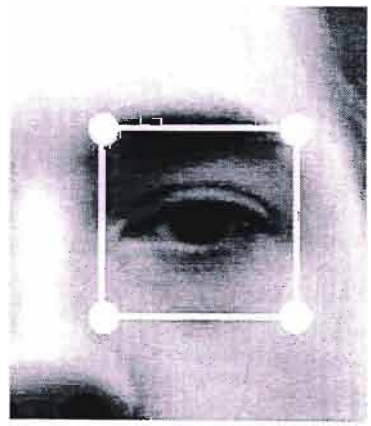
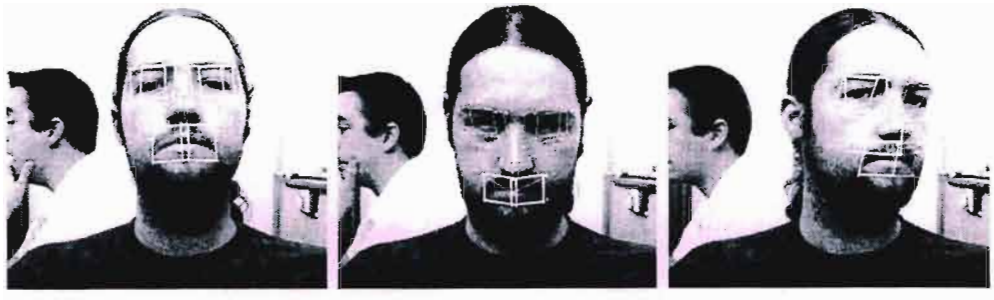


Figure 5.3: The four corners of each template are used as 2D tracking measurements.

Using the four template corners adds $4*2N = 8N$ measurements to the system, where N is the number of templates used. With the system overconstrained when $2N > N + 7$ (discussed in Section 3.4.2), the state parameters are recoverable when at least two templates are trackable at any time. This implies that at least two of those window regions selected over the eyes, nose and mouth must be trackable in each frame of the video sequence. When in-plane rotations occur, the structure of the face remains constant, as the surfaces are roughly planar and no significant changes in aspect occur. When Pitch or Yaw rotations occur, the slight curvature irregularities on the face become more prominent. Aspect changes due to these out-of-plane rotations can inhibit the ability of template windows to converge. It is thus necessary to ensure that when such rotations do occur, the minimum of two template regions are still trackable.

For this reason, the subsequent discussion focuses on the ability to track each template window under the various Pitch and Yaw rotations that can occur. For positive Pitch, the structure of the mouth and eye regions remain consistent with the frontally selected templates, and experiments typically show that a minimum of six windows consistently provide measurements to the Kalman filter, as shown in Figure 5.4(a). For increasing negative Pitch, Figure 5.4(b) shows how the overhang of the brow region begins to alter the structure of the target region. Also the nostril regions become occluded on the nose, resulting in aspect changes beyond the span of the 2D tracking algorithm. In this case, only the mouth region remains fairly constant, such that two templates are still providing the necessary measurements. For both positive and negative Yaw, the same aspect changes and occlusions occur. As the head rotates, one side of the nose becomes self-occluded and the aspect changes on one side of the eye and mouth region can become significant. In these cases, the measurements come from the four template regions facing the camera, which are still observable within the span of the 2D tracking algorithm. This is shown in Figure 5.4(c).



(a) (b) (c)

Figure 5.4: Tracking images demonstrating which template windows are most reliable under different out-of-plane rotations.

There will always be a trade-off between the accuracy gained by increasing number of points harnessed from each template window and the increase in computational burden placed on the Kalman filter. This point is argued in context of the work of Jebara et al.[22]. Of all the head tracking systems discussed in Section 2.6 of Chapter 2, their system is the most computationally efficient. By harnessing extra points on template windows, Kalman filter convergence speed and accuracy is increased. This thesis proposes a further increase in the number of measurements available from each template by using an even higher order motion model than the one used in [22]. In this way, one may argue, increased robustness can be achieved without any significant increase in computational burden.

5.2.1 Including an Appearance Basis

Section 4.5 presents an enhanced version of the Inverse Compositional algorithm which includes a linear appearance basis, making the algorithm more robust to changes in illumination and changes in aspect caused by pose or facial expression. The appearance basis is built up of a number of images chosen offline - essentially a training set. In this thesis, where the subjects of interest are faces, a subspace model is built up from a number of images of a users face under different appearance conditions. This has been demonstrated in the experiments of Section 4.7.3 of Chapter 4.

An obvious requirement on the appearance images is that the faces in each image are aligned as closely as possible with the original template frame. This ensures that when template windows are selected over the eyes, nose and mouth, the basis templates are suitably near in region to the regions selected from the template. In this thesis, this is a manual process performed offline.

Before images can be used to create an appearance basis and composed into the steepest descent parameters of Equation (4.18), a requirement on the training set is that the vector form of the appearance regions form an orthonormal basis [102]. Orthonormalising the images can be performed using a Gram-Schmidt process which can be found in [99]. Once orthonormalisation has been performed on the image vectors, the basis can be neatly folded into the offline calculations of the inverse compositional algorithm (following the discussion in Section 4.5 of Chapter 4), adding no further online computational burden, yet increasing tracking robustness.

5.3 Kalman Filter Initialisation

Kalman filtering has long been recognised as the optimal estimation technique for problems involving linear dynamics [50]. The extension to nonlinear models using the extended Kalman filter is a mathematical approximation, which has fortunately been shown to work well for many applications, and more specifically the recursive Structure from Motion formulation of Azarbayejani and Pentland [28]. Previously discussed in the synthetic experiments of Section 3.7, the performance response of the Kalman filter can vary quite significantly depending on the initial estimates of system state and noise characteristics used as a starting point for parameter estimation. The EKF used in this thesis requires initial estimates for $N + 7$ state parameters, an associated error covariance for each state, as well as system and measurement noise covariances. The choice of these parameters very closely resembles the choices made in the synthetic experiments of Section 3.7. The subsequent sections will briefly re-address these design choices, with emphasis on a more accurate initial structure estimate.

5.3.1 Initial Structure

A key assumption made in this work is that initial head pose is known in the first frame of the video sequence. Because all initialisation procedures are performed offline, one can ensure that the system starts only when the head in the video sequence meets two important criteria. Firstly, the head must fill at least one-third of the viewing space, such that sufficient structural information is visible to ensure the integrity of the tracking templates. Secondly the head must be in a frontal position, shown in Figure 5.5.

These assumptions would not be acceptable in a real system, where face detection would be utilised. For the purposes of this thesis however, the emphasis is on demonstration of parameter recovery more than completeness of the head tracking system. Thus, these requirements are not a significant issue.

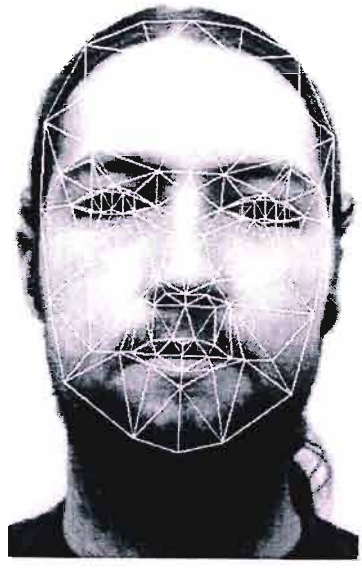


Figure 5.5: Candide face model [1] manually aligned to the face for initial structure estimates.

In the synthetic experiments performed in Section 3.7, initial structure assumes that all points lie in a common plane at some distance parallel to the image plane. To speed up structural convergence when tracking a real object, it is advantageous to utilise some

a priori knowledge of object structure. In the head tracking system, the frontal position constraint can be exploited such that an offline manual alignment of a generic face model can be performed. For simplicity, this work uses the Candide face model [1]. The model consists of approximately 100 polygons. The positions of specific coordinates on the model surface are then used as a starting point for structure estimation in the Kalman filter framework. The model positions are not the true 3D locations of the points, but they provide a reasonable (scaled) starting point for structure estimation. The image coordinates of each of the feature points (u^0, v^0) are used to represent the structure according to Equation (3.5).

5.3.2 Model Coordinates

Structure estimates are calculated using some basic ray tracing concepts. In ray tracing, a ray of light is traced in a backwards direction. That is, starting from the eye or camera, the ray is traced through a pixel in the image plane into the scene to determine what that ray intersects. The intersection is thus found by generating rays that pass from the COP through the image plane, which one knows will intersect the face model at some point in space, as in Figure 5.6. The locations of the face model vertices are known as it has been manually aligned with the face. Thus finding the intersection point of the ray and the particular surface on the model constitutes estimating the depths of the structure points on the face.

For each template window used in the 2D tracking algorithm, the corners of the template constitute the desired structure points of interest on the face. Thus for each of the eight templates used, four depth estimates must be found. The first step of this process requires the association of the correct three-vertex plane on the model surface with the ray passing through each template corner.

The three vertices which form the desired plane are found by projecting *all* of the Candide model vertices onto the image plane. For each template corner, the three projected vertices which surround it constitute the points of the desired plane on the model surface. Thus for each corner, three vertex points are recorded. A typical set of vertices are shown in

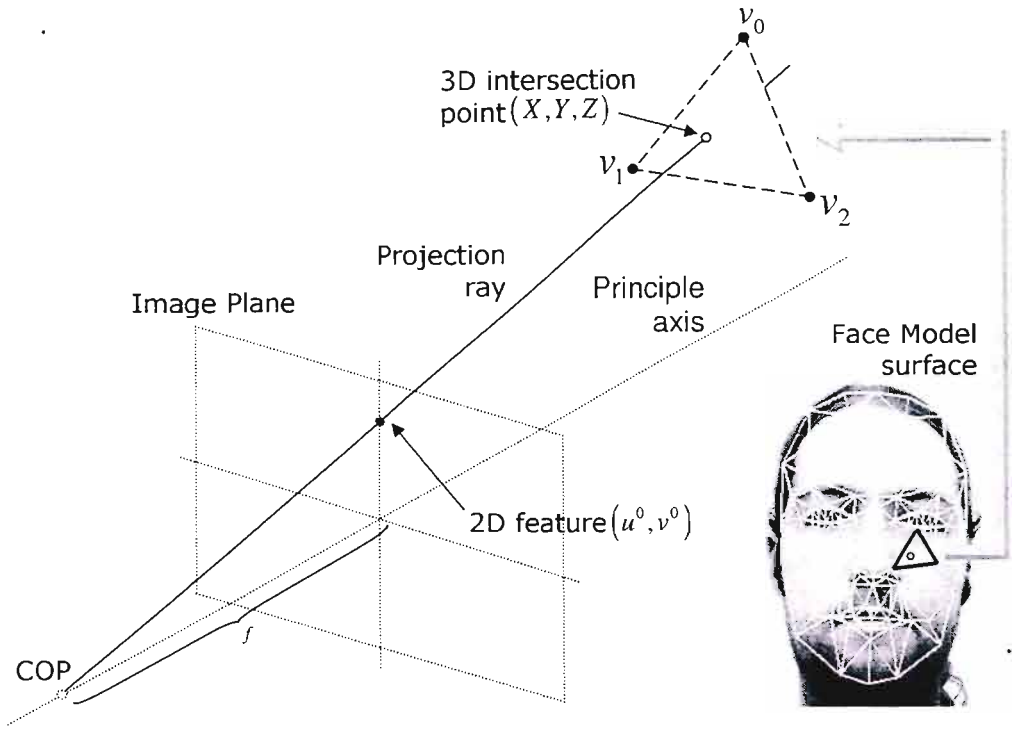


Figure 5.6: Ray tracing method for the estimation of depth.

Figure 5.7. In the figure, dark circles represent the structure points on the corners of the template windows. Light squares form the three associated vertices for each of the structure points. Section 5.3.3 covers the calculation of the three-dimensional intersection points.

5.3.3 Calculating Intersections

Each vertex on the model has a known three dimensional location in the camera frame of the form $\mathbf{V_i} = (v_x, v_y, v_z)$. When finding each intersection point, one may consider each plane individually, with their corresponding vertices of $(\mathbf{V_0}, \mathbf{V_1}, \mathbf{V_2})$. These can be used to generate the vectors $\mathbf{V}_{01} = \mathbf{V}_0 - \mathbf{V}_1$ and $\mathbf{V}_{02} = \mathbf{V}_0 - \mathbf{V}_2$. Taking the cross product of these vectors

$$N = \mathbf{V}_{01} \times \mathbf{V}_{02} \tag{5.1}$$

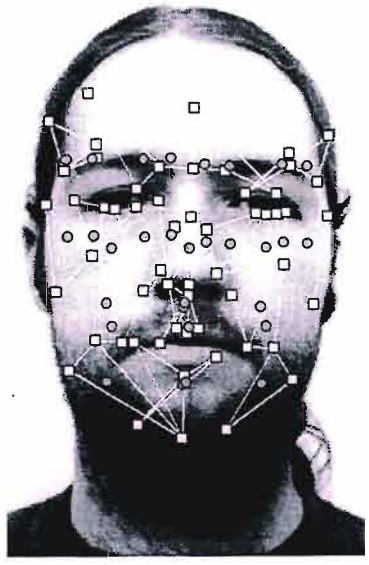


Figure 5.7: Location of surrounding model vertices for each template corner.

gives a vector $N = (n_x, n_y, n_z)$, which is normal to the planar surface formed by the vertices. The equation of the plane can then be written as

$$n_x x + n_y y + n_z z = C \tag{5.2}$$

where the location (x, y, z) defines any 3D point that lies on the plane. Finding the constant C is just a matter of substituting one of the vertex points V_0,V_1 or V_2 into Equation (5.2).

Next, one must consider the parametric equations of the line passing from the COP through the image plane at the specific structure point. A standard perspective projection is simple to work with in this case, so it is assumed that the center of projection (COP) is located at $\mathbf{X_0} = (0, 0, 0)$ and the point on the image is defined by $\mathbf{X_1} = (u^0, v^0, f)$, where f is the camera focal length. The parametric equations for the line passing through this point may then be written as:

$$
\begin{aligned}
x &= x_0 + t(x_1 - x_0) = tu^0 \\
y &= y_0 + t(y_1 - y_0) = tv^0 \\
z &= z_0 + t(z_1 - z_0) = tf
\end{aligned}
\tag{5.3}
$$

Substituting Equation (5.3) into Equation (5.2) permits the calculation of a value for t, which on back-substitution into Equation (5.3), gives the (x, y, z) location of the point on the surface of the model in the camera frame. Typical depth estimates from the Candide face model are shown in Figure 5.8.

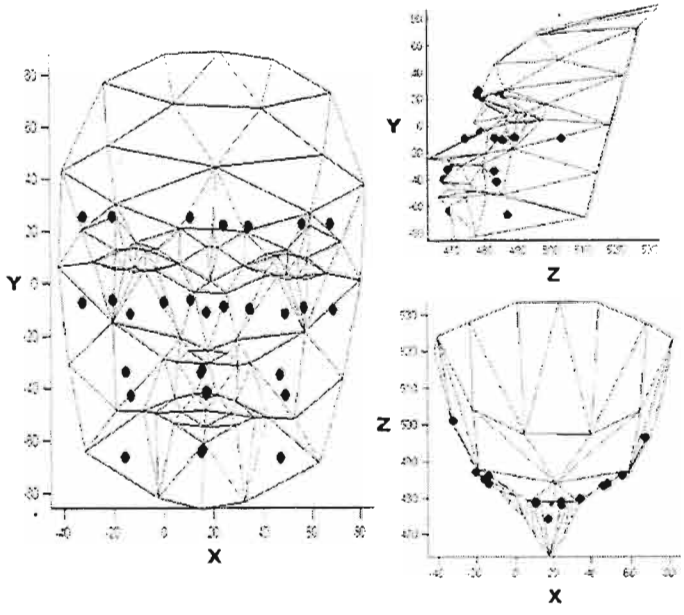


Figure 5.8: Initial structure estimates on the Candide face model.

Due to the form of the central projection camera model, the depth point α can be found by subtracting the focal length f from the calculated z coordinate. This parameter can then be used in the structure representation of Equation (3.5) as the estimated depth of the point for initialisation in the Kalman filter. As with the synthetic experiments, in order to control the scaling of the estimated parameters it is necessary to fix one of the structure points to its initial value as tracking begins. This scales all other structure points around that value. Again, α_0 is arbitrarily chosen as the fixed point by setting its initial error

covariance, $P_{0_{\alpha_0}}$, to zero.

5.3.4 Camera and Motion Parameters

Following the approach in the synthetic experiments of Section 3.7, fairly intuitive initial estimates for rotation and translation are used. Again, it is assumed that the head is starting from rest in an approximately frontal position. Consequently, the initial motion on the camera is set to zero.

In Section 3.7, the camera parameter β is shown to converge well when using synthetic data under both high and low noise conditions. For the real video sequences used in Chapter 6, a fair estimate of the camera parameter is known from the manufacturer's specifications ($f = 5$). Consequently, the initial camera parameter is set at this value and assigned a low initial error covariance. Assuming the *actual* focal length may vary up to 0.5 mm in either direction of this initial estimate, the error covariance is set at $P_{0_\beta} \sim 0.0005\ mm^2$. This value is calculated in the same manner as the error covariance estimate of Section 3.7.4.

5.3.5 Noise Statistics

Initial estimates for the noise characteristics of the system model again emulate those chosen in the synthetic experiments of Section 3.7. The hemispherical object was purposefully chosen with dimensions similar to a human head. Also, the magnitude and velocity of the motion assumed in the synthetic scene is characteristic of a moving head in a videophone scenario. For this reason, the same values used in the experiments of Section 3.7 are used as system noise covariance values for motion and structure. For translation, rotation and structure, this translates to $Q_t \sim 50\ mm^2$, $Q_r \sim 0.0025\ rad^2$ and $Q_\alpha \sim 0.01\ mm^2$ respectively. The camera parameter is also allowed to vary, randomly but smoothly via a small system covariance estimate. Again, from Section 3.7.4, this is set at $Q_\beta \sim 4E{-}8\ mm^{-2}$.

5.4 Measurement Noise Covariance

Thus far, the Kalman filter has been discussed as a reliable state parameter estimator. Within the framework of the filtering equations however, other useful properties exist which one can exploit to increase estimation accuracy. Traditionally, the covariance matrix R_k, as well as Q_k and P_0, are assumed to be known *a priori* [50]. These values give an indication to the filter of the relative accuracy of current state estimates and the importance that should be given to arriving measurements. In some cases, the values of R_k may vary from time step to time step. If these variations can be modeled, then filtering performance can be improved. This, as discussed in Gelb [50], is known as *adaptive* Kalman filtering, where a dynamically varying measurement covariance R_k changes with the arrival of new observation vectors z_k, in order to model the confidence of the new data. This reliability is inherent within the Kalman gain equation, repeated here for convenience as:

$$K_k = P_k^- H_k \left(\hat{\mathbf{x}}_k^-\right)^T \left[H_k \left(\hat{\mathbf{x}}_k^-\right) P_k^- H_k \left(\hat{\mathbf{x}}_k^-\right)^T + R_k\right]^{-1} \tag{5.4}$$

In the case of 2D feature tracking, such a situation occurs and appropriate weighting of the spatial measurements being fed to the filter becomes a fundamental aspect of the tracking system. For instance, some points during tracking may be occluded, and their position measurements will therefore have large errors. Consequently, the corresponding elements in R_k will be large. In this this thesis, variation in R_k is modeled by measuring the normalised correlation coefficient, ρ, between the template and tracked target region; a low ρ indicates a high R_k element and vice versa. The equation for normalised correlation between the template and target region is given by:

$$\rho = \frac{\sum_i (t_i - \bar{t})(i_i - \bar{i})}{\sqrt{\sum_i (t_i - \bar{t})^2 \cdot \sum_i (i_i - \bar{i})^2}} \tag{5.5}$$

where $\bar{t}$ and $\bar{i}$ are the means of the template and target regions respectively.

In the case of template matching, two types of matching errors can occur. Either the correct match is found, offset only slightly because of small changes in lighting, small errors in the projective transformation of the patch, or some small variations in aspect etc. In this thesis this is called the *minor error* case. The other possibility is that the match is completely wrong. This may occur due to occlusions or large changes in aspect or illumination beyond the span of the appearance basis. This is named the *major error* case.

During experimentation, with both synthetic data as well as real data, it was found that assigning a variance too low could be dangerous, causing a divergence of the solution because the filter assumes a high confidence despite the measurements inaccuracy. Manual optimisation of this parameter resulted in a minimum standard deviation, σ, of one pixel being selected.

5.4.1 Major Error Case

In the major error case, there is virtually no correspondence between the template and the target region over which the normalised correlation is being performed. It is thus preferable to persuade the filter to ignore that measurement. As discussed by Calvagno et al. [55] and from experimental testing, this can be done by increasing the measurement covariance value for each feature point by some large scale factor; experiments have shown that anything above 6 orders of magnitude works well.

Detection of diverged or occluded features is based on the result of the normalised correlation using Equation (5.5). In the small error case, a step type function is used in choosing the associated covariance value from the correlation. This is discussed in Section 5.4.2 below. From experimentation, a correlation value of $\rho < 0.7$ typically represents a template window that has diverged or has become partially occluded. This is deemed an unreliable measurement. Consequently the covariance on the pixel measurements for that template are assigned a very high value, as discussed above.

5.4.2 Minor Error Case

In the minor error case, the selection of measurement covariance again relies on the normalised correlation between the template and the target region during the template matching process. A step function, based on a minimum standard deviation of one pixel, is used to assign a covariance based on the correlation result. For a certain correlation range (typically a range of 0.5 between stepping points) a fixed covariance is assigned. The choice of range boundaries and value of assigned covariances were selected after iterative adjustments during an implementation calibration phase. The assigned covariance can be calculated from the standard deviation (in pixels) assigned to each correlation range. These standard deviations can be read from Table 5.1. Each normalised correlation range is given in the column labeled ρ and the relative standard deviation (in pixels) given in the corresponding row of the column labeled σ. The covariance is then calculated as $R = \sigma^2$.

Table 5.1: Table of normalised correlation ranges with associated standard deviation.

ρ	σ
$0.95 < 1.0$	1
$0.9 < 0.95$	2
$0.85 < 0.9$	4
$0.8 < 0.85$	8
$0.75 < 0.8$	16
$0.7 < 0.75$	24

As an example, assume that the normalised correlation result is $\rho = 0.87$. From the table, this is a standard deviation of 4 pixels. It must be noted here, however, that all measurements are at the image plane and not on the camera CCD. A conversion using the known intrinsic parameters of the camera is required. As stated in Section 3.7.4 of Chapter 3, these intrinsic parameters are assumed to be calibrated, and can often be found with the manufacturers specifications. For a standard deviation of 4 pixels on a CCD with

a relative pixel size of:

$$x_{PIX} = 0.0074\ mm$$

the assigned measurement covariance is:

$$R = (\sigma * x_{PIX})^2 = 8.7616 * 10^{-4}\ mm^2$$

5.5 Controlling Tracking Using Structure

An important and very useful capability of the Structure from Motion solution lies in the ability to constrain 2D feature tracking. As a means of establishing how this is achieved, it is useful to look at a typical patch tracking scenario. For simplicity one may consider the center of a template to represent a single feature point. The template is to be aligned with a corresponding target region using some alignment algorithm.

Assuming the search area is currently set around the optimal estimate of the track from the previous frame, a template tracking system can experience serious stability problems, as explained in Section 4.5. In the next frame, the template patch may undergo a level of occlusion, illumination change or an orientation change which results in a better match than the correct position. This will result in an incorrect starting position in the next frame. This may cause the 2D tracker to diverge, moving around randomly and possibly never find the feature point again other than by pure chance.

A useful way of solving this problem is to track several feature points on the same rigid object, and provide a confidence value to each of the feature matches. For a feature point with a low confidence value, rather than using a flawed initial position in the next frame, the two-dimensional starting position can be estimated from the known three-dimensional structure.

Within the SfM solution discussed in Chapter 3, such structural knowledge is available. Confidence values can be attached to each feature using the normalised correlation evaluation in Section 5.4. With a good estimate of the three-dimensional structure of the object and its motion relative to the camera, it is possible to correct any bad feature estimates.

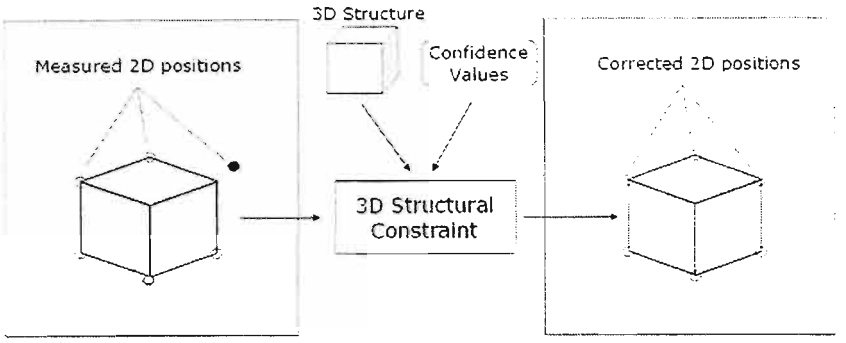


Figure 5.9: Badly matched two-dimensional features can be constrained using knowledge of the three-dimensional structure of the object in the scene. Bad (darker coloured) features are corrected to the two-dimensional projections of the structure.

This is done by projecting the known three-dimensional location of the points onto the image plane, as shown in Figure 5.9. The constrained two-dimensional estimates can then be used as the starting points for the 2D tracker in the next frame. This can be seen as a form of feedback, contributing towards a more stable tracking system.

This form of feedback control can be taken one step further. Instead of merely providing the 2D tracking algorithm with an estimate of the location of a single center point for each template in the next frame, the full motion of the tracking template can be constrained. In this thesis, four corners are used on each template window to infer structural knowledge of the head being tracked. When a template receives low confidence and needs to be reset, a method of estimating the initial affine motion of the template in the following frame can be derived using the projected locations of the structure points. This is addressed in Section 5.5.1.

5.5.1 Affine Parameter Estimates

In order to use the structural control described in Section 5.5, it is necessary to generate an estimate of the affine motion parameters of the alignment algorithm in the next frame, based on image plane projections of the known structural points. The projection of the structure points onto the image plane is performed using Equation (3.45), which

is the same projection equation used in the measurement transfer model of the EKF in Chapter 3. This is a combination of the camera to image projection Equation (3.3), the structure representation in Equation (3.5) and the reference frame transformation in Equation (3.22).

A note should be made at this point about the nature of projections onto the image plane. All feature measurements in the image plane are in the same units of measurement as those in the camera frame. In this thesis, units of millimeters (mm) have been assumed. This differs from coordinates on the camera CCD, which are in pixel units. Thus, in working with coordinates in both frames, a conversion must be performed on pixel measurements to convert them to image plane coordinates. Again, this is done using the known intrinsic parameters of the camera, namely the camera's center of projection (COP) and the relative pixel size on the CCD. A conversion is then written as:

$$\begin{pmatrix} u_{im} \\ v_{im} \end{pmatrix} = \begin{pmatrix} (u_{CCD} - COP_x)\, x_{PIX} \\ (COP_y - v_{CCD})\, y_{PIX} \end{pmatrix} \tag{5.6}$$

which equates image coordinates $(u_{im}, v_{im})^T$ to their pixel coordinates $(u_{CCD}, v_{CCD})^T$ on the CCD. The point (COP_x, COP_y) represents the intersection between the principle axis and the CCD. The values x_{PIX} and y_{PIX} represent the relative pixel size on the CCD in the x and y directions.

To generate an estimate of the affine parameters $\mathbf{p}= (p_1, p_2, p_3, p_4, p_5, p_6)^T$ for a template in the next frame, it is necessary to reorganise the affine transformation of Equation (4.22). First, three initial positions of the template corners on initialsation of the system are identified as:

$$\mathbf{x}^0 = \begin{pmatrix} x_1^0 & y_1^0 & 1 \\ x_2^0 & y_2^0 & 1 \\ x_3^0 & y_3^0 & 1 \end{pmatrix} \tag{5.7}$$

Considering that an affine transformation generates a parallelogram, three corners are

sufficient to fully describe the shape and structure of the (affinely) deformed template region. Three corners generate two lines on adjacent sides of the parallelogram. The length of these sides encode the structure, with the angle between them defining the shape.

After transformation, the corresponding three projected structure points can be written as:

$$\mathbf{x} = \begin{pmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ x_3 & y_3 & 1 \end{pmatrix} \tag{5.8}$$

This transformation of the initial corners via the affine transformation matrix A can be written as:

$$\mathbf{x} = \mathbf{x}^0 A^T \tag{5.9}$$

where:

$$A = \begin{pmatrix} (1+p_1) & p_3 & p_5 \\ p_2 & (1+p_4) & p_6 \end{pmatrix} \tag{5.10}$$

Trivially then, the estimated affine transformation can be found as:

$$A = (\mathbf{x}^0)^{-T} \mathbf{x}^T \tag{5.11}$$

This then constitutes the initial motion estimate given to the 2D tracking algorithm on starting the alignment for the next frame. This method of estimating the affine motion is used whenever the template matching in each frame is sufficiently flawed that it warrants correction.

5.6 System Design Conclusions

In this chapter, the implementation details of the final head tracking system have been reviewed, focusing on system integration and feedback. In order to speed up filter convergence, structural estimates from a wireframe model of the face have been used. Only once structural convergence has been achieved is accurate pose estimation possible. Thus any filter bootstrapping is advantageous for improving stability and speed of convergence. Other initial state estimates and noise statistics mirror those selected during the synthetic experimentation presented in Section 3.7 of Chapter 3. The integration of the 2D tracking algorithm and the EKF includes a reliability measure on pixel coordinates being fed to the EKF during state estimation. In this way, good measurements are made to contribute towards state estimation more than bad ones. Further, feedback benefits of a Structure from Motion solution include the ability to constrain feature tracking in each frame. This further contributes to system stability and robustness over a range of possible divergence scenarios.

In Chapter 6 results for real video sequences are presented. These include evaluation of the recovery of pose estimates with ground truth validation in Section 6.3, the ability of the tracker to cope with partial occlusions in Section 6.1 and the recovery of head structure and inverse focal length in Section 6.2. Also important to these discussions is tracking failure. Section 6.4 highlights typical scenarios when this occurs.

Chapter 6

System Evaluation

In this chapter the performance of the tracking system is evaluated using real video sequences. All sections include evaluation on sequences captured at a resolution of $(640, 480)$ using Point Grey research [103] Dragonfly cameras at a framerate of 30 fps. Section 6.1 assesses the robustness of the tracking system to partial occlusions, facilitated by the changing measurement covariance associated with the pixel measurements from each template. Section 6.2 looks at the convergence of object structure α and the camera parameter β. Convergence of these parameters is vital to system accuracy and stability. Section 6.4 concludes the chapter with an assessment on the limits of the tracking system which cause tracking to fail.

Section 6.3 presents two sets of video sequences that evaluate different aspects of the pose estimation problem. The first set of sequences (using the Point Grey [103] Dragonfly cameras) are intended to convey the scope of pose recovery and ability to deal with unusual visual circumstances. The second set of sequences provide a means of validating the accuracy of the head tracking results. In many works, [27, 28, 49, 29, 30], magnetic orientation trackers are used to validate the results of non-intrusive visual pose estimation systems.

The ground truth video sequences and magnetic tracking results were originally captured and used for analysis in the work of Cascia et al.[27]. All sequences are now publicly

available from [104]. The test video sequences were collected using a Sony Handycam on a tripod. Ground truth was collected via the "Flock of Birds" 3D magnetic tracker [105]. The images in the video signal were digitized at 30 fps at a resolution of $(320, 240)$.

To collect ground truth of the position and orientation of the head, the transmitter of the magnetic tracker is attached to the subject's head. The "Flock of Birds" [105] system measures the relative position of the transmitter with respect to the receiver (in inches) and the orientation (in Euler angles) of the transmitter. The magnetic tracker, in an environment devoid of large metal objects and electromagnetic frequencies, has a positional accuracy of 0.1 inches and angular accuracy of 0.5 degrees [27]. Both accuracies are averaged over the translational range. Typically, accuracy falls in a laboratory environment where metal furniture and computers can interfere with the signal. However, as stated in [27], the captured measurements are still sufficiently accurate to evaluate a visual tracker.

Measured ground truth data and visual tracker estimates are expressed in two different coordinate systems. The visually estimated position is in a coordinate system with origin in the camera frame and is known only up to the previously mentioned scale factor in Section 3.4.5. Cascia et al.[27] consider this as an absolute orientation problem [106], where there are two sets of measurements expressed in two coordinate systems with different position, orientation, and units. To avoid this problem, when capturing the sequences, the magnetic receiver and the camera were carefully aligned such that the two coordinate systems were parallel, see Figure 6.1. The scale factor in the three axis directions was then estimated using calibration sequences. All visual tracker estimates in [27] were then transformed according to these scale factors before comparison with ground truth data.

In comparing the ground truth with the visually estimated position and orientation, it is also assumed in [27] that the visual estimate is coincident with the ground truth at the first frame of the sequence. The ground truth validation graphs of Section 6.3.3 are also based on this assumption.

In the tracking sequences presented in Section 6.3, the wireframe head model Candide [1] is warped and projected into the scene. This is provided as a visually intuitive means of evaluating results. As shown, the model correctly synthesises the motion of the target

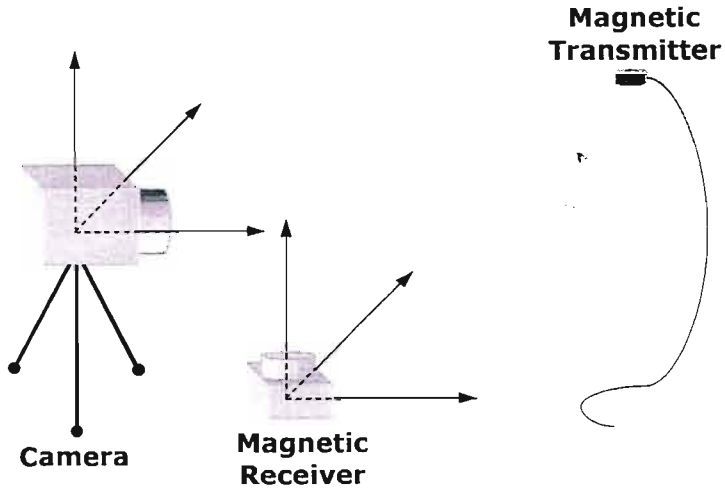


Figure 6.1: Camera and magnetic tracker coordinate systems.

head. In so doing, the motion estimates achieve the ultimate goal of rigidly animating a head model (i.e. global motion) in a model-based face coding system. Further tracking sequences are presented in Appendix D.

All 2D tracking and Structure from Motion aspects of the system have been implemented in C++. Video sequences, captured at 30 *fps*, are handled as discrete images using the image processing package CImg [107]. Projection of the wireframe head model to calculate initial structure estimates is performed offline in MATLAB. The Candide head model [1] is then warped and projected back into the sequence (again using MATLAB) once the motion parameters for the sequences have been estimated.

6.1 Robustness to Occlusion

Occlusion handling is an important aspect of any tracking system and it is elegantly managed within the Kalman filter framework through a changing measurement covariance R_k, discussed in Sections 5.4 and 5.5. If a feature becomes occluded, several things happen within the tracking system.

1. The 2D feature tracker will not be able to recover the motion of the template as the

target region is no longer visible.

2. A bad match is declared and a very high covariance is assigned to all measurements associated with that template.

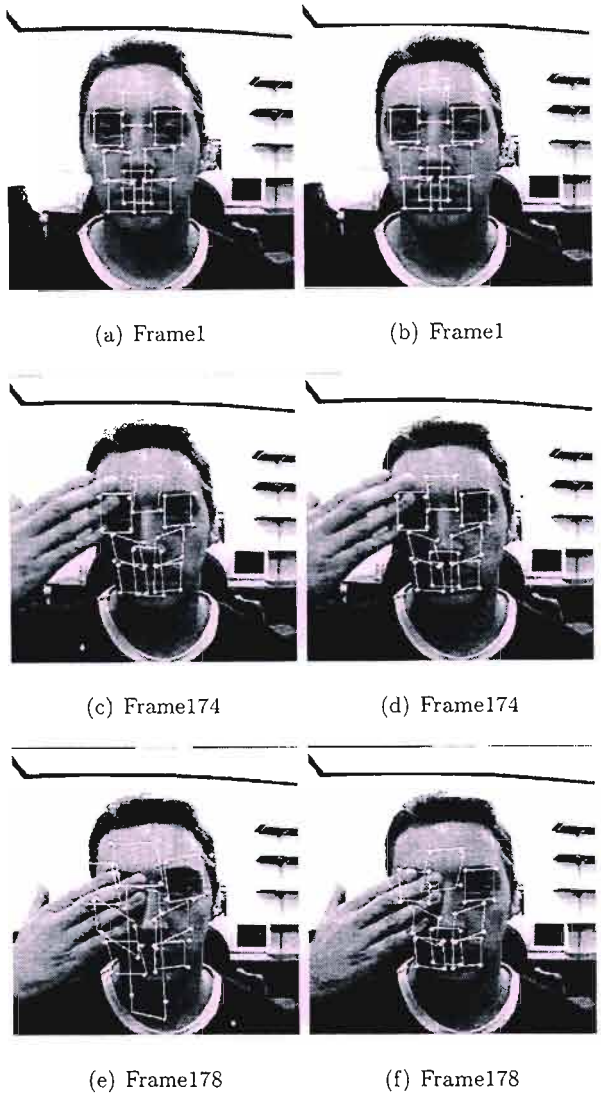
3. The measurement is ignored in the EKF estimation process.

4. The affine motion of the tracking window is reset using structural knowledge of the object being tracked.

Poor measurements are still fed along with good ones into the filter framework, but the high variance on these measurements results in them being disregarded for that *a posteriori* (measurement) update step. Thereafter, starting positions in the 2D feature tracker are constrained to projections of the estimated structure. This increases the likelihood that *lost* template windows will regain track of their associated target regions when the object causing the occlusions leaves the scene.

Figure 6.2 depicts a typical occlusion scenario. The images shown depict the boxed target regions with markers on the corners representing the structure points of each template. In the first column of Figure 6.2 (frames 1 – 178), the covariance R_k for all arriving measurements is fixed. Consequently, when target regions become occluded in frame 174 (and the 2D tracker cannot perform an alignment), the Kalman filter continues to believe that arriving measurements are reliable and state estimates become corrupted, resulting in tracking failure by frame 178. This demonstrates how quickly tracking can fail (just 4 frames) and typically a re-initialisation phase would take over at this stage. This is discussed Section 6.4.

By utilising a dynamically varying covariance R_k associated with each measurement being fed to the Kalman filter, the proposed tracking system does not fail when such partial occlusions occur. This is demonstrated in the second column of Figure 6.2, where structure points remain fixed to the head.



(a) Frame1 (b) Frame1

(c) Frame174 (d) Frame174

(e) Frame178 (f) Frame178

Figure 6.2: Occlusion sequences. In $(a), (c)$ *and* (e), the measurement covariance R_k is kept constant and the system diverges when occlusions occur. In $(b), (d)$ *and* (f), the covariance R_k is dynamically varying, so the system maintains stability.

The potential to constrain the 2D tracking process using the known object structure is also demonstrated. Despite the total occlusion of the target region for some templates, the window regions remain closely bound to their respective target regions which are obscured by the occluding object, in this case a hand. Figure 6.2 uses a frontal sequence for demonstration purposes. A similar level of occlusion immunity exists when the head is

moving as well. The system maintains tracking and does not exhibit instability unless very jerky motion is used or extreme out-of-plane rotations are observed. This is confirmed when out-of-plane rotations cause features to become occluded in the pose estimation sequences of Section 6.3. Again, poor measurements from hidden features are ignored until they return into view. The system has been tested on multiple subjects and tracking performance remains consistent.

The sequence in Figure 6.2 provides a good representation of how the system works under occlusion. This method of reducing sensitivty to partial occlusions was originally proposed in [22]. Authors in [20] and [55] have also implemented similar tracking systems to that proposed in this thesis which take advantage of this capability within the Kalman filter framework. Understandably, tracking can still fail if too many points are occluded. Section 6.4 addresses this issue.

6.2 Structure and Camera Convergence

In this section, convergence of the camera parameter β, and structure α_i is reviewed. These parameters are integral to the accuracy and stability of the tracking system. Firstly, the convergence rate of both the camera and structure directly influences how soon accurate pose estimates can be expected. Only once these parameters settle is a consistent scale set for the recovered translation parameters. Secondly, the use of structure to constrain the template tracking algorithm is unreliable until structural convergence occurs.

In Figure 6.3, initial depth estimates are calculated using a projected version of the Candide head model [1] and the ray-plane intersection method described in Section 5.3.1. As these estimates are closer to the optimal structure than the estimates of the initial condition assumed in the synthetic data testing (i.e. that all structure points lie in a common plane), structural convergence occurs soon after tracking begins. Optimal structure in this case refers to the recoverable structure up to some scale, as discussed in Section 3.4.5. The converged structure parameters shown in Figure 6.3 assume that the variance on one of the initial depth estimates from the face model is set to zero, and all other structure

states in the Kalman filter then scale themselves around this value. Structural convergence typically occurs within 100 frames as illustrated in Figure 6.3. Thus for a video sequence captured at 30 fps, reliable tracking is achieved in under 4s.

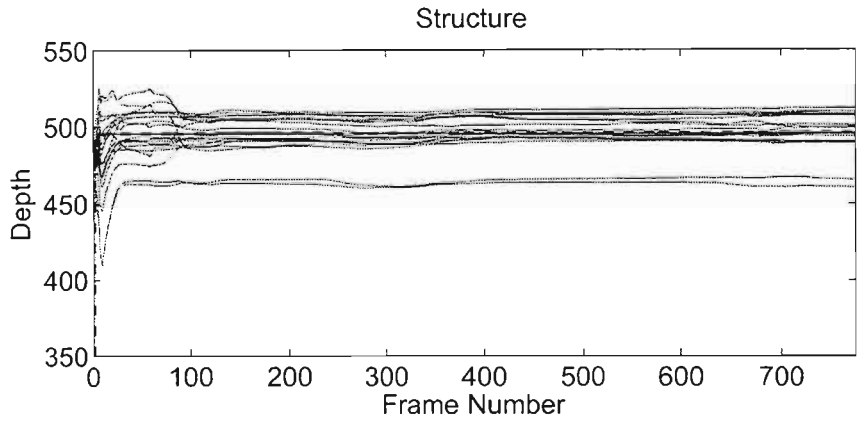


Figure 6.3: Convergence of the structure parameters α.

Because the face region is roughly planar, two benefits arise from the use of initial structure estimates from the wireframe face model, viz. faster convergence and the introduction of a convexity bias which lowers the probability of depth reversal occuring. The convexity constraint is mentioned in the experiments performed by Azarbayejani and Pentland in [28] and also in the synthetic experiments of Section 3.7.4.

The camera parameter β is initialised from the manufacturers specification of the lens used. Figure 6.4 depicts the estimation result of a typical tracking sequence. When tracking begins, a low initial error covariance P_{0_β} causes the system to momentarily diverge from the known initial estimate. Along with the converging structure however, the camera parameter soon returns to a level very close to the known initial estimate of $\beta = 0.2$ in under 100 frames. The estimation of the known camera parameter merely serves to confirm the ability of the SfM algorithm to recover camera geometry when it is not known with such certainty, as shown in the synthetic experiments of Section 3.7.4.

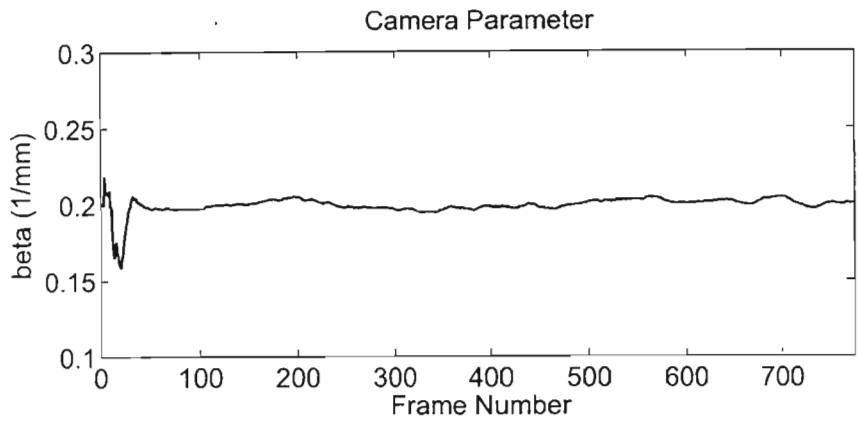


Figure 6.4: Convergence of the camera parameter β.

6.3 Pose Estimation

The primary goal of this thesis is the recovery of pose estimates that can be used to rigidly animate a model of the head. In the process of achieving this goal, a number of other mechanisms are harnessed to improve robustness and stability, as well as to contribute towards the accuracy of the estimated pose parameters. An essential one of these is object structure.

The initial estimates from the head model are intended purely as a bootstrapping mechanism. Object structure initially relies on these initial estimates from the projected face model, with convergence being achieved within a short period after tracking has begun. The recovered pose parameters improve in accuracy after structural convergence occurs.

In the sections which follow, two different sets of testing sequences are presented. The sequences in Figures 6.5 and 6.8 of Section 6.3.1 were captured using our own video capture equipment. Because there is no way of quantitatively validating the accuracy of the pose estimates achieved for these sequences, separate ground truth sequences [104] with independently measured motion trajectories are used for evaluation in Section 6.3.3. Despite the lower resolution and quality of the ground truth sequences (as compared with our own video), pose estimation remains stable and accurate. Results are presented in

Sections 6.3.1 and 6.3.3 below.

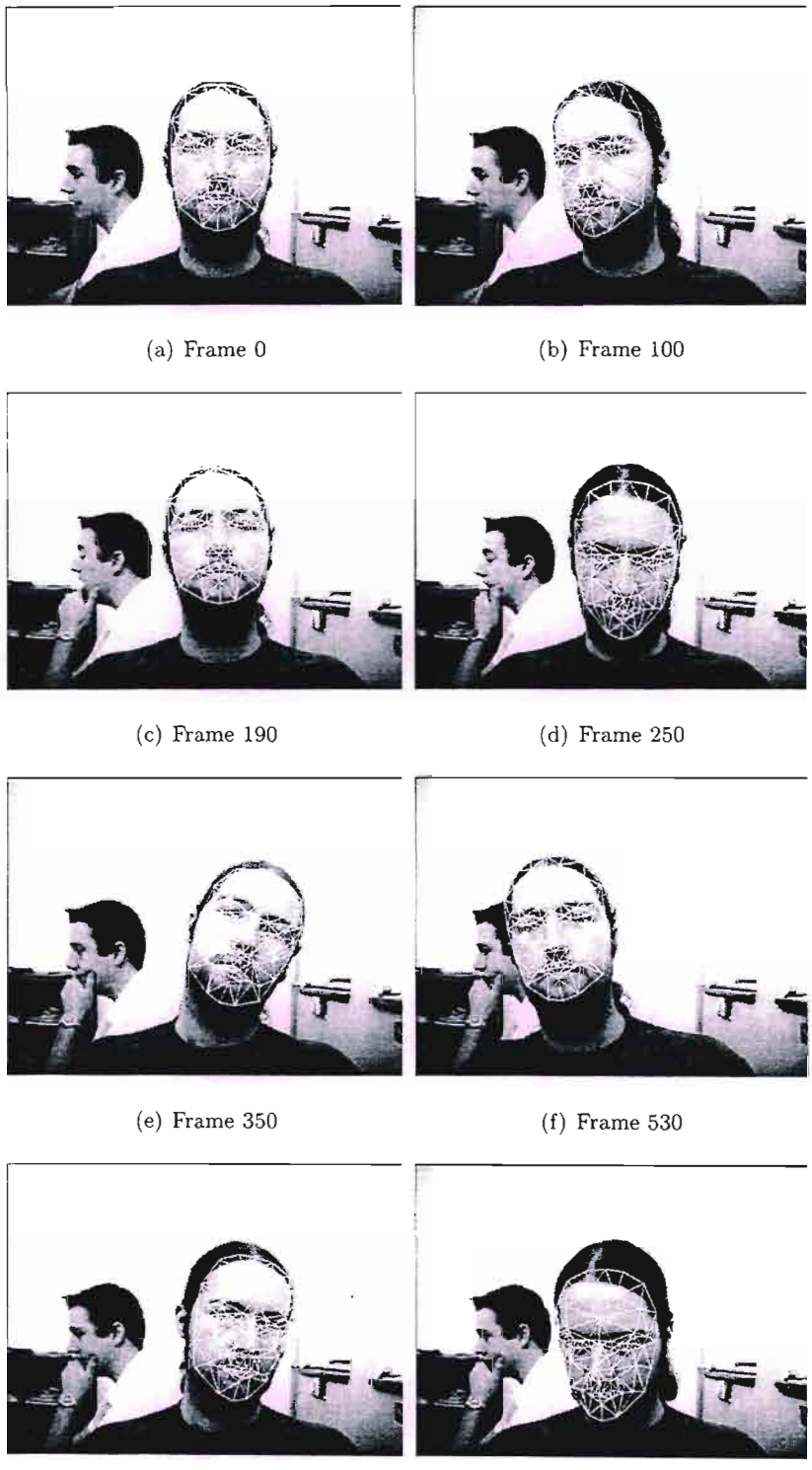
6.3.1 Example Sequences

In this section, video sequences of approximately 25 seconds in length were captured with a resolution of $(640, 480)$ at 30 fps. For the sake of neatness, only two of three example sequences are presented here, demonstrating the recovery of Pitch, Yaw, Roll and large translations. Further results are provided in Appendix D for the interested reader. This sequence demonstrates the ability of the tracker to cope with structural irregularities; for example when the user is wearing glasses. The fact that fairly long sequences have been used is intended to confirm the suitability of the approach as an online solution, with tracking remaining stable and accurate throughout.

For both the sequences in this section and that in Appendix D, results for translation, quaternion and Euler angle estimates are plotted. The Euler results are converted from the unit quaternion and are provided purely for ease of interpretation. The global rotation matrix is still constructed using the estimated unit quaternion, as explained in Section 3.4.3. Also included are frames from each sequence with the Candide [1] face model projected into the scene, appropriately warped using the estimated motion parameters. The outcomes of these results are discussed in Section 6.3.2.

Rotation Sequence

The example frames taken from a typical rotation sequence in Figure 6.5 exhibit a considerable amount of Pitch, Yaw and Roll motion. As depicted in the example frames, the motion of the projected head model remains coherent with the actual head throughout the sequence. Estimated pose parameters and the rotation quaternion are shown in Figure 6.6 and Figure 6.7 respectively.



(a) Frame 0 (b) Frame 100

(c) Frame 190 (d) Frame 250

(e) Frame 350 (f) Frame 530

(g) Frame 720 (h) Frame 750

Figure 6.5: Sequence demonstrating tracking over a wide range of rotations.

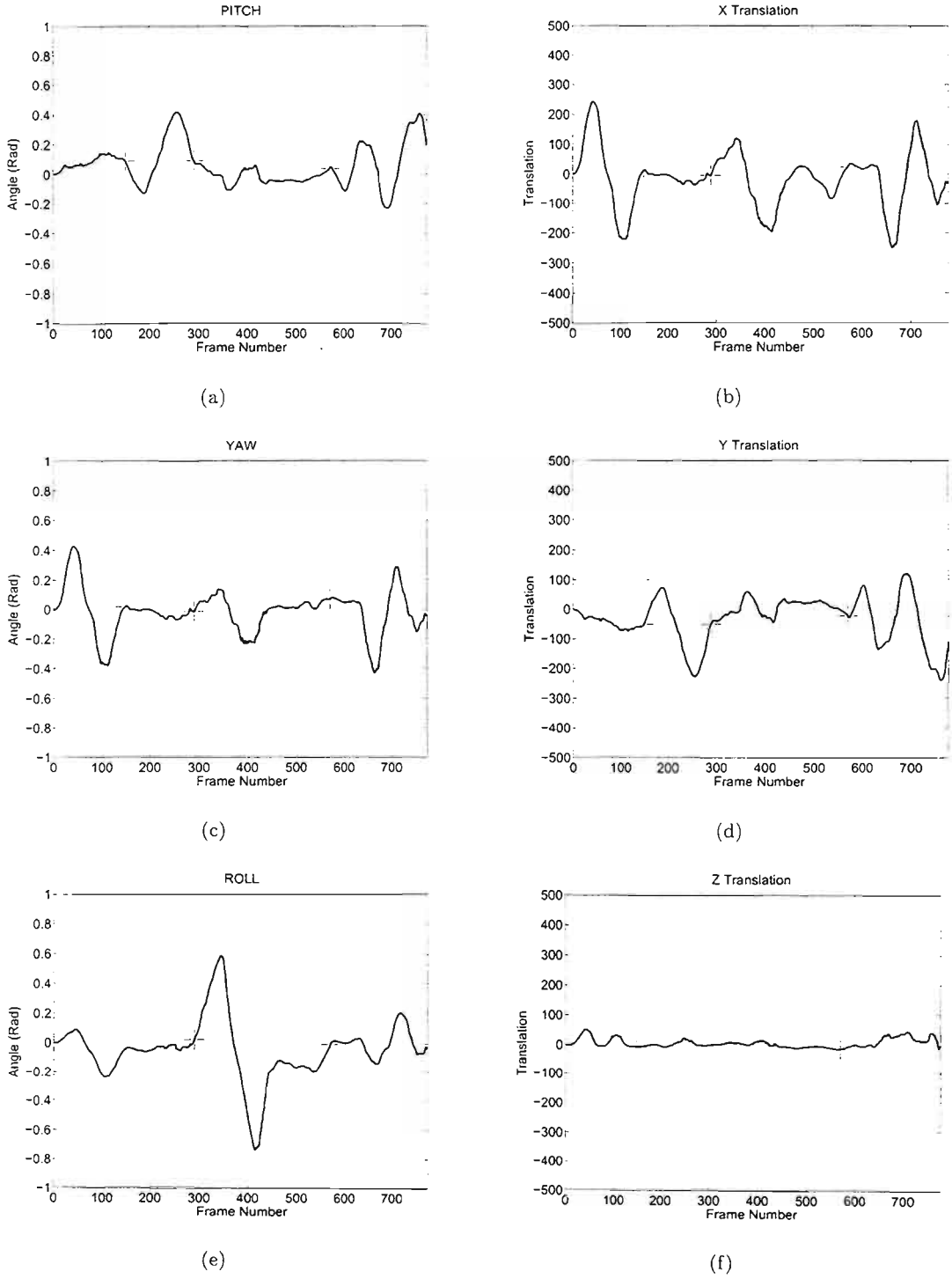


Figure 6.6: Pose estimates for the rotation sequence.

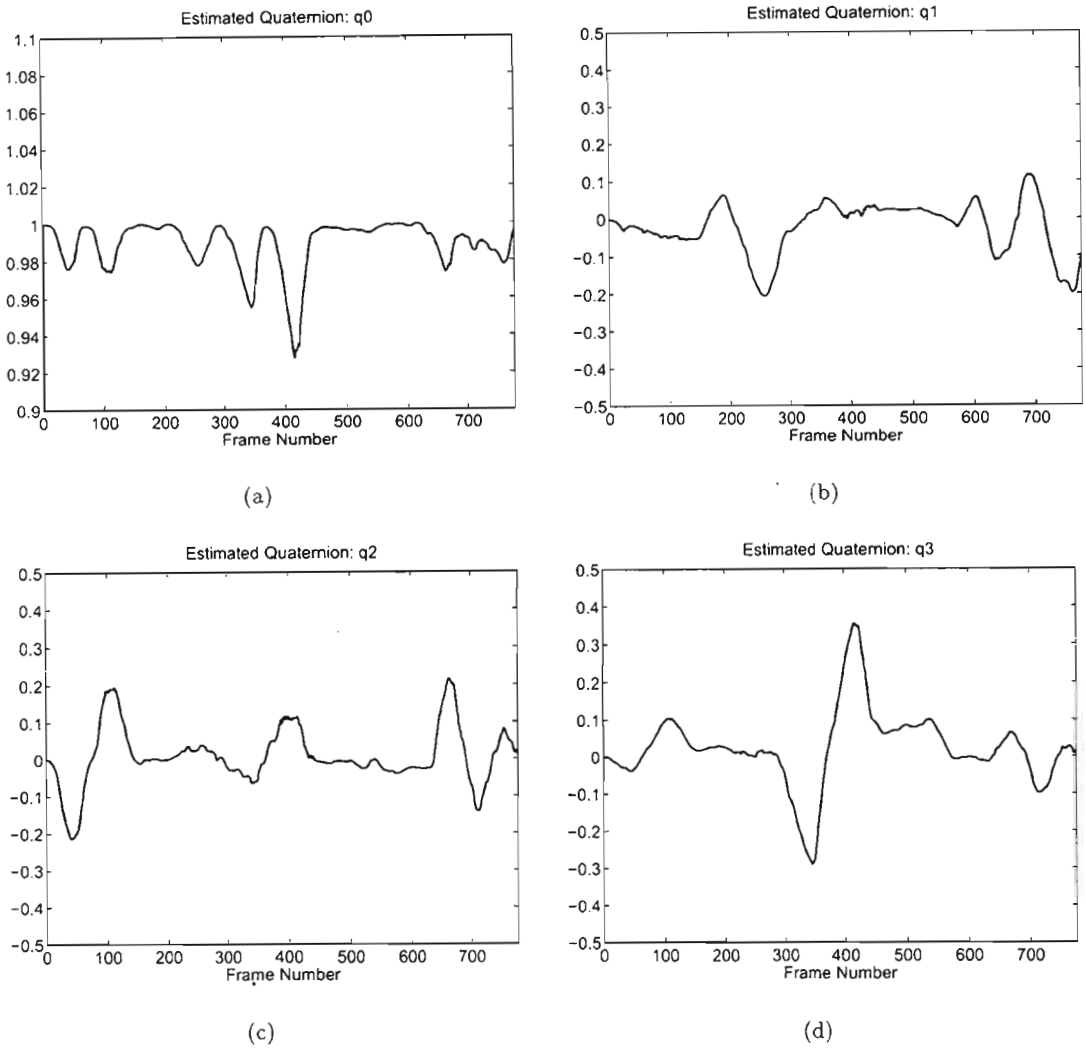
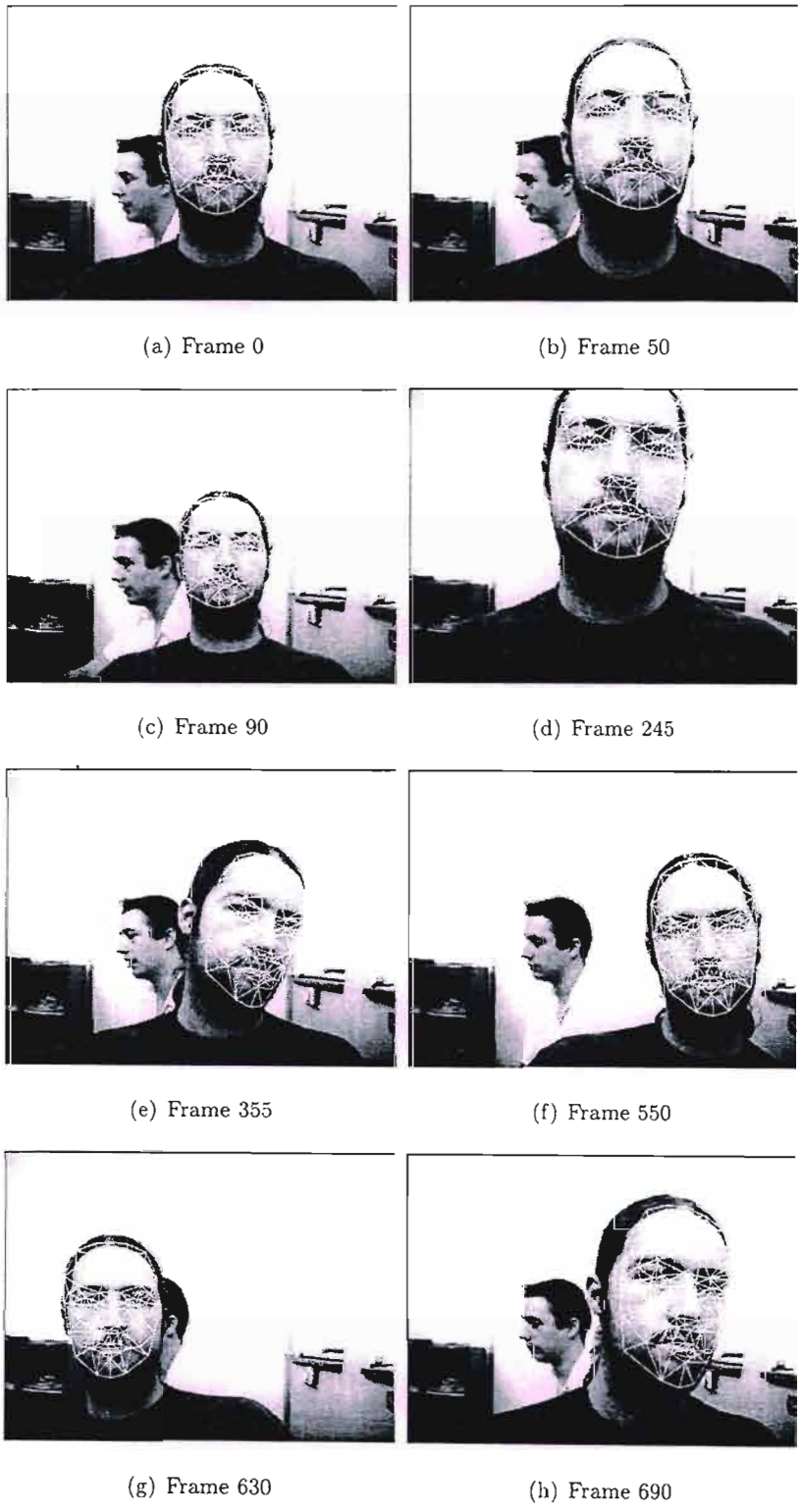


Figure 6.7: Quaternion estimates for the rotation sequence.

Translation Sequence

Figure 6.8 provides example frames from a sequence exhibiting large translations in each of the (X,Y,Z) cartesian directions. The recovery of large translations is demonstrated, especially the illusive depth parameter T_z, which is often difficult to recover in single camera systems. As with rotation, the motion of the head model remains coherent with the real head. Estimated pose parameters and the rotation quaternion are shown in Figure 6.9 and Figure 6.10 respectively.



(a) Frame 0 (b) Frame 50

(c) Frame 90 (d) Frame 245

(e) Frame 355 (f) Frame 550

(g) Frame 630 (h) Frame 690

Figure 6.8: Sequence demonstrating tracking over a wide range of translations.

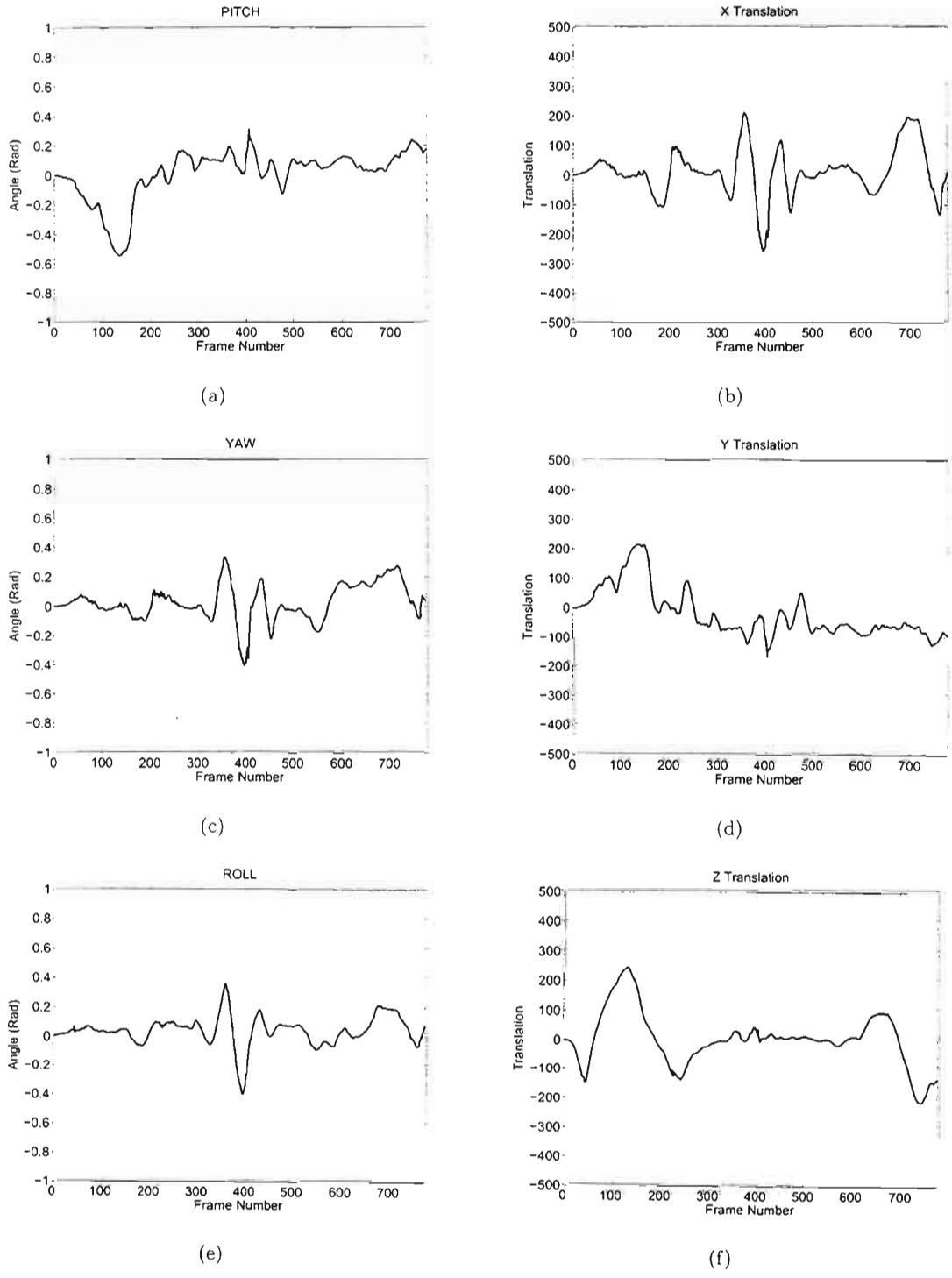


Figure 6.9: Motion estimates for the translation sequence.

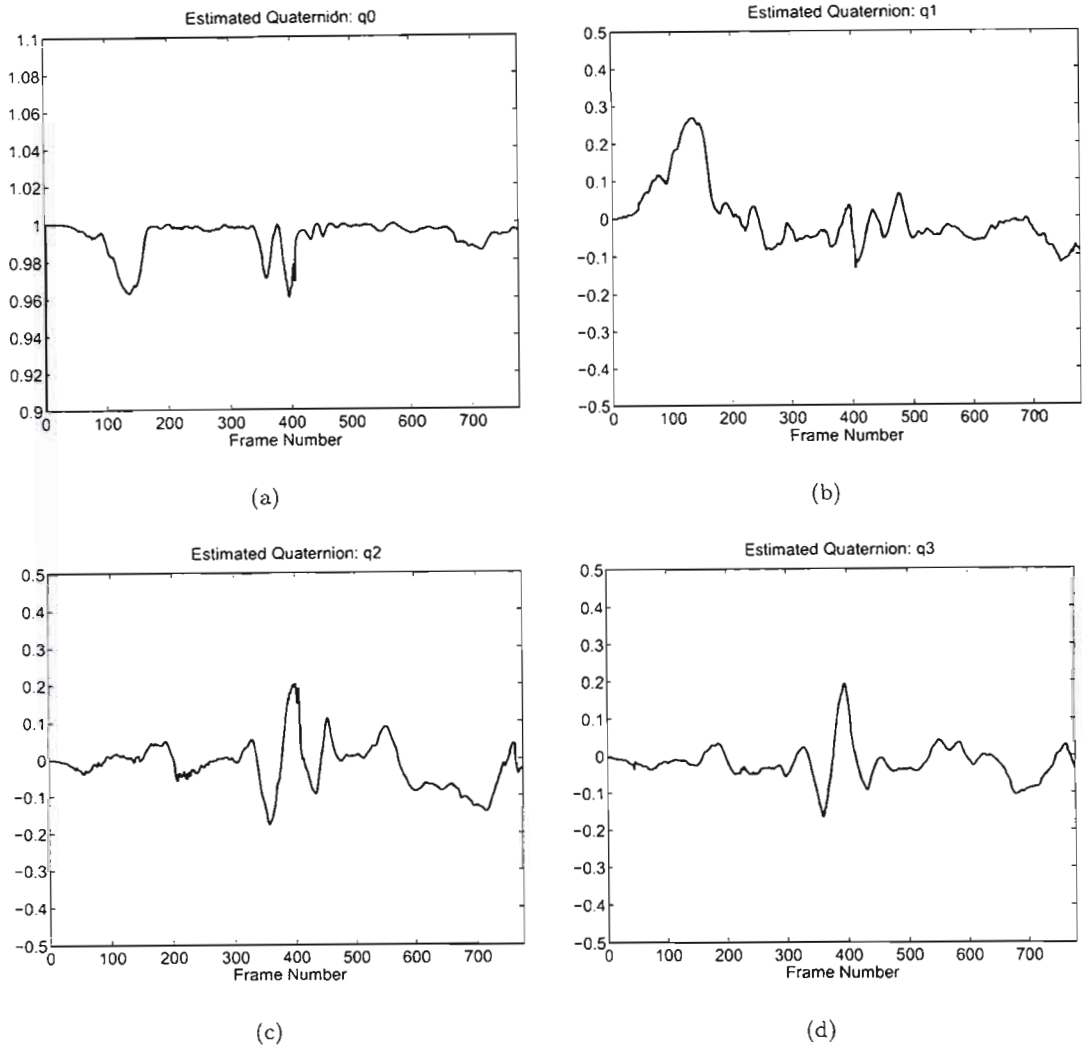


Figure 6.10: Quaternion estimates for the translation sequence.

6.3.2 Example Sequences Discussion

Before the ground truth validation presented in Section 6.3.3, the accuracy of results presented thus far can be discussed with reference to the consistency of the estimated motion and the accumulation (or lack thereof) of errors. A good way of reviewing the stability is to consider the situation when the pose estimates should return to a neutral, or initial-pose condition. If there were significant error accumulating in the system, the motion estimates would not have zeroed when the head returned to a frontal pose.

(a) Frame 150 (b) Frame 290 (c) Frame 570

Figure 6.11: Frames from the Rotation sequence where the pose is frontal.

For example, consider the frames shown in Figure 6.11, taken from the "Rotation" sequence. These frames are typical of frames where the head returns to an initial pose. Markers on the trajectories in Figure 6.6 illustrate how the 3D trajectories correctly return near to zero when the face is close to the initial frontal pose. This indicates that the system is correctly estimating the motion parameters, albeit to some scale factor for the translational component. A system that can consistently return to an initial pose is a stable system. Thus provided that sufficient measurements are available for pose estimation (thus preventing tracking failure, discussed in Section 6.4), the system should run indefinitely and consistently provide good motion estimates.

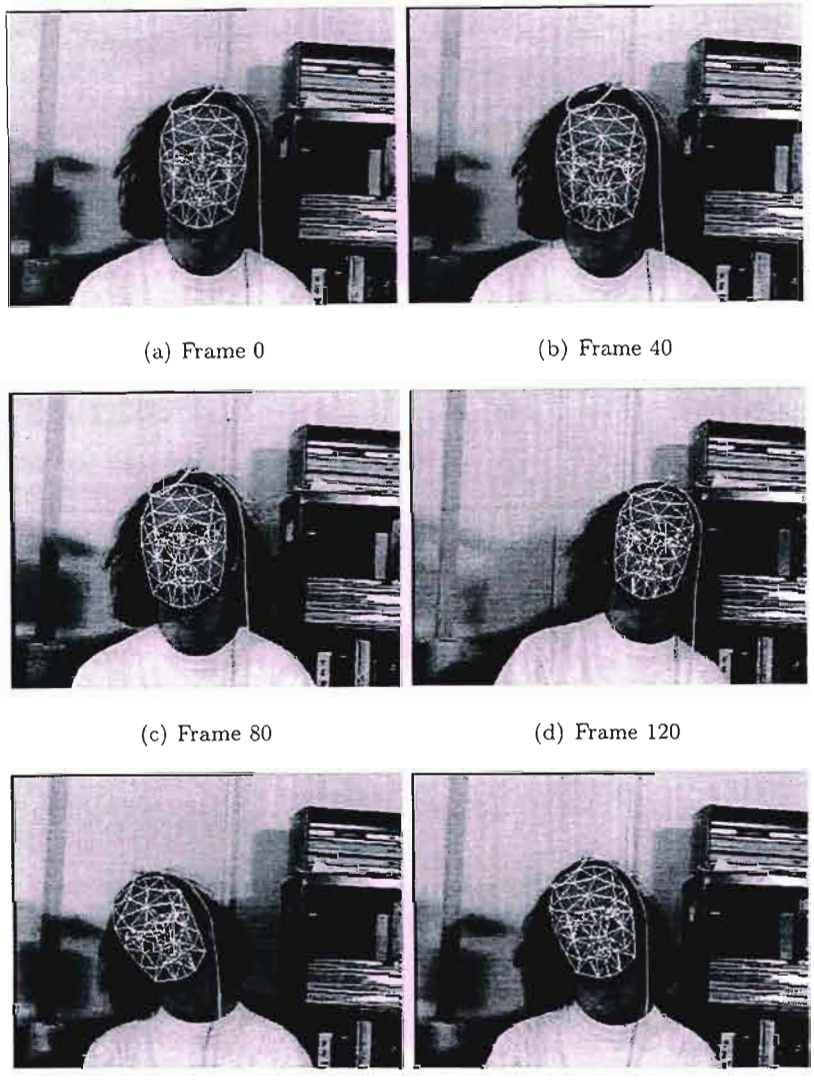
A further way of subjectively analysing the system is to compare the uniformity of motion displayed by the projected face model with the motion of the head in the video sequences. The principle objective of this work is to extract motion parameters from a video sequence that would sufficiently represent the 3D motion of a subjects head. These parameters could then be re-used as rigid animation parameters for some graphical avatar or more realistic head model representation that would be displayed to a human observer. For both the sequences in this section and those in Appendix D, the motion of the wireframe model is consistent with the *apparent* motion of the head in the sequence. Thus, from a visual perspective, the estimated motion parameters have met the requirements of a model-based coding system.

6.3.3 · Ground Truth Validation

The ground truth video sequences used to evaluate the accuracy of the head tracking system are publicly available from [104]. The beginning of this chapter introduces the "Flock of Birds" [105] magnetic tracking system, which measures the relative position and orientation of the transmitter with respect to the receiver. The units of measurement used in Section 6.3.1 are millimeters and radians for translation and rotation respectively, where the scale of the recovered translation is constrained via the fixed structure parameter α_0. The measurements returned by the magnetic tracker however, are in units of inches for translation and Euler angles (in degrees) for rotation. As a means of comparison against the ground truth and the head tracking results obtained in [27], the tracking results obtained using the SfM approach of this thesis are converted to the same units of measurement.

Re-iterating the assumption originally made in [27], comparing the ground truth with the visually estimated position and orientation assumes that the visual estimate is coincident with the ground truth in the first frame of the sequence. Thus, to bring the two reference frames into alignment a calibration process, approached as an *absolute orientation problem* [106], is performed. A discussion specific to using absolute orientation in estimating the transform between a magnetic and camera reference frame can be found in [108]. The results presented in Figure 6.13 reflect the use of this transformation.

Figure 6.13 presents tracking results plotted against one of the ground truth sequences orignially presented in [27]. Sample frames with the Candide head model projected into the scene are presented in Figure 6.12. The use of this specific ground truth sequence affords a means of comparing the tracking results of the head tracker in this thesis with those obtained by Cascia et al.[27]. The nominal values of rotation and translation achieved by Cascia et al. were acquired by reading between every 5 frames from the graphs presented in [27]. In the graphs of Figure 6.13, the thick noiseless line represents the pose estimate for each degree of freedom using the SfM solution. The dashed curve corresponds to the tracking results obtained in [27].



(a) Frame 0 (b) Frame 40

(c) Frame 80 (d) Frame 120

(e) Frame 160 (f) Frame 200

Figure 6.12: Ground truth tracking sequence.

As a brief note concerning the accuracy of the magnetic tracker, the solid jagged curve in Figure 6.13 indicates that the noise level in each of the the magnetic estimates is larger than the nominal accuracy of the magnetic tracker. Despite this, the accuracy of the ground truth results are sufficient to evaluate the pose estimates achievable using a visual tracking system. One can easily see that the tracking results presented in Figure 6.13 (using the SfM system) perform as well (if not better) against the ground truth than the tracking results obtained by the system of Cascia et al.[27]. Further comparison of ground

truth against acquired results is provided by the graphs and sample frames in Figures D.4 and D.5 of Appendix D. Section 6.3.4 presents a discussion on the range of recoverable pose using the proposed head tracking system.

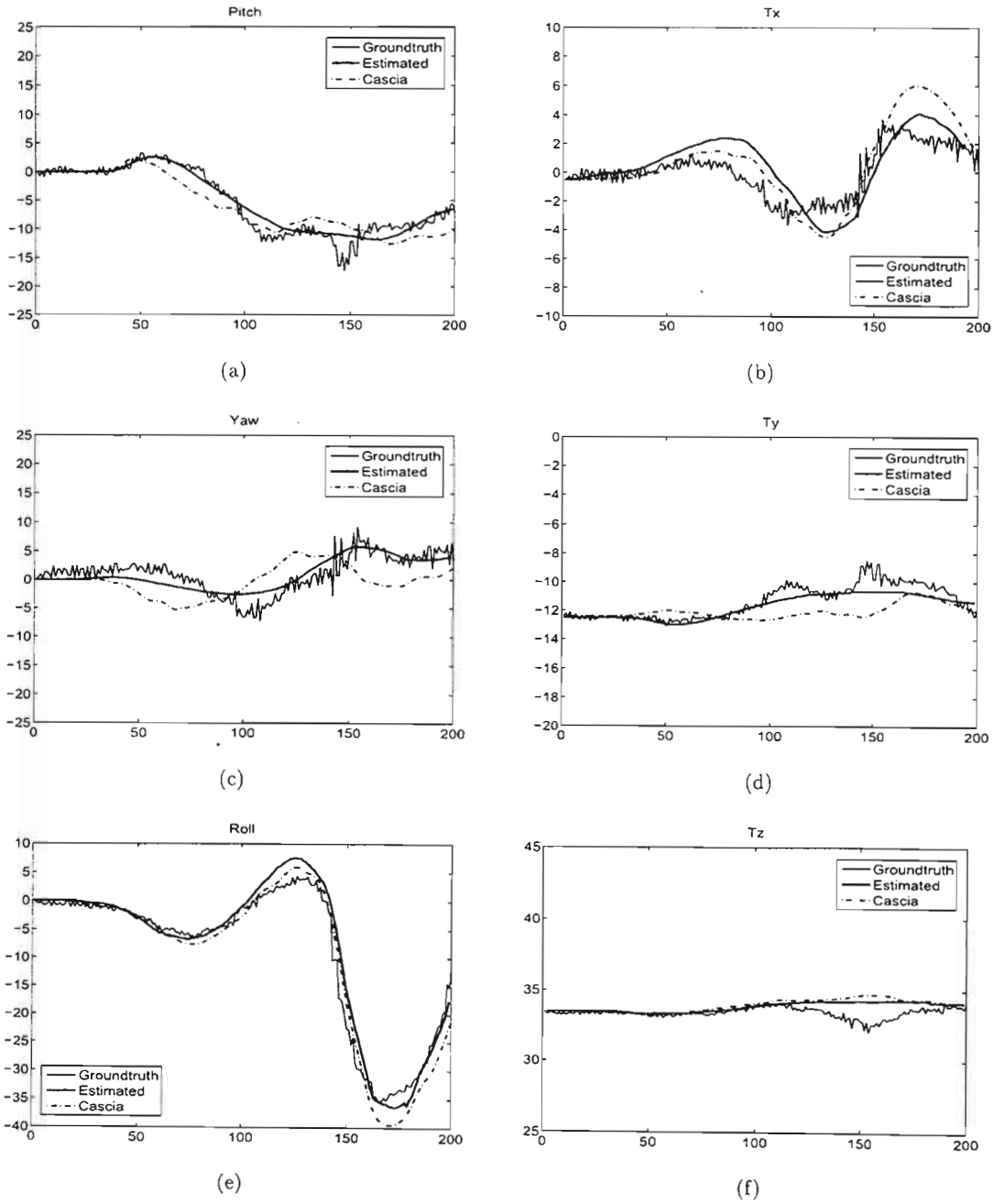


Figure 6.13: Ground truth sequence motion estimates.

6.3.4 Scope of Recovery

The results presented in Sections 6.3.1 and 6.3.3 confirm the ability of the tracking system to recover the required translational and rotational components of the SfM problem. The limits of pose estimation recovery were tested using various sequences that brought the system to a failure threshold, a topic expanded upon in Section 6.4. Assuming that range of recoverable motion is defined by the threshold when failure occurs, it was found that the system was able to handle up to approximately 45° rotation around the $x - axis$ (Pitch) and the same around $y - axis$ (Yaw). Thereafter, the system becomes unstable (loses lock) as most features have either changed in aspect beyond the span of the 2D tracker, or have become occluded. The rotational component around the $z - axis$ (Roll) does not suffer from the same matching issues, as the template regions do not have to accommodate any significant structural changes (due to the contours on the face) or perspective effects that occur on the face during a Yaw or a Pitch. The system has consequently been tested up to approximately 90° of pure Roll. Obviously, when combination rotations occur (such as Pitch and Yaw at the same time) these limits of recovery for each angular degree of freedom will fall somewhat, due to the combined structural changes of the (assumed planar) features on the face. Quantifying this deterioration in performance for all possible rotational combinations is a difficult task and will not be performed in this thesis. Neither is it evaluated in any of the head tracking systems presented in Chapter 2. It is only mentioned here to make the reader aware that the range of recovery presented ($Roll \sim 90°$, $Pitch \sim 45°$ and $Yaw \sim 45°$) defines the upper-bound on rotation for the proposed head tracking system.

6.4 Tracking Failure

Tracking failure can be considered as any situation where the system is no longer able to recover the pose of the head and becomes unstable. To interpret these situations, it is useful to first consider the times the tracking system *is* able to recover head pose, and then how it diverges from these circumstances. When tracking begins, one can consider the

tracker to be within some region of convergence (ROC), such that when the pose changes due to movement of the head in the scene, the tracker will then converge to the new pose of the head. This process will continue as long as the estimated pose from the previous frame is within the region of convergence of the current frame. The size of the ROC varies from frame to frame, mainly due to the availability of reliable feature measurements. This leads to the discussion on tracking failure.

Tracking failure can be broken down into three typical cases. The first and most obvious case is the case of *total* occlusion due to some foreign obstacle moving in front of the face. The convergence zone in this instance becomes almost zero. A typical tracking failure under these conditions is shown in Figure 6.14.

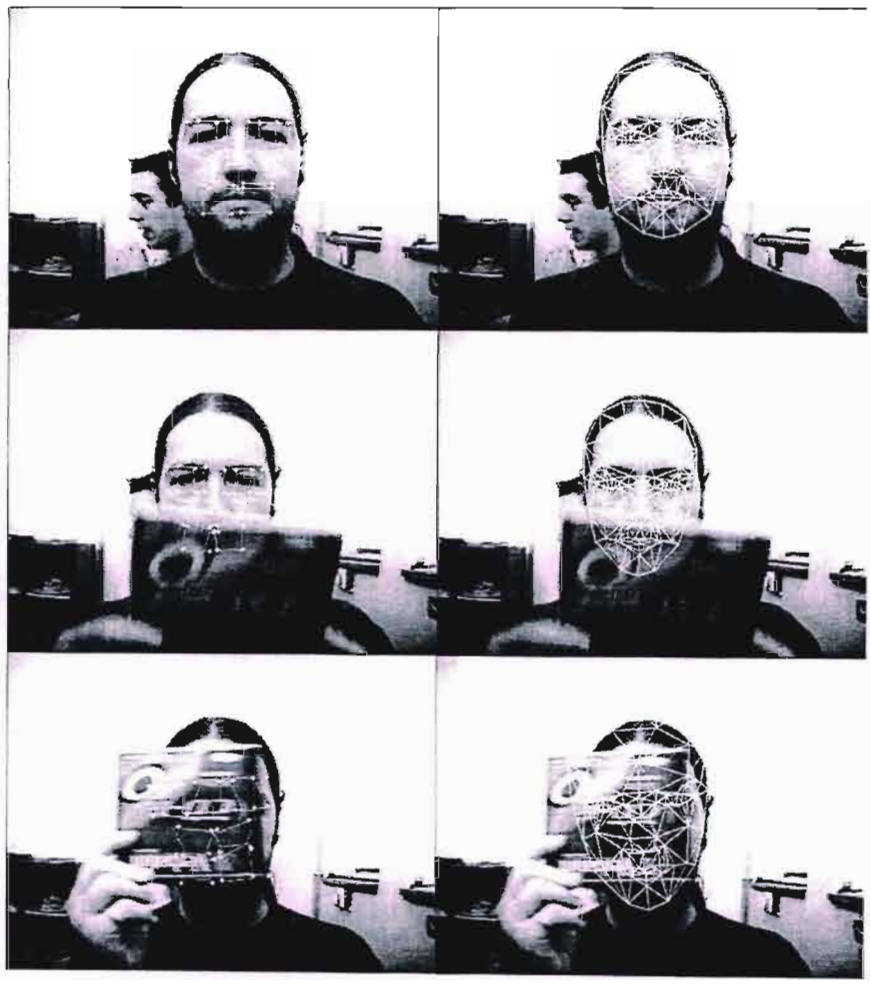


Figure 6.14: Tracking failure due to full occlusion.

The second case can occur when the head in the scene moves too quickly between frames. This results in the pose from the previous frame being outside of the convergence zone of the current frame, and tracking fails, as shown in Figure 6.15.

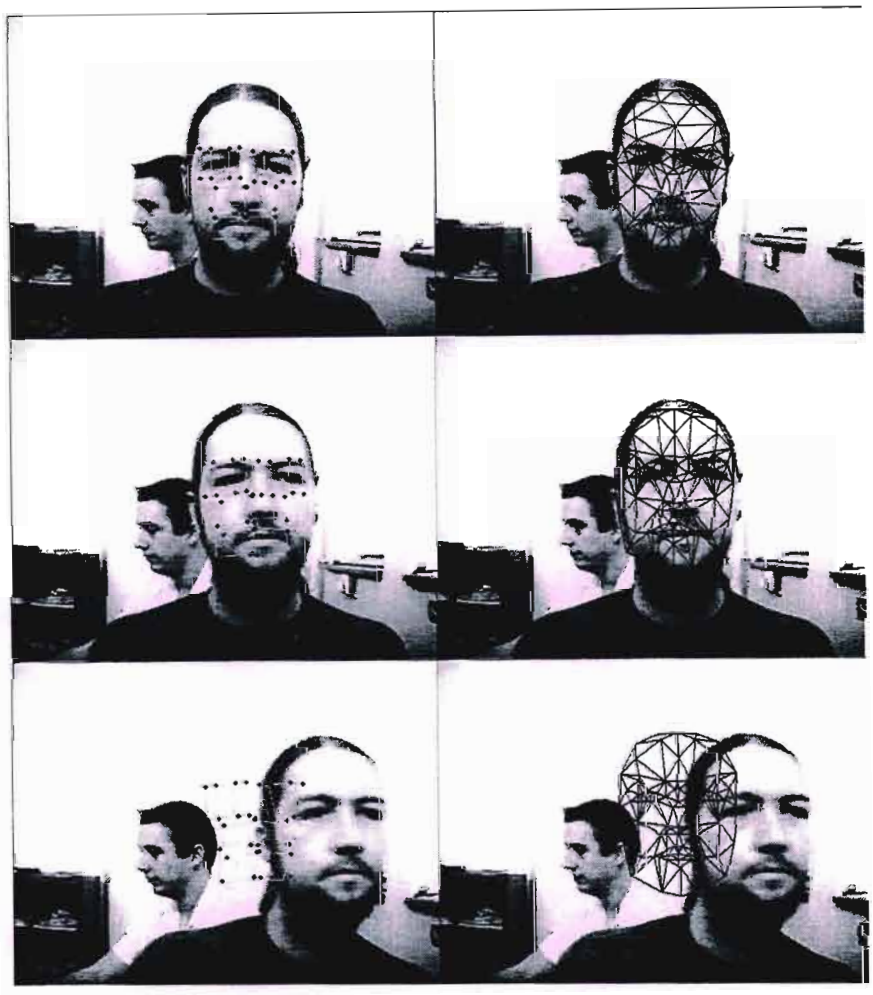


Figure 6.15: Tracking failure caused by very high inter-frame motion.

The third case, shown in Figure 6.16, occurs when large out-of-plane rotations occur. In this case most of the frontally located tracking points are facing away from the camera, and as in the case of occlusion, the convergence zone shrinks virtually to zero.

Figure 6.16: Tracking failure due to large out-of-plane rotation.

Tracking failure can be detected when the correlation values of more than 6 template windows meet the criteria of the Major Error case in Section 5.4.1. Currently, the proposed system is not able to recover from any of the aforementioned tracking failure scenarios, as no re-initialisation routine has been implemented. Just as in the initialisation phase, this would not be acceptable in a real system. Re-iterating however, the emphasis in this thesis is on demonstration of parameter recovery more than completeness of the head tracking system. The issue of re-initialisation is left as future work, which is discussed in Chapter 7. The subsequent paragraphs briefly discuss how some head tracking systems accomplish failure detection and re-initialisation.

Re-initialisation for face tracking systems is not a well documented task in the literature. Jebara and Pentland [22] normalise the face texture back to a frontal position and measure the "Distance From Face Space" (DFFS) as a means of detecting tracking failure. When this distance is too large, a face detection algorithm is used to restart the tracker from scratch, abandoning any tracking information previously gathered.

Two other works which employ related techniques are presented by Ström [3] and Matsumoto et al. [109]. The correlation values from a feature tracking process are used to detect tracking failure. A progressively refined template matching process for the whole face is then used to generate a new starting position for the face.

6.5 Tracking Performance

In looking at other systems, the tracker of Jebara et al. [22] achieves out-of-plane rotation estimation of up to ±45 degrees. Because the system presented in this thesis is largely based on the framework proposed in [22], it is appropriate that the the same range of recovery is achieved, as discussed in Section 6.3.4. Performance wise, the head tracking system of Jebara et al. [22] (based on the SfM solution in [28]) was shown (at the time this thesis began) to achieve the best frame rate of all the available tracking methods, achieving real-time (25 *fps*) video frame rates. This is discussed in Section 2.6.

Although the system design in this thesis relies heavily on the framework proposed in [22], the implementation was not intended for real-time execution, but rather to establish proof of principle only. Assuming that the face occupies at least one-third of the image in the initial frame, a best frame rate of 12 *fps* was achieved for the lower resolution sequences of Section 6.3.3. For the higher resolution sequences, performance decreases as larger templates (thus greater computational burden) are required around the facial features. It is assumed however, that efficient code and utilisation of graphics hardware (via OpenGL and Video for Windows (VFW)) would close the gap between the results achieved in this thesis and those in [22], where tracking frame rates up to 25 *fps* were achieved. This can be argued for two reasons. Firstly, no significant computational burden is added by using

a higher order motion model or adding an appearance basis in the 2D tracker. Frame rates of 35 fps have been achieved using the inverse compositional algorithm for tracking 2D features on the face. Secondly, because the Kalman filter framework is the same, the processing of measurement data should occur at the same rate. In fact, 25 Hz processing has been achieved in the synthetic experiments of Chapter 3. Thus, the primary bottleneck must occur between the video input and estimation module. Further literature proposing real-time frameworks using the same SfM approach can be found in [20] and [4].

Other head trackers which have achieved good frame rates include the system of Cascia et al. [27], running at 15 fps on the sequences from the ground truth database [104]. As shown in the ground truth validation of Section 6.3.3, the accuracy of the SfM based head tracker in this thesis is at least as good as the appearance based technique of Cascia et al. [27].

Near frame-rate ($\leq$ 15 fps) systems that have been published since the commencement of this thesis include those of Xiao et al. [29] and Zhu [30]. The head tracker in [29] achieves frame rates of 15 fps, with greater range of recovery in pitch (75 degrees) than any of the aforementioned systems in Chapter 2. This system was also evaluated on the ground truth sequences from [104]. A frame rate of 10 fps has been achieved in [30].

6.6 Evaluation Conclusion

This chapter presents an evaluation of the proposed head tracking system, looking at parameter convergence, estimation accuracy, system stability and computational performance. Two different sources of video data have been used, with validation of pose estimation accuracy achieved via a comparison with ground truth from a magnetic tracking system. The accuracy of the visual tracker against the ground truth has been shown to be as good as another well known visual tracking system in [27]. Full real-time performance has not been achieved with the implementation used for evaluation of the sequences in this chapter, but it assumed (based on the results achieved in similar works [22, 20, 4]) that performance can be improved. As the implementation was not intended for real-time

execution however, the presented results suitably establish proof of principle. Chapter 7 presents a conclusion to this thesis as well as proposals for future work.

Chapter 7

Conclusions and Future Work

7.1 Conclusions

An efficient method for estimating the rigid pose of a human head in head-and-shoulder video sequences typical of handheld devices (such as smartphones and PDAs) [5] has been proposed. The estimation of three-dimensional pose is one of several key problems still plaguing scene analysis at the encoder side of model-based video coding systems. The proposed technique is founded on well accepted (computer vision) theory, including a robust and versatile Structure from Motion solution, adopted by many, [28, 22, 20, 4, 49, 67, 55, 110] as a preferred method of deriving relative camera-to-object motion and object structure from a monocular video sequence.

Chapter 1 begins with a discussion on the merits and significant components of model-based coding schemes. Emphasis is placed on identifying the three-dimensional position and orientation parameters required to rigidly re-animate the facial model at the decoder side of a MBVC system.

Chapter 2 presents a discussion on the suitability of the Structure from Motion method of Azarbayejani and Pentland [28] relative to other proposed head tracking techniques. Of all the systems presented, only the adaptive feedback tracker (AFT) of Jebara et al. [22] (based on the SfM techniques in [28]) achieves full frame rate (25 fps). The technique is

robust to partial occlusions by probabilistically encoding the accuracy of measurements being fed into the SfM framework. Also, because the framework estimates object structure, it becomes possible to constrain feature tracking and prevent the drift that is typical of other head tracking systems. Thus the accuracy and stability achievable with the method (under quite diverse environmental variations) are key reasons that it has been identified as a robust framework for head pose estimation. This is evident from the abundance of work employing the method specifically for head tracking, including [22, 20, 55, 4]. The chapter closes with a system design proposal, utilising a more robust two-dimensional feature tracking technique (than that used in [22]) as a means of improving measurement acquisition for the SfM problem. The proposed feature tracking method employs an affine motion model and integrates a linear appearance basis into the tracking solution.

The Structure from Motion formulation of Azarbayejani et al. [28] is reviewed in Chapter 3. Their framework emphasises a unique structure representation as well as modifications to the standard camera, object and image transformation equations. Synthetic experimentation is performed to demonstrate how these revisions contribute towards system stability and robustness in the Kalman filter. Performance analysis is carried out under two different noise conditions. The algorithm is shown to extract reliable pose and structure information for both (high and low noise) cases. These results confirm the robustness of the approach to the pose estimation problem. Additionally, they highlight the possibility of controlling the level of measurement noise acceptable within the system as a means of ensuring the reliability of estimated state parameters.

In providing the Kalman filter with accurate measurements, Chapter 4 presents a short review of established two-dimensional tracking methods, highlighting the computational benefits of feature tracking techniques. Emphasis is placed on image alignment algorithms, with the development of an accurate and efficient tracking algorithm (the IC algorithm) based on the original Lucas-Kanade [79] image registration technique. When coupled with a small appearance basis, the robustness of the IC algorithm to variations in target appearance is improved. Baker et al. [91] show how an appearance basis can be integrated into the IC algorithm without any increase in online computational complexity. The suitability of the IC algorithm for measurement acquisition is evaluated using synthetic experiments,

with alignment accuracy assessed against known ground truth. Results demonstrate that sub-pixel accuracy is achievable, thereby validating the method for pixel measurement acquisition. Further experiments with a small appearance basis demonstrate algorithm stability under varying appearance conditions.

Chapter 5 reviews the implementation details of the final head tracking system and expands on the inherent control mechanisms that can be utilised from the EKF based SfM solution. Initial estimates for state parameters and noise statistics are discussed. Initial estimates from a projected wireframe face model are employed as a means of speeding up structural convergence. Integrating the 2D tracking algorithm and the EKF incorporates a reliability measure on pixel coordinates being fed to the EKF. In doing this, accurate measurements can be made to contribute more significantly towards state estimation than inaccurate ones. Furthermore, the feedback benefits of the Structure from Motion solution include the ability to constrain feature tracking in each frame using the estimated object structure. This contributes to system stability and robustness over a range of possible divergence scenarios.

Chapter 6 evaluates the head tracking system using real video sequences. The system is assessed under the system choices criteria of Chapter 2. These include accuracy of pose estimates, computational performance, range of recovery, occlusion handling and failure scenarios. The range of recoverable motion is found to perform favourably with other head tracking systems, with out-of-plane rotations of up to 45° and in-plane rotations up to 90° possible. The accuracy of recovered motion is assessed against independent ground truth from [104] where trajectories have been measured using a commercial magnetic orientation tracker. The accuracy of results achieved compare well with the ground truth as well as sample results from the work of [27]. By varying the measurement covariance associated with arriving pixel measurements, the system is shown to remain stable during partial occlusions. Three different failure scenarios for the system are also highlighted. These include excessive out-of-plane rotation, total facial occlusion and excessive inter-frame translation. These scenarios identify the need for a re-initialisation routine. Performance wise, a best frame rate of 12 *fps* can be achieved for the lower resolution sequences of Section 6.3.3. Efficient code and graphics hardware utilisation are proposed as methods

for improving the frame rate.

Overall, the SfM based head tracker has achieved the goals stipulated in Chapter 2, confirming the suitability of the approach as a global motion estimator in a MBVC system for faces. Improvements to the proposed system (such as a re-initialisation scheme) could however contribute to algorithm robustness. Section 7.2 closes this thesis by highlighting some relevant issues that still need to be addressed.

7.2 Future Work

The system proposed in this thesis is presented as a work in progress. Several aspects of the head tracking system can be modified in order to improve on efficiency, accuracy and stability in a model-based coding system.

An important addition to the system that would contribute significantly to practicality would be the addition of an automatic reinitialisation procedure, including both face detection and model alignment. Proposals for this have been discussed in Section 6.4. The same techniques could also easily be used for initialising the system at the start of the video sequence.

The choice of features being tracked within the alignment algorithm might also be considered. Once face detection (automatic or manual) has been performed, a better method of approaching the feature selection process would be to use some confidence measure. Automatic feature selection could then be implemented, where optimal regions are chosen based on a confidence measure, such as that proposed by Shi and Tomasi [111] or Papanikolopolous [112]. This would also afford the potential to add new feature points to the system after tracking has started, such as in the work of Strom [3]. Here, even a Yaw rotation through 90° is possible, as tracking points on the ear can be added, continuously providing an optimal number of good measurements for parameter estimation.

Further improvement to estimation accuracy could be achieved via improved modeling of measurement noise associated with features being fed into the extended Kalman filter.

In [22], the residual error of the template matching process is used to estimate the covariance matrix, providing a more accurate representation of the noise associated with each feature measurement to the Kalman estimator.

System efficiency could be greatly improved if all initialisation, tracking and display aspects of the system were efficiently written in some independent C++ application, possibly using OpenGL. Most modern computers have at least moderate graphics hardware that could be used to perform computationally intensive operations, such as the bilinear interpolation required in the IC algorithm. With this in mind, real-time frame rates of 25 Hz could be expected, as shown in [66] and [22].

In looking at developing the overall MBVC system further, many aspects still need to be considered, as discussed in Section 1.5.1. Perhaps the most significant step would be to estimate and model the non-rigid facial expressions that occur. Calvagno et al. [55] have proposed such a head tracking system, combining an EKF for global motion estimation with a module for estimating the Action Units caused by expression change.

Appendix A

Kalman Filtering

A.1 Linear Kalman Filtering

Following references such as Gelb et al. [50] or Brown [51], the Kalman filter addresses the general problem of estimating the *state vector*, $\bar{\mathbf{x}}_k$, of a discrete time, dynamic system governed by the linear stochastic difference equation:

$$\bar{\mathbf{x}}_k = F_{k-1}\bar{\mathbf{x}}_{k-1} + \bar{\mathbf{w}}_{k-1} \qquad \bar{\mathbf{w}}_k \sim N(\bar{0}, Q_k), \tag{A.1}$$

The matrix F_{k-1} describes the changes between state $\bar{\mathbf{x}}_{k-1}$ to $\bar{\mathbf{x}}_k$, and $\bar{\mathbf{w}}_k$ is a zero mean, white Gaussian sequence with covariance matrix Q_k, representing the process noise in the system.

This model further assumes that the state vector $\bar{\mathbf{x}}_k$ cannot be measured directly. Instead an observation vector of the state, $\bar{\mathbf{z}}_k$, can be measured according to

$$\bar{\mathbf{z}}_k = H_k\bar{\mathbf{x}}_k + \bar{\mathbf{v}}_k \qquad \bar{\mathbf{v}}_k \sim N(\bar{0}, R_k), \tag{A.2}$$

Here $\bar{\mathbf{z}}_k$ represents a vector of measurements or observations of the system at time k. $\bar{\mathbf{v}}_k$ is a zero mean Gaussian sequence with covariance matrix R_k, representing the random noise corrupting the measurements. Note that the matrix H_k can be of less than full rank or even rectangular. This means that the dimensionality of $\bar{\mathbf{z}}$ can be less than that of $\bar{\mathbf{x}}$.

Given an *a priori* estimate of the state at time t_k, denoted $\hat{\mathbf{x}}_k^-$, the aim is to find an updated estimate, $\hat{\mathbf{x}}_k^+$, based on measurements of the system $\hat{\mathbf{z}}_k$. The *a posteriori* state estimate $\hat{\mathbf{x}}_k^+$ is thus updated using

$$\hat{\mathbf{x}}_k^+ = \hat{\mathbf{x}}_k^- + K_k \left(\bar{\mathbf{z}}_k - H_k \hat{\mathbf{x}}_k^- \right) \tag{A.3}$$

where K_k is a time-varying weighting matrix, which still needs to be determined.

The estimate from the previous time step, $\hat{\mathbf{x}}_{k-1}^+$, can be improved before the measurement using the deterministic part of Equation (A.1) to generate the *a priori* estimate as

$$\bar{\mathbf{x}}_k^- = F_{k-1} \bar{\mathbf{x}}_{k-1}^+ \tag{A.4}$$

A.2 Calculating Kalman Gain

The filtering equations presented in Section A.1 perform a two step update process to the state vector $\bar{\mathbf{x}}_k$ at each time step k, using Equations (A.3) and (A.4). In updating the *a posteriori* estimate at each time step k, a weighting matrix K_k is required. The derivation of K_k below follows Gelb [50].

Letting P_k^+ denote the *a posteriori* error covariance matrix of $\hat{\mathbf{x}}_k^+$, defined as

$$P_k^+ = E\left[\left(\hat{\mathbf{x}}_k^+ - \bar{\mathbf{x}}_k \right) \left(\hat{\mathbf{x}}_k^+ - \bar{\mathbf{x}}_k \right)^T \right] \tag{A.5}$$

simplification of this expression, from [50], yields

$$P_k^+ = \left(I - K_k H_k \right) P_k^- \left(I - K_k H_k \right)^T + K_k R_k K_k^T \tag{A.6}$$

where $R_k = E\left[\bar{\mathbf{v}}_k \bar{\mathbf{v}}_k^T \right]$, i.e., the covariance matrix of $\bar{\mathbf{v}}_k$. Differentiating P_k^+ with respect to K_k and setting the result to zero yields

$$-2 \left(I - K_k H_k \right) P_k^- H_k^T + 2 K_k R_k = 0 \tag{A.7}$$

Solving for K_k results in

$$K_k = P_k^- H_k^T \left[H_k P_k^- H_k^T + R_k \right]^{-1} \tag{A.8}$$

which is the optimal solution to the *Kalman gain* matrix [50]. Substitution of (A.8) into the expression for the *a posteriori* error covariance Equation (A.6), with some manipulation, gives an optimum choice for P_k^+ as

$$P_k^+ = (I - K_k H_k) P_k^- \tag{A.9}$$

An update equation for the *a priori* error covariance P_k^- is also needed. Using the definition

$$P_k^- = E\left[\left(\hat{\mathbf{x}}_k^- - \bar{\mathbf{x}}_k\right)\left(\hat{\mathbf{x}}_k^- - \bar{\mathbf{x}}_k\right)^T\right] \tag{A.10}$$

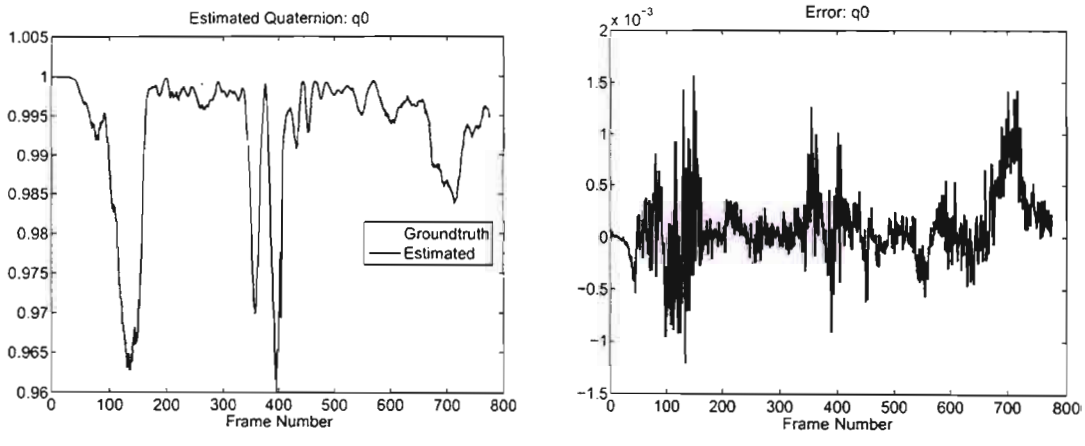
and Equation (A.4), yields, after some manipulations [50],

$$P_k^- = F_{k-1} P_{k-1}^+ F_{k-1}^T + Q_{k-1} \tag{A.11}$$

Appendix B

Synthetic Experiments

B.1 Motion Estimates: Low Noise



(a) (b)

Figure B.1: Quaternion estimates for q_0 with corresponding estimation error.

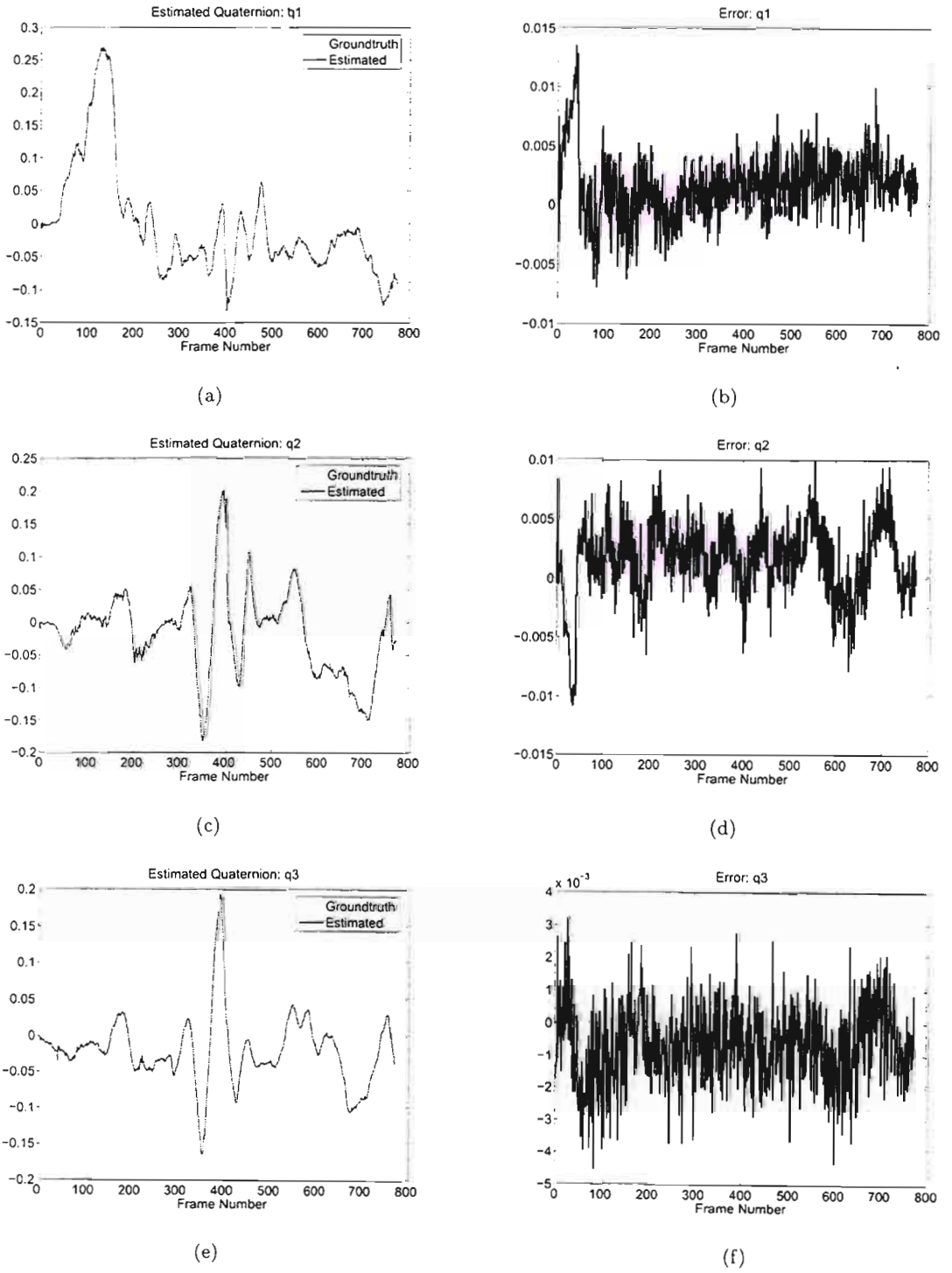


Figure B.2: Quaternion estimates for q_1, q_2 and q_3 with corresponding estimation errors.

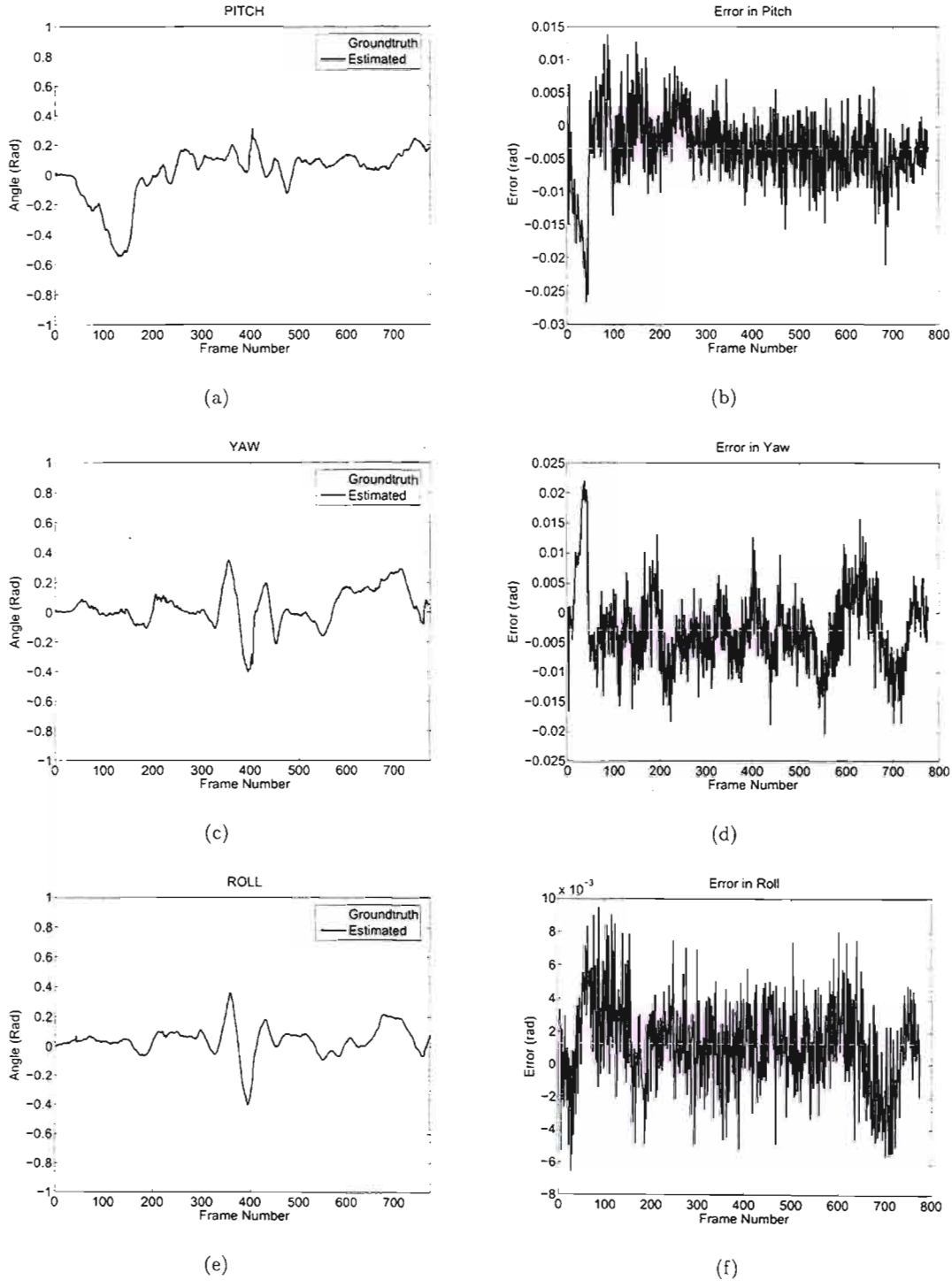


Figure B.3: Euler angles (converted from the estimated rotation quaternion) with corresponding estimation errors.

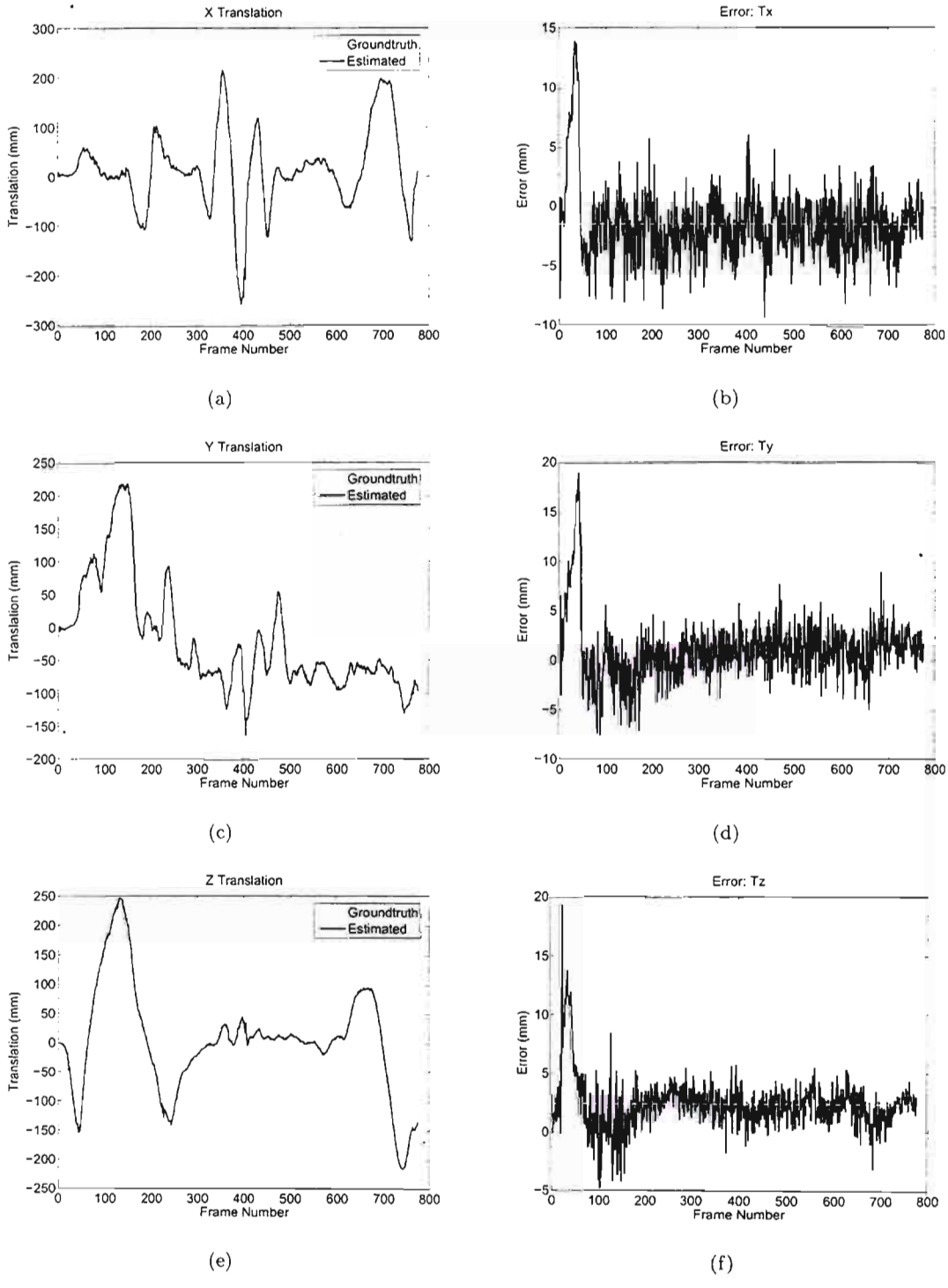


Figure B.4: Estimated translation parameters with corresponding estimation errors.

B.2 Motion Estimates: High Noise

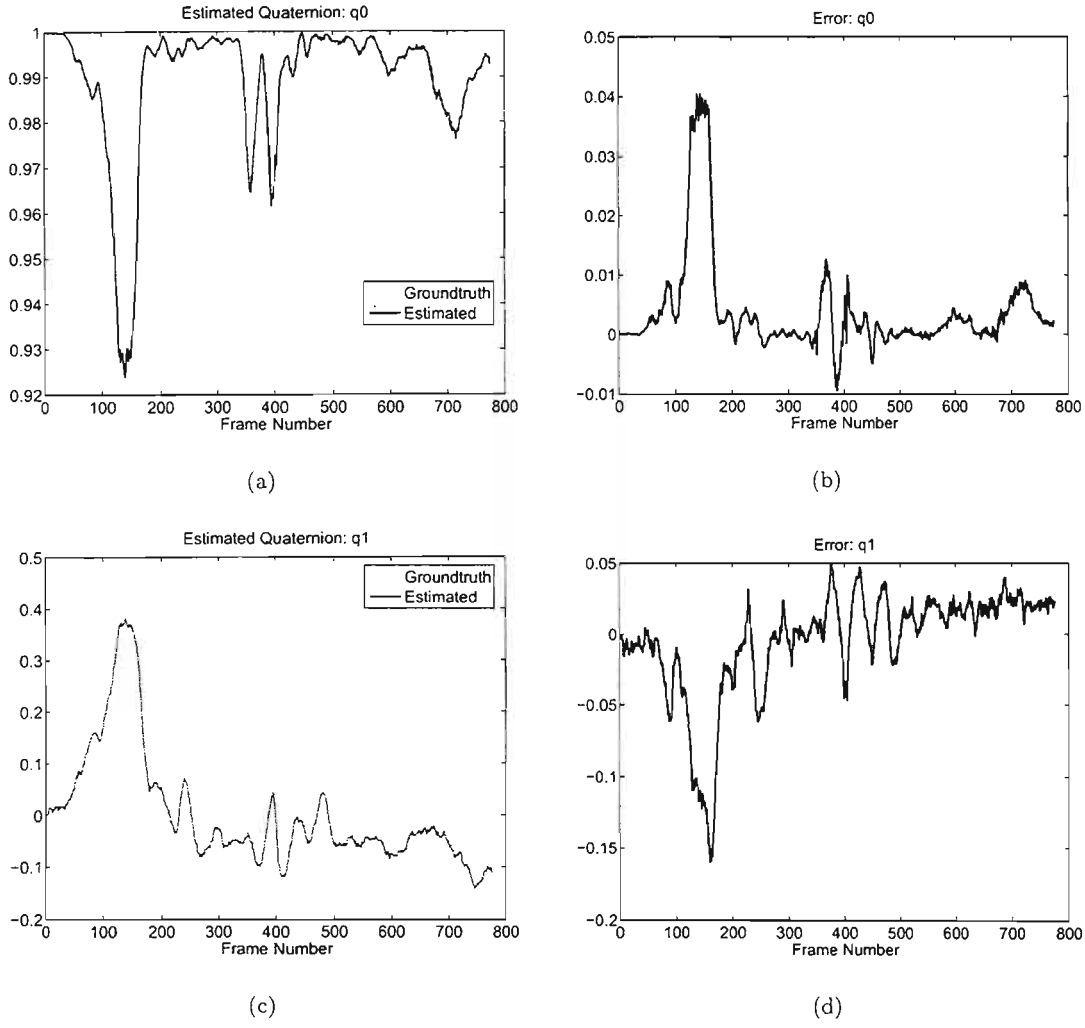


Figure B.5: Quaternion estimates for q_0 and q_1 with corresponding estimation errors using high noise measurements.

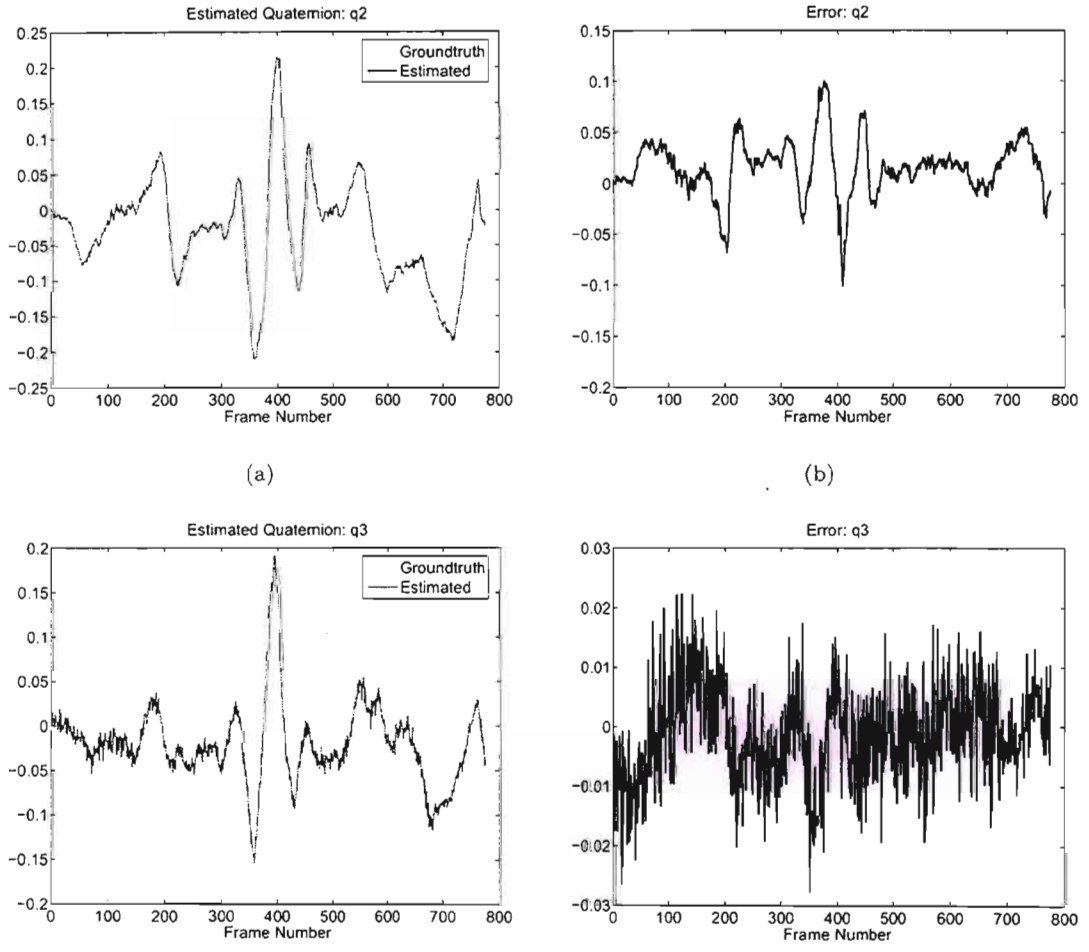


Figure B.6: Quaternion estimates for q_2 and q_3 with corresponding estimation errors using high noise measurements.

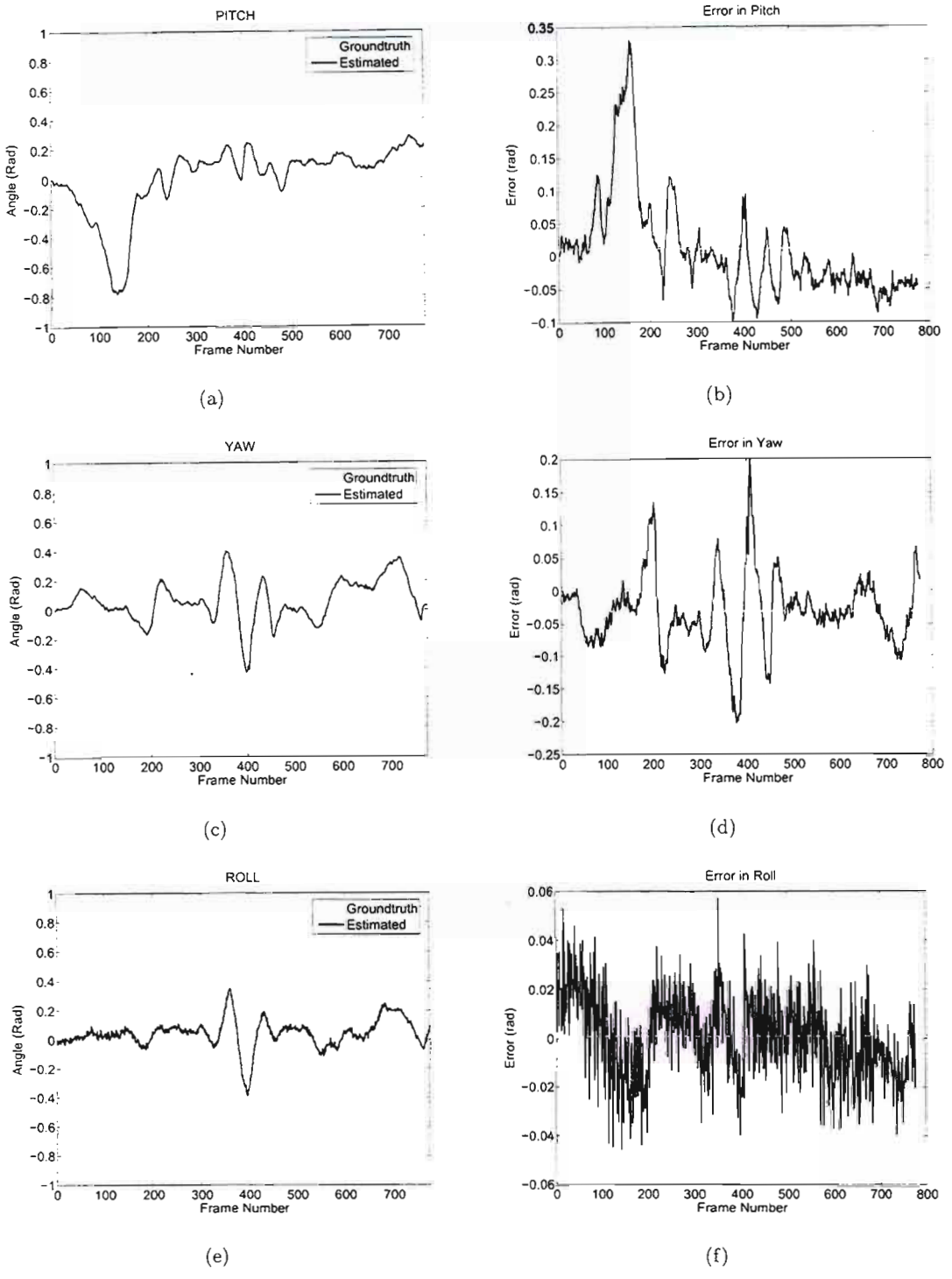


Figure B.7: Euler angles (converted from the estimated rotation quaternion) with corresponding estimation errors using high noise measurements.

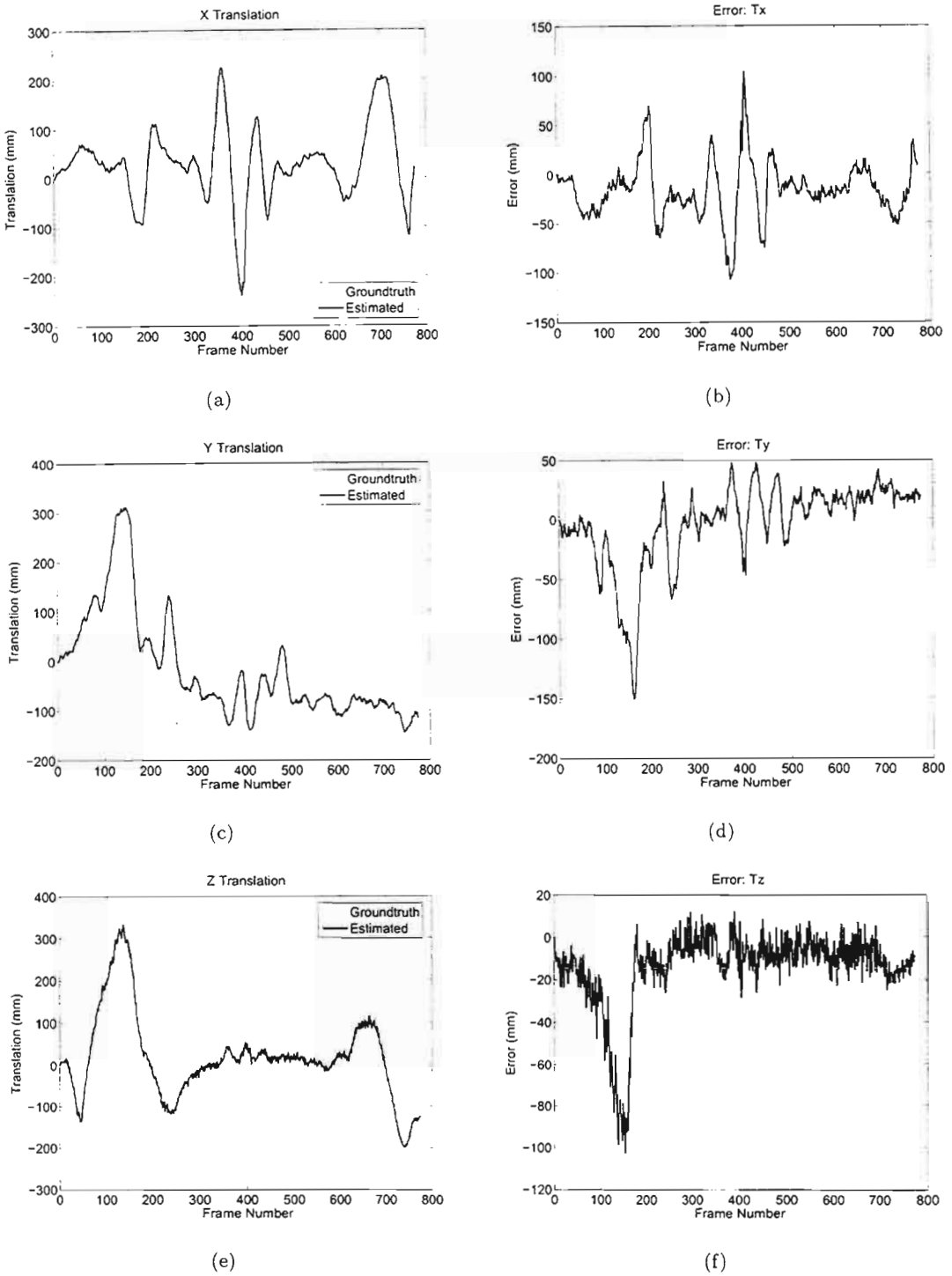


Figure B.8: Estimated translation parameters with corresponding estimation errors using high noise measurements.

B.3 Structure and Camera Parameter Estimates: High Noise

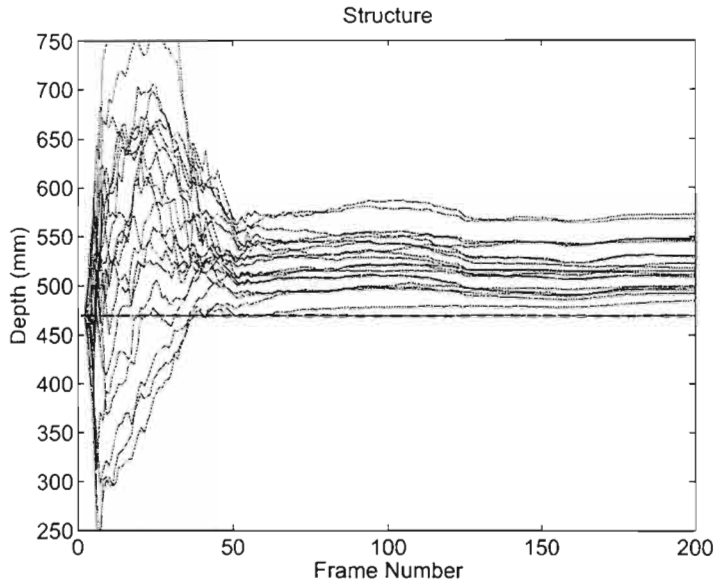


Figure B.9: Convergence of the 21 structural points using highly corrupted measurement data.

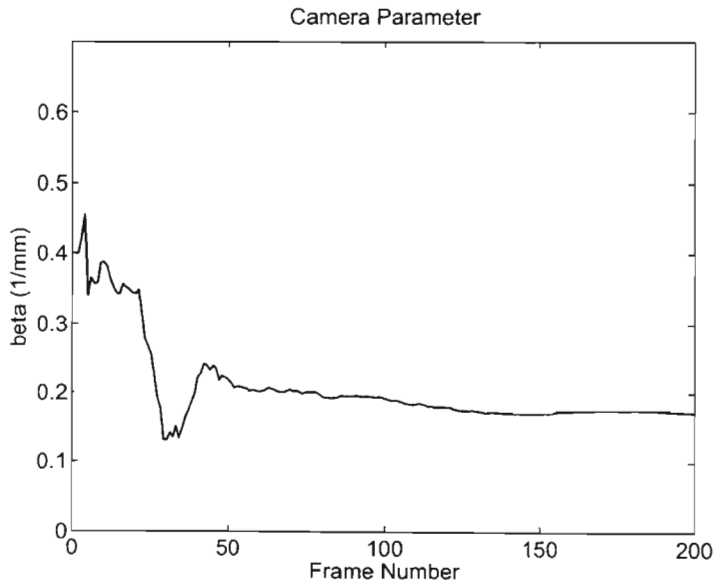
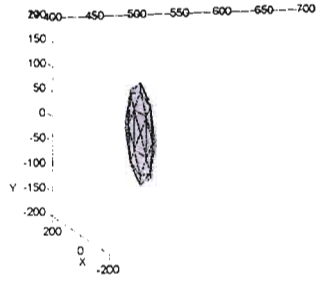
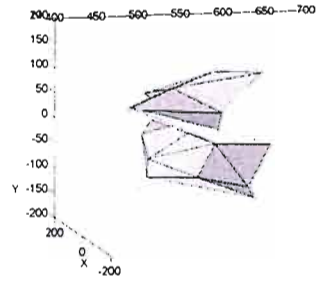


Figure B.10: Convergence of the camera parameter β using highly corrupted measurement data.

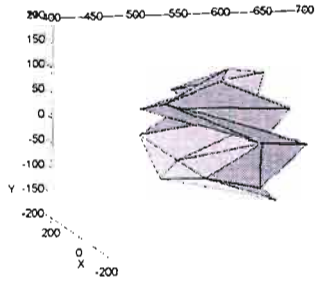
B.4 Structure Estimates: Structural Evolution



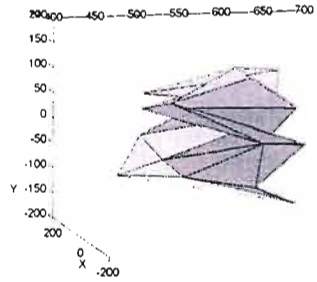
(a) Frame 1



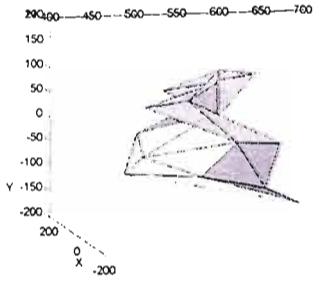
(b) Frame 3



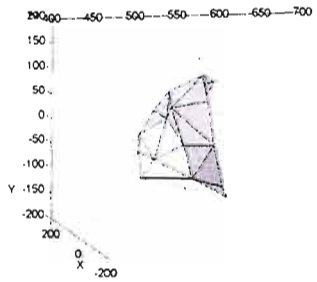
(c) Frame 5



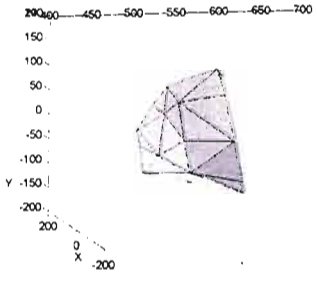
(d) Frame 10



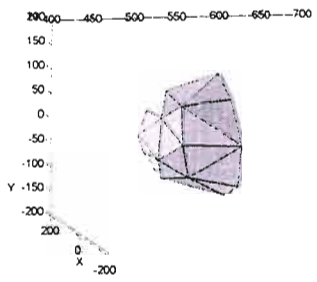
(e) Frame 20



(f) Frame 30



(g) Frame 40



(h) Frame 80

Figure B.11: Evolution of structure points to a hemispherical structure.

Appendix C

Linear Appearance Modeling

C.1 IC Algorithm with Linear Appearance Variation

The derivation of the inverse compositional algorithm which includes a linear appearance basis follows the derivation of Baker and Mathews [102]. The derivation begins by assuming that the expression:

$$T(\mathbf{x}) + \sum_{i=1}^{m} \lambda_i A_i(\mathbf{x}) \tag{C.1}$$

appears in the input image $I(\mathbf{x})$ and that the objective function to be minimised is:

$$\sum_{\mathbf{x} \in R} \left(T(\mathbf{x}) + \sum_{i=1}^{m} \lambda_i A_i(\mathbf{x}) - I(\mathbf{W}(\mathbf{x}; \mathbf{p})) \right)^2 \tag{C.2}$$

This is achieved by treating the images as vectors over the range of the pixels $\mathbf{x}$, rewriting Equation (C.2) as:

$$\begin{aligned}&\sum_{\mathbf{x}\in R}\left(T(\mathbf{x})+\sum_{i=1}^{m}\lambda_i A_i(\mathbf{x})-I(\mathbf{W}(\mathbf{x};\mathbf{p}))\right)^2\\&=\left\|T(\mathbf{x})+\sum_{i=1}^{m}\lambda_i A_i(\mathbf{x})-I(\mathbf{W}(\mathbf{x};\mathbf{p}))\right\|^2\end{aligned}\tag{C.3}$$

where $\|\cdot\|$ is the unweighted (Euclidean) least squares ($L2$) norm. This expression is then minimized simultaneously over the vector parameters $\mathbf{p}$ and the appearance coefficients λ.

Remember that the *span* of a set of vectors is the set of all linear combinations of those vectors. The span of this set of vectors in $\Re^n$ then provides a *subspace* of $\Re^n$ [99]. The linear subspace spanned by the collection of vectors A_i can now be denoted as $span(A_i)$, and its orthogonal complement by $span(A_i)^{\perp}$ [102]. Equation (C.3) can then be written as:

$$\begin{aligned}&\left\|T(\mathbf{x})+\sum_{i=1}^{m}\lambda_i A_i(\mathbf{x})-I(\mathbf{W}(\mathbf{x};\mathbf{p}))\right\|^2_{span(A_i)}\\&+\left\|T(\mathbf{x})+\sum_{i=1}^{m}\lambda_i A_i(\mathbf{x})-I(\mathbf{W}(\mathbf{x};\mathbf{p}))\right\|^2_{span(A_i)^{\perp}}\end{aligned}\tag{C.4}$$

In looking at these two terms, a simplification should be apparent. Since the norm in the second term only considers the component of the vector in the orthogonal complement of $span(A_i)$, any component in $span(A_i)$ can be dropped, which changes the the minimisation to:

$$\begin{aligned}&\left\|T(\mathbf{x})+\sum_{i=1}^{m}\lambda_i A_i(\mathbf{x})-I(\mathbf{W}(\mathbf{x};\mathbf{p}))\right\|^2_{span(A_i)}\\&+\left\|T(\mathbf{x})-I(\mathbf{W}(\mathbf{x};\mathbf{p}))\right\|^2_{span(A_i)^{\perp}}\end{aligned}\tag{C.5}$$

Now the second of the two terms does not depend upon λ, and because the term $\sum_{i=1}^{m}\lambda_i A_i(\mathbf{x})$ can represent any vector in $span(A_i)$, the minimum value of the first term is always exactly 0, no matter what the value of $\mathbf{p}$. The simultaneous minimum over both $\mathbf{p}$ and λ can now be found in a sequential manner by first minimizing the second term with respect to $\mathbf{p}$

alone. Then, treating the optimal value of $\mathbf{p}$ as a constant, the first term can be minimised with respect to λ. Using orthonormalised versions of the appearance vectors A_i (if they are not orthonormalised a Gram-Schmidt orthonormalisation procedure can be used [99]), a closed-form solution to the minimisation of the first term is:

$$\lambda_i = \sum_{\mathbf{x} \in R} A_i(\mathbf{x}) \left(I(\mathbf{W}(\mathbf{x}; \mathbf{p})) - T(\mathbf{x}) \right) \tag{C.6}$$

The only difference between minimizing the second term in Equation (C.4) and the original goal of the Lucas-Kanade Algorithm is the need to work in the linear subspace $span(A_i)^{\perp}$. Working in this subspace can be achieved by using a weighted least squares $L2$ norm. A discussion on weighted $L2$ norms is providied in [113]. In this case, Baker et al. [102] use the weighting function

$$Q(\mathbf{x}, \mathbf{y}) = \delta(\mathbf{x}, \mathbf{y}) - \sum_{i=1}^{m} \left(A_i(\mathbf{x}) \cdot A_i(\mathbf{y}) \right) \tag{C.7}$$

(again assuming that the vectors A_i are orthonormal) and minimise the expression

$$\sum_{\mathbf{x}} \sum_{\mathbf{y}} Q(\mathbf{x}, \mathbf{y}) \cdot \left(T(\mathbf{x}) - I(\mathbf{W}(\mathbf{x}; \mathbf{p})) \right) \cdot \left(T(\mathbf{y}) - I(\mathbf{W}(\mathbf{y}; \mathbf{p})) \right) \tag{C.8}$$

which is essentially a combination of the weighting function Equation (C.7) and the second term of Equation (C.5). Minimisation of a $L2$ norm with a weighting function is described in [113], with specific attention to the inverse compositional algorithm. The result of this minimisation for the weighted steepest descent images, from [113], is then

$$SD_Q(\mathbf{x}) = \sum_{\mathbf{x}} Q(\mathbf{x}, \mathbf{y}) \left(\nabla T \frac{\partial \mathbf{W}}{\partial \mathbf{p}} (\mathbf{y}; \mathbf{0}) \right)^T \tag{C.9}$$

so that with Equation (C.7) the new steepedst descent images become

$$SD(\mathbf{x}) = \sum_{\mathbf{y}} \left(\delta(\mathbf{x}, \mathbf{y}) - \sum_{i=1}^{m} (A_i(\mathbf{x}) \cdot A_i(\mathbf{y})) \right) \left(\nabla T \frac{\partial \mathbf{W}}{\partial \mathbf{p}} (\mathbf{y}; \mathbf{0}) \right) \tag{C.10}$$

and so can be computed:

$$SD(\mathbf{x}) = \nabla T \frac{\partial \mathbf{W}}{\partial \mathbf{p}} (\mathbf{x}; \mathbf{0}) - \sum_{i=1}^{m} \left(\sum_{\mathbf{y}} A_i(\mathbf{y}) \nabla T \frac{\partial \mathbf{W}}{\partial \mathbf{p}} (\mathbf{y}; \mathbf{0}) \right) A_i(\mathbf{x}) \tag{C.11}$$

This expression indicates the projection of the unweighted steepest descent images $\nabla T \frac{\partial \mathbf{W}}{\partial \mathbf{p}} (\mathbf{x}; \mathbf{0})$ into $span(A_i)^{\perp}$ by sequentially removing the components in the direction of A_i, for $i = 1, \ldots, m$.

The weighted Hessian matrix, again from [113], of the form

$$H_Q = \sum_{\mathbf{x}} \sum_{\mathbf{y}} Q(\mathbf{x}, \mathbf{y}) \left(\nabla T \frac{\partial \mathbf{W}}{\partial \mathbf{p}} (\mathbf{x}; \mathbf{0}) \right)^T \left(\nabla T \frac{\partial \mathbf{W}}{\partial \mathbf{p}} (\mathbf{y}; \mathbf{0}) \right) \tag{C.12}$$

can then also be computed as:

$$H = \sum_{\mathbf{x}} SD(\mathbf{x})^T SD(\mathbf{x}) \tag{C.13}$$

because the inner product of the two vectors projected onto a linear subspace is the same as if just one of the two is projected into the linear subspace [113].

Appendix D

Head Tracking Results

D.1 Example Sequence - Glasses

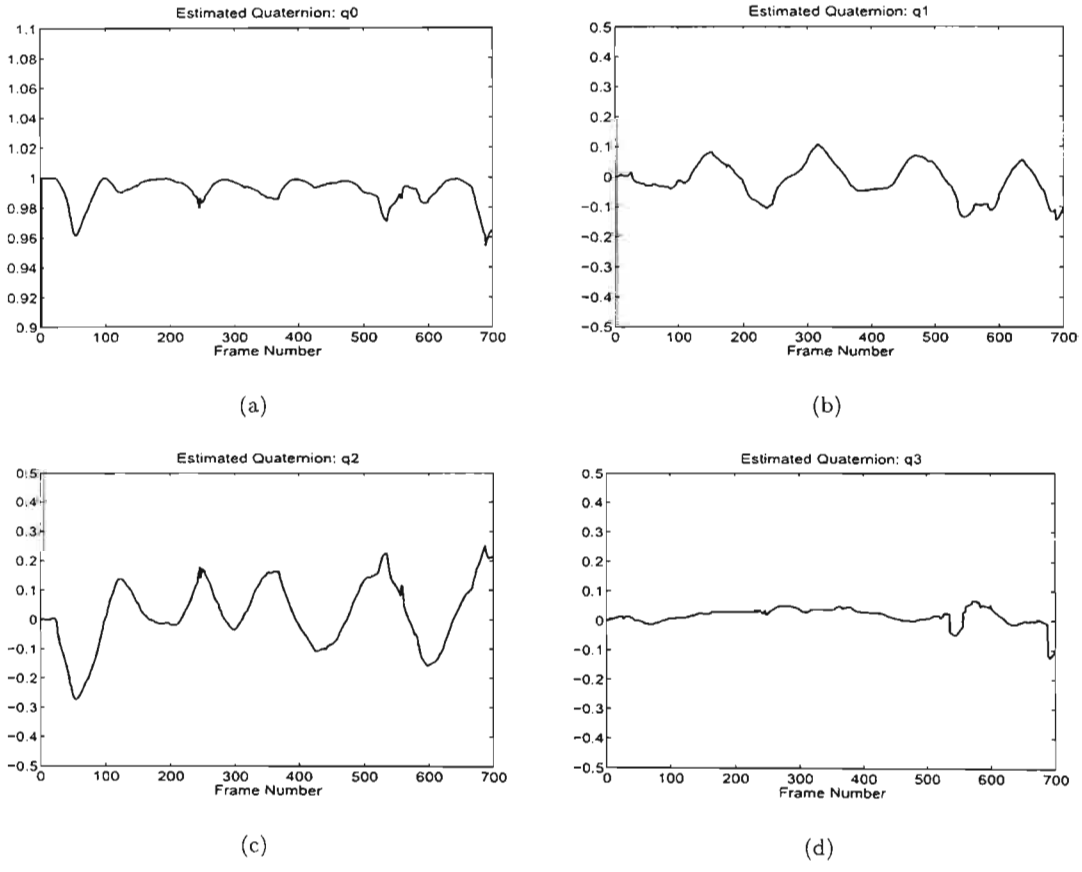
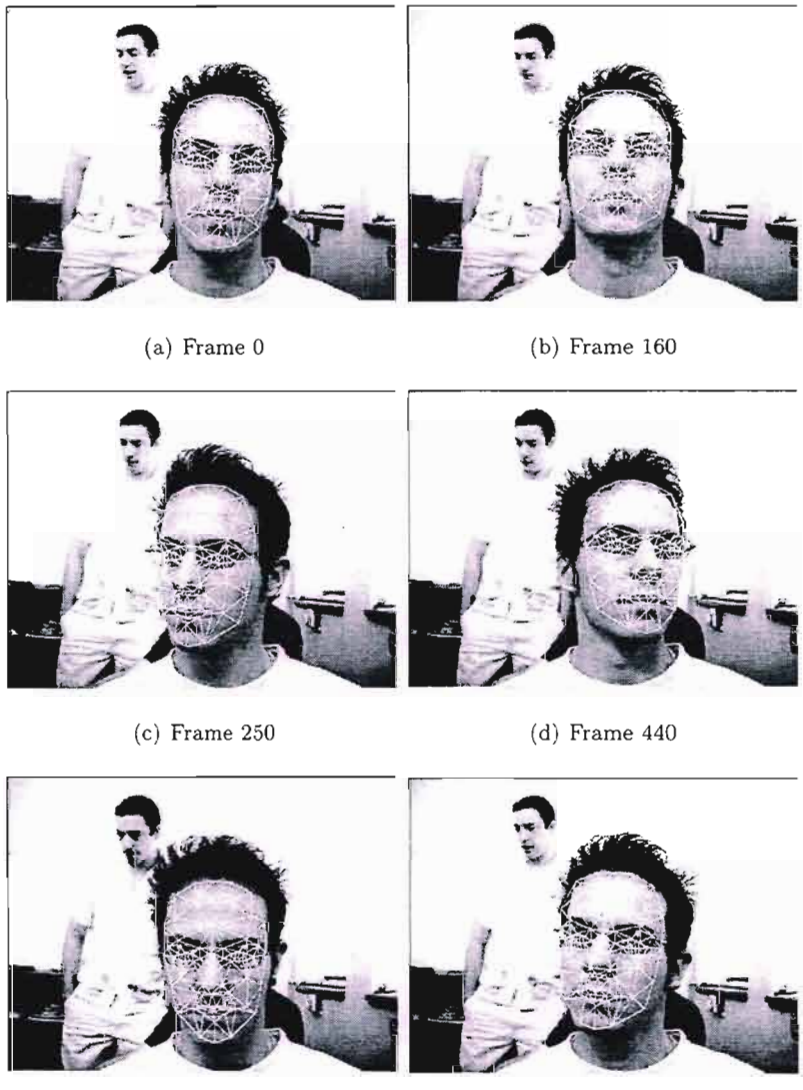


Figure D.1: Quaternion estimates for the sequence with the user wearing glasses.



(a) Frame 0 (b) Frame 160

(c) Frame 250 (d) Frame 440

(e) Frame 560 (f) Frame 660

Figure D.2: Sequence demonstrating tracking with an irregular face structure where the user is wearing glasses.

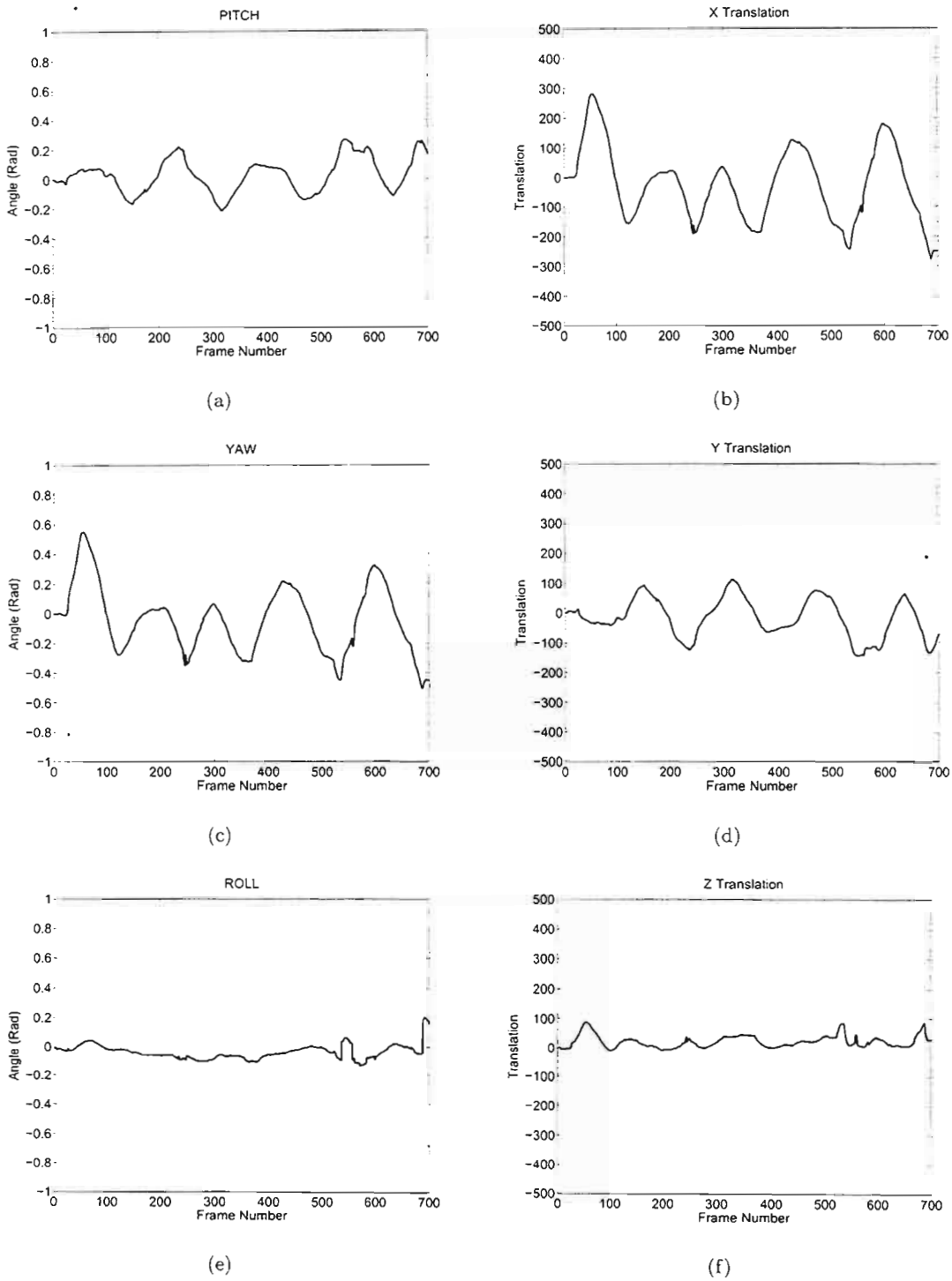
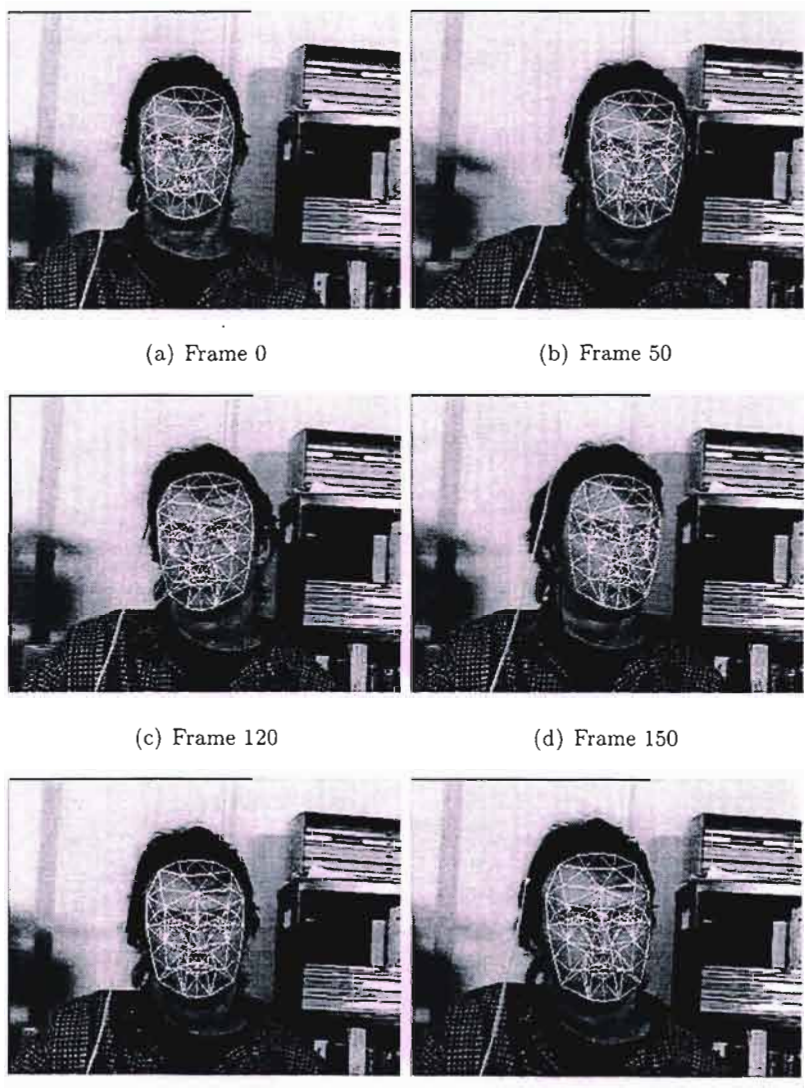


Figure D.3: Motion estimates for the sequence with the user wearing glasses.

D.2 Ground Truth Sequence



(a) Frame 0 (b) Frame 50

(c) Frame 120 (d) Frame 150

(e) Frame 180 (f) Frame 200

Figure D.4: Ground truth tracking sequence.

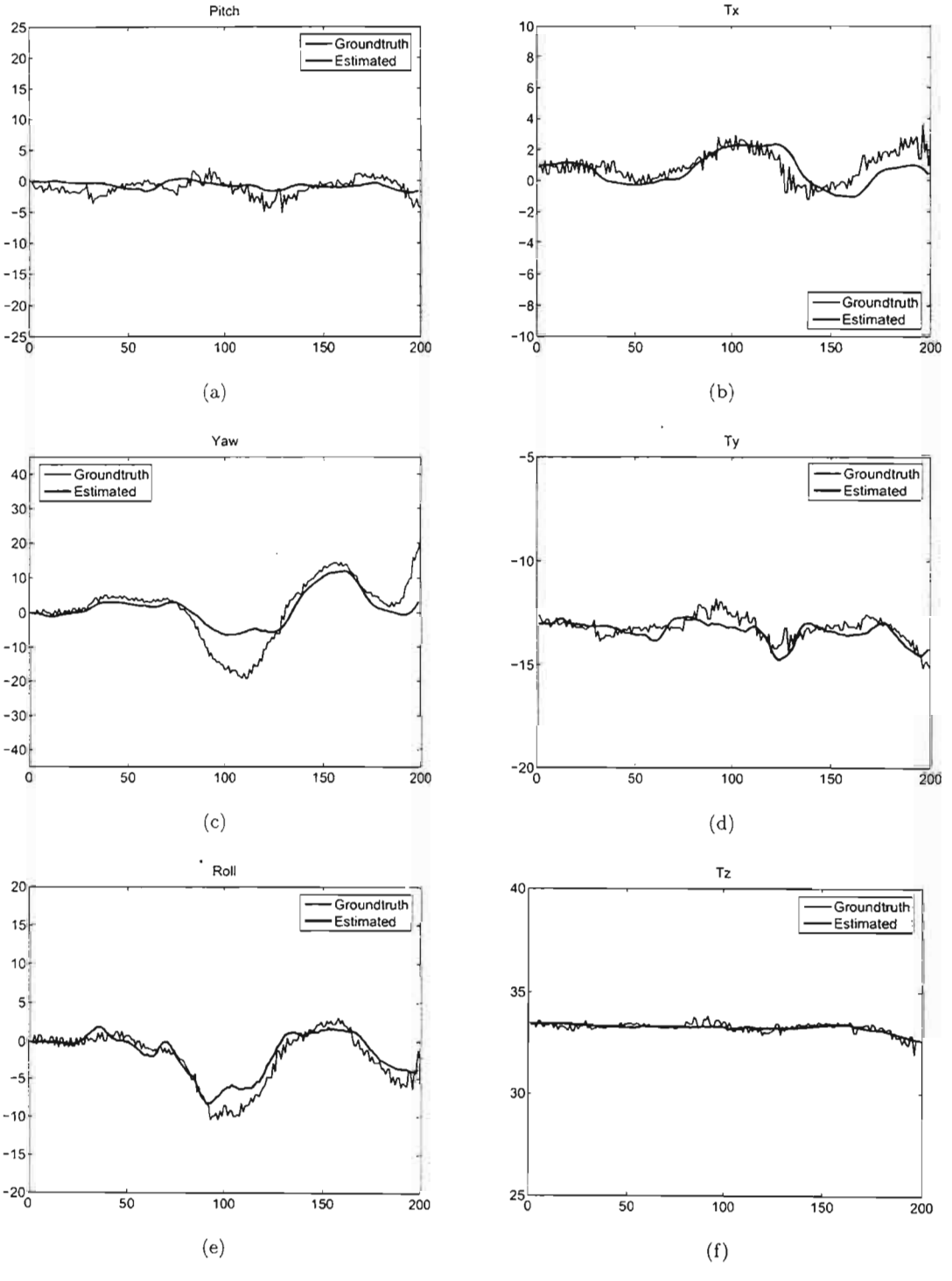


Figure D.5: Ground truth sequence motion estimates.

Bibliography

[1] M. Rydfalk, "CANDIDE, a parameterized face," tech. rep., Linköping University, 1987.

[2] I. Essa, S. Basu, T. Darrell, and A. Pentland, "Modeling, tracking and interactive animation of faces and heads using input from video," in *Proceedings of Computer Animation 96 Conference*, 1996.

[3] J. Strom, "Model-based real-time head tracking," *Journal of Applied Signal Processing*, vol. 10, pp. 1039–1052, October 2002.

[4] M. D. Cordea, D. C. Petriu, E. M. Petriu, N. D. Georganas, and T. E. Whalen, "3-D head pose recovery for interactive virtual reality avatars," *IEEE Transactions on Instrumentation and Measurement*, vol. 51, August 2002.

[5] P. Eisert, "Mpeg-4 facial animation in video analysis and synthesis," *International Journal Of Imaging Systems And Technology*, 2003.

[6] H. Li, P. Roivainen, and R. Forchheimer, "3-D motion estimation in model-based facial image coding," *Pattern Analysis and Machine Intelligence*, vol. 15, pp. 545–555, June 1993.

[7] D. Pearson, "Developments in model-based video coding," *Proceedings of the IEEE*, vol. 83, pp. 892–906, June 1995.

[8] H. Li, A. Lundmark, and R. Forchheimer, "Image sequence coding at very low bit rates: a review," *IEEE Transaction on Image Processing*, vol. 3, pp. 589–609, September 1994.

[9] W. Welsh, "Model-based coding of videophone images," *Electronic and Communication Engineering Journal*, 1991.

[10] K. Aizawa and T. Huang, "Model-based image-coding: Advanced video coding techniques for very-low bit-rate applications," *Proc. IEEE*, vol. 83, pp. 259–271, February 1995.

[11] P. Eisert and B. Girod, "Analyzing facial expressions for virtual conferencing," *IEEE, Computer Graphics and Applications*, vol. 18, no. 5, pp. 70–78, 1998.

[12] P. Ekman and W. Friesen, "The facial action coding system: A technique for the measurement of facial movement," in *Consulting Psychologists*, 1978.

[13] Moving Picture Experts Group, *ISO/IEC 14496, International Standard on Coding of Audio-Visual Objects (MPEG-4)*, 1999.

[14] R. Forchheimer and O. Fahlander, "Low bit-rate coding through animation," in *Proceedings of the Picture Coding Symposium (PCS)*, 1983.

[15] R. Forchheimer, O. Fahlander, and T. Kronander, "A semantic approach to the transmission of face images," in *Proceedings of the Picture Coding Symposium (PCS)*, (Cesson-Sevigne, France), 1984.

[16] K. Aizawa, H. Harashima, and T. Saito, "A model-based image coding system - construction of a 3-D model of a persons face," in *Proceedings of the International Picture Coding Symposium (PCS)*, 1987.

[17] H. Li, A. Lundmark, and R. Forchheimer, "Image sequence coding at very low bit rates: a review," *IEEE Transaction on Image Processing*, vol. 3, pp. 589–609, September 1994.

[18] J. yong Noh and U. Neumann, "A survey of facial modeling and animation techniques," tech. rep., Integrated Media Systems Center, University of Southern California.

[19] M. Yang, D. Kriegman, and N. Ahuja, "Detecting faces in images: A survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 1, p. 3458, 2002.

[20] J. Strom, T. Jebara, S. Basu, and A. Pentland, "Real time tracking and modeling of faces: An ekf-based analysis by synthesis approach," in *Proceedings of the Modelling People Workshop at ICCV'99*, August 1999.

[21] S. Basu, I. Essa, and A. Pentland, "Motion regularization for model-based head tracking," in *ICPR96*, p. C8A.3, 1996.

[22] T. Jebara and A. Pentland, "Parametrized structure from motion for 3D adaptive feedback tracking of faces," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR'97)*.

[23] K. Toyama, "Prolegomena for robust face tracking," Tech. Rep. MSR-TR-98-65, Microsoft Research, Nov 1998.

[24] Y. Yacoob, H. Lam, and L. S. Davis, "Recognizing faces showing expressions.," in *Proceedings of the International Conference on Pattern Recognition*, 1994.

[25] I. A. Essa and A. Pentland, "Facial expression recognition using visually extracted facial action parameters," in *Proceedings of the International Workshop on Automatic Face and Gesture Recognition*, 1995.

[26] C. Choi, K. Aizawa, H. Harashima, and T. Takebe, "Analysis and synthesis of facial image sequences in model-based image coding," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 4, pp. 257–275, June 1994.

[27] M. L. Cascia, S. Sclaro, and V. Athitsos, "Fast, reliable head tracking under varying illumination: An approach based on registration of texture-mapped 3d models," *IEEE Transactions on Pattern Analyis and Machine Intelligence*, vol. 22, pp. 322–336, April 2000.

[28] A. Azarbayejani and A. Pentland, "Recursive estimation of motion structure and focal length," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 17, pp. 562–575, June 1995.

[29] J. Xiao, T. Moriyama, T. Kanade, and J. Cohn, "Robust full-motion recovery of head by dynamic templates and re-registration techniques," *IJIST*, vol. 13, no. 1, pp. 85–94, 2003.

[30] Z. Zhu and Q. Ji, "Real time 3D face pose tracking from an uncalibrated monocular camera," in *ICPR04*, pp. IV: 400–403, 2004.

[31] J. Heathcote, B. Naidoo, and S. McDonald, "Structure from motion for real-time head pose recovery," in *Proceedings of the The Military Information and Communications Symposium of South Africa (MICSSA)*, 2003.

[32] J. Heathcote, B. Naidoo, and S. McDonald, "Full 6-DOF head pose recovery: A parameterized structure from motion approach," in *Proceedings of the South African Telecommunications, Networks and Applications Conference (SATNAC)*, 2003.

[33] J. Heathcote, B. Naidoo, and S. McDonald, "Structure from motion for real-time 3D head pose recovery," in *Proceedings of the Annual Symposium of the Pattern Recognition Association of South Africa (PRASA)*, 2003.

[34] J. Heathcote, B. Naidoo, and S. McDonald, "Recursive estimation of head pose in a model based video coding system," in *Proceedings of the South African Telecommunications, Networks and Applications Conference (SATNAC)*, 2004.

[35] J. Heathcote, B. Naidoo, and S. McDonald, "A head pose estimator for model based video coding of faces," in *Proceedings of the The Military Information and Communications Symposium of South Africa (MICSSA)*, 2005.

[36] I. Essa, "Computers seeing people," *American Association for Artificial Intelligence*, 1999.

[37] A. Azarbayejani, T. Starner, B. Horowitz, and A. Pentland, "Visually controlled graphics," *PAMI*, vol. 15, pp. 602–605, June 1993.

[38] M. Black and Y. Yacoob, "Recognizing facial expressions in image sequences using local parameterized models of image motion," *IJCV*, vol. 25, pp. 23–48, October 1997.

[39] Y. Zhang and C. Kambhamettu, "Robust 3D head tracking under partial occlusion," in *Fourth International Conference on Face and Gesture Recognition*, 2000.

[40] P. Roivainen, *Motion estimation in model-based coding of human faces.* PhD thesis, ISY, Linköping University, Sweden, 1990.

[41] D. DeCarlo and D. Metaxas, "The integration of optical flow and deformable models: Applications to human face shape and motion estimation," in *CVPR96*, pp. 231–238, 1996.

[42] M. Harville, A. Rahimi, T. Darrell, G. Gordon, and J. Woodfill, "3D pose tracking with linear depth and brightness constraints," in *ICCV99*, pp. 206–213, 1999.

[43] T.Cootes, G. Edwards, and C.J.Taylor, "Active appearance models," in *European Conference on Computer Vision 1998* (H.Burkhardt and B. Neumann, eds.), 1998.

[44] T. Cootes, G. Edwards, and C. Taylor, "A comparative evaluation of active appearance model algorithms," in *BMVC98*, 1998.

[45] G. Edwards, T. Cootes, and C. Taylor, "Face recognition using active appearance models," in *ECCV98*, pp. 581–695, 1998.

[46] F. Dornaika and J. Ahlberg, "Model-based head and facial motion tracking," in *12th International Conference on Image Analysis and Processing (ICIAP)*, 2003.

[47] J. Xiao, S. Baker, I. Matthews, and T. Kanade, "Real-time combined 2D+3D active appearance models," in *CVPR04*, pp. II: 535–542, 2004.

[48] Z. Zivkovic and F.Heijden, "A stabilized adaptive appearance changes model for 3D head tracking," in *IEEE ICCV Workshop on Recognition, Analysis, and Tracking of Faces and Gestures in Real-Time Systems*, 2001.

[49] T. Jebara, A. Azarbayejani, and A. Pentland, "3D structure from 2D motion," *IEEE Signal Processing Magazine*, pp. 66–84, May 1999.

[50] A. Gelb, *Applied Optimal Estimation.* MIT Press, 1974.

[51] R. G. Brown, *Introduction to Random Signal Analysis and Kalman Filtering.* John Wiley & Sons, New York, 1983.

[52] M. D. Cordea, D. C. Petriu, E. M. Petriu, N. D. Georganas, and T. E. Whalen, "Real-time 2(1/2)-D head pose recovery for model-based video-coding," *IEEE Transactions on Instrumentation and Measurement*, vol. 50, August 2001.

[53] G. Hager and P. Belhumer, "Efficient region tracking with parametric models of geometry and illumination," *IEEE Pattern Analysis and Machine Intelligence*, vol. 20, pp. 1025–1039, October 1998.

[54] Cyberware Inc., http://www.cyberware.com.

[55] G. Calvagno, F. Fantozzi, R. Rinaldo, and A. Viareggio, "Model-based global and local motion estimation for videoconference sequences," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 14, pp. 1156–1161, September 2004.

[56] J. Oliensis, "A critique of structure from motion algorithms," tech. rep., NEC Research Institute, Princeton, N.J., 2000.

[57] R. Szeliski and S. B. Kang, "Recovering 3D shape and motion from image streams using non-linear least squares," in *IEEE Conference on Computer Vision and Pattern Recognition*, (Los Alamitos, CA), IEEE Computer Society, IEEE Computer Society Press. (New York), June 1993.

[58] T. Huang and A. Netravali, "Motion and structure from feature correspondences: A review," *Proceedings of the IEEE*, vol. 82, 1994.

[59] T. J. Broida, S. Chandrashekhar, and R.Chellappa, "Recursive estimation of 3d motion from a monocular image sequence," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 16, no. 4, pp. 639–656, 1990.

[60] B. K. P. Horn, "Relative orientation," *International Journal of Computer Vision*, vol. 4, no. 1, pp. 59–78, 1990.

[61] J. Oliensis and J. I. Thomas, "Incorporating motion error in multi-frame structure from motion," in *In IEEE Workshop on Visual Motion*, (Los Alamitos, CA), pp. 8–13, IEEE Computer Society, IEEE Computer Society Press, October 1991.

[62] S. Soatto, P. Perona, R. Fraezza, , and G. Picci, "Recursive motion and structure estimation with complete error characterization," in *IEEE Conference on Computer Vision and Pattern Recognition*, (Los Alamitos, CA), pp. 428–433, IEEE Computer Society, IEEE Computer Society Press.(New York), June 1993.

[63] T. J. Broida and R. Chellappa, "Estimation of object motion parameters from noisy images," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 8, pp. 90–99, January 1986.

[64] L. Matthies, R. Szeliski, and T. Kanade, "Kalman filter-based algorithms for estimating depth from image sequences," Tech. Rep. CMU-RI-TR-88-01, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, January 1988.

[65] G. Young and R. Chellappa, "3-D motion estimation using a sequence of noisy stereo images: Models, estimation and uniqueness," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 12, pp. 735–759, January 1990.

[66] J. Ström, T. Jebara, S. Basu, and A. Pentland, "Real time tracking and modeling of faces: An EKF-based analysis by synthesis approach," in *Proc. Modeling People Workshop (ICCV'99).*

[67] A. Yao and A. Calway, "Uncalibrated narrow baseline augmented reality," in *International Symposium on 3D Data Processing Visualization and Transmission (3DPVT'02*, 2002.

[68] Alias Software, http://www.alias.com.

[69] B. Horn, *Robot Vision.* MIT Press, 1986.

[70] J. Schmidt and H. Niemann, "Using quaternions for parametrizing 3D rotations in unconstrained nonlinear optimization," in *Chair for Pattern Recognition (Informatik 5).*

[71] A. Watt and M. Watt, *Advanced Animation and Rendering Techniques.* Addison-Wesley, 1992.

[72] K. Shoemake, "Quaternions," tech. rep., Dept. Computer and Information Science, Univ. Pennsylvania, Philadelphia.

[73] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Trans. ASME, Series D, J. Basic Eng.*, vol. 82, pp. 33–45, March 1960.

[74] G. Hager and K. Toyama, "X Vision: A portable substrate for real-time vision applications," *CVIU*, vol. 69, pp. 23–37, January 1998.

[75] C. Tomasi and T. Kanade, "Detection and tracking of point features," Tech. Rep. CMU-CS-91-132, Carnegie Mellon University, Pittsburg, Pennsylvania, 1991.

[76] Z. Zhang, "Estimating motion and structure from correspondences of line segments between two perspective images," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 17, no. 12, pp. 1129–1139, 1995.

[77] C. J. Taylor and D. J. Kriegman, "Structure and motion from line segments in multiple images," in *Proc. IEEE Int. Conf. on Robotics and Automation*, 1992.

[78] A. Blake, R. Curwen, and A. Zisserman, "A framework for spatio-temporal control in the tracking of visual contours," *Int. J. Comput. Vision*, vol. 11, no. 2, pp. 127–145, 1993.

[79] B. D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," in *Proc. International Joint Conference on Artificial Intelligence*, 1981, pp. 674–679.

[80] E. Trucco and A. Verri, *Introductory Techniques for 3-D Computer Vision*. Prentice Hall, 1998.

[81] S. S. Beauchemin and J. L. Barron, "The computation of optical flow," *ACM Computing Surveys*, vol. 27, no. 3, pp. 433–467, 1995.

[82] J. Barron, D. Fleet, S. Beauchemin, and T. Burkitt, "Performance of optical flow techniques," *CVPR*, vol. 92, pp. 236–242.

[83] B. K. P. Horn and B. G. Schunck, "Determining optical flow," *Artificial Intelligence*, vol. 17, pp. 185–204, 1981.

[84] E. Simoncelli and E. Adelson, "Computing optical flow distributions using spatio-temporal filters," Tech. Rep. 165, M.I.T. Media Lab Vision and Modeling, 1991.

[85] D. Heeger, "Optical flow using spatiotemporal filters," *International Journal of Computer Vision*, 1988.

[86] D. J. Fleet and A. D. Jepson, "Computation of component image velocity from local phase information," *International Journal of Computer Vision*, pp. 77–104, 1990.

[87] J. Weber and J. Malik, "Robust computation of optical flow in a multi-scale differential framework," *International Journal of Computer Vision*, vol. 14, pp. 67–81, Jan 1995.

[88] D. B. Gennery, "Visual tracking of known three-dimensional objects," *International Journal of Computer Vision*, vol. 7, no. 3, pp. 243–270, 1992.

[89] G. D. Hager, G. Grunwald, and K. Toyama, "Feature-based visual servoing and its application to telerobotics," in *Intelligent Robots and Systems* (V. Graefe, ed.), 1995.

[90] A. Lanitis, C. Taylor, and T. Cootes, "Automatic interpretation and coding of face images using flexible models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 7, pp. 742–756, 1997.

[91] S. Baker and I. Matthews, "Equivalence and efficiency of image alignment algorithms," in *Proc. of International Conference on Computer Vision and Pattern Recognition*, vol. 1, pp. 1090–1097, IEEE, 2001.

[92] S. Baker and I. Matthews, "Lukas-kanade 20 years on: A unifiying framework," *Int. Journal of Computer Vision*, vol. 56, no. 3, pp. 221–255, 2004.

[93] H. Shum and R. Szeliski, "Systems and experiment paper: Construction of panoramic image mosaics with global and local alignment," *IJCV*, vol. 36, pp. 101–130, February 2000.

[94] A. Jepson, D. Fleet, , and T. El-Maraghi, "Robust online appearance models for visual tracking," in *Int. Conf. on Computer Vision and Pattern Recognition*, vol. I, pp. 415–422, IEEE, 2001.

[95] D. Comaniciu, V. Ramesh, , and P. Meer, "Real-time tracking of non-rigid objects using mean shift," in *Int. Conf. on Computer Vision and Pattern Recognition*, pp. 142–149, IEEE, 2000.

[96] S. Birchfield, "Elliptical head tracking using intensity gradients and color histograms," in *Int. Conf. on Computer Vision and Pattern Recognition*, pp. 232–237, IEEE, 1998.

[97] M. J. Black and A. D. Jepson, "Eigentracking: Robust matching and tracking of articulated objects using a view-based representation," *International Journal of Computer Vision*, vol. 26, no. 1, pp. 63–84, 1998.

[98] I. Matthews and S. Baker, "Active appearance models revisited," *IJCV*, 2004.

[99] J. Fraleigh and R. Beauregard, *Linear Algebra*. Addison-Wesley, 1995.

[100] J. Shi and C. Tomasi, "Good features to track," *IEEE Conf. Comput. Vision Pattern Recognit,Seattle, WA*, June 1994.

[101] G. D. Hager and P. N. Belhumeur, "Real-time tracking of image regions with changes in geometry and illumination," in *Computer Vision and Pattern Recognition*, pp. 403–410, IEEE Computer Soc., 1996.

[102] S. Baker, R. Gross, and I. Matthews, "Lucas-kanade 20 years on: A unifying framework: Part 3," Tech. Rep. CMU-RI-TR-03-35, Carnegie Mellon University Robotics Institute.

[103] Point Grey Research, http://www.ptgrey.com.

[104] Head Tracking Ground Truth, http://www.cs.bu.edu/groups/ivc/HeadTracking/.

[105] *The Flock of Birds*, Ascension Technology Corp., P.O. Box 527, Burlington, Vt. 05402.

[106] B. Horn, "Closed-form solution of absolute orientation using unit quaternions," *Journal of the Optical Society of America*, vol. 4, no. 4, 1987.

[107] CImg– The C++ Template Image Processing Library, http://cimg.sourcefourge.net.

[108] Z. Yao, *Model-Based Coding: Initialization, Parameter Extraction and Evaluation*. PhD thesis, Department of Applied Physics and Electronics, Umeå University, 2005.

[109] Y. Matsumoto and A. Zelinsky, "Real-time face tracking system for human-robot interaction," in *Proceedings of 1999 IEEE International Conference on Systems, Man, and Cybernetics (SMC99)*, 1999.

[110] J. Alon and S. Sclaroff, "Recursive estimation of motion and planar structure," in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR 2000)*, 2000.

[111] J. Shi and C. Tomasi, "Good features to track," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR'94)*, (Seattle), June 1994.

[112] N. Papanikolopoulos, "Selection of features and evaluation of visual measurements during robotic visual servoing tasks," *Journal of Intelligent and Robotic Systems*, vol. 13, pp. 279–304, August 1995.

[113] S. Baker, R. Gross, and I. Matthews, "Lucas-kanade 20 years on: A unifying framework: Part 2," Tech. Rep. CMU-RI-TR-03-01, Carnegie Mellon University Robotics Institute.