From Pose Estimation to Structure and Motion

YU Ying-Kin

A Thesis Submitted in Partial Fulfilment
of the Requirements for the Degree of
Master of Philosophy
in
Computer Science and Engineering

Supervised by

Prof. Wong Kin-Hong

July 2004

Abstract of thesis entitled:

From Pose Estimation to Structure and Motion

Submitted by YU Ying-Kin

for the degree of Master of Philosophy

at The Chinese University of Hong Kong in July 2004

The application of computer vision techniques to movie productions has become popular in recent years. One example is the integration of cartoon synthetic characters into movie scenes, which is known as augmented reality. Traditionally, if the model structure of the scene is known, the first step is to find the pose of the camera and is called pose estimation. It requires at least 3 model-to-scene point correspondences to function. Specially designed markers may be placed into the scene for robust feature tracking. However for old video footages, in which models are neither known nor markers can be placed, a marker-less structure and motion (SAM) based algorithm is necessary.

The ultimate goal of this thesis is to develop a robust recursive SAM-based pose tracking algorithm. On the way, various approaches have been studied. Three related algorithms have been proposed. *1)* The first algorithm, a model-based pose estimation method, targets to solve the problem using genetic algorithms. It searches simultaneously for the object's pose and for a set containing the most reliable feature points. This mismatch

filtering strategy makes the algorithm robust under the presence of point mismatches and outliers. *2)* The second algorithm, a recursive SAM approach, aims to find an efficient solution to the problem of structure and motion. In that, both the structure and motion of an object are recovered simultaneously. Two sets of extended Kalman filters, a set for structure refinement and another set for pose estimation, are used in the algorithm. They are executed in an interleaved manner so that the computation time can be reduced. *3)* To reach the final goal, the third algorithm is an improved SAM method, which is an extension of the second approach. It improves the accuracy of the recovered pose sequences by employing three extended Kalman filters, (each describes a frequently occurring camera motion in real situations: general, pure translation, pure rotation) based on an interacting multiple model (IMM) framework. To evaluate the proposed approaches, both analytical and empirical comparisons with the existing methods, such as the extended Lowe's method, the recursive solution by Azarbayejani and Pentland, Lowe's method and the genetic algorithm by Hati and Sengupta, are made when appropriate. It is shown in the experiments that the proposed approaches outperformed other similar algorithms. In addition, they have been applied to 3D model reconstruction and augmented reality applications to demonstrate their performances in real situations. In that, the full 360^o view of a paper box model has been recovered successfully. Also, a synthetic car has been successfully inserted into a real image sequence that contains complicated and discontinuous camera motions.

論文摘要

題目：從姿勢測估到物體結構及運動計算

作者：余英健

計算機視覺技術於電影製作上的應用在近年愈趨普遍，例如整合影片中的虛擬卡通人物與現實場景，這稱爲混合真實。如果場景的結構是預先知道的，首個步驟是計算場景的姿勢，這稱爲姿勢測估，而傳統的算法利用至少三點去計算場景的姿勢。我們亦可使用特別設計的標記來追蹤場景中的特徵，從而令測出的姿勢更加準確。不過，舊有的影片中沒有特殊標記，其中場景的結構亦不可能知道，在這情況下便需要使用能夠同時計算物體結構及運動的算法。

本論文的最終目的是發展出一個準確並基於遞歸的結構及運動算法。在論文中，我們將探討各種現有的方法，並提出三種新的算法。一、首個算法是利用遺傳基因演算法解決基於模型的姿勢測估問題，這新方法同時搜尋物件的姿勢以及一個包含最多可靠點的集合，當中的錯配過濾策略成功地令整個算法在點錯配和奇異值存在的情況下正常運作。二、算法二的目標是有效地解決結構及運動問題，在過程中物件的姿勢和結構同時被復原。我們使用兩組擴張卡爾曼濾波器，一組負責姿勢測估，另一組則負責結構計算，它們被交錯執行，從而減少所需的計算時間。三、爲了達到最終的目標，我們延伸了方法二，爲的是改善運算出來的姿勢序列準確度。此算法運用三個擴張卡爾曼濾波器，分別敘述一種常見的攝影機運動模式—綜合模式、純移動模式和純旋轉模式。它們被嵌入在互動複合模型內，用以追蹤物件的姿勢。爲了評定文中所提出的方法是否合理，我們適當地把它們與其中一些現存的算法作分析和比較，結果顯示本論文提出的算法優於同類型的算法。再者，我們亦把這些方法應用於三維模型再建以及混合真實上，成功地復原了整個紙盒模型，以及成功地把一虛擬汽車模型插入一套含有複雜攝影機運動的影片之中。

Acknowledgements

I would like to thank my supervisor, Prof. Wong Kin-Hong, for teaching me knowledge, research skills and giving me help and guidance in the completion of this thesis and several related publications. More importantly, he initialized the 3D model reconstruction project and started the vision research group so that I can have the opportunity to carry out my research.

Thanks to Prof. Chang Ming-Yuen Michael for giving me inspiration and assistance. He also involved in the implementation of Lowe's method and the extended Lowe's method, which were used for comparison in the experiments.

Thanks to Prof. Wong Tien-Tsin and Prof. Heng Peng-Ann for their suggestions on my thesis and to be the markers in my M.Phil study.

Last but not the least, I would like to thank the Sir Edward Youde Memorial Fund Council, which awarded me the Sir Edward Youde Memorial Fellowship twice in my graduate study. The fellowship, to me, is a great reward and encouragement for my efforts on researches and studies in the past two years.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation and Objectives

The application of computer vision based movie productions has become very popular in recent years. One example is the integration of cartoon synthetic characters into movie scenes. Figure 1.1 gives an illustration of the idea. The picture on the left is the original image taken with a web-camera. The picture on the right is the resulting image. In that, a synthetic car, which is drawn by wire-frames, has been inserted. Originally, producing such a kind of images involves tedious work and requires experienced photo editors. With the aid of computers, this process can be automated. This is known as augmented reality or mixed reality in computer science.

There are various real-life augmented reality applications besides movie productions. Some of the examples include direct marketing in electronic commerce [54], advanced tele-conference systems [53], manufacturing and assembling of doorlocks [52], games and entertainment [55]. To put a virtual object into the

Figure 1.1: An example of augmented reality. A synthetic car, which is drawn by wire-frames, has been put on top of the yellow box in the real scene.

real scene in a video, the first step is to compute the camera motion. With the motion parameters, the synthetic object can be rotated and translated in a way that is consistent with the background.

Traditionally, if the model structure of the scene is known, at least 3 model-to-scene point correspondences are required to compute the motion. This is known as pose estimation. Specially designed markers may be placed into the scene for robust feature tracking. Those systems mentioned in the last paragraph are some of the examples. However for old video footages, in which models are neither known nor markers can be placed, a marker-less structure and motion (SAM) based algorithm is necessary. The ultimate goal of this thesis is to develop a robust recursive SAM-based pose tracking algorithm. With SAM algorithms, it is feasible to proceed further to reconstruct the 3D model of a scene from a video sequence. One novel idea is to implement the algorithm in a digital videodisc player such that the audiences are allowed to change their viewpoints.

This thesis has three objectives. *1)* A pose estimation approach—The first objective is to make the computation of the scene's pose accurate in a high noise environment. A model-based pose estimation approach using genetic algorithms has been proposed. The importance of the method is that pose estimation is one of the two essential steps in the interleaved bundle adjustment scheme for the SAM problem. With a better pose estimation algorithm, the recovered structure can reach a higher accuracy. *2)* A recursive SAM approach—We proceed one step forward to acquire the structure, in addition to the pose sequence, from 2D images. We also target to make the algorithm able to process long sequences containing a large number of point features by personal computers within a reasonable time limit. Thus, a fast recursive SAM algorithm based on Kalman filtering has been formulated. *3)* An improved SAM approach—The third objective is to improve the accuracy of the pose sequences resulting from the second algorithm. At the same time, the required computation is kept to a minimum. Therefore, a new algorithm that makes use of the Interacting Multiple Model has been proposed.

1.2 Problem Definition

The problem of pose estimation is defined as follows. Given the object's structure, the aim is to compute the orientation of the object with respect to the camera. The pose estimation problem can be extended to compute the pose of an object with respect to a continuous sequence of images. Such an extension to the

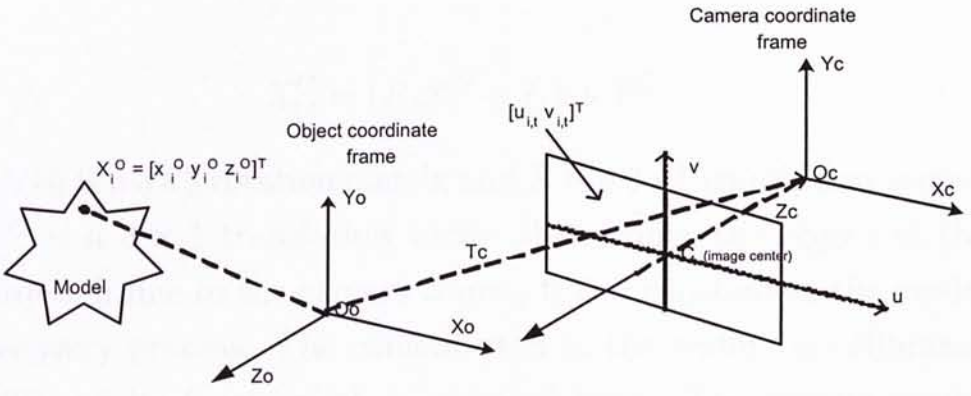

Figure 1.2: The geometry of the system.

problem is known as pose tracking. Usually, the structure of a natural object or an object created by the third parties is unknown. For these cases, the structure should be computed at the time when the pose sequence is tracked. The problem of simultaneous recovery of structure and pose is called structure and motion (SAM).

To understand the pose estimation and the structure and motion problem mathematically, let us look at the geometry of the system, which is shown in figure 1.2. $X_i^O = [x_i^O, y_i^O, z_i^O]^T$ denotes the coordinates of the point X_i with respect to the object coordinate frame. The notation $X_{i,t}^C = [x_{i,t}^C, y_{i,t}^C, z_{i,t}^C]^T$ represents the coordinates of point X_i in the camera coordinate frame at time-step t. A point on the image plane at the t^{th} frame is denoted by $p_{i,t} = [u_{i,t}, v_{i,t}]^T$. The target object is centered at the origin O_o of the object coordinate frame. The relationship between the object frame and the camera frame can be described by the following equation:

$$X_{i,t}^C = (R_t X_i^O + T_t) + T^C \tag{1.1}$$

R_t is a 3×3 rotation matrix and T_t is a 3×1 translation vector. T^C is a 3×1 translation vector that brings the object in the object frame to the camera frame. It is a constant in the model recovery process. The camera used in the system is calibrated [40] and its focal length is assumed fixed. The camera model is full perspective and the projection can be mathematically represented as:

$$\begin{pmatrix} u_{i,t} \\ v_{i,t} \end{pmatrix} = \frac{f}{z_{i,t}^C} \begin{pmatrix} x_{i,t}^C \\ y_{i,t}^C \end{pmatrix} \tag{1.2}$$

where f is the focal length of the camera.

Definition of the Structure and Motion Problem In technical terms, the problem of structure and motion is to recover the coordinates of model point X_i^O in the object coordinate frame and the pose of the object, i.e. the rotation R_t and translation T_t, with respect to the views at each time-step such that the back-projection error (BE), which is defined as:

$$BE = \sum_{i=1}^{N} d_i^2 \tag{1.3}$$

is to be minimized. In the equation, d_i is the difference in distance between a real image point and its corresponding back-projected point.

Definition of the Pose Estimation Problem The problem of pose estimation can be regarded as a constrained problem of structure and motion. Since the structure is given, the only task is to find out the rotation R and translation T of the object with respect to a single image. The problem of pose tracking means computing the pose sequence R_t and T_t with respect to a continuous sequence of images.

1.3 Contributions

The contributions of this thesis are four-folded. They are summarized as follows:

1) Robust Pose Estimation A model-based pose estimation method using genetic algorithms has been proposed and implemented. The algorithm searches simultaneously for the pose of the object and for a set containing the most reliable feature points. This mismatch filtering strategy successfully makes the algorithm more robust under the presence of point mismatches and outliers in the images. It outperformed Lowe's method [41] and another genetic algorithm by Hati and Sengupta [21] in terms of accuracy under the presence of point mismatches and outliers.

2) High Speed Structure and Motion Recovery A high-speed two-step method to recover structure and motion from continuous image sequences based on Kalman filtering has been proposed. The structure refinement and the pose estimation step are executed in an interleaved sense so that the computation

time is greatly reduced. The implementation can handle the changeable set of point features. It outperformed the interleaved bundle adjustment method [33] and the recursive algorithm by Azarbayejani and Pentland [8] in terms of computation speed without loss in the quality of model reconstruction.

3) Robust High Speed Structure and Motion Recovery The Interacting Multiple Model has been introduced to the problem of structure and motion. The proposed approach uses three extended Kalman filters, each describes a frequently occurring camera motion in real situations (general, pure translation, pure rotation), to track the pose of an object within the Interacting Multiple Model framework. Since the ambiguities among the recovered structure and recovered pose parameters have been resolved, the acquired pose sequence is more accurate than the existing approaches. Comparisons with different approaches show that the proposed method is more efficient and accurate.

4) 3D Model Reconstruction and Augmented Reality Applications The three proposed algorithms have been applied to 3D model reconstruction and augmented reality in real situations. In that, the full 360^o view of a paper box model has been recovered successfully. Also, a synthetic car has been successfully inserted into a real image sequence that contains complicated and discontinuous camera motions. Related experimental results using real images can be found in section 3.8.2, 4.7.2 and 5.7.2.

1.4 Related Publications

With regard to contribution stated in the last section, the following papers have been submitted to or published in international journals and conferences. They are categorized according to the chapter that they appear:

Chapter 3:

Ying-Kin Yu, Kin-Hong Wong and Michael Ming-Yuen Chang, "Pose estimation using genetic algorithms for augmented reality applications", submitted to a journal.

Chapter 4:

1) **Ying-Kin Yu**, Kin-Hong Wong and Michael Ming-Yuen Chang, "Recursive 3D model reconstruction based on Kalman filtering", IEEE Transactions on Systems, Man and Cybernetics—Part B. (to appear)
2) **Ying-Kin Yu**, Kin-Hong Wong and Michael Ming-Yuen Chang, "A fast recursive 3D model reconstruction algorithm for multimedia applications", in proceedings of the International Conference on Pattern Recognition 2004 (ICPR-2004), Cambridge, August 2004. (to appear)
3) **Ying-Kin Yu**, Kin-Hong Wong and Michael Ming-Yuen Chang, "Recursive 3D model reconstruction based on Kalman filtering", in proceedings of the Hong Kong

Chapter of Signal Processing Postgraduate Forum 2004, Hong Kong, May 2004.

Chapter 5:

1) **Ying-Kin Yu**, Kin-Hong Wong and Michael Ming-Yuen Chang, "A fast and robust simultaneous pose tracking and structure recovery algorithm for augmented reality applications", in proceedings of the IEEE International Conference on Image Processing 2004 (ICIP-2004), Singapore, October 2004. (to appear)

2) **Ying-Kin Yu**, Kin-Hong Wong and Michael Ming-Yuen Chang, "Merging artificial objects with marker-less video sequences based on the interacting multiple model method", submitted to a journal.

1.5 Organization of the Paper

The rest of this paper is organized as follows. A background on the previous work related to the pose estimation and the structure and motion problem are introduced in chapter 2. In chapter 3, an approach that tackles the pose estimation problem using genetic algorithms is discussed. In chapter 4, a recursive 2-step structure acquisition algorithm based on Kalman filtering is presented. In chapter 5, a natural extension to the algorithm in chapter 4 using the Interacting Multiple Model is proposed. In chapter 6, empirical comparisons among several structure and motion algorithms are made. The experimental results from the

2-step recursive algorithm in chapter 4, the Interacting Multiple Model based approach in chapter 5, the recursive solution proposed by Azarbayejani and Pentland [8] and the interleaved bundle adjustment method [33] are compared. In the last chapter, the conclusion of the achievements is made.

□ **End of chapter.**

2-step recursive algorithm in chapter 4, the Interacting Multiple Model based approach in chapter 5, the recursive solution proposed by Azarbayejani and Pentland [8] and the interleaved bundle adjustment method [33] are compared. In the last chapter, the conclusion of the achievements is made.

□ **End of chapter.**

Chapter 2

Background

2.1 Introduction

There are various techniques to deal with the pose estimation and the structure and motion problem. Here we are going to discuss some of the existing solutions. In the first part, a general overview of the previous work on pose estimation is given. In addition, two pose estimation methods, namely Lowe's method [41] and the genetic algorithm by Hati and Sengupta [21], are discussed in details. These two methods were implemented and compared with the proposed approach in chapter 3. In the second part, existing approaches that tackle the structure and motion problem are introduced. In particular, the details of the extended Lowe's method by Chang and Wong [33] and the extended Kalman filter by Azarbayejani and Pentland [8] are given. These two algorithms were also implemented and empirical comparisons were made with the approaches proposed in chapter 4 and 5.

2.2 Pose Estimation

2.2.1 Overview

In short, the problem of pose estimation is to compute the orientation of the object with respect to the camera given the image and the 3D structure of the model. Mathematically, the rotation R and translation T of the object, which are shown in figure 1.2 on page 4, are to be estimated.

Let us have a glance on the previous work on the pose estimation problem. Early pose estimation algorithms make use of a small number of point features in the scene. Fishler and Bolles [36] took three feature points with the "Random Sample Consensus" method to compute the pose of an object. Horaud et al [37] estimated the pose of an object using four non-coplanar points. The solution is computed by solving biquadratic polynomial equations of one unknown with geometric constraints. Another four-point algorithm was proposed by Liu and Wong [43].

The iterative steepest descent method [41] [44] is also a common approach to solve the pose estimation problem. In that, nonlinear optimization schemes, such as Newton's method, are used to estimate the object's pose by minimizing the difference between the object's re-projection and the real image measurements. The optimization procedure is executed iteratively until the image residual error is small and within a predefined limit.

Another approach is to tackle the problem by Kalman filtering. Lippiello et al [31] [32] used extended Kalman filter for motion estimation. Their algorithms find the pose of the ob-

ject based on a known CAD model from stereo images. The position and orientation of the camera are recovered in realtime and the results are applied to visual servoing of robot manipulators. Wong et al [27] applied the CONDENSATION framework to track the pose of an object for the construction of a virtual walk-through environment.

Genetic algorithms are alternatives to the traditional solutions. Most of the existing methods [21] [23] [24] model the problem as a camera calibration procedure. Hati and Sengupta [21] used the genetic algorithm framework to estimate the extrinsic parameters of a camera. The work by Ji and Zhang [23] is quite similar to [21] but the authors applied a genetic algorithm to search for both the intrinsic and extrinsic parameters of a camera. They incorporated a time dependent function for the adjustment of the step size into the mutation operator to facilitate the convergence of the fitness value. Cerveri et al described an approach in [24] that calibrates a stereo camera system with the enhanced evolutionary search. The only work that directly addresses the pose estimation problem with genetic algorithms is by Toyama et al [22]. The inputs to their algorithm are the edge images instead of point features. They adopted the phenotypic forking genetic algorithm [29] to perform the search. This strategy combines the advantages of conventional genetic algorithms and steepest descent methods such that the solution can be found within a smaller number of generations.

2.2.2 Lowe's Method

Among all the existing pose estimation algorithms, Lowe's method [41] is one of most famous approaches and can be regarded as one of the standard solutions to the problem. It is actually a steepest descent method. It searches for a combination of pose parameters such that the re-projection of the object fits the image measurements best. The underlying assumption of the method is that the object's structure is known in prior.

To initialize the algorithm, an initial guess that is close to the solution is required. Let the initial guess be R_0 and T_0, which are respectively the rotation matrix and translation vector as defined in figure 1.2 on page 4. We assume that this guess is not too far from the true solution. The image residues of the guess can be written as:

$$\delta u_{i,t} = G_i(R_0, T_0) - u_{i,t} \tag{2.1}$$

$$\delta v_{i,t} = G_i(R_0, T_0) - v_{i,t} \tag{2.2}$$

$G_i(R, T)$ is the projection function of a 3D point X_i after the R and T transformations. $u_{i,t}$ and $v_{i,t}$ are the real image measurements at time-step t. To compute the pose of the object at time-step t, let the translation vector be $T_t = [t_x, t_y, t_z]^T = [t_1, t_2, t_3]^T$ and the rotation angles about the three axes be $\phi_t = [\alpha, \beta, \gamma] = [\phi_1, \phi_2, \phi_3]$ (i.e. the Yaw, Pitch and Roll angle respectively). The image residues at time-step t can be approximated by the following first order expansion:

$$\sum_{j=1}^{3}[\frac{du_{i,t}}{dt_j}\Delta t_j + \frac{du_{i,t}}{d\phi_j}\Delta\phi_j] = \delta u_{i,t}$$
$$\sum_{j=1}^{3}[\frac{dv_{i,t}}{dt_j}\Delta t_j + \frac{dv_{i,t}}{d\phi_j}\Delta\phi_j] = \delta v_{i,t} \quad (2.3)$$

With at least three point correspondences, the six unknowns of system 2.3 (i.e. Δt_1, Δt_2, Δt_3, $\Delta\phi_1$, $\Delta\phi_2$ and $\Delta\phi_3$) can be solved. The new estimates of the object's pose, R_t and T_t, can be computed by:

$$T_t = T_{t-1} + \Delta T_t \quad (2.4)$$

R_t is calculated by repeated multiplications of R_t with the three rotation matrices defined by the correction angles $\Delta\phi_1$, $\Delta\phi_2$, $\Delta\phi_3$. The above procedure should be iterated for at least five times in order to obtain an accurate pose of the object. By individually computing the object's pose in each image in an image sequence, the camera motion can be tracked.

2.2.3 The Genetic Algorithm by Hati and Sengupta

Unlike the steepest descent method introduced in the previous section, genetic algorithms are made to avoid local optima and find a global optimum in the search. So, it has also been applied to camera calibration and pose estimation. The genetic algorithm by Hati and Sengupta [21] is one of the examples. Their method can be regarded as a model-based pose estimation approach. The purpose of using genetic algorithms is to make

their approach immune to noise and the presence of outlying point features. The implementation of their genetic algorithm is outlined in this section. Related mathematical details are discussed in chapter 3 on the way we present the proposed genetic algorithm.

The idea of searching with genetic algorithms is as follows. Starting with an initial pool of population produced randomly, the chromosomes in the current population have a certain chance to reproduce their offsprings. Fitter chromosomes have a higher chance to be selected for reproduction. An offspring is reproduced either by mutation of one chromosome or crossover between two chromosomes. The algorithm stops until the fitness of the best chromosome converges to a desired value. The solution of the search can be extracted from the chromosomes that survive at the end. Usually, a genetic algorithm takes a much longer period of time to compute the solutions compared to any other iterative steepest descent methods.

In the genetic algorithm by Hati and Sengupta [21], the search space is defined by the three translation parameters (i.e. t_x, t_x, t_z) and the three rotation angles (i.e. Yaw, Pitch and Roll angle). These parameters are encoded in the chromosomes using real numbers within a predefined interval. Overlapping population has been considered in their algorithm. It means that a temporary population of offsprings is created and is added to the previous population. The worse group of chromosomes is then removed to return the population size to the original one. Thus, the population of the next generation consists of a certain portion of chromosomes from the parent generation and

some from their offsprings. The proportion between two types of chromosomes in the new generation is defined by the probability of replacement. The fitness of a chromosome is calculated based on the back-projection error defined in equation 1.3 on page 5. For the ease of chromosome selection, it is further scaled using sigma-truncated scaling method. The roulette-wheel proportionate selection scheme, which is described in details in section 3.6, is used in their genetic algorithm.

The genetic operations are defined by two operators: the mutation and crossover operator. The mutation operation is actually a Gaussian mutator, which picks a new value based on a Gaussian distribution around the current value. If the new value is less than the lower bound of the parameter, it is replaced by the lower bound. It is similar if the new value exceeds the upper bound. For the crossover operator, the one-point Blend crossover [20] is used. Intuitively, this operation generates a new value based on the interval between the parents. Besides, the size of the population in each generation is 300. The probability of mutation and crossover are 0.01 and 0.90 respectively. A summary of the genetic algorithm parameters can be found in table 2.1.

2.3 Structure and Motion

2.3.1 Overview

The objective of the structure and motion problem is to recover both the pose and structure of a scene from a sequence of 2D

Parameters	*Values*
Population size	300
Probability of mutation	0.01
Probability of crossover	0.90
Probability of replacement	0.5

Table 2.1: A table summarizes the parameters of the genetic algorithm by Hati and Sengupta

images. Specifically, the model point X_i^O, as shown in figure 1.2 on page 4, in the object coordinate frame and the pose of the object, i.e. the rotation R_t and translation T_t, with respect to the views at each time-step are to be computed.

There are various approaches to tackle the structure and motion problem. One of the most popular approaches is by epipolar geometry [14]. With known correspondences between two views, a constraint, which is known as the epipolar constraint, between these views can be set up. This constraint can be computed and is encoded in a matrix called the Fundamental matrix if the camera is fully calibrated. The Fundamental matrix contains the camera parameters, such as the focal length, the center of projection, together with the pose parameters between these two views. It can be used to recover the camera motion up to a scale factor. Based on the triangulation of the point features in the corresponding images, an Euclidean structure of the scene can be found by solving a system of equations. In a similar sense, the technique has been extended to three views or more [13] [15] [5].

Factorization [2] [3] is another common approach to tackle

the problem of structure and motion. The main idea of the factorization method is that the image measurements, which are contained in the registered measurement matrix, are factorized into two components, such that one of the resulting matrices represents the camera motion while another represents the scene's structure. Classical factorization methods work under the assumption of orthographic projection. The work in [2] is an example. The factorization method has been extended to tackle the para-perspective projection camera model [16] and handle the reconstruction and pose estimation of multiple independently moving objects [3]. Bundle adjustment is also an effective method [35]. It minimizes the re-projection error between the estimated model and the image measurements. The minimization procedure can be achieved in batch either by the Newton's or Levenberg-Marquardt iteration. A branch of it is the interleaved bundle adjustment as described in [33] and [35]. It breaks up the minimization problem into two steps so as to reduce the size of the Jacobian involved, resulting in speeding up the algorithm. The methods mentioned previously tackle the problem in a batch, in which the structure and motion are optimized for all the images at one time.

There are solutions that recover the structure and motion in a sequential way. This is known as recursive or casual structure and motion. In that, images from a continuous sequence are processed one by one instead of in a batch. Most of these recursive solutions are based on Kalman filtering. Among them, a portion of solutions use Kalman filters to integrate the scene's structure. For example, the iterated extended Kalman filter

(IEKF) is adopted for updating the structure in Euclidean [1] or projective framework [30]. In that, the pose and the structure of the object are recovered alternately by a RANSAC-based equation solving technique and the IEKF. Thomas and Oliensis applied standard Kalman filtering to fuse the recent structure estimates found by Horn's algorithm using the most recent image pair with the previous structure estimates at each time-step [9].

Unlike the approaches in the last paragraph, in which Kalman filters are used solely for structure updating, the series of methods in [7] [8] [10] [11] [12] recover both the structure and motion at one time in each time-step in a recursive manner. The work by Broida et al [11] is the ancestor of this series of researches. They applied a single IEKF to recover the structure and pose of the object. Azarbayejani and Pentland described a method in [8] that makes significant improvements over [11]. Extension is made to recover the focal length of the camera in addition to the pose and structure. The most recent work of recursive structure recovery is by Chiuso et al [7]. Similar techniques in structure and motion have also been applied to simultaneous localization and map-building for robot navigation [51].

2.3.2 The Extended Lowe's Method

Full scale bundle adjustment is an effective method [35] to compute the structure and the pose of a scene from a sequence of 2D images. It minimizes the re-projection error between the estimated model and the image measurements by adjusting the

structure and pose sequence at one time. Usually, it is employed to fine tune the final solution after an initial guess, which can be made by using epipolar geometry [14] [25] or the factorization method [2]. A particular form of the full scale bundle adjustment is the interleaved bundle adjustment method. It has higher computation speed than the conventional approach since the Jacobian involved in the calculation is broken down into small pieces. The extended Lowe's method by Chang and Wong [33] that we are going to discuss is an example.

The extended Lowe's method extends the original Lowe's method, which is for model-based pose estimation, to recover both the structure and pose of a scene. It consists of two major passes. The first pass estimates the pose sequence for all the frames of the object based on the current guess of the 3D model. The second pass updates the model of the last guess with the newly acquired pose sequence. After a number of iterations (alternation between the two passes), the initial model converges to the final solution. The overview of the algorithm can be summarized in the following pseudo-code:

Do the following until the total re-projection error of the model is less than a threshold or the number of iteration is too large

First pass: The pose sequence of the model with respect to the Γ frames is estimated using Lowe's method.

Second pass: The model points of the previous estimate are refined using Newton's method to minimize the image residual error.

The details of Lowe's method in the first pass has been described in section 2.2.2 on page 14 and is not repeated here. For the second pass, the computation of the structure can be achieved by first deriving the image residual as in the following first order expansion:

$$\delta p_{i,t} = \frac{G_i(R_t, T_t)}{dx_i}\delta x_i + \frac{G_i(R_t, T_t)}{dy_i}\delta y_i + \frac{G_i(R_t, T_t)}{dz_i}\delta z_i \quad (2.5)$$

As in section 2.2.2, $G_i(R_t, T_t)$ is the function that projects the 3D point X_i onto the image plane with R_t and T_t transformations. Please refer to the geometric model in section 1.2 on page 4 for the definitions of the other notations. Separating $\delta p_{i,t}$ into two components and calculating the values of the partial derivatives, which is represented by $l_{mn}^{i,t}$, the image residual error of the i^{th} point in the t^{th} frame can be rewritten as the following simultaneous equations:

$$\delta u_{i,t} = l_{11}^{i,t}\delta x_i + l_{12}^{i,t}\delta y_i + l_{13}^{i,t}\delta z_i \quad (2.6)$$

$$\delta v_{i,t} = l_{21}^{i,t}\delta x_i + l_{22}^{i,t}\delta y_i + l_{23}^{i,t}\delta z_i \quad (2.7)$$

where

$$\begin{aligned} l_{11}^{i,t} &= f[\frac{r_{11}}{z_i} - \frac{r_{31}x_i}{z_i^2}], l_{12}^{i,t} = f[\frac{r_{12}}{z_i} - \frac{r_{32}x_i}{z_i^2}] \\ l_{13}^{i,t} &= f[\frac{r_{13}}{z_i} - \frac{r_{33}x_i}{z_i^2}], l_{21}^{i,t} = f[\frac{r_{21}}{z_i} - \frac{r_{31}y_i}{z_i^2}] \\ l_{22}^{i,t} &= f[\frac{r_{22}}{z_i} - \frac{r_{32}y_i}{z_i^2}], l_{23}^{i,t} = f[\frac{r_{23}}{z_i} - \frac{r_{33}y_i}{z_i^2}] \end{aligned} \quad (2.8)$$

r_{mn} is the component of the rotation matrix R_t. For Γ number of frames, the following linear system can be derived:

$$\begin{bmatrix} \delta u_{i,1} \\ \delta v_{i,1} \\ . \\ \delta u_{i,t} \\ \delta v_{i,t} \\ . \\ \delta u_{i,\Gamma} \\ \delta v_{i,\Gamma} \end{bmatrix} = \begin{bmatrix} l_{11}^{i,1} & l_{12}^{i,1} & l_{13}^{i,1} \\ l_{21}^{i,1} & l_{22}^{i,1} & l_{23}^{i,1} \\ . & . & . \\ l_{11}^{i,t} & l_{12}^{i,t} & l_{13}^{i,t} \\ l_{21}^{i,t} & l_{22}^{i,t} & l_{23}^{i,t} \\ . & . & . \\ l_{11}^{i,\Gamma} & l_{12}^{i,\Gamma} & l_{13}^{i,\Gamma} \\ l_{21}^{i,\Gamma} & l_{22}^{i,\Gamma} & l_{23}^{i,\Gamma} \end{bmatrix} \begin{bmatrix} \delta x_i \\ \delta y_i \\ \delta z_i \end{bmatrix} \tag{2.9}$$

The motion of the camera R_t and T_t are encoded in $l_{mn}^{i,t}$, which is acquired in the first pass of the algorithm. Thus, the vector $[\delta x_i, \delta y_i, \delta z_i]^T$ in system 2.9 can easily be solved using the singular value decomposition technique [18]. Thus, the 3D model points can be updated as:

$$[x_i, y_i, z_i]^T = [\hat{x}_i, \hat{y}_i, \hat{z}_i]^T + [\delta x_i, \delta y_i, \delta z_i]^T \tag{2.10}$$

$[\hat{x}_i \hat{y}_i \hat{z}_i]^T$ is the previous estimate of the model point. In this way, the 3D structure of a scene can be recovered.

2.3.3 The Extended Kalman Filter by Azarbayejani and Pentland

Among all the Kalman filter based solutions mentioned in section 2.3.1, the extended Kalman filter (EKF) by Azarbayejani

and Pentland [8] can be regarded as one of the representatives to the structure and motion problem in the past decade. In their algorithm, a single EKF is used to recover the pose and structure of an object in t Kalman filtering cycles for a sequence of t images. Since their algorithm has been compared with the proposed approaches in the following chapters, some of the related mathematical details are illustrated.

Firstly, let us investigate the camera model used in their EKF. The camera model can be written mathematically as:

$$\begin{bmatrix} u_{i,t} \\ v_{i,t} \end{bmatrix} = \begin{bmatrix} x_{i,t}^C \\ y_{i,t}^C \end{bmatrix} \frac{1}{1 + \frac{z_{i,t}^C}{f}} \tag{2.11}$$

The major difference between this model and the traditional one (i.e. the camera model defined by equation 1.2) is that the origin of this coordinate system is fixed at the image plane.

To increase the algorithm stability, special modifications are made to the structure representation. In their EKF, one parameter X_i' is used to represent a 3D point feature X_i^O in the pointwise structure such that:

$$\begin{bmatrix} x_i^O \\ y_i^O \\ z_i^O \end{bmatrix} = \begin{bmatrix} u_{i,1} \\ v_{i,1} \\ 0 \end{bmatrix} + \frac{X_i'}{f} \begin{bmatrix} u_{i,1} \\ v_{i,1} \\ f \end{bmatrix} \tag{2.12}$$

$u_{i,1}$ and $v_{i,1}$ denote the image coordinates of the i^{th} point in the first frame of the image sequence. In other words, each 3D point is expressed in terms of its position in the first image.

Since the filter that they used is a full covariance EKF, the pose and the structure parameters are encoded in a single state vector ω, which is written as:

$$\omega = \left(t_x, t_y, \frac{t_z}{f}, \psi_X, \psi_Y, \psi_Z, X'_1, X'_2, ..., X'_N\right)^T \tag{2.13}$$

Note that $\frac{t_z}{f}$ is recovered instead of t_z in the computation. The reason is that $\frac{t_z}{f}$ does not degenerate at long focal length. ψ_X, ψ_Y and ψ_Z are the incremental rotations around the x, y and z axis respectively at each time-step.

A global rotation between the object coordinate frame and the current camera coordinate frame is kept by a unit quaternion defined as:

$$R = \begin{bmatrix} q_0^2 + q_1^2 - q_2^2 - q_3^2 & 2(q_1q_2 - q_0q_3) & 2(q_1q_3 + q_0q_2) \\ 2(q_1q_2 + q_0q_3) & q_0^2 - q_1^2 + q_2^2 - q_3^2 & 2(q_2q_3 - q_0q_1) \\ 2(q_1q_3 - q_0q_2) & 2(q_2q_3 + q_0q_1) & q_0^2 - q_1^2 - q_2^2 + q_3^2 \end{bmatrix} \tag{2.14}$$

q_0, q_1, q_2 and q_3 are the components of the unit quaternion q. The global quaternion can be updated from the incremental rotation as follows:

$$\delta q = \left((1-\tau)^{\frac{1}{2}}, \frac{\psi_X}{2}, \frac{\psi_Y}{2}, \frac{\psi_Z}{2}\right) \tag{2.15}$$

$$\tau = \frac{\psi_X^2 + \psi_Y^2 + \psi_Z^2}{4} \tag{2.16}$$

The dynamic model used in their EKF is an identity transform plus Gaussian noise. With the model, the core Kalman filtering equations can be derived. Detailed discussions about

Kalman filtering and its implementation can be found in the appendix.

In the computation, the state vector ω is updated from frames to frames in the Kalman filtering cycles. The final object structure can be obtained by decoding ω after the last image measurement. The pose sequence can also be obtained by recording the values of the global unit quaternion q at each time-step.

□ **End of chapter.**

Chapter 3

Model-based Pose Tracking Using Genetic Algorithms

3.1 Introduction

The objective of the pose tracking problem is to compute the pose of an object with respect to the images in a continuous sequence given the object's structure. This chapter aims to propose an approach to tackle the problem using genetic algorithms. The importance of pose estimation in the structure and motion problem is that it is one of the two essential steps in the interleaved bundle adjustment scheme. With a better pose estimation algorithm, the recovered structure can reach a higher accuracy. The method proposed in this chapter is suitable for applications in any robust interleaved bundle adjustment algorithms.

The proposed model-based pose tracking algorithm is based on the work by Hati and Sengupta [21]. Real numbers are used to represent the object's pose in the chromosomes. The improvement over the work in [21] is on incorporating a feature searching

strategy into the algorithm. That means while searching for the pose of an object, the algorithm also searches for the set containing the most reliable features among all the available model points in the process. To achieve this, indexes of the reliable feature points are encoded in the chromosomes. The chromosome now comprises two sections: *1)* One section encodes the translation and rotation parameters *2)* The other section encodes the indexes to the selected point features. The genetic operators are also modified to cope with this new encoding scheme.

3.2 Overview of the Algorithm

The implementation of the genetic algorithm follows the framework of the conventional genetic algorithm methodology as described in [20]. In addition, the concept of overlapping population, as indicated in step 4 and 5 in the pseudo-code below, is adopted in creating the next generation of chromosomes. The population of the next generation consists of a certain portion of the chromosomes from the parent generation and some from their offsprings, in which the proportion is defined by the probability of replacement. The overview of the proposed genetic algorithm is outlined below:

1 Generate a random population consisting of n chromosomes.

2 Calculate the fitness value of each chromosome.

3 Choose the parents from the current population using the roulette wheel proportionate selection for reproduction.

4 Create a temporary population of offsprings by the mutation or crossover of the parents according to the corresponding probabilities.

5 Select, with the roulette wheel proportionate selection, the chromosomes into the next generation from the pool of the offsprings and current generation according to the probability of replacement.

6 Repeat step 1) to step 5) until one or more of the following conditions has been reached: *i)* The fitness of the best chromosome has reached a desired value. *ii)* It has no further improvements. *iii)* The time limit exceeds.

3.3 Chromosome Encoding

The chromosomes of the proposed genetic algorithm consists of $k+6$ elements, where k is the number of selected point features. The chromosome vector b_i^t is defined as:

$$\begin{aligned} b_i^t &= (a_{1i}^t, a_{2i}^t) \\ a_{1i}^t &= (\alpha, \beta\,\gamma, t_x, t_y, t_z) \\ a_{2i}^t &= (d_1, d_2, ..., d_k) \end{aligned} \tag{3.1}$$

The notation b_i^t refers to an individual chromosome at the t^{th} generation. a_{1i}^t encodes the pose of an object. Its elements comprise the first section of the chromosome. The encoding of this section follows exactly the work of Hati and Sengupta described in [21]. Real number representation, together with an appropriate upper and lower, are adopted for each field in a_{1i}^t.

The term a_{2i}^t encodes the indexes of the feature points selected for pose estimation. Its elements comprise the second section of the chromosome. Each field in a_{2i}^t stores the index to the selected model feature. Integer representation is adopted. The value k in the implementation is equal to $\frac{N}{2}$, where N is the total number of available point features. It means that the proposed approach attempts to find a set containing the half most reliable point features in the search and relies on them for pose estimation.

3.4 The Genetic Operators

3.4.1 Mutation

The mutation operation of the first section a_{1i}^t and the second section a_{2i}^t of the chromosomes are defined differently due to their differences in their physical meanings. For the first section a_{1i}^t, a Gaussian mutator, similar to that in [21], is adopted.

$$a_{1i}^{t+1} = a_{1i}^t + U(\lambda_1^t) \tag{3.2}$$

$U(\lambda_1^t)$ is a Gaussian function which generates values with Gaussian distribution according to λ_1^t, which is the variance of a_1^t. The mutation of the second section a_{2i}^t is trivial. In this section, the genes of the parent are copied to its offspring. The content of this section is preserved in the mutation operation.

$$a_{2i}^{t+1} = a_{2i}^t \tag{3.3}$$

3.4.2 Crossover

Similar to the mutation operation, the crossover operation of the first section a_{1i}^t and the second section a_{2i}^t of the chromosomes are different. For the first section, the Blend crossover [20] is adopted. In Blend crossover, a new value between two chromosomes is generated based on the interval between the two parents. The offspring has the properties inherited from both of its parents. The Blend crossover can be expressed mathematically as:

$$a_{1i}^{t+1} = I a_{1i}^t + (1 - I) a_{1j}^t \tag{3.4}$$

I is a real number that ranges within [0,1]. For the second section a_{2i}^t of the chromosome, the crossover is achieved by combining two sections of chromosomes of the parents. To be more precise, the operation can be expressed mathematically as:

$$\begin{aligned} a_{2i}^{t+1} &= (d_{i1}, d_{i2}, ..., d_{ih}, d_{j(h+1)}, d_{j(h+2)}, ..., d_{jk}) \\ a_{2i}^{t} &= (d_{i1}, d_{i2}, ..., d_{ik}) \\ a_{2j}^{t} &= (d_{j1}, d_{j2}, ..., d_{jk}) \end{aligned} \tag{3.5}$$

h is an integer that ranges within [1,k]. This random number h defines the point of crossover of the two parent chromosomes.

3.5 Fitness Evaluation

To evaluate the fitness of each chromosome, the first section a_{1i}^t is decoded to get the pose represented by it. Then the second part a_{2i}^t is decoded to get the 3D coordinates of the selected model

features. These selected model features are back-projected to the image plane with the pose decoded from a_{1i}^t to calculate the back-projection error (BE) as defined in equation 1.3. Since it is originally a minimization problem, the fitness calculation is needed to be changed. The fitness value of the chromosome i is equal to:

$$e_i(BE) = C_{GA} - BE \tag{3.6}$$

C_{GA} is a constant chosen to be 10000. The problem is reformulated to be a maximization problem as in the conventional genetic search problem.

3.6 The Roulette Wheel Proportionate Selection Scheme

The roulette wheel proportionate selection scheme [20] is adopted for selecting the chromosomes from the current population to the mating pool for reproduction. The concept of roulette proportionate wheel selection is simple. The chromosomes that have higher fitness values will have a higher probability to be selected. It simulates natural selection in the real world. To perform roulette wheel proportionate selection in the implementation, we need to further scale the fitness function to suit the purpose. This is achieved by applying a sigma-truncated scaling to the fitness value in equation 3.6. Mathematically, sigma truncation is defined as:

$$e_i' = e_i - (\overline{e} - C_{GA}'\sigma) \tag{3.7}$$

Parameters	*Values*
Population size	300
Probability of mutation	0.4
Probability of crossover	0.6
Probability of replacement	0.7
Number of selected features	50% of the model features

Table 3.1: A table summarizes the parameters of the proposed genetic algorithm

$$\overline{e} = \frac{1}{n}\sum_{j=1}^{n} e_j \tag{3.8}$$

e'_i is the scaled fitness. $\overline{e}$ is the mean fitness of all the chromosomes in the population. σ is the standard deviation of the fitness of all chromosomes. C'_{GA} is a real constant which can be chosen such that $C'_{GA}\sigma$ is a reasonable multiple of the population standard deviation. C'_{GA} is equal to 3 in the implementation. If e'_i is negative, it is truncated to zero. With the scaled fitness, then the probability of choosing the i^{th} chromosome for reproduction is:

$$\frac{e'_i}{e'_1 + e'_2 + ... + e'_n} \tag{3.9}$$

3.7 The Genetic Algorithm Parameters

Table 3.1 summarizes the parameters used in the proposed genetic algorithm. Note that the number of selected features in each chromosome can be adjusted freely according to the actual situation without deteriorating the quality of the solution.

3.8 Experiments and Results

3.8.1 Synthetic Data Experiments

This experiment aims to demonstrate the robustness of the proposed genetic algorithm. Empirical comparisons with two other model-based pose estimation approaches, i.e. the genetic algorithm by Hati and Sengupta [21] and the traditional Lowe's method in [41], are made under the presence of point mismatches and outliers.

The specification of the object and camera is as follows. The object contains 300 random 3D feature points within a cube of volume of $0.13m^3$. It is centered at a place 0.33 meters away from the viewing camera. The camera has a focal length of 6mm. The sensor of the camera is not perfect. It imposes a 2D zero mean Gaussian noise with standard deviation 0.1 pixels on the image captured.

The object had a rate of rotation motion not more than 0.5 degrees per frame around each axis and a rate of translation motion not more than 0.05 meters along each axis. Special conditions such as point mismatches were added to each test. The test for each condition was repeated 10 times with independent data sets.

Effects of Point Mismatches In a real situation, it is quite common that some features are mistracked, resulting in wrong point correspondences between the model points and the image points. This part is to simulate the problem in such a situation. The point mismatches were generated as follows. A number of point

pairs were selected randomly from the model features. Their correspondences with the points in the 2D image were swapped. The relationship between the pose errors and the number of point mismatches was studied.

Figure 3.1 shows the effects of point mismatches to the proposed genetic algorithm. You can see that the pose errors do not depend on the percentage of mismatches. When the feature mismatch percentage is low, say smaller than 20%, its overall performance remains unchanged. The plots in figure 3.2 show a direct comparison of the three methods under the presence of point mismatches. It is obvious that the proposed genetic algorithm outperformed the other two methods. For the rotation error, it achieves an error below 0.5 degrees even for 20 percents of mismatches. The other two algorithms have error more than 5 degrees for 20 percents of mismatches. For translation error, it has only 1mm error while the other two have errors more than 1cm. You may also notice that the pose errors increase with the addition of the percentage of wrong correspondences for both the genetic algorithm by Hati and Sengupta [21] and Lowe's method [41]. From this experiment, we see that the proposed genetic algorithm was more robust under the presence of point mismatches than the other two algorithms.

Effects of Outliers Occlusion and disocclusion of a point feature in the image sequence or extracting a point feature from a reflective surface by feature trackers may cause the presence of incorrect point features in the images. This results in an inaccurate estimation of the object's pose. The relationship of the

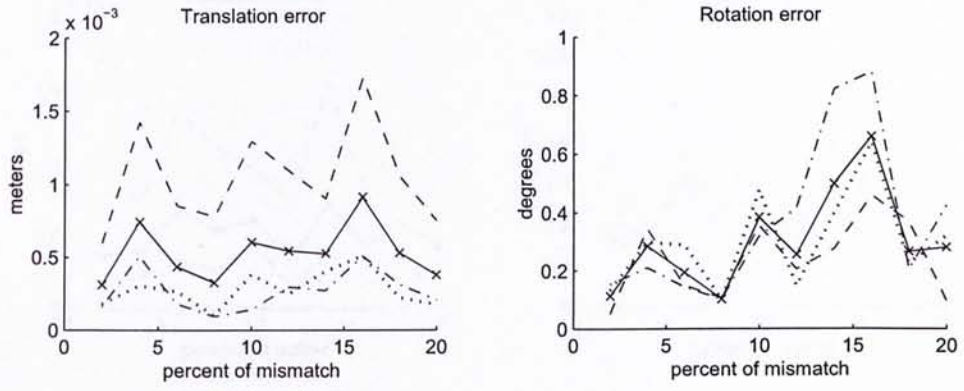

Figure 3.1: Graphs showing the effects of point mismatches to the proposed genetic algorithm in this chapter. The left one shows the translation errors against the number of point mismatches. The solid line with markers 'X' is the average translation error. The dash dotted line, dotted line and the dash line are the errors of t_x, t_y and t_z respectively. The right one shows the rotation errors. The solid line with markers 'X' is the average rotation error of the 3 angles. The dash line, the dotted line and dash dotted line are the errors of the Roll, Pitch and Yaw angle respectively.

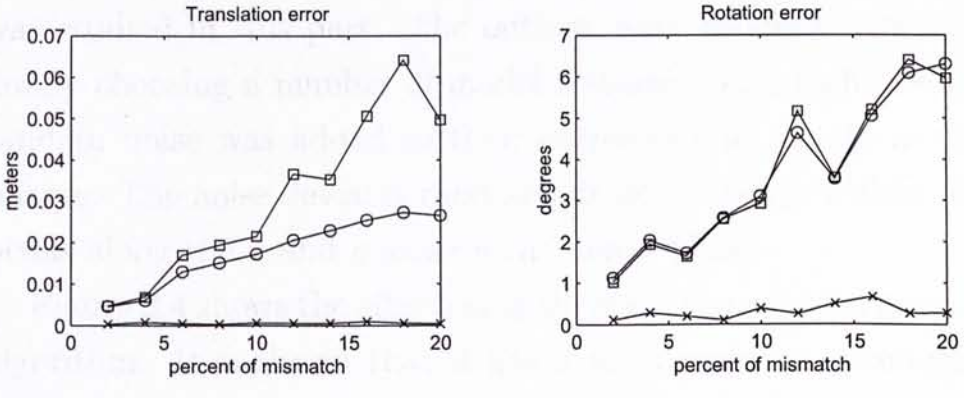

Figure 3.2: Graphs showing the comparison of the accuracy among the three algorithms with the presence of point mismatches. The left plot shows the translation errors while the right one shows the rotation errors. The lines with markers 'X', 'O' and '[]' are the results of proposed approach in this chapter, the GA by Hati and Sengupta and Lowe's method respectively.

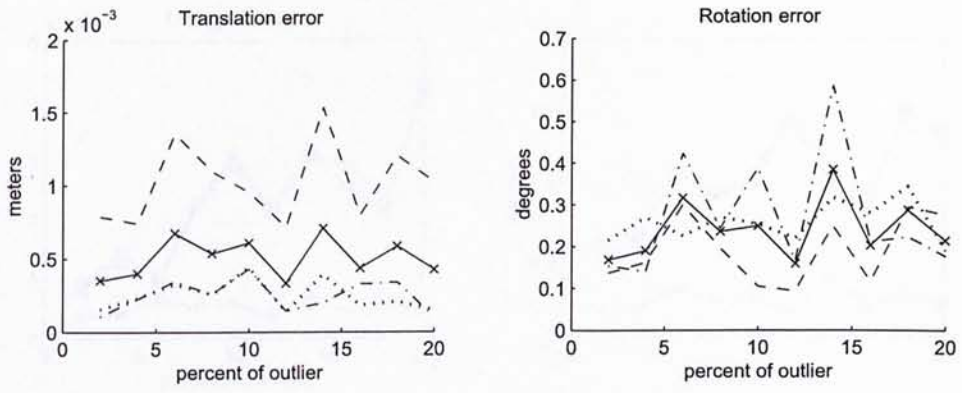

Figure 3.3: Graphs showing the effects of outliers to the proposed genetic algorithm. The left one shows the translation errors against the percentage of outliers present. The solid line with markers 'X' is the average translation error. The dash dotted line, dotted line and the dash lines are the errors of t_x, t_y and t_z respectively. The right one shows the rotation errors. The solid line with markers 'X' is the average rotation error of the 3 angles. The dash line, dotted line and dash dotted line and are the errors of the Roll, Pitch and Yaw angle respectively.

pose error and the percentage of outliers present in the images was studied in this part. The outliers were generated by randomly choosing a number of model features and a high level of random noise was added to their corresponding points in the images. The noise deviates randomly from the range within 100 pixels along the x and y axis on the image plane.

Figure 3.4 shows the effects of outliers to the proposed genetic algorithm. It is shown that it has a lower error in estimating both the rotation and translation parameters than the other two algorithms. Its average angular error falls below 0.5 degrees for different percentage of outliers but the errors of the other two methods are higher than 0.5 degrees even for 4 percents of outliers. The average translation error of the proposed genetic

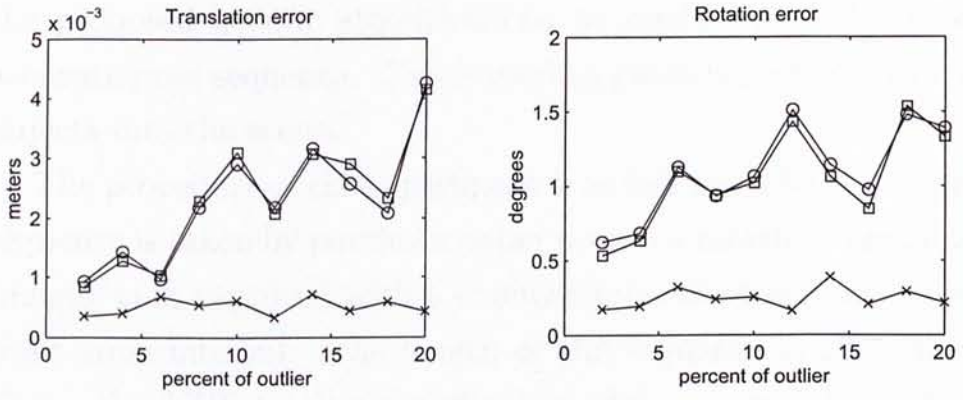

Figure 3.4: Graphs showing the comparison of the accuracy among the three algorithms with the presence of outliers. The left plot shows the translation errors while the right one shows the rotation errors. The lines with markers 'X', 'O' and '[] ' are the results of the approach proposed in this chapter, the GA by Hati and Sengupta and Lowe's method respectively.

algorithm is below 1mm while the other two algorithms have errors more than 1mm. In addition, the estimation errors of the proposed genetic algorithm remain steady with the increase in the percentage of outliers. From this experiment, we can see that the proposed approach in this chapter outperformed the other two algorithms under the presence of outliers.

3.8.2 Real Scene Experiments

In this section, the proposed genetic algorithm was applied to real scene pose estimation and an augmented reality application. We make the following assumptions for this experiment. The model and approximate pose of the scene in a video sequence is recovered by structure from motion methods such as [33] [45]. Because of noise, and problems of local optima, the pose may not be too accurate. Thus, based on the structure found earlier,

the proposed genetic algorithm can be applied to find a more accurate pose sequence. This is used to guide us to place virtual objects into the scene.

The procedure of the experiment is as follows. The test image sequence is taken by putting a paper box on a rotating turntable. Images were captured with a commercial web camera at a constant time interval. The length of the sequence is 20 frames. Then, the KLT tracker described in [39] was used to extract feature points and track them in the image sequence. The 3D structure of the paper box was then acquired by using an interleaved bundle adjustment algorithm described in the literature [33]. After the acquisition of the 3D structure and point features of the paper box, they were used by the genetic algorithm proposed in this chapter to track its pose in the image sequence. Then, the synthetic chair was placed on a plane using the pose sequence computed.

The results of the experiment are shown in figure 3.5 and 3.6. Figure 3.5 shows the pictures that a synthetic chair, which is drawn by wire-frames, was put on the top of the paper box using the proposed approach. You can see that in the video the motion of the virtual chair is consistent with the paper box. Figure 3.6 shows the recovered pose parameters of the paper box in the image sequence. The results are reasonable. You may notice that the lines for translation parameters are flat. This is due to the fact that the paper box was on a rotating turntable.

□ **End of chapter.**

Figure 3.5: Results of augmenting a virtual chair into the real scene using the proposed genetic algorithm. First row: The first and the last image (from left to right) in the original image sequence. Second row: The results of putting the synthetic chair into the image sequence. Demonstrations can be found at http://www.cse.cuhk.edu.hk/~vision/demo/

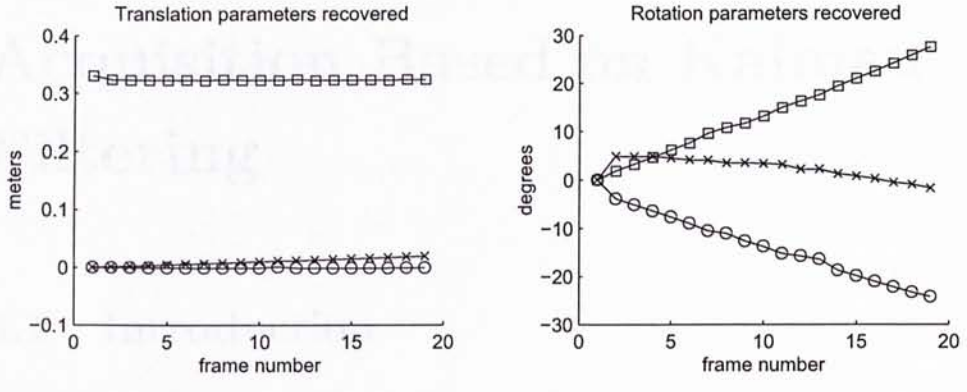

Figure 3.6: Graphs showing the pose parameters of the paper box recovered using the proposed genetic algorithm in the real scene experiment. The plot on the left shows the translation parameters recovered in meters. The lines with markers 'X', 'O' and '[]' correspond to t_x, t_y and t_z respectively. The plot on the right shows the rotation parameters recovered in degrees. The lines with markers 'X', 'O' and '[]' correspond to the Yaw, Pitch and Roll angle respectively.

Chapter 4

Recursive 3D Structure Acquisition Based on Kalman Filtering

4.1 Introduction

The problem of pose estimation discussed in the last chapter can be regarded as a constrained problem of structure and motion. To generalize the method, we have to move one step forward to recover both the structure and pose of an object simultaneously. The estimation of structure and motion involves the optimization of a large number of parameters. The required computation resources would be huge, especially when genetic algorithms are used. This chapter aims to deal with the problem using real-time recursive techniques. Thus, Kalman filtering is employed instead. In addition, the acquired structure is applied to reconstruct 3D model with texture mapping.

A two-step Kalman filter based algorithm, which is inspired by the methods of interleaved bundle adjustment [35], is pro-

posed in this chapter. In the algorithm, the pose and the structure of the object are computed sequentially in an interleaved sense. A set of extended Kalman filters (EKFs) similar to that of in [30] for structure updating is applied to refine the 3D positions of each feature. One EKF is used for computing the pose of the next frame with the previous structure estimates. The main advantage of this two-step approach over other similar recursive approaches such as the work of [7] and [8] is that the state vectors involved are broken down into smaller ones. The increase in the number of point features causes a linear increase in the number of matrix entries needed to be handled. This two-step algorithm saves a lot of computation when the number of features needed to be handled is large, which is quite common for the reconstruction of objects with full details. In addition to the gain in speed, the implementation can handle the structure from motion problem with changeable set of feature points using the KLT tracker [38] [39].

4.2 Overview of the Algorithm

The proposed system can be divided into three parts: feature extraction and tracking, model initialization, structure and pose updating. To make the presentation of the algorithm easy to understand, it is first assumed that all the features are observable from the first to the last frame in the image sequence. The details of the extra treatments needed to handle occlusion and disocclusion of feature points are discussed in section 4.5.

4.2.1 Feature Extraction and Tracking

The KLT tracker described in [38] [39] is used to extract feature points and track them from images to images. With KLT, features are selected as described by Shi and Tomasi [4] and points between two frames are matched on a 2D basis. It is assumed that the problem of tracking has been solved and point matches from KLT are reliable enough for model reconstruction.

4.2.2 Model Initialization

The model initialization is achieved by assuming that the projection of the first image in the sequence is orthographic. Under this assumption, the depth of the object should be less than 1/3 distance between the object and the camera in the first frame. Perspective projection is assumed for image formation process in the remaining frames so that the algorithm can deal with perspective images. The orthographic projection is expressed mathematically as:

$$\begin{pmatrix} u_{i,1} \\ v_{i,1} \end{pmatrix} = \frac{f}{z_{init}} \begin{pmatrix} x_{i,1}^C \\ y_{i,1}^C \end{pmatrix} \tag{4.1}$$

z_{init} is the distance between the object and camera center. It is a parameter given by the user of the system and can be approximated easily. Please refer to section 1.2 on page 3 for the definitions of the notations $u_{i,1}$, $v_{i,1}$, $x_{i,1}^C$ and $y_{i,1}^C$. To obtain the initial model, features in the first image are back-projected from the image plane to the camera coordinate frame according

to equation 4.1. The resulting initial structure is a planar model located at a distance z_{init} from the camera.

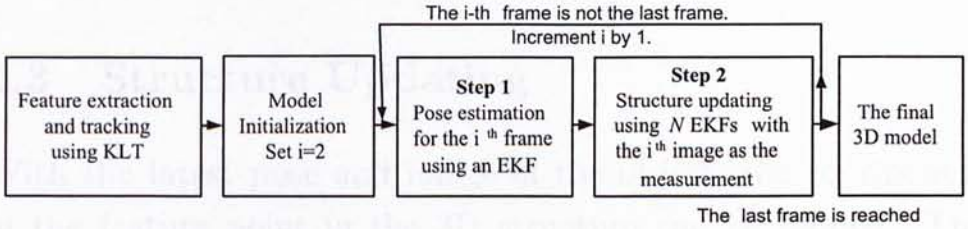

Figure 4.1: The flowchart of the two-step Kalman filter based algorithm.

4.2.3 Structure and Pose Updating

After the planar model is initialized, the model and its pose are updated with the extended Kalman filters. Figure 4.1 shows the flow of the algorithm.

The initial model and the second image are fed into the first step for pose estimation. An EKF similar to that of described in [32] is adopted. The pose of the object with respect to the second image is estimated with a calculation of the prediction and update equations of the EKF. The newly recovered pose and the input image are passed to the second step of the algorithm for structure updating.

The second step consists of a set of N EKFs. Each EKF corresponds to each coordinate point in the reconstructed 3D

model. N EKFs are needed for a model of N feature points. With the observations and the pose recovered for the current image frame, the coordinates of each feature point are updated accordingly. The algorithm alternates between the step 1 and 2 until all images in the sequence are used.

4.3 Structure Updating

With the latest pose and image of the object, the coordinates of the feature point in the 3D structure can be refined. The refinement of model structure in the algorithm is achieved by extended Kalman filter (EKF). Harris and Pike [1] used Iterated Extended Kalman Filter (IEKF) for the reconstruction of Euclidean structure from motion. Beardsley and Zisserman [30] repeated and applied it within a projective framework. Both systems show that Kalman filtering is useful for integrating information of a 3D structure from image measurements over time. The proposed algorithm follows the formulation of Beardsley and Zisserman [30] but with some modifications.

Here EKF is used instead of IEKF as in [30] and [1]. The main reason is that the improvements of using IEKF instead of EKF are marginal. Using EKF can minimize the computation of the algorithm.

The following is the formulation of the EKF for structure updating. For N model points, N EKFs are needed for the structure update. The notation $X_{i,t}$ represents the i^{th} point at the t^{th} time-step in the object coordinate frame. $\hat{X}_{i,t,t-1}$ and $\hat{X}_{i,t,t}$ are the positions of point $X_{i,t}$ after the prediction and

update respectively.

We first define the dynamic model of a 3D point. The state transition equation for a model point can be written as:

$$X_{i,t} = X_{i,t-1} + \gamma_t \tag{4.2}$$

γ_t is the zero mean Gaussian noise with covariance Q_t. The observation equation is:

$$\epsilon_{i,t} = h_t(X_i) + \upsilon_t \tag{4.3}$$

υ_t is the zero mean Gaussian noise with covariance R_t. $\epsilon_{i,t}$ is the real measurement from the image sequence. $h_t(X_i)$ is the projection function of the system.

$$h_t(X_i) = f \left[\begin{array}{cc} \frac{x^C_{i,t}}{z^C_{i,t}} & \frac{y^C_{i,t}}{z^C_{i,t}} \end{array} \right]^T \tag{4.4}$$

$X^C_{i,t}$ is obtained from the equation 1.1 by substituting X^O_i by X_i. The extended Kalman filter first provides an optimal estimate of the state at the next sample time in accordance with the following equations:

$$\hat{X}_{i,t,t-1} = \hat{X}_{i,t-1,t-1} \tag{4.5}$$

$$\wedge_{i,t,t-1} = \wedge_{i,t-1,t-1} + Q_t \tag{4.6}$$

They are known as the prediction equations. $\wedge$ is the 3×3 covariance matrix of X. Here a noise covariance Q_t of γ_t is added. Q_t is a 3×3 diagonal matrix. In normal situations, only

the entry corresponds to the z coordinates of the 3D point is set to non-zero. The reason is that the initial guess of the structure is a planar object. The z coordinate of the structure is rough and a model noise is needed to reflect the actual situation.

Followed by the state prediction, the filter improves the previous estimate using the measurements acquired as:

$$\hat{X}_{i,t,t} = \hat{X}_{i,t,t-1} + W(\epsilon_{i,t} - h_t(\hat{X}_{i,t,t-1})) \tag{4.7}$$

$$\wedge_{i,t,t} = \wedge_{i,t,t-1} - W \nabla h_X \wedge_{i,t,t-1} \tag{4.8}$$

They are known as the update equations. W is known as the 3×2 Kalman gain matrix of the filter.

$$W = \wedge_{i,t,t-1} \nabla h_X^T (\nabla h_X \wedge_{i,t,t-1} \nabla h_X^T + R_t)^{-1} \tag{4.9}$$

R_t is the measurement noise covariance matrix. It is a tuning parameter and is set according to the quality of the images. It can also be acquired during the process of camera calibration. ∇h_X is the Jacobian of the non-linear projection function $h_t(X_i)$ evaluated at $\hat{X}_{i,t,t-1}$. ∇h_X can be written as:

$$\nabla h_X = \left. \frac{dh_t(X_i)}{dX_i} \right|_{X_i = \hat{X}_{i,t,t-1}} \tag{4.10}$$

In this way, the coordinates of the model points can be updated accordingly with the measurements from the image sequence.

4.4 Pose Estimation

With the newly refined model and the image of the next frame, the pose of the object with respect to the next image in the sequence can be estimated. Lippiello et al [32] used EKF for pose estimation of a known CAD model with stereo image sequence. The work of [31] improves the previous one with a feature pre-selection scheme using BSP tree. We follow the method in [32] but adopt it for a single camera view image sequence. A simple feature selection scheme is used to improve the accuracy and convergence of the algorithm.

Similar to the Kalman filter for structure updating, we have first to consider the dynamic model that describes the motion of the object. w is the state of the system and is defined as follows:

$$w = \left[\begin{array}{cccccccccccc} t_x & \dot{t_x} & t_y & \dot{t_y} & t_z & \dot{t_z} & \alpha & \dot{\alpha} & \beta & \dot{\beta} & \gamma & \dot{\gamma} \end{array} \right]^T \tag{4.11}$$

t_x, t_y, and t_z are the translation parameters of the object along the x, y and z axis respectively. $\dot{t_x}$, $\dot{t_y}$ and $\dot{t_z}$ are their corresponding velocities. α, β and γ are respectively the Yaw, Pitch and Roll angle. Their corresponding angular velocities are $\dot{\alpha}$, $\dot{\beta}$ and $\dot{\gamma}$. T_s is used to denote the duration over the sample period. Over the sample period, both the velocities and the angular velocities are assumed constant.

The state transition equation for the model is written as:

$$w_t = Aw_{t-1} + \gamma'_t \tag{4.12}$$

γ'_t is the zero mean Gaussian noise with covariance Q'_t for the system. A is known as the state transition matrix. It is a 12×12 block diagonal defined as follows:

$$A = diag\left\{ \begin{bmatrix} 1 & T_s \\ 0 & 1 \end{bmatrix}, ..., \begin{bmatrix} 1 & T_s \\ 0 & 1 \end{bmatrix} \right\} \tag{4.13}$$

The observation equation is:

$$\epsilon'_t = g_t(w_t) + \upsilon'_t \tag{4.14}$$

υ'_t is the zero mean Gaussian noise with covariance R'_t for the measurement model. ϵ'_t is a $k \times 1$ column vector representing the real measurements from the image sequence for k selected feature points in the object. $g_t(w_t)$ is the projection function defined as:

$$g_t(w_t) = f\left[\begin{array}{cccccccc} \frac{x^C_{1,t}}{z^C_{1,t}} & \frac{y^C_{1,t}}{z^C_{1,t}} & \cdots & \frac{x^C_{i,t}}{z^C_{i,t}} & \frac{y^C_{i,t}}{z^C_{i,t}} & \cdots & \frac{x^C_{k,t}}{z^C_{k,t}} & \frac{y^C_{k,t}}{z^C_{k,t}} \end{array} \right]^T \tag{4.15}$$

Similar to $h_t(X_i)$, $X^C_{i,t}$ is obtained from the equation 1.1 by substituting X^O_i by X_i. The rotation matrix R_t and translation vector T_t are evaluated with the parameters of the column vector w_t.

In the system, not all the feature points extracted by the tracker are passed to the extended Kalman filter for pose prediction and update. It relies on the most accurate portion of model points. In the implementation, a fixed portion of feature points is chosen. The model points are chosen based on

how much they are updated in the step of structure refinement. Those points that are steady and have a high tendency to remain at the same 3D coordinates are regarded as reliable. The reason is obvious because they are accurate enough and less update is needed.

Here are the four core Kalman filter equations for pose estimation. The prediction equations for calculating the optimal estimates are:

$$\hat{w}_{t,t-1} = A\hat{w}_{t-1,t-1} \tag{4.16}$$

$$P_{t,t-1} = AP_{t-1,t-1}A^T + Q'_t \tag{4.17}$$

The update equations for the corrections of estimates are:

$$\hat{w}_{t,t} = \hat{w}_{t,t-1} + K(\epsilon'_t - g_t(\hat{w}_{t,t-1})) \tag{4.18}$$

$$P_{t,t} = P_{t,t-1} - K\nabla g_w P_{t,t-1} \tag{4.19}$$

$\hat{w}_{t,t-1}$ and $\hat{w}_{t,t}$ are the states of w_t after the prediction and update respectively. $P_{t,t-1}$ and $P_{t,t}$ are 12×12 matrices and they are respectively the covariance of $\hat{w}_{t,t-1}$ and $\hat{w}_{t,t}$. K is the $12 \times 2k$ Kalman gain matrix for the filter.

$$K = P_{t,t-1}\nabla g_w^T(\nabla g_w P_{t,t-1}\nabla g_w^T + R'_t)^{-1} \tag{4.20}$$

∇g_w is the Jacobian of the non-linear observation equation $g_t(w_t)$ evaluated at $\hat{w}_{t,t-1}$.

$$\nabla g_w = \left.\frac{dg_t(w_t)}{dw_t}\right|_{w_t=\hat{w}_{t,t-1}} \tag{4.21}$$

The pose of the model to the next frame is estimated with the Kalman filtering equations. The pose and the image of the next frame are passed to the filters as described in the last section for structure refinement until all the frames are utilized.

4.5 Handling of the Changeable Set of Feature Points

Since there is a relative motion between the camera and the scene, features of the scene appear and disappear throughout the image sequence. It is quite seldom that a feature can survive from the first to the last frame. The set of "active" feature points is changing from time to time due to occlusion and disocclusion. The literature in [8] and [11] do not address the problem of changeable active feature set. The paper in [7] proposes a solution to handle occlusion and disocclusion of feature points in their implementation. For disappearing of point features, the authors just remove the related entries of those disappeared points from the main filter. For appearing of new point features, they first apply a subfilter, which is an EKF, to reconstruct the initial condition of the new feature. Then the related entries of the new feature are inserted back to the main filter.

Extra treatments are needed in the steps of feature tracking, model initialization and structure recovery to deal with changeable feature set in the proposed two-step algorithm. In the process of feature tracking, the whole image sequence is divided into a number of sections. For each section, a number of frames near the end of that section are overlapped with the frames at the

beginning of the succeeding section. The KLT tracker in [39], with feature replacement mechanism, is applied to each section independently. The main reason of applying the tracker in such a way is that some features may stick to the "boundary" of the object even the features disappear. In other words, the tracker is unable to release those old features even they go out of the camera view. Cutting the image sequence into sections forces the tracker to release obsolete features after a finite time limit. This step is important in the implementation since this results in a wrong point correspondence between the 2D image and the 3D model.

New model points in the structure are initialized when new point features appear in the image sequence. This is done by assuming the projection of that point on its first appeared image frame is orthographic. The initial position, expressed in camera coordinate frame, is computed according to equation 4.1. The coordinates are then transformed back to the object coordinate frame by equation 1.1. After the initialization of the 3D position, a new extended Kalman filter is set up for updating its position with the measurements in the succeeding frames. In addition, this new point is added to the pool ready to be selected for pose estimation. No special modification is needed for the extended Kalman filter for pose estimation.

When a point feature vanishes from the image sequence, the Kalman filter that corresponds to the point is removed. The 3D position of that feature will no longer be updated. The index of that feature is also marked invalid for pose estimation since no related measurements in future time-steps can be used for find-

ing the pose of the object. The treatments for handling changeable feature set are simple in the proposed algorithm compared to the procedure in [7]. The reason is that a large single filter to handle the whole 3D structure is broken into smaller ones, one filter for each model point.

4.6 Analytical Comparisons with Other Algorithms

4.6.1 Comparisons with the Interleaved Bundle Adjustment Method

Computation Efficiency

The main advantage of the Kalman filter based recursive approach is the gain in speed and scalability. The 3D model is reconstructed by scanning through it once along the time-step sequentially. The interleaved bundle adjustment approach needs roughly tens iterations, i.e. scanning the image sequence 10 times, for the reconstruction. For the proposed 2-step EKF, an extra view of the object can be handled naturally by calculating the prediction and update equations for both the pose and structure only for that new measurement. However, the interleaved bundle adjustment method needs to re-compute from the first frame to the latest frame for a several iterations. The proposed recursive approach is much more effective in handling extra measurement information.

Algorithm Stability

Another advantage of the Kalman filter based method is that it has a better convergence rate in handling long sequences with large object motion, say a total of 90 degrees rotation along one of the axes. Consider the rotation of the object between the first and the last frame. In the first pass and the first iteration of the interleaved bundle adjustment method, the initial model is used to estimate the pose of the object with respect the last frame. This results in a large pose estimation error and eventually leads the computation in the second pass of the interleaved bundle adjustment to diverge. For the proposed recursive approach, the model is updated incrementally from frames to frames, the most up-to-date model is used for pose estimation. The problem of using an inaccurate model to compute the pose is eliminated.

Accuracy of the Solutions

Dynamic systems have been incorporated into the EKF for pose tracking in the proposed 2-step EKF. It means that the temporal relations of the six pose parameters in the pose sequence have been exploited to filter off the measurement noise present in the pose tracking process. On the other hand, the computation of the camera pose of each frame is independent in the traditional interleaved bundle adjustment method. The recovered pose of a single frame may deviate too much from the preceding or succeeding frames, which is probably due to the errors in feature tracking and extraction. The proposed approach is able to avoid such an error by relying on the dynamic system defined and ig-

noring those noisy measurements within the Kalman filtering framework.

4.6.2 Comparisons with the EKF by Azarbayejani and Pentland

Algorithm Complexity

The major difference between the proposed 2-step EKF and the EKF by Azarbayejani and Pentland in [8] is that the structure refinement and pose estimation is broken down into two steps and each correspondence point in the structure is decoupled. In the 2-step EKF, there are one 12×1 state vector for pose estimation plus N 3×1 state vectors for structure updating, where N is the total number of available features. This respectively results in one 12×12 and N 3×3 state covariance matrices. Since the number of measurements involved in the pose estimation step is fixed, both the storage and computation complexity are $O(N)$. For the EKF described in [8], the pose and structure are encoded in a single state vector. The size of the state covariance matrices is $(N + 7) \times (N + 7)$. The storage and computation complexity are $O(N^2)$ and $O(N^3)$ respectively. The proposed modification is actually a tradeoff between speed and accuracy. However, experimental results show that the loss in accuracy is little and acceptable in real applications. Moreover, the reduction in complexity is useful for real-time robotics application since the computation resources in micro-controllers are tight. Also, the 2-step EKF is ready to be implemented on distributed micro-processing system. The set of N EKFs used for structure

updating can be run in parallel in a set of N micro-processors in a robotic system to speed up the computation.

The Problem of Scaling

Another advantage of the proposed algorithm is that the exact model can be reconstructed given the alignment of translation T^C between the object coordinate frame and camera coordinate frame. The EKF by Azarbayejani and Pentland is subject to severe scaling problem even T^C is given in the reconstruction process. This is due to the nature of the structure model adopted in the filter. A small deviation in the estimation of the z coordinates of the point features causes a large change in the scale of the object. The problem can be fixed by giving the EKF the real 3D coordinates of one of the feature points and setting their corresponding entries in the state covariance matrix zero.

4.7 Experiments and Results

4.7.1 Synthetic Data Experiments

This experiment is to test the performance of the proposed 2-step EKF. The specification of the synthetic object and camera settings is the same as that in section 3.8.1 on page 34.

To make the synthetic sequences contain motion discontinuities, the motion of the object was divided into three different segments, a pure translation section, a pure rotation section and a general motion section. The sequence of occurrence of the sections was by random. The motion parameters were generated

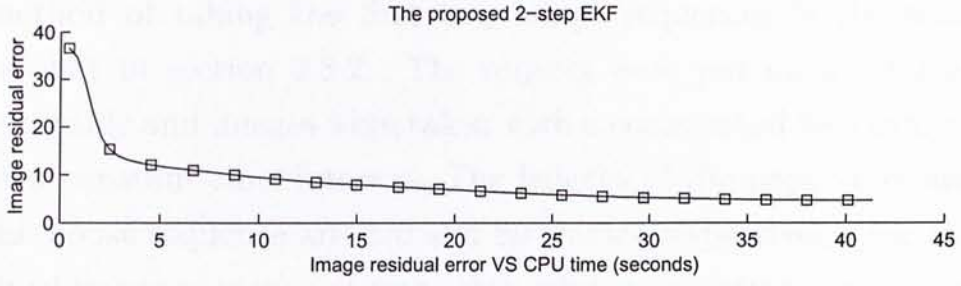

Figure 4.2: The image residual error versus CPU time for the proposed 2-step EKF.

randomly from 0.05 to 0.15 degrees per frame for Yaw, Pitch, Row angle and 0.0005 to 0.0015 meters per frame for t_x, t_y and t_z. The length of each synthetic sequence is 99 frames. A total of 20 independent tests were carried out.

Figure 4.2 shows the time for the proposed algorithm to optimize the image residual error of the back-projected model. The algorithm minimizes the residual error in a short period of time and finishes the processing of the 99-frame sequence in 41 seconds. One average, it takes 0.42 seconds to process an extra image.

4.7.2 Real Scene Experiments

In this experiment, the proposed algorithm was applied to acquire the 3D structures of three real scenes from three test image sequences. With the structures, the 3D models with texture mapping were built. The resultant objects were output in the form of VRML files.

The three test sequences are the paper box sequence, the house model sequence and the laboratory scene sequence. The

method of taking the first two image sequences is the same as that in section 3.8.2. The objects were put on a rotating turntable and images were taken with a commercial web camera at a constant time interval. The lengths of the paper box and the house sequence are 200 and 80 frames respectively. For the third image sequence, it was taken while translating the camera sideway on a rig. The length of the laboratory sequence is 100 frames.

Figure 4.3, 4.4 and 4.5 are the results of the experiment. The 360^o view of the paper box model was successfully reconstructed from its 2D images. The total number of point features present in the model is about 500. The quality is good in general. However, you may notice that there is a presence of outlying model features in the recovered model, resulting in flaws in some part of the model. The outlying model features are mainly due to point mismatches that arise from the process of feature tracking by the KLT tracker [39]. For the recovered house and laboratory scene model, the total number of model features in each scene is more than a double to that of the paper box model, i.e. at least 1000 for each model. More geometric details are revealed in the reconstructed model in these cases. This demonstrates that the proposed 2-step EKF is able to handle complex scene reconstruction within a reasonable time limit. More results can be found at http://www.cse.cuhk.edu.hk/~vision/demo/

□ **End of chapter.**

Figure 4.3: The reconstruction results of the paper box on the turntable. First row: The first and the 100^{th} image of the paper box. Second and the third row: The reconstructed 3D paper box model viewed in a VRML browser. Here are the two views with texture mapping (the left one) and their wire-frames (the right one). Note that the total length of the image sequence is 200. Demonstrations can be found at http://www.cse.cuhk.edu.hk/~vision/demo/

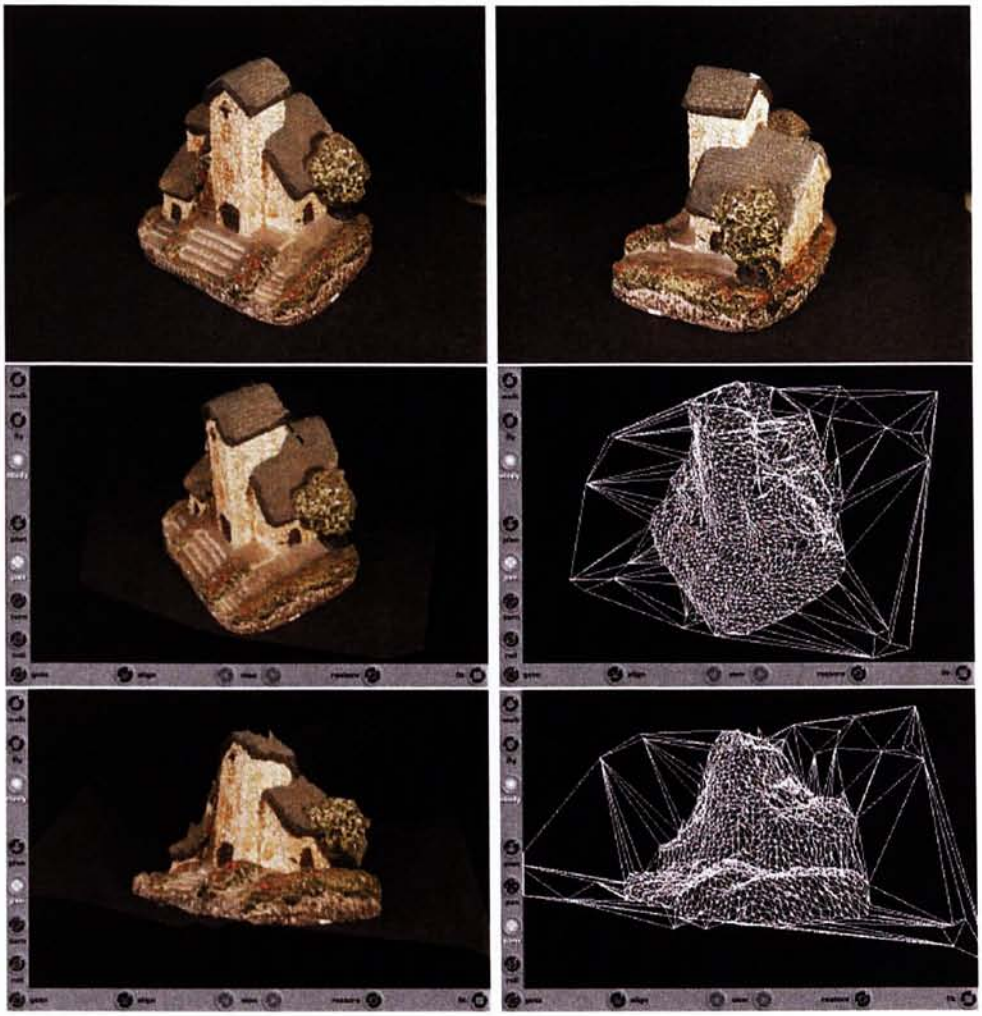

Figure 4.4: The reconstruction results of the house model on the turntable. First row: The first and the last (80^{th}) image of the house model sequence. Second and the third row: The reconstructed 3D house model viewed in a VRML browser. Here are the two views with texture mapping (the left one) and their wire-frames (the right one).

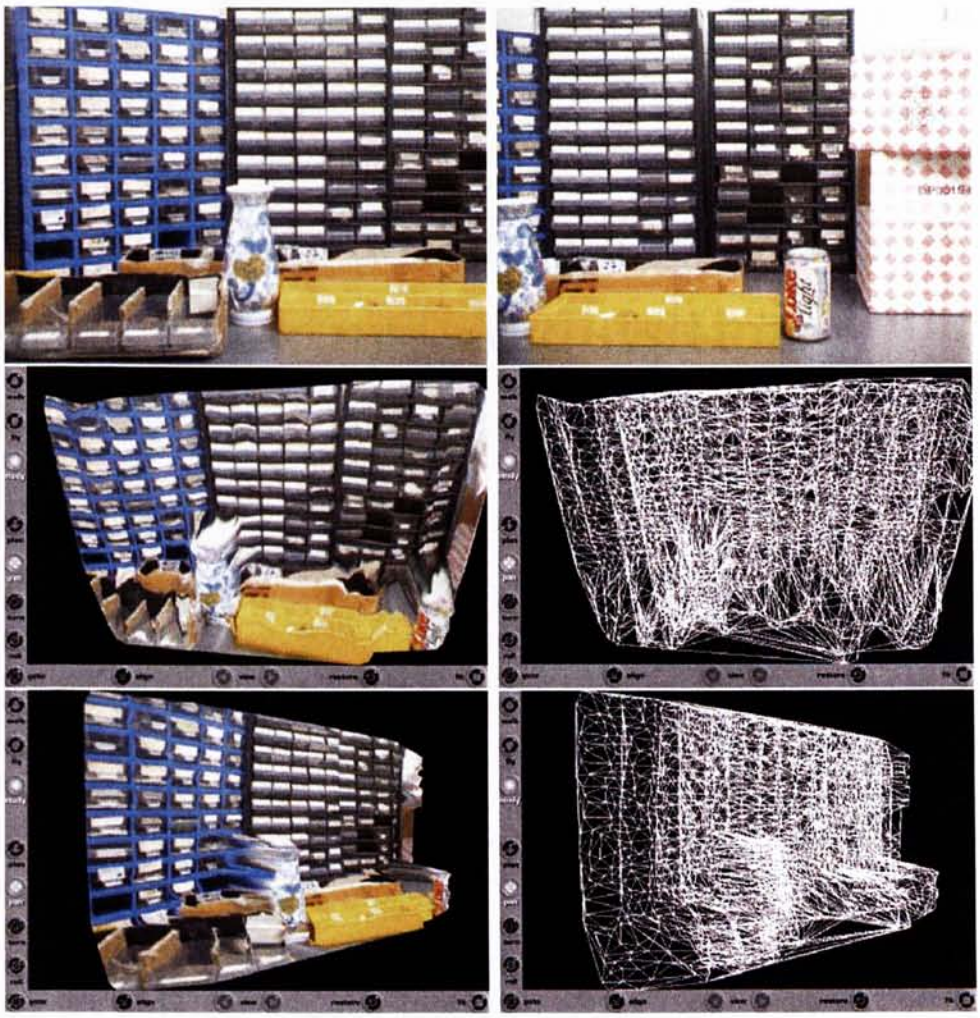

Figure 4.5: The reconstruction results of the laboratory scene. First row: The first and the last (100^{th}) image of the laboratory sequence. Second and the third row: The reconstructed 3D laboratory scene viewed in a VRML browser. Here are the two views with texture mapping (the left one) and their wire-frames (the right one).

Chapter 5

Simultaneous Pose Tracking and Structure Acquisition Using the Interacting Multiple Model

5.1 Introduction

Although the reconstructed 3D models shown in the last chapter are of good qualities, the pose and structure error still exist under certain camera motions. The reason is that one dynamic system is assumed in the previously proposed method, thus it favours only one camera motion, i.e. constant velocity, in the pose tracking process. Real image sequences sometimes contain motion discontinuity. Thus, the previous assumption may not be true, resulting in inaccuracy of the 2-step extended Kalman filter (2-step EKF). In this chapter, the objective is to improve the accuracy of the tracked pose sequence in the process of recovering the structure and motion. At the same time, the required

computation time is kept to a minimum.

The Interacting Multiple Model based (IMM-based) structure and motion (SAM) algorithm is proposed in this chapter. In short, the Interacting Multiple Model (IMM) algorithm [46] is a suboptimal hybrid state filter. It has been widely used as a tool for tracking maneuvering targets in RADAR [50] and vision-based systems [47] [48]. There is no existing work that uses IMM to deal with the SAM problem directly.

The idea of applying IMM algorithm to the SAM problem is inspired by the fact that inaccuracy due to the ambiguities can be minimized if prior information about the structure or motion is utilized, which is well-known and has been reported by Szeliski and Kang in [26]. In the pose estimation step of the algorithm, three EKFs, each describes a unique motion dynamics, embedded within the IMM framework are adopted. They represent those frequently occurring camera motions in real situations. Intuitively, the IMM provides a mechanism to "select" suitable filters automatically in order to set constraints on the camera's motion once prior information is available. With these constraints, the total number of parameters to be estimated is reduced and the accuracy of the recovered pose and structure can be improved. In addition, the problem of motion discontinuity in real images can be handled properly with the IMM framework.

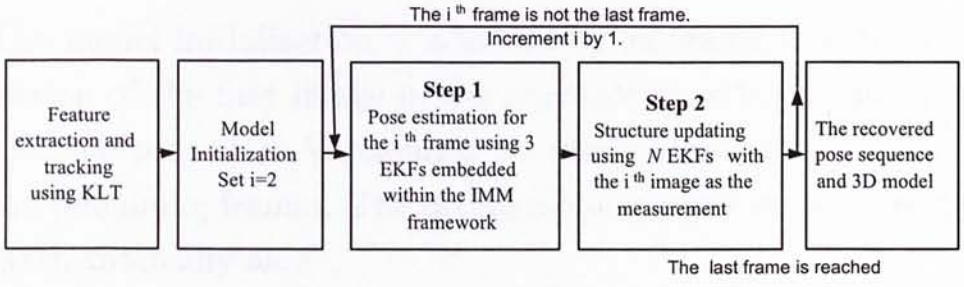

Figure 5.1: The flowchart of the proposed IMM-based approach

5.2 Overview of the Algorithm

The algorithm proposed is this chapter is an extension of the 2-step approach in chapter 4. The framework of these two approaches are similar. For completeness of the algorithm description, some of the implementation details are repeated in this chapter.

To make the presentation of the algorithm clear, it is first assumed that all the features are observable from the first to the last frame in the image sequence. The details of the extra treatments needed to handle the changeable set of feature points are discussed in section 5.5.

5.2.1 Feature Extraction and Tracking

The KLT tracker described in [39] is used to extract feature points and track them from images to images. In our work, it is assumed that the problem of tracking has been solved.

5.2.2 Model Initialization

The model initialization is achieved by assuming that the projection of the first image in the sequence is orthographic. Perspective projection is assumed for image formation process in the remaining frames. The orthographic projection is expressed mathematically as:

$$\begin{pmatrix} u_{i,1} \\ v_{i,1} \end{pmatrix} = \frac{f}{z_{init}} \begin{pmatrix} x^C_{i,1} \\ y^C_{i,1} \end{pmatrix} \tag{5.1}$$

z_{init} is the distance between the object and camera center. Please refer to section 1.2 on page 3 for the definitions of the notations $u_{i,1}$, $v_{i,1}$, $x^C_{i,1}$ and $y^C_{i,1}$. To obtain the initial model, features in the first image are back-projected from the image plane to the camera coordinate frame according to equation 5.1.

5.2.3 Structure and Pose Updating

The initial model and the second image are fed into the first step of the main loop for pose estimation. Three extended Kalman filters (EKFs), each represents a unique motion dynamics, are adopted. These three filters interact with one another using the Interacting Multiple Model (IMM) [46]. The recovered pose is a mixture of output of the three filters. It is passed to the second step for structure updating.

The second step consists of a set of N EKFs. Each EKF corresponds to one coordinate point in the recovered 3D structure. With the observations and the pose recovered for the current

image frame, the coordinates of each feature point are updated accordingly. The algorithm alternates between the step 1 and 2 until all images in the sequence are used.

5.3 Pose Estimation

The pose estimation step consists of three extended Kalman filters (EKFs) embedded within the Interacting Multiple Model (IMM). Each of the three EKFs describes a frequently occurring motion dynamics in real situations. The IMM algorithm provides a probability framework for filter switching. We first see how the IMM algorithm works and then discuss the formulations of the three EKFs.

5.3.1 The Interacting Multiple Model Algorithm

The IMM algorithm is a sub-optimal filter originally proposed by Blom [49]. It is regarded as one of most cost-effective hybrid state estimation schemes. It achieves an excellent compromise between performance and complexity. There are several variations of the IMM algorithm. The Sojourn Time Dependent Markov based (STDM-based) IMM estimator, the Interacting Multiple Bias Model (IMBM) algorithm and the Selected Filter Interacting Multiple Model (SFIMM) are some of the examples. They have been discussed extensively in [46]. The algorithm adopted in the system is the baseline IMM in [46].

The basic IMM algorithm consists of several steps, which can be visualized in figure 5.2. Firstly, the likelihood of each filter

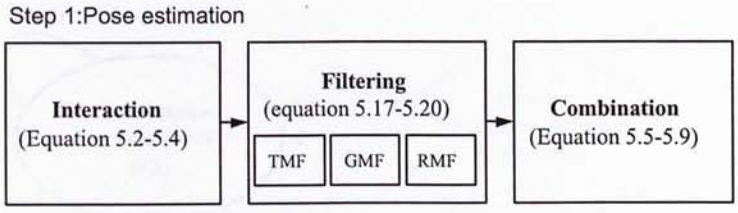

Figure 5.2: The flowchart of the baseline IMM algorithm. The terms TMF, GMF and RMF are respectively the short forms of the pure translation motion filter, the general motion filter and the pure rotation motion filter. The exact definitions can be found in section 5.3.2.

$u_t(m)$ is updated according to the 3×3 switching matrix J:

$$u_t^*(m) = \sum_n J(mn) u_t(n) \tag{5.2}$$

m and n are integers. In the implementation, the initial likelihood of the filters is equal. The switching matrix is set by assuming that the model under reconstruction continues with a single motion for an extended period of time with occasional transition to another motion model. The diagonal entries of $J(m'm') \approx 1$ with off-diagonal entries $J(m'n') = (1 - J(m'm'))/2$. Figure 5.3 shows the switching process of the three EKFs in the system.

Secondly, the state estimates and their corresponding covariances of the previous time-step are mixed:

$$\hat{w}^*_{t-1,t-1}(m) = \frac{1}{u_t^*(m)} \sum_n J(mn) u_t(n) \hat{w}_{t-1,t-1}(n) \tag{5.3}$$

$$\begin{aligned} P^*_{t-1,t-1}(m) = & \frac{1}{u_t^*(m)} \sum_n J(mn) u_t(n) \\ & (P_{t-1,t-1}(n) + [\hat{w}_{t-1,t-1}(n) - \hat{w}_{t-1,t-1}(m)] \\ & [\hat{w}_{t-1,t-1}(n) - \hat{w}_{t-1,t-1}(m)]^T) \end{aligned} \tag{5.4}$$

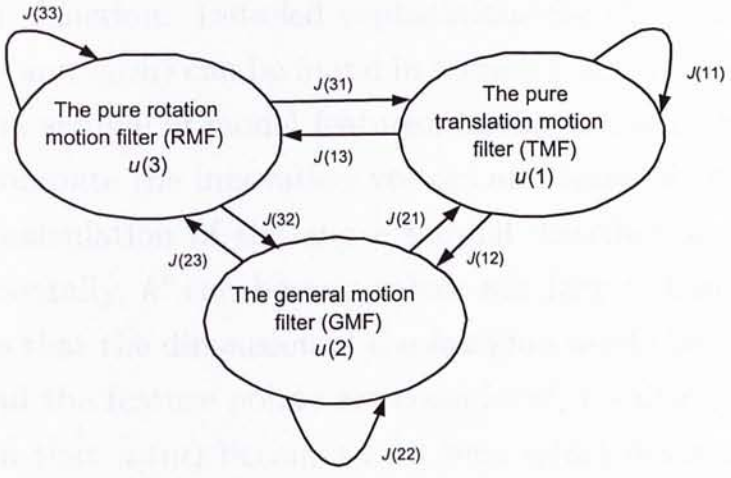

Figure 5.3: The IMM model switching process.

$\hat{w}^*_{t-1,t-1}(m)$ and $P^*_{t-1,t-1}(m)$ are then passed to the EKFs, which are described in the next section, for prediction and smoothing with the measurements in the current time-step . The outputs of the prediction phase are $\hat{w}_{t,t-1}(m)$ and $P_{t,t-1}(m)$ while the outputs of the smoothing phase are $\hat{w}_{t,t}(m)$ and $P_{t,t}(m)$.

After the Kalman filtering cycle, the likelihood of each filter is updated with regard to the innovation vector $v_t(m)$ and its corresponding residual covariance $S_t(m)$ of the filer:

$$u_t(m) = \kappa \frac{u^*_t(m) exp\left[-\frac{1}{2} v^T_t(m) S^{-1}_t(m) v_t(m)\right]}{||S_t(m)||^{1/2}} \quad (5.5)$$

$$v_t(m) = \epsilon'_t - g_t\left(\hat{w}_{t,t-1}(m)\right) \quad (5.6)$$

$$S_t(m) = \nabla g_w P_{t,t-1}(m) \nabla g^T_w + R'_t \quad (5.7)$$

κ is a normalization factor such that $\sum_m u_t(m) = 1$. $u_t(m)$ is computed according to an k'-dimension zero-mean normal dis-

tribution function. Detailed explanations for the computation of $S_t(m)$ and $v_t(m)$ can be found in section 5.3.2. In the system, a random sample of model features, having a size of k', is chosen to compute the innovation vectors and residual covariances for the calculation of the above normal distribution function. Experimentally, k' can be any values not larger than 10. The reason is that the dimension of the function would be extremely high if all the feature points are considered, resulting in a phenomenon that $u_t(m)$ becomes zero even $v_t(m)$ deviates a little from zero. This causes the IMM algorithm to degenerate once the likelihood of all filters becomes zero. Setting such a limit can prevent the degeneracy and in addition reduces the computation complexity.

Lastly, the usable output state vector $\hat{w}_{t,t}$ and covariance matrix $P_{t,t}$ at time-step t are generated with the following equations:

$$\hat{w}_{t,t} = \sum_m u_t(m)\hat{w}_{t,t}(m) \tag{5.8}$$

$$P_{t,t} = \sum_m u_t(m)\left(P_{t,t}(m) + [\hat{w}_{t,t}(m) - \hat{w}_{t,t}] [\hat{w}_{t,t}(m) - \hat{w}_{t,t}]^T\right) \tag{5.9}$$

The final output of the system is a linear sum of the smoothed state and covariance estimates of each filter weighted by the corresponding updated filter likelihood.

5.3.2 Design of the Individual EKFs

The three EKFs describing three different motion dynamics are defined as follows:

The General Motion Filter (GMF): GMF is designed to handle arbitrary object motion with unrestricted rotation and translation. Constant velocity is assumed for the GMF.

The Pure Translation Motion Filter (TMF): TMF is designed for tracking the objects with zero rotation motion.

The Pure Rotation Motion Filter (RMF): RMF is dedicated to tackle the objects with pure rotation around the y-axis (i.e. non-zero pitch angle).

The state vector w is common for all the filters:

$$w = \begin{bmatrix} t_x & \dot{t_x} & t_y & \dot{t_y} & t_z & \dot{t_z} & \alpha & \dot{\alpha} & \beta & \dot{\beta} & \gamma & \dot{\gamma} \end{bmatrix}^T \tag{5.10}$$

t_x, t_y, and t_z are the translation parameters of the object along the x, y and z axis respectively. $\dot{t_x}$, $\dot{t_y}$ and $\dot{t_z}$ are their corresponding velocities. α, β and γ are respectively the Yaw, Pitch and Roll angle with $\dot{\alpha}$, $\dot{\beta}$ and $\dot{\gamma}$ as their corresponding angular velocities. The state transition and measurement equation for the filters are:

$$w_t = Aw_{t-1} + \gamma'_t \tag{5.11}$$

$$\epsilon'_t = g_t(w_t) + v'_t \tag{5.12}$$

$$g_t(w_t) = f\left[\begin{array}{cccccccc} \frac{x^C_{1,t}}{z^C_{1,t}} & \frac{y^C_{1,t}}{z^C_{1,t}} & \ldots & \frac{x^C_{i,t}}{z^C_{i,t}} & \frac{y^C_{i,t}}{z^C_{i,t}} & \ldots & \frac{x^C_{k,t}}{z^C_{k,t}} & \frac{y^C_{k,t}}{z^C_{k,t}} \end{array}\right]^T \tag{5.13}$$

γ'_t and υ'_t are zero mean Gaussian noise. ϵ'_t is an $k \times 1$ column vector representing the selected real measurements from the images. The model points are chosen based on how much they are updated in the step of structure refinement. Those points that are steady and have a high tendency to remain at the same 3D coordinates are used. $g_t(w_t)$ is the $k \times 1$-output projection function. The rotation matrix R_t and translation vector T_t are evaluated with the parameters of the column vector w_t. A is a 12×12 block diagonal state transition matrix. A is different for the 3 EKFs and is defined as follows:

For the GMF:

$$A = A_{GMF} = diag\left\{\left[\begin{array}{cc} 1 & T_s \\ 0 & 1 \end{array}\right]\left[\begin{array}{cc} 1 & T_s \\ 0 & 1 \end{array}\right]\left[\begin{array}{cc} 1 & T_s \\ 0 & 1 \end{array}\right]\right.$$
$$\left.\left[\begin{array}{cc} 1 & T_s \\ 0 & 1 \end{array}\right]\left[\begin{array}{cc} 1 & T_s \\ 0 & 1 \end{array}\right]\left[\begin{array}{cc} 1 & T_s \\ 0 & 1 \end{array}\right]\right\} \tag{5.14}$$

For the TMF:

$$A = A_{TMF} = diag\left\{\left[\begin{array}{cc} 1 & T_s \\ 0 & 1 \end{array}\right]\left[\begin{array}{cc} 1 & T_s \\ 0 & 1 \end{array}\right]\left[\begin{array}{cc} 1 & T_s \\ 0 & 1 \end{array}\right]\right.$$
$$\left.\left[\begin{array}{cc} 1 & 0 \\ 0 & 0 \end{array}\right]\left[\begin{array}{cc} 1 & 0 \\ 0 & 0 \end{array}\right]\left[\begin{array}{cc} 1 & 0 \\ 0 & 0 \end{array}\right]\right\} \tag{5.15}$$

For the RMF:

$$A = A_{RMF} = diag\left\{\begin{bmatrix}1 & 0\\ 0 & 0\end{bmatrix}\begin{bmatrix}1 & 0\\ 0 & 0\end{bmatrix}\begin{bmatrix}1 & 0\\ 0 & 0\end{bmatrix}\right.$$
$$\left.\begin{bmatrix}1 & 0\\ 0 & 0\end{bmatrix}\begin{bmatrix}1 & T_s\\ 0 & 1\end{bmatrix}\begin{bmatrix}1 & 0\\ 0 & 0\end{bmatrix}\right\} \quad (5.16)$$

From the above dynamic systems, the four core Kalman filtering equations for pose estimation can be derived:

$$\hat{w}_{t,t-1} = A\hat{w}_{t-1,t-1} \quad (5.17)$$
$$P_{t,t-1} = AP_{t-1,t-1}A^T + Q'_t \quad (5.18)$$

$$\hat{w}_{t,t} = \hat{w}_{t,t-1} + K(\epsilon'_t - g_t(\hat{w}_{t,t-1})) \quad (5.19)$$
$$P_{t,t} = P_{t,t-1} - K\nabla g_w P_{t,t-1} \quad (5.20)$$
$$K = P_{t,t-1}\nabla g_w^T(\nabla g_w P_{t,t-1}\nabla g_w^T + R'_t)^{-1} \quad (5.21)$$

$\hat{w}_{t,t-1}$ and $\hat{w}_{t,t}$ are the states of w_t after the prediction and update respectively. $P_{t,t-1}$ and $P_{t,t}$ are 12×12 matrices. They are respectively the covariances of $\hat{w}_{t,t-1}$ and $\hat{w}_{t,t}$. K is the $12 \times 2k$ Kalman gain matrix for the filter. ∇g_w is the Jacobian of the non-linear observation equation $g_t(w_t)$ evaluated at $\hat{w}_{t,t-1}$. With the IMM algorithm and three EKFs, the pose of the model can be estimated.

5.4 Structure Updating

Similar to the 2-step EKF in chapter 4, the structure updating step consists of N identical extended Kalman filters (EKFs), each corresponds to one model point in the 3D space. The dynamic model of a 3D point and its measurement equation are:

$$X'_{i,t} = X'_{i,t-1} + \gamma_t \tag{5.22}$$

$$\epsilon_{i,t} = h_t(X'_i) + \upsilon_t \tag{5.23}$$

$$h_t(X'_i) = f \begin{bmatrix} \frac{x^C_{i,t}}{z^C_{i,t}} & \frac{y^C_{i,t}}{z^C_{i,t}} \end{bmatrix}^T \tag{5.24}$$

γ_t and υ_t are the zero mean Gaussian noise. $\epsilon_{i,t}$ is the real measurement from the image sequence. $h_t(X'_i)$ is the projection function. $X^C_{i,t}$ is obtained by substituting values into equation 5.25 and 1.1. X'_i is a scalar that represents a model point:

$$X^O_i + T^C = \begin{bmatrix} x^S_i \\ y^S_i \\ z^S_i \end{bmatrix} = \begin{bmatrix} u_{i,1} \\ v_{i,1} \\ 0 \end{bmatrix} + \frac{X'_i}{f} \begin{bmatrix} u_{i,1} \\ v_{i,1} \\ f \end{bmatrix} \tag{5.25}$$

Each model point is represented by a single parameter. Such a representation is made under the assumption that the measurements acquired by the camera are non-biased. This is valid for most of the modern high-resolution image capturing devices. Intuitively, the 3D coordinates of the points are expressed in terms of the first images that the features appear. This measure

reduces the computation time required for the EKFs and at the same time maintains the rigidity of the object. Also, the computation is over-determined at every frame when the number of features is larger than 7, resulting in a better convergence and stability of the filter. Detailed discussion about the advantages arising from this structure representation and the method to handle biased measurement can be found in [8].

With the dynamic model, the required EKFs can be derived. The prediction equations are as follows:

$$\hat{X'}_{i,t,t-1} = \hat{X'}_{i,t-1,t-1} \tag{5.26}$$

$$\wedge_{i,t,t-1} = \wedge_{i,t-1,t-1} + Q_t \tag{5.27}$$

$\wedge_{i,t,t-1}$ is the variance of $X'_{i,t,t-1}$. Q_t is the variance of the noise term γ_t, which is a scalar. The reason for incorporating a noise term having a considerable value into a static structure is that our initial structure is a planar model. Adjustment of the depth of the recovered object is necessary. The the update equations of the EKF are:

$$\hat{X'}_{i,t,t} = \hat{X'}_{i,t,t-1} + W(\epsilon_{i,t} - h_t(\hat{X'}_{i,t,t-1})) \tag{5.28}$$

$$\wedge_{i,t,t} = \wedge_{i,t,t-1} - W \nabla h_X \wedge_{i,t,t-1} \tag{5.29}$$

$$W = \wedge_{i,t,t-1} \nabla h_X^T (\nabla h_X \wedge_{i,t,t-1} \nabla h_X^T + R_t)^{-1} \tag{5.30}$$

W is known as the 1×2 Kalman gain matrix of the filter. R_t is a 2×2 measurement noise covariance matrix. ∇h_X is the Jacobian of the non-linear projection function $h_t(X'_i)$ evaluated

at $\hat{X}'_{i,t,t-1}$. After the Kalman filtering cycle, the updated estimate $\hat{X}'_{i,t,t}$ is computed. The actual 3D structure is obtained by transforming X'_i back to the object coordinate frame using equation 5.25.

5.5 Handling of the Changeable Set of Feature Points

The set of "active" feature points is changing due to occlusion and disocclusion. Extra treatments are needed in each step of the pose tracking and structure acquisition process. In feature tracking, the whole image sequence is divided into a number of sections. For each section, a number of frames near the end of that section are overlapped with the frames at the beginning of the succeeding section. The KLT tracker in [39], with feature replacement mechanism, is applied to each section independently. This procedure is exactly the same as that described in section 4.5.

New model points in the structure are initialized when new point features appear in the image sequence. This is obtained by assuming the projection of that point on its first appeared image, say at frame t_a, is orthographic. The initial position, expressed in camera coordinate frame, is computed according to equation 5.1. This is equivalent to setting parameter X'_i equal to t_{z,t_a} in our structure representation. t_{z,t_a} is a term in the translation vector T_t such that $T_{t_a} = \begin{bmatrix} t_{x,t_a} & t_{y,t_a} & t_{z,t_a} \end{bmatrix}^T$. The structure parameter for that point is now expressed in terms of

its image coordinates at time-step t_a. The relationship between the structure parameter X_i' and its coordinates in the object frame is:

$$X_i^O = R_{t_a}^{-1} \left\{ \left(\begin{bmatrix} u_{i,t_a} \\ v_{i,t_a} \\ 0 \end{bmatrix} + \frac{X_i'}{f} \begin{bmatrix} u_{i,t_a} \\ v_{i,t_a} \\ f \end{bmatrix} \right) - T_{t_a} \right\} \tag{5.31}$$

A new EKF, as described in section 5.4, is set up to refine its position in its camera coordinate frame with parameter X_i'. Its final position can be obtained by calculating its coordinates in the object frame using equation 5.31. In addition, this new point is added to the pool ready to be selected for use in the pose estimation step. No other modifications are needed for the IMM algorithm and the EKFs for pose estimation.

When a point feature vanishes from the image sequence, the filter that corresponds to the point is removed. The index of that feature is also marked invalid for pose estimation since no related measurements in future time-steps can be used for finding the pose of the object.

5.6 Analytical Comparisons with Other EKF-Based Algorithms

5.6.1 Computation Speed

Both the IMM-based approach in this chapter and the 2-step EKF in chapter 4 have the same characteristic that the struc-

ture refinement and pose estimation is broken down into two steps and each correspondence point in the structure is decoupled in the process of filtering. This results in a gain in speed in these approaches. Originally, the decoupling is a tradeoff between speed and accuracy, which is the deficiency of the 2-step EKF in the previous chapter, since the rigidity assumption of the object can no longer be preserved. In the proposed IMM-based approach, the coordinates of each 3D model point are expressed in terms of the first image that they appear. The rigidity of the object is partially or even fully maintained for the case that the set of active features can or cannot be changed respectively. The only underlying assumption is that the measurements acquired by the camera are non-biased. It is valid for most of the existing modern digital cameras. In this way, the adverse effects on decoupling the points in the structure refinement step can be minimized.

To compare the speed with the 2-step EKF, the proposed IMM-based approach should have a higher computation cost because of the IMM calculation overhead and the two additional EKFs for pose estimation. However, the three EKFs for pose tracking can be run concurrently in parallel systems in theory. Also, the special structure representation in the EKFs for structure refinement reduces the computation to at least one-third of the original one. The only possible addition of computation time that cannot be avoided results from the mixing of states and covariances in the IMM algorithm. This small rise in computation time can result in a great increase in accuracy, which has been demonstrated in the comparisons in chapter 6.

5.6.2 Accuracy of the Recovered Pose Sequences

The improvement in the recovered pose accuracy is mainly due to the incorporation of the IMM algorithm. Intuitively, the three EKFs, each with a unique motion dynamics, try out three different constraints on the camera motion. The IMM algorithm makes use of the residual information from the filters to weight for the contribution of each filter to the final computed pose with a probability framework. Constraints on the object's motion are set dynamically in an indirect manner. That is exactly the suggestion by Szeliski and Kang [26] for solving the problem of structure and motion ambiguities. Prior information should be given for the pose or the structure to achieve high accuracy in structure acquisition or pose estimation. The two EKF-based approaches in [8] and in chapter 4 do not have such a mechanism and thus should have a lower accuracy because of the ambiguities. The proposed IMM-based approach can solve the problem with two commonly occurring motions. It can be extended to unlimited number of known motions in a theoretical sense. When applying such a robust structure and motion algorithm, it is inevitable that there is a computation overhead. This is minimized using the techniques mentioned in the previous section. The actual performance in speed and accuracy of the proposed approach compared to other similar EKF-based algorithms can be found in the following chapter.

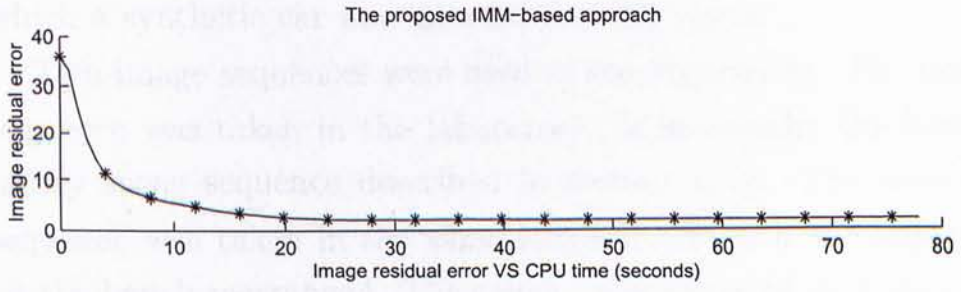

Figure 5.4: The image residual error versus CPU time for the proposed IMM-based approach.

5.7 Experiments and Results

5.7.1 Synthetic Data Experiments

The performance of the proposed IMM-based approach was tested using synthetic data. The data set that was used to evaluated the 2-step EKF in the last chapter was re-used. For details, please refer to section 4.7.1 on page 57.

To see its performance, the average of the resulting image residual error versus CPU time of the 20 independent tests was plotted in figure 5.4. The proposed IMM-based algorithm falls to a low error before the 25^{th} frame. It finishes the processing of the 99-frame sequence in 78 seconds and takes 0.79 seconds to process an extra image on average.

5.7.2 Real Scene Experiments

The proposed IMM-based approach was applied to track the camera motion of two real image sequences while reconstructing the scenes' structures in this experiment. The pose sequences recovered were used to produce augmented reality videos, in

which a synthetic car was inserted into the scenes.

Two image sequences were used in the experiment. The first sequence was taken in the laboratory. It is actually the laboratory scene sequence described in section 4.7.2. The second sequence was taken in the same location but with the objects on the bench rearranged. The camera was mounted on a tripod on the top of a trolley. Images were taken while the trolley was moving. The height of the tripod was allowed to be adjusted. The length of this sequence is 90 frames.

Results of the first image sequence are shown in figure 5.5. An augmented reality video has been made successfully. The orientation of the synthetic car is consistent with the real scene in the whole video sequence. Figure 5.6 also shows the switching of the three EKFs and likelihood of each individual filter in the IMM-based approach. Figure 5.6a was plotted by taking the filter that has a maximum likelihood in figure 5.6b at each time-step. From figure 5.6a, you can see that the embedded IMM algorithm detected the camera motion correctly. It used the pure translation motion filter (TMF) most of the time for this test case, which reflected the actual motion of the camera. You may notice that the system switched to the other two filters, i.e. the general motion filter (GMF) and the pure rotation motion filter (RMF), from the 69^{th} to the 75^{th} frame. It is due to the fact the camera was vibrated. After that, the system switched back to TMF.

Results of the second image sequence can be found in figure 5.7. The camera motion of this test sequence is much more complicated. It contains both the pure translation motion and

the pure rotation motion as defined in section 5.3.2. The car in wire-frame has been successfully put at a place between the paper box and the vase in the video. The motion of the synthetic car is consistent with the background even the camera is under the rotational motion. Figure 5.8 demonstrates the switching the three EKFs used in pose estimation for the second test case. From figure 5.8a, you can see that the system switched to the pure rotation motion filter (RMF) quite frequently in the first half of the image sequence. This reflects the fact that the camera underwent a pure rotation around the pitch angle. The switching of filter to pure translation motion filter (TMF) from the 30^{th} to the 33^{rd} frame is due to the motion discontinuity. In addition, the trolley stopped at the 46^{th} frame and the camera was lowered to a new position until the 55^{th} frame. The IMM-based system was able to detect this motion and switched to the TMF during these time-steps except for the 47^{th} frame. It switched back to the general motion filter (GMF) after the 55^{th} frame when the trolley continued to move. More information regarding the cooperation and weighing of each filter at each time-step can be observed from figure 5.8b.

□ **End of chapter.**

Figure 5.5: Results of augmenting an artificial object into the first test image sequence using the proposed IMM-based approach. First row: The first and the last image of the first test sequence. Second row: A synthetic car, which is drawn by wire-frames, was augmented into the scene. The original and the resulting augmented reality video can be downloaded at http://www.cse.cuhk.edu.hk/~vision/demo/

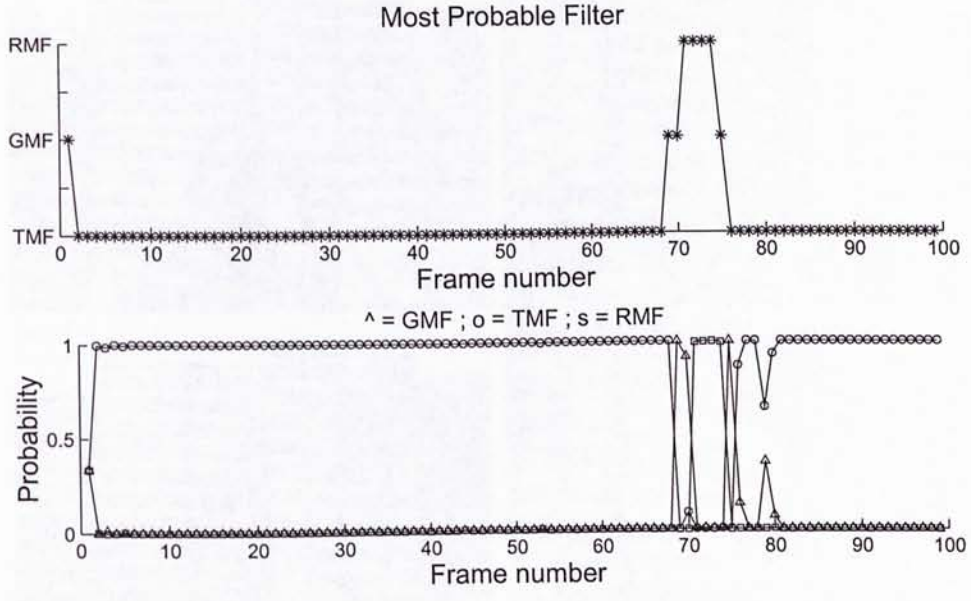

Figure 5.6: The plot on the top (Fig. 5.6a) shows the most probable EKF for pose estimation against frame number resulting from the first test image sequence. The plot at the bottom (Fig. 5.6b) shows the likelihood of each EKF for pose estimation against frame number. The line with triangle (Δ), circle (O) and square ([]) are for the general motion filter (GMF), the pure translation motion filter (TMF) and the pure rotation motion filter (RMF) respectively.

Figure 5.7: Results of augmenting an artificial object into the second test image sequence using the proposed IMM-based approach. First row: The 1^{st} and the 50^{th} image of the second test sequence. Second row: A synthetic car, which is drawn by wire-frames, was augmented into the scene. Both the original and the resulting augmented reality video can be downloaded at http://www.cse.cuhk.edu.hk/~vision/demo/

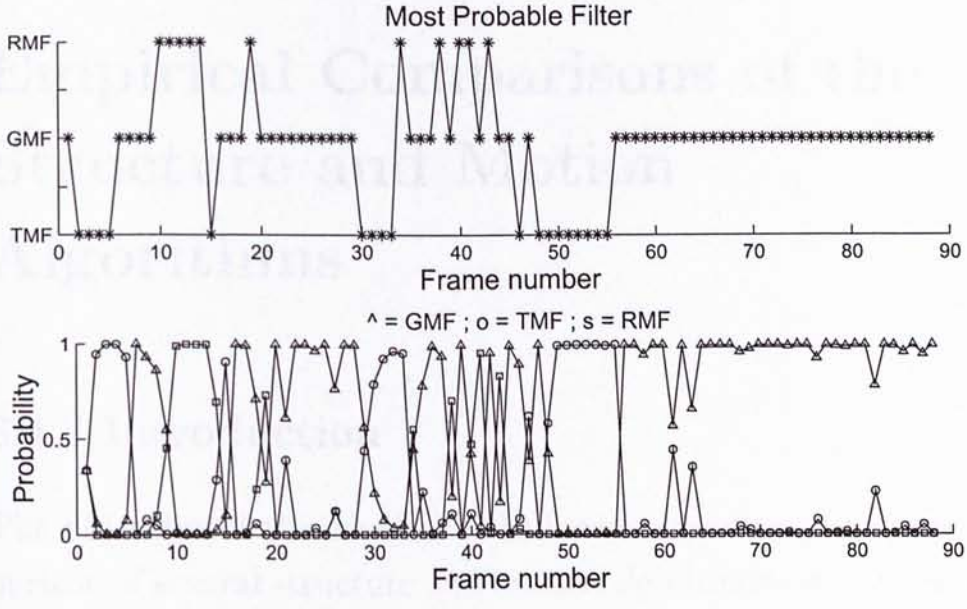

Figure 5.8: The plot on the top (Fig. 5.8a) shows the most probable EKF for pose estimation against frame number resulting from the second test image sequence. The plot at the bottom (Fig. 5.8b) shows the likelihood of each EKF for pose estimation against frame number. The line with triangle (Δ), circle (O) and square ([]) are for the general motion filter (GMF), the pure translation motion filter (TMF) and the pure rotation motion filter (RMF) respectively.

Chapter 6

Empirical Comparisons of the Structure and Motion Algorithms

6.1 Introduction

The objective of this chapter is to have a direct empirical comparison of several structure and motion algorithms introduced in this thesis. They are: *A)* the proposed 2-step extended Kalman filter (2-step EKF) in chapter 4, *B)* the Interacting Multiple Model based (IMM-based) approach in chapter 5, *C)* the interleaved bundle adjustment method [33] and *D)* the extended Kalman filter (EKF) by Azarbayejani and Pentland [8]. The evaluation is based on the resulting image residual errors, accuracy of the recovered pose parameters and the required computation time. To make the comparison fair, all the algorithms were implemented in Matlab and were run on the same Pentium III 1Ghz machine. In particular, a side-by-side comparison of the proposed 2-step EKF in chapter 4 and the IMM-based approach

in chapter 5 using real images has been made.

6.2 Comparisons Using Synthetic Data

6.2.1 Image Residual Errors

In this section, a comparison has been made using synthetic data. For consistency, the data sets in the synthetic experiments in chapter 4 and 5 were re-used. Please refer to section 4.7.1 and 5.7.1 on page 57 and 80 for detailed descriptions. The four structure and motion algorithms have been applied to these 20 independent test cases. The resulting average image residual errors, pose parameter errors and the computation time are plotted, which are shown in figure 6.1-6.4. In these figures, the line with asterisk (*), triangle (Δ), circle (O) and square ([]) markers are for the IMM-based approach, the interleaved bundle adjustment method, the EKF by Azarbayejani and Pentland and the 2-step EKF respectively.

Figure 6.1 shows the time for the four algorithms to optimize the image residual errors of the back-projected model. This plot reveals the overall accuracy of the algorithms under comparison. The two proposed algorithms, i.e. 2-step EKF in chapter 4 and the IMM-based approach in chapter 5, fall to low errors at an earlier time than the other two existing approaches.

Among the three recursive approaches (i.e. the 2-step EKF in chapter 4, the IMM-based approach in chapter 5 and the EKF by Azarbayejani and Pentland), the IMM-based approach achieves the lowest final residual error. The higher errors re-

sulting from the 2-step EKF and the EKF by Azarbayejani and Pentland is due to the fact that their algorithms were trapped into poorer local optima. In addition, the results also show that the decoupling of EKF in the IMM-based algorithm does not cause a loss in accuracy. The special structure representation, together with the IMM algorithm, are able to compensate for such a deficiency and on the other hand result in a gain in computation speed compared to traditional full covariance EKFs, in particular the EKF by Azarbayejani and Pentland.

When the recursive approaches are compared to the interleaved bundle adjustment method, which is a batch processing method, the latter approach reaches a lower final error. It is reasonable since the three recursive algorithms cannot optimize the structure and pose error for all the images in the sequence simultaneously. However, the interleaved bundle adjustment method is ranked the third in pose accuracy (see table 6.1 on page 92). It seems that their algorithm has found the solutions that overfit the data. The time taken by the four algorithms to finish the computation has also been revealed in figure 6.1. The IMM-based approach completed in 78 seconds. The interleaved bundle adjustment method, the EKF by Azarbayejani and Pentland and the 2-step EKF finished in 41 seconds (for 5 iterations), 254 seconds and 41 seconds respectively.

6.2.2 Computation Efficiency

The computation time needed to reconstruct a model when extra frames were added sequentially to the image sequence is

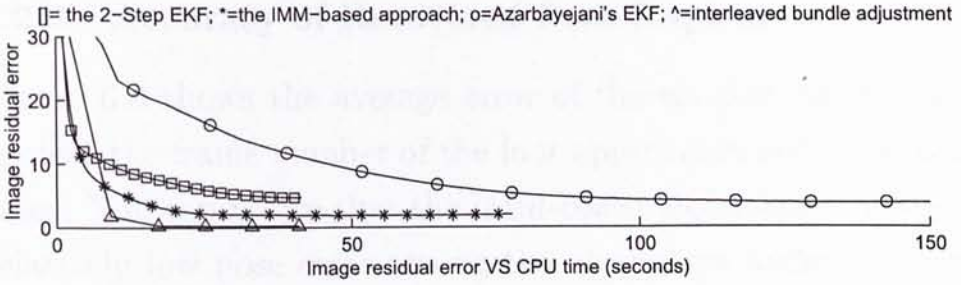

Figure 6.1: The relationship between the CPU time and the image residual error. Note that the algorithms were implemented in Matlab with a Pentium III 1GHz machine and the time measurement is in seconds

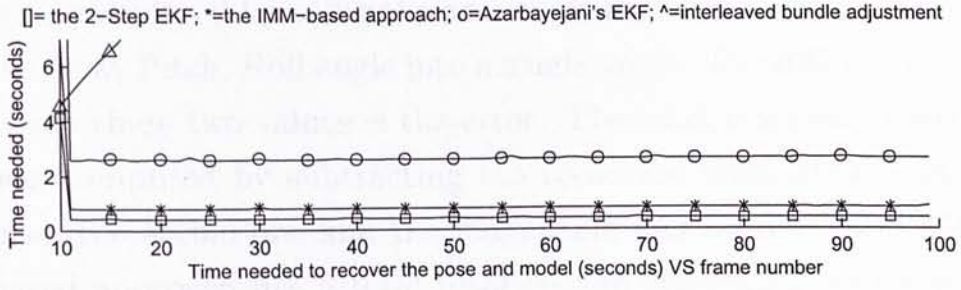

Figure 6.2: A graph showing the time needed for the four algorithms to reconstruct the model and pose when extra frames were added to the image sequence.

shown in figure 6.2. The first step in creating this plot was to reconstruct a model with the first 10 frames. The succeeding 89 frames were sequentially fed to the algorithms as the new measurements of the scene. The proposed IMM-based approach, the EKF by Azarbayejani and Pentland and the interleaved adjustment method need 0.79, 2.60 and at least 4.55 seconds to update the structure and pose of the scene for every extra frame respectively. The proposed 2-step EKF is the most computation efficient, which takes 0.42 seconds to process an extra image.

6.2.3 Accuracy of Recovered Pose Sequences

Figure 6.3 shows the average error of the six pose parameters against the frame number of the four approaches under the test cases. You can notice that the IMM-based algorithm achieves a relatively low pose error among the algorithms under comparison. To have a clearer picture, the average total rotation and translation errors versus frame number were plotted in figure 6.4. Here, the total rotations of the actual and recovered object were calculated by using the axis-angle representation to reduce the Yaw, Pitch, Roll angle into a single angle. The difference between these two values is the error. The total translation error was computed by subtracting the recovered translation vector from the actual one and the magnitude was taken. The IMM-based approach has a total rotation and translation errors not larger than 0.39 degrees and 0.41mm on each frame respectively, which are lower than the other methods most of the time in the synthetic sequences.

To quantize the comparison, the average pose parameter errors per frame for the four algorithms were computed. The values are equal to dividing the summation of errors by the number of frames. They were tabulated in table 6.1. The IMM-based approach in chapter 5 achieves the lowest total rotation and translation error per frame. For each of the individual parameters, it achieves the lowest error for the Yaw angle, Pitch angle, t_x, t_z and the second lowest for the Roll angle and t_y parameter. The EKF by Azarbayejani and Pentland and the interleaved adjustment method are respectively ranked the second and the

	2-Step EKF	IMM-based approach	Azarbayejani's EKF	Interleaved bundle adj.
Roll	3.7069	0.4055	1.0467	2.6789
Pitch	3.3599	0.2411	0.5496	1.3259
Yaw	0.3487	0.0257	0.0205	0.1141
t_x	0.2	0.0750	0.1102	0.1
t_y	0.4	0.2196	0.2732	0.2
t_z	1.2	0.1867	0.2172	1.4
Total rotation	3.7747	0.3356	0.6669	1.3405
Total translation	1.4	0.32055	0.38894	1.5

Table 6.1: A table showing the average errors per frame of each pose parameter of the four algorithms under comparison. Note that the angular errors (i.e. the total rotation, the Roll, Pitch and Yaw angle error) are in degrees and the translational errors (i.e. the total translation, t_x, t_y and t_z error) are in meters

third in the accuracy of the recovered pose. The 2-step EKF in chapter 4 has the highest error because the decoupling of the EKFs in the implementation is actually a tradeoff between speed and accuracy.

6.3 Comparisons Using Real Images

A comparison on the recovered pose sequences between the 2-step EKF proposed in chapter 4 and the IMM-based approach in chapter 5 has been made using real images. The reason for only comparing these two algorithms in real situations is that

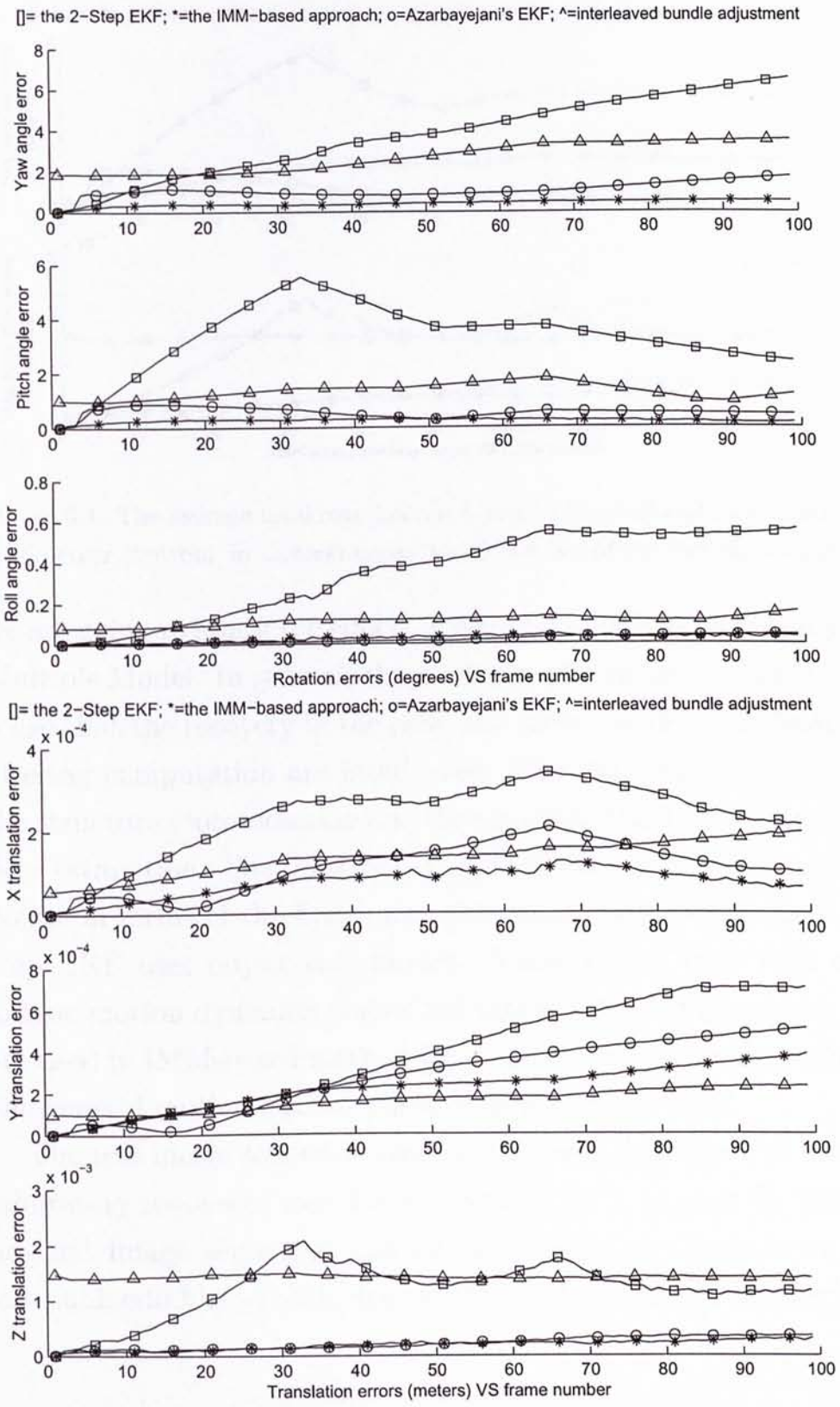

Figure 6.3: The average error of each recovered pose parameter versus frame number of the four algorithms.

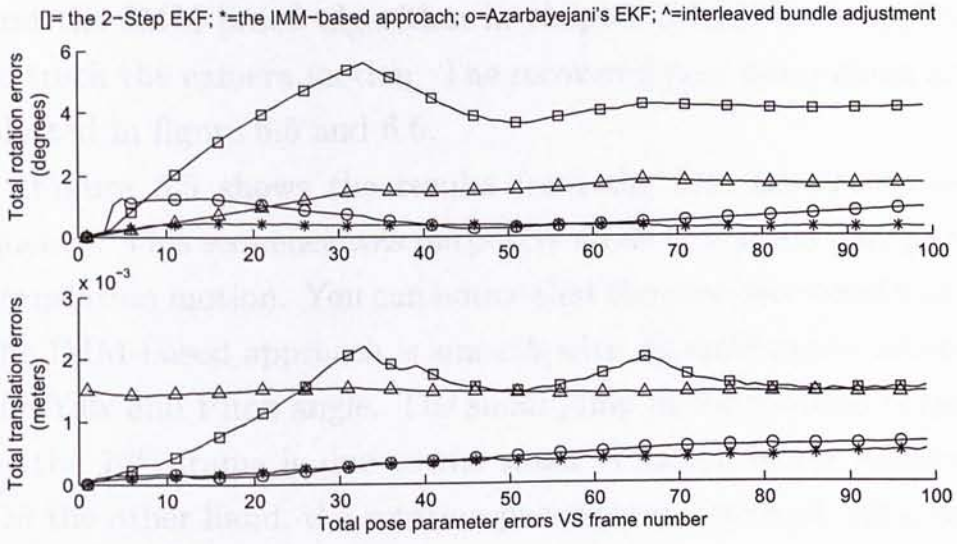

Figure 6.4: The average total rotation error (top, in degrees) and total translation error (bottom, in meters) versus frame number of the four algorithms

we are going to study the effects of incorporating the Interacting Multiple Model. In general, these two algorithms are similar in a sense that the recovery of the pose and structure in the Kalman filtering computation are interleaved. The only differences are the structure representation and the use of the IMM algorithm in pose estimation. The IMM-based approach represents 3D model points in terms of the first image that they appear while the 2-step EKF uses object coordinates. Three EKFs, each with a unique motion dynamics embedded within the IMM framework, are used in IMM-based method for pose computation while only one general motion EKF is adopted in the 2-step EKF.

The real image sequences used in the comparison are the two laboratory sequences mentioned in section 5.7.2 on page 80. The original image sequences can be downloaded at http://www-.cse.cuhk.edu.hk/~vision/demo/. The 2-step EKF in chapter 4

and the IMM-based algorithm in chapter 5 have been applied to track the camera motion. The recovered pose parameters are plotted in figure 6.5 and 6.6.

Figure 6.5 shows the results from the first laboratory sequence. This sequence was purposely made to contain only pure translation motion. You can notice that the pose recovered using the IMM-based approach is smooth with no ambiguities among the Yaw and Pitch angle. The small jump in the rotation angles at the 76^{th} frame is due to the small vibration of the camera. On the other hand, the rotation parameters recovered using the 2-step EKF is fluctuating even the actual camera motion is almost pure translation. It has a deviation about 2 degrees for the Yaw and Pitch angle. This shows that the IMM algorithm is useful in resolving ambiguities and is efficient in smoothing the output.

Figure 6.6 shows the comparison using the second laboratory sequence. This sequence of images is taken by putting the camera on a trolley. Only the height of the tripod was allowed to change. Therefore, the recovered camera motion should contain minimal Yaw angle rotation. From the figure, the pose sequence tracked by the 2-step EKF has an overall Yaw angle 2 degrees larger than that of the IMM-based approach. The increase in the Yaw angle causes a decrease in the overall Pitch angle, which is similar to the ambiguities resulting from the 2-step EKF approach in the first test sequence. However, it is less observable in this test sequence since the camera motion in this sequence involves a large change in the Pitch angle. The IMM-based approach can solve such an ambiguity in this test case. Another

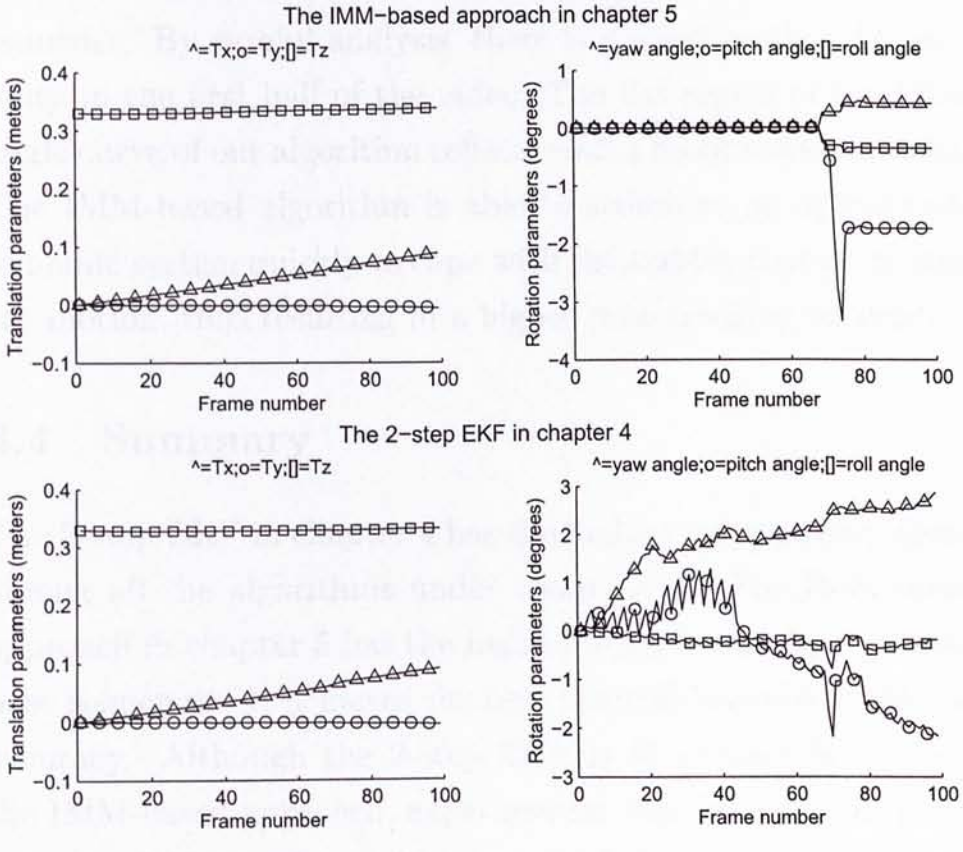

Figure 6.5: The pose sequences recovered from the first laboratory sequence with the IMM-based approach (the top two plots) and with the 2-step EKF (the bottom two plots). The line with triangle (Δ), circle (O) and square ([]) on the left column are for the translation parameter t_x, t_y and t_z respectively while the line with triangle (Δ), circle (O) and square ([]) on the right column are respectively for the Yaw, Pitch and Roll angle.

difference between the results of the two approaches is that the Pitch angle plot flattens from the 28^{th} to the 33^{rd} frame in the IMM-based approach but not in the 2-step EKF. To see which algorithm is correct, we should investigate the original video sequence. By careful analysis, there is a small motion discontinuity in the first half of the video. The flat region of the Pitch angle curve of our algorithm reflects such a motion discontinuity. The IMM-based algorithm is able to switch to an appropriate dynamic system quickly to cope with the sudden change in camera motion, thus resulting in a higher pose tracking accuracy.

6.4 Summary

The 2-step EKF in chapter 4 has the highest computation speed among all the algorithms under comparison. The IMM-based approach in chapter 5 has the highest accuracy in the recovered pose sequences. It achieves the best tradeoff between speed and accuracy. Although the 2-step EKF is 46 percent faster than the IMM-based approach, experimental results show that it has a total rotation and translation error that are respectively 11.2 and 4.4 times larger than the IMM-based approach. In addition, the IMM-based approach can recover a more reasonable pose sequence in real situations. It can be concluded that the incorporation of the Interacting Multiple Model can make substantial improvements on the algorithm performance.

□ **End of chapter.**

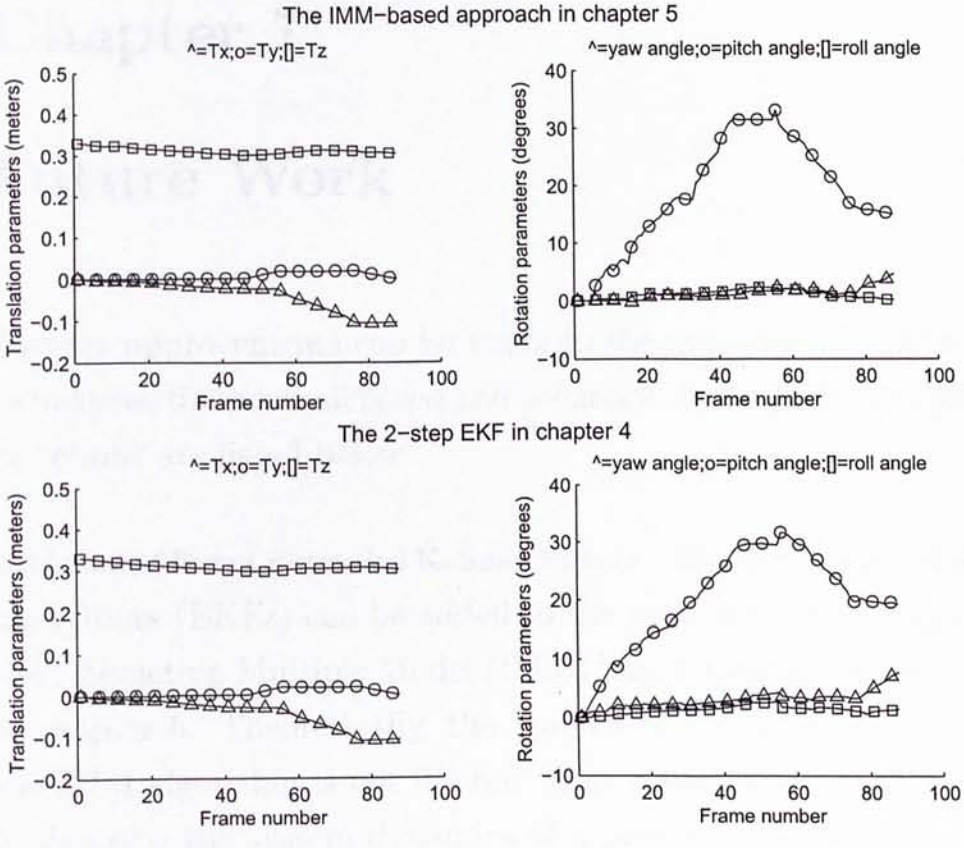

Figure 6.6: The pose sequences recovered from the second laboratory sequence with the IMM-based approach (the top two plots) and with the 2-step EKF (the bottom two plots). The line with triangle (Δ), circle (O) and square ([]) on the left column are for the translation parameter t_x, t_y and t_z respectively while the line with triangle (Δ), circle (O) and square ([]) on the right column are respectively for the Yaw, Pitch and Roll angle

Chapter 7

Future Work

Further improvements can be made to the proposed algorithms to increase their overall speed and accuracy. Some of the possible directions are listed below:

Addition of Extra Extended Kalman Filters Extra extended Kalman filters (EKFs) can be added to the pose estimation step of the Interacting Multiple Model (IMM) based approach proposed in chapter 5. Theoretically, the number of EKFs embedded in the IMM algorithm is not limited. If an additional EKF is able to describe the system dynamics of a particular application, it can be incorporated into the algorithm to improve the accuracy. The literature in [56] describes some sophisticated dynamic systems that may be useful for this purpose. The addition of the EKFs does not increase the speed too much if the algorithm is implemented on a parallel processing system, since the filters can be run concurrently.

Point Mismatch Filtering Using Epipolar Constraints A point mismatch filtering scheme can be added to the KLT tracker [39] used by the recursive algorithms in chapter 4 and 5. The point mismatches that arise from the process of feature tracking cause a significant loss in accuracy of the recovered structures and pose sequences. The flaws in the reconstructed 3D model in figure 4.3 are the results of point mismatches. Epipolar constraints can be employed to eliminate any wrong correspondences among the tracked features. A similar method has been proposed by Gibson et al in [42].

Advanced Feature Selection Strategies A more sophisticated feature selection scheme, for example the technique described in [31], can be used to choose reliable point features in the pose estimation step of Kalman filter based algorithms in chapter 4 and 5. In these algorithms, only a fixed portion of point features are used for computing the pose. If that new scheme can sort out the reliable point features at a high accuracy, these two algorithms can depend on a smaller number of points in the pose estimation step, resulting in an overall increase in the computation speed.

□ **End of chapter.**

Chapter 8

Conclusion

Three algorithms solving the pose estimation and the structure and motion problem have been proposed in this thesis. Firstly, a method for computing the pose of an object using a genetic algorithm has been introduced. The embedded mismatch filtering strategy makes the algorithm robust under the presence of point mismatches and outliers. Secondly, a 2-step recursive 3D structure acquisition algorithm using two sets of interleaved Kalman filters has been proposed. It achieves linear time and space complexity in terms of the number of available point features. Thirdly, an extension to the previously proposed 2-step algorithm has been made using the Interacting Multiple Model (IMM) framework. With that, ambiguities among the recovered structure and pose parameters are reduced. Motion discontinuities in the image sequences can also be handled elegantly.

Among the three proposed approaches, the genetic algorithm in chapter 3 is the most efficient in avoiding local optima, excluding outliers and point mismatches. The 2-step extended Kalman filter (EKF) in chapter 4 has the highest computation

speed. It is shown in the experiments that it needs less than one-forth the time of the EKF by Azarbayejani and Pentland [8] to process a sequence containing a few hundred point features per frame. The IMM-based algorithm in chapter 5 achieves the highest accuracy in the recovered pose sequences. Experimental results show that it has a total rotation and translation errors not larger than half a degree and half millimeters on each frame respectively, which are lower than the other existing methods. It is the best tradeoff between speed and accuracy among all the structure and motion algorithms under comparison. These three methods have been applied individually to 3D model reconstruction and augmented reality applications.

To proceed further, additional EKFs can be incorporated into the IMM-based algorithm to improve the accuracy, if these extra EKFs are capable of describing the system dynamics of that particular application. The addition of the filters does not reduce the speed too much since the filters can be run concurrently on parallel processing systems. Besides, improvements to the feature extraction and tracking strategy can be made. Feature extraction and tracking are significant in the recovery of structure and motion from real images. One possible approach is to make a filter on the top of the original KLT tracker to eliminate the outlying point features with the epipolar constraints.

□ **End of chapter.**

Appendix A

Kalman Filtering

This appendix aims to introduce the techniques of Kalman filtering to readers. In particular, its application to the structure and motion problem is highlighted.

Kalman filter is an estimator for the linear-quadratic-gaussian problem. It is a problem of estimating the instantaneous state of a linear dynamic system, which is perturbed by Gaussian white noise, using measurements linearly related to the state, but corrupted by Gaussian white noise [28]. There are several variations of Kalman filters. The basic one is for linear systems. Other variations include the extended Kalman filter (EKF), the iterated extended Kalman filter (IEKF), the linearized Kalman filter and the hierarchical Kalman filter (HKF).

To apply Kalman filters to a particular problem, the dynamic system and measurement model should be defined first. Let X_t be an n-dimension state of the dynamic system and ϵ_t be m-dimension vector of measurements that can be obtained physically. The linear system that describes the transition of states and the relation between the state and physical measurements

can be written as follows:

A linear system

System model:

$$X_t = M_t X_{t-1} + \gamma_t \tag{A.1}$$

Measurement model:

$$\epsilon_t = C_t X_t + \upsilon_t \tag{A.2}$$

M_t is a time-dependent $n \times n$ state transition matrix and γ_t is an n-dimension vector that accounts for the noise associated with the dynamic system. C_t is a time-dependent $m \times n$ transformation matrix between the state and the measurement while υ_t is the noise associated with the measurement process. The goal of the filter is to find an estimate of the state vector X_t, represented by $\hat{X}_t$, that minimizes the weighted mean-squared error:

$$E\left\langle [X_t - \hat{X}_t]^T U [X_t - \hat{X}_t] \right\rangle \tag{A.3}$$

The function $E\langle X \rangle$ is the expectation of X. U is any symmetric non-negative definite weighting matrix. After a series of mathematical derivations, the recursive form of Kalman filter comes up with the following four equations:

The baseline Kalman filter

Prediction equations:

$$\hat{X}_{t,t-1} = M_t \hat{X}_{t-1,t-1} \tag{A.4}$$

$$\wedge_{t,t-1} = M_t \wedge_{t-1,t-1} M_t^T + Q_t \tag{A.5}$$

Update equations:

$$\hat{X}_{t,t} = \hat{X}_{t,t-1} + W(\epsilon_t - C_t\hat{X}_{t,t-1}) \quad \text{(A.6)}$$

$$\wedge_{t,t} = \wedge_{t,t-1} - WC_t \wedge_{t,t-1} \quad \text{(A.7)}$$

$$W = \wedge_{t,t-1} C_t^T (C_t \wedge_{t,t-1} C_t^T + R_t)^{-1} \quad \text{(A.8)}$$

$\wedge_{t,t}$ is the covariance of the state vector $\hat{X}_{t,t}$. $\wedge_{0,0}$ is given by $\wedge_{0,0} = E\langle X_0 X_0^T \rangle$. Q_t and R_t are the covariances of the noise terms γ_t and υ_t respectively. W is known as the Kalman gain. Intuitively, it controls how much the filter believes on the previous estimate or the current measurements. The higher the covariance of the measurement noise R_t, the smaller the value of the Kalman gain matrix and the larger is the extent of the filter to believe on the previous estimate, since the noise of the dynamic system is relatively small. The version of Kalman filter discussed is a discrete one and is valid for linear systems only.

The measurement model of a machine vision system is non-linear if perspective camera model is adopted. To deal with non-linear measurement model, the extended Kalman filter (EKF), the linearized Kalman filter or the iterated extended Kalman filter (IEKF) should be used. Assume that the system now becomes:

A non-linear system

System model:

$$X_t = M_t X_{t-1} + \gamma_t \quad \text{(A.9)}$$

Measurement model:

$$\epsilon_t = c_t(X_t) + \upsilon_t \quad \text{(A.10)}$$

$c_t(X_t)$ is a non-linear observation function. Note that the system model is linear in this example. A first order expansion can be used to approximate the non-linear function $c_t(X_t)$ such that:

$$\nabla c_X = \left. \frac{dc_t(X_t)}{dX_t} \right|_{X_t = \hat{X}_{t,t-1}} \tag{A.11}$$

With this approximation, the Kalman filtering equations for the extended Kalman filter become:

The extended Kalman filter

Prediction equations:

$$\hat{X}_{t,t-1} = M_t \hat{X}_{t-1,t-1} \tag{A.12}$$

$$\wedge_{t,t-1} = M_t \wedge_{t-1,t-1} M_t^T + Q_t \tag{A.13}$$

Update equations:

$$\hat{X}_{t,t} = \hat{X}_{t,t-1} + W(\epsilon_t - c_t(\hat{X}_{t,t-1})) \tag{A.14}$$

$$\wedge_{t,t} = \wedge_{t,t-1} - W \nabla c_X \wedge_{t,t-1} \tag{A.15}$$

$$W = \wedge_{t,t-1} \nabla c_X^T (\nabla c_X \wedge_{t,t-1} \nabla c_X^T + R_t)^{-1} \tag{A.16}$$

Since the system model is linear, the prediction equations in this EKF are the same as the baseline Kalman filter. This is enough for the machine vision applications introduced in this thesis. If the state transition matrix M_t is replaced by a non-linear function $m_t(X_t)$, the first order expansion of the function is also valid to approximate its value:

$$\nabla m_X = \left. \frac{dm_t(X_t)}{dX_t} \right|_{X_t = \hat{X}_{t,t-1}} \tag{A.17}$$

The matrices M_t in equation A.12 and A.13 are replaced by ∇m_X. Then the EKF can handle both non-linear dynamic system and non-linear observations.

The linearized Kalman filter has a function similar to the extended Kalman filter. The only difference between them is that the former one uses linear approximation over a larger range of state space. It assumes linearity over the range of the trajectory perturbations, together with state estimation errors. The latter one makes the linearity assumption only over the range of state estimation. This discrepancy contributes to the fact that the linearized Kalman filter can be implemented more efficiently as some of its components can be pre-computed offline. On the other hand, the extended Kalman filter is proficient in a way that it is more robust against non-linear approximation errors. Thus, it is preferred to be used in computer vision systems.

For the iterated extended Kalman filter (IEKF), it is actually an extension of the EKF. The difference between EKF and IEKF is that the latter one re-evaluates the filter around the new state estimates until little extra improvements are obtained. Thus, IEKF should have a higher accuracy but a slower speed. Detailed mathematical derivation of the Kalman filters can be found in [28].

□ **End of chapter.**

Bibliography

[1] C.G.Harris and J.M.Pike, "3D positional integration from image sequence", Image and Vision Computing, vol. 6, no. 2, 1988.

[2] C.Tomasi and T.Kanade, "Shape and motion from image streams under orthography: A factorization method", International Journal of Computer Vision, vol. 9, no. 2, pp. 137-154, 1992.

[3] J.Costeira and T.Kanade, "A multibody factorization method for independently moving objects", International Journal of Computer Vision, vol. 29, no. 3, pp. 159-179, 1998.

[4] J.Shi, C.Tomasi, "Good Features to Track", in proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 593-600, Seattle, 1994.

[5] P.Sturm, "Mixing catdioptric and perspective cameras", Workshop on Omni-directional Vision, Copenhagen, Denmark, 2002.

[6] J.N.pan, Y.Q.Shi and C.Q.Shu, "A Kalman filter in motion analysis from stereo image sequences", in proceedings of the

IEEE International Conference on Image Processing, pp. 63-67, 1994.

[7] A.Chiuso, P.Favaro, H.Jin and S.Soatto, "Structure from motion casually integrated over time", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 24, no. 4, 2002.

[8] A.Azarbayejani and A.P.Pentland, "Recursive estimation of motion, structure, and focal length", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 17, no. 6, June 1995.

[9] J.I.Thomas and J.Oliensis, "Recursive multi-frame structure from motion incorporating motion error", in proceedings of DARPA Image Understanding Workshop, 1992.

[10] J.Weng, N.Ahuja and T.S.Huang, "Optimal motion and structure estimation", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 15, no. 9, September 1993.

[11] T.J.Broida, S.Chandrasiiekhar and R.Chellappa, "Recursive 3-D motion estimation from monocular image sequence", IEEE Transactions on Aerospace and Electronic Systems, vol. 26, no. 4, July 1990.

[12] T.Jebara, A.Azarbayejani and A.Pentland, "3D structure from 2D motion", IEEE Signal Processing Magazine , vol. 16, no. 3, pp. 66-84, May 1999.

[13] R.Hartley and A.Zisserman, *Multiple View Geometry in Computer Vision*, Cambridge University Press, 2000.

[14] Z.Zhang and G.Xu, *Epipolar Geometry in Stereo, Motion and Object Recognition: A Unified Approach*, Academic Publishers, 1996.

[15] O.Faugeras, Q.T.Luong and T.Papadopoulo, *The Geometry of Multiple Images: The Laws that Govern the Formation of Multiple Images of a Scene and some of Their Applications*, MIT Press.

[16] C.J.Poelman and T.Kanade, "A paraperspective factorization method for shape and motion recovery", IEEE Transactions on Pattern Analysis Machine Intelligence, vol. 19, no. 3, pp. 206-218, March 1997.

[17] P.Sturm and B.Triggs, "A factorization based algorithm for multi-image projective structure and motion", in proceedings of the European Conference on Computer Vision, 1996.

[18] G.Strang, *Introduction to Linear Algebra*, 2^{nd} edition, Wellesley-Cambridge Press, 1998.

[19] M.Pollefeys, "Self-calibration and metric 3D reconstruction from uncalibrated image sequences", PhD thesis, K.U.Leuven, 1999.

[20] D.D.Goldberg, *Genetic algorithm in search, optimization and machine learning*, Addison-Wesley, Reading, MA, Wokingham, 1989.

[21] S.Hati and S.Sengupta, "Robust camera parameter estimation using genetic algorithm", Pattern Recognition Letters, vol. 22, pp. 289-298, 2001.

[22] F.Toyama, K.Shoji and J.Miyamochi, "Model-based pose estimation using genetic algorithm", in proceedings of the International Conference on Pattern Recognition, pp. 198-201, 1998.

[23] Q.Ji and Y.Zhang, "Camera calibration with genetic alglorithm", IEEE Transactions on Systems, Man and Cybernetics—Part A: Systems and Humans, vol. 31, no. 2, March 2001.

[24] P.Cerveri, A.Pedotti and N.A Borghese, "Combined evolution strategies for dynamic calibration of video-based measurement systems", IEEE Transactions on Evolutionary Computation, vol. 5, no. 3, June 2001.

[25] E.Trucco and A.Verri, *Introductory Techniques for 3-D Computer Vision*, Prentice Hall, 1998.

[26] R.Szeliski and S.B.Kang, "Shape ambiguities in structure from motion", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 19, no. 5, May 1997.

[27] K.H.Wong, S.H.Or and M.M.Y.Chang, "Pose tracking for virtual walk-through environment construction", in proceedings of the International Conference on Inverse Problems, Hong Kong, China, January 2002.

[28] M.S.Grewal and A.P.Andrews, *Kalman Filtering Theory and Practice*, Prentice Hall, 1993.

[29] S.Tsutsui and Y.Fujimoto. "The-p-fGA: Phenotypic forking genetic algorithm", JSAI, vol. 11, no. 4, pp. 619-628, 1996.

[30] P.A. Beardsley, A.Zisserman and D.W.Murray, "Sequential updating of projective and affine structure from motion", International Journal of Computer Vision, vol. 23, pp. 235-259, 1997.

[31] V.Lippiello, B.Siciliano and L.Villani, "Objects motion estimation via BSP tree modeling and Kalman filtering of stereo images", in proceedings of the IEEE International Conference on Robotics and Automation Washington DC, pp. 2968-2973, 2002.

[32] V.Lippiello, B.Siciliano and L.Villani, "Position and orientation estimation based on Kalman filtering of stereo images", in proceedings of the IEEE International Conference on Control Applications, pp. 702-707, Mexico City, 2001.

[33] M.Y.Y.Chang and K.H.Wong, "Model reconstruction and pose acquisition using extended Lowe's method", IEEE Transactions on Multimedia (to appear).

[34] X.Zhang, S.Fronz and N.Navab, "Visual marker detection and decoding in AR systems: A comparative study", in proceedings of the International Symposium on Mixed and Augmented Reality, Darmstadt, Germany, September 2002.

[35] B.Triggs, P.McLauchlan, R.Hartley and A.Fitzgibbonm, "Bundle adjustment—A modern synthesis", in proceedings of the International Workshop on Visual Algorithm: Theory and Practice, pp. 298-372, Corfu Greece, 1999.

[36] M.A.Fischler and R.C.Bolles, "Random sample concensus: A paradigm for model fitting with applications to image analysis and automated cartography", Communications of the ACM, vol. 24, no. 6, pp. 882-887, June 1981.

[37] R.Horaud, B.Conio and O.Leboulleux, "An analytic solution for the perspective 4-point problem", Computer Vision, Graphics and Image Processing, vol. 47, pp. 33-44, 1989.

[38] C.Tomasi and T.Kanade, "Detection and tracking of point features", Carnegie Mellon University Technical Report CMU-CS-91-132, April 1991.

[39] KLT: An Implementation of the Kanade-Lucas-Tomasi Feature Tracker (http://vision.stanford.edu/ birch/klt/).

[40] Camera Calibration Toolbox for Matlab (http://www.vision.caltech.edu/bouguetj/calib_doc/).

[41] D.G.Lowe, "Fitting parameterized three-dimensional models to images", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 13, no. 5, pp. 441-450, May 1991.

[42] S.Gibson, J.Cook, T.Howard, R.Hubbold and D.Oram, "Accurate camera calibration for off-line, video-based augmented reality", in proceedings of the International Symposium on Mixed and Augmented Reality, Darmstadt, Germany, September 2002.

[43] M.L.Liu and K.H.Wong, "Pose estimation using four corresponding points", Pattern Recognition Letters, vol. 20, no. 1, pp. 69-74, January 1999.

[44] S.H.Or, W.S.Luk, K.H.Wong and I.King, "An efficient iterative pose estimation algorithm", Image and Vision Computing Journal, vol. 16, no. 5, pp. 355-364, May 1998.

[45] M.Pollefeys, "Tutorial on 3D Modeling from Images", in conjunction with the European Conference on Computer Vision, June 2000.

[46] E.Mazor, A.Averbuch, Y.Bar-Shalom and J.Dayan, "Interacting multiple model methods in target trackings: A Survey", IEEE Transcations on Aerospace and Electrons Systems, vol. 34, no. 1, pp. 103-123, January 1998.

[47] K.J.Bradshaw and I.D.Reid, "The active recovery of 3D motion trajectories and their use in prediction", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 19, no. 3, pp. 219-234, March 1997.

[48] M.E.Farmer, R.L.Hsu and A.K.Jain, "Interacting multiple model (IMM) Kalman filters for robust high speed human motion tracking", in proceedings of the International Conference on Pattern Recognition, pp. 20-23, Quebec City, 2002.

[49] H.A.P.Blom, "An efficient filter for abruptly changing systems", in proceedings of the 23^{rd} IEEE Conference on Decision and Control, pp. 656-658, Las Vegas, NV, December 1984.

[50] E.Daeipour and Y.Bar-Shalom, "An interacting multiple model approach for target tracking with glint noise", IEEE

Transcations on Aerospace and Electrons Systems, vol. 31, no. 2, pp. 706-715, April 1995.

[51] A.J.Davison and D.W.Murray, "Simultaneous localization and map-building using active vision", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 24, no. 7, July 2002.

[52] D.Reiners, D.Stricker, G.Klinker, and S.Muller, "Augmented reality for construction tasks: Doorlock assembly", in proceedings of IEEE International Workshop on Augmented Reality, 1998.

[53] H.Kato and M.Billinghurst, "Marker tracking and hmd calibration for a video-based augmented reality conferencing system", in proceedings of IEEE International Workshop on Augmented Reality, pp. 125-133, 1999.

[54] X.Zhang, N.Navab, and S.Liou, "E-commerce direct marketing using augmented reality", in proceedings of IEEE International Conference on Multimedia and Expo., 2000.

[55] S.Princel, A.D.Cheok, F.Farbiz and T.Williamson, "3D live: Real time captured content for mixed reality", in proceedings of the International Symposium on Mixed and Augmented Reality, pp. 7-13, Darmstadt, Germany, September 2002.

[56] X.R.Li, V.P.Jilkov, "Survey of maneuvering target tracking part I: Dynamic model", IEEE Transactions on Aerospace and Electronics Systems, vol. 39, no. 4, October 2004.