3D Coarse-to-Fine Reconstruction from Multiple Image Sequences

IP Che Yin

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

Abstract of thesis entitled:

3D Coarse-to-Fine Reconstruction from Multiple Image Sequences

Submitted by IP Che Yin

for the degree of Master of Philosophy

at The Chinese University of Hong Kong in September 2004

Structure from motion is the problem of recovering the 3D structure from an image sequence for a scene. The literature proposes a lot of methods for this problem and they do obtain good results. However, because of the limited resolution of digital images from common domestic use cameras, the detailed reconstruction for a large environment is still difficult. Some methods separate the reconstructions for the detailed objects in the scene from that for the large environment, and then combine them together manually or semi-automatically. Other methods improve the reconstruction for the large environment using additional constraints, such as planes and cylinders. However, these methods are only applicable to particular applications.

This thesis addresses the problem of detailed reconstructions for a large environment and proposes a framework that reconstructs the detailed objects as well as the large environment automatically. We first take two image sequences, one for the large scene and another one for the detailed object in the scene. Subsequently, we reconstruct the 3D structures of the large scene and the detailed object independently. Then, we find the point correspondences between the two image sequences using multi-scale image matching techniques. Finally, the 3D structures from the two different scales are merged together using the point correspondences, and the overall structure with fine details is recovered.

In this thesis, the techniques for 3D reconstruction and image matching of different resolutions will be described. Synthetic experiments on the rematching algorithm and the 3D model registration were performed and will be reported. Experiments on real images have shown that we can successfully build up the 3D structure of a large environment with fine details.

香港中文大學

計算機科學與工程學課程

哲學碩士論文

二零零四年九月

論文題目： 利用由粗糙到細緻的多影像序列來重建場景的三維結構

作者： 葉子筵

摘要：

「由運動到結構」是一個怎樣利用一個場景的影像序列去重建這場景的三維結構的問題。許多學者發表了很多解決這問題的方法，而且得到不錯的成績。但是，由於家用相機的解象度是有限的，怎樣去重建一個廣闊的場景的三維結構始終都是一個困難的課題。有一些方法先分開重建廣闊的場景和場景內細緻的物件的三維結構，然後將它們用全人手或半人手結合。有另一些方法利用一些另加的條件，例如平面和柱狀，去改善廣闊的場景的三維結構；可是，這方法只適用於特定的應用。

這篇論文發表廣闊環境的細緻三維重建的問題，以及提出一個自動地重建廣闊的場景和細緻的物件的方法。我們首先拍攝兩輯影像序列；一輯是爲廣闊的場景拍攝的，而另一輯是爲場景內細緻的物件拍攝的。然後，我們獨立地重建它們的三維結構。接著，我們利用多尺度影像配對方法去尋找這兩輯影像序列之間的點關係。最後，利用這些點關係去將那兩個不同尺度的三維結構結合。

這篇論文將會解釋三維結構重建的技術和不同解象度的影像配對的技術。我們亦進行了兩個重新點對點配對和三維結構的配對的人工資料實驗。一個利用真實的影像序列的實驗亦証明了我們的方法可以重建一個廣闊環境的細緻的三維結構。

Acknowledgement

Many people have contributed to my education through their support in my graduate school years. I especially wish to thank my Master degree supervisor Prof. Kin-hong Wong for his guidance, suggestions, idea, opinions and support in my graduate school years.

My great gratitude goes to Prof. Irwin King and Prof. Hanqiu Sun, who marked the first version of my thesis and gave me valuable suggestions. Also, I would like to thank Prof. Michael Chang and Dr. S. H. Or for their valuable discussions and opinions.

I would like to thank the Department of Computer Science and Engineering, the Chinese University of Hong Kong. It provides the best equipment and office environment required for my research.

I would like to acknowledge Prof. Michael Chang and Prof. Kin-hong Wong for their implementation of the two-pass bundle adjustment. I also would like to acknowledge Mr. Y. K. Yu for the implementation of the Eight-point algorithm.

Moreover, I wish to acknowledge Dr. Philip Torr for the Structure and Motion Toolkit in Matlab. I used some routines of the toolkit for this project.

I would like to thank my colleagues. In particular, Mr. Y. K. Hui, Mr. C. W. Lau, Mr. C. L. Fung, Mr. C. W. Wong, Miss J. Y. Zheng, Miss K. Y. Lee, Miss P. W. Chan, Miss N. S. Lau, Mr. C. H. Chan, Mr. Y. Lam, Miss C. H. Ngai and Mr. Johnson W. Hung for their assistance and support.

Finally, I would like to thank my parents and sister for their love, warmth and encouragement.

To my family

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

1.1 Motivation

Recovering the 3D structure of an object is a popular research topic in computer vision. It is very useful in many applications including scene structure recovery, virtual reality, augmented reality and game development. Large amount of work is proposed to solve this problem including structure from motion [74, 77, 29, 18, 17, 9], shape from splines [71, 6, 67, 46, 82], and reconstruction from aerial images [31, 11]. Our main concern is the investigation of practical but accurate structure from motion approaches.

The reconstruction of the 3D structure for a large scene is important in many applications. Many algorithms could recover the rough 3D structure for a large scene from images of a fixed resolution taken by hand-held cameras. In some cases, the fine and detailed 3D structure for the large scene may also be required. However, because of the limited resolution of the digital images, the 3D structures recovered by these algorithms usually lack the fine structures even if the maximum number of reliable feature points are extracted. Also, the textures placed on these structures are poor

to view in detail.

Figure 1.1(a) shows an example of 3D reconstruction for a large scene containing several objects. Figure 1.1(b) shows the 3D reconstruction for a small object inside the large scene. The top left picture in Figure 1.1 is one of the images from the image sequence for the large scene, and the top right picture is that for the small object. The middle and the bottom pictures show their reconstructed texture-mapped 3D models and their wire-frame models respectively. The wire-frame model for the small object has a denser structure than that for the large scene. Also, its texture mapped image is better and clearer. To reconstruct the fine structures of a large scene, a coarse-to-fine approach that integrates the 3D reconstruction for the large scene and that for the detailed objects can be used.

This thesis proposes an automatic coarse-to-fine method that integrates the 3D structures of a large scene and a fine object inside the large scene in order to recover a detailed 3D structure for the whole scene.

1.2 Previous Work

This section introduces several approaches that are relevant to our approach.

1.2.1 Reconstruction for Architecture Scene

El-Hakim [14] proposed a method that first reconstructs a rough 3D model of a man-made object, and then allows the user to add specific constraints on the detailed elements of the object. The user can add a number of seed points and the type of elements manually, and the method computes the detailed 3D structure for the object. El-Hakim *et al.* [15] extended the above method by combining with the methods of range-based modelling [4, 59].

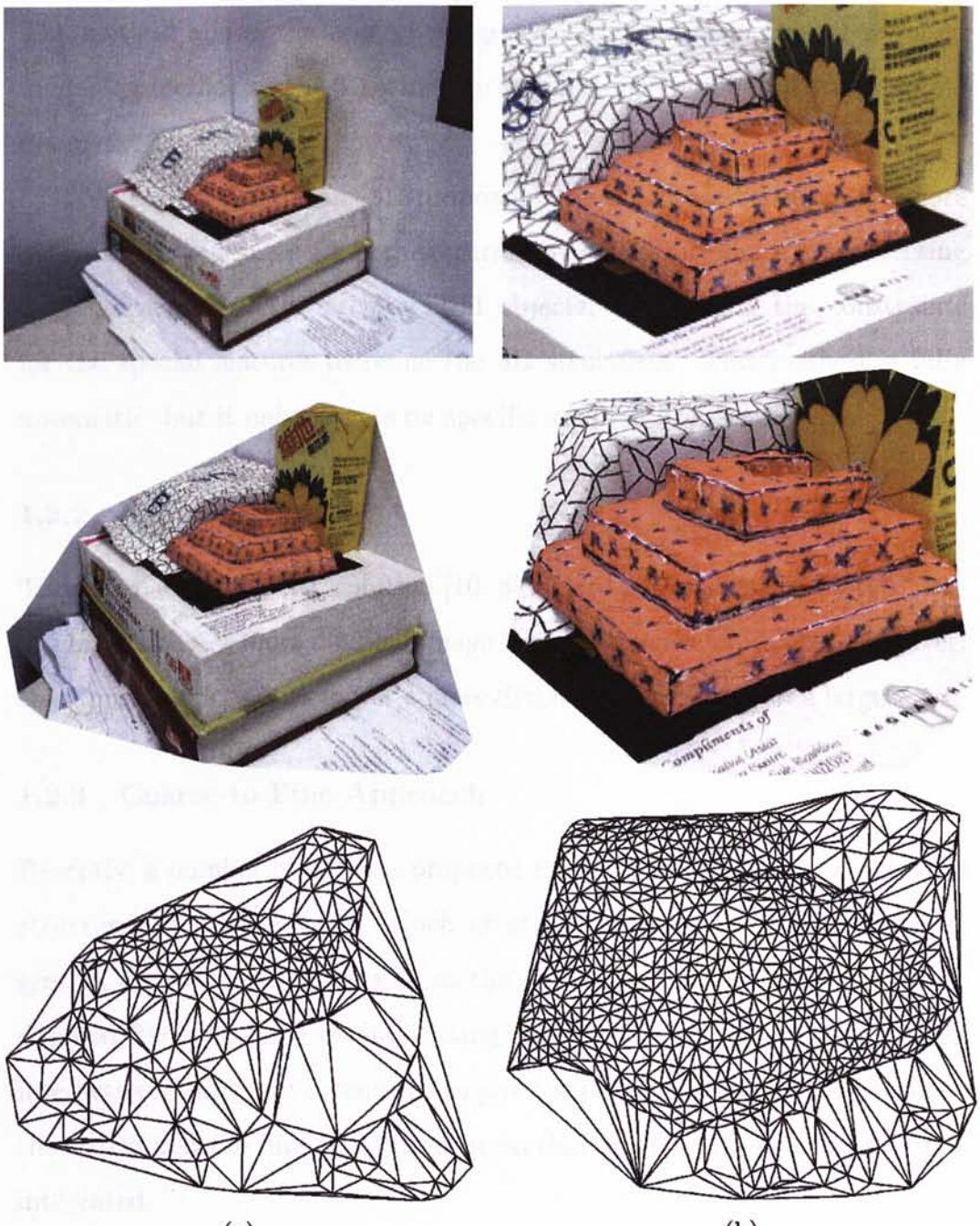


Figure 1.1: The comparison of the 3D reconstructions for (a) a coarse structure and (b) a fine structure. Each top image shows one of the image frame, the middle image shows the texture-mapped 3D model, and the bottom image shows the wire-frame model for each structure.

Range-based modelling method can recover a surface in very high detail. The method allows the user to merge the structure from the range-based modelling method to the 3D structure by selecting a number of merge points manually.

Werner and Zisserman [81] proposed a method that reconstructs more detailed 3D structures for architectural objects by automatically detecting special features of the architectural objects, and adding the constraints for the special features to refine the 3D structures. This method is fully automatic, but it only targets on specific architectural scenes.

1.2.2 Super-resolution

The approach of super-resolution [10, 8] is used to construct a "better" image or estimate a more detailed image from a sequence of images. However, this approach does not recover a more detailed 3D structure of a large scene.

1.2.3 Coarse-to-Fine Approach

Recently, a number of authors proposed methods that reconstruct detailed structures of a large scene. Koch *et al.* [30] proposed a multi-scale integration approach (also refers to as the coarse-to-fine approach) to recover detailed 3D structures by integrating the coarse and the fine 3D models interactively. The user is required to give the point correspondences between the coarse and the fine 3D structures, so that the two 3D structures can be integrated.

Ramalingam and Lodha [58] proposed an automatic hierarchical registration method of texture-mapped 3D models. This approach integrates a coarse structure and a fine structure automatically by first finding the point correspondences between the two image sequences using a scale-sensitive

image registration algorithm, and then registering the 3D models using a scale-sensitive 3D model registration algorithm. The scale-sensitive image registration algorithm registers the 2D points extracted for 3D reconstruction on the two image sequences directly, and the scale-sensitive 3D model registration algorithm estimates the metric transformation between the coarse and the fine 3D models.

This method is similar to our method, but our method differs from it in two ways. First, instead of matching the 2D points for 3D reconstruction directly, we first match a pair of images from each image sequence using the multi-scale image matching approaches [62, 40, 13, 47] and obtain the 2D transformation information between the pair of images. Then, we use a 2D rematching method and the 2D transformation aforementioned to link the 2D points that are used for 3D reconstruction. This method can make the correspondence problem between the coarse and the fine sequences more flexible such that we can use any multi-scale image matching method for the problem. Also, as described in [58], the multi-scale image matching methods can handle larger multi-scale differences in general for image matching as compared to the scale-sensitive image registration algorithm. Moreover, the scale-sensitive image registration is based on correlation, it cannot match the images with a rotation difference.

Second, instead of estimating the metric transformation between the coarse and the fine 3D models, our method estimates the affine transformation for the 3D registration. Our experiment has shown that the affine transformation provides better 3D registration than the metric transformation.

1.3 Proposed solution

In this thesis, we describe our solution to the detailed reconstruction for a large scene. In brief, we take the first (coarse) image sequence for the large scene, and the second (fine) image sequence for the object that we want to reconstruct the more detailed 3D model. Then, one image from each image sequence is selected for image matching.

The 2D image point sequence is extracted and tracked, and the 3D model is reconstructed for each image sequence. Then, one of the major steps of our solution is to match the image sequences through a multi-scale image matching algorithm on the selected pair of images. Because of the limitation of the multi-scale image matching method we used in our system, an assumption that the selected pair of images have similar view angles to the scene is required.

Using the rematching algorithm, we obtain the common point correspondences between the 2D image point sequences. Then, we register and integrate the coarse and the fine 3D models using these point correspondences.

1.4 Contribution

The following is the contribution in this thesis:

- An alternative solution to the automatic coarse-to-fine 3D registration problem is developed. Our method merges the 3D structures reconstructed from a coarse image sequence (for a large scene) and a fine image sequences (for a smaller scene inside the large scene) to build a model for a large environment with fine details. Because of using the

rematching methology to build the linkage between the coarse and the fine image sequences, the image matching step is flexible and is not restricted to one matching algorithm, so that a lot of image matching algorithms can be used in our framework.

1.5 Publications

- S. H. Or, K. H. Wong, M. M. Y. Chang, and C. Y. Ip. Large scene reconstruction with local details recovery. In *International conference on pattern recognition*, August 2004.
- C. Y. Ip, K.-H. Wong, and M. M. Y. Chang. Detailed 3d reconstruction for large environment. *IEEE Transactions on Circuits and Systems for Video Technology.* In preparation.

1.6 Layout of the thesis

Chapter 2 describes the background techniques for the proposed method, and Chapter 3 explains the details of the proposed method. Chapter 4 presents the synthetic and real experiments and their results. Finally, the thesis is concluded in Chapter 5.

□ **End of chapter.**

Chapter 2

Background Techniques

This chapter describes the background knowledge about the techniques used in our project including interest point detectors and descriptors, steerable filters, the techniques for feature point tracking, RANSAC (Random sample consensus) [16], and the structure-from-motion algorithms.

2.1 Interest Point Detectors

In order to understand an image, researchers have been proposing a lot of feature detectors for different applications. The common features include edges, corners, contours, textures, colors, regions, shapes, and etc. Most of these topics are introduced in the literature [20, 60].

In this project, interest points of an image are very useful. As defined in [65], an interest point is defined as a point feature where local image intensity changes in two orthogonal directions.

This project is about the multi-resolution image matching, so this section first introduces the concept of scale-space representation. Then, four corner detectors including the standard Harris point detector and three

modifications of the Harris point detector adapting the idea of scale-space are introduced. Moreover, a blob detector, and a corner detector for color images are introduced in this section.

2.1.1 Scale-space

Before the introduction of various interest point detectors, this section describes the concept of *scale-space* [34, 35, 12, 47].

The scale-space representation is a set of image feature points that is used to represent an image at different levels of resolutions. In general, different resolutions of an image (and so different scales of the image) can be created using the convolutions of the image with Gaussian kernel $G(x, y, \sigma)$ of different standard deviations (σ). The 2D Gaussian function is defined in Equation 2.1.

$$G(x, y, \sigma) = \exp^{-(x^2+y^2)/2\sigma}. \tag{2.1}$$

where (x, y) is the position relative to the center of the kernel. Thus, the different resolutions of an image is computed by Equation 2.2.

$$L(x, y, \sigma) = G(x, y, \sigma) \star I(x, y), \tag{2.2}$$

where $I(x, y)$ is the intensity of the image at position (x, y), and "$\star$" is a convolution operation.

For the Gaussian derivative, let $\sigma_2 = s\sigma$ and s is the scale ratio of σ_2 to σ, the derivative of the image $I(x, y)$ with respect to directions $d_1, \ldots, d_m$

is

$$\begin{aligned} L_{d_1,\ldots,d_m}(x,y,\sigma_2) &= \frac{\partial L^m}{\partial d_1 \ldots \partial d_m}(x,y,\sigma_2) \\ &= L_{d_1,\ldots,d_m}(x,y,s\sigma) \\ &= \frac{1}{s^m} L_{d_1,\ldots,d_m}(x,y,\sigma). \end{aligned} \tag{2.3}$$

From Equation 2.3, we can easily observe that the amplitude of the derivatives decreases with scale (s), in general. To maintain the property of scale invariance, the derivative functions must be normalized with respect to the scale. Define the m-th order *scale-normalized* Gaussian derivative of the image $I(x,y)$ with respect to directions $d_1,\ldots,d_m$ of scale σ by:

$$\begin{aligned} D_{d_1,\ldots,d_m}(x,y,\sigma) &= \sigma^m L_{d_1,\ldots,d_m}(x,y,\sigma) \\ &= \sigma^m G_{d_1,\ldots,d_m}(\sigma) \star I(x,y), \end{aligned} \tag{2.4}$$

where $G_{d_1,\ldots,d_m}(\sigma)$ is the Gaussian derivative with respect to directions $d_1,\ldots,d_m$.

Using the concept of scale-space representation and the scale-normalized Gaussian derivatives, various scale-invariant interest point detectors were proposed to detect the features response to different resolutions of an image.

2.1.2 Harris Corner detectors

In this project, corners are the major concerns for the interest point detection. In this section, we first describe the standard Harris point detector, which is not scale-invariant. Then, three variations of Harris point detector including Dufournaud's scale-space Harris point detector, Lindeberg's corner detector and Harris-Laplacian corner detector, which are used to detect

interest points of different resolutions, are introduced.

Standard Harris point detector

Harris and Stephens [25] proposed an interest point detector to detect intensity changes in 2 directions on an intensity image. More precisely, consider an image point (x, y) and the associated image intensity value $I(x, y)$. The standard Harris point detector consists of the following steps:

1. Compute the image derivatives $L_x(x, y, \sigma_{\mathcal{D}})$ and $L_y(x, y, \sigma_{\mathcal{D}})$ of the whole image in the x and y directions, respectively, by the convolution with a Gaussian first order derivative kernel of standard deviation $\sigma_{\mathcal{D}}$:

$$\begin{aligned} L_x(x, y, \sigma_{\mathcal{D}}) &= I(x, y) \star G_x(x, y, \sigma_{\mathcal{D}}) \\ L_y(x, y, \sigma_{\mathcal{D}}) &= I(x, y) \star G_y(x, y, \sigma_{\mathcal{D}}) \end{aligned} \tag{2.5}$$

   where $G_x(x, y, \sigma_{\mathcal{D}})$ and $G_y(x, y, \sigma_{\mathcal{D}})$ are the Gaussian first order derivatives along x-direction and y-direction of standard deviation $\sigma_{\mathcal{D}}$ respectively.

2. Form the *auto-correlation matrix* $M_{harris}(x, y, \sigma_{\mathcal{D}}, \sigma_{\mathcal{I}})$ as shown in Equation 2.6.

$$\begin{aligned} M_{harris}(x, y, \sigma_{\mathcal{D}}, \sigma_{\mathcal{I}}) &= G(x, y, \sigma_{\mathcal{I}}) \\ &\star \begin{bmatrix} L_x^2(x, y, \sigma_{\mathcal{D}}) & L_x(x, y, \sigma_{\mathcal{D}}) L_y(x, y, \sigma_{\mathcal{D}}) \\ L_x(x, y, \sigma_{\mathcal{D}}) L_y(x, y, \sigma_{\mathcal{D}}) & L_y^2(x, y, \sigma_{\mathcal{D}}) \end{bmatrix} \end{aligned} \tag{2.6}$$

   where the Gaussian kernel of standard deviation $\sigma_{\mathcal{I}}$, $G(x, y, \sigma_{\mathcal{I}})$, acts as the weighting factor for the matrix.

3. The image position (x, y) is considered as a candidate point if the auto-correlation matrix $M_{harris}(x, y, \sigma_{\mathcal{D}}, \sigma_{\mathcal{I}})$ has two significant eigenvalues. Therefore, if the value of the *Harris function*, $C_{harris}(x, y, \sigma_{\mathcal{D}}, \sigma_{\mathcal{I}})$, in Equation 2.7 is larger than a threshold, (x, y) is considered as a candidate point.

$$C_{harris}(x, y, \sigma_{\mathcal{D}}, \sigma_{\mathcal{I}}) = \det(M_{harris}(x, y, \sigma_{\mathcal{D}}, \sigma_{\mathcal{I}})) - \alpha \text{trace}^2(M_{harris}(x, y, \sigma_{\mathcal{D}}, \sigma_{\mathcal{I}})) \quad (2.7)$$

where α is a parameter to discriminate the case of two large eigenvalues against one large eigenvalue. α is usually set to 0.04 [47]. $\det(Z)$ is a determinant operation of a square matrix Z. $\text{trace}(Z)$ is the trace of Z or the sum of the diagonal elements of Z.

4. The candidate point (x, y) is chosen as an interest point if the value of $C_{harris}(x, y, \sigma_{\mathcal{D}}, \sigma_{\mathcal{I}})$ is larger than a threshold and the maximum among its 8-neighboring pixels.

According to [65], the standard deviation $\sigma_{\mathcal{I}}$ is usually called an *integration scale* because the Gaussian kernel of the standard deviation $\sigma_{\mathcal{I}}$ in Equation 2.6 performs integral smoothing. The standard deviation $\sigma_{\mathcal{D}}$ is usually called *derivative scale* because it controls the scale of the Gaussian derivations described in Equation 2.5.

Because of the symmetry of the auto-correlation matrix M_{harris} in Equation 2.6, the standard Harris point detector is rotation-invariant. However, it is not invariant to scale change, so it is not adapted to the scale-space framework. Dufournaud *et al.*[12] proposed a modification of the standard Harris point detector which is adapted to the scale-space framework.

Scale-space Harris point detector

Harris point detector is not invariant to the change of the image resolution or the image scale. Dufournaud *et al.*[12] proposed the scale-space Harris point detector by changing the values of the standard deviations and normalizing the derivatives in Equation 2.6.

To detect interest points on an image of different scales, Dufournaud *et al.* [12] redefined the Harris function (Equation 2.7). Suppose $I^{(1)}(x_1, y_1) = I^{(2)}(x_2, y_2)$, where $I^{(1)}$ is the high-resolution image and $I^{(2)}$ is the low-resolution image, such that $(x_1, y_1) = (sx_2, sy_2) + (t_x, t_y)$, where t_x and t_y are the translation along x- and y-direction respectively, s is the scale ratio of $I^{(1)}$ over $I^{(2)}$. By taking the derivatives of the above expression with respect to x- and y-direction, we obtain $sL_x^{(1)}(x_1, y_1) = L_x^{(2)}(x_2, y_2)$ and $sL_y^{(1)}(x_1, y_1) = L_y^{(2)}(x_2, y_2)$. Applying these two relations into Equation 2.6, we can obtain the relation:

$$s^2 M_{harris}^{(1)}(x_1, y_1, s\sigma_{\mathcal{D}}, s\sigma_{\mathcal{I}}) = M_{harris}^{(2)}(x_2, y_2, \sigma_{\mathcal{D}}, \sigma_{\mathcal{I}}) \tag{2.8}$$

From Equation 2.4 and Equation 2.8, Dufournaud *et al.* [12] observed that the auto-correlation matrix can be modified to Equation 2.9:

$$M_{scale}(x, y, s\sigma_{\mathcal{D}}, s\sigma_{\mathcal{I}}) = s^2 G(s\sigma_{\mathcal{I}}) \star \begin{bmatrix} L_x^2(x, y, s\sigma_{\mathcal{D}}) & L_x(x, y, s\sigma_{\mathcal{D}}) L_y(x, y, s\sigma_{\mathcal{D}}) \\ L_x(x, y, s\sigma_{\mathcal{D}}) L_y(x, y, s\sigma_{\mathcal{D}}) & L_y^2(x, y, s\sigma_{\mathcal{D}}) \end{bmatrix}. \tag{2.9}$$

Then, the scale-space Harris function of scale s is defined in Equation 2.10:

$$C_{scale}(x, y, s\sigma_{\mathcal{D}}, s\sigma_{\mathcal{I}}) = \det(M_{scale}(x, y, s\sigma_{\mathcal{D}}, s\sigma_{\mathcal{I}})) - \alpha \text{trace}^2(M_{scale}(x, y, s\sigma_{\mathcal{D}}, s\sigma_{\mathcal{I}})) \quad (2.10)$$

To detect interest points on an image of different scales, we apply the framework of the standard Harris point detector but replace the Harris function in Equation 2.7 with the scale-space Harris function in Equation 2.10 of the corresponding values of s. For the values of $\sigma_{\mathcal{D}}$ and $\sigma_{\mathcal{I}}$, Dufournaud *et al.*[12] suggested that they are set to 1 and 2 respectively.

Lindeberg's point detector

Lindeberg [34] observed that a scale-normalized derivative function of an interest point may consist of local maxima along scales. The scale corresponding to the maximum response of the scale-normalized derivative function reflects the characteristic length of an interesting computational structure at a single spatial point. Lindeberg called this scale as the *characteristic scale.* The characteristic scale of a feature point is the scale of the local maximum over scales of a derivative function, $F(x, y, s_n)$ of that point. Scale-space Harris function (Equation 2.10), square gradient (Equation 2.11), Different-of-Gaussian (DoG) (Equation 2.12) and Laplacian function (Equation 2.13)

are the examples of the derivative functions.

$$F_{sg}(x, y, s_n) = s_n^2(L_x^2(x, y, s_n) + L_y^2(x, y, s_n)) \quad (2.11)$$

$$F_{DoG}(x, y, s_n) = |I(x, y) \star G(x, y, s_n - 1) - I(x, y) \star G(x, y, s_n)| \quad (2.12)$$

$$F_{laplacian}(x, y, s_n) = |s_n^2(L_{xx}(x, y, s_n) + L_{yy}(x, y, s_n))| \quad (2.13)$$

Lindeberg [34] designed a framework that detects the interest points at their characteristic scales. In order to maintain uniform information change between two successive levels of resolution, according to [47], the scale factor s_n should be distributed exponentially, such that $s_n = k^n s_0$ with s_0 is the scale factor of the finest level of resolution, and k is the factor of scale change between 2 successive levels. The framework consists of two steps:

Step 1: (Response Calculation) For every level of resolutions (s_n), compute the scale-normalized derivative function $F(x, y, s_n)$ for all pixels of the image.

Step 2: (Point Selection) For every level of resolutions (s_n), select a point (x, y) if the response of the scale-normalized derivative function is larger than a threshold and reaches the local maximum among the successively larger and smaller scales, and its 8-neighboring pixels.

The scale-normalized derivative function at the characteristic scales of interest points on images of different resolutions may have the same response if the interest points correspond to the same point in the scene. Therefore, the concept of characteristic scale is useful for image matching by reducing the search space of feature points.

Harris-Laplacian point detector

For the application of image matching, Dufournaud's scale-space Harris point detector described above is not efficient because it scans every interest points at every scale, and so creates additional mismatches and increases the storage space for the scale-space representation. Lindeberg's approach described above applies the concept of characteristic scale, and reduces the mismatched points and the storage space.

Mikolajczyk and Schmid evaluated the Lindeberg's approach for image matching by applying different derivative functions including Square gradient (Equation 2.11), Different-of-Gaussian (DoG) (Equation 2.12), Scale-space Harris function (Equation 2.10) and Laplacian function (Equation 2.13). They found that the scale-space Harris point detector was the best derivative function for the point detection and Laplacian function was the best derivative function for the point selection step.

However, Lindeberg's approach requires 3D space search (x,y, and n) and requires huge storage for the 3D search space. So, Mikolajczyk and Schmid proposed the Harris-Laplacian point detector by first applying scale-space Harris point detector and then selecting the scale using Laplacian function. The following steps are the algorithm of the Harris-Laplacian detector.

Step 1: (Point Detection) At each level of the scale-space s_n, detect the candidate points using Dufournaud's scale-space Harris point detector of scale s_n.

Step 2: (Point Selection) For each candidate point (x, y) detected at each level of the scale-space s_n, use Laplacian function $F_{laplacian}(x, y, s_n)$ for the selection of the candidate point. If $F_{laplacian}(x, y, s_n)$ is greater than $F_{laplacian}(x, y, s_{n-1})$ and $F_{laplacian}(x, y, s_{n+1})$, and $F_{laplacian}(x, y, s_n)$ is

greater than a threshold (t), the feature point (x, y) is selected.

2.1.3 Other Kinds of Interest Point Detectors

This section briefly introduces the other kinds of interest point detectors including a blob-like feature detector and a corner detector for color images.

Blob Detector

A Gaussian blob is defined by

$$h(x, y, t_0) = \frac{1}{2\pi t_0} \exp^{-(x^2+y^2)/2t_0}, \tag{2.14}$$

where t_0 is the width of the Gaussian blob.

The Laplacian operation (Equation 2.13) is usually used for blob-like feature detection because it gives a strong response at the center of blob-like structure [44, 5, 35]. Lindeberg [34, 35] proposed a method that detects blob-like features in the scale-space framework. The principle of the method is to apply the framework of Lindeberg's point detector (Section 2.1.2) and set the derivative function used in the framework to the scale-normalized Laplacian operator (Equation 2.13). Lindeberg proved that the characteristic scale for each detected Gaussian blob directly reflects the width t_0 of that blob.

Corner detector for Color Images

Montesinos *et al.* [52, 51] proposed a modification of Harris corner detector for color images. A color image is usually composed by three color channels: Red($\mathcal{R}$), Green($\mathcal{G}$) and Blue($\mathcal{B}$). Based on the framework of Harris corner detector, Montesinos' method applies Gaussian function and

Gaussian derivatives to the three channels and modifies the auto-correlation matrix (Equation 2.6) in the framework of Harris corner detector. The auto-correlation matrix is modified as in Equation 2.15.

$$M_{color}(x, y, \sigma_{\mathcal{D}}, \sigma_{\mathcal{I}}) = G(x, y, \sigma_{\mathcal{I}}) \star \begin{bmatrix} \mathcal{R}_x^2 + \mathcal{G}_x^2 + \mathcal{B}_x^2 & \mathcal{R}_x\mathcal{R}_y + \mathcal{G}_x\mathcal{G}_y + \mathcal{B}_x\mathcal{B}_y \\ \mathcal{R}_x\mathcal{R}_y + \mathcal{G}_x\mathcal{G}_y + \mathcal{B}_x\mathcal{B}_y & \mathcal{R}_y^2 + \mathcal{G}_y^2 + \mathcal{B}_y^2 \end{bmatrix} \tag{2.15}$$

where $\mathcal{I}_k$ is the short form of $\mathcal{I}_k(x, y, \sigma_{\mathcal{D}})$, the first-order Gaussian derivative of $\mathcal{I}$ of standard derivative $\sigma_{\mathcal{D}}$ with respect to direction k at image position (x, y), for $\mathcal{I} = \{\mathcal{R}, \mathcal{G}, \mathcal{B}\}$ and $k = \{x, y\}$.

Montesinos' corner detector for color images described above can be easily upgraded to scale-invariant by multiplying the Gaussian derivatives for every channel in Equation 2.15 with the scale $\sigma_{\mathcal{D}}$, so that the Gaussian derivatives are scale-normalized similar to Dufournaud's scale-space Harris point detector (Section 2.1.2).

2.1.4 Summary

Scale-space point detectors are commonly used for the matching for multi-resolution images. They are robust and effective if the scale difference of the two images is not very large and the affine change between the two images is small, that is, when the viewing angles of the two images are similar.

If the view angles of the two images are different, the affine-invariant point detectors [36, 48, 2] can be used. But iterative approaches are required for the detection of affine-invariant points, and hence the computation of the affine-invariant point detection is more expensive than the scale-space point detection.

Scale-space point detectors are good for the image matching. However, because of the operation of the Gaussian blur, the number of points extracted is usually smaller while the scale is larger. Therefore, for the application of structure from motion, we would apply the point detector in the smallest scale to extract more feature points.

2.2 Steerable filters

To find the responses of a filter at many directions, we are required to apply many versions of the same filter. Freeman and Adelson [72] proposed an approach that applies a few filters corresponding to a few angles and interpolates between their responses to approximate a response in any direction.

A steerable filter is a filter which can be synthesized as a linear combination of a set of basis filters to give a response in any direction. Gaussian directional derivatives are steerable as stated in [72] and suitable in the scale-space framework. Let G be Gaussian function, the Gaussian n^{th} derivative with respect to the direction θ can be formulated as:

$$G_{\theta^n} = (\cos\theta\partial_x + \sin\theta\partial_y)^n G \tag{2.16}$$

where θ is the direction of the desired derivative measured counter-clockwise from the x-axis, ∂_k is the partial derivative operator with respect to direction k, and $\partial_x G$ or $\partial_y G$ is the Gaussian derivatives along x- or y-direction respectively.

From Equation 2.16, the first (G_θ) and the second order (G_{θ^2}) Gaussian

derivatives with respect to the direction θ are formulated as follows:

$$\begin{aligned} G_\theta &= \cos\theta G_x + \sin\theta G_y \\ G_{\theta^2} &= \cos^2\theta G_{xx} + 2\cos\theta\sin\theta G_{xy} + \sin^2\theta G_{yy} \end{aligned}$$

2.2.1 Orientation estimation

Steerable filters are very useful in the estimation of point orientation. The orientation of a point is the same direction to the image gradient of that point. From the interpolation of the basis filters, we can relate the world coordinates with the local coordinates of a point by the angle of the orientation θ of that point. Then, we can estimate the orientation of that point using the values of the responses of the basis filters. More specifically, there are two methods to estimate the point orientation.

By an image gradient Let local coordinates $\mathbf{v}$ be the direction of the image gradient about a point, and $\mathbf{u}$ be the direction orthogonal to the image gradient. From the measured first order Gaussian derivatives along x and y axes, L_x and L_y respectively (as defined in Equation 2.5), the direction vectors of $\mathbf{v}$ and $\mathbf{u}$ are

$$\mathbf{v} = \frac{1}{\|\nabla L\|}\begin{pmatrix} L_x \\ L_y \end{pmatrix} = \begin{pmatrix} \cos\theta \\ \sin\theta \end{pmatrix} \quad \text{and} \tag{2.17}$$

$$\mathbf{u} = \frac{1}{\|\nabla L\|}\begin{pmatrix} L_y \\ -L_x \end{pmatrix} = \begin{pmatrix} \sin\theta \\ -\cos\theta \end{pmatrix}, \tag{2.18}$$

where $\|\nabla L\| = \sqrt{L_x^2 + L_y^2}$ and θ is the angle of the direction counter-clockwise from x-axis.

When the image gradient is orthogonal to the local direction of a point, the magnitude of the image gradient is zero. On the other hand, when the image gradient is in the local direction of that point, the magnitude of the image gradient is $\|\nabla L\|$. That means, L_u is zero and $L_v = \sqrt{L_x^2 + L_y^2}$. Therefore, we can find θ simply by $\theta = \arctan(L_y, L_x)$, where θ represents the point orientation.

By the eigenvectors of a Hessian matrix From [35], we can align a local (p, q)-coordinate system to the eigenvectors of the Hessian matrix in order to find the orientation. This means that we can rotate the coordinate system by an angle θ so that L_{pq} is zero.

$$\begin{aligned} L_{pq}(\theta) &= \partial_p \partial_q L \\ &= (\cos\theta\partial_x + \sin\theta\partial_y)(\sin\theta\partial_x - \cos\theta\partial_y)L \\ &= -L_{xy}\cos^2\theta + (L_{xx} - L_{yy})\cos\theta\sin\theta + L_{xy}\sin^2\theta \\ &= 0 \end{aligned} \tag{2.19}$$

By solving Equation 2.19, we can get $\cos\theta$ and $\sin\theta$, and hence the orientation θ.

$$\cos\theta = \sqrt{\frac{1}{2}(1+w)}, \tag{2.20}$$

$$\sin\theta = (\text{sign}L_{xy})\sqrt{\frac{1}{2}(1-w)}. \tag{2.21}$$

where

$$w = \frac{L_{xx} - L_{yy}}{\sqrt{(L_{xx} - L_{yy})^2 + 4L_{xy}^2}}$$

By estimating the orientation of a point, we can rotate the window patch of that point to the estimated orientation. Then we can cause the point descriptors invariant to orientation of that point because its orientation is normalized.

2.3 Point Descriptors

The correspondence problem on a pair of images refers to finding the relationship between the images. A point correspondence means a point on the first image is related to a point on the other image, and both points are the same image point of a 3D feature.

Point descriptors are very important for the matching of a pair of interest points on two images. A similarity measurement has been performed on the point descriptors to check whether the matching is valid or not. In the case of a small transformation between the images, a point correspondence is usually created by first detecting an interest point on each image using an interest point detector, e.g. Harris point detector. Then, the nearest interest point on the second image to that on the first image is mutually compared. If they are similar, the pair of interest points are linked together and form a point correspondence. In earlier methods, the comparison of the interest points is done by cropping a patch around the interest point on each image as the descriptor of the interest point, and then comparing the patch using a similarity measurement. The common similarity measurement is normalized cross-correlation (NCC) based on the intensity of the patch [60]. This approach is very successful for many 3D computer vision applications and has been used for many years.

However, NCC performs poorly for point matching in the situations of

large scale (or resolution) change, and large illumination change [49]. Moreover, the above method can only be applied if the transformation between the two images is small. Therefore, the above method cannot be applied to the matching of images with different scales or resolutions. Recently, Schmid and Mohr [62], Lowe [40], Dufornaud *et al.* [12, 13], Mikolajczyk and Schmid [47] introduced local image descriptors based on image derivatives for image matching, indexing in image databases and object recognition. For the application of the matching of images of different scales, the major idea is to detect interest points at each scale-space level. Then, the interest points are represented using scale-invariant point descriptors. For each interest point on an image, the most similar interest point on the other image is found using similarity measurements on the point descriptors.

This section first studies the effect of image derivatives under illumination change and geometric scale change. Then, we briefly describe the point descriptors used in [47].

2.3.1 Image derivatives under illumination change

Let $I^{(1)}$ and $I^{(2)}$ be two images related by an affine illumination change in Equation 2.22. c and m are the contrast change and the mean illumination change (or the brightness change) respectively.

$$I^{(1)} = cI^{(2)} + m \tag{2.22}$$

It is obvious to observe that the derivatives of images are invariant to

the mean illumination change.

$$\begin{aligned} L_x^{(1)} &= cL_x^{(2)} \\ L_{x^2}^{(1)} &= cL_{x^2}^{(2)} \\ &\vdots \end{aligned}$$

However, the derivatives are not invariant to the contrast change. To make the derivatives invariant to the contrast change, we can normalize the second or higher order derivatives by the magnitude of the first derivatives:

$$\begin{aligned} \frac{L_{x^2}^{(1)}}{\|\nabla L^{(1)}\|} &= \frac{cL_{x^2}^{(2)}}{c\|\nabla L^{(2)}\|} \\ \frac{L_{x^3}^{(1)}}{\|\nabla L^{(1)}\|} &= \frac{cL_{x^3}^{(2)}}{c\|\nabla L^{(2)}\|} \\ &\vdots \end{aligned}$$

where the first derivatives ∇L is

$$\nabla L = \begin{pmatrix} L_x \\ L_y \end{pmatrix} \quad \text{and} \quad \|\nabla L\| = \sqrt{L_x^2 + L_y^2}.$$

2.3.2 Image derivatives under geometric scale change

Let $\mathbf{x} = (x, y)^T$ be the coordinates on an image I. Suppose two images, $I^{(1)}$ and $I^{(2)}$, are related by

$$I^{(1)}(\mathbf{x}^{(1)}) = I^{(2)}(\mathbf{x}^{(2)})$$

under the geometric scale transformation of scale different s,

$$\mathbf{x}^{(1)} = s\mathbf{x}^{(2)}.$$

Section 2.1.2 already shows that an m-th order scale-normalized derivative in Equation 2.4 is invariant to the geometric scale change.

2.3.3 An example of a point descriptor

Previous sections show that the scale-normalized image derivatives are invariant to the scale change and the mean illumination change. If the second or higher order derivatives are divided by the magnitude of the first derivative, they become invariant to the affine illumination change. Therefore, image derivatives were commonly used as the descriptor of a point.

Mikolajczyk and Schmid [47] designed a point descriptor based on the Gaussian derivatives of a patch. They first separate a half circle by a certain number of parts. Then, the first Gaussian derivatives in each direction are computed. They use the angle of the maximum derivative as the orientation of the point and steer the derivatives to the orientation of the point. They compute the Gaussian derivatives up to the 4th order to obtain the descriptor. Finally, to make the descriptor invariant to the affine intensity changes, they divide the derivatives by the steered first derivative.

2.3.4 Other examples

Other examples of the point descriptors are the Scale Invariant Feature Transform (SIFT) [40] and the differential invariants [62]. Both use Gaussian derivatives. Montesinos *et al.* [52, 51] proposed a descriptor for color

images using Hilbert's invariants, which involve only Gaussian derivatives up to the first order. The original Montesinos' descriptor is not scale-invariant, but updating it to scale-invariant is an easy work by normalizing the Gaussian derivatives with the scale.

Another kind of descriptor is the descriptor involving affine-invariant Gaussian derivatives [36, 48, 2]. But the computation of the affine-invariant descriptor is more costly, and the affine-invariant is not necessary for the multi-resolution problem, because it is assumed that the affine change of the texture is small and can be neglected.

2.4 Feature Tracking Techniques

An important step of structure from motion is to extract a set of interest points or feature points from the images, and relate the images by tracking the feature points. However, a lot of feature points are of poor quality or mismatched. They weaken the results of camera self-calibration, camera pose estimation and 3D reconstruction. Therefore, selecting good feature points and discarding outliers are important in feature tracking process.

This section introduces the Kanade-Lucas-Tomasi (KLT) tracker, a famous feature point tracker, selecting good feature points by the *texturedness* of the feature window, and discards bad feature points by *dissimilarity*. This section also introduces the guided tracking technique, which discards feature points using the geometric criteria after the tracking of feature points.

2.4.1 Kanade-Lucas-Tomasi (KLT) Tracker

The KLT tracker was initially proposed by Kanade and Lucas [41], further developed by Tomasi and Kanade [73], and fully described in [63]. The aim

of the KLT tracker is to track an individual point between a pair of images.

Given two images I_1 and I_2, and a feature window centered at image coordinates $\mathbf{x} = (x_1, x_2)^T$ in image I_1, the aim of feature tracking is to determining the 2×2 deformation matrix D and the 2×1 displacement vector $\mathbf{d}$ such that

$$I_2(A\mathbf{x} + \mathbf{d}) = I_1(\mathbf{x}) \tag{2.23}$$

where $A = \mathbf{1} + D$, $\mathbf{1}$ is a 2×2 identity matrix.

Because of image noise, Equation 2.23 cannot be exactly satisfied in general. Therefore, we have to determine the motion parameters D and $\mathbf{d}$ such that the dissimilarity is the minimum. The dissimilarity is defined as follow:

$$\epsilon = \int\int_W [I_2(A\mathbf{x} + \mathbf{d}) - I_1(\mathbf{x})]^2 w(\mathbf{x}) d\mathbf{x} \tag{2.24}$$

where W is the given feature window and $w(\mathbf{x})$ is a weighting function.

The main concern of [63] is to select good features during tracking. It proposes two selection criteria: texturedness and dissimilarity.

Texturedness Let us consider the 2×2 gradient matrix of the feature window:

$$Z = \begin{bmatrix} g_x^2 & g_x g_y \\ g_x g_y & g_y^2 \end{bmatrix} \tag{2.25}$$

where g_x and g_y are the gradient of the intensity along x and y direction respectively. According to [63], If both of the eigenvalues of Z are small, the intensity profile within the feature window is roughly constant. A large and a small eigenvalues mean an unidirectional texture pattern. Both large eigenvalues can represent corners, salt-and-pepper textures, or any other patterns that can be tracked reliably. Therefore, to choose reliable corners,

we only accept the feature points which Z's have two large eigenvalues. That is,

$$t = \min(\lambda_1, \lambda_2) > \lambda \tag{2.26}$$

where λ_1 and λ_2 are eigenvalues of Z, λ is a predefined threshold.

The texturedness t defined in Equation 2.26 can also be considered as the rank of feature points when the limited number of feature points are required.

Dissimilarity A feature with high texture content can still be a bad feature to track. The dissimilarity defined in Equation 2.24 is added to indicate the quality of the feature and select a good feature. If the dissimilarity of a feature point becomes large relative to the other feature points during tracking along the image sequence, the feature point is discarded or replaced by another feature point.

Originally, the KLT tracker is used for stereo images. It is now usually extended to track feature points along an image sequence by continuing the point selecting and discarding processes along the image sequence.

The KLT tracker is robust and accurate to track the feature points if the motion of the image sequence is small enough. If the motion is large, the KLT tracker may cause a lot of outliers and mismatched feature points due to the limited size of the feature window. A larger feature window maybe useful, however, it would require much more time to execute.

2.4.2 Guided Tracking Algorithm

Outlying feature points reduce the accuracy of motion estimation and so is the accuracy of the 3D shape reconstructed. Reducing the number of

the outliers is useful for motion recovery and 3D shape reconstruction. The main idea of the guided tracking algorithm is to apply the geometric criteria to select suitable feature points for motion recovery and 3D shape reconstruction.

Gibson el al. [66] introduces a guided tracking algorithm that employs an estimate of inter-frame camera motion to guide the feature tracking ([64] also discusses a similar approach briefly). The main idea of the algorithm is that the algorithm first tracks the feature points through the image sequence using the KLT tracker, and estimates the *fundamental matrix* [78] between the first image and the last image of the sequence. Then, the tracking algorithm removes each feature that moves a significant distance from its corresponding *epipolar line* (Section 2.6.2) during the tracking process. RANSAC [16] (Section 2.5) is usually used to remove the outlying features in this case. In [66], an experiment showed that higher percentage of inliers was tracked by the guided tracking algorithm compared with the standard KLT tracker.

2.5 RANSAC

RANSAC (RANdom SAmple Consensus) [16] is a model fitting paradigm that is often used in 3D computer vision. The principle of RANSAC is to repeatedly and randomly draw samples from the data set of the possible combinations, and compute the relation from the samples. RANSAC chooses and outputs the relation as the best relation if the relation contains the maximum number of inliers that fulfill certain constraints among the trials. The relation can be the fundamental matrix or the 2D homography [28]. Moreover, the expected number of trails required, N, is determined

by Equation 2.27:

$$N = \log(1-p)/\log(1-(1-e)^n) \tag{2.27}$$

where n is the number of data points required in the error-free set, p is the probability that at least one of the random selections is an error-free set, and e is the probability that a selected data point is outside the error tolerance of the model.

RANSAC can be used to remove the outlying features using the fundamental matrix as described in Section 2.4.2. Let $\mathbf{x} = (u, v, 1)^T$ and $\mathbf{x}' = (u', v', 1)^T$ be a pair of 2D feature points in the first and the last image frames that are supposed to be corresponding each other, respectively. Let F be the fundamental matrix, Equation 2.28 shows the vector for the epipolar line, $\mathbf{l}$, on the last frame corresponding to the point $\mathbf{x}$ on the first frame, and Equation 2.29 is the distance d of a point $\mathbf{x}'$ on the last frame to the epipolar line.

$$\mathbf{l} = \frac{F\mathbf{x}}{\|F\mathbf{x}\|} \tag{2.28}$$

$$d = |\mathbf{l} \cdot \mathbf{x}'| \tag{2.29}$$

Algorithm 2.1 shows the RANSAC algorithm using the fundamental matrix. Let t be a threshold for the error tolerance. If the normalized eight point algorithm [26] described in Section 2.6.2 is used to estimate the fundamental matrix, n is 8. The point correspondence, $\mathbf{x}$ and $\mathbf{x}'$, is an inlier if the pair fulfills the epipolar constraint that the corresponding fundamental matrix contains the maximum number of inliers among the trials. The RANSAC algorithm using the fundamental matrix requires a

reliable fundamental matrix [75], so the difference between the view angles of the cameras should be large enough.

Algorithm 2.1 RANSAC using Fundamental Matrix

1: Initialize the expected number of trails N using Equation 2.27.
2: Initialize $i \leftarrow 0$.
3: **repeat**
4: Randomly select 8 points from the data set, $\mathcal{D}$. Calculate the fundamental matrix F.
5: Set the set of inliers $\mathcal{I} \leftarrow \emptyset$.
6: **for all** point correspondences ($\mathbf{x}$ and $\mathbf{x}'$) from the data set **do**
7: Find the epipolar lines, $\mathbf{l}_1$ and $\mathbf{l}_2$, of $\mathbf{x}$ and $\mathbf{x}'$ on the last frame and the first frame using Equation 2.28, respectively.
8: Calculate the distances, d_1 and d_2, of $\mathbf{x}$ and $\mathbf{x}'$ to $\mathbf{l}_2$ and $\mathbf{l}_1$ using Equation 2.29, respectively.
9: **if** $d_1 < t$ and $d_2 < t$ **then**
10: $\mathcal{I} \leftarrow \mathcal{I} \cup (\mathbf{x}, \mathbf{x}')$
11: **end if**
12: **end for**
13: $i \leftarrow i + 1$
14: Update $e \leftarrow (1 - |\mathcal{I}|)/|\mathcal{D}|$.
15: Update N by Equation 2.27.
16: **until** $i \geq N$

2.6 Structure-from-motion (SFM) Algorithm

Structure from motion (SFM) refers to the problem of estimating the three-dimensional information about the environment from the motion of two-dimensional projection onto a surface [68]. It is an important task for a lot of applications including 3D model reconstruction, 3D motion matching, camera calibration, 3D coding of image sequences, navigation, 3D scene structure recovering and 3D videos and movies.

3D shape reconstruction is generally formulated as follows. Suppose an

object is placed at a certain distance from the camera. The object rotates and translates relative to the camera, and the camera takes a number of images of the object. Suppose we take a sequence of F images of the object, and then extract the feature points from the images and obtain the relation of the feature points along the image sequence using a feature point tracker. We then track N feature points $q_{ij} = (u_j, v_j)_i$ in the i-th image, $1 \leq i \leq F$, $1 \leq j \leq N$, the feature points q_{ij} in all images $1 \leq i \leq F$ correspond to the 3D point $\mathbf{X}_j = (X_j, Y_j, Z_j)$, $1 \leq j \leq N$. 3D shape reconstruction is the problem of estimating the 3D coordinates $\mathbf{X}_j = (X_j, Y_j, Z_j)$ of the feature points from the feature points q_{ij}. At the same time, we usually also recover the motions including the rotation and the translation of the object among all image views. Figure 2.1 shows the major idea of structure from motion.

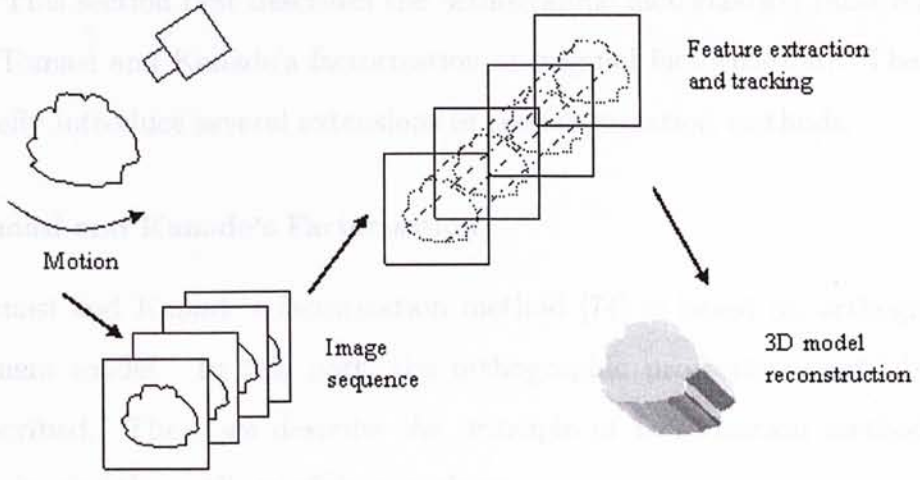


Figure 2.1: Overview of Structure from Motion

Generally, the process of 3D shape reconstruction consists of camera positioning, feature point extraction and tracking, 3D model reconstruction, and texture mapping. 3D model reconstruction is the major step, and has been studied in the literature using different assumptions. Famous

algorithms include factorization methods for image sequences [74, 55, 53, 19, 33, 43, 24, 7, 76, 1], methods for the epipolar geometry [78, 26, 37], and bundle adjustment [77, 64, 9].

In the remaining of this section, we describe briefly the factorization methods, the techniques for stereo, and the techniques of bundle adjustment.

2.6.1 Factorization methods

Factorization method is a mathematical method that recovers the shape and the motion of an rigid object from an image sequence of that object. Factorization method for shape and motion recovery was proposed by Tomasi and Kanade [74], which is based on orthographic projection model.

This section first describes the orthographic factorization (also refer to as Tomasi and Kanade's factorization or original factorization). Then, we briefly introduce several extensions of the factorization methods.

Tomasi and Kanade's Factorization

Tomasi and Kanade's factorization method [74] is based on orthographic camera model. In this part, the orthographic projection model is first described. Then, we describe the principle of factorization method and summarize the outlines of the algorithm.

Orthographic Projection Figure 2.2 illustrates the orthographic projection model. Suppose the orthonormal unit vectors $\mathbf{i}_f$, $\mathbf{j}_f$ and $\mathbf{k}_f$ represent the image frame f, where $\mathbf{i}_f$ and $\mathbf{j}_f$ correspond to the x and y axes of the image plane of the camera respectively, and $\mathbf{k}_f$, where $\mathbf{k}_f = \mathbf{i}_f \times \mathbf{j}_f$, represents the direction along the optical axis of the camera plane. A feature point p that

we are tracking is located at position $\mathbf{s}_p$ in the fixed world origin. The vector $\mathbf{t}_f$ indicates the position of the focal point of the camera corresponding to the world origin. The imaging rays are projected from the feature point p along the direction parallel to the optical axis of the camera plane. This point p will be observed at image coordinates (u_{fp}, v_{fp}) in the camera frame f. Therefore, (u_{fp}, v_{fp}) is the projection of $(\mathbf{s}_p - \mathbf{t}_p)$ onto the camera plane, such that

$$u_{fp} = \mathbf{i}_f \cdot (\mathbf{s}_p - \mathbf{t}_f) \quad \text{and} \quad v_{fp} = \mathbf{j}_f \cdot (\mathbf{s}_p - \mathbf{t}_f). \tag{2.30}$$

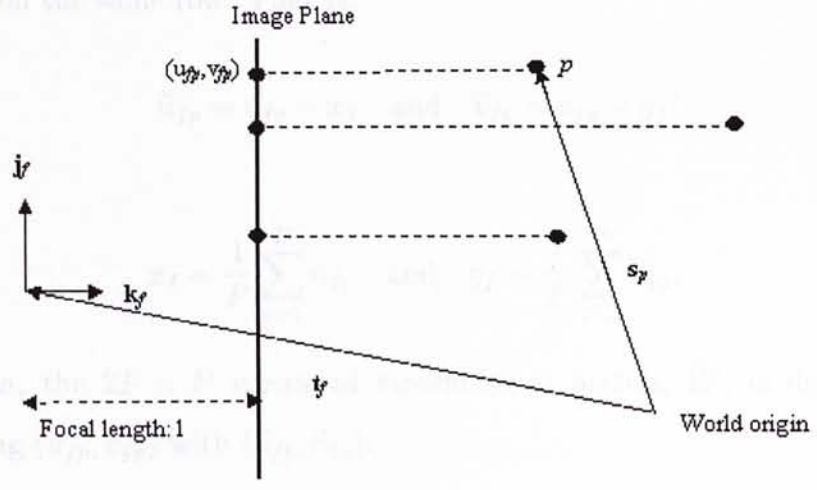


Figure 2.2: Orthographic projection.

Factorization Algorithm Suppose we have tracked P feature points over F frames in an image stream. We get the coordinates of the feature points (u_{fp}, v_{fp}) where $f = 1, ..., F$ and $p = 1, ..., P$. We can think of the u_{fp} and v_{fp} as the entries of two $F \times P$ matrices, X and Y respectively. Then, the $2F \times P$ *measurement matrix*, W, can be formed using X and Y.

$$W = \begin{bmatrix} X \\ Y \end{bmatrix} = \begin{bmatrix} u_{11} & \dots & u_{1P} \\ \vdots & \ddots & \vdots \\ u_{F1} & \dots & u_{FP} \\ v_{11} & \dots & v_{1P} \\ \vdots & \ddots & \vdots \\ v_{F1} & \dots & v_{FP} \end{bmatrix} \tag{2.31}$$

The rows of the matrix W are then subtracted by the means of the entries on the same row. That is,

$$\widetilde{u}_{fp} = u_{fp} - x_f \quad \text{and} \quad \widetilde{v}_{fp} = v_{fp} - y_f \tag{2.32}$$

where

$$x_f = \frac{1}{P}\sum_{p=1}^{P} u_{fp} \quad \text{and} \quad y_f = \frac{1}{P}\sum_{p=1}^{P} v_{fp}.$$

Then, the $2F \times P$ *registered measurement matrix*, $\widetilde{W}$, is defined by replacing (u_{fp}, v_{fp}) with $(\widetilde{u}_{fp}, \widetilde{v}_{fp})$:

$$W = \begin{bmatrix} \widetilde{u}_{11} & \dots & \widetilde{u}_{1P} \\ \vdots & \ddots & \vdots \\ \widetilde{u}_{F1} & \dots & \widetilde{u}_{FP} \\ \widetilde{v}_{11} & \dots & \widetilde{v}_{1P} \\ \vdots & \ddots & \vdots \\ \widetilde{v}_{F1} & \dots & \widetilde{v}_{FP} \end{bmatrix} \tag{2.33}$$

Suppose that the column vector $\mathbf{s}_p$ represents the 3D coordinates of

the object point corresponding to the p-th feature point on the images with respect to the world origin. The orthogonal column vectors $\mathbf{i}_f$ and $\mathbf{j}_f$ represent the orientation of the world coordinate system of the x- and y-axes respectively, of f-th image frame. The target of Tomasi and Kanade's factorization method is to estimate the structure composed of $\mathbf{s}_p$ for $1 \leq p \leq P$, and the pose of the cameras $\mathbf{i}_f$ and $\mathbf{j}_f$ for $1 \leq f \leq F$.

Assume that the world origin is placed at the centroid of the object points, therefore,

$$\frac{1}{P}\sum_{p=1}^{P}\mathbf{s}_p = 0. \tag{2.34}$$

Then, the expressions for $\widetilde{u}_{fp}$ and $\widetilde{v}_{fp}$ defined in (2.32) can be rewritten as follow:

$$\begin{aligned} \widetilde{u}_{fp} &= u_{fp} - x_f & \widetilde{v}_{fp} &= v_{fp} - y_f \\ &= \mathbf{i}_f^T\left(\mathbf{s}_p - \tfrac{1}{P}\textstyle\sum_{q=1}^{P}\mathbf{s}_q\right) \quad \text{and} & &= \mathbf{j}_f^T\left(\mathbf{s}_p - \tfrac{1}{P}\textstyle\sum_{q=1}^{P}\mathbf{s}_q\right) \\ &= \mathbf{i}_f^T\mathbf{s}_p & &= \mathbf{j}_f^T\mathbf{s}_p \end{aligned} \tag{2.35}$$

Because of (2.35), the registered measurement matrix $\widetilde{W}$ can be expressed using the following equation:

$$\widetilde{W} = RS \tag{2.36}$$

where the $3 \times P$ *shape matrix*, S, and the $2F \times 3$ *rotation matrix*, R, are defined in Equation 2.37.

$$\begin{aligned} S &= [\mathbf{s}_1\ \mathbf{s}_2\ \ldots\ \mathbf{s}_P] \\ R &= [\mathbf{i}_1\ \ldots\ \mathbf{i}_f\ \mathbf{j}_1\ \ldots\ \mathbf{j}_f]^T. \end{aligned} \tag{2.37}$$

Assume that $2F \geq P$, $\widetilde{W}$ can be decomposed using the *singular-value decomposition* (*SVD*) into a $2F \times P$ matrix U, a diagonal $P \times P$ matrix D, and a $P \times P$ matrix V,

$$\widetilde{W} = UDV^T, \tag{2.38}$$

such that D is a diagonal matrix whose diagonal entries are the singular values $\sigma_1 \geq \cdots \geq \sigma_P$ sorted in non-increasing order.

Since $\mathbf{i}_f$, $\mathbf{j}_f$ and $\mathbf{k}_f$ are orthonormal, the rank of R is 3 for the number of feature points $P \geq 3$. Also, the rank of S is at most 3. Therefore, according to the rank theorem in [74], $\widetilde{W}$ is at most of rank 3 in the absence of noise. In the presence of noise, the rank of $\widetilde{W}$ is approximately 3.

The method of estimating the best rank-3 approximation to $\widetilde{W}$ is by SVD. By setting all but the three largest singular values in D to zero, we can define D' as the 3×3 top left submatrix of D corresponding to the three largest singular values, and U' and V' as the $2F \times 3$ and $P \times 3$ submatrices of U and V formed by the columns corresponding to the three largest singular values in D.

Two matrices $\hat{R}$ and $\hat{S}$ are defined such that $\widetilde{W} \approx \hat{R}\hat{S}$. The two matrices can be formed by U', V' and D' as follow:

$$\hat{R} = U'D'^{1/2} \quad \text{and} \quad \hat{S} = D'^{1/2}V'^T \tag{2.39}$$

The camera orientation vectors $\mathbf{i}_f$ and $\mathbf{j}_f$ of the matrix R are orthogonal, such that $\mathbf{i}_f \cdot \mathbf{i}_f = 1$, $\mathbf{j}_f \cdot \mathbf{j}_f = 1$, and $\mathbf{i}_f \cdot \mathbf{j}_f = 0$. However, according to [78](p. 207), $\hat{\mathbf{i}}_f$ and $\hat{\mathbf{j}}_f$ of the matrix $\hat{R}$ will not be orthogonal. An invertible 3×3

matrix Q is required so that

$$\begin{aligned} \hat{\mathbf{i}}_f^T Q Q^T \hat{\mathbf{i}}_f &= 1 \\ \hat{\mathbf{j}}_f^T Q Q^T \hat{\mathbf{j}}_f &= 1 \\ \hat{\mathbf{i}}_f^T Q Q^T \hat{\mathbf{j}}_f &= 0 \end{aligned} \tag{2.40}$$

With the help of Q, we can find R and S by

$$R = \hat{R}Q \qquad \text{and} \qquad S = Q^{-1}\hat{S}. \tag{2.41}$$

As a result, the structure of the object is estimated in the shape matrix S and the poses of the cameras $(\mathbf{i}_f, \mathbf{j}_f, \mathbf{k}_f)$ are also estimated where $\mathbf{k}_f = \mathbf{i}_f \times \mathbf{j}_f$. Algorithm 2.2 shows the outline of Tomasi and Kanade's Factorization.

Algorithm 2.2 Outline of the Factorization method

1: Compute the registered measurement matrix $\widetilde{W}$, defined in (2.33).
2: Compute the singular-value decomposition $\widetilde{W} \approx U'D'V'$.
3: Calculate $\widetilde{R} = U'D'^{1/2}$ and $\widetilde{S} = D'^{1/2}V'^T$, defined in (2.39).
4: Compute the matrix Q by imposing the metric constraints in (2.40).
5: Compute the rotation matrix R and the shape matrix S by the equation (2.41).
6: Align the first camera reference system with the world reference system by computing the products RG and G^TS, where the orthonormal matrix $G = [\mathbf{i}_1\mathbf{j}_1\mathbf{k}_1]$ rotates the first camera reference system into the identity matrix.

Extensions of Factorization

Factorization method was extended to adapt to more general camera models, such as weak perspective [80] and para-perspective [55]. Combined with Epipolar Geometry, as described in [70], factorization method was further extended to the perspective projection model, which is the idealized

mathematical camera model [50]. Mahamud and Hebert [43] also proposed an iterative factorization method which is based on perspective projection model.

On the other hand, the factorization method was extended to allow sequential inputs of images [53] or reconstruct the model recursively [19, 33]. These methods are based on orthogonal projection model, which is the same as the original Tomasi and Kanade's Factorization method.

The methods introduced above do not estimate the intrinsic parameters of the cameras. Han and Kanade [21] proposed a factorization method which is built on the top of Mahamud and Hebert's iterative perspective factorization method [43] to extract the extrinsic parameters as well as the intrinsic parameters of the cameras.

Moreover, Han and Kanade [22, 23, 24] further extended the factorization method to solve the dynamic scene problem. The dynamic scene problem refers to the estimation of the motion of cameras and the positions of multiple moving objects for a sequence of images. Han and Kanade proposed methods that estimate the motion of the cameras and the positions of the objects if the objects move in constant velocities.

Factorization method is a common technique for 3D reconstruction of a non-rigid object. Bregler *et al.* [7] proposed a factorization method for recovering a non-rigid object. Torresani *et al.* [76] further extended the idea to track the feature points and recover the 3D structure of a non-rigid object.

2.6.2 Epipolar Geometry

This section introduces the basic idea of the geometry of stereo, also referred to as the *epipolar geometry* [37, 26, 78]. Figure 2.3 shows the epipolar

geometry.

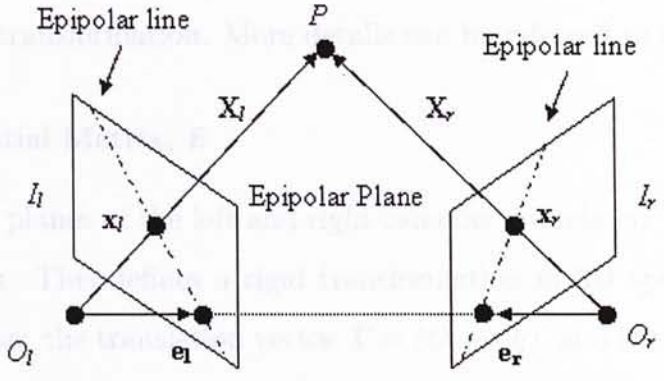


Figure 2.3: Epipolar Geometry

Figure 2.3 shows two pinhole perspective cameras with their centers of projection O_l and O_r, and image planes I_l and I_r respectively. $\mathbf{x}_l$ and $\mathbf{x}_r$ are the vectors from O_l and O_r to the projection of the 3D point P on I_l and I_r respectively. $\mathbf{X}_l$ and $\mathbf{X}_r$ are the vectors from O_l and O_r to the point P, respectively. The line joining O_l and O_r intersects on I_l and I_r at the *epipoles* $\mathbf{e}_l$ and $\mathbf{e}_r$ respectively. You may imagine that $\mathbf{e}_l$ is the projection of O_r on I_l; vice versa.

For each 3D point P, an *epipolar plane* passes through P and the two center of projections O_l and O_r. The epipolar plane intersects each image planes at the *epipolar line*. The projections of P must lie on the epipolar lines on I_l and I_r, this relation is called the *epipolar constraint*.

This section mainly introduces the basic components of the epipolar geometry including the *essential matrix* and the *fundamental matrix*. With these matrices, if we know both intrinsic and extrinsic parameters, we can reconstruct the 3D model by triangulation. If we only know the intrinsic parameters, we can still reconstruct the 3D model and estimate the extrinsic

parameters up to a scaling factor. If none of the intrinsic and extrinsic parameters are known, we can reconstruct the model only up to an unknown projective transformation. More details can be referred to in [78].

The Essential Matrix, *E*

The image planes of the left and right cameras are related via the extrinsic parameters. This defines a rigid transformation in 3D space between the two cameras: the translation vector $\mathbf{T} = (O_r - O_l)$, and the rotation matrix R. Therefore, the relation between $\mathbf{X}_l$ and $\mathbf{X}_r$ can be described as follow:

$$\mathbf{X}_r = R(X_l - \mathbf{T}) \tag{2.42}$$

From the epipolar plane, we can find the following relation:

$$(\mathbf{X}_r - \mathbf{T})^T \mathbf{T} \times \mathbf{X}_l = 0 \tag{2.43}$$

Using Equation 2.42, we obtain

$$(R^T \mathbf{X}_r)^T \mathbf{T} \times \mathbf{X}_l = 0 \tag{2.44}$$

Because we can rewrite $\mathbf{T} \times \mathbf{X}_l$ to $S\mathbf{X}_l$, where S is a rank 2 matrix:

$$S = \begin{bmatrix} 0 & -T_z & T_y \\ T_z & 0 & -T_x \\ -T_y & T_x & 0 \end{bmatrix} \tag{2.45}$$

Therefore, Equation 2.44 becomes

$$\mathbf{X}_r^T E \mathbf{X}_l = 0 \tag{2.46}$$

with the essential matrix $E = RS$. By using the fact that $\mathbf{x}_l = (\frac{f_l}{Z_l})\mathbf{X}_l$ and $\mathbf{x}_r = (\frac{f_r}{Z_r})\mathbf{X}_r$, where f_l and f_r are the focal length of the left and right cameras respectively, and $\mathbf{X}_l = [X_l, Y_l, Z_l]^T$ and $\mathbf{X}_r = [X_r, Y_r, Z_r]^T$, we can rewrite Equation 2.46 as

$$\mathbf{x}_r^T E \mathbf{x}_l = 0 \tag{2.47}$$

According to [78], the essential matrix provides a link between the epipolar constraint and the extrinsic parameters of the stereo system. Also, it is the mapping between points and epipolar lines with the prior information on the intrinsic parameters.

The Fundamental Matrix, *F*

We can obtain the mapping between points and epipolar lines without the prior information on the intrinsic parameters through the fundamental matrix.

Suppose $\bar{\mathbf{x}}_l$ and $\bar{\mathbf{x}}_r$ are the points in pixel coordinates corresponding to $\mathbf{x}_l$ and $\mathbf{x}_r$ in camera coordinates respectively such that

$$\mathbf{x}_l = M_l^{-1}\bar{\mathbf{x}}_l \quad \text{and} \quad \mathbf{x}_r = M_r^{-1}\bar{\mathbf{x}}_r \tag{2.48}$$

where M_l and M_r are the matrices of intrinsic parameters (described briefly in Appendix A.1) of the left and right cameras respectively. From Equation 2.47, we have

$$\mathbf{x}_r^T F \mathbf{x}_l = 0 \tag{2.49}$$

where

$$F = M_r^T E M_l^{-1}. \tag{2.50}$$

Therefore, according to [78], we can reconstruct the epipolar geometry with no information at all on the intrinsic or extrinsic parameters through the fundamental matrix.

The Eight-Point Algorithm

The eight-point algorithm [37] is a commonly used algorithm to compute the essential matrix and the fundamental matrix. To compute the essential matrix and the fundamental matrix, the eight-point algorithm requires at least 8 point correspondences between the pair of images. The principle of the eight-point algorithm is first to construct a homogeneous linear system from Equation 2.49, and then to compute the entries of the fundamental matrix. From Equation 2.50, we can also obtain the essential matrix. Algorithm 2.3 shows the basic structure of the eight-point algorithm.

According to [26], the estimation may be unstable. To avoid numerical instabilities, we have to normalize x_l and x_r. The details of the normalization method and the analysis on stability issues can be found in [26, 42, 78].

Estimation of Projective Matrix

Beardsley *et al.* [3] proposed a method to estimate the projective matrix for stereo images. The principle of the method is to first assume that the projective matrix with respect to the first camera (or left image) in the canonical form, and then compute the fundamental matrix and the epipole in the right image. Finally, it computes the projective matrix with respect to the right image by setting 4 parameters arbitrarily. Algorithm 2.4 shows

Algorithm 2.3 The Eight-Point Algorithm

1: Transform Equation 2.49 using n ($n \geq 8$) point correspondences to a linear system in the form $A\mathbf{f} = 0$.

2: Compute the singular value decomposition of A, $A = UDV^T$. The column vector $\mathbf{f}$ is the column of V corresponding to the least singular value of A.

3: Compute the singular value decomposition of F, $F = U_f D_f V_f^T$.

4: Set the smallest singular value in D_f to 0 and form D'_f. Recompute the corrected estimate of F, F' by

$$F' = U_f D'_f V_f^T$$

Algorithm 2.4 Estimation of Projective Matrix

1: Set the first projection matrix P_l to the canonical form:

$$P_l = [I|\mathbf{0}]$$

where I is a 3×3 identity matrix and $\mathbf{0}$ is a 3×1 zero vector.

2: Compute the fundamental matrix F.

3: Compute the epipole $\mathbf{e}_r$ in the second image using $F^T\mathbf{e}_r = 0$.

4: $M_r = e_r \times F$.

5: The projection matrix with respect to the second camera is computed by:

$$P_r = [M_r + \mathbf{e}_r\mathbf{b}^T|c\mathbf{e}_r]$$

where $\mathbf{b}$ and c are an arbitrary 3×1 vector and a scalar respectively.

the overview of the method.

Reconstruction of 3D Model

If we obtain the projection matrices for both images using the technique described in the preceding part, we can reconstruct the 3D model using the techniques of triangulation [27]. In Figure 2.3, imagine the projection matrices are fixed and known, we can obtain the 3D position of a point by intersecting the two lines emitted from the corresponding 2D points in both image frames. The triangulation problem is to estimate the intersection of these two lines in the space. Hartley and Sturm [27] proposed a polynomial method for the triangulation problem and studied several linear and iterative methods. The details of the triangulation techniques can be found in [27].

From Stereo to Sequence

The techniques for epipolar geometry are only suitable for stereo images. If the problem is extended to an image sequence, merging techniques are required. An image sequence can be divided into a lot of pairs of images. Therefore, the problem becomes the reconstructions of a lot of stereo images. However, each stereo is reconstructed individually, the structure reconstructed for each stereo is up to a projective transformation [78, 56]. To transform the structures in different projective transformations to be with respect to the same projective space, we have to estimate those projective transformations. Fitzgibbon and Zisserman [18] proposed a sequence merging technique that estimates the projective transformation between two structures.

Suppose that an image sequence consists of three image frames. Let I_i

be the i-th image for $i = \{1, 2, 3\}$. Suppose I_1 and I_2 form a stereo $S^{(1)}$, and similarly I_2 and I_3 form another stereo $S^{(2)}$. That is, I_2 becomes the common frame. Let $\mathbf{X}_j^{(1)}$ and $\mathbf{X}_j^{(2)}$, where $\mathbf{X}_j^{(1)} = (X_1^{(1)}, X_2^{(1)}, X_3^{(1)}, X_4^{(1)})^T$ and $\mathbf{X}_j^{(2)} = (X_1^{(2)}, X_2^{(2)}, X_3^{(2)}, X_4^{(2)})^T$, be the j-th 3D point with respect to the first cameras of $S^{(1)}$ and $S^{(2)}$ respectively, and $P^{(1)}$ and $P^{(2)}$ be the projection matrices on I_2 with respect to the first cameras of $S^{(1)}$ and $S^{(2)}$ respectively. Let $H^{(3D)}$ be the projective transformation from the first camera of $S^{(2)}$ to that of $S^{(1)}$ such that

$$\mathbf{X}_j^{(1)} = H^{(3D)}\mathbf{X}_j^{(2)} \tag{2.51}$$

$$P^{(1)} = P^{(2)}(H^{(3D)})^{-1} \tag{2.52}$$

The target of the merging techniques is to estimate the projective transformation $H^{(3D)}$, which minimizes $\sum D(\mathbf{X}_j^{(1)}, H^{(3D)}\mathbf{X}_j^{(2)})$ subject to the constraint $P^{(1)} = P^{(2)}(H^{(3D)})^{-1}$. The distance $D(\mathbf{X}, \mathbf{Y})$ can be Algebraic distance (Equation 2.53), Euclidean distance (Equation 2.54), or the reprojection error [18].

$$D_A(\mathbf{X}, \mathbf{Y}) = \sum_{k=1}^{3}(X_k Y_4 - Y_k X_4)^2 \tag{2.53}$$

$$D_E(\mathbf{X}, \mathbf{Y}) = \sum_{k=1}^{3}(\frac{X_k}{Y_4} - \frac{Y_k}{X_4})^2 \tag{2.54}$$

If the number of common correspondences between $S^{(1)}$ and $S^{(2)}$ is larger than 4, $H^{(3D)}$ can be computed by Fitzgibbon's merging method by Equation 2.55.

$$H^{(3D)} = (P^{(1)})^+ P^{(2)} + \mathbf{h}\mathbf{v}^T \tag{2.55}$$

where $\mathbf{h}$ and $(P^{(1)})^+$ are the null vector and the pseudo-inverse [32, 69] of

$P^{(1)}$ respectively. If Algebraic distance (Equation 2.53) is used, the 4×1 column vector $\mathbf{v}$ can be estimated by solving the following linear equation:

$$\mathbf{b}(\mathbf{X}_j^{(2)})^T\mathbf{v} = \mathbf{c} \tag{2.56}$$

where $\mathbf{b}$ and $\mathbf{c}$ are 3×1 column vectors that $b_k = X_4^{(1)}h_k - X_k^{(1)}h_4$ and $c_k = X_k^{(1)}a_4 - X_4^{(1)}a_k$, $\mathbf{a} = (P^{(1)})^+P^{(2)}X^{(2)}$.

According to [18], if Euclidean distance or the reprojection error is used for the distance, we have to solve Equation 2.55 using iterative optimization algorithms because the system becomes non-linear. Gibson *et al.* [66] proposed an approximate linear solution for Euclidean distance. More details can be found in [66].

2.6.3 Bundle Adjustment

The 3D model of an object (model) and the motion of the camera (pose) can be estimated using bundle adjustment. Using bundle adjustment to estimate the model and the pose, we have to define the error function and the parameters.

Optimization for Re-projection Error

Suppose we have F views of N 3D points, we wish to estimate the projection matrix of the camera P_i, $1 \leq i \leq F$, and the 3D points $\mathbf{X}_j$, $1 \leq j \leq N$, which are projected to the image points $\mathbf{x}_{ij}$ such that $\mathbf{x}_{ij} = P_i\mathbf{X}_j$. We estimate all P_i and all $\mathbf{X}_j$ such that the total re-projection error e of all 3D points on all views is the minimum [28]. That is,

$$e = \sum_{i=1}^{F}\sum_{j=1}^{N} d^2(P_i\mathbf{X}_j, \mathbf{x}_{ij}) \tag{2.57}$$

where $d^2(\mathbf{x}_r, \mathbf{x}_s)$ is the square of Euclidean distance between $\mathbf{x}_r$ and $\mathbf{x}_s$.

Equation 2.57 presents the general idea of bundle adjustment. The form of the re-projection error and the parameters adjusted can be modified dependent on the situations assumed. Generally, the projection matrix P_i, the 3D point $\mathbf{X}_j$, and the 2D point $\mathbf{x}_{ij}$ are defined below:

$$P_i = \begin{bmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ p_{31} & p_{32} & p_{33} & p_{34} \end{bmatrix}$$

$$\mathbf{X}_j = [X_1 \quad X_2 \quad X_3 \quad X_4]^T$$

$$\mathbf{x}_{ij} = [x_1 \quad x_2 \quad x_3]^T$$

If we assume that the intrinsic parameters are known, the problem can be reduced to Euclidean space. Hence, the projection matrix P_i, the 3D point $\mathbf{X}_j$, and the 2D point $\mathbf{x}_{ij}$ are redefined below:

$$P_i = M_{int} \begin{bmatrix} r_{11} & r_{12} & r_{13} & T_x \\ r_{21} & r_{22} & r_{23} & T_y \\ r_{31} & r_{32} & r_{33} & T_z \end{bmatrix}$$

$$\mathbf{X}_j = [X_1 \quad X_2 \quad X_3 \quad 1]^T$$

$$\mathbf{x}_{ij} = [x_1 \quad x_2 \quad 1]^T$$

where

$$R = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}$$

is the rotation matrix dependent on three rotation angles ϕ_x,ϕ_y,ϕ_z. M_{int} is the matrix contains the intrinsic parameters of the camera.

Lowe's pose estimation algorithm

Lowe's method [38, 39] assumes that we know the position of the 3D points corresponding to a view and the intrinsic parameters of that view, the problem can be reduced to estimating the pose (3 rotation angles ϕ_x,ϕ_y and ϕ_z, and a translation vector $\mathbf{T} = [T_x, T_y, T_z]^T$, totally 6 parameters). Because Lowe's method only need to find the pose of a view, only that view is required. Assume that the camera is calibrated, Lowe formulates the normal equations of the Levenberg-Marquardt minimization as follow:

$$\delta u_j = \sum_{k=x,y,z} \frac{\partial u_j}{\partial T_k} \Delta T_k + \frac{\partial u_j}{\partial \phi_k} \Delta \phi_k \tag{2.58}$$

$$\delta v_j = \sum_{k=x,y,z} \frac{\partial v_j}{\partial T_k} \Delta T_k + \frac{\partial v_j}{\partial \phi_k} \Delta \phi_k \tag{2.59}$$

where u and v is the x and y coordinates of j-th 3D projection in the image, respectively.

Lowe computes the re-projection errors of at least three 3D points along u and v direction of the image and the Jacobian matrix from an initial guess of $(\phi_x, \phi_y, \phi_z, T_x, T_y, T_z)$, and then uses standard least square methods to estimate $(\Delta\phi_x, \Delta\phi_y, \Delta\phi_z, \Delta T_x, \Delta T_y, \Delta T_z)$. Hence, improved estimates are given by $\phi'_k = \phi_k + \Delta\phi_k$ and $T'_k = T_k + \Delta T_k$ for $k = x, y, z$.

Two-pass Bundle Adjustment

Two-pass Bundle Adjustment [9] extends Lowe's pose estimation algorithm [38, 39] to estimate the optimal 3D structure of an object by giving a set of 2D feature point sequences along F frames. It assumes that the intrinsic parameters of that view are known.

Two-pass Bundle Adjustment, which is an interleaving bundle adjustment method, consists of two phases per iteration. The first phase is to estimate the poses from previous guess of 3D points by Lowe's method. The second phase is to estimate the 3D points from the poses estimated in first phase by Levenberg-Marquardt minimization [56]. Algorithm 2.5 shows the overview of Two-pass Bundle Adjustment.

Algorithm 2.5 Two-pass Bundle Adjustment

1: Initialize the guess of the 3D points $\{\mathbf{X}_j\}_1^N$ and the poses $\{\theta_i\}_1^F$
2: **repeat**
3: (First Phase) For each view i, using previous guess of $\{\mathbf{X}_j\}_1^N$ and the image features $\mathbf{x}_{ij}$ for all 3D points in i-th image, estimate the pose θ_i by Lowe's method.
4: (Second Phase) For each 3D point $\mathbf{X}_j$, using the guess of the poses $\{\theta_i\}_1^F$ and the j-th image feature of every image, estimate the better-predicted 3D point $\mathbf{X}_j$ using Levenberg-Marquardt minimization [56].
5: **until** the 2D total re-projection error is small enough or too many iterations are run.

2.6.4 Summary

Factorization methods and the triangulation methods only obtain the sub-optimal solution, while bundle adjustment can get the optimal solution. However, bundle adjustment is an iterative approach, hence it is much slower than factorization methods and the triangulation methods. In practice,

the suboptimal solution estimated by the factorization methods or the triangulation methods is usually used as the initial solution for the bundle adjustment [18, 55], and the bundle adjustment refines the solution to obtain the optimal solution.

□ **End of chapter.**

Chapter 3

Hierarchical Registration of 3D Models

This project aims to reconstruct a detailed 3D structure of a large scene from at least two image sequences. One for the large scene and the others are for the objects that are reconstructed in more details. The idea of our approach is first to take two image sequences, one is for the large scene and the other is for the object. Then, the image sequences are related by matching an image of each sequence. Using the point correspondences from the previous step, the relative poses of the 3D structures are estimated and the 3D structures are integrated to form the detailed 3D structure of the large scene.

Our approach is designed for the large scene that contains only rigid and opaque objects, and the scene is static such that the objects do not move. Also, the scene and the objects contain enough features (corner points) for 3D reconstruction. Moreover, in this thesis, the image sequence for the large scene is described as *low resolution* or *coarse*, and so are its images. On the other hand, the image sequence for the object in the scene

is described as *high resolution* or *fine*, and so are its images. In most cases, camera calibration is not necessary. If the radial distortion of the camera is significant, camera calibration may be required, and the intrinsic parameters estimated are given to the 3D reconstruction process.

In this chapter, we describe our approach for the detailed reconstruction for a large scene. Section 3.1 describes the overview of our method. Then, the remaining sections explain the details of our method.

3.1 Overview

In this section, we describe the arrangement of the image sequences and the flow of our framework.

3.1.1 The Arrangement of Image Sequences

In order to reconstruct both of the detailed objects as well as the large environment, we take separate image sequences for each detailed object and the large environment. For the large environment, each image includes the whole scene. For the fine object, the camera is placed nearer to the object, so the view of the camera can take the detailed contents of the object. For simplicity, in this thesis, we assume that there is one fine image sequence and one coarse image sequence. Figure 3.1 illustrates this arrangement. Let S_c be the coarse image sequence and S_f be the fine image sequence. Define $I_i^{(c)}$ and $I_j^{(f)}$ be the i-th and j-th image frames of the coarse and the fine image sequences respectively.

For the better matching between the image sequences, the user should select a pair of images (an image from each image sequence) which are taken from similar view angles. Let $I_p^{(c)}$ and $I_q^{(f)}$ be the selected images from S_c

and S_f respectively. Also, $I_p^{(c)}$ and $I_q^{(f)}$ should have an overlapping region which is large enough for the matching. Figure 3.2 shows an example of the coarse image sequence and fine image sequence. In this example, both of the image sequences consist of 19 frames.

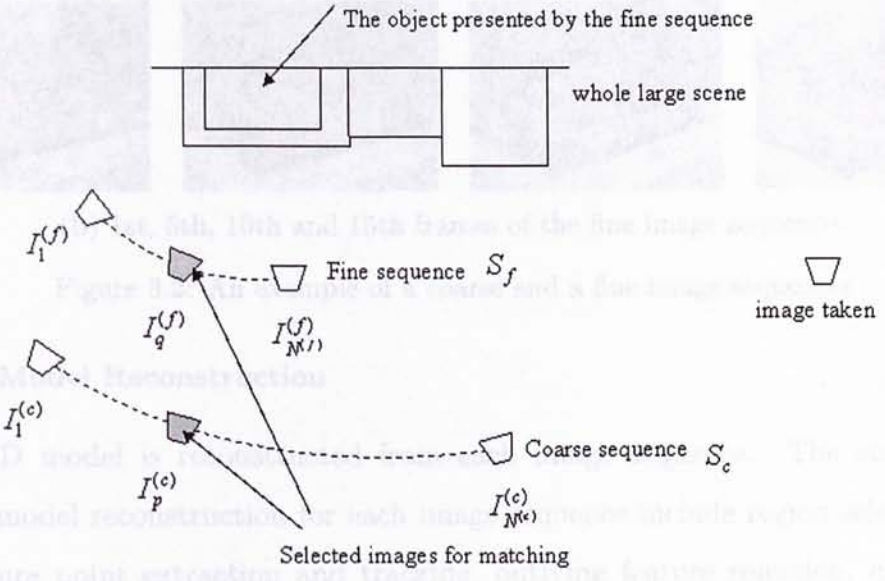


Figure 3.1: The arrangement of image sequences

3.1.2 The Framework

Figure 3.3 shows the flow of the framework. After we arrange the camera setup and take the images, the framework consists of four major steps: (1) reconstruct the 3D model for each image sequence, (2) match the selected pair of images using the multi-scale image matching technique, (3) build the linkage between the coarse and the fine sequence, (4) register the 3D models together, and (5) form the final VRML 3D model. These steps are described briefly in the remaining of this section, and the details are explained in the remaining of this chapter.

(a) 1st, 5th, 10th and 15th frames of the coarse image sequence.

(b) 1st, 5th, 10th and 15th frames of the fine image sequence.

Figure 3.2: An example of a coarse and a fine image sequences

3D Model Reconstruction

A 3D model is reconstructed from each image sequence. The steps of 3D model reconstruction for each image sequence include region selection, feature point extraction and tracking, outlying feature rejection, camera self-calibration and structure from motion.

For each image sequence, the feature points extracted for 3D reconstruction are of the smallest scale. But they are unsuitable for the multi-scale image matching, so we need the multi-scale image matching technique and the linkage establishment technique to link up the image sequences of different resolutions.

The details of the 3D model reconstruction are described in Section 3.2.

Multi-scale Image Matching

The selected pair of images, $I_p^{(c)}$ and $I_q^{(f)}$, are of different resolutions, but they are taken in similar view angles. Therefore, we can use the multi-scale image matching technique to find the relation, $H^{(2D)}_{(I_p^{(c)}, I_q^{(f)})}$, between these two

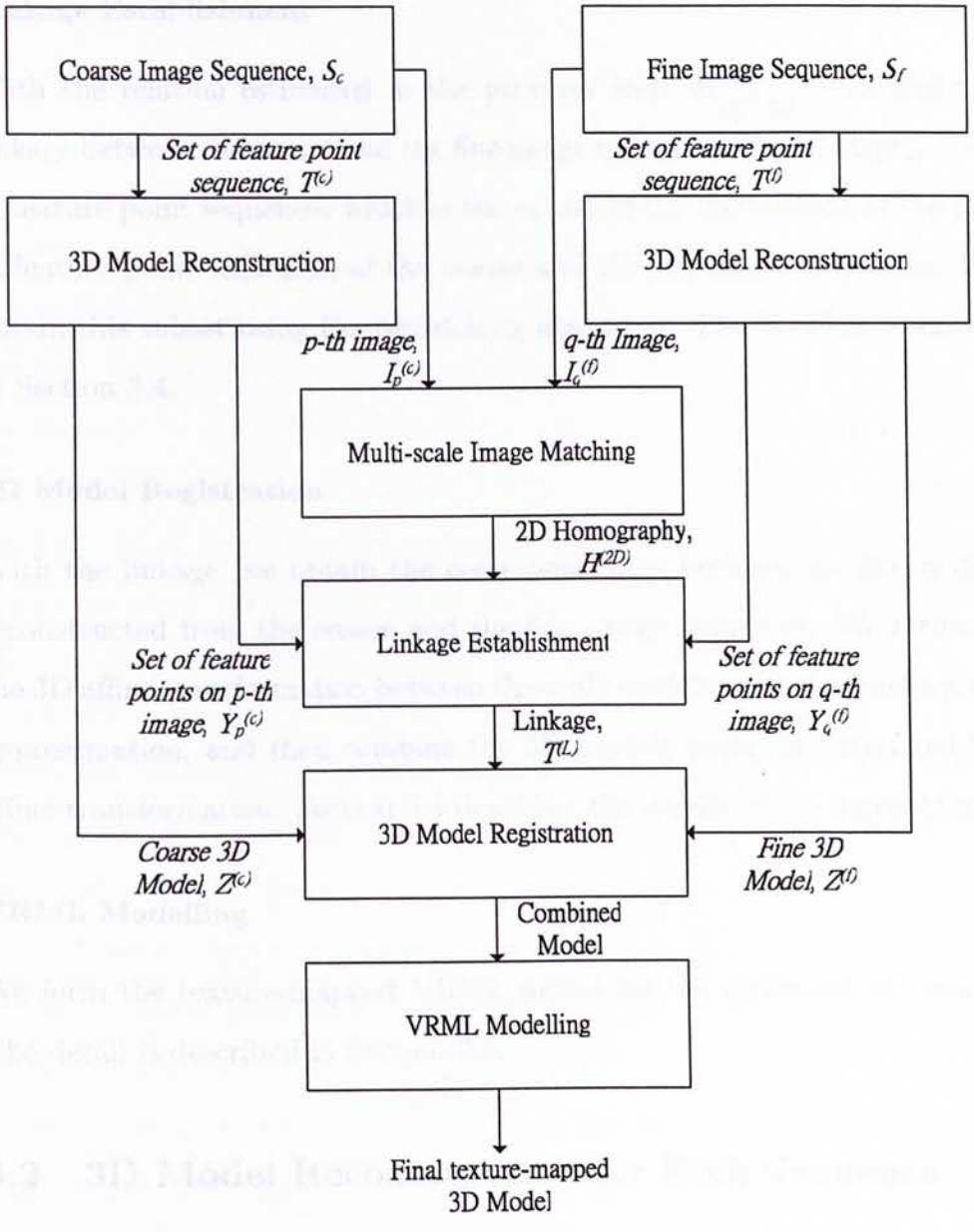


Figure 3.3: The multi-scale reconstruction framework

images. Section 3.3 describes the multi-scale image matching in details.

Linkage Establishment

With the relation estimated in the previous step, $H^{(2D)}_{(I_p^{(c)}, I_q^{(f)})}$, we find the linkage between the coarse and the fine image sequences. The linkage is a set of feature point sequences which is the subset of the intersection of the sets of feature point sequences of the coarse and the fine image sequences. We obtain this subset using the rematching algorithm. The detail is described in Section 3.4.

3D Model Registration

With the linkage, we obtain the correspondences between the 3D models reconstructed from the coarse and the fine image sequences. We estimate the 3D affine transformation between these 3D models using the least square approximation, and then combine the 3D models using the estimated 3D affine transformation. Section 3.5 describes the details of the registration.

VRML Modelling

We form the texture-mapped VRML model for the combined 3D model. The detail is described in Section 3.6.

3.2 3D Model Reconstruction for Each Sequence

Typically, 3D model reconstruction for an image sequence consists of five parts including region selection, feature extraction and tracking, rejection of outlying feature points, camera self-calibration, and 3D structure recovery.

Region Selection Since not the whole region of the image is preferred and useful for finding the 3D structure of the scene (or the object), a region is selected before the feature extraction. In our project, a rectangular region on the image is selected by the user. Only the points in the region are extracted in the next process.

Feature Extraction and Tracking To reconstruct the 3D structure of an object, we have to obtain the feature point sequences from the images. We use the standard Harris corner detector described in Section 2.1.2 to extract feature points from the images. However, not all feature points are useful for the tracking in the image sequence. The KLT tracker described in Section 2.4.1 is used in our project to track the feature points through the image sequence, and remove the poor feature points or the feature points that do not exist in the whole image sequence.

Outlying Feature Rejection The selection process of the KLT tracker only considers the textural criteria, but it is not enough to remove the mismatched feature points. The geometric criteria can be added to remove those mismatched feature points. The common technique of RANSAC [16] is used to estimate the accurate fundamental matrix and obtain the inlying feature points [75, 64].

Here, we assume that the difference in view angles between the first and the last frames is large enough to obtain a reliable fundamental matrix [75]. Thus, we estimate the fundamental matrix between the first and the last image frames as the relation, and the epipolar constraint [28] (or the distance from the epipolar line) becomes the constraint to determine the inliers. The details of the RANSAC algorithm using the fundamental matrix is described

in Section 2.5.

Camera Self-calibration We use Pollefeys' method [57] to find the focal length of each image frame. The self-calibration method only estimates the intrinsic parameters up to scale. That is, the focal lengths estimated are actually in ratio between the frames. We have to fix the focal length of the first image frame and compute the focal lengths of the other frames. In this project, we assume that we know the focal length of the first image frame of the coarse image sequence. But for the fine image sequence it can be fixed arbitrarily. The problem of the scale factor of the fine sequence can be solved by 3D model registration (Section 3.5).

3D Structure Recovery The two-pass bundle adjustment [9] is used to reconstruct the 3D structure for each image sequence. The two-pass bundle adjustment is faster than the standard bundle adjustment [77], and both algorithms can estimate the optimal 3D structure. Therefore, we use the two-pass bundle adjustment in our project. The algorithm of the two-pass bundle adjustment is presented in Section 2.6.

3.3 Multi-scale Image Matching

The most important problem of the coarse-to-fine 3D model merging is to obtain the correspondences between the coarse image sequence (S_c) and the fine image sequence (S_c). The coarse and the fine image sequences consist of sets of feature point sequences among their image frames, $T^{(c)}$ and $T^{(f)}$, respectively. Each set of feature point sequence is used to reconstruct the 3D model. The linkage problem between the two image sequences is to obtain a subset ($T^{(\mathrm{L})}$) of the intersection of the sets of point sequences of

both the coarse and the fine image sequences, such that $T^{(\mathrm{L})} \subseteq T^{(c)} \cap T^{(f)}$.

We obtain this subset using two steps. First, we estimate the 2D transformation between the selected pair of images ($I_p^{(c)}$ and $I_q^{(f)}$) using a multi-scale image matching technique. Then, with the estimated 2D transformation, we obtain this subset using the rematching algorithm. Compared with the approach of Ramalingam and Lodha [58] that obtains this subset by matching the point correspondences of the sets $T^{(c)}$ and $T^{(f)}$ directly using scale-sensitive correlation technique, our approach is more flexible and is not restricted to one multi-scale image matching algorithm, so that a lot of multi-scale image matching algorithms which can estimate the 2D transformation between the selected image pair can be used in our framework. Moreover, in general, as shown in [47] and claimed in [58], the multi-scale image matching techniques can match higher scale factor than the scale-sensitive correlation technique.

The multi-scale matching technique consists of four steps:

1. Scale-space interest point detection
2. Point descriptor construction
3. Point-to-point matching
4. Image transformation estimation

The remaining of this section describes the multi-scale image matching technique of our framework in details, and the extension of the scale-factor using the matching technique hierarchically. Section 3.4 describes how we use the rematching algorithm to obtain the linkage ($T^{(\mathrm{L})}$) between the sets $T^{(c)}$ and $T^{(f)}$.

3.3.1 Scale-space interest point detection

We use the multi-scale image matching algorithm similar to [47]. The first step of matching the two images is to detect a set of interest points on each image. In the scale-space framework, we use the Harris-Laplacian interest point detector (as described in Section 2.1.2) to detect the interest points in different scale-space levels on each image. We set the scales increasing from 2 exponentially with a scale factor 1.2 up to 15 successive steps. Also, circular window patches are used in the convolution of the Harris-Laplacian detector in order to make the detector more invariant to rotation. This method effectively detects the feature points on their characteristic scales, and these characteristic scales will be used in the computation of the point descriptors. Figure 3.4 shows the interest points extracted from an example of the image pair, $I_p^{(c)}$ and $I_q^{(f)}$. The plus signs represent the positions of the interest points, and the circle around each of them represents the corresponding circular patch. The radius of the circular patch is linearly proportional to the characteristic scale of the interest point. As shown in the figures, a corner may contain several circles with different characteristic scales.

3.3.2 Point descriptor

For each interest point detected on each image, we have to compute a point descriptor representing this point using the circular patch around it. In the scale-space framework, the scale-normalized Gaussian derivatives are used as the point descriptor. In more details, for each interest point detected in previous step, a circular patch around it is selected. The radius of the patch is linearly proportional to its characteristic scale. We estimate the

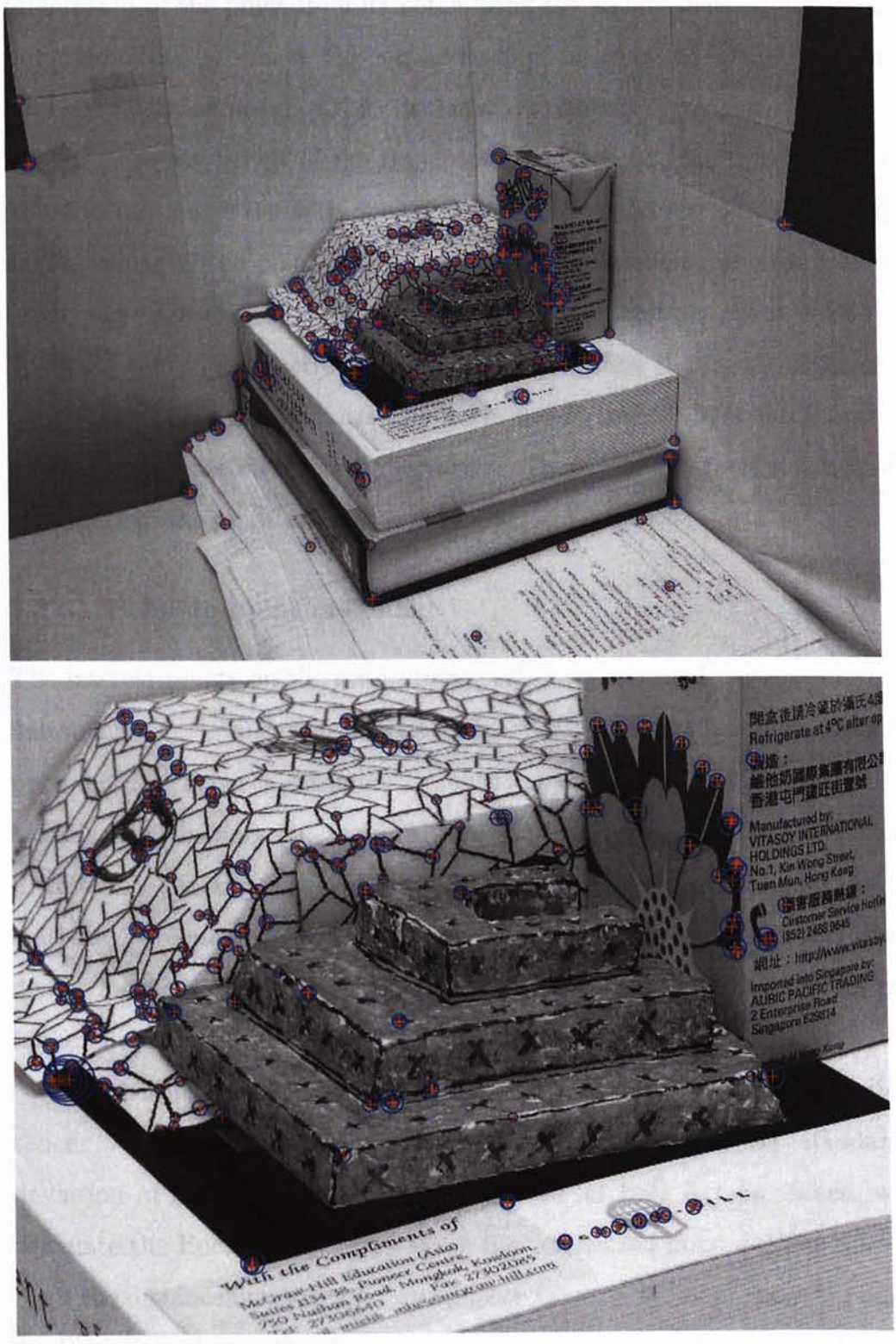


Figure 3.4: Interest points extracted for multi-scale image matching from (a) the selected coarse image ($I_p^{(c)}$) and (b) the selected fine image ($I_q^{(f)}$).

orientation of the point from its patch using the second method described in Section 2.2.1. To make the point descriptor invariant to orientation, the circular patch is steered to the orientation estimated.

The absolute values of the responses of the second, the third and the fourth order image derivatives in several directions on the patch are used as the values of the point descriptor. Totally 8 directions are used in our experiment and the angle difference between successive directions is 22.5°. To make the point descriptor invariant to the affine illumination change (the contrast change), we normalize the image derivatives by dividing them by the magnitude of the first derivative. Therefore, the total number of invariants of the point descriptor is 24.

3.3.3 Point-to-point matching

The interest points on the coarse image ($I_p^{(c)}$) are matched to those on the fine image ($I_q^{(f)}$). For each interest point on $I_p^{(c)}$, we find the most similar point on $I_q^{(f)}$ using a dissimilarity measurement on the point descriptor. Mahalanobis distance ($(d_1 - d_2)^T \Lambda^{-1} (d_1 - d_2)$) is commonly used [12, 47] as the distance function on the descriptors, d_1 and d_2. However, the covariance matrix Λ is required to be trained by a large number of image samples. We do not have any image sample in the situation of the 3D reconstruction, so Mahalanobis distance is not suitable in our project.

The dissimilarity measurement we use is the *weighted* Euclidean distance. We normalized each invariant by dividing it with the standard deviation of its distribution of all descriptors for both images. Then, we calculate the Euclidean distance using the normalized point descriptors.

If the distance between the most similar point on $I_q^{(f)}$ of an interest point on $I_p^{(c)}$ is less than a threshold, we count it as a matched pair from $I_p^{(c)}$ to

$I_q^{(f)}$ and put it into the set $Q_{(I_p^{(c)}, I_q^{(f)})}$. Matching is also applied on the reverse direction from $I_q^{(f)}$ to $I_p^{(c)}$, and we obtain the set of matched pairs $Q_{(I_q^{(f)}, I_p^{(c)})}$. Only the matched pairs of the intersection, $Q = Q_{(I_p^{(c)}, I_q^{(f)})} \cap Q_{(I_q^{(f)}, I_p^{(c)})}$, between the 2 sets of matched pairs are kept. That is, only the matched pairs from the matching of both directions are counted as the initial matches.

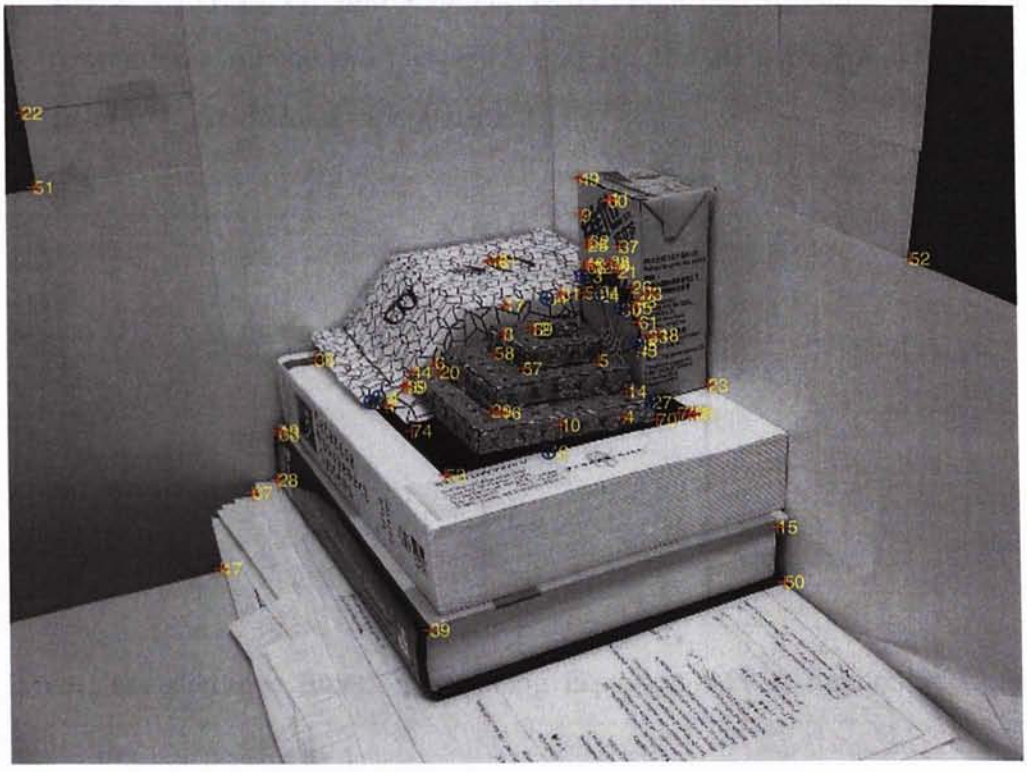
Moreover, because a single feature point on an image may contain more than one characteristic scales, it is possible that a single point correspondence contains duplicated pairs in the set Q. We have to remove the duplicated pairs so that each point correspondence has equal weight in the RANSAC counting process during the estimation of the 2D transformation (Section 3.3.4).

3.3.4 Image transformation estimation

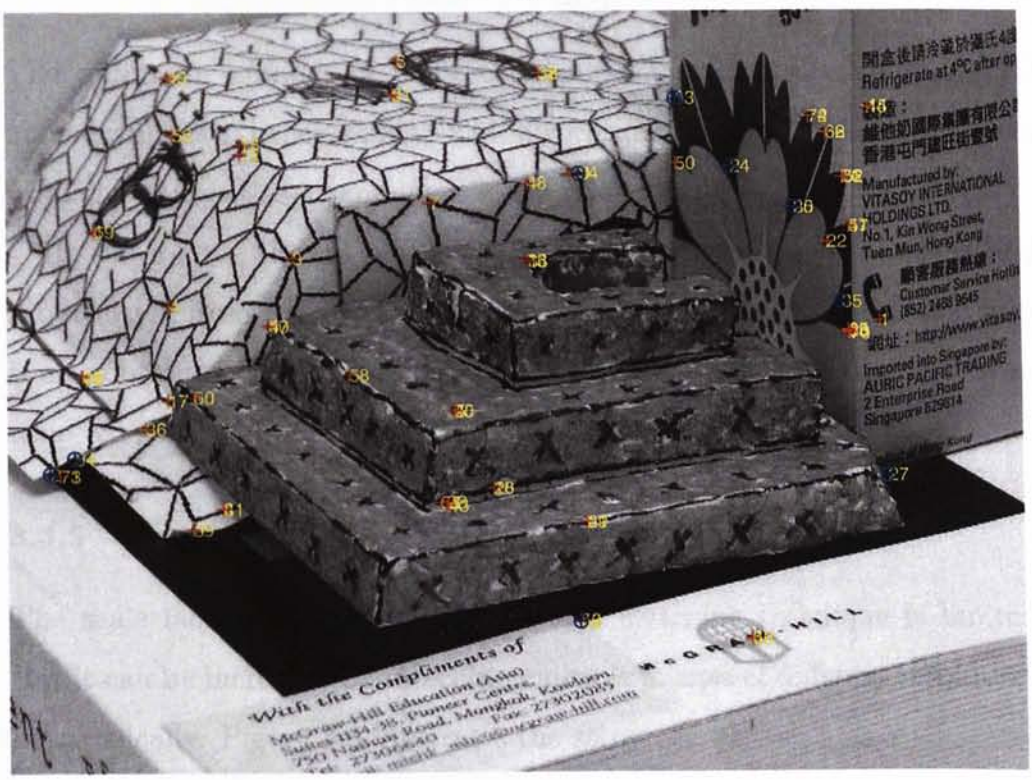
After we obtain the initial matches, we may find the transformation between the two images. However, the set of initial matches obtained contains a large portion of mismatches. Figure 3.5 shows the initial matches of the example. The plus signs represent the positions of the initial matches. The number next to a plus sign represents the index of the pair of the matched points.

The percentage of correct matches may be less than 20%. Therefore, we add the geometric constraint using RANSAC [16] to estimate the most voted transformation and separate the inliers from the outliers.

As described in Section 3.2, the relation between the images can be the fundamental matrix [26, 78, 28] or the 2D homography [28]. However, the view angles of the two images are similar, this may cause the fundamental matrix to degenerate [75]. Therefore, the 2D homography is used as the relation between the point correspondences on the two images in RANSAC, and also is used as the transformation between the two images.



(a)



(b)

Figure 3.5: Matches from multi-scale image matching for the example. Initial matches (plus) and correct matches (circled plus) on (a) the selected coarse image ($I_p^{(c)}$) and (b) the selected fine image ($I_q^{(f)}$).

Let $\mathbf{x} = (x_1, x_2, 1)^T$ and $\mathbf{x}' = (x'_1, x'_2, 1)^T$ be a pair of points of a point correspondence on the two images (I_1 and I_2), the 2D homography from I_1 to I_2, $H^{(2D)}_{(I_1,I_2)}$, is defined in Equation 3.1.

$$\mathbf{x}' = H^{(2D)}_{(I_1,I_2)}\mathbf{x} \tag{3.1}$$

where $H^{(2D)}_{(I_1,I_2)}$ is in the form below

$$H^{(2D)}_{(I_1,I_2)} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ 0 & 0 & 1 \end{bmatrix}$$

Then, the distance function replacing Equation 2.29 is defined in Equation 3.2.

$$d = \|\mathbf{x}' - H^{(2D)}_{(I_1,I_2)}\mathbf{x}\| \tag{3.2}$$

The minimum number of point correspondences for calculating the 2D homography is 3, hence, in RANSAC algorithm, we randomly select 3 points from the data set as a sample for every trail.

In Figure 3.5, the circled plus sign means the match pair is correct and is outputted from RANSAC algorithm. The corresponding 2D homography is also obtained. In this case, from the 2D homography estimated, the scale ratio of $I_q^{(f)}$ over $I_p^{(c)}$ is about 2.96.

3.3.5 Multi-level image matching

The scale factor of the multi-scale image matching technique is limited. But it can be increased by matching multiple images of different resolutions hierarchically. Figure 3.6 illustrates the setup.

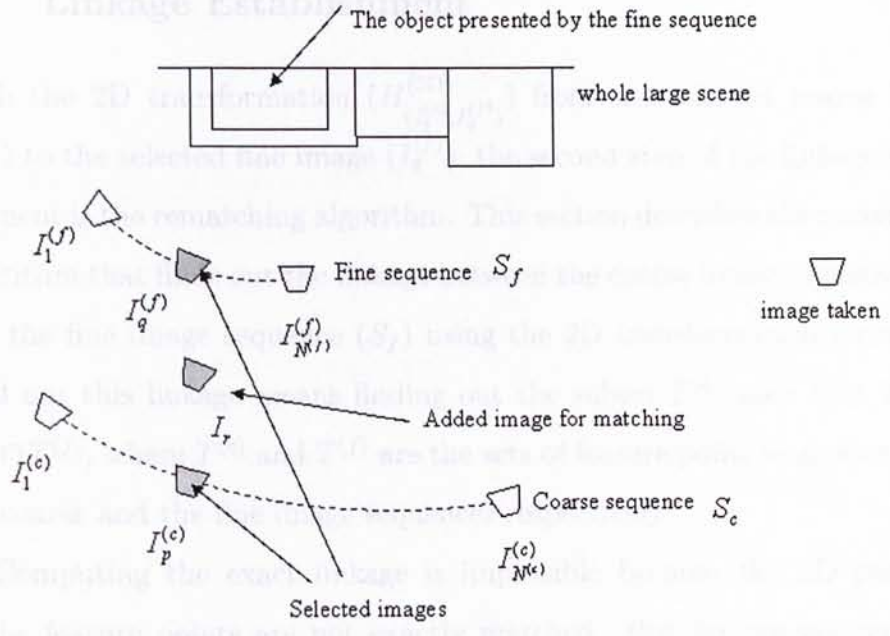


Figure 3.6: Multi-level image matching.

The 2D homography can be accumulated using Equation 3.3.

$$H^{(2D)}_{(I_p^{(c)},I_q^{(f)})} = H^{(2D)}_{(I_p^{(c)},I_r)} H^{(2D)}_{(I_r,I_q^{(f)})} \tag{3.3}$$

However, from the experiment in Section 4.1.1, the accuracy of the rematching algorithm drops when the scale ratio is too large, and the noise is more sensitive and destructive to the result of the rematching algorithm when the scale ratio is larger. Therefore, the accumulated 2D homography cannot be too large. In this case, the multiple 3D model registrations instead of multi-level image matching should be used. That is, we hierarchically register more than two 3D models of different resolutions.

3.4 Linkage Establishment

With the 2D transformation $(H^{(2D)}_{(I_p^{(c)}, I_q^{(f)})})$ from the selected coarse image $(I_p^{(c)})$ to the selected fine image $(I_q^{(f)})$, the second step of the linkage establishment is the rematching algorithm. This section describes the rematching algorithm that finds out the linkage between the coarse image sequence (S_c) and the fine image sequence (S_f) using the 2D transformation estimated. Find out this linkage means finding out the subset $T^{(L)}$ such that $T^{(L)} = T^{(c)} \cap T^{(f)}$, where $T^{(c)}$ and $T^{(f)}$ are the sets of feature point sequences along the coarse and the fine image sequences respectively.

Computing the exact linkage is impossible because the 2D positions of the feature points are not exactly matched. But we can compute the approximate linkage between $T^{(c)}$ and $T^{(f)}$, $\hat{T}^{(L)}$. Let $\mathbf{x}^{(c)}_{(i,k)}$ be the k-th 2D feature point tracked in the i-th image frame of S_c, where $1 \leq i \leq M^{(c)}$ and $M^{(c)}$ is the number of image frames of S_c, and $1 \leq k \leq N^{(c)}$ and $N^{(c)}$ is the number of features tracked along S_c. On the other hand, let $\mathbf{x}^{(f)}_{(j,l)}$ be the l-th 2D feature point tracked in the j-th image frame of S_f, where $1 \leq j \leq M^{(f)}$ and $M^{(f)}$ is the number of image frames of S_f, and $1 \leq l \leq N^{(f)}$ and $N^{(f)}$ is the number of features tracked along S_f.

Suppose $Y_p^{(c)} = \{\mathbf{x}^{(c)}_{(p,1)}, \mathbf{x}^{(c)}_{(p,2)}, \ldots, \mathbf{x}^{(c)}_{(p,N^{(c)})}\}$ is the set of feature points in the selected image frame $I_p^{(c)}$ of S_c, and $Y_q^{(f)} = \{\mathbf{x}^{(f)}_{(q,1)}, \mathbf{x}^{(f)}_{(q,2)}, \ldots, \mathbf{x}^{(f)}_{(q,N^{(f)})}\}$ is that of S_f. Our approximation is to find the set, $\hat{Y}^{(L)}$, containing the feature points of $Y_p^{(c)}$ and $Y_q^{(f)}$ that are approximately common in position under the transformation of $H^{(2D)}_{(I_p^{(c)}, I_q^{(f)})}$. Two positions are said to be approximately common if the Euclidean distance between them is below a threshold d_s. Algorithm 3.1 shows the rematching algorithm that computes $\hat{Y}^{(L)}$.

$$\hat{\mathbf{x}}^{(c)}_{(p,k)} = H^{(2D)}_{(I_p^{(c)}, I_q^{(f)})} \mathbf{x}^{(c)}_{(p,k)} \tag{3.4}$$

Algorithm 3.1 Rematching Algorithm

1: **for all** $\mathbf{x}^{(c)}_{(p,k)} \in P_p^{(c)}$ **do**
2: Transform $\mathbf{x}^{(c)}_{(p,k)}$ to $\hat{\mathbf{x}}^{(c)}_{(p,k)}$ by the transformation, $H^{(2D)}_{(I_p^{(c)}, I_q^{(f)})}$, using Equation 3.4.
3: Find the nearest feature point in $I_q^{(f)}$ of $\hat{\mathbf{x}}^{(c)}_{(p,k)}$ using Euclidean distance as the distance function.
4: If the distance is below the threshold d_s, the transformed point and the nearest feature point is counted as the matched pair and put it into the set $E_{(I_p^{(c)}, I_q^{(f)})}$.
5: **end for**
6: Repeat lines 1 to 5 by reversing S_c with S_f. The transformation between the two images is changed to $H^{(2D)}_{(I_q^{(f)}, I_p^{(c)})}$, $H^{(2D)}_{(I_q^{(f)}, I_p^{(c)})} = \text{inv}(H^{(2D)}_{(I_p^{(c)}, I_q^{(f)})})$ and $\text{inv}(A)$ is the inverse of the square matrix A. The set of matched pairs is $E_{(I_q^{(f)}, I_p^{(c)})}$.
7: Compute $\hat{Y}^{(\text{L})}$ by $\hat{Y}^{(\text{L})} = E_{(I_p^{(c)}, I_q^{(f)})} \cap E_{(I_q^{(f)}, I_p^{(c)})}$.

Figure 3.7 illustrates the rematching algorithm. The point $\mathbf{x}_C$ is within the range of distance d_s centered at the transformed co-ordinates $(H^{(2D)}_{(I_p^{(c)}, I_q^{(f)})} \mathbf{x}_A)$ of the point $\mathbf{x}_A$, and it is the nearest point to that co-ordinates. Also, the point $\mathbf{x}_A$ is the nearest point and within the range of distance d_s centered at the transformed co-ordinates $(H^{(2D)}_{(I_q^{(f)}, I_p^{(c)})} \mathbf{x}_C)$ of the point $\mathbf{x}_C$. Therefore, $\mathbf{x}_A$ and $\mathbf{x}_C$ form a correspondence. The points $\mathbf{x}_B$ and $\mathbf{x}_C$ do not form a correspondence because $\mathbf{x}_B$ is more far away from the transformed co-ordinates of point $\mathbf{x}_C$ than the point $\mathbf{x}_A$.

In our project, the threshold d_s is set to twice to the scale ratio of S_f over S_c which can be estimated from $H^{(2D)}_{(I_p^{(c)}, I_q^{(f)})}$.

Figure 3.5 shows an example of the matches. The circled plus signs are the matched points in the coarse and the fine image sequences.

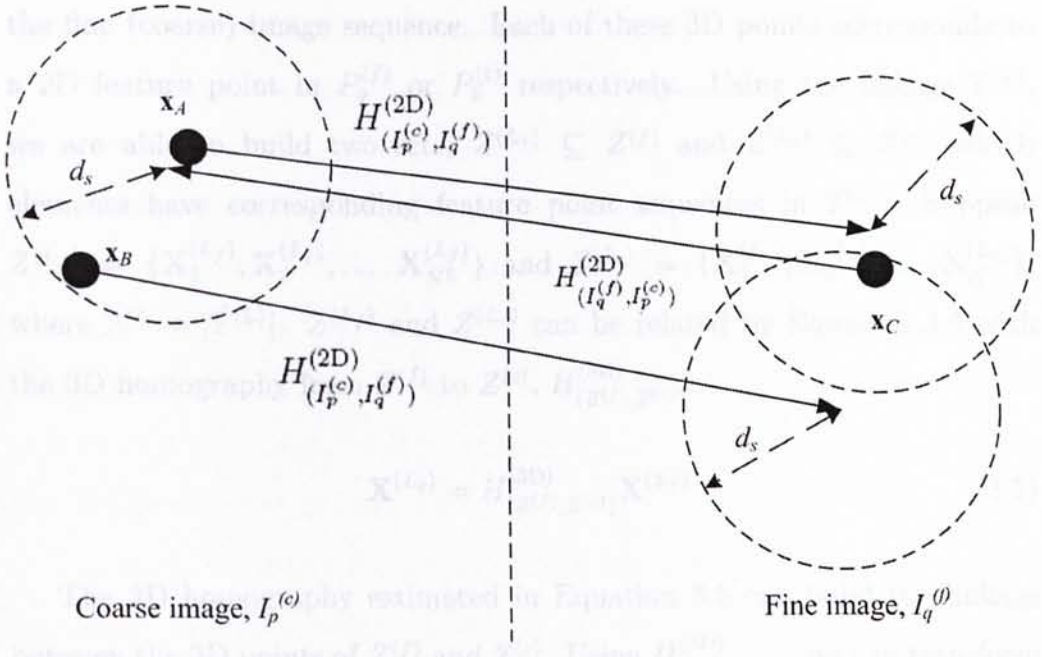


Figure 3.7: Illustration of the rematching algorithm

With $\hat{Y}^{(\mathrm{L})}$, $\hat{T}^{(\mathrm{L})}$ can also be found very easily because the feature points are tracked along the whole image sequence. The index of a feature point is the same along the whole sequence. So, the feature point sequences in $\hat{T}^{(\mathrm{L})}$ correspond to the same feature points in $\hat{Y}^{(\mathrm{L})}$.

3.5 3D Model Registration

Using the linkage on the feature point sequences found in Section 3.4, we can also establish the linkage of the 3D points between the 3D structures. Using this linkage, we can merge the 3D models to form a combined model.

Let $Z^{(f)} = \{\mathbf{X}_1^{(f)}, \mathbf{X}_2^{(f)}, \ldots, \mathbf{X}_{N(f)}^{(f)}\}$ and $Z^{(c)} = \{\mathbf{X}_1^{(c)}, \mathbf{X}_2^{(c)}, \ldots, \mathbf{X}_{N(c)}^{(c)}\}$ be the sets of 3D coordinates of the feature points of the fine image sequence and the coarse image sequence relative to their first cameras respectively. That is, $\mathbf{X}_k^{(f)}$ ($\mathbf{X}_l^{(c)}$) are the k-th (l-th) 3D points in the 3D structures of

the fine (coarse) image sequence. Each of these 3D points corresponds to a 2D feature point in $P_q^{(f)}$ or $P_p^{(c)}$ respectively. Using the linkage $\hat{T}^{(\mathrm{L})}$, we are able to build two sets, $Z^{(L_f)} \subseteq Z^{(f)}$ and $Z^{(L_c)} \subseteq Z^{(c)}$, which elements have corresponding feature point sequences in $\hat{T}^{(\mathrm{L})}$. Suppose $Z^{(L_f)} = \{\mathbf{X}_1^{(L_f)}, \mathbf{X}_2^{(L_f)}, \ldots, \mathbf{X}_{N^L}^{(L_f)}\}$ and $Z^{(L_c)} = \{\mathbf{X}_1^{(L_c)}, \mathbf{X}_2^{(L_c)}, \ldots, \mathbf{X}_{N^L}^{(L_c)}\}$, where $N^L = |\hat{T}^{(\mathrm{L})}|$. $Z^{(L_f)}$ and $Z^{(L_c)}$ can be related by Equation 3.5 with the 3D homography from $Z^{(f)}$ to $Z^{(c)}$, $H^{(3\mathrm{D})}_{(Z^{(f)},Z^{(c)})}$.

$$\mathbf{X}^{(L_c)} = H^{(3\mathrm{D})}_{(Z^{(f)},Z^{(c)})}\mathbf{X}^{(L_f)} \tag{3.5}$$

The 3D homography estimated in Equation 3.5 can build the linkage between the 3D points of $Z^{(f)}$ and $Z^{(c)}$. Using $H^{(3\mathrm{D})}_{(Z^{(f)},Z^{(c)})}$, we can transform the 3D structures, $Z^{(f)}$ and $Z^{(c)}$, to each other's coordinate system by Equation 3.6 or Equation 3.7, respectively. Therefore, the 3D structures of the two sequences are transformed and merged, and we obtain the combined model.

$$\mathbf{X}^{(c)} = H^{(3\mathrm{D})}_{(Z^{(f)},Z^{(c)})}\mathbf{X}^{(f)} \tag{3.6}$$

$$\mathbf{X}^{(f)} = \mathrm{inv}(H^{(3\mathrm{D})}_{(Z^{(f)},Z^{(c)})})\mathbf{X}^{(c)} \tag{3.7}$$

The 3D transformation can be metric, affine or projective [56, 28]. In this project, we use affine transformation because affine transformation can register the two 3D models better than metric transformation (as shown in Section 4.1.2). Projective transformation is useful when the 3D models reconstructed for the image sequences are scale up to projective transformation. But the 3D models reconstructed in this project are scale up to a scalar, so projective transformation is not necessary for this project. Also,

the computation of projective transformation (Section 2.6.2) is more complex than affine transformation, in our observation, the result of projective transformation is less stable than affine transformation for this project.

The image sequences can be calibrated using the self-calibration algorithm up to a scale factor. We assume that we know the focal length of the first image frame of the coarse image sequence to fix the scale factor, but for the fine image sequence it can be fixed arbitrarily. From the experiment in Section 4.1.2, the registration error is small even if the focal length of the fine image sequence is fixed arbitrarily.

Equation 3.8 shows the format of the 3D homography for affine transformation, $\mathbf{X}$ is in the form of $(X_1, X_2, X_3, 1)^T$.

$$H^{(3D)}_{(Z^{(f)},Z^{(c)})} = \begin{bmatrix} h_{11} & h_{12} & h_{13} & h_{14} \\ h_{21} & h_{22} & h_{23} & h_{24} \\ h_{31} & h_{32} & h_{33} & h_{34} \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.8}$$

The 3D homography for affine transformation can be estimated using the least square approximation on Equation 3.5.

RANSAC can also be used to remove the outlying point correspondences for the estimation of the 3D homography. We can change the relation to affine transformation (Equation 3.8) and replace the distance function by that is defined in Equation 3.9.

$$d = \|\mathbf{X}^{(c)} - H^{(3D)}_{(Z^{(f)},Z^{(c)})}\mathbf{X}^{(f)}\| \tag{3.9}$$

The minimum number of correspondences required for the estimation of

affine transformation is 4. Hence, the number of correspondences required for each trial is 4.

3.6 VRML Modelling

In our implementation, the wire-frame for each individual 3D model is generated using the "delaunay" function in Matlab [45] on the middle image frame of the corresponding image sequence. The 3D wire-frame model is then created with the 3D points by the 3D model reconstruction. For each triangle of the wire-frame model, the texture of the corresponding triangle in the largest area among image frames of the corresponding image sequence is chosen as the texture of that triangle. The 3D models are outputted using the VRML format.

For the combined 3D model, we first remove the triangles of the coarse 3D model which are inside the "region" of the fine 3D model. Then, we directly place the fine 3D model into the coarse 3D model. The region is a rectangular region that its boundaries are the minimum and the maximum x and y co-ordinates of the re-projection of the fine 3D model on the middle frame of the coarse image sequence.

□ **End of chapter.**

Chapter 4

Experiment

This chapter presents the experiments that show the characteristics of our system and evaluate the proposed method. Two synthetic experiments that investigate the rematching algorithm and the 3D model registration are presented in Section 4.1. Section 4.2 presents the results of the proposed algorithm for the real cases. The platform we used was MatLab [45] version 6.5 in Windows. The machine we used was a PC with CPU of Intel Pentium®4 2GHz.

4.1 Synthetic Experiments

This section presents two synthetic experiments that evaluate the rematching algorithm and compare affine transformation with metric transformation for the 3D model registration.

4.1.1 Study on Rematching Algorithm

This experiment aims to test the effectiveness of the rematching algorithm (Section 3.4) using synthetic data. In this section, we first describe the

setup and the measurement of this experiment, and then we present and discuss the results.

Setup

For each run of the experiment, a set of n 2D points ($\mathcal{X}_1^f$) and a set of n_1 2D points ($\mathcal{X}_2^f$) for the object of size w in the fine image ($I^{(f)}$) are randomly generated. The first set of points ($\mathcal{X}_1^f$) were transformed by the metric transformation, H (3×3), and generated the set of 2D points, $\mathcal{X}_1^c$. This metric transformation contains a fixed scale ratio s, and a randomly generated rotation angle and a randomly generated translation vector. A set of n_2 2D points ($\mathcal{X}_2^c$) for the whole scene in the coarse image ($I^{(c)}$) were randomly generated. The two sets ($\mathcal{X}_1^f$ and $\mathcal{X}_2^f$) were combined to form a set $\mathcal{X}_n^f$, and the other two sets ($\mathcal{X}_1^c$ and $\mathcal{X}_2^c$) were combined to form a set $\mathcal{X}_n^c$.

The combined set $\mathcal{X}_n^f$ simulates the set of 2D points tracked by the KLT tracker (Section 2.4.1) for the fine image sequence, and the combined set $\mathcal{X}_n^c$ simulates the set of 2D points tracked by the KLT tracker for the coarse image sequence.

To test the effectiveness of the rematching algorithm under noise, we added noise or error to the data points or the transformation. The 2D points in the sets $\mathcal{X}_1^n$ and $\mathcal{X}_2$ were added with 2D noise with amplitude $noise^{2D}$. The rotation angle, the translation vector and the scale were added with fixed errors with values ϵ_ϕ, ϵ_t and $\epsilon_s \times s$ (ϵ_s is a scale error ratio), respectively. An evaluation transformation, $H^{(e)}$ was then formed using the new rotation angle, translation vector and scale. The rematching algorithm was then used to rematch the sets $\mathcal{X}_n^f$ and $\mathcal{X}_n^c$ using $H^{(e)}$. The points in $\mathcal{X}_1^f$ should be rematched to the corresponding points in $\mathcal{X}_1^c$.

Table 4.1: The parameters used in the experiment

Parameter	Part 1	Part 2	Part 3	Part 4
The no. of runs	100	100	100	100
n	50	50	50	50
n_1	250	250	250	250
n_2	250	250	250	250
w (pixel)	300	300	300	300
s	1 to 4	1 to 4	1 to 4	1 to 4
$noise^{2D}$ (pixel)	0 to 6	0	0	0
ϵ_ϕ (degree)	0	0 to 5	0	0
ϵ_t (pixel)	0	0	0 to 5	0
ϵ_s	0	0	0	0 to 0.1

To evaluate the results in different situations, we divided the experiment into four parts: the probability for different scales against 1) the 2D noise ($noise^{2D}$), 2) the rotation angle error (ϵ_ϕ), 3) the translation error (ϵ_t) and 4) the scale ratio error (ϵ_s) will be investigated.

Table 4.1 summaries the values of the parameters used in this experiment.

Measurement Let N_m be the number of rematched pairs and N_c be the number of correctly rematched pairs. The case is "valid" if N_c is larger than or equal to 8, and $\frac{N_c}{N_m} > 0.5$. The former criterion to the doubling of the minimum number of correspondences required for 3D affine transformation is 4. The latter one exists because we can remove the outliers (mismatched pairs) using RANSAC for 3D affine transformation (Section 3.5). In this experiment, we used 100 runs of the experiment and count the number of the valid runs. The measurement was the probability that the run is valid, $(= \frac{\text{Number of the valid runs}}{\text{Number of runs}})$.

Results & Discussions

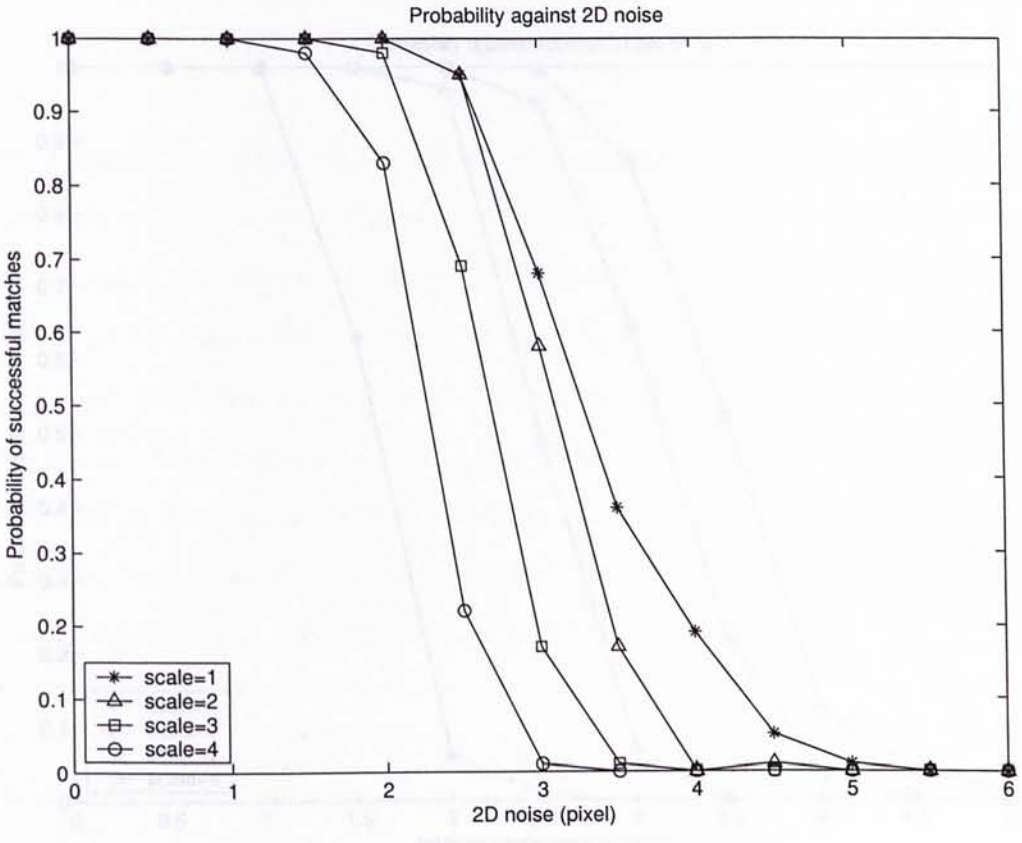


Figure 4.1: Probability of successful matching against 2D noise for different scales

1) 2D noise (*noise2D*) Figure 4.1 shows the results of part 1 of the experiment. The probabilities were 1 when the 2D noise to the data sets was small ($noise2D \leq 1$). When the 2D noise was small, the pairs of points were correctly rematched by the rematching algorithm. However, when the 2D noise became larger, the probabilities began to drop. The situation was the most serious for the scale of 3, and the least serious for the scale of 1. This is because when the scale ratio is larger, the points in the coarse image ($\mathcal{X}_n^c$) is more concentrated together and the distance between each pair of points in the same image is smaller. The points are more easily mismatched

during the rematching process.

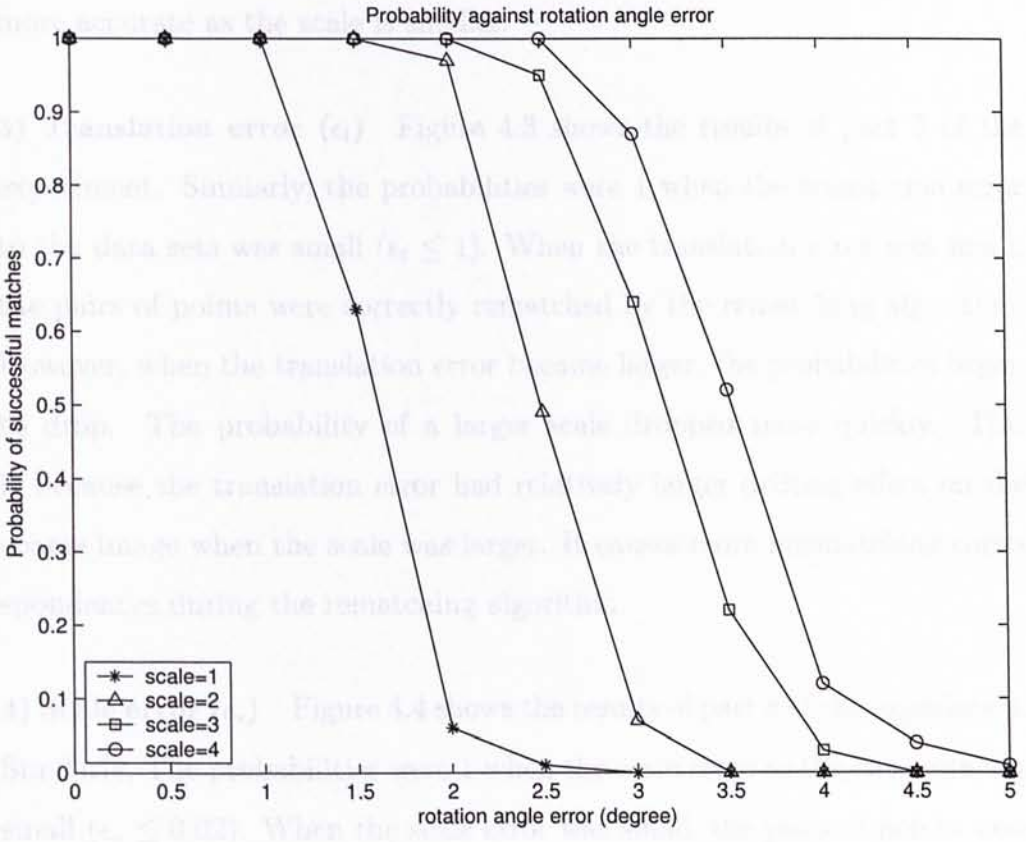


Figure 4.2: Probability of successful matching against rotation angle error for different scales

2) Rotation angle error (ϵ_ϕ) Figure 4.2 shows the results of part 2 of the experiment. Similarly, the probabilities were 1 when the rotation angle error to the data sets was small ($\epsilon_\phi \leq 1$). When the rotation angle error was small, the pairs of points were correctly rematched by the rematching algorithm. However, when the rotation angle error became larger, the probabilities began to drop. The situation was the most serious for the scale of 1, and the least serious for the scale of 3. This is because the threshold d_s of the rematching algorithm (Section 3.4) increases as the scale. It has a larger

range for a point to find the corresponding point in the other image when the threshold is larger. But the 2D transformation estimated is usually more accurate as the scale is smaller.

3) Translation error (ϵ_t) Figure 4.3 shows the results of part 3 of the experiment. Similarly, the probabilities were 1 when the translation error to the data sets was small ($\epsilon_t \leq 1$). When the translation error was small, the pairs of points were correctly rematched by the rematching algorithm. However, when the translation error became larger, the probabilities began to drop. The probability of a larger scale dropped more quickly. This is because the translation error had relatively larger drifting effect on the coarse image when the scale was larger. It causes more mismatching correspondences during the rematching algorithm.

4) Scale error (ϵ_s) Figure 4.4 shows the results of part 4 of the experiment. Similarly, The probabilities were 1 when the scale error to the data sets was small ($\epsilon_s \leq 0.02$). When the scale error was small, the pairs of points were correctly rematched by the rematching algorithm. However, when the scale error became larger, the probabilities began to drop. Similar to part 2, the situation was more serious for a smaller scale.

Summary

This experiment shows the effectiveness of the rematching algorithm. Generally speaking, the rematching algorithm can endure small error on the 2D transformation and the noise on the data points. However, when the error or the noise becomes larger, the results of the rematching algorithm turn poor quickly. Therefore, the rematching algorithm requires an accurate

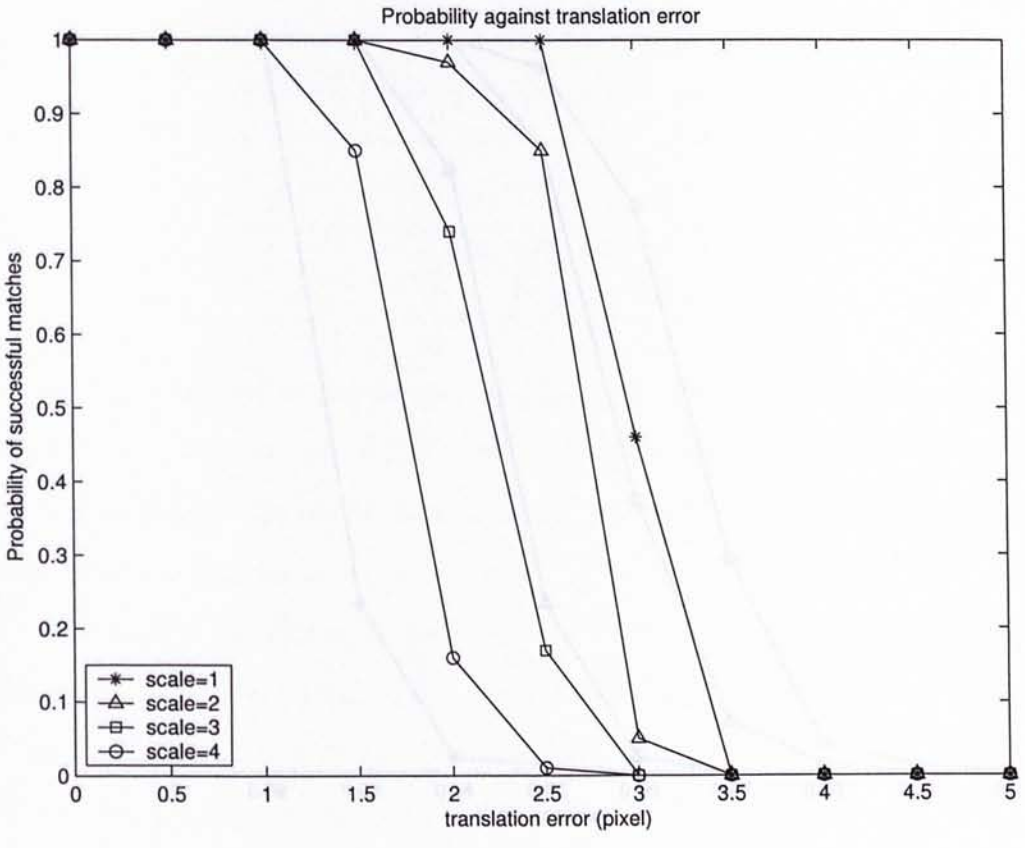


Figure 4.3: Probability of successful matching against translation error for different scales

relation between the selected image pair to build the linkage between the coarse and the fine image sequences.

4.1.2 Comparison between Affine and Metric transformations for 3D Registration

This experiment is used to compare the affine transformation with the metric transformation for 3D registration using synthetic data. This section first describes the setup of the experiment, and then presents and discusses the results.

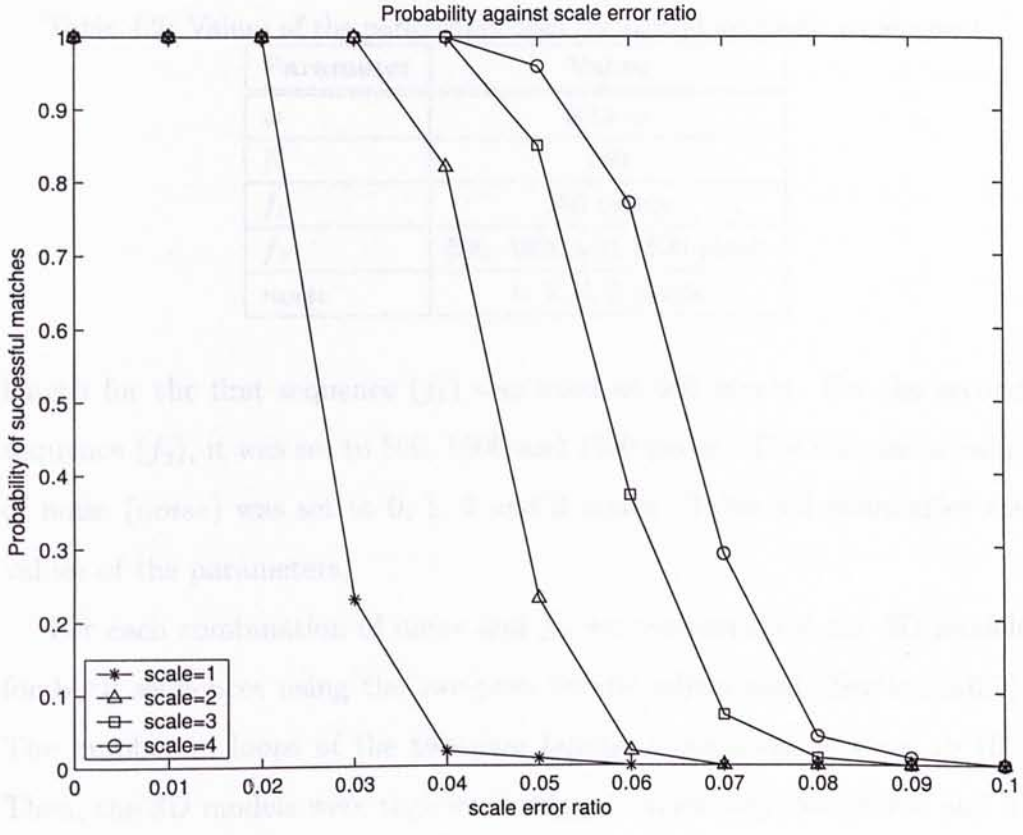


Figure 4.4: Probability of successful matching against scale error for different scales

Setup

For each run of the test, a 3D object of size w with N 3D points were randomly generated. The 3D object was projected to two sequences of 10 frames. For each sequence, the object rotated 2 degrees along both x-axis and y-axis for every successive frame of the sequence. Also, the focal length for every frame was fixed along the sequence. Let f_1 and f_2 be the focal lengths for the first and the second sequences respectively. 2D noise with the maximum value, *noise*, was added to every 2D point.

In this experiment, we set the size of the object (w) to 0.13 m and the pixel width to 5.42 μm. The number of points (N) was 100. The focal

Table 4.2: Values of the parameters used for second synthetic experiment

Parameter	Value
w	0.13 m
N	100
f_1	500 pixels
f_2	500, 1000 and 1500 pixels
noise	0, 1, 2, 3 pixels

length for the first sequence (f_1) was fixed at 500 pixels. For the second sequence (f_2), it was set to 500, 1000 and 1500 pixels. The maximum value of noise (*noise*) was set to 0, 1, 2 and 3 pixels. Table 4.2 summaries the values of the parameters.

For each combination of *noise* and f_2, we reconstructed the 3D models for both sequences using the two-pass bundle adjustment (Section 2.6.3). The number of loops of the two-pass bundle adjustment was set to 100. Then, the 3D models were registered using 1) affine transformation and 2) metric transformation. For each transformation, we measured the registration error using the 3D registration root-mean-square (rms) error which is defined by Equation 4.1.

$$e = \sqrt{\frac{\sum_{i=1}^{N} (\|\mathbf{X}_i^{(1)} - \mathbf{X}_i^{(2)}\|)^2}{N}} \tag{4.1}$$

where $\mathbf{X}_i^{(1)}$ and $\mathbf{X}_i^{(2)}$ are 3D co-ordinates of the i-th points of the first reconstructed 3D model, and the second 3D model transformed by the estimated transformation, respectively.

The two-pass bundle adjustment (Section 2.6.3) we used does not adjust the focal length of each frame, hence we must fix a focal length for it. To test the 3D transformations under the case of wrong setting of the focal

length, we used the correct focal length of the reconstruction for the first sequence, and we set the focal length of the reconstruction for the second sequence to the same value as the first one.

We repeated the experiment 100 times, and then obtained the average 3D registration rms error for each case.

Results

Figure 4.5 shows the experiment results that the reconstruction for the second sequence used the correct focal lengths. Figure 4.6 shows the experiment results that the reconstruction for the second sequence used the focal length of the first sequence.

For all combinations of focal lengths of the second sequence (f_2) and the maximum noise (*noise*), and for both using correct and incorrect focal lengths, the 3D registration rms errors of affine transformation were smaller than that of metric transformation.

Also, for both transformation methods, the error increased while the maximum noise (*noise*) increased if the focal length of the second sequence was fixed.

Moreover, in Figure 4.6, when the maximum noise (*noise*) was 0, the 3D registration rms error increased while the focal length increased. On the other hand, in Figure 4.5, the 3D registration rms errors were near the same for all focal lengths when *noise* was 0. However, when *noise* was larger, the 3D registration errors for using a correct and an incorrect focal length were similar if affine transformation was used.

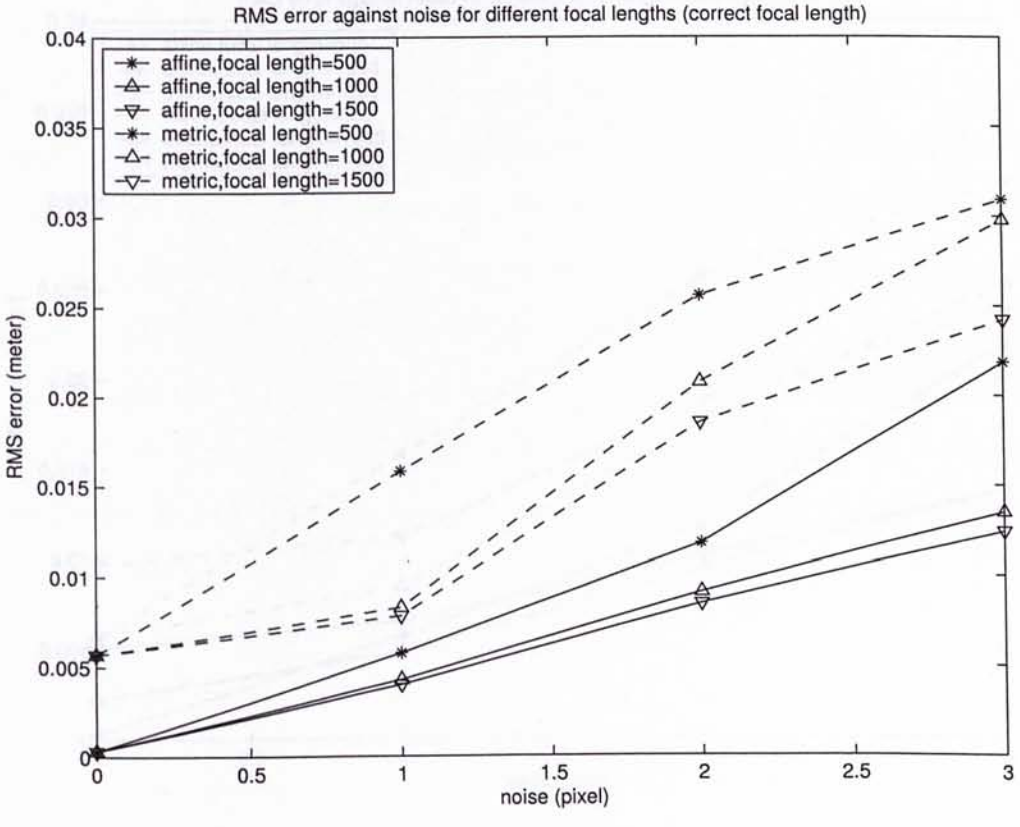


Figure 4.5: Average 3D registration rms errors against noise for different focal lengths of the second sequence. The reconstruction for the second sequence used the correct focal lengths. The dotted lines show the results of metric transformation, and the solid lines show the results of affine transformation.

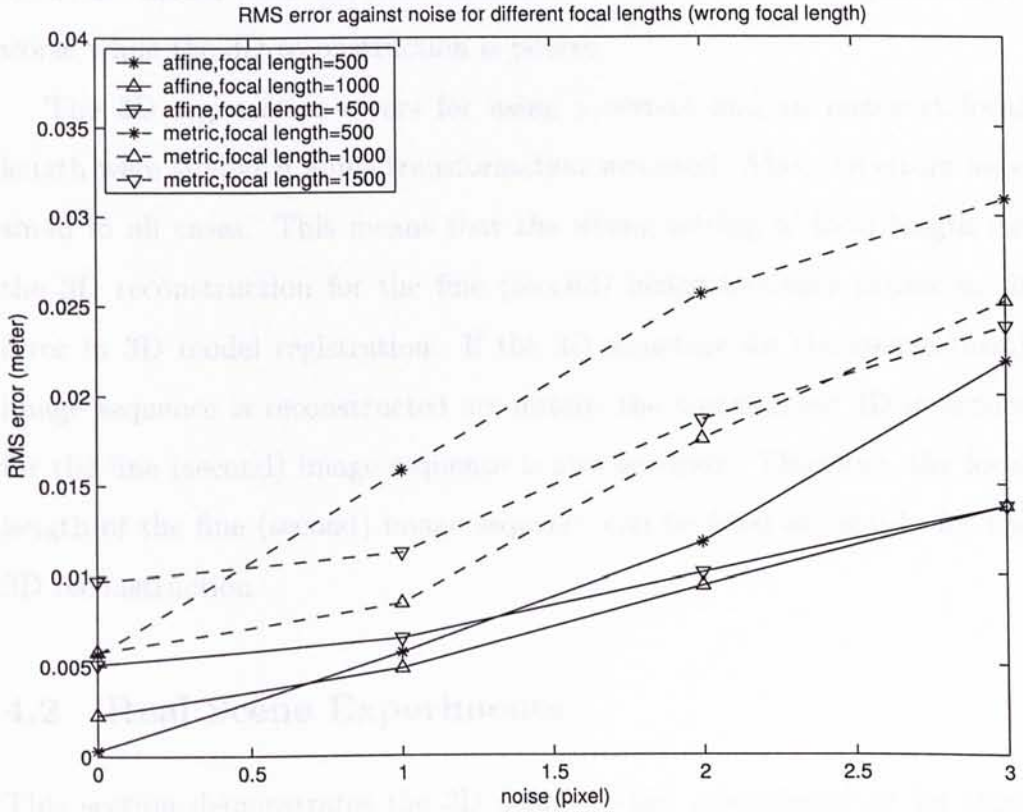


Figure 4.6: Average 3D registration rms errors against noise for different focal lengths of the second sequence. The reconstruction for the second sequence used the focal length same as the first sequence. The dotted lines show the results of metric transformation, and the solid lines show the results of affine transformation.

Discussions

Generally, affine transformation is better than metric transformation in registering 3D models. Also, the results also show that 3D registration is worse when the noise increases. This is because of the poor reconstruction result of the 3D model when the noise increases. Therefore, the 3D registration is worse while the 3D reconstruction is poorer.

The 3D registration errors for using a correct and an incorrect focal length were similar if affine transformation was used. Also, the errors were small in all cases. This means that the wrong setting of focal length for the 3D reconstruction for the fine (second) image sequence causes small error in 3D model registration. If the 3D structure for the coarse (first) image sequence is reconstructed accurately, the transformed 3D structure for the fine (second) image sequence is also accurate. Therefore, the focal length of the fine (second) image sequence can be fixed arbitrarily for the 3D reconstruction.

4.2 Real Scene Experiments

This section demonstrates the 3D coarse-to-fine reconstructions for three real scenes using our proposed method.

"Steps" Scene

Figure 4.7 shows the coarse and the fine image sequences for the "Steps" scene. Figure 4.8 shows the selected images. Figure 4.9 shows the correct matched points of the multi-scale image matching algorithm for the selected coarse and fine images. Figure 4.10 shows the results of the rematching algorithm. The magenta plus signs are the points tracked by the KLT

tracker and the points circled by yellow circles are the rematched points.

Figure 4.11 and Figure 4.12 show the individual 3D VRML models for the coarse and the fine image sequences respectively. Table 4.3 shows the information of the coarse and the fine image sequences for the "Steps" scene. Figure 4.13 shows the combined 3D VRML model for the "Steps" scene. Also, Table 4.4 shows the 2D and 3D scales approximated from the 2D homography and the 3D homography respectively. Both homographies are affine transformation, the approximated 2D and 3D scales can be obtained by Equation 4.2 and Equation 4.3 respectively. Moreover, the computation time for the "Steps" scene is shown in Table 4.5.

$$s^{(2D)} = \frac{\sqrt{(h_{11}^2 + h_{12}^2)} + \sqrt{(h_{21}^2 + h_{22}^2)}}{2} \tag{4.2}$$

$$s^{(3D)} = \frac{\sum_{i=1}^{3} \sqrt{(h_{i1}^2 + h_{i2}^2 + h_{i3}^2)}}{3} \tag{4.3}$$

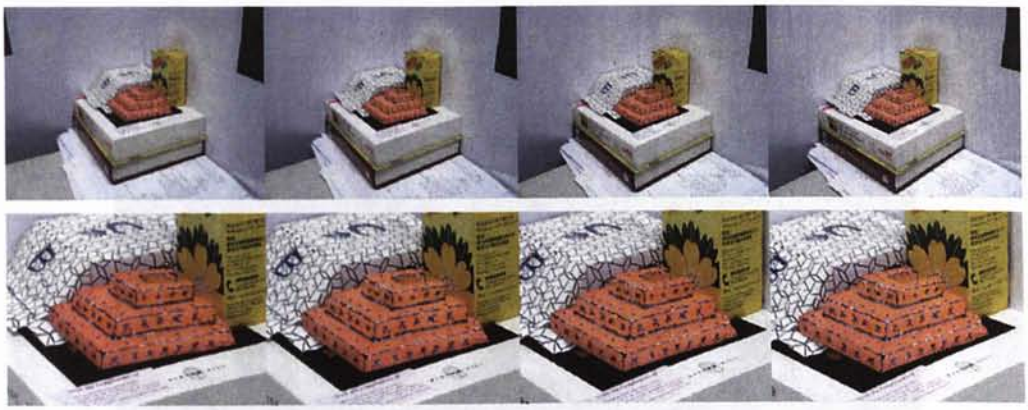
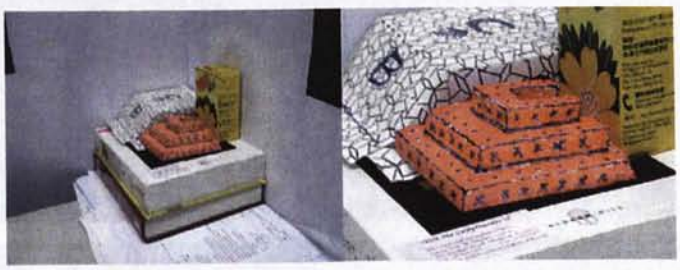


Figure 4.7: Frames for the "Steps" scene. 1st, 5th, 10th and 15th frames of (Upper row) the coarse image sequence and (Lower row) the fine image sequence.



coarse fine

Figure 4.8: Selected images for "Steps" scene.

Table 4.3: Information of the "Steps" scene

Information	**Coarse Sequence**	**Fine Sequence**
No. of frames of the sequence	19	19
No. of requested features	2000	2000
No. of features tracked by KLT	419	888
No. of features selected by RANSAC	260	796
No. of 3D points	260	796
Selected image	first image	first image
Width of each image (pixel)	640	640
Height of each image (pixel)	480	480

Table 4.4: The scales approximated for the "Steps" scene

Information	**Combined Model**
2D scale approximated	2.96
3D scale approximated	2.80

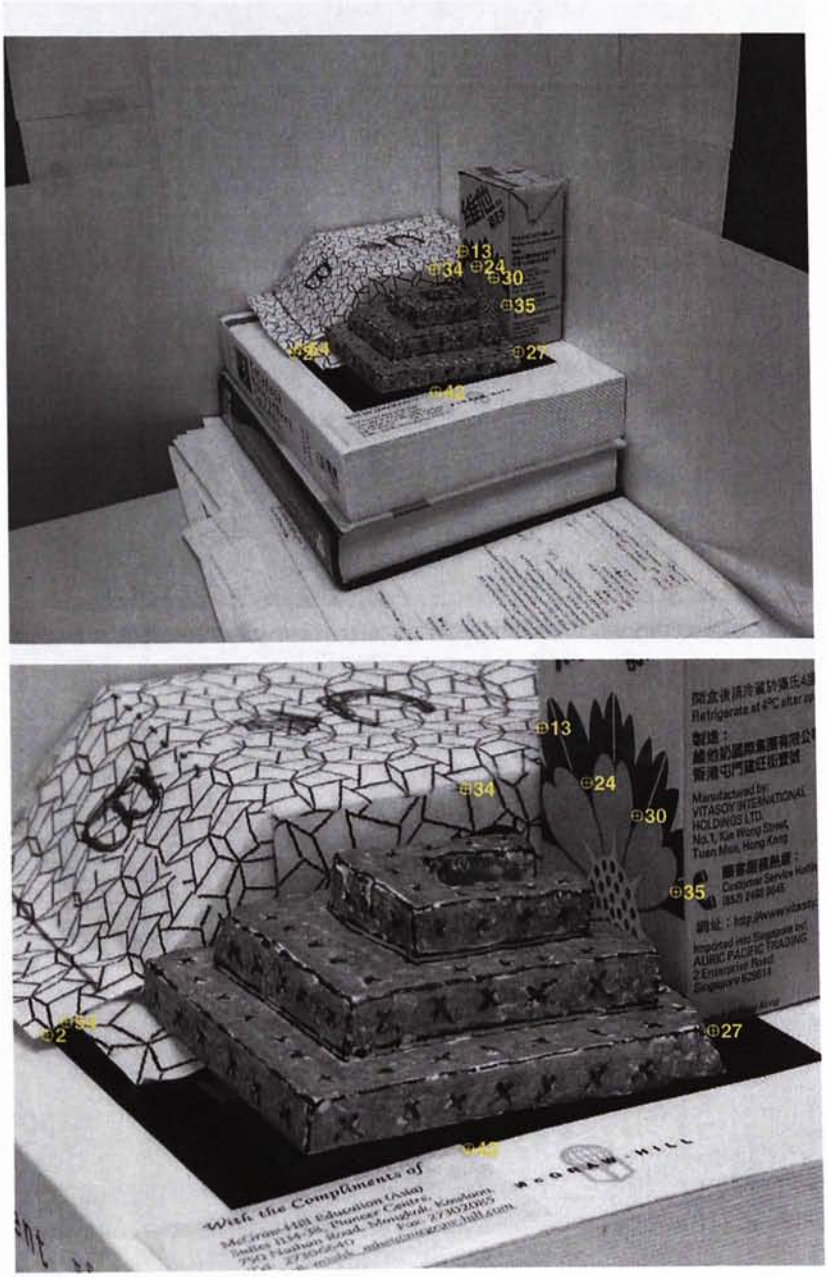


Figure 4.9: The result of the multi-scale image matching for the "Steps" scene. (Upper) The coarse image, (Lower) the fine image.

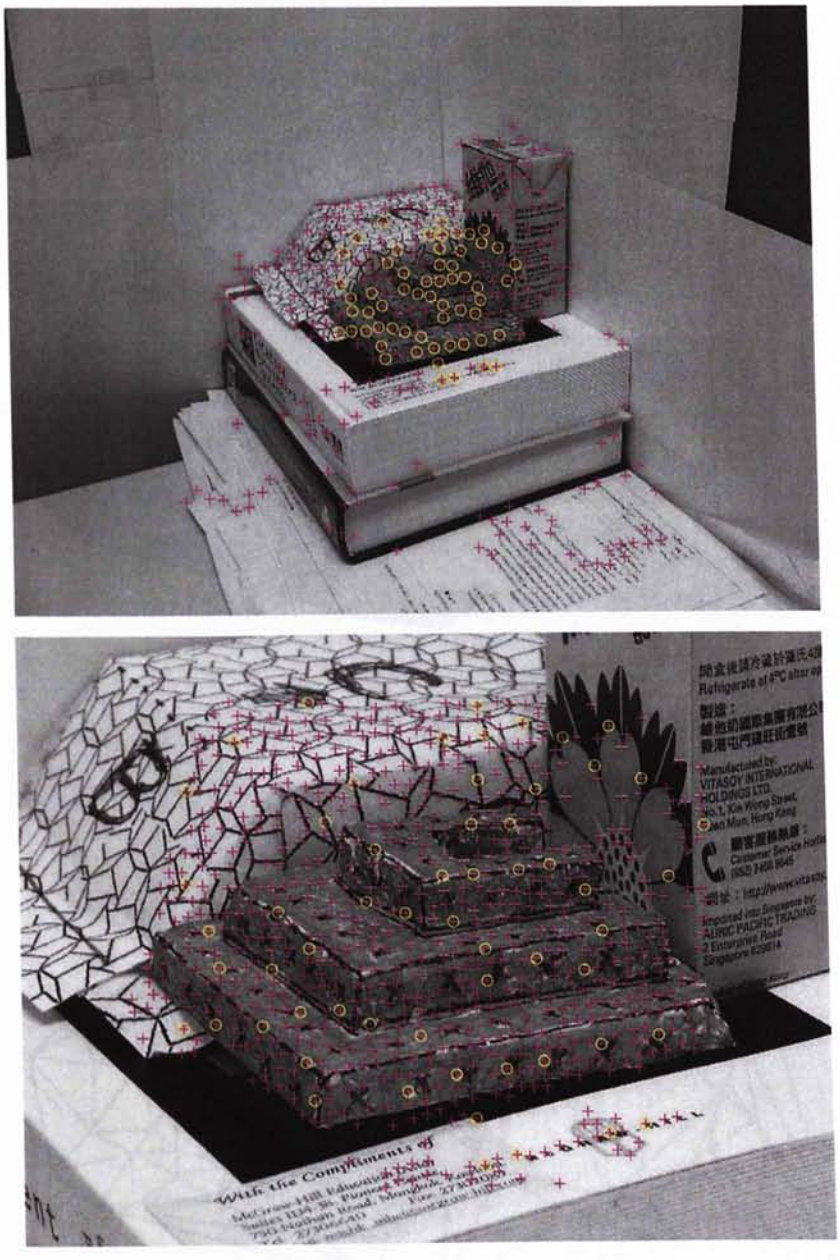


Figure 4.10: The result of the rematching algorithm for the "Steps" scene. (Upper) The coarse image, (Lower) the fine image. The plus signs are the points tracked by the KLT tracker and the points circled by yellow circles are the rematched points.

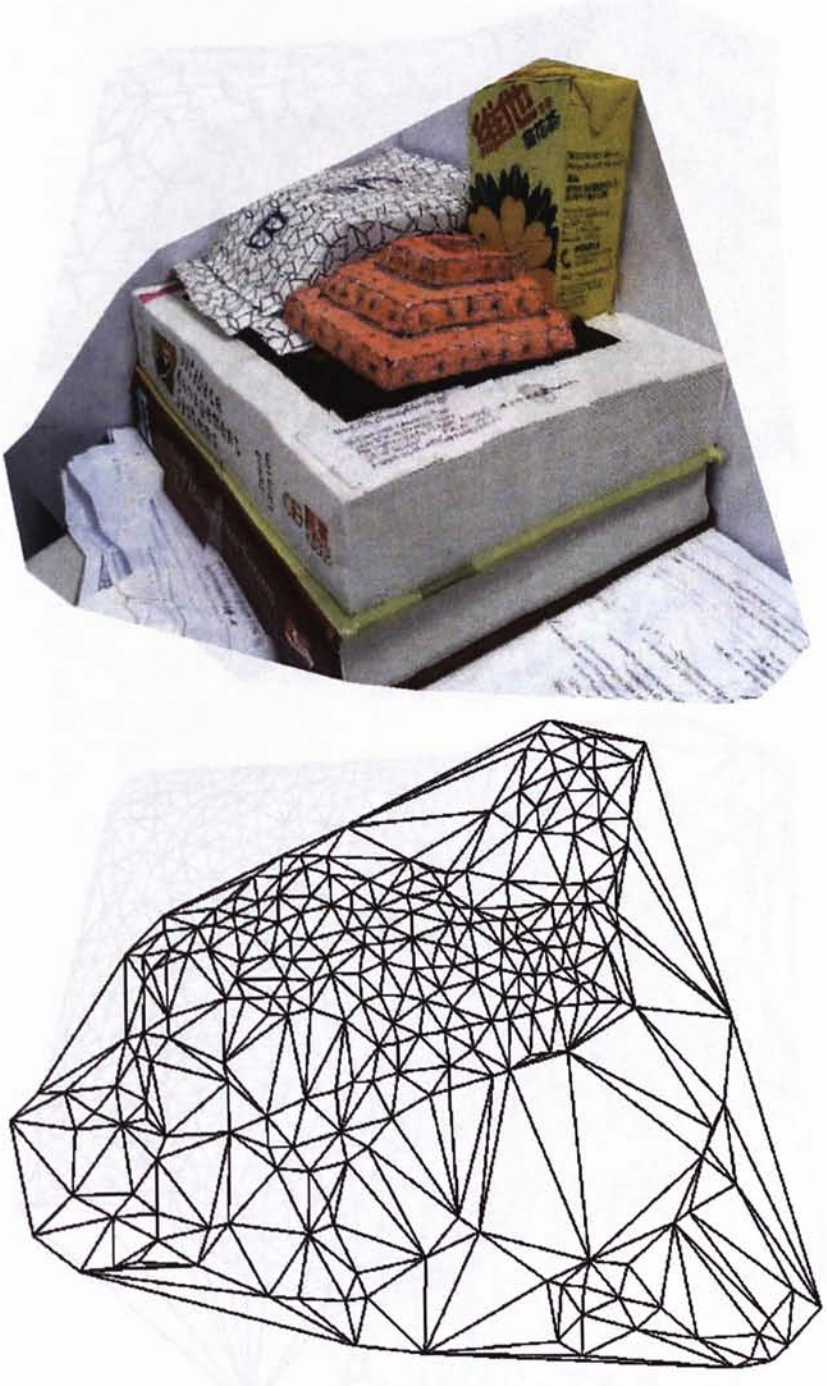


Figure 4.11: The 3D coarse model for the "Steps" scene. (Upper) The texture-mapped 3D model and (Lower) the wire-frame 3D model.

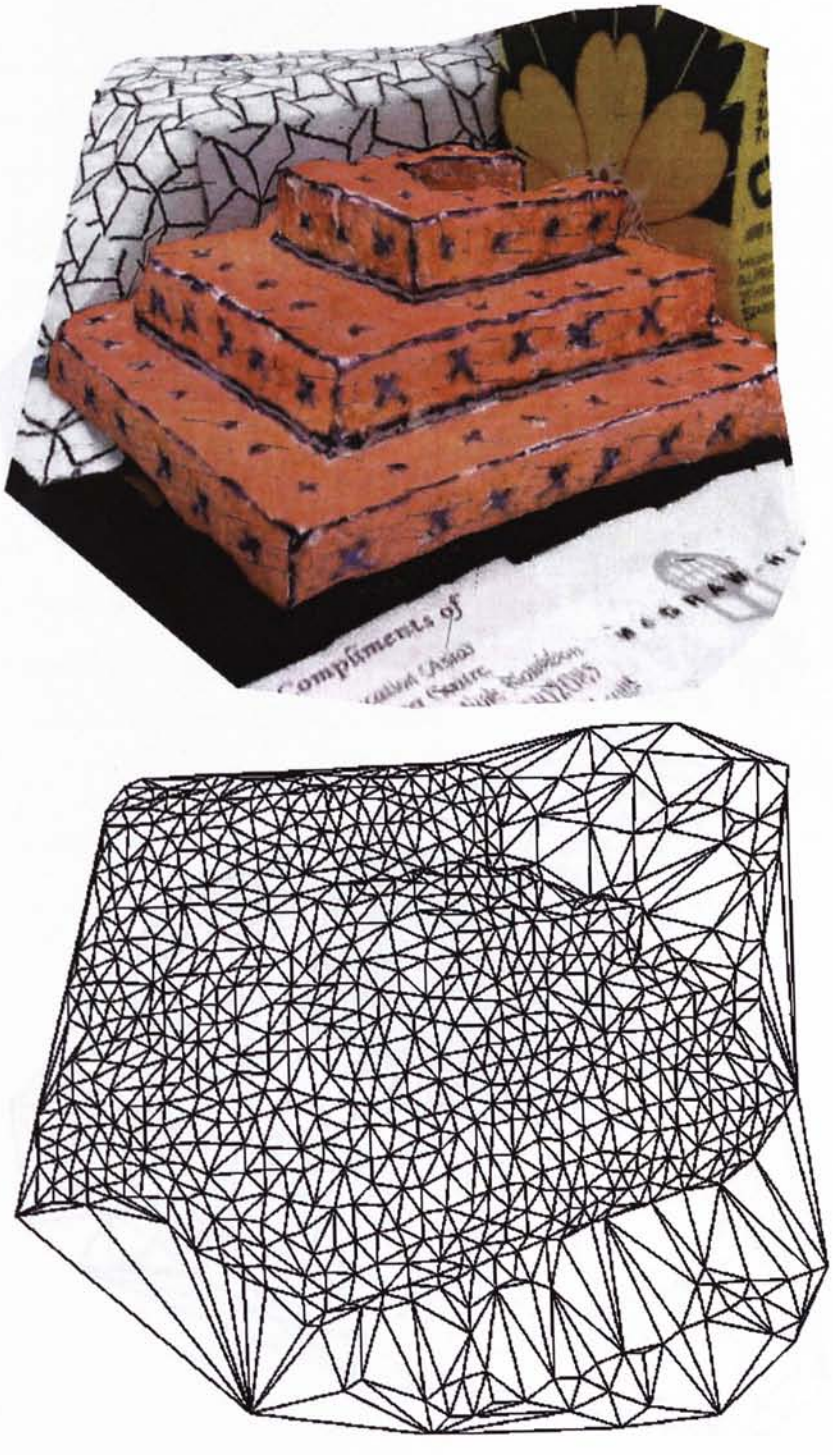


Figure 4.12: The 3D fine model for the "Steps" scene. (Upper) The texture-mapped 3D model and (Lower) the wire-frame 3D model.

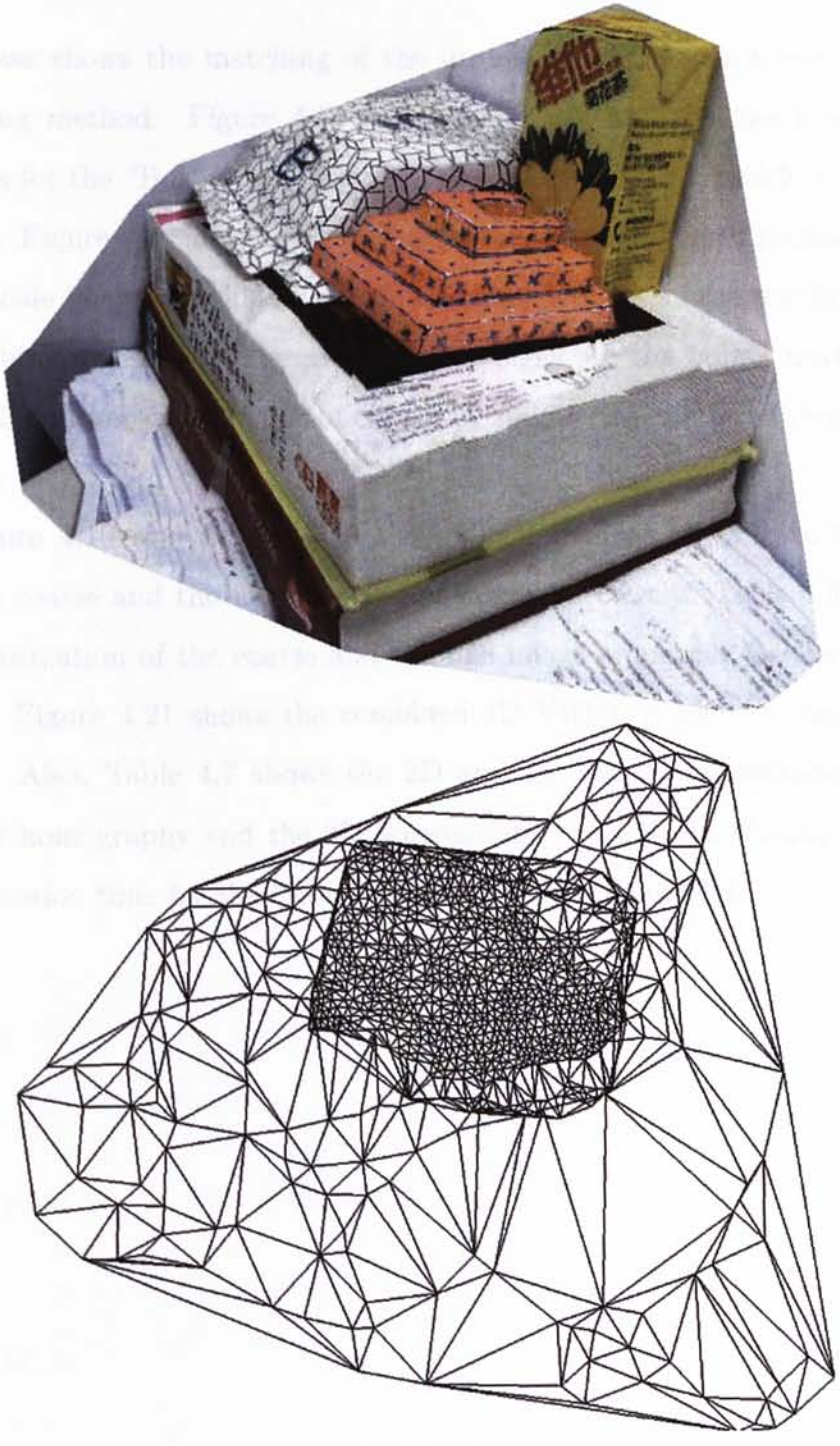


Figure 4.13: The combined 3D VRML model for the "Steps" scene. (Upper) The texture-mapped 3D model and (Lower) the wire-frame 3D model.

"Box" Scene

This case shows the matching of the images using the multi-level image matching method. Figure 4.14 shows the coarse and the fine image sequences for the "Box" scene. Figure 4.15 shows the coarse, middle and fine images. Figure 4.16 and Figure 4.17 show the correct matched points of the multi-scale image matching algorithm. Figure 4.18 shows the results of the rematching algorithm. The magenta plus signs are the points tracked by the KLT tracker and the points circled by yellow circles are the rematched points.

Figure 4.19 and Figure 4.20 show the individual 3D VRML models for the coarse and the fine image sequences respectively. Table 4.6 shows the information of the coarse and the fine image sequences for the "Box" scene. Figure 4.21 shows the combined 3D VRML model for the "Box" scene. Also, Table 4.7 shows the 2D and 3D scales approximated from the 2D homography and the 3D homography respectively. Moreover, the computation time for the "Box" scene is shown in Table 4.8.

Table 4.5: The computation time for the "Steps" scene

Step	**Time (sec)**
KLT tracking for coarse model	41.55
RANSAC for coarse model	0.87
KLT tracking for fine model	40.63
RANSAC for fine model	0.11
3D reconstruction for coarse model	78.39
3D reconstruction for fine model	205.34
Multi-scale image matching	175.81
Rematching	1.61
3D model registration	0.54
VRML modelling for coarse model	27.72
VRML modelling for fine model	126.38
Others	1.46
Total:	700.41

Figure 4.14: Frames for the "Box" scene. 1st, 5th, 10th and 15th frames of (Upper row) the coarse image sequence and (Lower row) the fine image sequence.

coarse middle fine

Figure 4.15: Selected images for the "Box" scene.

Table 4.6: Information of the "Box" scene

Information	**Coarse Sequence**	**Fine Sequence**
No. of frames of the sequence	15	18
No. of requested features	2000	2000
No. of features tracked by KLT	790	837
No. of features selected by RANSAC	595	580
No. of 3D points	595	580
Selected image	6th image	15th image
Width of each image (pixel)	640	640
Height of each image (pixel)	480	480

Table 4.7: The scales approximated for the "Box" scene

Information	**Combined Model**
2D scale approximated between coarse and middle	1.90
2D scale approximated between middle and fine	1.37
3D scale approximated	2.71

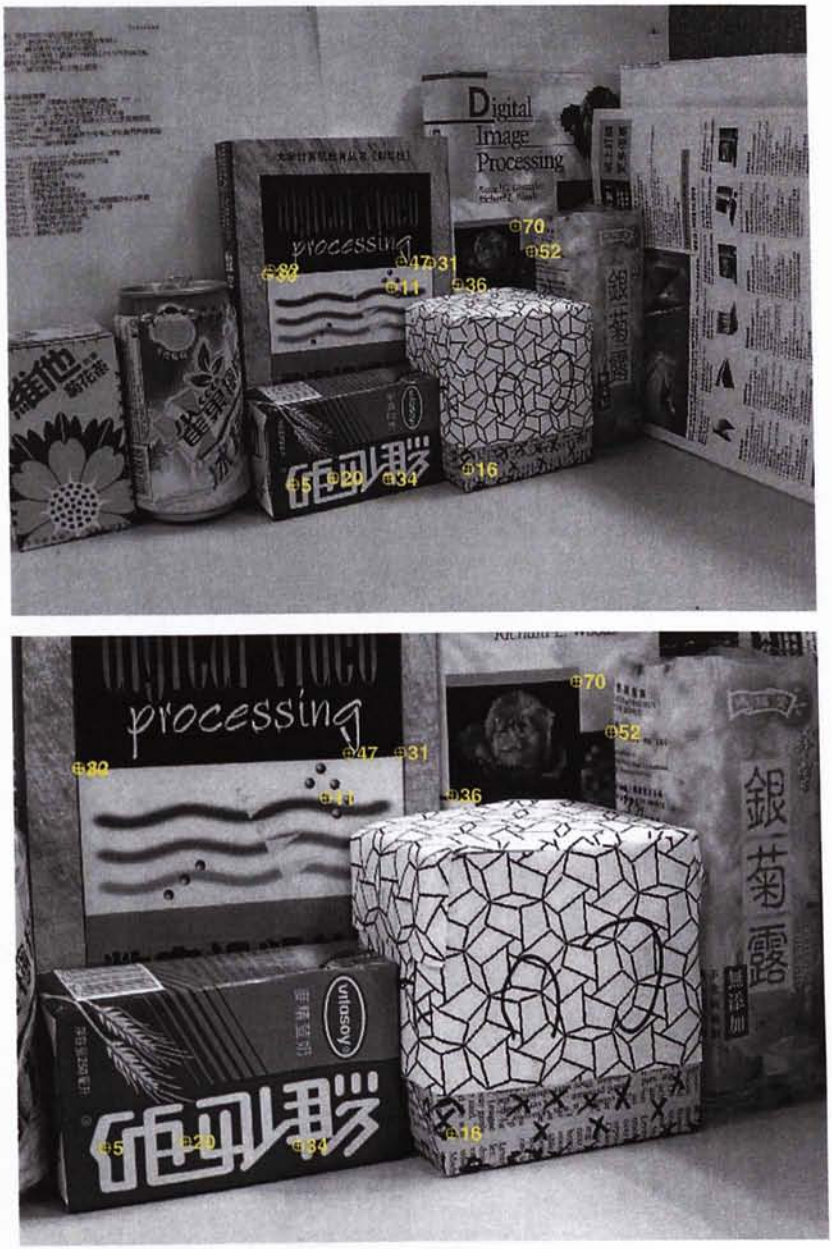


Figure 4.16: The result of the multi-scale image matching for the "Box" scene. (Upper) The coarse image, (Lower) the middle image.

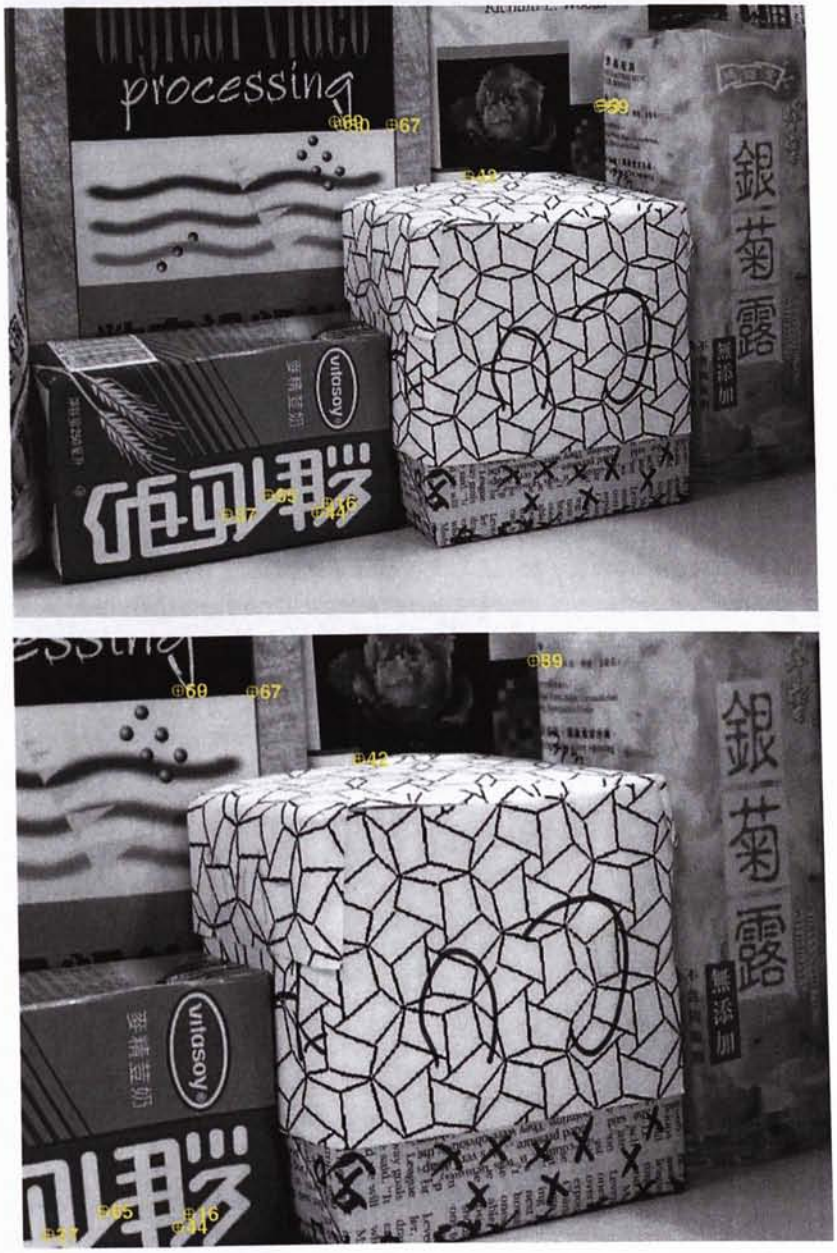


Figure 4.17: The result of the multi-scale image matching for the "Box" scene. (Upper) The middle image, (Lower) the fine image.

Figure 4.18: The result of the rematching algorithm for the "Box" scene. (Upper) The coarse image, (Lower) the fine image. The plus signs are the points tracked by the KLT tracker and the points circled by yellow circles are the rematched points.

Figure 4.19: The 3D coarse model for the "Box" scene. (Upper) The texture-mapped 3D model and (Lower) the wire-frame 3D model.

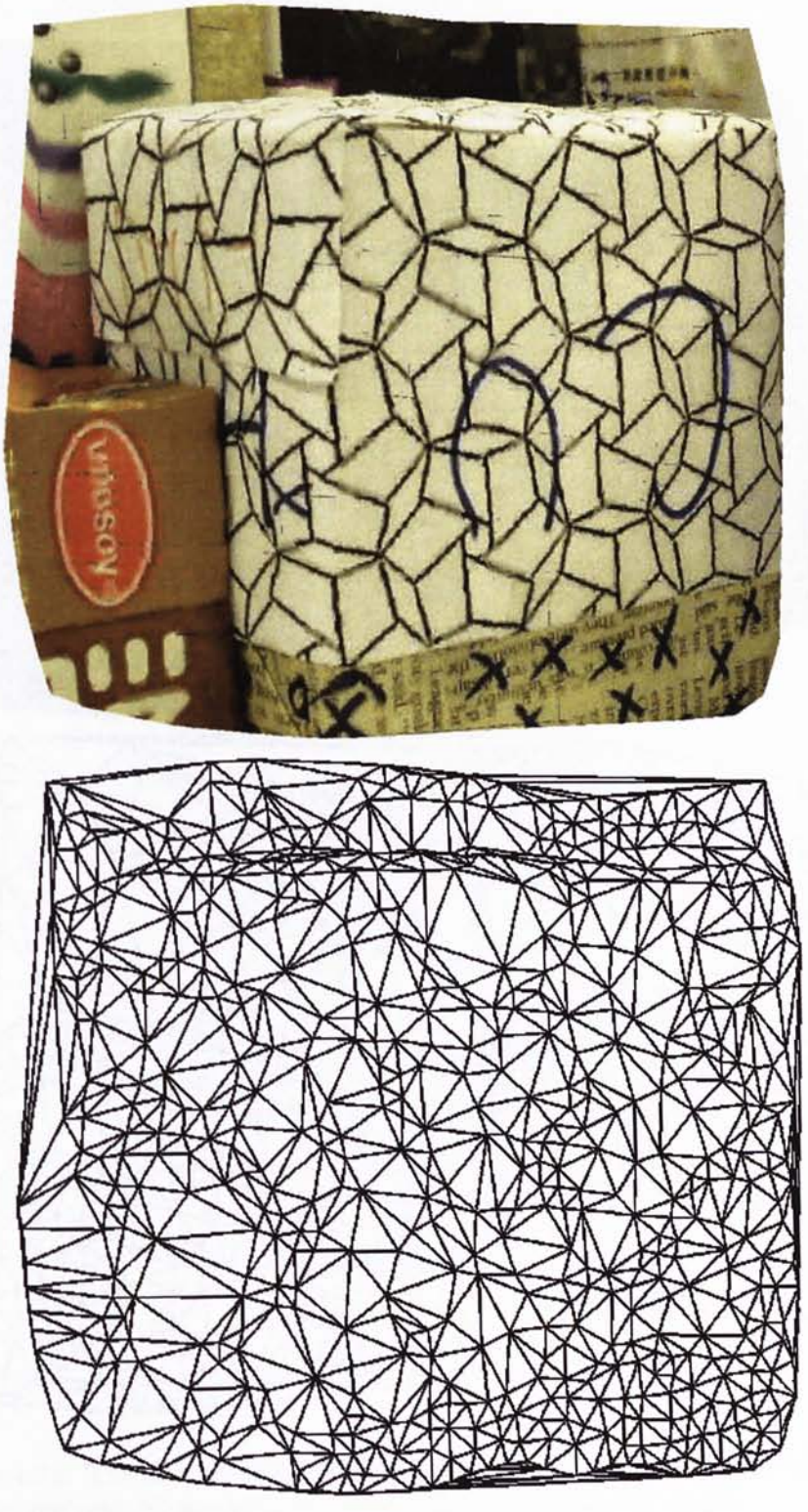


Figure 4.20: The 3D fine model for the "Box" scene. (Upper) The texture-mapped 3D model and (Lower) the wire-frame 3D model.

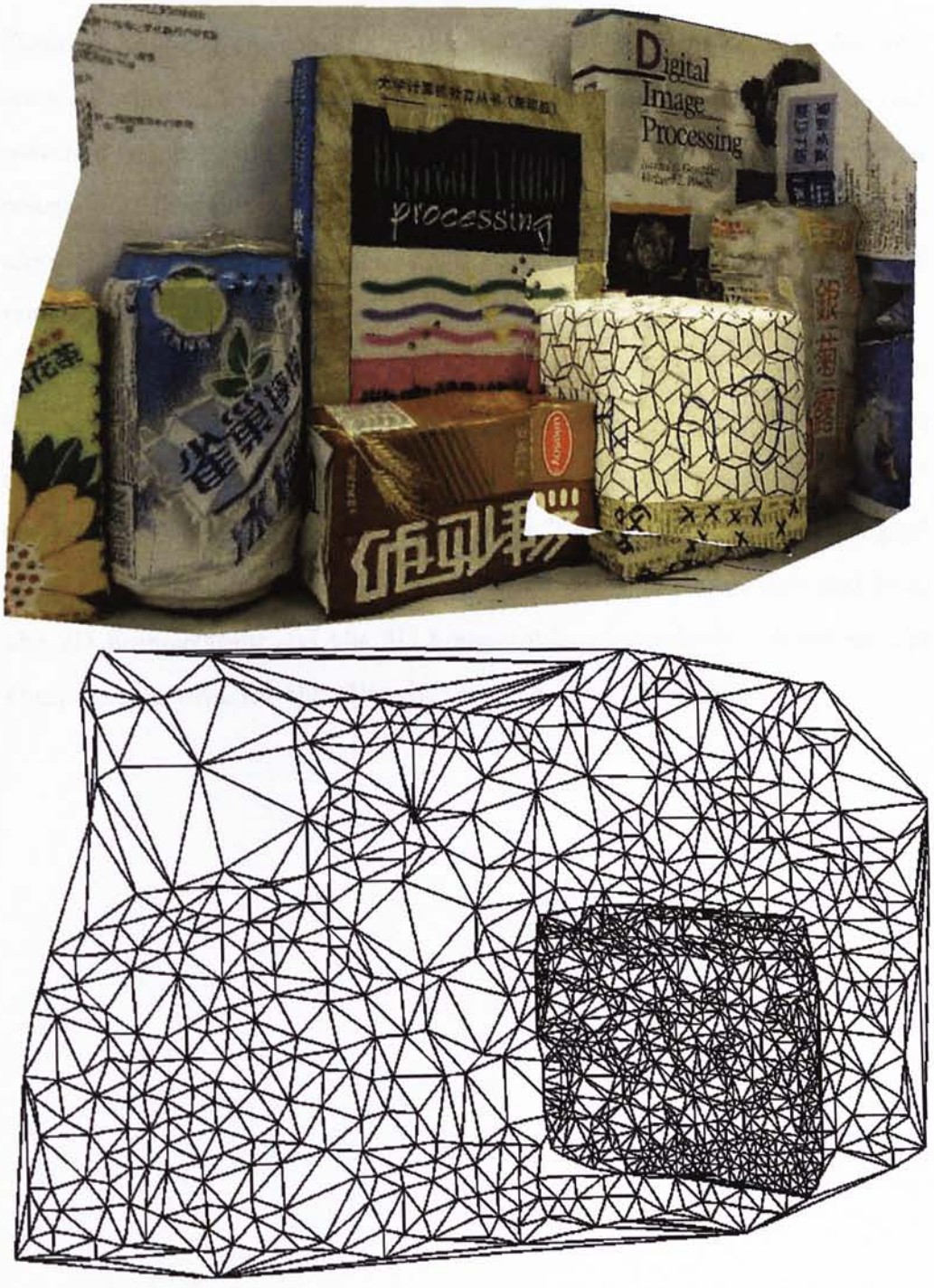


Figure 4.21: The combined 3D VRML model for the "Box" scene. (Upper) The texture-mapped 3D model and (Lower) the wire-frame 3D model.

"Weight" Scene

Figure 4.22 shows the coarse and the fine image sequences for the "Weight" scene. Figure 4.23 shows the selected images. Figure 4.24 shows the correct matched points of the multi-scale image matching algorithm for the selected coarse and fine images. Figure 4.25 shows the results of the rematching algorithm. The magenta plus signs are the points tracked by the KLT tracker and the points circled by yellow circles are the rematched points.

Figure 4.26 and Figure 4.27 show the individual 3D VRML models for the coarse and the fine image sequences respectively. Table 4.9 shows the information of the coarse and the fine image sequences for the "Weight" scene. Figure 4.28 shows the combined 3D VRML model for the "Weight" scene. Also, Table 4.10 shows the 2D and 3D scales approximated from the 2D homography and the 3D homography respectively. Moreover, the computation time for the "Weight" scene is shown in Table 4.11.

Table 4.8: The computation time for the "Box" scene

Step	**Time (sec)**
KLT tracking for coarse model	32.77
RANSAC for coarse model	0.39
KLT tracking for fine model	38.20
RANSAC for fine model	0.75
3D reconstruction for coarse model	126.45
3D reconstruction for fine model	148.08
Multi-scale image matching (Coarse to middle)	193.44
Multi-scale image matching (Middle to fine)	157.11
Rematching	2.04
3D model registration	1.21
VRML modelling for coarse model	76.95
VRML modelling for fine model	90.79
Others	2.55
Total:	870.73

Figure 4.22: Frames for the "Weight" scene. 1st, 4th, 7th and 10th frames of (Upper row) the coarse image sequence and (Lower row) the fine image sequence.

coarse fine

Figure 4.23: Selected images for "Weight" scene.

Table 4.9: Information of the "Weight" scene

Information	**Coarse Sequence**	**Fine Sequence**
No. of frames of the sequence	20	12
No. of requested features	2000	2000
No. of features tracked by KLT	733	538
No. of features selected by RANSAC	481	432
No. of 3D points	481	432
Selected image	11th image	3rd image
Width of each image (pixel)	640	640
Height of each image (pixel)	480	480

Table 4.10: The scales approximated for the "Weight" scene

Information	**Combined Model**
2D scale approximated	1.74
3D scale approximated	1.73

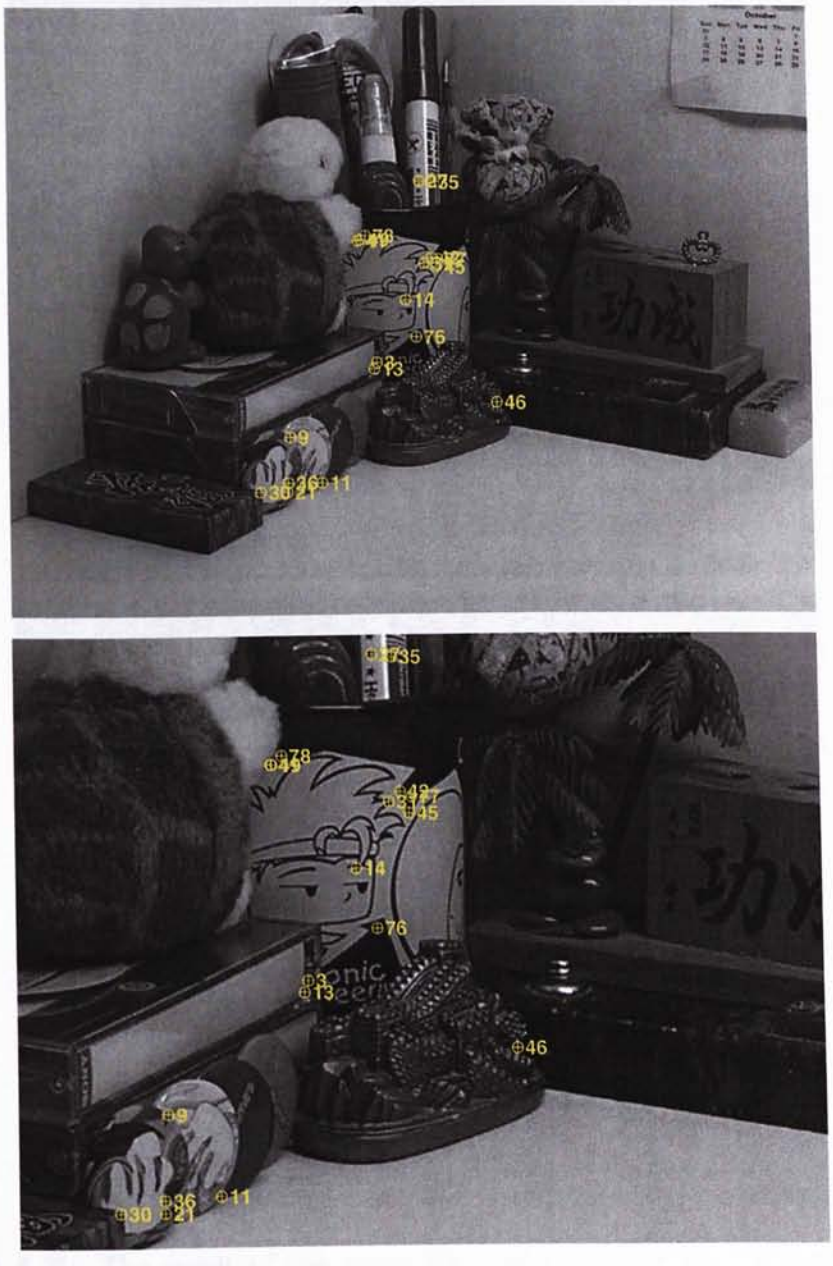


Figure 4.24: The result of the multi-scale image matching for the "Weight" scene. (Upper) The coarse image, (Lower) the fine image.

Figure 4.25: The result of the rematching algorithm for the "Weight" scene. (Upper) The coarse image, (Lower) the fine image. The plus signs are the points tracked by the KLT tracker and the points circled by yellow circles are the rematched points.

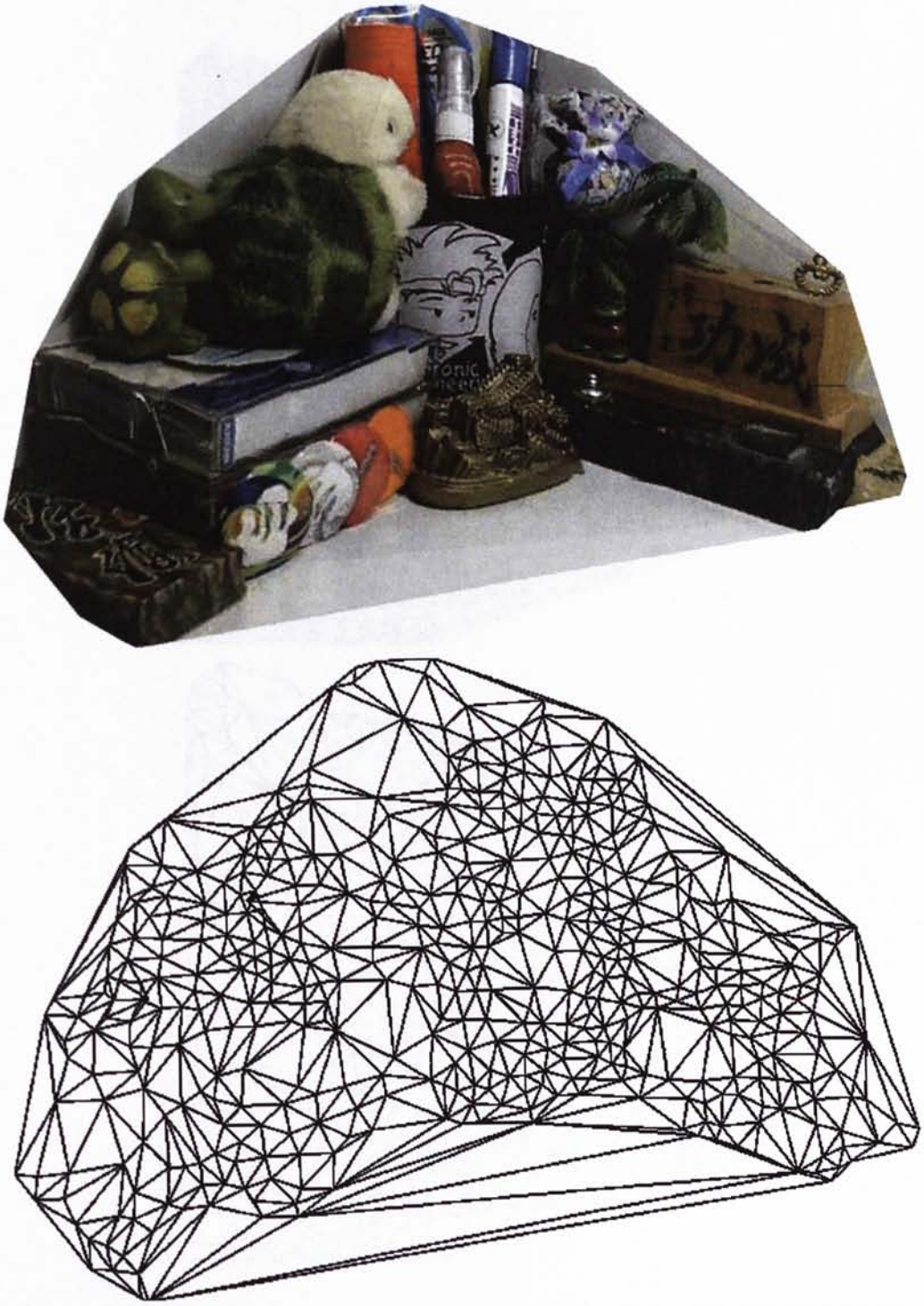


Figure 4.26: The 3D coarse model for the "Weight" scene. (Upper) The texture-mapped 3D model and (Lower) the wire-frame 3D model.

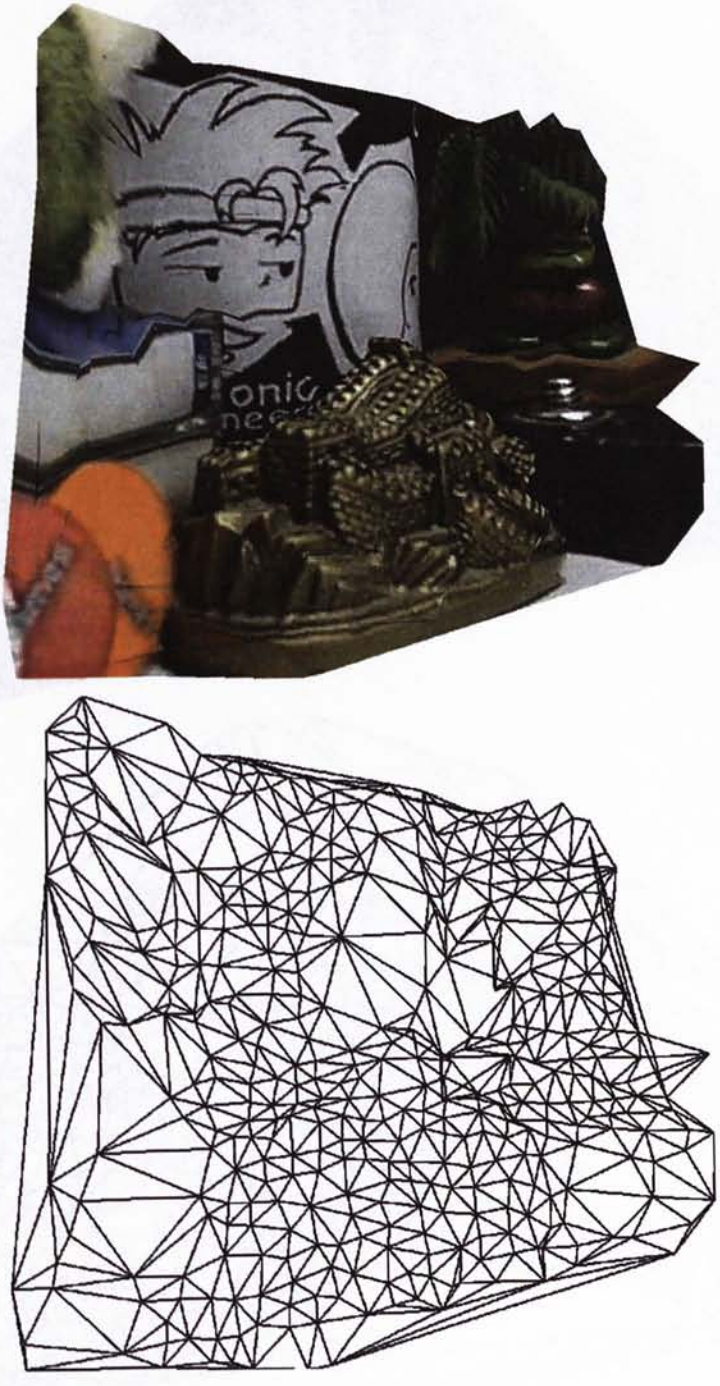


Figure 4.27: The 3D fine model for the "Weight" scene. (Upper) The texture-mapped 3D model and (Lower) the wire-frame 3D model.

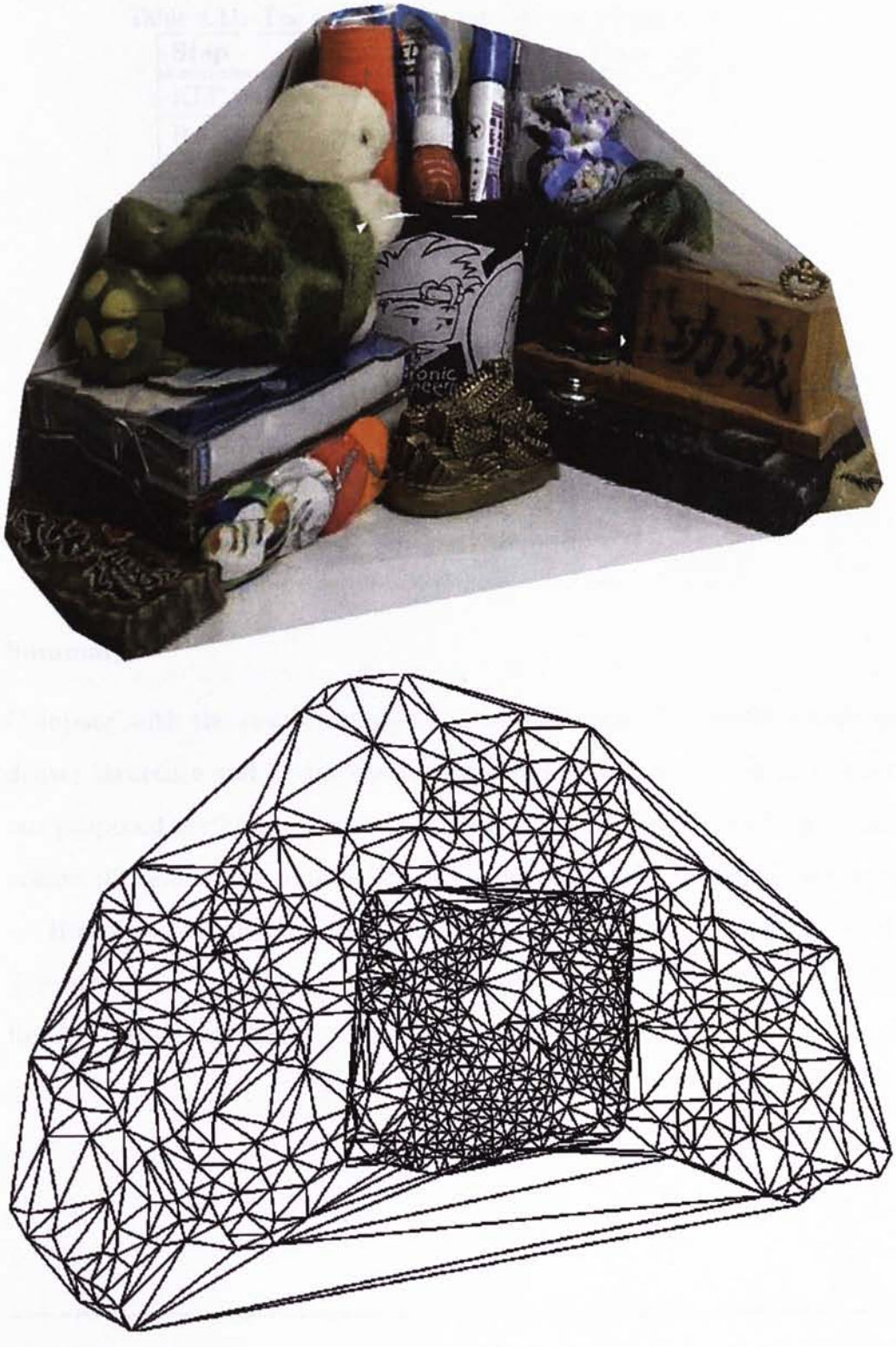


Figure 4.28: The combined 3D VRML model for the "Weight" scene. (Upper) The texture-mapped 3D model and (Lower) the wire-frame 3D model.

Table 4.11: The computation time for the "Weight" scene

Step	Time (sec)
KLT tracking for coarse model	47.75
RANSAC for coarse model	0.57
KLT tracking for fine model	21.88
RANSAC for fine model	0.20
3D reconstruction for coarse model	135.19
3D reconstruction for fine model	74.73
Multi-scale image matching	162.36
Rematching	1.72
3D model registration	0.94
VRML modelling for coarse model	47.69
VRML modelling for fine model	50.49
Others	2.27
Total:	545.79

Summary

Compare with the coarse 3D models, the combined 3D models consist of denser structure and better texture for the fine objects. It can show that our proposed method can register and combine the fine 3D model with the coarse 3D model, and reconstruct the denser 3D model for the whole scene.

However, our method does not handle the merging of the 3D meshes. Therefore, there are splits at the boundaries between the coarse and the fine 3D VRML models.

□ **End of chapter.**

Chapter 5

Conclusion

This thesis proposes an alternative solution to the automatic coarse-to-fine 3D registration problem. The method merges the 3D structures reconstructed from a coarse image sequence for a coarse (or large) scene and a fine image sequence for a finer (or smaller) scene inside a scene of the same environment. The coarse and fine models reconstructed are combined to build a more detailed 3D model for the whole scene. This solution does not require human interaction by giving correspondences between the coarse image sequence and the fine image sequence. It only requires the arrangement of the image sequences, and our system matches the image sequences and integrates the coarse and the fine 3D models automatically.

The method first reconstructs the 3D models for the coarse image sequence and the fine image sequence individually. Subsequently, the linkage between the two image sequences are built up by first matching one image of each sequence using the multi-scale image matching method, and then linking up the feature points of the sequences using the rematching algorithm. Using this linkage, the 3D transformation between the two 3D models is estimated, and finally we can combine the two 3D models and

form the combined 3D VRML model.

Because of using the rematching algorithm to build the linkage between the coarse and the fine image sequences, the image matching step is flexible and is not restricted to one image matching algorithm. A lot of multi-scale image matching algorithms can be used in our framework.

Two synthetic experiments and an experiment on real cases were held to evaluate the method. The first synthetic experiment was used to evaluate the rematching algorithm. It shows that the rematching algorithm accepts small error on the 2D homography estimated by the multi-scale image matching algorithm and small noise on the 2D feature points. But when the error and the noise become larger, the probability that the rematching algorithm successfully rematches the image sequences will drop. Therefore, the multi-scale image matching algorithm used in the proposed method should estimate an accurate 2D homography, and the feature point extraction and tracking algorithm should provide accurate feature points for both image sequences.

The second synthetic experiment was used to compare affine transformation with metric transformation for 3D model registration. The results show that affine transformation is better than metric transformation in registering two 3D models.

The real scene experiment shows that our proposed solution of coarse-to-fine matching and rematching can automatically register the 3D structures of a large scene and the fine object inside the scene. Hence, the 3D model for the large scene with fine details can be recovered.

5.1 Future Work

The following can be the directions of future investigation.

- Many image matching algorithms may be suitable for our method. Different matching algorithms such as [61, 79] can be further investigated in future.

- The rematching algorithm requires an accurate 2D homography; otherwise, the linkage will not be accurate enough for the 3D model registration. A refining algorithm that refines the linkage can be investigated in future.

- The assumption that the selected pair of images have to be taken in similar view angles can be removed by using the affine image matching algorithms [36, 2, 48], and the fundamental matrix instead of 2D homography. But the rematching algorithm has to be updated for the fundamental matrix. The affine image matching algorithms and the new rematching algorithm can be further investigated.

- This project does not deal with the problem of 3D mesh merging. A 3D mesh merging algorithm with better texture morphing that creates a better combined 3D model can be investigated in future.

- The proposed framework can be modified for more applications. One of the applications is the automatic 3D registration for a planar background with non-planar 3D fine objects in the scene. This problem can be further extended to the automatic 3D registration for the 3D structure from the panorama around a fixed camera point for an indoor environment with 3D detailed objects in the scene. The solution

of the manual registration was proposed in [54]. An automatic 3D registration for the problem can be investigated in future.

- The proposed method can be incorporated with the other approaches, such as super-resolution and the approaches using edges and planes, to obtain a finer and more accurate reconstruction with better texture mapped model for large scenes.

□ **End of chapter.**

Appendix A

Camera Parameters

This section describe briefly the camera parameters including intrinsic parameters and extrinsic parameters.

A.1 Intrinsic Parameters

The intrinsic parameters [78, 56] are a set of parameters characterizing the optical, geometric and digital characteristics of the camera. For a perspective (or pinhole) camera, we have the following intrinsic parameters:

1. Focal length of the camera, f.

2. The transformation between camera frame coordinates and pixel coordinates. Let (x_{img}, y_{img}) be the coordinates of an image point in pixel units in image reference frame with the coordinates (x, y) of the same point in the camera reference frame. Then, they have the following relation:

$$\begin{aligned} x &= -(x_{img} - o_x)s_x \\ y &= -(y_{img} - o_y)s_y \end{aligned} \tag{A.1}$$

where (o_x, o_y) are the coordinates in pixel of the image center, and (s_x, s_y) are the effective width and height of the pixel respectively.

3. The geometric distortion introduced by optics. Since the quality of the digital camera nowadays is good, the distortion due to optical effect can be ignored.

For a perspective camera, we can define a matrix M_{int} to include the intrinsic parameters f, (o_x, o_y) and (s_x, s_y) for linear matrix operation.

$$M_{int} = \begin{pmatrix} -\frac{f}{s_x} & 0 & o_x \\ 0 & -\frac{f}{s_x} & o_y \\ 0 & 0 & 1 \end{pmatrix} \tag{A.2}$$

A.2 Extrinsic Parameters

As defined in [78], the extrinsic parameters identity the transformation between the unknown camera reference frame and a known world reference frame. The typical choice of notation describing the transformation includes a 3-dimensional translation vector $\mathbf{T}$, and a 3×3 orthogonal rotation matrix R. $\mathbf{T}$ describes the translation between the origins of the reference frames, and R describes the transformation between the corresponding axes of the two reference frames.

From Figure A.1, suppose there is a transformation $(R, \mathbf{T})$ from the world reference frame to the camera reference frame, the coordinates of a point $\mathbf{P}$ relative to the world reference frame and camera reference frame, $\mathbf{P}_w$ and $\mathbf{P}_c$, respectively, can be related by

$$\mathbf{P}_c = R(\mathbf{P}_w - \mathbf{T}) \tag{A.3}$$

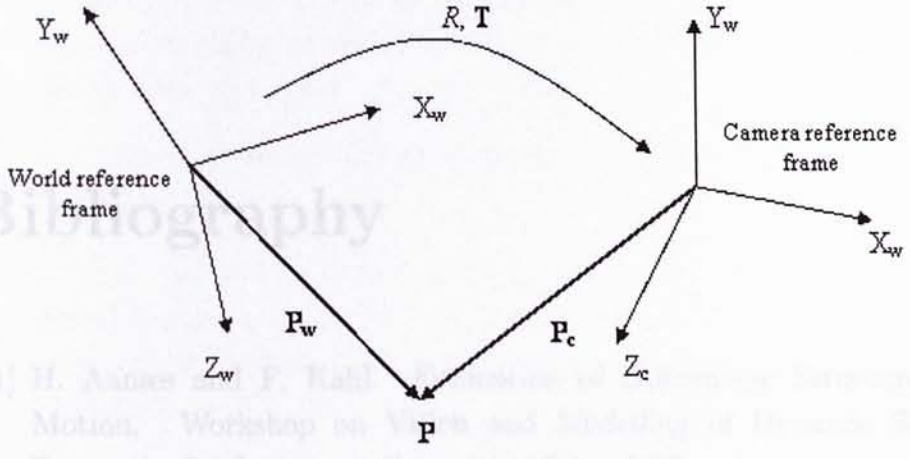


Figure A.1: Relation between camera and world reference frames.

□ **End of chapter.**

Bibliography

[1] H. Aanæs and F. Kahl. Estimation of Deformable Structure and Motion. Workshop on Vision and Modelling of Dynamic Scenes, European Conference on Computer Vision 2002.

[2] A. Baumberg. Reliable Feature Matching Across Widely Separated Views. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 774–781, 2000.

[3] P. Beardsley, A. Zisserman, and D. Murray. Sequential update of projective and affine structure from motion. *International Journal of Computer Vision*, 23(3):235–259, 1997.

[4] P. J. Besl. Active, optical range imaging sensors. *Machine Vision and Applications*, 1(2):127–152, 1988.

[5] D. Blostein and N. Ahuja. Shape from Texture: Integrating Texture-ELement Extraction and Surface Estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(12):1233–1251, December 1989.

[6] E. Boyer and M.-O. Berger. 3D Surface Reconstruction Using Occluding Contours. *Internation Journal of computer Vision*, 22(3):219–233, 1997.

[7] C. Bregler, A. Hertzmann, and H. Biermann. Recovering Non-Rigid 3D Shape from Image Streams. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 690–696, 2000.

[8] D. Capel and A. Zisserman. Computer Vision Applied to Super-resolution. *IEEE Signal Processing Magazine*, 20(3):75–86, May 2003.

[9] M. M.-Y. Chang and K.-H. Wong. Model reconstruction and pose acquisition using extended lowe's method. *IEEE Transactions on Multimedia*. to appear.

[10] S. Chaudhuri, editor. *Super-Resolution Imaging*. Kluwer Academic Publisher, 2001.

[11] L. D. Cohen and S. Vinson. Segmentation of Complex Buildings from Aerial Images and 3D Surface Reconstruction. In *Proceedings of the Sixth IEEE Workshop on Applications of Computer Vision*, pages 215–219, 2002.

[12] Y. Dufournaud, C. Schmid, and R. Horaud. Matching Images with Different Resolutions. In *Proceedings of the Conference on Computer Vision and Pattern Recognitio*, pages 612–618. IEEE Computer Society Press, 2000.

[13] Y. Dufournaud, C. Schmid, and R. Horaud. Image matching with scale adjustment. Technical report, INRIA, 2002.

[14] S. F. El-Hakim. Semi-automatic 3D reconstruction of occluded and unmarked surfaces from widely separated views. In *Proceedings of ISPRS Commission V Symposium*, pages 143–148, Corfu, Greece, September 2002.

[15] S. F. El-Hakim, J.-A. Beraldin, and M. Picard. Detailed 3D Reconstruction of monuments using multiple techniques. In *Proceedings of the International Workshop on Scanning for Cultural Heritage Recording - Complementing or Replacing Photogrammetry*, pages 13–18, Corfu, Greece, September 2002.

[16] M. A. Fischler and R. C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, June 1981.

[17] A. W. Fitzgibbon, G. Cross, and A. Zisserman. Automatic 3D Model Construction for Turn-Table Sequences. In *Proceedings of the European Workshop on 3D Structure from Multiple Images of Large-Scale Environments*, pages 155–170, 1998.

[18] A. W. Fitzgibbon and A. Zisserman. Automatic Camera Recovery for Closed or Open Image Sequences. In *Proceedings of 5th European Conference on Computer Vision*, volume 1, pages 311–326, 1998.

[19] J. Fujiki and T. Kurata. Recursive Factorization Method for the Paraperspective Model based on the Perspective Projection. In *Proceedings of International Conference on Pattern Recognition*, pages 406–410, 2000.

[20] R. C. Gonzalez and R. E. Woods. *Digital image processing.* Addison-Wesley Publishing Company, 1992.

[21] M. Han and T. Kanade. Creating 3D Models with Uncalibrated Cameras. In *Proceedings of the IEEE Computer Society Workshop on the Application of Computer Vision*, pages 178–185, December 2000.

[22] M. Han and T. Kanade. Reconstruction of a Scene with Multiple Linearly Moving Objects. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 2542–2549, June 2000.

[23] M. Han and T. Kanade. Multiple motion scene reconstruction from uncalibrated views. In *Proceedings of the Eighth IEEE International Conference on Computer Vision*, volume 1, pages 163 – 170, July 2001.

[24] M. Han and T. Kanade. Multiple Motion Scene Reconstruction with Uncalibrated Cameras. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(7):884–894, July 2003.

[25] C. Harris and M. Stephens. A combined corner and edge detector. In *Proceedings of The Fourth Alvey Vision Conference*, pages 147–151, 1988.

[26] R. I. Hartley. In Defence of the 8-Point Algorithm. In *Proceedings of 5th International Conference on Computer Vision*, pages 1064–1070, Cambridge (MA), 1995.

[27] R. I. Hartley and P. Sturm. Triangulation. *Computer Vision and Image Understanding: CVIU*, 68(2):146–157, 1997.

[28] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision.* Cambridge University Press, ISBN: 0521623049, 2000.

[29] S. B. Kang and R. Szeliski. 3-D Scene Data Recovery Using Omnidirectional Multibaseline Stereo. *International Journal of Computer Vision*, 25(2):167–183, 1997.

[30] R. Koch, M. Pollefeys, and L. J. V. Gool. Realistic surface reconstruction of 3D scenes from uncalibrated image sequences. *Journal of Visualization and Computer Animation*, 11(3):115–127, 2000.

[31] S. C. Lee, S. K. Jung, , and R. Nevatia. Automatic Pose Estimation of Complex 3D Building Models. In *Proceedings of the Sixth IEEE Workshop on Applications of Computer Vision*, pages 148–152, 2002.

[32] S. J. Leon. *Linear Algebra with Applications.* Prentice Hall, 1998.

[33] Y. Li and M. J. Brooks. An Efficient Recursive Factorization Method for Determining Structure from Motion. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 138–143, 1999.

[34] T. Lindeberg. Feature detection with automatic scale selection. *International Journal of Computer Vision*, 30(2):79–116, 1998.

[35] T. Lindeberg. Principles for automatic scale selection. Technical report, CVAP, Department of Numerical Analysis and Computer Science, Royal Institute of Technology, 1998.

[36] T. Lindeberg and J. Garding. Shape-Adapted smoothing in estimation of 3-D depth cues from affine distortions of local 2-D brightness structure. In *Proceedings of European Conference on Computer Vision*, pages 389–400, 1994.

[37] H. Longuet-Higgins. A Computer Algorithm for Reconstructing a Scene from Two Projections. *Nature*, 293(10):133–135, 1981.

[38] D. G. Lowe. Three-Dimensional Object Recognition from Single Two-Dimensional Images. *Artificial Intelligence*, 31(3):355–395, 1987.

[39] D. G. Lowe. Fitting Parameterized Three-Dimensional Models to Images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-13(5):441–450, 1991.

[40] D. G. Lowe. Object Recognition from Local Scale-Invariant Features. In *Proceedings of IEEE International Conference on Computer Vision*, pages 1150–1157, 1999.

[41] B. D. Lucas and T. Kanade. An Iterative Image Registration Technique with an Application to Stereo Vision. In *Proceedings of International Joint Conference on Artificial Intelligence*, pages 674–679, 1981.

[42] Q. Luong and O. Faugeras. The Fundamental Matrix: Theory, Algorithms, and Stability Analysis. *International Journal of Computer Vision*, 17:43–75, 1995.

[43] S. Mahamud and M. Hebert. Iterative projective reconstruction from multiple views. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 430 – 437, June 2000.

[44] D. Marr. *Vision : a computational investigation into the human representation and processing of visual information.* W.H. Freeman, 1982.

[45] Matlab. http://www.mathworks.com/.

[46] P. R. S. Mendonça, K.-Y. K. Wong, and R. Cipolla. Camera Pose Estimation and Reconstruction from Image Profiles under Circular Motion. In *Proceedings of European Conference on Computer Vision*, volume II, pages 864–877, Dublin, Ireland, June 2000.

[47] K. Mikolajczyk and C. Schmid. Indexing based on scale invariant interest point. In *IEEE International Conference on Computer Vision*, pages 525–531, 2001.

[48] K. Mikolajczyk and C. Schmid. An affine invariant interest point detector. In *Proceedings of European Conference on Computer Vision*, pages 128–142, 2002.

[49] K. Mikolajczyk and C. Schmid. A performance evaluation of local descriptors. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2, pages 257–263, 2003.

[50] R. Mohr and B. Triggs. Projective geometry for image analysis. Tutorial given at International Symposium Photogrammetry and Remote Sensing, July 1996.

[51] P. Montesinos and V. Gouet. Matching Color Uncalibrated Images using Differential Invariants. Technical report, INRIA, 1999.

[52] P. Montesinos, V. Gouet, R. Deriche, and D. Pelé. Differential Invariants for Color Images. Technical report, INRIA, 1998.

[53] T. Morita and T. Kanade. A Sequential Factorization Method for Recovering Shape and Motion from Image Streams. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(8):858–867, 1997.

[54] S. H. Or, K. H. Wong, M. M. Y. Chang, and C. Y. Ip. Large scene reconstruction with local details recovery. In *International conference on pattern recognition*, August 2004.

[55] C. J. Poelman and T. Kanade. A Paraperspective Factorization Method for Shape and Motion Recovery. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(3):206–218, 1997.

[56] M. Pollefeys. Tutorial on 3D Modeling from Images, 2000.

[57] M. Pollefeys, R. Koch, and L. J. V. Gool. Self-Calibration and Metric Reconstruction in Spite of Varying and Unknown Internal Camera Parameters. In *Proceedings of Sixth International Conference on Computer Vision*, pages 90–95, 1998.

[58] S. Ramalingam and S. K. Lodha. Adaptive Enhancement of 3D Scenes using Hierarchical Registration of Texture-Mapped 3D models. In *Proceedings of the Fourth International Conference on 3-D Digital Imaging and Modeling*, pages 203–210, October 2003.

[59] M. Rioux, G. Bechthold, D. Taylor, and M. Duggan. Design of a large depth of view three-dimensional camera for robot vision. *Optical Engineering*, 26(12):1245–1250, December 1987.

[60] G. X. Ritter and J. N. Wilson. *Handbook of computer vision algorithms in image algebra.* CRC Press LLC, 2nd edition, 2000.

[61] F. Schaffalitzky and A. Zisserman. Viewpoint invariant texture matching and wide baseline stereo. In *Proceedings of the Internation Conference on Computer Vision*, volume 2, pages 636–643, 2001.

[62] C. Schmid and R. Mohr. Local Grayvalue Invariants for Image Retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(5):530–535, 1997.

[63] J. Shi and C. Tomasi. Good features to track. In *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 593–600, 1994.

[64] H.-Y. Shum, Q. Ke, and Z. Zhang. Efficient Bundle Adjustment with Virtual Key Frames: A Hierarchical Approach to Multi-Frame Structure from Motion. In *Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition*, June 1999.

[65] C. Silpa-Anan. Midterm report: A visual map for a robot. Technical report, The Australian National University, 2003.

[66] T. H. R. H. Simon Gibson, Jon Cook and D. Oram. Accurate Camera Calibration for Off-line, Video-Based Augmented Reality. In *Proceedings of International Symposium on Mixed and Augmented Reality*, pages 37–46, 2002.

[67] G. Slabaugh, B. Culbertson, T. Malzbender, and R. Schafer. A Survey of Methods for Volumetric Scene Reconstruction from Photographs. In *Proceedings of the Joint IEEE TCVG and Eurographics Workshop (VolumeGraphics-01)*, pages 81–100. Springer-Verlag, June 2001.

[68] S. Soatto and R. Brocket. Optimal and Suboptimal Structure from Motion. Technical report, Harvard University, 1997.

[69] G. Strang. *Introduction to Linear Algebra.* Wellesley-Cambridge Press, 1998.

[70] P. Sturm and B. Triggs. A Factorization Based Algorithm for Multi-Image Projective Structure and Motion. In *Proceedings of 4th European Conference on Computer Vision*, volume 2, pages 709–720, 1996.

[71] R. Szeliski. Rapid Octree Construction from Image Sequences. *CVGIP: Image Understanding*, 58(1):23–32, July 1993.

[72] W. T. Freeman and E. H. Adelson. The design and use of steerable filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(9):891–906, September 1991.

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

[74] C. Tomasi and T. Kanade. Shape and motion from image streams – a factorization method. *International Journal of Computer Vision*, 9(2):137–154, 1992.

[75] P. Torr, A. Zisserman, and S. Maybank. Robust detection of degenerate configurations for the fundamental matrix. In *Proceedings of International Conference on Computer Vision*, pages 1037–1042, 1995.

[76] L. Torresani, D. B. Yang, E. J. Alexander, and C. Bregler. Tracking and Modeling Non-Rigid Objects with Rank Constraints. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 493–500, 2001.

[77] B. Triggs, P. McLauchlan, R. Hartley, and A. Fitzgibbon. Bundle adjustment – A modern synthesis. In W. Triggs, A. Zisserman, and R. Szeliski, editors, *Vision Algorithms: Theory and Practice*, LNCS, pages 298–375. Springer Verlag, 2000.

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

[79] T. Tuytelaars and L. V. Gool. Wide baseline stereo matching based on local, affinely invariant regions. In *Proceedings for the British Machine Vision Conference*, 2000.

[80] D. Weinshall and C. Tomasi. Linear and Incremental Acquisition of Invariant Shape Models From Image Sequences. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(5):512–517, 1995.

[81] T. Werner and A. Zisserman. New Techniques for Automated Architectural Reconstruction from Photographs. In *Proceedings of European Conference on Computer Vision*, volume 2, pages 541–555, 2002.

[82] K.-Y. K. Wong and R. Cipolla. Structure and Motion from Silhouettes. In *Proceedings of 8th IEEE International Conference on Computer Vision*, volume II, pages 217–222, Vancouver, Canada, July 2001.