MULTIVIEW 3D RECONSTRUCTION OF A SCENE CONTAINING INDEPENDENTLY MOVING OBJECTS

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

ENGİN TOLA

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
ELECTRICAL AND ELECTRONICS ENGINEERING

AUGUST 2005

Approval of the Graduate School of Natural and Applied Sciences

_____
Prof. Dr. Canan Özgen
Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of Master of Science.

_____
Prof. Dr. İsmet Erkmen
Head of Department

This is to certify that we have read this thesis and that in our opinion it is fully adequate, in scope and quality, as a thesis for the degree of Master of Science.

_____
Assoc. Prof. Dr. A. Aydın Alatan
Supervisor

Examining Committee Members

Prof.Dr. Kemal Leblebicioğlu (METU,EE) _____

Assoc. Prof. Dr. A. Aydın Alatan (METU,EE) _____

Assoc. Prof. Dr. Gözde Bozdağı Akar (METU,EE) _____

Dr. İlkay Ulusoy (METU,EE) _____

Prof. Dr. Levent Onural (BILKENT UNI.,EE) _____

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Engin Tola

ABSTRACT

MULTIVIEW 3D RECONSTRUCTION OF A SCENE
CONTAINING
INDEPENDENTLY MOVING OBJECTS

Tola, Engin
M.S., Department of Electrical and Electronics Engineering
Supervisor: Assoc. Prof. Dr. A. Aydın Alatan

August 2005, 156 Pages

In this thesis, the structure from motion problem for calibrated scenes containing independently moving objects (IMO) has been studied. For this purpose, the overall reconstruction process is partitioned into various stages. The first stage deals with the fundamental problem of estimating structure and motion by using only two views. This process starts with finding some salient features using a sub-pixel version of the Harris corner detector. The features are matched by the help of a similarity and neighborhood-based matcher. In order to reject the outliers and estimate the fundamental matrix of the two images, a robust estimation is performed via RANSAC and normalized 8-point algorithms. Two-view reconstruction is finalized by decomposing the fundamental matrix and estimating the 3D-point locations as a result of triangulation. The second stage of the reconstruction is the generalization of the two-view algorithm for the N-view case.

This goal is accomplished by first reconstructing an initial framework from the first stage and then relating the additional views by finding correspondences between the new view and already reconstructed views. In this way, 3D-2D projection pairs are determined and the projection matrix of this new view is estimated by using a robust procedure. The final section deals with scenes containing IMOs. In order to reject the correspondences due to moving objects, parallax-based rigidity constraint is used. In utilizing this constraint, an automatic background pixel selection algorithm is developed and an IMO rejection algorithm is also proposed. The results of the proposed algorithm are compared against that of a robust outlier rejection algorithm and found to be quite promising in terms of execution time vs. reconstruction quality.

Keywords: 3D Scene Reconstruction, Independently Moving Objects, Robust Estimation.

ÖZ

BAĞIMSIZ OLARAK HAREKET EDEN NESNELER İÇEREN BİR SAHNENİN ÇOKLU RESİMLERDEN 3 BOYUTLU SAHNE YAPISININ ÇIKARILMASI

Tola, Engin
Yüksek Lisans, Elektrik ve Elektronik Mühendisliği Bölümü
Tez Yöneticisi: Doç. Dr. A. Aydın Alatan

Ağustos 2005, 156 Sayfa

Bu tezde bağımsız hareket eden nesneler içeren kalibre edilmemiş sahnelerdeki hareketten yapı problemleri incelenmektedir. Bu amaçla geriçatım süreci üç aşamaya bölünmüştür. Birinci kısım, 3-B yapı ve hareketi sadece iki resim kullanarak tahmin etme problemidir. Bu süreç, Harris köşe bulucusunun piksel-altı uyarlaması kullanılarak, gürbüz özelliklerin bulunmasıyla başlar. Bu özellikler benzerlik ve komşuluk özellikleri temelli bir eşleyiciyle ilişkilendirilirler. Aykırı örnekleri atmak ve temel (*fundamental*) matrisi hesaplayabilmek için RANSAC ve normalleştirilmiş 8-nokta algoritmaları kullanılarak, gürbüz bir kestirim uygulanır. İki görüntüden geriçatma, temel matrisi parçalarına ayırma ve 3B noktaların yerlerinin, üçgenleştirme kullanılarak bulunmasıyla sonuçlandırılır. Geriçatmanın ikinci aşaması, iki görüntü için elde edilmiş olan algoritmanın N-görüntü için genelleştirilmesidir. Bu amaca, ilk olarak birinci aşamadaki algoritma kullanılarak

başlangıç iskeletinin kurulması ve ilave görüntülerin daha önceden iskelete katılmış görüntülerle ilişkisini elde edilmesiyle, ulaşılır. Bu şekilde, 3B-2B izdüşüm noktaları elde edilir ve bu noktalardan, gürbüz bir işlemle yeni görüntünün izdüşüm matrisi hesaplanır. Son bölüm, bağımsız hareket nesneler içeren sahnelerde geriçatma ile ilişkilidir. Hareketli nesneleri atmak için paralaks temelli katılık sınırı kullanılmaktadır. Bu sınırı kullanmak için, otomatik bir arkaplan piksel seçici algoritma geliştirilmiş ve bu sınıra dayanan bir bağımsız nesneleri çıkartma algortiması önerilmiştir. Önerilen algoritmanın sonuçları gürbüz bir aykırı örnek eleme algoritmasıyla kıyaslanmıştır ve sonuçlar işlem zamanı-yapılandırma kalitesi açısından oldukça ümit verici bulunmuştur.

Anahtar Kelimeler: 3B Sahne yapılandırması, Bağımsız Hareket Eden Nesneler, Gürbüz Tahmin

ACKNOWLEDGEMENTS

I would like to express my gratitude and appreciation to my supervisor Assoc. Prof. Dr. Aydın Alatan for his guidance, and stimulations and also for the great research environment he had provided.

I would like to express my thanks to my friend Birant Örten with whom we have started together and came up to this point. His support and assistance was invaluable. I will surely miss working with him.

Finally, I would like to thank my family for their understanding, support and patience; especially to my father and mother.

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

In recent years, due to significant amount of devoted resources, there had been a lot of progress in 3-D display technologies. Publicly unpopular glass-based 3-D visualization solutions are currently being replaced with their glass-free counterparts, which are auto-stereoscopic displays. It is now possible to purchase an auto-stereoscopic display for a reasonable price and hence, the manufacturers are producing stereo displays for not only the professional applications, but also the consumer market. However, the content, which can be viewed by using these devices, is not vastly available. Hence, 3-D visualization is still only privileged to the researchers and professionals. It should be noted that in order to produce content, it is also possible to capture new data, which is compatible with these devices, by the help of some extra hardware, such as stereo cameras or LIDAR devices. Obviously, it will be a waste of resources, if one does not use the 3-D information which is available in a typical mono-view camera recording. Apart from this fact, it should also be remembered that for many years, mankind has already collected images and videos via mono-view cameras. Instead of re-capturing new data or losing already available content, such information sources should be converted into the appropriate format for such 3-D displays.

The discipline that relates image formation to 3-D scene structure is a very exciting branch of study and it has attracted much attention over the years and as a result, a new field of study, called as *computer vision*, has emerged. Vision researchers are working on algorithms to estimate 3-D information by using only images or single camera shots for the past 20 years. Currently, the evolved algorithms are mature enough to give good representations of the scenes without requiring much human intervention.

1.1 Scope of the Thesis

This thesis is devoted to the problem of developing the fundamental building blocks of a complete 3-D scene reconstruction system that operates on calibrated image sequences, which might also contain independently moving objects, as well as the stationary background. After processing of the mono-view in a cascaded set of algorithms, the system finally produces a 3-D sparse (Virtual Reality Modeling Language, VRML) model of the scene for visualization purposes.

In this thesis, as well as a complete 3-D scene reconstruction system, different triangulation algorithms, which are quite critical while locating the 3D points in space, are also compared and a novel algorithm to reject the independently moving objects within the scene is proposed. The outputs for two different outlier rejection techniques are evaluated and some hypotheses are validated through simulations.

1.2 Outline of the Thesis

In Chapter 2, some background information is given about camera models and the epipolar geometry.

Chapter 3 is devoted to the basic building blocks of the 3D reconstruction algorithm from two calibrated images. These blocks include correspondence estimation, robust computation of the fundamental matrix, computation of the relative pose and orientation between the views and triangulation. Different methods for triangulation are presented and the chapter ends with some simulation results.

Chapter 4 discusses the generalization of the two-view reconstruction to the multiple views. The presented algorithm starts with an initial framework and each new frame is inserted into the system, sequentially. Finally, the whole structure is refined through a general bundle adjustment.

Chapter 5 considers the multiple view reconstruction problem with independently moving objects within the scene. A novel algorithm is presented for this purpose and these results are compared with that of the sequential algorithm, given in Chapter 4.

Finally, Chapter 6 gives a summary of the thesis and concluding remarks about certain blocks of the algorithm. Some future work plan is also suggested in this last chapter.

CHAPTER 2

CAMERA MODEL AND EPIPOLAR GEOMETRY

In this chapter, some background information, which is necessary to better understand the developed procedures and analyze the presented material, is discussed briefly. The chapter contains some information about the camera models and the epipolar geometry. Most of the following definitions follow the text in [1-2] and hence, the reader should refer to these resources for more detail.

2.1 Camera Model

A camera model is a simple transformation that relates the 3-D world coordinate system and a 2-D image plane in order to simulate the imaging process of an optical camera. This transformation is usually represented in matrix form and when the projection is considered over points, the matrix is a 3x4 matrix, called *Projection Matrix* (P), which maps homogeneous 3-D world coordinates to homogeneous 2-D image plane coordinates. The projection matrix encapsulates information about the *intrinsic parameters* of the camera, such as *focal length* and *principal point*, as well as the *extrinsic parameters,* rotation and transformation.

Throughout this thesis, finite projective camera model is assumed and hence, in this chapter, basic definitions of this camera model will be introduced, starting from a simple model and generalizing it by adding degradations. Then, a nonlinear distortion of the camera lens will be taken into account and explained, briefly. Finally, *camera calibration,* which is a procedure to estimate the parameters of the camera matrix, will be outlined and a popular algorithm to easily accomplish this task will be presented.

2.1.1 Finite Camera Model

In this section, the most basic camera model, *pinhole camera model,* is explained and more general models are also introduced by considering imperfections for this model.

2.1.1.1 Basic pinhole model

Basic pinhole camera model (see Figure 2.1) assumes that a 3-D point in space is projected onto the image plane by drawing a line from the 3-D point to the center of projection.

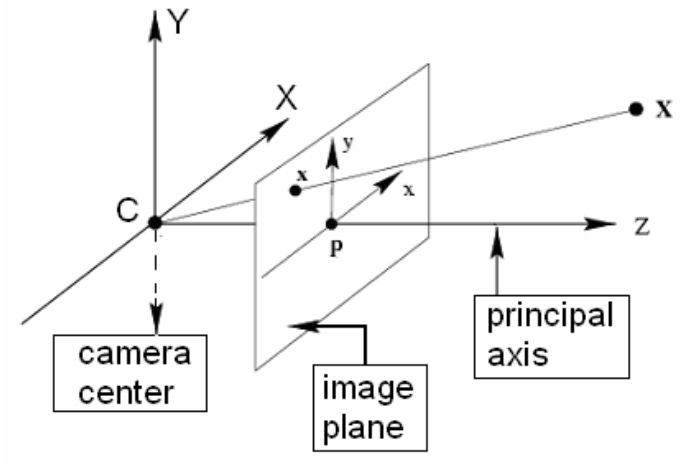


Figure 2.1: Basic pinhole camera geometry

The intersection of this line with the image plane is the point of projection. The projection operation is shown in Figure 2.2. ***f*** is the *focal length*, ***P*** is the *principal point,* ***X*** is a 3-D point and ***x*** is the projection of ***X***. The center of the projection is called as the *camera center* and it is also known as the *optical center*. The ray, which is perpendicular to the image plane, passing through the camera center, is called *principle axis*. Lastly, the point of intersection of this ray with the image plane is known as *principal point*.

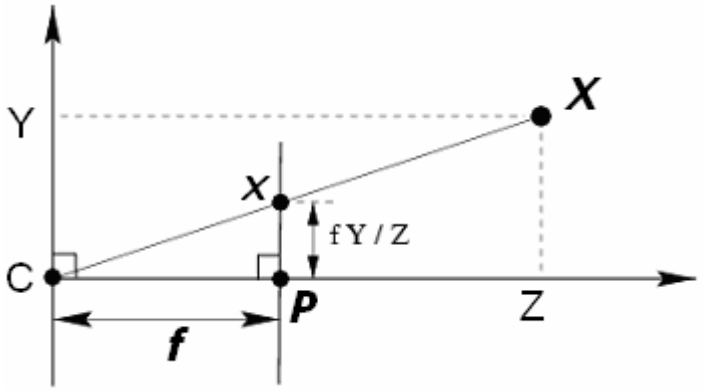


Figure 2.2: Side view of the projection of a 3-D point

A 3-D point ***X*** is projected to a point ***x***. If the coordinates of the point ***X*** is taken as $(X \quad Y \quad Z)^T$, then the projected coordinates can be easily calculated as $\left(fX/Z \quad fY/Z \quad f\right)^T$ from the similarity of triangles.

$$(X \quad Y \quad Z)^T \rightarrow \left(fX/Z \quad fY/Z \quad f\right)^T \qquad (2.1.1)$$

By using homogeneous coordinates, this transformation can be represented in matrix form.

$$\begin{bmatrix} fx \\ fy \\ z \end{bmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} [\mathrm{I} \mid 0] \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \qquad (2.1.2)$$

Then,

$$\overline{x} = P\overline{X} \text{ with } \overline{x} = \begin{bmatrix} fx \\ fy \\ z \end{bmatrix}, \ \overline{X} = \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \text{ and } P = \left[\begin{array}{c|ccc} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{array} \right] \quad (2.1.3)$$

where the *P* matrix is entitled as the *camera projection matrix*.

2.1.1.2 Updating the model to include origin shifts

Basic pinhole camera model assumes the center of the image plane as the origin. However, in general, the lower left corner is utilized as the image origin. The mapping for this case can be shown as

$$(X \quad Y \quad Z)^T \rightarrow \left(fX/Z + p_X \quad fY/Z + p_Y \right)^T \qquad (2.1.4)$$

and in matrix form

$$\begin{bmatrix} fx + p_X z \\ fy + p_Y z \\ z \end{bmatrix} = \begin{bmatrix} f & 0 & p_X & 0 \\ 0 & f & p_Y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} f & 0 & p_X \\ 0 & f & p_Y \\ 0 & 0 & 1 \end{bmatrix} [\mathrm{I} \mid 0] \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \quad (2.1.5)$$

$$\overline{x} = K[I \mid 0]\overline{X}_C \quad (2.1.6)$$

where the ***K*** matrix is denoted as the *camera calibration matrix*. This matrix is the most important parameter in 3-D reconstruction problems and if it is known beforehand, the frames are referred as "calibrated", otherwise as "uncalibrated".

The 3-D coordinates are denoted by $\overline{X}_C$ to notify that they are measured with respect to a coordinate system that is embedded to the camera coordinate system. The next section presents the change in the camera projection matrix, when a different coordinate system is used.

2.1.1.3 Updating the model to include coordinate system changes

In the current projection matrix, it is assumed that the 3D coordinates are measured with respect to the camera coordinate system. When the 3D coordinates are measured with respect to another coordinate system, the projection matrix has to be updated accordingly.

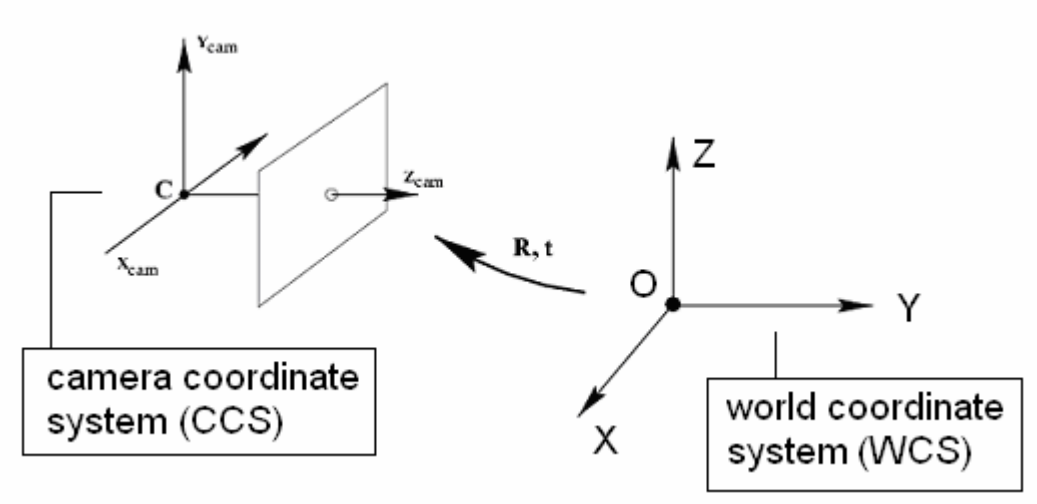


Figure 2.3: Transformation between world and camera coordinate systems

In Figure 2.3, the coordinate system, which is used to measure the 3D points, is called as the *world coordinate system (WCS)* whereas the other one as the *camera coordinate system (CCS).* Denoting the rotation and translation between the two coordinate systems with **R** and **t**, the relation between a coordinate that is measured with CCS and WCS is written as,

$$\bar{X}_{cam} = R(\bar{X} - C) \text{ with } t = -RC \qquad (2.1.6)$$

Hence, (2.1.6) is updated to,

$$\bar{x} = K[I \mid 0]\bar{X}_{cam} \rightarrow \bar{x} = K[I \mid 0]\begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix}\bar{X} \rightarrow \bar{x} = K[R \mid t]\bar{X} \qquad (2.1.7)$$

and hence

$$\bar{x} = P\bar{X} \text{ with } P = K[R \mid t] \qquad (2.1.8)$$

The parameters that are contained in the K matrix are entitled as *intrinsic parameters*, while the rotation matrix and translation vector are denoted as the *exterior parameters* of a camera. The estimation of these parameters is termed as *interior calibration* and *exterior calibration,* respectively.

2.1.1.4 Updating the model to pixel units

The derived camera projection matrix ignores the fact that a non-isotropic scaling in x and y-direction might occur. This disorder could occur in today's CCD cameras, when the pixel manufacturing results in non-square pixels. In order to avoid introducing unequal scale factors in each direction the camera projection matrix is multiplied by

$$\text{diag}(m_x, m_y, 1) \tag{2.1.9}$$

where m_x and m_y are the number of pixels per unit distance in x and y directions. The calibration matrix becomes

$$K = \begin{bmatrix} \alpha_X & 0 & x_0 \\ 0 & \alpha_Y & y_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.1.10}$$

α_X and α_Y are the focal lengths in x- and y-directions and (x_0,y_0) is the principal point in terms of pixel dimensions.

2.1.1.5 Updating the model to include skew

The skew parameter in the camera calibration matrix is due to the tilt of the pixels. When the pixels are not manufactured to have a 90-degree angle, the skew is non-zero. In today's cameras, the skew may be considered, as zero. However, for the former cameras, this degradation has to be considered.

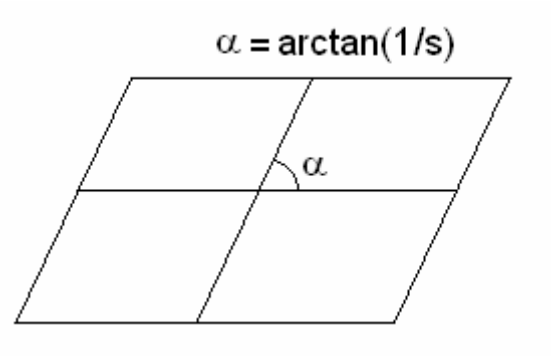


Figure 2.4: Skew in pixels

The final camera calibration with skew parameter is obtained as

$$K = \begin{bmatrix} \alpha_X & s & x_0 \\ 0 & \alpha_Y & y_0 \\ 0 & 0 & 1 \end{bmatrix} \qquad (2.1.11)$$

2.1.1.6 Final words

When a camera has a calibration matrix of the form, as in (2.1.11), it is called as *finite projective camera*. It has 11 degrees of freedom (5 internal and 6 external parameters), as a 3x4 homogeneous matrix. The camera center can be obtained as the right null vector of the projection matrix.

2.1.2 Radial distortion

The imaging operation is assumed to be perfectly linear up to this point. However, due to a phenomenon, called *lens distortion*, the process is in fact nonlinear. The degree of lens distortion increases as the focal length decreases. In Figure 2.6, a typical example for lens distortion is presented.

Figure 2.5 Radial distortion [1]: Left image represents the image before correction and right image is the corrected linear image.

The lens of the camera projects the points in the scene nonlinearly, according to their distance from the origin of the image plane, thus, this distortion is called as *radial lens distortion.*

Figure 2.6: Radial distortion example [1]: Notice the distortion in the linear lines in the left image. Right image is the corrected one; lines are straight in this image.

A solution to this problem is to apply a nonlinear transformation to the image pixels in order to remove the effects of distortion. However, it is crucial to correct this distortion in the right place. The distortion takes place in the projection of world coordinates onto the image plane before the application of calibration matrix.

$$\begin{bmatrix} x_d \\ y_d \end{bmatrix} = KL(r)[I \mid 0]X_{cam} \qquad (2.1.12)$$

2.1.2.1 Radial Distortion parameters

As stated before, the radial distortion occurs according to the radial distance of a pixel to the optical center. This distortion should be compensated for in some of the applications, such as reconstruction problems. The *un-distortion* function is modeled as a *Taylor series expansion* of the radial distance, since it depends on this value.

$$L(r) = 1 + K_1 r + K_2 r^2 + ... \qquad (2.1.13)$$

where $r = (x - x_C)^2 + (y - y_C)^2$ and (x_C, y_C) being the optical center.

$$\begin{aligned} x_{cap} &= x_C + L(r)(x - x_C) \\ y_{cap} &= y_C + L(r)(y - y_C) \end{aligned} \qquad (2.1.14)$$

In the above equation, x and y are the measured pixel coordinates and x_{cap} and y_{cap} are the corrected pixel coordinates. $L(r)$ function

is only defined for the positive values of r and $L(0) = 1$. The parameters of the radial distortion are also considered among the internal parameters of a camera. The estimation of these parameters is accomplished by minimizing a cost function, which measures the deviation of the model from a linear counterpart. In most of the systems, it is sufficient to estimate only the first two values of the expansion and furthermore adding more parameters to the un-distortion operation is avoided, in order not to cause numerical problems.

2.1.3 Camera Calibration

Camera calibration is the process of obtaining camera intrinsic parameters [1,2]. It is one of the most important steps in 3D computer vision for the extraction of 3D information from the captured scene. Structure and motion problems require a high level of accuracy of the camera matrix due to the nonlinearity of the problem of 3-D scene reconstruction. Moreover, without an accurate camera matrix, most of the algorithms are expected to fail to converge or converge to a physically meaningless solution. This important problem has been studied extensively by the researchers over the years [2-9]. Taxonomy of the methods can be proposed roughly in 4 categories, according to the dimension of the utilized calibration pattern [8]:

- Calibration with 3-D patterns
- Calibration with 2-D patterns
- Calibration with 1-D patterns
- Calibration with 0-D patterns (self calibration)

2.1.3.1 Calibration with 3-D patterns:

In this approach, camera calibration is performed by using a 3D pattern (see Figure 2.7), whose structure is known with a very high precision in 3-D space.

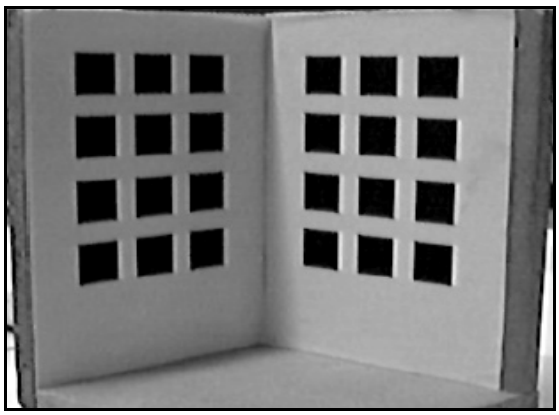


Figure 2.7: A 3D calibration pattern [11]

For example, the calibration procedure explained in [2] uses a 3D calibration pattern and it has been shown that, the calibration can be performed very efficiently [8]. Another example of this approach is the famous paper, by Tsai [4]. Tsai's method involves a 2D plane undergoing a precisely known translation, which also results with an information for the 3rd dimension. Although, the results of the Tsai's method are quite precise, it is a difficult procedure to achieve in practice.

2.1.3.2 Calibration with 2-D patterns:

The methods in this part involve observing a planar pattern (see Figure 2.8) from a limited number of views [6, 9]. The motion of the plane is unspecified, in contrast to the Tsai's technique [4],

and the required calibration pattern can be prepared by anyone easily and the results are quite acceptable.

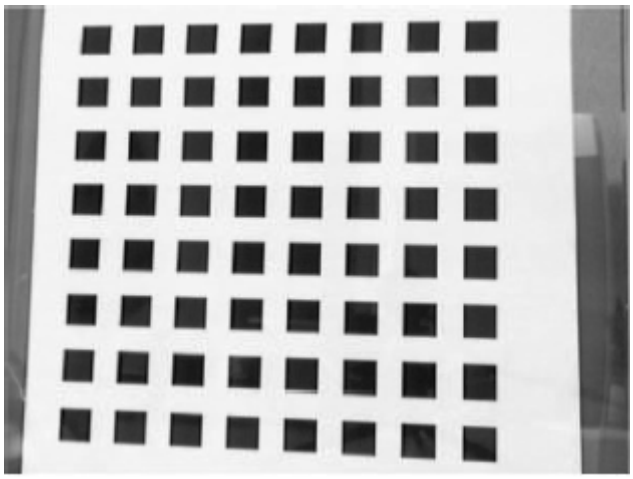


Figure 2.8: 2D Calibration Pattern [6]

In Zhang's method [6], a coplanar calibration pattern is captured a few times with different orientations by moving either the camera or the model plane. The world coordinate system is assumed to be aligned with the model plane, i.e. calibration pattern is on $z = 0$ plane and the x- and y-axes are parallel to the pattern features. The feature points are automatically detected from the captured images. As in [4], only this information is used in order to extract *intrinsic*, *extrinsic* and *distortion parameters* of the camera.

The estimation of the unknown calibration parameters in principle is quite similar to the method by Tsai [4]. The major difference is the absence of strict motion requirement for the camera to gather some depth information. The assumption of coinciding the z=0 plane with the calibration pattern simplifies the formulation of the

procedure. For more details of Zhang's method, the readers should refer to Appendix A.

2.1.3.3 Calibration with 1-D patterns:

Calibration pattern by using 1-D objects (see Figure 2.9) has not been studied extensively in comparison to the other classes of calibration.

Figure 2.9: 1D calibration pattern [8]

The method in [8] involves observing a linear pattern that is moved around a fixed point. This method is especially important, when multiple cameras are to be calibrated, where the calibration objects are required to be observed simultaneously [8].

2.1.3.4 Calibration with 0-D patterns (Self Calibration):

In self-calibration, no calibration pattern is used and therefore can be considered as a 0-D approach, since it only requires point matches between different views [1, 2, 3, 5, 6]. The rigidity of the scene [2] is used to compute the internal parameters of the

camera and if the images are captured by the same camera with constant internal parameters, three images are enough to compute the camera internal and external parameters, which are used to compute 3D structure of the scene [1, 3].

In self-calibration problem, the only available data are the images captured from various locations and orientations to estimate the camera intrinsic parameters. There are many different methods for self-calibration. As pioneers, Maybank and Faugeras [14] proposed a method, in which the nonlinear quadratic equations, called as *Kruppa equations*, are constructed by using Fundamental matrices and unknown camera matrices. After this pioneering work, these equations are attempted to be solved in different ways [14, 15, 16, 18, 19]. In another type of self-calibration method [22, 23], the camera intrinsic parameters are obtained by using the relation between the virtual conic and the camera intrinsic parameters. These methods later update the projective reconstruction to a metric reconstruction. In a marginally recent method by Pollefeys [24], calibration is performed in a stratified way. First of all, a projective reconstruction of the scene is formed and then, this is updated to affine by using the position of the plane of the virtual conic determined by solving a number of constraints [25]. Finally, this reconstruction is updated to metric by using the estimated camera intrinsic parameters, determined by solving the general camera self-calibration equations.

2.2 Epipolar Geometry and the Fundamental matrix

Epipolar geometry is the geometry of two views of a scene captured from different locations or orientations. It depends on

the camera intrinsic parameters, as well as the relative rotation and translation of these views. It is independent of the scene structure and can be expressed with a 3x3 matrix, denoted as *Fundamental matrix*. Since the Fundamental matrix encapsulates both the intrinsic and the extrinsic relations, it can be used to obtain a projective reconstruction of the scene. If the intrinsic parameters of the cameras are known, fundamental matrices are enough to complete a metric reconstruction of the scene. In fact, for the calibrated camera case, fundamental matrices may be further reduced to a *normalized* form, which is called as *Essential matrix* [27].

In this section, the relationship between two perspective views of a scene is to be explained. The concepts, such as *epipole, epipolar line* and *epipolar constraint* are introduced to the reader and the algebraic representation of these geometric concepts – Fundamental matrix - will be derived as well a brief explanation of its properties.

2.2.1 Epipolar geometry

Epipolar geometry is the study of two perspective views by the help of projective geometry tools. It investigates the relations and constraints that are imposed on certain geometric elements of the structure formed by camera locations and orientations.

In Figure 2.10, the plane formed by the two camera centers and the 3D point is called as *epipolar plane.* For different 3D points, there exist various epipolar planes. However, they all pass from

the line formed by two camera centers, C and C', called the *baseline*. The intersections of the baseline with the image planes are defined as *epipolar points (or epipoles)*. These points are the projections of the camera centers onto the other image plane.

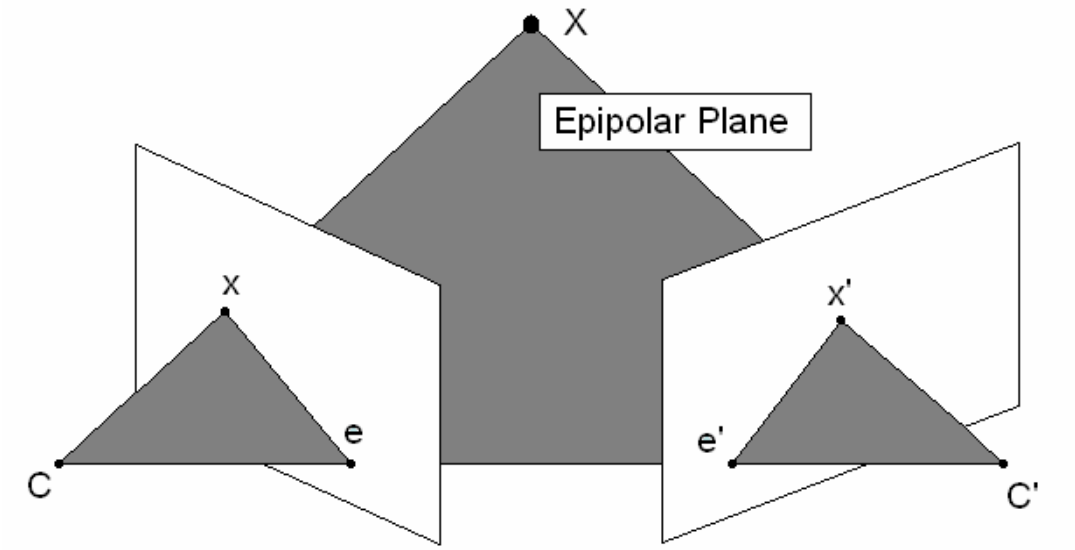


Figure 2.10: Epipolar Geometry: *C* and *C'* are camera centers. *X* is any 3D point and *x*, *x'* are its projections on different cameras

The location of the epipole depends both on the extrinsic and intrinsic parameters of the cameras. Therefore, changing the location and orientation of the image planes also relocates the epipole.

An *epipolar line* is the intersection of an epipolar plane with the image plane. Since all epipolar planes contain the baseline, all epipolar lines pass from the epipole.

As it can be observed from Figure 2.11, it is not possible to determine the exact location of the 3D point given only an image of the point in one image plane and the camera centers. In fact,

only the line that contains that 3D point can be obtained, since no information about the depth of the point exists. However, for calibrated cameras, the position of the correspondent point in the second image is constrained to a line by the help of epipolar geometry.

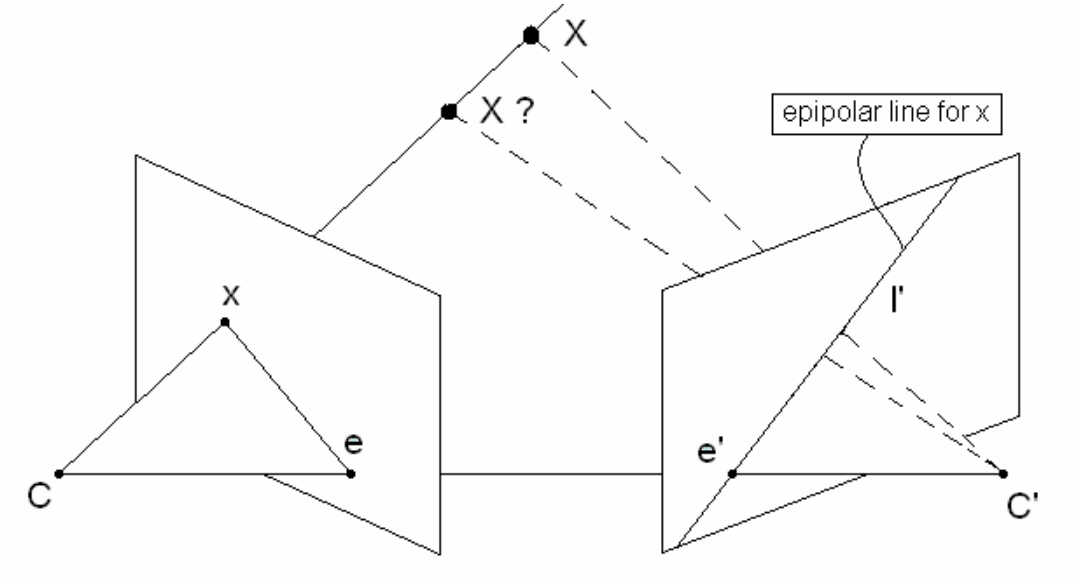


Figure 2.11: Back-projected point ambiguity: For a pair of calibrated cameras (*C* and *C'* known), knowing only *x* will not be sufficient to find the 3D point *X*.

Since all epipolar lines pass from the epipole, given two epipolar lines, the location of the epipole can be computed easily by a cross product. Examples for different camera configurations can be seen in Figure 2.12 and Figure 2.13.

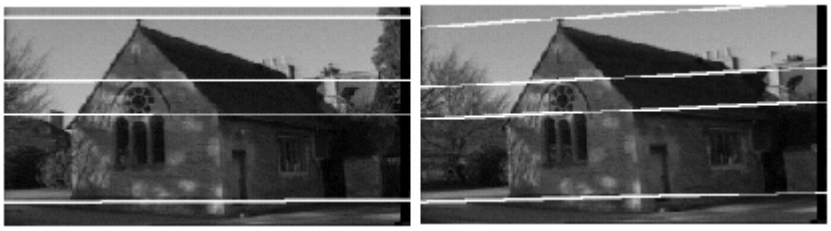


Figure 2.12: Parallel camera case [1]

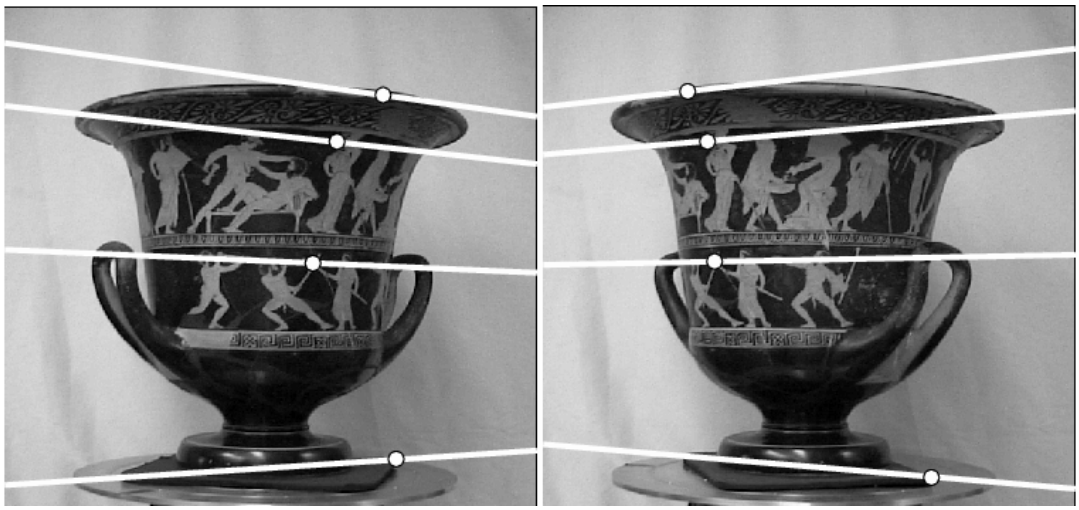


Figure 2.13: Converging camera case [1]

2.2.2 Fundamental matrix

The epipolar geometry describes the relation between two perspective images and Fundamental matrix is the algebraic relation of this geometry. Fundamental matrix is used to represent a geometric mapping between a point and a line in a stereo image pair. It encapsulates camera intrinsic and extrinsic information.

It is observed in the previous section that for a given point in the first image, there exists a line, *l'*, which contains the match of the first point. This line is in fact the projection of the ray in 3-space that emits outward from the camera center to the selected point *x*.

$$x \rightarrow l' \qquad (2.2.1)$$

This mapping can be represented by a 3x3 matrix (which is in fact the Fundamental matrix and the derivation of this matrix is given

in the next sections.) and this matrix is a projective mapping of a point to a line.

2.2.2.1 Geometric derivation

Let the transformation of a point *x* in the first image to the second image be performed by using a plane.

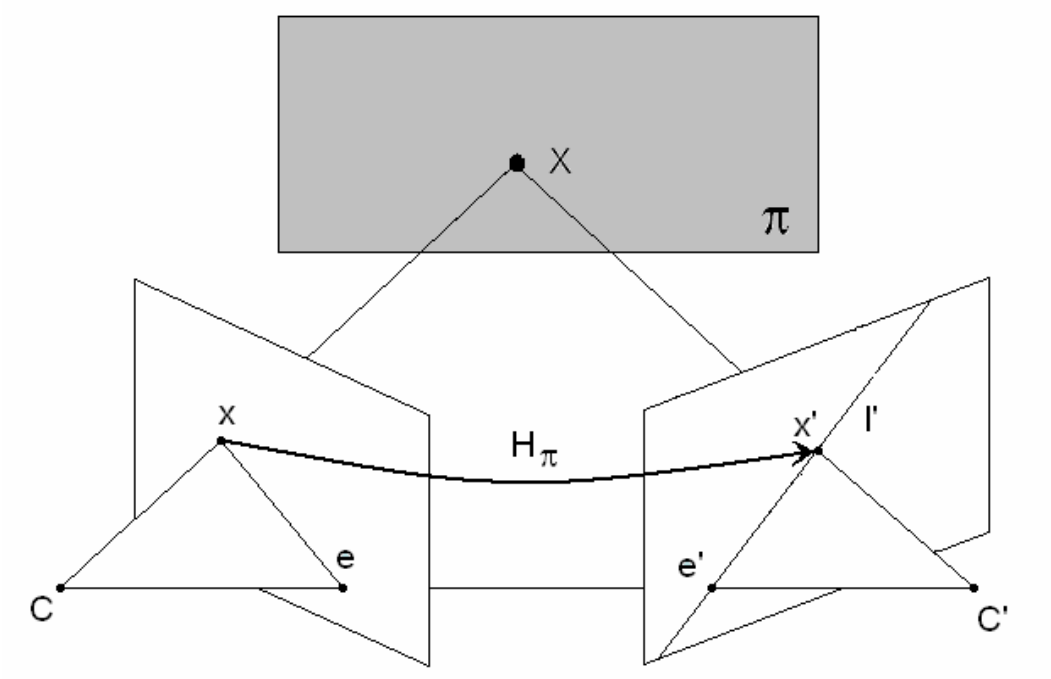


Figure 2.14: Transformation via a plane**.**

This transformation can be achieved by using any plane and it is called a *homographic transformation*, *H* [1]. Therefore, the homographic correspondence of *x* in the second image $\tilde{x}'$ can be obtained as,

$$\tilde{x}' = H_{\pi} x \qquad (2.2.2)$$

The point $\tilde{x}'$ has to be on the epipolar line that contains the correct match of the point x, since the ray which passes through x and the first camera center is not disturbed. Hence, the epipolar line equation can be obtained as,

$$l' = e' \times \tilde{x}' = [e']_x H_\pi x = Fx \text{ where } F = [e']_x H_\pi \qquad (2.2.3)$$

F is the fundamental matrix and $[\ ]_x$ expression is the cross product in the matrix form which is defined for an arbitrary vector $q = [a \quad b \quad c]^T$ as $[q]_x = \begin{bmatrix} 0 & -c & b \\ c & 0 & -a \\ -b & a & 0 \end{bmatrix}$.

It is observed from this derivation that there exists an equivalent class of fundamental matrices that can be used to represent the same setting of cameras. Furthermore, since $[e']_x$ term has rank 2, the fundamental matrix is also of rank 2. This is meaningful, since the fundamental matrix represents a mapping from a point (2D) to a line (1D), thus should have rank 2.

2.2.2.2 Algebraic Derivation

The expression of the fundamental matrix in terms of two projection matrices P and P' is first derived by [26].

The equation for the back-projected ray can be given as:

$$X(\lambda) = P^+ x + \lambda C \qquad (2.2.4)$$

where λ is any positive real number, C is the first camera center and P^+ is the pseudo inverse of the first projection matrix to give the relation $P^+P = I$. Since $PC = 0$, one gets

$$PX(\lambda) = x \tag{2.2.5}$$

For any given λ, $X(\lambda)$ corresponds to a 3-D point on this ray. Therefore, the projection of this 3-D point onto the second image plane is given as,

$$x' = PX(\lambda) = P'P^+x + \lambda P'C = P'P^+x + \lambda e' \tag{2.2.6}$$

Finally, the cross product of this point with the epipole will yield the epipolar line equation.

$$e' \times x' = e' \times (P'P^+x + \lambda e') = [e']_x P'P^+x = Fx \tag{2.2.7}$$

Finally, one reaches the following relation for F :

$$F = [e']_x P'P^+ \tag{2.2.8}$$

2.2.2.3 Epipolar Constraint

Since the fundamental matrix maps a point in the first image to a line in the second image, the correct match of this point should be on this line. This relation can be expressed as,

$$l' = Fx \tag{2.2.9}$$

$$l'^T x' = x'^T l' \rightarrow x'^T F x = 0 \qquad (2.2.10)$$

This expression is called as the *epipolar constraint* and it is a quite important equality, since it enables estimation of the fundamental matrix without any necessity for the camera internal or external parameters. There exist many algorithms which only use point correspondences to estimate the fundamental matrix [1,2]. Once the fundamental matrix is computed, it is possible to compute the camera calibration matrix and the extrinsic parameters.

2.2.2.4 Properties of the Fundamental matrix

Fundamental matrix, as explained above, is a projective mapping from a point to a line (i.e. *F* is a *correlation*). Hence, it maps the elements of two-dimensional space to the elements of one-dimensional space. Therefore, it is of rank 2. This result can also be observed from the fact that if two lines are corresponding epipolar lines, then any point on the first line should be mapped to the second line for which there is no inverse mapping and hence, *F* is not of full rank.

It is observed that every epipolar line mapped by the fundamental matrix passes through the epipoles. Therefore, it is not surprising to find the positions of the epipoles at the right and left null spaces of the fundamental matrix. For a brief summary of the properties of the fundamental matrix, one should refer to Table 2.1

Table 2.1: Properties of the fundamental matrix

- F is a rank-2 homogeneous matrix with 7 degrees of freedom.
- **Epipolar constraint**: If x and x′ are corresponding image points then $x'^{T} Fx = 0$
- **Epipolar lines:**
  - $l' = Fx$ is the epipolar line corresponding to x
  - $l = F^{T} x'$ is the epipolar line corresponding to x′
- **Epipoles:**
  - $Fe = 0$
  - $F^{T} e' = 0$
- **Formulation of F:**
  - **with projection matrices:** $F = [e']_{x} P' P^{+}$ where P^{+} is the pseudo-inverse of P
  - **transformation via a plane**: $F = [e']_{x} H_{\pi}$ where H_{π} is any homographic transformation

2.2.3 Essential Matrix

Essential matrix is the *normalized* version of the fundamental matrix, which is introduced to the literature by Longuet-Higgins [27]. It is also sometimes denoted as *normalized fundamental matrix* and includes information only about the rotation and the translation of the image planes. It is independent of the camera calibration parameters and hence, it is denoted as *normalized*.

Given two projections of a 3D point *X,* as $x = PX$ and $x' = P' X$, the *normalized image coordinates* can be easily found as $x_{cap} = K^{-1} x$ and $x'_{cap} = K'^{-1} x'$. x_{cap} and x'_{cap} are independent of their respective calibration matrices and the new projection matrices $K^{-1}P$ and $K'^{-1} P'$ are called *normalized projection*

matrices. The fundamental matrix between the normalized coordinates are called as the *essential matrix* and it is equal to [1],

$$E = t \times R \tag{2.2.11}$$

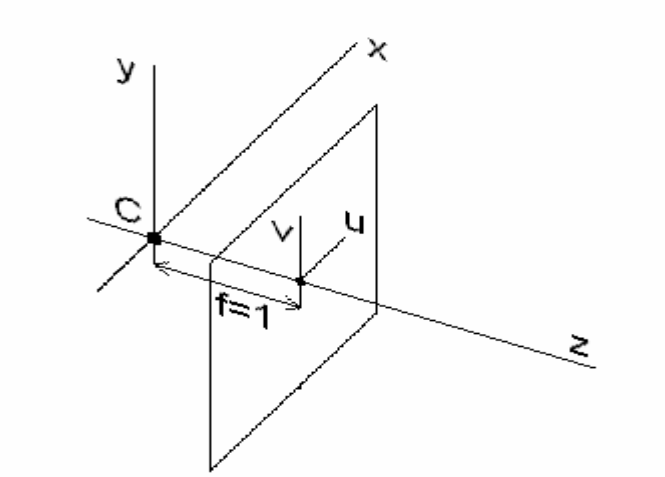


Figure 2.15: Normalized image coordinate system

The epipolar constraint between the image coordinates and the fundamental matrix exists between the essential matrix and the normalized image coordinates, as well:

$$\hat{x}_{cap}{}^{T} E x_{cap} = 0 \tag{2.2.12}$$

The relationship between the fundamental matrix and the essential matrix also exists:

$$E = K'^{T} F K \tag{2.2.13}$$

Since the parameters of the calibration matrices are excluded, essential matrix has only 5 degrees of freedom: rotation and

translation has each three degrees of freedom, whereas the overall scale ambiguity decreases the freedom by one.

CHAPTER 3

3D SCENE RECONSTRUCTION FROM TWO-VIEWS

This chapter presents a scene reconstruction algorithm at sparse points from two calibrated views. *Sparse*ness is meant in the sense that the reconstructed scene does not contain the depth information for all the pixels of an image, but only a small subset of them can be estimated. On the other hand, *calibrated* term denotes the availability of the internal parameters of the recording cameras, a priori.

The chapter is organized as 6 sections. The first one presents the outline of a typical 3-D reconstruction algorithm and following four sections gives some detailed information about the main blocks of this algorithm. Finally, the simulation results are presented in the last section to asses the performance of this algorithm.

3.1 Outline of the reconstruction method

Although, there might be different solutions to the 3-D scene reconstruction problem, the two-view reconstruction algorithm,

which is utilized in this thesis, can be summarized in 4 main steps (see Figure 3.1):

- Finding a set of putative correspondence pairs
- Estimating the fundamental matrix between these views
- Computing the pose of the views with respect to each other and calculating the camera matrices of the views
- For each pair of correspondence, determining a point in 3-D space that project to these points.

In order to estimate the relative geometry between two images, it is necessary to find some point matches between these views. The first step of the reconstruction algorithm is therefore the estimation of a set of putative correspondences. During the estimation of correspondences, some differentiable features of the images should be obtained. The computation of the salient features and the following matching processes are explained in Section 3.2.

Given a set of correspondences, it is now possible to estimate the geometric relation between these two images by using the epipolar constraint. Given at least eight correspondences, it is possible to estimate the fundamental matrix in a linear manner. If more than eight correspondences are present, then the solution can be determined by any least squares method. The estimation, however, is not a straightforward process, in case of a set of correspondences containing outliers. In such a situation, a robust method is required. Section 3.3 discusses the estimation of the fundamental matrix in a robust manner.

Once the Fundamental matrix is estimated, Essential matrix is calculated as a result of basic matrix operations from the available calibration information. The computation of the projection matrices, however, requires rotation and translation parameters between the two views. Therefore, the decomposition of the Essential matrix into rotation and translation parameters is necessary. This process is explained in Section 3.4.

Finally, once a set of correspondences and the projection matrices of the views are determined, only the estimation of the positions of the object points remains. This process is usually denoted as *triangulation.* Some extra constraints should be considered in the estimation of 3-space points, such as their invariance to certain transformations and their projection errors. Triangulation is another important step, since the final output of the system is obtained at this stage. In Section 3.5, five different triangulation methods are explained and lastly, an optimal one is introduced.

General outline of the reconstruction algorithm is given in Figure 3.1.

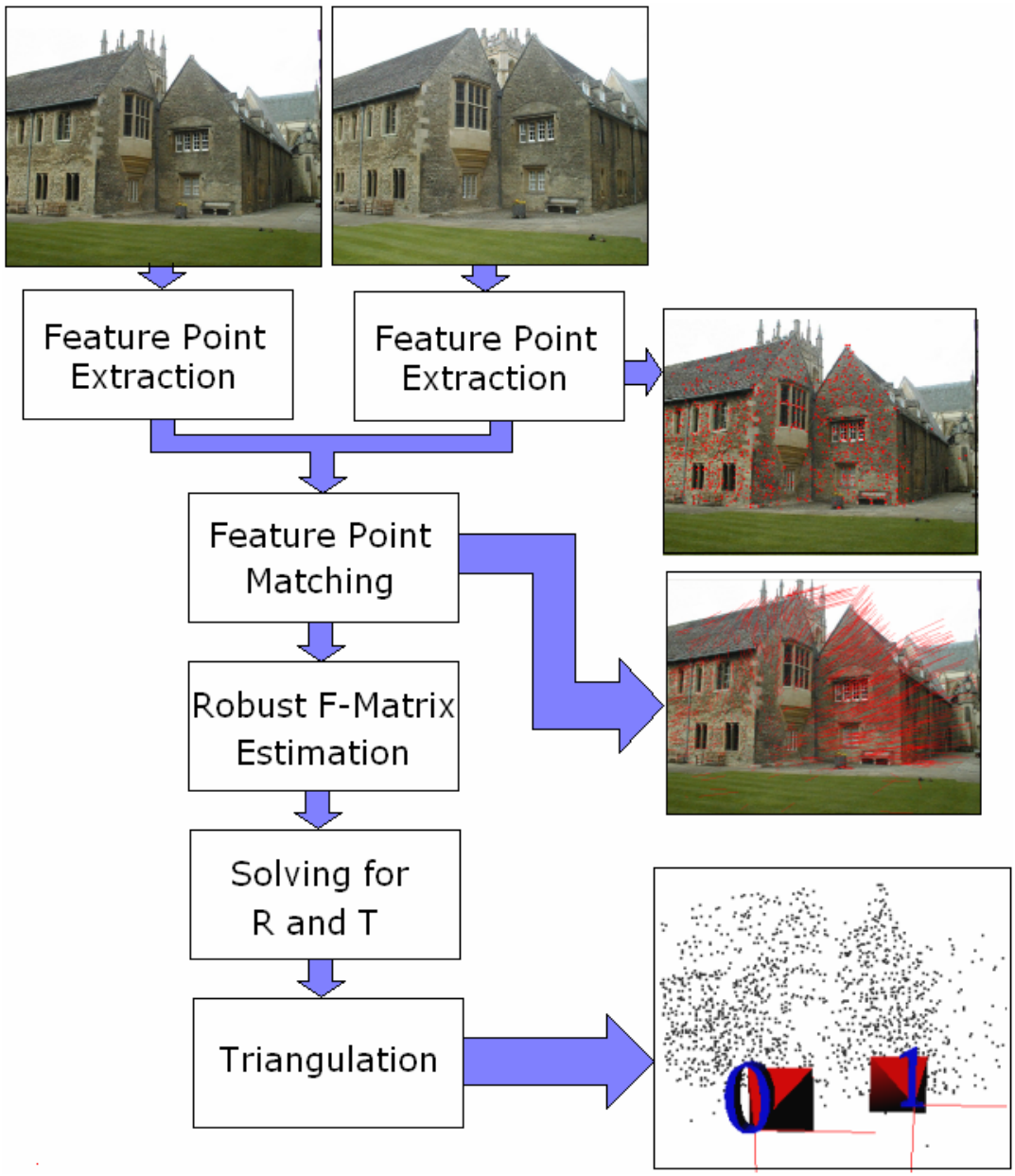


Figure 3.1: Outline of the reconstruction method

3.2 Finding correspondence pairs

Every image of a scene contains abundant information for the problem of estimating the relative geometry between these frames. Therefore, it is rational to reduce the processed information by using the most distinct properties of the images for estimation. For this purpose, *features,* salient primitives of images, are extracted. Although, many other interest points can be selected, the usual approach is to use corners on the images, as salient primitives. The two-dimensional location of a corner is called as a *feature* point, and the 3-D position of such a corner is termed as an *object point*.

A correspondence pair is a pair of feature points from different images to which an object point is projected. The correspondence estimation problem is to find the location of a given pixel in a different image. In most of the cases, the only input is the intensity map of the image and from this map, one would like to find the position of the searched pixel. This objective is not a trivial operation, since the transformation that a pixel might undergo is quite diverse. Some of these transformations are rotation, translation, scale changes, affine transformation, intensity changes due to illumination and the camera variations.

It is observed in Section 2.2.2 that from a set of correspondence pairs, it is possible to estimate the fundamental matrix and hence the geometric relations between the inspected image pair. Therefore, correspondence estimation is a crucial step in scene reconstruction problems. In order to achieve this goal, one should

first detect certain features from the frames at hand. In the next section, this topic is elaborated, while the following section discusses correspondence estimation problem.

3.2.1 Feature point detection

Feature points are discernable, salient elements of an image such that it is possible to find a match of the feature in another image of the same scene. This definition simply states that the feature points should be *traceable*.

There are many approaches that try to detect feature points in different ways [13, 42, 43, 49, 50, 52]. The method by Harris and Stephens [13], for example, depends on image gradient evaluation. This method is insensitive to illumination changes and translation differences. It is one of the most widely used feature extractor, which performs quite well for small camera movement, where captured images do not change in a large extent. Mikolajczyk, *et al.* [42], on the other hand, present a more complex feature detector, whose features are insensitive to affine transformations, including scale changes. Their method obtains invariant feature points under arbitrary moving conditions for various scales. However, this method has a quite high computation requirement and it is unnecessary to utilize such an approach for an input set from a video sequence which is not very arbitrary.

It has been shown that [43] Harris corner detector finds feature points in image sequences more consistently than many other

feature detectors. Therefore, in this thesis a modified version of the Harris corner detector with a subpixel resolution has been used.

3.2.1.1 Algorithm Overview:

Harris corner detector examines the gradients of the image intensity values and it aims to select the features by choosing points that has strong intensity changes in both *x*- and *y*-directions. In this way, the method eliminates the problem of selecting *edge pixels* that are not suited for tracking and matching tasks due to their tendency for giving similar matching scores with the remaining pixels in an edge (see Figure 3.2).

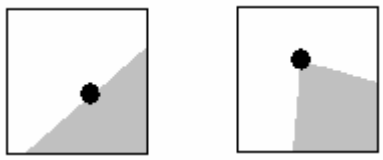


Figure 3.2: Harris corner detector does not prefer the feature in the left image. However, due to its high gradient value, the right one will be chosen

An approximation to the intensity dissimilarity between an image patch and a slightly shifted patch can be represented as [13]:

$$D(\Delta x, \Delta y) = \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} M \begin{bmatrix} \Delta x & \Delta y \end{bmatrix} \text{ with } M = \int\int_{W} \begin{bmatrix} I_x \\ I_y \end{bmatrix} \begin{bmatrix} I_x & I_y \end{bmatrix} \omega(x, y) dx dy \tag{3.2.1}$$

where I_x and I_y refer to the intensity derivatives in *x*- and *y*-directions and *w(x,y)* is a smoothing operator.

The computation of *M* matrix for discrete valued images should be obtained via summation and *M* matrix will become:

$$M = \begin{bmatrix} \sum \hat{I}_x^2 & \sum \hat{I}_x \hat{I}_y \\ \sum \hat{I}_x \hat{I}_y & \sum \hat{I}_y^2 \end{bmatrix} \qquad (3.2.2)$$

where $\hat{I}_{...}$ represents the smoothed image intensity gradients.

It is desired to have large eigenvalue terms for the *M* matrix, since it gives a measure of the intensity change around the considered pixel. If both of the eigenvalues are large, then this situation should indicate a peak shaped change. In order to ensure large eigenvalues without calculating them explicitly, Harris proposed to use a measure of the form,

$$R = \det(C) - k * trace^2(C) \qquad (3.2.3)$$

This measure is called as the *Harris cornerness measure* [13]. The feature points are selected at those pixels which give high cornerness values.

Once the corner pixels are detected by the Harris corner detector, a subpixel resolution corner is determined by fitting a bi-quadric polynomial to the cornerness surface in a window. Details of bi-quadric polynomial fitting are presented in Appendix B. In this implementation, the value of k has been taken as 0.04 (a

suggestion also made by Harris [13]) to provide preference against high contrast pixel step edges. The feature point extraction algorithm is summarized below.

Algorithm 3.2.1: Feature point extraction

1. Compute the image gradients in the x and y directions
2. Apply an NxN Gaussian filter to the image gradients.
3. Compute the M matrix and the R measure for every pixel
4. By sliding a window of NxN, find the points that are local maxima and have R values greater than a threshold
5. Fit a bi-quadric polynomial to the R surface in the NxN neighborhood of selected corners and compute the coordinates that give maximum cornerness value (R) from the fitted polynomial.

3.2.1.2 Conclusion

The presented algorithm for feature extraction is tested for the gain in the error measure. It is observed from the experiments performed (Table 3.2) that for a relatively minor computational load, subpixel accurate feature-detection increases the performance considerably. Moreover, during these experiments, it is observed that if the support rectangle size (N) of the fit is chosen different from the size of the Gaussian filter, then more than one local maximum might be obtained within the support. Such a situation should surely disrupt the detection of the true maxima due to the inferior approximation of the fit. Therefore, it is recommended to use same sized filters and windows throughout the process.

3.2.2 Finding putative matches

Once the salient features for the two images are extracted, one should use a procedure for finding the correspondence of a feature in the second image. This problem is denoted as the *matching* (association) problem. There are many proposed algorithms for the solution of this problem. The simplest method is the *correlation-based matching* [44]. In this method, the features are matched according to their correlation score with each other in a predefined pixel neighborhood. Although, this method might be used for images with some small disparity, it is not very suitable for general views. In order to improve this method, imposing some extra constraints on such candidate matches have been proposed [36, 45, 46]. *Neighborhood* constraint is one of such limitations to minimize erroneous matches. In this type of matching, an extra score is calculated for the goodness of the match by considering the neighbor match states and through a relaxation procedure, the correspondences are established. These methods are, in fact, quite successful for small or medium baseline settings [44]. In this thesis work, the aim is to reconstruct a scene from video frames and thus, the level of success and complexity of the neighborhood-based methods are quite sufficient. Therefore, this type of a matching algorithm has been selected for the implementation.

3.2.2.1 Matching through correlation

Given a feature point in the first image, a set of candidate matches is formed by using a normalized cross correlation

measure. The operation is performed in a search area that restricts the distance of a pixel that may traverse. This is sensible due to the small baseline assumption. The correlation window is usually selected as a square window of size NxN (see Figure 3.3)

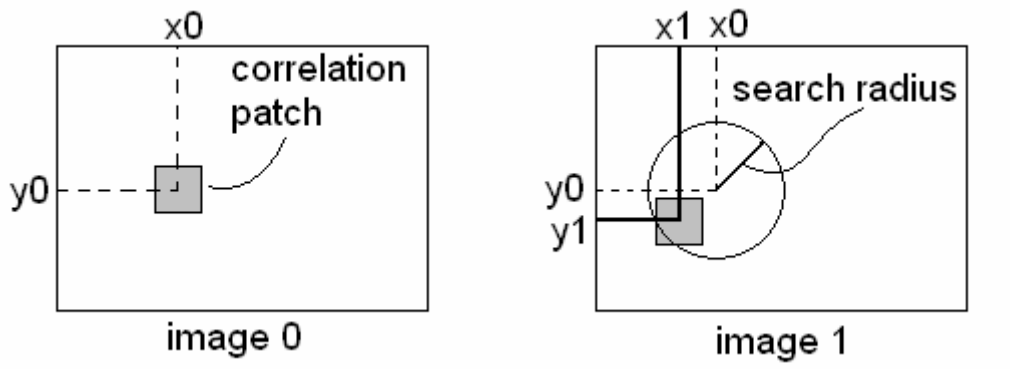


Figure 3.3: Correlation operation: Correlation patch and the search radius

Normalized cross correlation (NCC) is defined as [1]

$$NCC(p_1,p_2)=\frac{\sum\sum\left[I_0(x_0+i,y_0+j)-\overline{I_0(x_0,y_0)}\right]\left[I_1(x_1+i,y_1+j)-\overline{I_1(x_1,y_1)}\right]}{(2n+1)^2\sqrt{\sigma^2(I_0)\times\sigma^2(I_1)}} \quad (3.2.4)$$

where $\overline{I_k(x,y)}=\frac{1}{(2n+1)^2}\sum_{i=-n}^{n}\sum_{j=-n}^{n}I_k(x+i,y+j)$ is the average intensity value at point (x,y) of $I_k, k=0,1$ and $\sigma(I_k)$ is the standard deviation of the image I_k in the neighborhood $(2n+1)\times(2n+1)$ of (x,y), which is defined by:

$$\sigma(I_k)=\sqrt{\frac{\sum_{i=-n}^{n}\sum_{j=-n}^{n}(I_k(x+i,y+j)-\overline{I_k(x,y)})^2}{(2n+1)^2}} \quad (3.2.5)$$

The measure in (3.2.5) ranges from -1 (for the two correspondences totally "mismatch") till 1 (for the two correspondences exactly the same). Utilization of only NCC as a matching constraint does not yield good results (See Figure 3.4, Figure 3.5, and Table 3.1). From Table 3.1, it can easily be observed that for surfaces that contain repetitive textures, NCC might return high values for the geometrically incorrect points. Therefore, another mechanism is necessary in order to disambiguate matches.

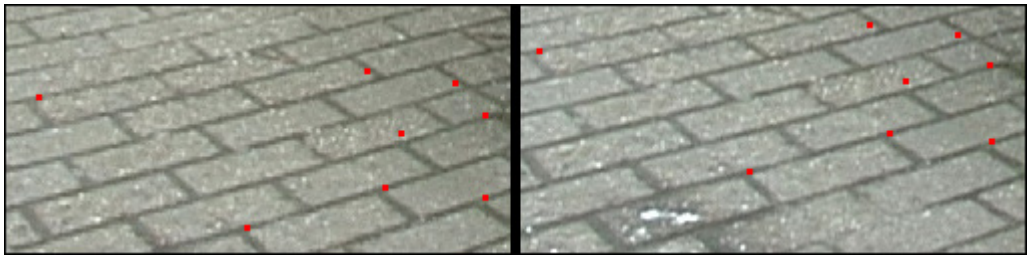


Figure 3.4: Images with extracted corners by using NCC

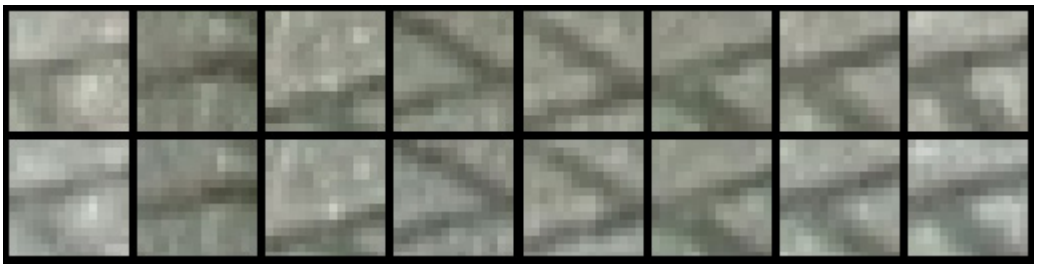


Figure 3.5: Details for the local regions around the marked pixels in Figure 3.4. Upper regions are taken from the first image and lower regions are taken from the second image. Note the similarities between the patches at the upper and lower rows for different columns.

Table 3.1: NCC scores for all possible combinations of the image patches.

0.8518	0.4412	-0.0284	-0.1795	-0.1907	0.5583	0.5011	0.5704
0.2467	**0.8817**	-0.2159	0.0868	0.0997	0.2214	0.2348	0.2560
0.0048	-0.2532	**0.9446**	0.0570	0.4038	0.2000	-0.0198	0.0188
-0.2527	0.0747	-0.0120	**0.7930**	0.7012	-0.2370	-0.2892	-0.3637
-0.2090	0.0459	0.4001	0.2350	**0.7642**	-0.1993	-0.2322	-0.2122
0.5947	0.2872	0.1879	-0.1801	-0.2308	**0.9645**	0.7482	0.7749
0.7252	0.3607	0.0121	-0.2000	-0.2280	0.7920	**0.9228**	0.9010
0.7343	0.3389	0.0378	-0.2996	-0.2591	0.7516	0.8488	**0.9554**

In the above table, columns are the image patches taken from the first image (first row of Figure 3.5) and rows are the image patches taken from the second image (second row of Figure 3.5). Notice that for some matches NCC still gives "good results" for wrong matches (good results: light shaded matches at the off-diagonals).

3.2.2.2 Disambiguating matches

A point in one image might be matched to more than one point in the other image, while yielding high correlation measures (see Table 3.1). Such a collection is called as *candidate match set.*

There are a number of methods proposed to solve these uncertainty problems [45, 46, 36]. The procedure that is preferred in this system uses the *neighborhood constraint* [36] together with a *relaxation* process. The inspiration of the algorithm is its allowance of the candidate matches to structure themselves by propagating some constraints throughout the set, such as

permanence and uniqueness, by using the neighborhood constraint.

3.2.2.2.1 Strength of a candidate match

Let there exist a candidate match (m_{1i}, m_{2j}) where m_{1i} is a point in the first image and m_{2j} is a point in the second image. Representing the neighbor set of m_{1i} by $N(m_{1i})$ and the neighbor set of m_{2j} by $N(m_{2j})$, which are formed by the feature points that are located within a disc of radius R around m_{1i} and m_{2j}, respectively. The essence of the neighboring constraint is that if the (m_{1i}, m_{2j}) candidate match is a good match, then it is highly probable to find more matches in the neighbor set of these two points such that the position of these neighbors relative to the original points m_{1i} and m_{2j} are similar. Conversely, if the (m_{1i}, m_{2j}) match is an inferior one, then one should expect to find a small number of matches or even not any at all in the neighborhood set.

The formal expression of this rationale is called as *strength measure* and it is equal to

$$S_M(m_{1i}, m_{2j}) = c_{ij} \sum_{n_{1k} \in N(m_{1i})} \left[\max_{n_{2l} \in N(m_{2j})} \frac{c_{kl}\, \delta(m_{1i}, m_{2j}; n_{1k}, n_{2l})}{1 + dist(m_{1i}, m_{2j}; n_{1k}, n_{2l})} \right] \quad (3.2.6)$$

where c_{ij} and c_{kl} are the normalized cross correlation scores explained in the previous section and $dist(m_{1i}, m_{2j}; n_{1k}, n_{2l})$ is the average distance of the pairing which is calculated as,

$$dist(m_{1i}, m_{2j}; n_{1k}, n_{2l}) = \frac{[d(m_{1i}, m_{2j}) + d(n_{1k}, n_{2l})]}{2} \qquad (3.2.7)$$

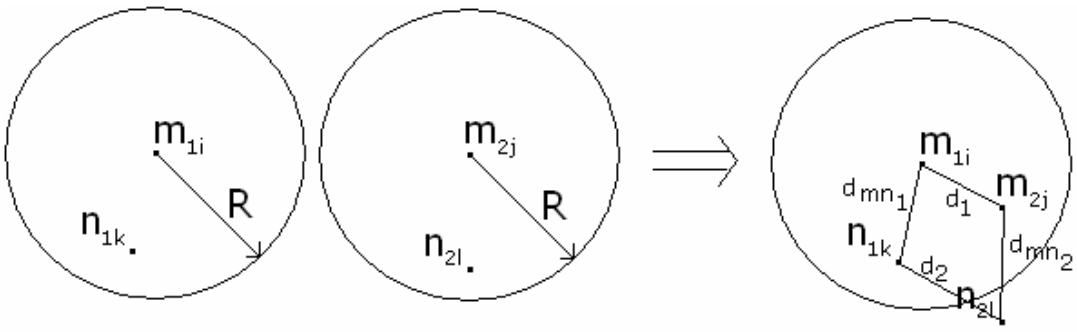
with $d(m,n)$ is the Euclidean distance between m and n. The final term left is the gain of pairing,

$$\delta(m_{1i}, m_{2j}; n_{1k}, n_{2l}) = \begin{cases} e^{-r/\varepsilon_r} & \text{if } (n_{1k}, n_{2l}) \text{ is a candidate} \\ & \text{match and } r < \varepsilon_r \\ 0 & \text{otherwise} \end{cases} \qquad (3.2.8)$$

where r is the relative distance of the pairing given as

$$r = \frac{\left| dist(m_{1i}, n_{1k}) - d(m_{2j}, n_{2l}) \right|}{dist(m_{1i}, m_{2j}; n_{1k}, n_{2l})} \qquad (3.2.9)$$

and ε_r is a threshold on the relative distance difference.



$d_1 = d(m_{1i}, m_{2j})$ $d_2 = d(n_{1k}, n_{2l})$ $d_{mn_1} = d(m_{1i}, n_{1k})$ $d_{mn_2} = d(m_{2j}, n_{2l})$

$$Dist_{AVG} = dist(m_{1i}, m_{2j}; n_{1k}, n_{2l}) = \frac{d_1 + d_2}{2} \qquad r = \frac{\left| d_{mn_1} - d_{mn_2} \right|}{Dist_{AVG}}$$

Figure 3.6: Strength measure equations

The strength measure of (3.2.6) has some preferable properties to worth mentioning. Firstly, the idea of finding more matches around a good candidate match is included to the measure by the summation term which effectively counts the neighbors. Secondly, the weighting is carried out according to the relative distance term (r). This selection is due to the second part of the assumption that the position of the neighbor matches relative to the original points to be similar. This approach, in fact, is justified by the premise that an affine transformation can be used to approximate the change between the neighborhoods of candidate matches, which are considered in a small area. Another property of this weighting is that it is a strictly monotonous function. This monotony makes distant matches less effective on the overall measure, compared to the close ones. The overall weighting function has also been normalized according to its distance to the match. This normalization has a similar influence on the measure, since being monotonous for close matches effect the strength compared to the distant matches more. Lastly, *max* expression helps to include only the closest match of the neighbor set, if there is more than one match.

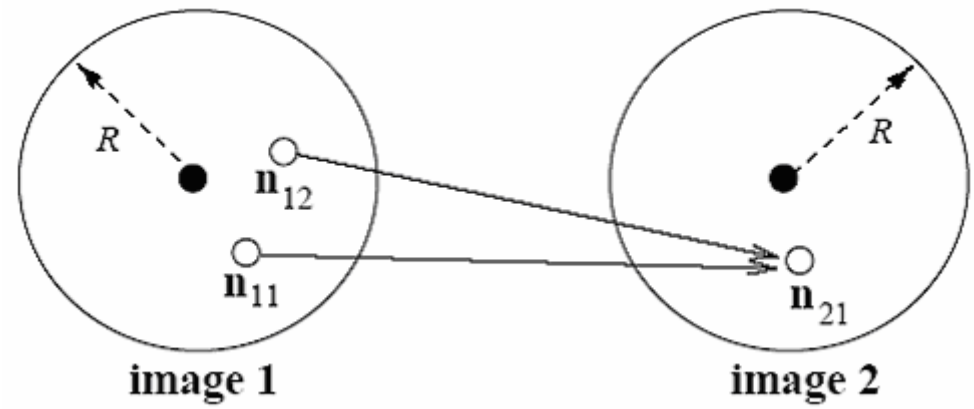


Figure 3.7: Non-symmetricity problem of the strength measure [36]

The overall measure has also an important disadvantage: it is not symmetric. The strength value will be different for a candidate match pair (m_{1i}, m_{2j}) if more than one point in the $N(m_{1i})$ neighborhood gives the maximal values with the same point in the $N(m_{2j})$ set (see Figure 3.7).

This problem can be avoided easily with a slight modification in the matching algorithm. For this end, before computing the summation, if more than one point from the $N(m_{1i})$ neighborhood scores maximal value with the same point from the $N(m_{2j})$ neighborhood, only the point that results with larger point is counted. In this way, when the order of the images is reversed, the same strength measures will be calculated.

3.2.2.3 Relaxation procedure

The strength measures of all the candidate match pairs formed in the correlation phase are calculated in the previous section. At this step, establishing correspondences according to these strength values should be the next aim. The relaxation method [34] is a solution for this problem. In this approach, the best matches throughout the whole set are selected and then, the remaining points are matched within themselves. Clearly, this is an iterative procedure. The formal expression for relaxation can be given as follows,

```
While( !convergence )
{
        • Update matches by looking at the SM values
        • Reduce the set of unmatched points
}
```

Updating matches can be performed in a number of approaches. One method is the *winner-take-all,* which is introduced by Rosenfeld [47]. In this method, for two points to be declared as a match, none of them should have a greater SM value with another point. For every iteration of the relaxation, the matches, which are selected as explained, are immediately stated as correct and due to the uniqueness constraint, all the remaining strength measures associated with the matched points, are removed from further consideration. In the next iterations, this approach should result in finding more matches that are not assigned or eliminated before. This method works similar to a steepest descent procedure and hence, it is relatively quite fast, but sometimes, as in all the steepest descent approaches, it may stuck to a local minima.

On the other hand, a slightly modified version of this method is more robust to the local minima problem. The name of the method is *some-winners-take-all* [36]. In this method, not all of the matches are stated as correct, but only the best α-percent of them are selected. The "goodness" is decided by the use of two tables. The first table is the list of all matches and their SM values sorted in a decreasing order according to the SM values. The second table is also a list of matches; but its second column is formed by the ambiguity score of the matches. This second table is also sorted according to its second column in a decreasing order. The ambiguity of a match is defined as the difference of the ratio of the highest two SM scores of it with 1 i.e.,

$$U_A = 1 - S_M^{(2)} / S_M^{(1)} \qquad (3.2.10)$$

From these two tables, only the matches that are within the first α-percent of both of the tables are selected. The rest of the method is similar to the first one: SM values associated with the matched points are extracted from the overall set and in the next iteration new matches are found from the reduced set. Due to its more robust structure, *some-winners-take-all* approach is adopted into our system.

Algorithm 3.2.2: Correspondence estimation

1. Estimate the candidate match set for feature points in the first and second image. For every feature point of the first image, compute the NCC score with the feature points in the second image within a disc of radius R and choose the ones that give high scores over some threshold (Equation 3.2.4).
2. Compute SM values for every candidate match according to Equation 3.2.6
3. Relaxation
   Until convergence
   a. Compute sorted SM and ambiguity tables
   b. Choose candidate matches that are present in the α-percent of both of the tables and mark them as "correct".
   c. Remove the SM values associated with the selected candidate matches
   d. If no other candidates remain or the SM scores of the best match in an iteration is below some threshold, terminate the loop

3.2.2.4 Conclusion

Feature matching operation by using only the normalized cross correlation (NCC) measure has been found out to be insufficient for the repetitive textured regions (Table 3.1). For this reason, a

neighbor-based matching measure together with NCC, called the strength measure (SM), is included to the algorithm. The results are improved to be satisfactory (see Figure 3.8).

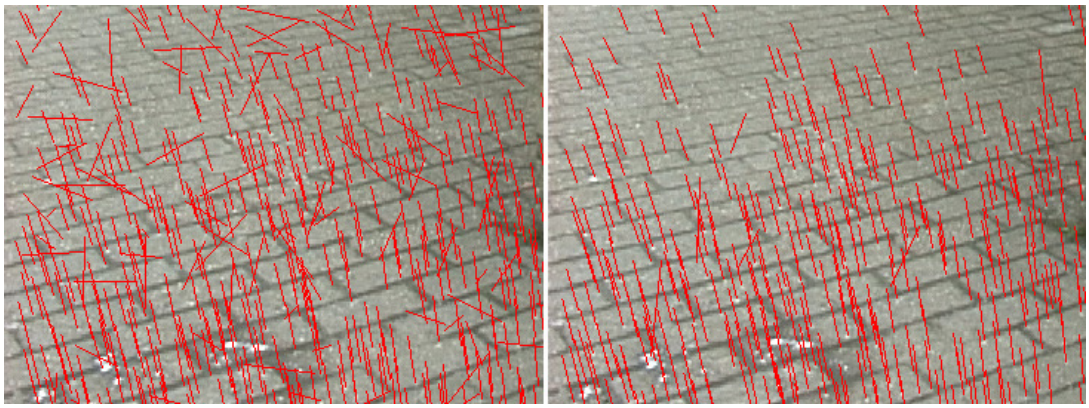


Figure 3.8: Comparison of single NCC vs NCC+SM results. In the left image, the results for using only the similarity measure can be observed. In the right image NCC is used together with SM. Most of the outliers due to the repetitive texture of the scene are eliminated.

3.3 Robust Computation of the fundamental matrix

As explained in the previous chapters, Fundamental matrix is an algebraic relation that relates the geometry between two perspective images of a scene. It is used to represent a geometric mapping between a point and a line in a stereo image pair. This relation must hold for all the correspondences of the image pair. Therefore, this property might also be utilized as a consistency measure for the computed correspondence pairs.

It is known that the fundamental matrix can be estimated from the computed correspondences of the scene. In fact, from eight given correspondences, it is possible to find a unique solution for F defined up to a scale factor. This approach is denoted as the *8-*

Point Algorithm, which is introduced by Longuet-Higgins for the computation of the *essential matrix* for the case of calibrated cameras [27]. The method does not impose the rank 2 constraint, and hence, it has been found out to be very sensitive to noise [28, 29, 21]. However, a clear advantage of this algorithm against more complex algorithms is its linearity, hence its speed and ease in implementation. On the other hand, Hartley [31] has shown that after making a slight modification to this algorithm by normalizing the correspondences, its performance increases significantly and becomes comparable with the best iterative methods. The modified version of the 8-point algorithm is called as the *normalized 8-point algorithm* and in this thesis, this algorithm is exploited.

3.3.1 8-Point Algorithm

The epipolar constraint

$$x'^{T} Fx = 0 \tag{3.3.1}$$

can be reformulated to be a linear equation in terms of F parameters.

$$u^{T} f = 0 \tag{3.3.2}$$

where

$$\begin{aligned} u &= [u_1u_2, v_1u_2, u_2, u_1v_2, v_1v_2, v_2, u_1, v_1, 1]^T \\ f &= [\, F_{11}, F_{12}, F_{13}, F_{21}, F_{22}, F_{23}, F_{31}, F_{32}, F_{33}\,]^T \end{aligned} \tag{3.3.3}$$

with $x = \begin{bmatrix} u_1 \\ v_1 \\ 1 \end{bmatrix}$ and $x' = \begin{bmatrix} u_2 \\ v_2 \\ 1 \end{bmatrix}$ representing a point match.

From all point matches, stacking these equations row by row, a set of linear equations in the form of $Af = 0$ is obtained, where *f* is the column vector containing the elements of the fundamental matrix and *A* is the equation matrix. The fundamental matrix is defined up to a scale and therefore the magnitudes of the elements in the *f* vector are not important. Hence, adding an additional constraint $\|f\| = 1$ to avoid the trivial solution will not change the problem.

For finding a unique solution to (3.3.3), at least eight point correspondences are required. If more than eight matches are utilized, then the system becomes over-determined. For an over-determined system to have a non-zero solution, the rank of the *A* matrix must be at most eight. However, in the existence of noise, (i.e., for correspondences found from a real stereo pair) *A* matrix might have a rank value of nine. In this case, it will be not possible to find a non-zero solution for the $Af = 0$ relation. Instead, the solution to this problem will be the least-squares solution of minimizing $\|Af\|$ subject to the $\|f\| = 1$ constraint. It is known that the solution to this problem is the unit eigenvector, corresponding to the smallest eigenvalue of $A^T A$ [35]. Instead of finding the eigenvalues and eigenvectors of $A^T A$, singular value decomposition (SVD) can also be used (see [35] for more details on SVD).

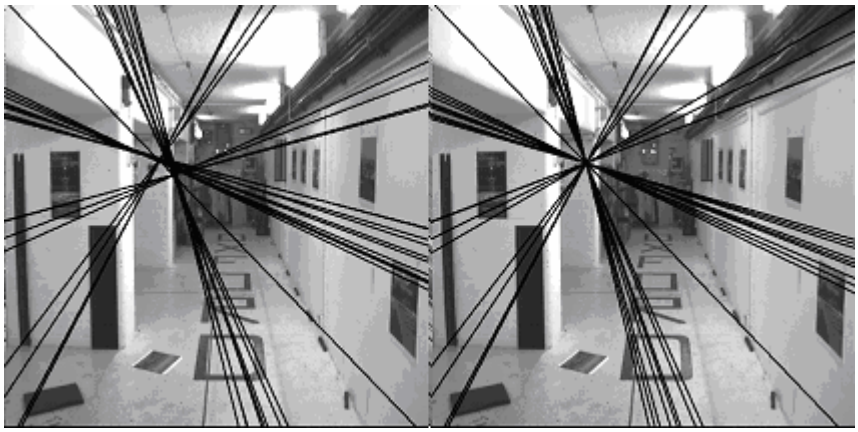


Figure 3.9: Rank of the fundamental matrix [1]: Left image shows the epipolar lines for a rank 3 fundamental matrix. Notice that the lines do not converge at a single point. In the right image on the other hand, lines coincide at a single point. Rank-2 constraint has been forced while obtaining this fundamental matrix.

The fundamental matrix is a rank-2 homogeneous matrix and fail to enforce this property to the solution might cause problems. If this constraint is not enforced, the epipolar lines will not meet at a single point and most of the algorithms should fail, since they depend on this property of the Fundamental matrix (See Figure 3.9). The linear solution of the fundamental matrix does not force this property and to correct this deficiency, one approach is to find another Fundamental matrix that is nearest to the computed solution. This problem is stated formally as,

$$\text{Minimize the Frobenius Norm } \|F - F'\| \text{ subject to } rank(F') = 2 \tag{3.3.4}$$

The solution to (3.3.4), F', is determined as:

$$\begin{aligned} &\text{If} \quad F = USV^T \text{ where } S = diag(s_1, s_2, s_3) \text{ and } s_1 \geq s_2 \geq s_3 \\ &\text{Then} \quad F' = Udiag(s_1, s_2, 0)V^T \end{aligned} \tag{3.3.5}$$

The last part of the algorithm below is called as the *constraint enforcement*, whereas the first part is the linear solution for the fundamental matrix.

Algorithm 3.3.1: 8-Point Algorithm

Given n ≥ 8 corresponding point pairs, $\mathbf{x}_1$, ..., $\mathbf{x}_n$, $\mathbf{x}_1'$, ..., $\mathbf{x}_n'$,

1. Form the rows of the A matrix from 8 point correspondences as

   $u = [\ u_1u_2, v_1u_2, u_2, u_1v_2, v_1v_2, v_2, u_1, v_1, 1\ ]^T$

2. Compute the SVD of the A matrix.

   $A = USV^T$ → f = last column of V where diagonal elements of S are in decreasing order.

3. Reshape the f matrix to its 3x3 form

4. Compute the SVD of F matrix and set the smallest element of the S matrix to zero. Recalculate the F matrix from the modified S.

   If $F = USV^T$ where $S = diag(s_1, s_2, s_3)$ and $s_1 \geq s_2 \geq s_3$

   then $F' = Udiag(s_1, s_2, 0)V^T$

3.3.2 Normalized 8-Point algorithm

Although the algorithm presented in the previous section is very simple to implement and it is linear, it is very sensitive to noise [28, 29, 21]. In order to correct this problem, a simple transformation of the utilized data has been shown to be quite useful [31]. This version of the algorithm is usually denoted as the *normalized 8-point algorithm* and its performance is shown to be quite successful [31]. Apart from the normalization part, the rest of the algorithm is same.

The performed normalization is a translation and a scaling of each image, so that the centroid of the reference points is shifted to the origin of the new coordinates and the root-mean-square (RMS) distance of the points from the origin is equal to $\sqrt{2}$.

Algorithm 3.3.2: Normalized 8-Point Algorithm

Given n ≥ 8 corresponding point pairs, $\mathbf{x}_1$, ..., $\mathbf{x}_n$, $\mathbf{x}_1'$, ..., $\mathbf{x}_n'$,

1. Normalization: Transform the image coordinates according the $\hat{x}_i = Tx_i$ and $\hat{x}'_i = T'x'_i$ where T and T′ are the normalizing matrices consisting of a translation and scaling

$$T = \begin{bmatrix} \sqrt{2}\,\mathrm{var}^{-1} & 0 & -\sqrt{2}m_x\,\mathrm{var}^{-1} \\ 0 & \sqrt{2}\,\mathrm{var}^{-1} & -\sqrt{2}m_y\,\mathrm{var}^{-1} \\ 0 & 0 & 1 \end{bmatrix}$$

   where (m_x , m_y) is the mean of the image points and var is the variance of the distances of the points to the centroid
2. Compute the F matrix using the 8-point algorithm and the transformed coordinates using Algorithm 3.3.1. Output is the F′ matrix
3. Denormalization: Compute the F matrix for the denormalized correspondences as, $F = T'^{T} F' T$

3.3.3 Outlier rejection

In Section 3.2, it is explained how to find some putative point correspondences. Although the results of the algorithm show that many correspondences still can be obtained, there also exist many outliers. Clearly, a reliable estimation of the fundamental matrix can not be achieved by using all of these correspondences. Some robust mechanism has to be used in order to get rid of the

erroneous matches and estimate the fundamental matrix more precisely.

There are many algorithms for estimating a model and the supporting set that obeys this model in the presence of outliers [32, 34, 48]. Random sample consensus (RANSAC) [32] is one of the mostly used robust estimator and for reasons to become clear in the next subsection; RANSAC is preferred for the scene reconstruction algorithm in this thesis.

3.3.3.1 Random sample consensus (RANSAC)

The organization of the RANSAC is simple and potent. In this method, some subsets of the data are selected randomly and the model is estimated by only using this small subset, recursively. The size of the random samples is usually selected as the smallest sufficient number that is required to determine the model parameters. The goodness of the model is determined by the full data set. Usually, goodness measure is the number of data points that are "consistent" with the model. The resulting best model is saved and the recursion is finished, when the likelihood of finding a better model becomes arbitrarily low, or a maximum number of iteration is reached.

The strength of RANSAC results from the fact that selecting a single random subset that is not contaminated by outliers is sufficient to find a good solution. It is noted that RANSAC can handle more than 50 % of outlier ratios depending on the complexity of the model [33].

While using RANSAC during the estimation of the fundamental matrix, an error measure is required to decide whether the points are inliers or not. There are different error measures that can be used, while one of them is being *Sampson error* [1]:

$$S_i = \frac{\left(m_{2i}^T F m_{1i}\right)^2}{\left(Fm_{1i}\right)_1^2 + \left(Fm_{1i}\right)_2^2 + \left(F^T m_{2i}\right)_1^2 + \left(F^T m_{2i}\right)_2^2} \qquad (3.3.6)$$

where $(v)_m$ representing the m^{th} entry for a column vector v.

Sampson error is the first order approximation of the reprojection error, which has a geometric interpretation, and therefore, it is quite reasonable to use this measure. The computation of the geometric error is quite complex and involves the estimation of both the model and perfect projection points. Sampson error, on the other hand, is a good approximation to it and it is easy to implement. Due to these reasons, Sampson error is used during the robust estimation of the fundamental matrix.

The selection of the random samples is also another crucial matter. The samples should be selected randomly; however they must not be close to each other. Such a situation will be useless, since the estimated model will not represent the general structure of the data. As a remedy to this problem, a regular random selection approach, based on *bucketizing,* can be employed [36]. In this method, the data set is divided into a regular grid, like *n*x*n*, and points are assigned to these *buckets*. In order to avoid selecting close points, first, 8 different buckets are selected and then one point is selected from each bucket (see Figure 3.10).

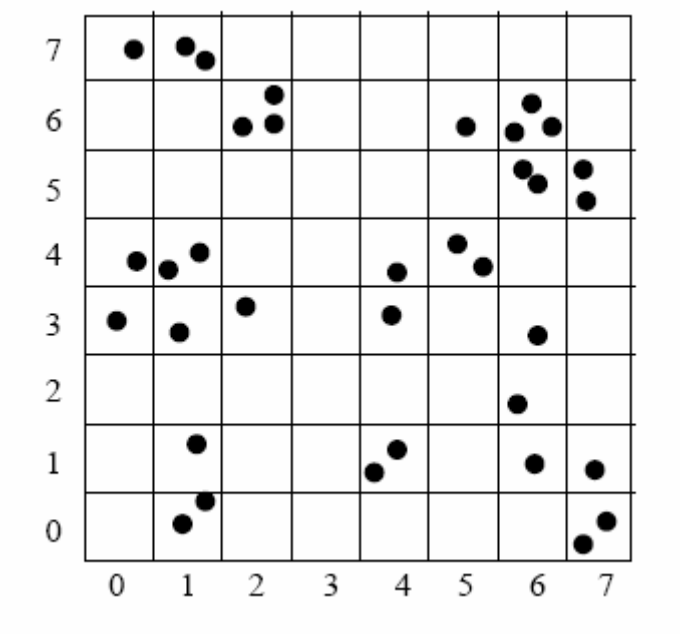


Figure 3.10: Bucketizing [36]

Some of the buckets may have more points in themselves, compared to other buckets. Therefore, their probability of selection should be higher than other buckets for the points to have equal probabilities to be selected. This can be realized in this manner: for a total of k buckets, divide [0-1] unit segment into k intervals such that the interval length is equal to the $p_i / \sum_{i=1}^{k} p_i$ where p_i is the number of data points in i[th] bucket and $\sum_{i=1}^{k} p_i$ is the total number of points. While selecting the bucket, a random number generator is used to select a number between [0-1] and the bucket containing the selected number will be marked as chosen (see Figure 3.11). For the implementation in this thesis, the grid width is selected as n=8.

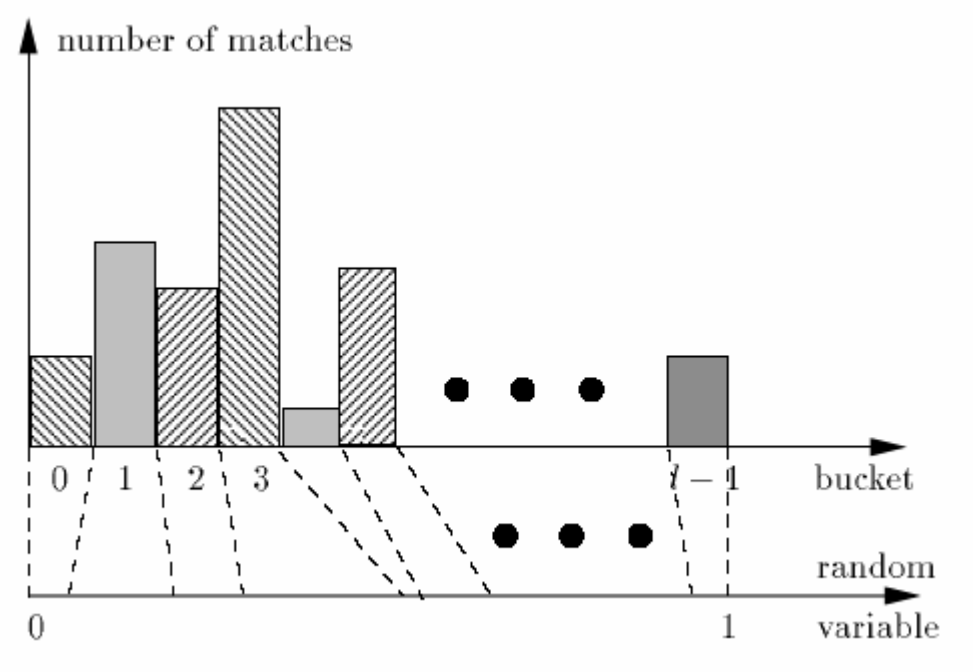


Figure 3.11: Interval and bucket mapping [36]

Another important point to mention is the number of iterations required for the RANSAC; in other words the major question is "when should the iterations stop?". The point of termination can be calculated as follows:

The number of iterations, N, is chosen sufficiently high to ensure with a probability, p, that at least one of the random samples of s points is free from outliers. Suppose e is the probability that any selected data point is an outlier (thus, w=1-e is the probability that it is an inlier). Then, N should be equal to:

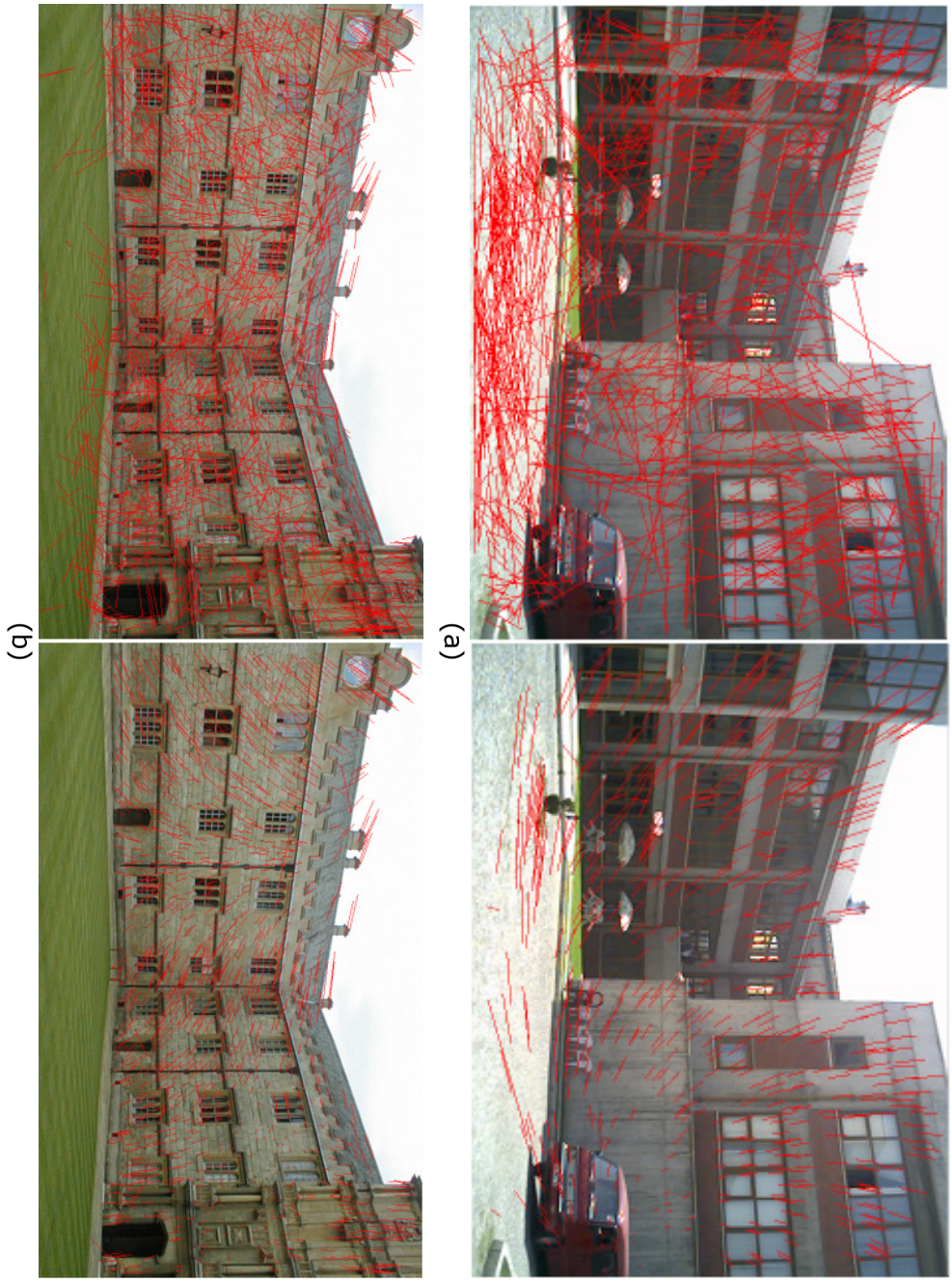
$$N = \frac{\log(1-p)}{\log(1-(1-e)^s)} \qquad (3.3.7)$$

The overall algorithm for the robust computation of the fundamental matrix can be summarized as follows:

Algorithm 3.3.3: Robust computation of the Fundamental Matrix

Repeat for N times,

1. Select a random sample of 8 correspondences and compute the fundamental matrix, F, by using the normalized 8-point algorithm given in Algorithm 3.3.2.
2. Calculate the error e by using the Sampson error (Equation 3.3.6) for each putative match for the fundamental matrix obtained e.
3. If they are below a threshold, then count them as inliers, otherwise as outliers.
4. Choose F with the largest number of inliers, and reject those pairs which yield $e > t$ for this particular F.
5. Recalculate the number of iterations N using the Equation 3.3.7
6. If number of iterations is larger than N, terminate.


(a)
(b)

(c)

Figure 3.12: Examples for RANSAC (a)*BILTEN*, (b) *Lueven Castle*, (c) *Church*: Images on the left show the motion vectors before RANSAC and images on the right show the motion vectors after RANSAC. It can be observed that RANSAC rejects outliers with a good performance.

3.3.4 Nonlinear optimization of F parameters

In the previous sections, the robust estimation of fundamental matrix is explained for a data, which is contaminated with outliers. Such a robust estimation also provides a set of data points that are consistent with the estimated model. The estimated fundamental matrix is the result of a linear algorithm and hence, the error due to the consistent data set (inliers) can be decreased in a great extent by nonlinear optimization. The Sampson error, given in Equation 3.3.6, is used once more as the error measure to be consistent with the previous step. However, during minimization, Levenberg-Marquardt (LM) algorithm [30, 35] is employed. The minimization is performed over the whole set of

inliers and the estimated fundamental matrix from the previous step is considered as the initial point. The minimized cost is the total Sampson error given as:

$$C = \sum_{i=1:N} S_i \tag{3.3.8}$$

where S_i is calculated as given in (3.3.6)

A detailed explanation on Levenberg-Marquardt algorithm can be found in Appendix C.

Table 3.2: Improvements by using subpixel accurate correspondence values and a non-linear minimization algorithm.

	Subpixel Accuracy		**Pixel Accuracy**	
	Before LM	**After LM**	**Before LM**	**After LM**
Iteration number	1000		1000	
Average Inlier Number	841		841	
Sampson Error	4.12865	0.82323	6.54395	0.92315
Sampson Error per Inlier	0.00491	0.00098	0.00778	0.00110
Epipolar constraint error power	0.10213	0.04255	0.59159	0.10056
Epipolar constraint error power per inlier	0.00012	0.00005	0.00070	0.00012

Table 3.2 shows the results of applying the LM algorithm. The experiments are performed over 10 different image pairs and the average number of inliers obtained by RANSAC after a constant

number of 1000 iterations is 841. The procedure is repeated for the coordinates both in pixel and subpixel resolution. Two different error measures are calculated: Sampson error (Equation 3.3.6) and the epipolar error (Equation 3.3.1). It can be easily observed from the table that utilization of subpixel resolution coordinates over that of pixel resolution decreases the error. Moreover, LM improves the error performance for both of them. Therefore, in this implementation, nonlinear minimization is applied with subpixel resolution coordinates during the estimation of the fundamental matrix.

3.3.5 Algorithm for robust Fundamental matrix estimation from two images

The resulting algorithm for the automatic estimation of the epipolar geometry between two image pairs by using RANSAC is obtained as follows:

Algorithm 3.3.4: F matrix computation algorithm starting from a pair of images

1. Find the interest points in each image
2. Compute a set of putative correspondences based on correlation similarity and neighborhood constraints
3. Robustly estimate the fundamental matrix:
   Repeat N times, where N is estimated according to Equation3.3.7 at each iteration
   a. Select a random sample of 8 correspondences and compute the fundamental matrix *F*, using the normalized 8-point algorithm given in Algorithm 3.3.2.
   b. Calculate the error *e* by using the Sampson error (Equation 3.3.6) for each putative match for the fundamental matrix obtained e.

c. If they are below a threshold count them as inliers, otherwise as outliers.

d. Choose F with the largest number of inliers, and reject those pairs which yield $e > t$ for this particular F.

e. If number of iterations is larger than N, terminate.

4. Nonlinear Estimation: Recalculate the fundamental matrix using all correspondences counted as inliers by minimizing the cost function given in Equation 3.3.6 by using the Levenberg-Marquardt algorithm.

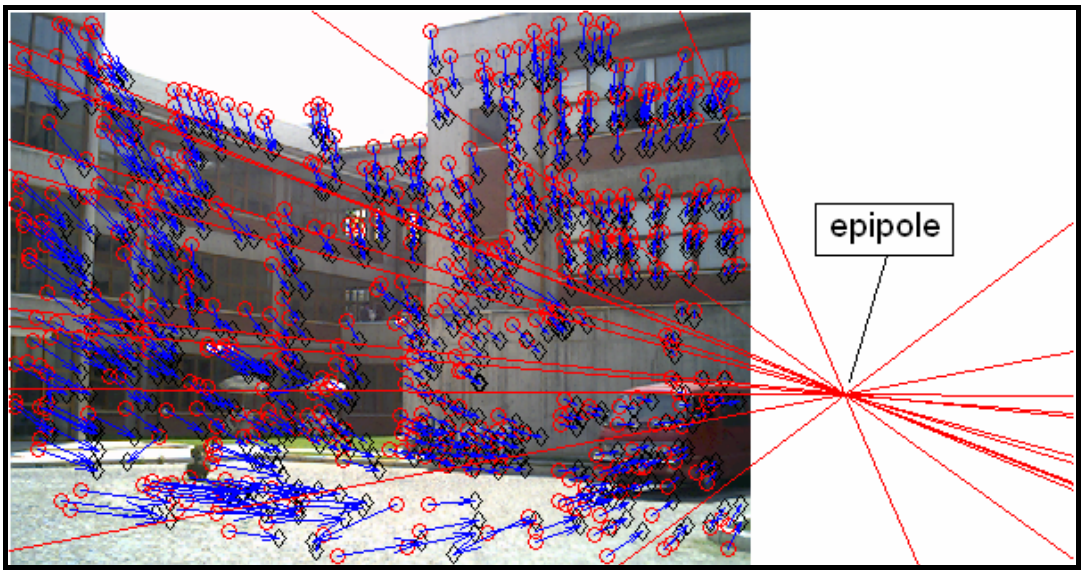


Figure 3.13: Displacement vectors between correspondence pairs and the estimated epipole of *BILTEN* image

3.4 Solving for Rotation and Translation

Two different views of a single rigid scene are related by the so-called epipolar geometry, which is described by a 3x3 singular matrix. If the intrinsic parameters of the images are known a priori, the image coordinates can be transformed into *normalized image coordinates* [1, 52], and the matrix is known as the *Essential matrix* [27, 52]; otherwise, the matrix is denoted as the

Fundamental matrix [1]. Remembering the relationship between the fundamental matrix and the essential matrix [12]:

$$E = K^T F K \tag{3.4.1}$$

where K is the camera calibration matrix, the normalization for the measured correspondences can be determined as:

$$m'_E = K^{-1} m' \text{ and } m_E = K^{-1} m \tag{3.4.2}$$

where m and m' are the real coordinates on the first and second images and m_E, m'_E are coordinates of the first and second camera matrix projected by normalized camera model.

If the first camera coordinate frame is selected as the world coordinate frame, the rotation matrix R and the translation vector t both describe the transformation of the second camera coordinate frame with respect to the first camera coordinate frame (see Figure 3.14)

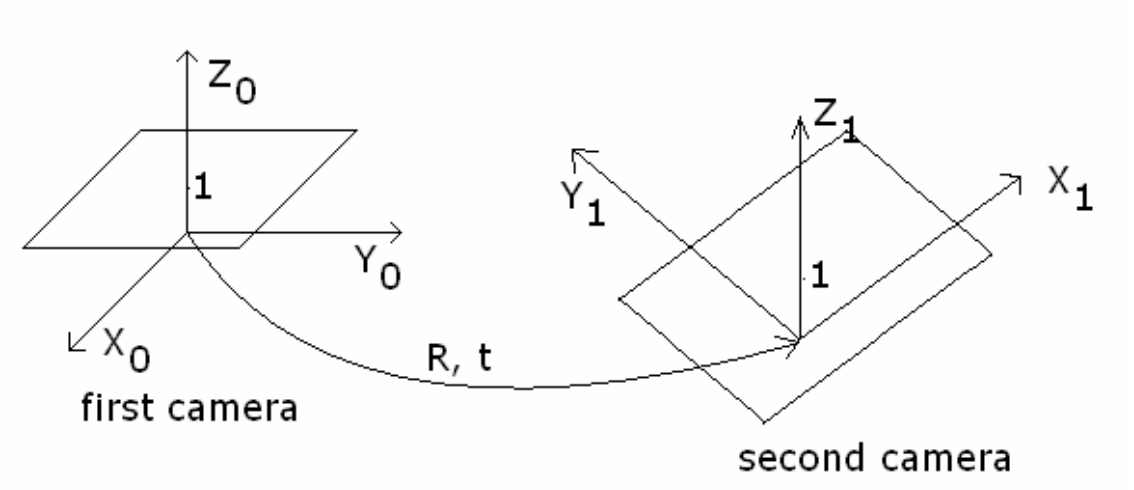


Figure 3.14 : Relative camera positions

Thus, any point $M = [M_x, M_y, M_z]^T$ with respect to the first camera coordinate frame is transformed to the point $M' = [M'_x, M'_y, M'_z]^T$ with respect to the second coordinate frame by using the relation below:

$$M' = RM + t \tag{3.4.3}$$

Then, the points are projected onto the first and second image planes by using the normalized camera model. Thus, the image points become:

$$m_E = \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} M_x / M_z \\ M_y / M_z \\ 1 \end{bmatrix} \;,\; m'_E = \begin{bmatrix} u' \\ v' \\ 1 \end{bmatrix} = \begin{bmatrix} M'_x / M'_z \\ M'_y / M'_z \\ 1 \end{bmatrix} \tag{3.4.4}$$

Combining (3.4.3) and (3.4.4), one should get,

$$M'_Z \, m'_E = M_Z R m_E + t \tag{3.4.5}$$

If $\|t\| \neq 0$, one should obtain

$$\frac{M'_Z}{\|t\|} m'_E = \frac{M_Z}{\|t\|} R m_E + t^0 \text{ where } t^0 = \frac{t}{\|t\|} \tag{3.4.6}$$

Therefore, the rotation matrix R can be calculated, if n corresponding points (m_i, m'_i) are given. In addition, if the translation vector t does not vanish, the translational direction, represented by a unit vector t^0, can also be estimated. Since only

the direction of the translation vector can be determined, the absolute 3D coordinates of the corresponding points cannot be obtained. This phenomena is called "s*caling ambiguity*" [52] and it means that only the scaled version of the scene can be determined after R and t^0 are estimated. From (3.4.3), note that M', RM and t are coplanar. So, $t \times RM$ is perpendicular to M' and hence,

$$M'(t \times RM) = 0 \text{ where } E \equiv t \times R \qquad (3.4.7)$$

3.4.1 Linear Algorithm for determining R and t

In (3.4.7), it has been shown that E is the cross product of t and R. By modifying this expression slightly, one can get the following relation:

$$E = [e_1 \quad e_2 \quad e_3] = \left[k\hat{t} \times r_1 \quad k\hat{t} \times r_2 \quad k\hat{t} \times r_3\right] \qquad (3.4.8)$$

where $\hat{t}$ is a unit vector in the direction of t, k is the unknown magnitude of t and r_i's are the column vectors of the rotation matrix R. From (3.4.8), it can be shown that [52],

$$\hat{t} \perp e_1, \hat{t} \perp e_2 \text{ and } \hat{t} \perp e_3 \qquad (3.4.9)$$

Hence,

$$\hat{t} = \mp \frac{e_i \times e_j}{\left\| e_i \times e_j \right\|} \text{ for i≠j, i,j = 1,2,3 } \& \; k^2 = 0.5 * (e_1^2 + e_2^2 + e_3^2) \qquad (3.4.10)$$

Finally, after some vector algebra, rotation matrix can be obtained as [52]:

$$r_1 = \left[\frac{1}{k^2}\hat{t}(e_2 \times e_3)\right]\hat{t} + \frac{1}{k}\left(e_1 \times \hat{t}\right) \tag{3.4.11}$$

and similar derivations can be achieved for r_2 and r_3. However, this approach is known to be extremely susceptible to errors, which makes it almost useless in a practical application.

3.4.2 Robust algorithm for determining R and t

It is known that E matrix is perpendicular to the t vector due to (3.4.9). Hence, the following relation should hold:

$$E^T \hat{t} = 0 \tag{3.4.12}$$

However, due to the noise present in the estimation of the E matrix, it is more realistic trying to find the solution for

$$\min_{\hat{t}} \left\| E^T \hat{t} \right\| \text{ subject to } \left\| \hat{t} \right\| = 1 \tag{3.4.13}$$

instead of trying to solve (3.4.12).

It is known that the solution of the optimization problem $\min_x \|Ax\|$ subject to $\|x\| = 1$ is the eigenvector associated with the smallest eigenvalue [35]. Hence, the solution for $\hat{t}$ can be determined as the unit eigenvector of $E^T E$ for smallest eigenvalue.

The rotation matrix in the presence of noise can be obtained by minimizing $\min_R \left\| R^T \left[-t_s \right]_x - E^T \right\|$ subject to "R is a rotation matrix". Instead of performing a minimization, the solution can be found using quaternion notation [52]. A matrix B is defined as,

$$B = \sum_1^3 B_i^T B_i \text{ where } B_i = \begin{bmatrix} 0 & (C_i - D_i)^T \\ (D_i - C_i) & \left[D_i + C_i \right]_x \end{bmatrix}$$
$$\text{such that } C = \left[-t_s \right]_x \text{ and } D = E^T \qquad (3.4.14)$$

Then, the eigenvector (q) associated with the minimum eigenvalue of the B matrix is the optimal quaternion. Using this quaternion, R can found as [52],

$$R = \begin{bmatrix} q_0^2 + q_1^2 - q_2^2 - q_3^2 & 2*(q_1 q_2 - q_0 q_3) & 2*(q_1 q_3 + q_0 q_2) \\ 2*(q_1 q_2 + q_0 q_3) & q_0^2 - q_1^2 + q_2^2 - q_3^2 & 2*(q_2 q_3 - q_0 q_1) \\ 2*(q_1 q_3 - q_0 q_2) & 2*(q_3 q_2 + q_0 q_1) & q_0^2 - q_1^2 - q_2^2 + q_3^2 \end{bmatrix}$$
$$\text{where } q = \begin{bmatrix} q_0 & q_1 & q_2 & q_3 \end{bmatrix}^T \qquad (3.4.15)$$

The linear algorithm, although theoretically correct, does not always yield correct estimates of rotation and translation due to the noise in the E-matrix estimate. Therefore, the robust algorithm is usually preferred in any scene reconstruction algorithm.

3.5 Finding the location of 3D points

One of the most important stages in structure estimation is the triangulation step, in which the position of a point in 3-D, is tried to be estimated from point correspondences. This section describes the methods for computing the position of a point in 3-D

coordinates, given its projection in two views and the respective camera projection matrices. It is assumed that the fundamental matrix is estimated up to a good accuracy and there are errors in the corresponding images of the points. Under these assumptions, the back-projected rays should not meet at a single point in 3-space in general and therefore, simple triangulation might not give good results. It is therefore necessary to employ noise resistant techniques to estimate the position of a point in 3-space. Apart from noise, the calibration parameters of the camera are not always available in the reconstruction step and in order to build up the data necessary for the automatic calibration, projective (or affine) invariant depth values are also necessary [1, 54]. Hence, it is another important property of the triangulation method to be projective (or affine, which ever the reconstruction is) invariant.

In the following sections, most common triangulation methods are examined and compared. These methods can be classified into 4 major groups: *midpoint method* [56], *linear methods* [2], *iterative linear methods* [55] and finally, *polynomial triangulation method* [55].

3.5.1 Problem Definition:

It is assumed that the fundamental matrix (F) from which camera matrices can be constructed, are known with great accuracy and the computed matching points are assumed to be noisy.

The epipolar relationship

$$x'^{T} Fx = 0 \tag{3.5.1}$$

must be satisfied, if there is a point, X, in 3D space, such that $x = PX$ and $x' = P'X$. Since it is assumed that the measured image points are noisy, back-projected rays will not intersect at a point in 3-D space in general.

Denoting a triangulation method, which is used to compute a 3-D point, by T, X is represented with

$$X = T(x, x', P, P') \tag{3.5.2}$$

A method is said to be invariant under transformation H, if

$$T(x, x', P, P') = H^{-1}T(x, x', PH^{-1}, P'H^{-1}) \tag{3.5.3}$$

It is desired to have a triangulation method that is invariant under the appropriate class of transformations in which the reconstruction is to be performed. For example, for the case of projective reconstruction, it is not very suitable to minimize 3D errors, since distance measures are not preserved in a projective coordinate system. The solutions for such minimizations should be different for the every projective reconstruction that is considered [1]. Instead of dealing with this large set of different reconstructions, it is more rational to minimize a geometric cost function that is invariant to the desired level of transformations. The reprojection error cost function:

$$\Gamma = d(x,\hat{x})^2 + d(x',\hat{x}')^2 \quad \text{subject to the constraint } \hat{x}'^T F\hat{x} = 0 \quad (3.5.4)$$

In the following sub-sections, major triangulation methods in the literature are described and lastly, a projective invariant method is also presented.

3.5.2 Midpoint Method:

A popular approach for triangulation is selection of the midpoint of the common perpendicular to the back-projected rays of the matched points (see Figure 3.15) [56]. This method behaves worst under projective and affine transformations, since "perpendicularity" is not an affine and "midpoint" is not a projective concept [55]. Hence, it should be used only for the Euclidean reconstruction problems.

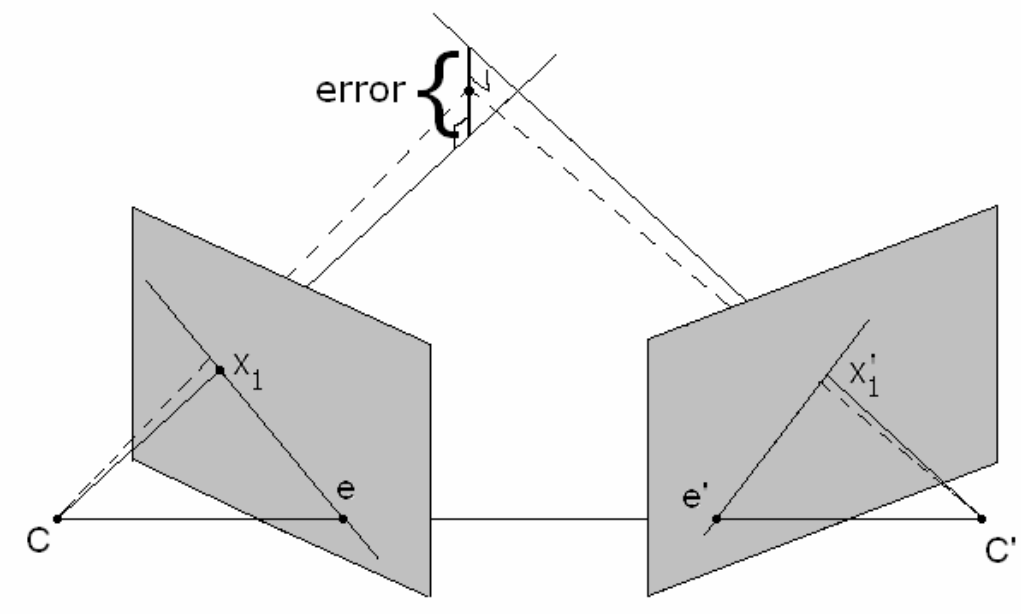


Figure 3.15: Midpoint method

The back-projection of the points to rays can be calculated from the two points that are on the ray: camera center C and the point $P^{+}x$, where P^{+} is the pseudo-inverse of the projection matrix P. The pseudo-inverse is calculated as $P^{+} = P^{T}(PP^{T})^{-1}$ for which $PP^{+} = I$. The point $P^{+}x$ should be on the ray, since it projects to the image point x. Then, joining these two points forms the ray:

$$X(\lambda) = P^{+}x + \lambda C \tag{3.5.5}$$

Once, two ray equations are obtained, the midpoint at which the lines are closest to each other are taken as the solution.

3.5.3 Linear Triangulation Methods:

Linear triangulation method [1, 54] is the most common method due to its ease in implementation. Consider the projection equation $m = PM$ where $m = w(u \quad v \quad 1)^{T}$ with (u,v) are the observed point coordinates and w is the unknown scale factor. If the ith row of the projection matrix is denoted as p_i^{T}, the relation $m = PM$ can be written as,

$$\begin{aligned} wu &= p_1^{T}M \\ wv &= p_2^{T}M \\ w &= p_3^{T}M \end{aligned} \tag{3.5.6}$$

and rearranging

$$\begin{aligned} (up_3^{T} - p_1^{T})M = 0 \\ (vp_3^{T} - p_2^{T})M = 0 \end{aligned} \tag{3.5.7}$$

The corresponding pixel to m on the other image will result another set of equations similar to (3.5.7). The problem now can be stated as,

$$\text{Find } M \text{ such that } AM = 0 \text{ where } A = \begin{bmatrix} up_3^T - p_1^T \\ vp_3^T - p_2^T \\ u' p'^T_3 - p'^T_1 \\ v' p'^T_3 - p'^T_2 \end{bmatrix}$$

$$\text{with } P = \begin{bmatrix} p_1^T \\ p_2^T \\ p_3^T \end{bmatrix} \text{ and } m = w\begin{bmatrix} u \\ v \\ 1 \end{bmatrix}, m' = w'\begin{bmatrix} u' \\ v' \\ 1 \end{bmatrix} \tag{3.5.8}$$

A non-zero solution to this problem can be found in various ways.

3.5.3.1 Linear-Eigen Method:

The solution to the "$AM = 0$ problem" cannot be found exactly due to the noise present in the A matrix and hence, some cost function should be defined. In the Linear-Eigen method, M is determined from the well-known $\|AM\| = 0$ subject to $\|M\| = 1$ optimization. The solution to this problem is the unit eigenvector corresponding to the minimum eigenvalue of the $A^T A$ matrix [35].

Although, this method is quite easy to implement, it is not suitable for projective or affine reconstructions. This case can be observed by applying a transformation H to the camera matrices such that P and P' are transformed to PH^{-1} and $P'H^{-1}$. In this case, A becomes AH^{-1} and a point M is then equivalent to a point HM in

the sense that they will give the same errors ($AM = e$ and $AH^{-1}HM = e$). However, the condition $\|M\| = 1$ is not invariant under projective or affine transformations. Hence, linear-eigen method is not projective or affine invariant in general.

3.5.3.2 Linear Least Squares Method:

Linear LS method solves the $AM = 0$ problem by fixing the fourth parameter of M vector to 1. In this approach, $AM = 0$ relation is transformed into a "4 equations, 3 unknowns" problem. A solution to this over-determined problem can be obtained by using pseudo-inverse or SVD [35].

This method assumes that the solution is not on the plane at infinity by setting the fourth parameter to 1. This assumption becomes a problem for the projective reconstruction, where points can be on the plane at infinity. Apart from the points on the plane at infinity, this method is also not suitable for the projective reconstruction, since $[x, y, z, 1]^T$ is not invariant under a projective transformation H. On the other hand, since the affine transformation does not change the plane at infinity, $[x, y, z, 1]^T$ is invariant to affine transformations. Hence, the linear LS method is affine invariant.

3.5.4 Iterative Linear Triangulation Methods:

Linear triangulation methods minimize $\|AX\|$ which do not have any geometric meaning at all. Due to this fact, some inaccuracies might occur in the results. By weighting the rows of the A matrix,

however, a better solution can be obtained [55]. Iterative linear methods, tries to find the solution by changing the weights of the A matrix in (3.5.8) adaptively, so that the adapted A matrix gives a measure of a geometric error function.

It can be shown that, by properly weighting the A matrix, the iterative procedure will be equal to the minimization of the cost function in (3.5.4). In the solution of the $BAX = 0$, both of the linear-eigen and linear LS solutions can be used and the corresponding methods are named as *Iterative Eigen* and *Iterative LS,* respectively. Details for these methods can be found in [55].

These methods are more easy to implement, as well as do not need a separate initialization algorithm and have a simple stopping criteria, compared to the other iterative least squares minimization algorithms, such as Levenberg-Marquardt [30]. However, like most of the algorithms that include iteration, there is no guarantee for convergence and these methods fail to converge about 5% of the time [55]. Although, these methods are not projective invariant, it is stated in [55] that they are quite insensitive to projective transformations.

3.5.5 Polynomial Triangulation

The noisy point matches in general will not satisfy the epipolar constraint and therefore their back-projected rays will not form a single 3-D point in space. However, in [55] it is shown that, by defining a cost function, which minimizes the reprojection error, an optimal solution can be found. It is also possible to reach this

optimal solution by using a nonlinear optimization method, such as Levenberg Marquardt for the cost function given in (3.5.4). By reformulating the problem, however, polynomial triangulation method minimizes this reprojection error in a non-iterative manner.

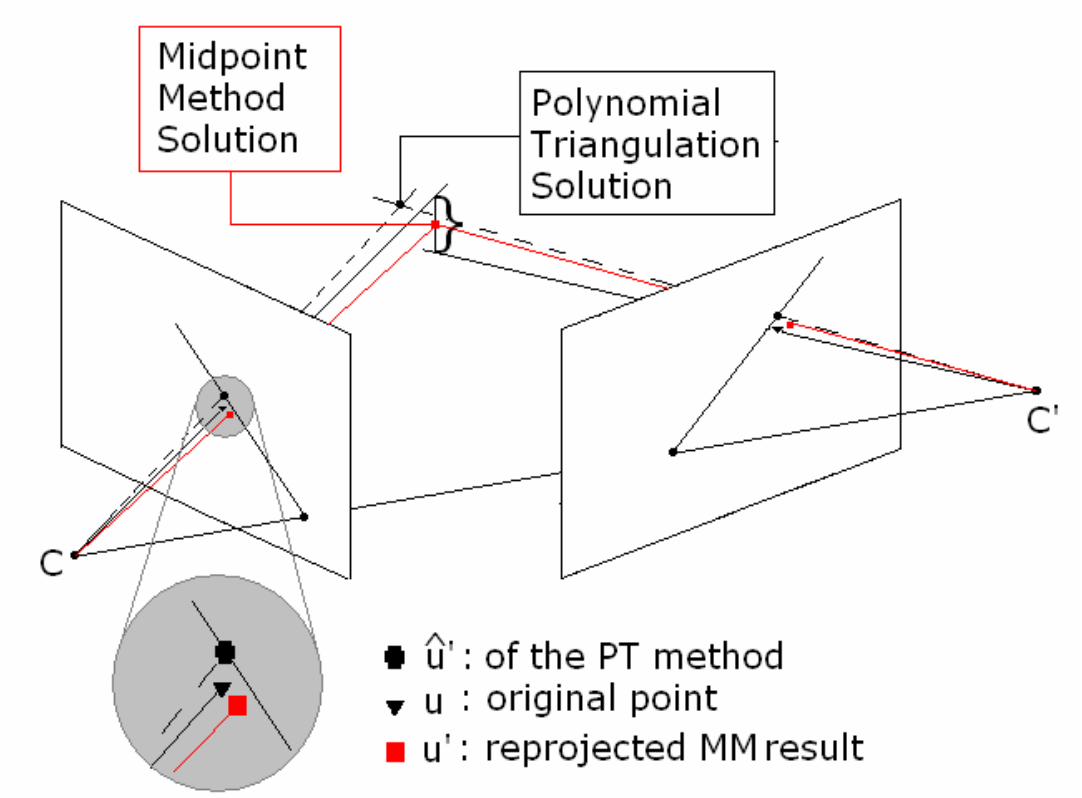


Figure 3.16: Polynomial Triangulation (PT): PT finds the closest points on the pencil of epipolar lines and estimates the location of the 3D point using these points. Midpoint method (MM), on the other hand, minimizes the 3D error by selecting the midpoint of the closest point of back-projected rays.

In this method, the problem is reduced to finding the roots of a 6^{th} degree polynomial in one variable by parameterizing the pencil of epipolar lines. The method then finds the pair of matched epipolar lines closest to the given pair of point matches. After the closest epipolar lines are determined, the closest points to the matched points on these lines are selected and the 3D point in space is

calculated by using these matches which satisfy the epipolar constraint exactly. Since these points satisfy the epipolar constraint, their back-projected rays meet in space at a single point.

The method is projective and affine invariant, since it minimizes a cost function, which is invariant under projective and affine transformations. Moreover, the method is provably optimal in the sense that under the assumption of a Gaussian noise model, the most probable reconstruction is the one that minimizes the reprojection error and polynomial triangulation exactly minimizes this cost function [55].

3.5.5.1 Reformulation of the minimization problem:

For a given pair of correspondences $u \leftrightarrow u'$, one should seek for $\hat{u} \leftrightarrow \hat{u}'$ in order to minimize the reprojection error given in (3.5.4), such that $\hat{u}'^T F\hat{u} = 0$. Since the points satisfying the epipolar constraint must lie on the epipolar lines, the cost function definition may be modified without making any change in the final output:

$$\text{Minimize } d(u,\lambda)^2 + d(u',\lambda')^2 \tag{3.5.9}$$

where λ and λ' are chosen from the all possible epipolar lines. If the line equations that minimize the above error given in (3.5.9) are obtained, then the points $\hat{u} \leftrightarrow \hat{u}'$ can be found easily by projecting the original pair to their respective lines. The algorithm is thus obtained as follows:

Algorithm 3.5.1: Polynomial triangulation algorithm

1. Parameterize the pencil of epipolar lines of the first image as a function of a single variable, i.e., $\lambda(t)$
2. Find the corresponding epipolar line by using the fundamental matrix F, i.e, $\lambda'(t)$
3. Express the distance function $d(u,\lambda(t))^2 + d(u',\lambda'(t))^2$ as a function of t.
4. Find the value of the t which minimizes the cost function.

The above minimization problem can be solved non-iteratively by rearranging the terms of the cost function. In the end, the minimizer of this cost function can be obtained by solving a 6^{th} degree polynomial.

3.5.5.2 Details of minimization

By applying a rigid transformation in order to place the correspondences to the origin and shifting the epipoles to $(1,0,f)^T$ and $(1,0,f')^T$, one may simplify the cost equation without changing the result. However, the fundamental matrix has to be compensated for the rigid transformation (i.e., $F(1,0,f)^T = (1,0,f')F = 0$). In order to move the origin to the correspondence pixel locations, the pixel is transformed by,

$$L = \begin{bmatrix} 1 & 0 & -u_0 \\ 0 & 1 & -v_0 \\ 0 & 0 & 1 \end{bmatrix} \qquad (3.5.10)$$

where (u_o,v_0) is the correspondence point location.

Similarly, in order to rotate the images such that the epipoles are on the respective x-axes, a rotation in the form of

$$R = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.5.11}$$

is applied. Corresponding rotation angles θ are found from the equality,

$$RLe = (1,0,f)^T \tag{3.5.12}$$

By developing the left-hand side, an equation for θ can be found as:

$$\sin(\theta)(e_1 - e_3u_1) + \cos(\theta)(e_2 - e_3u_2) = 0 \tag{3.5.13}$$

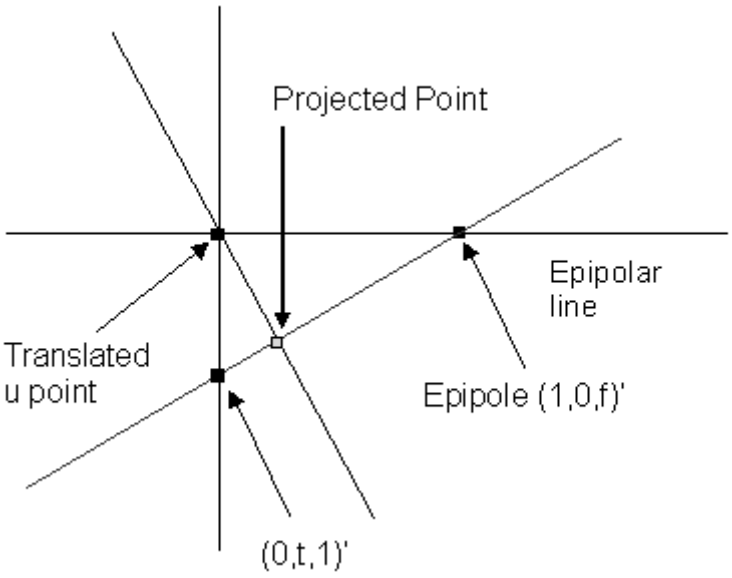


Figure 3.17: Polynomial triangulation

An overall transformation of $T = RL$ and $T' = R'L'$ is applied to the correspondence pairs u and u', respectively. After applying these

transformations, however, fundamental matrix has to be adapted as well. The transformation applied to the fundamental matrix is $F = T' F_0 T^{-1}$ where F_0 denotes the original matrix before carrying out transformations T and T'. The final fundamental matrix becomes,

$$F = \begin{bmatrix} ff'd & -f'c & -f'd \\ -fb & a & b \\ -fd & c & d \end{bmatrix} \tag{3.5.14}$$

Consider a point $(0,t,1)^T$; the epipolar line passing through this point is found by $(0,t,1)^T x(1,0,f)^T = (tf,1,-t)^T$ and the corresponding epipolar line in the second image is obtained by $F(0,t,1)^T = (-f'(ct+d), at+b, ct+d)^T$. Hence, the cost of this point is

$$\Gamma = \frac{t^2}{1+(tf)^2} + \frac{(ct+d)^2}{(at+b)^2 + f'^2(ct+d)^2} \tag{3.5.15}$$

In order to find the minimum value for this function, one should take its derivative and equate it to zero. The derivative of (3.5.15) is equal to:

$$\frac{\delta \Gamma}{\delta t} = \frac{2t^2}{(1+(tf)^2)^2} - \frac{2(ad-bc)(at+b)(ct+d)}{((at+b)^2 + f'^2(ct+d)^2)^2} = 0 \tag{3.5.16}$$

Rearranging the terms,

$$\begin{aligned} r(t) = {} & t((at+b)^2 + f'^2(ct+d)^2)^2 \\ & -(ad-bc)(a+(tf)^2)^2(at+b)(ct+d) \\ = {} & 0 \end{aligned} \tag{3.5.17}$$

The final equation is a sixth degree polynomial of a single variable. By solving the roots of this polynomial, one can find up to six different real roots. The roots of a polynomial can be obtained by calculating the eigenvalues of the companion matrix. The real root giving the minimum error according to (3.5.15) is selected as the minimizer, t_0. Then, for finding the closest points on these lines to the points u and u', the origin (since the images are transformed in order to place the points to the origin) is projected onto the epipolar lines $\lambda(t_0)$ and $\lambda'(t_0)$.

The projection of (0,0) onto the line $\lambda(t_0)$ to find $\hat{u}$ is calculated by,

$$\hat{u}_x = \frac{ft_0^2}{1+f^2t_0^2} \quad \text{and} \quad \hat{u}_y = \frac{t_0}{1+f^2t_0^2} \quad \text{where } \lambda(t_0) = (ft_0, 1, -t_0)^T \quad (3.5.18)$$

Next, the projection of (0,0) onto the line $\lambda'(t_0)$ to find $\hat{u}'$ is calculated by,

$$\hat{u}'_x = \frac{-l_1 l_3}{l_1^2 + l_2^2} \quad \text{and} \quad \hat{u}'_y = \frac{-l_2 l_3}{l_1^2 + l_2^2} \quad \text{where } \lambda'(t'_0) = F\begin{bmatrix} \hat{u}_x \\ \hat{u}_y \\ 1 \end{bmatrix} \quad (3.5.19)$$

where (l_1, l_2, l_3) denotes the line parameters.

The resulting point coordinates are obtained, according to the transformed coordinate systems. In order to find the actual point locations, T^{-1} and T'^{-1} transformations are applied to the

calculated points. Finally, after the $\hat{u}$ and $\hat{u}'$ points are determined, linear-eigen triangulation method is applied in order to find the 3D object point. Since the $\hat{u} \leftrightarrow \hat{u}'$ points satisfy the epipolar constraint exactly, their back-projected rays must meet in space at a single point. This step concludes the polynomial triangulation algorithm.

3.5.6 Simulations on Triangulation Algorithms

Among the presented algorithms in the previous sections, four of them are tested for evaluating their performance against projective and Euclidean reconstructions under additive Gaussian noise. The utilized methods are polynomial triangulation, midpoint method, linear-eigen and linear least-squares methods. For different levels of additive Gaussian noise, median of the reprojection error powers are calculated. The results are given in Figure 3.18 and Figure 3.19.

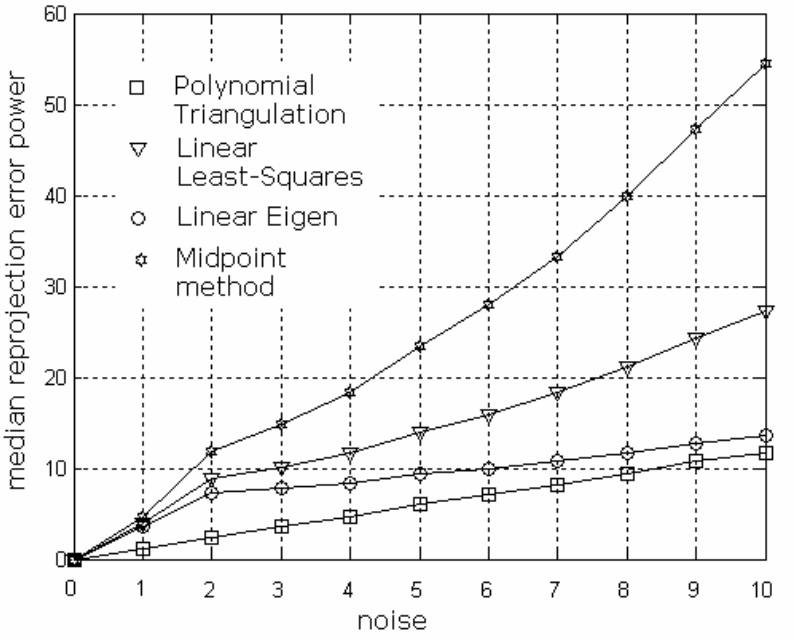


Figure 3.18: Reprojection error for projective reconstruction

The tests are performed over synthetic data and to measure the invariance of the method, a projective transformation is applied to each camera matrix. The projective transformation is chosen so that first camera projection matrix is of the form $[I \mid 0]$. This is a significant distortion, since the normal projection matrix is of the form $[K \mid 0]$ where K is the calibration matrix. It is observed from Figure 3.18 that polynomial triangulation behaves best under projective transformation. On the other hand, midpoint method gives the worst results and should be avoided. In Figure 3.19, almost all of the methods behave equally and can be used alternatively for Euclidean reconstruction problems.

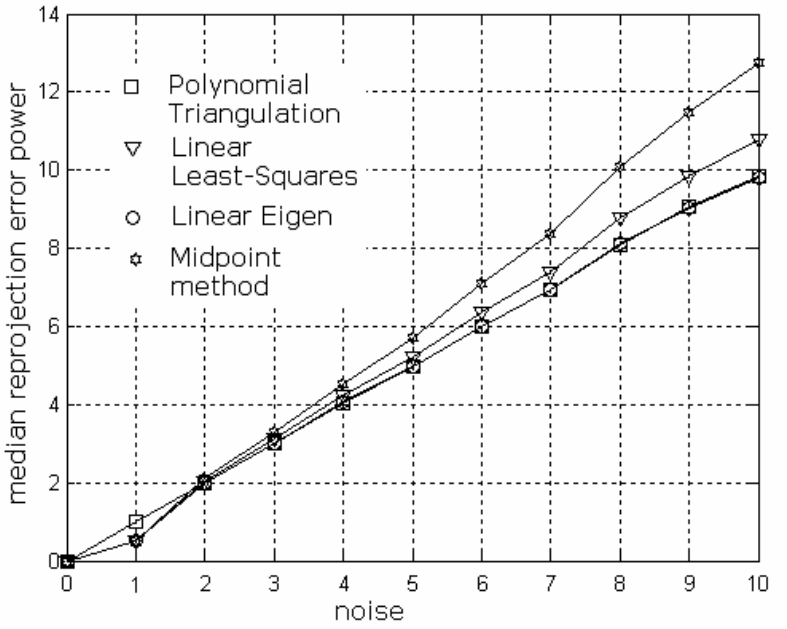


Figure 3.19: Reprojection error for Euclidean reconstruction

3.6 Simulation results

The complete algorithm, which takes two calibrated images as input to return 3-D locations for the automatically found correspondences, is tested with various types of images for very different camera matrices. Some of the results captured from the VRML output illustrate the performance of 3-D reconstruction. In all the figures below, (a) and (b) present, the input images, whereas (c), (d), and (e) are the top, frontal and side views, respectively (see Figure 3.20).

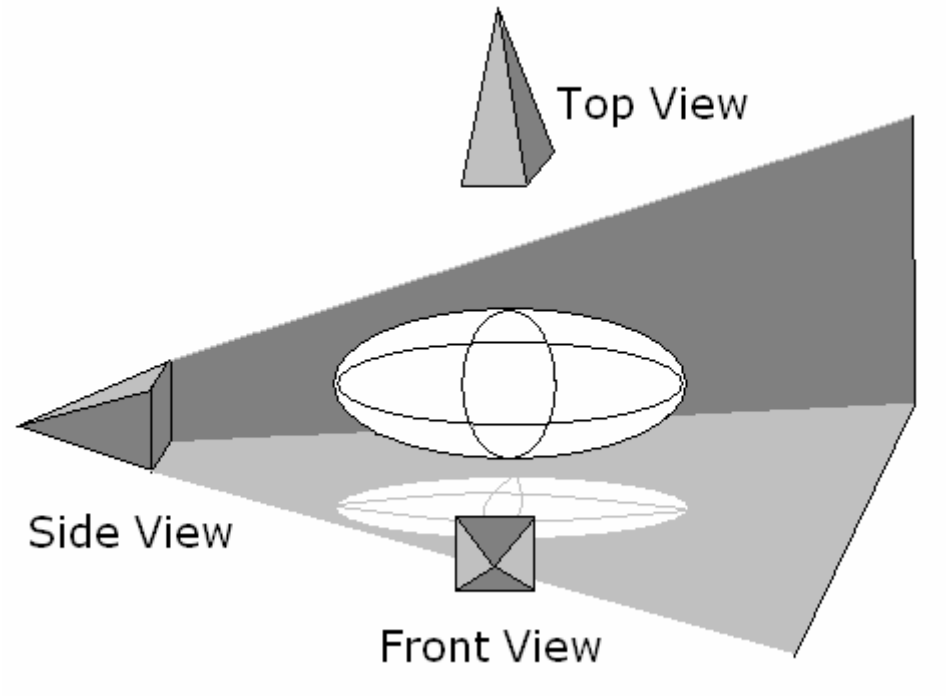


Figure 3.20: Viewing angles: Triangle prisms denote the camera locations and orientations

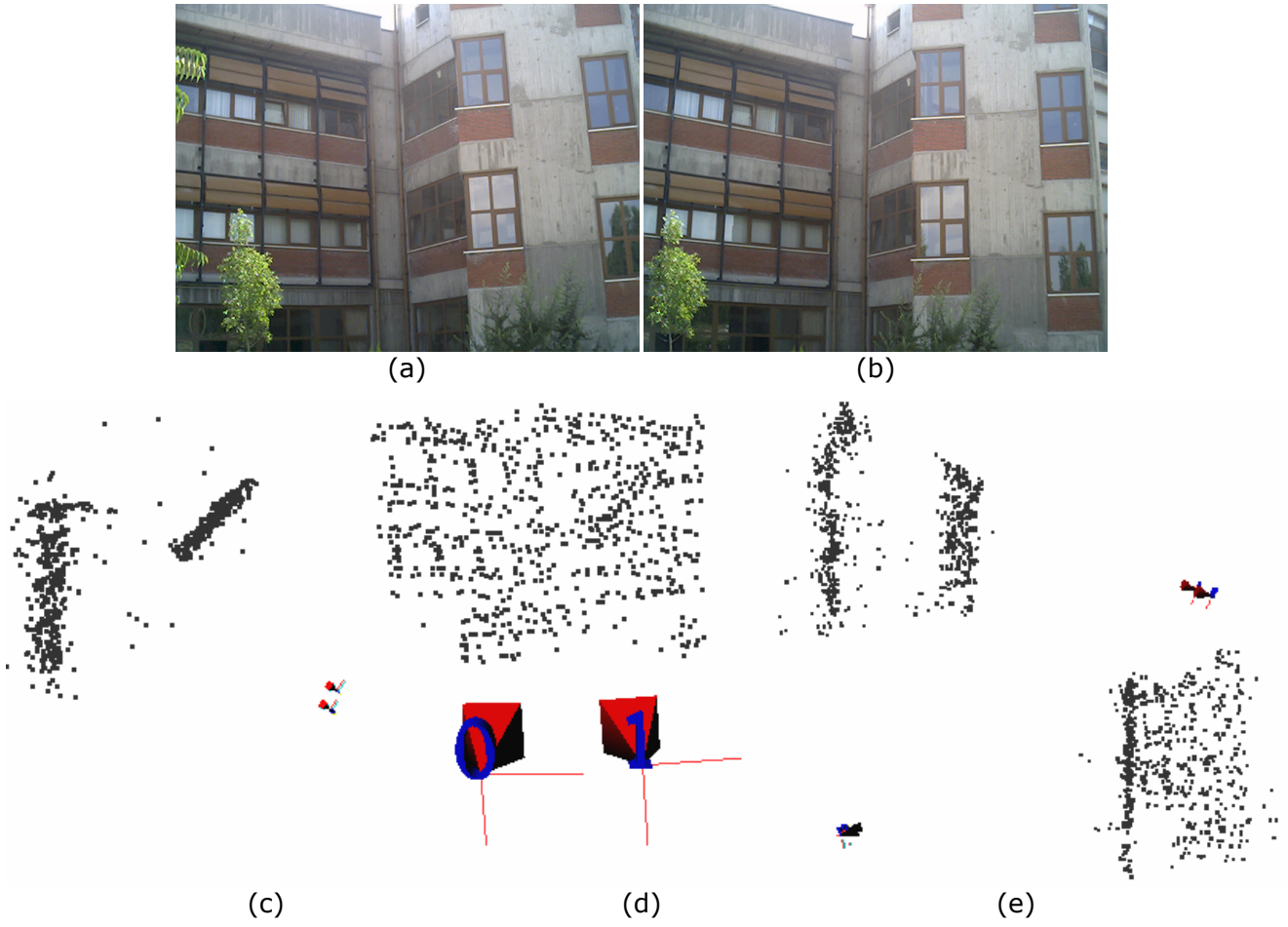


Figure 3.21: 3-D reconstruction results for Bilten data

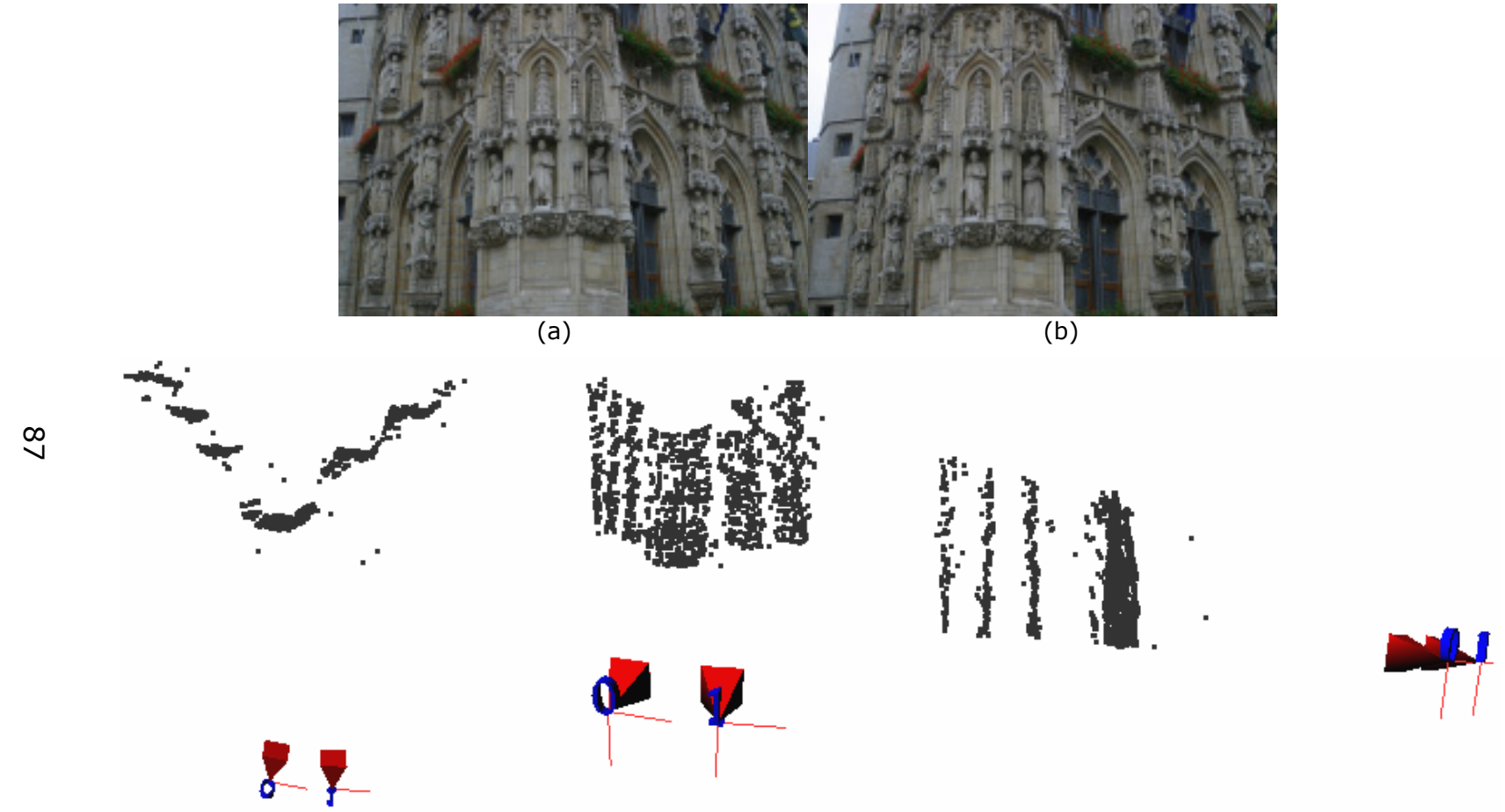


Figure 3.22: 3-D reconstruction results for Cityhall Sequence[58]

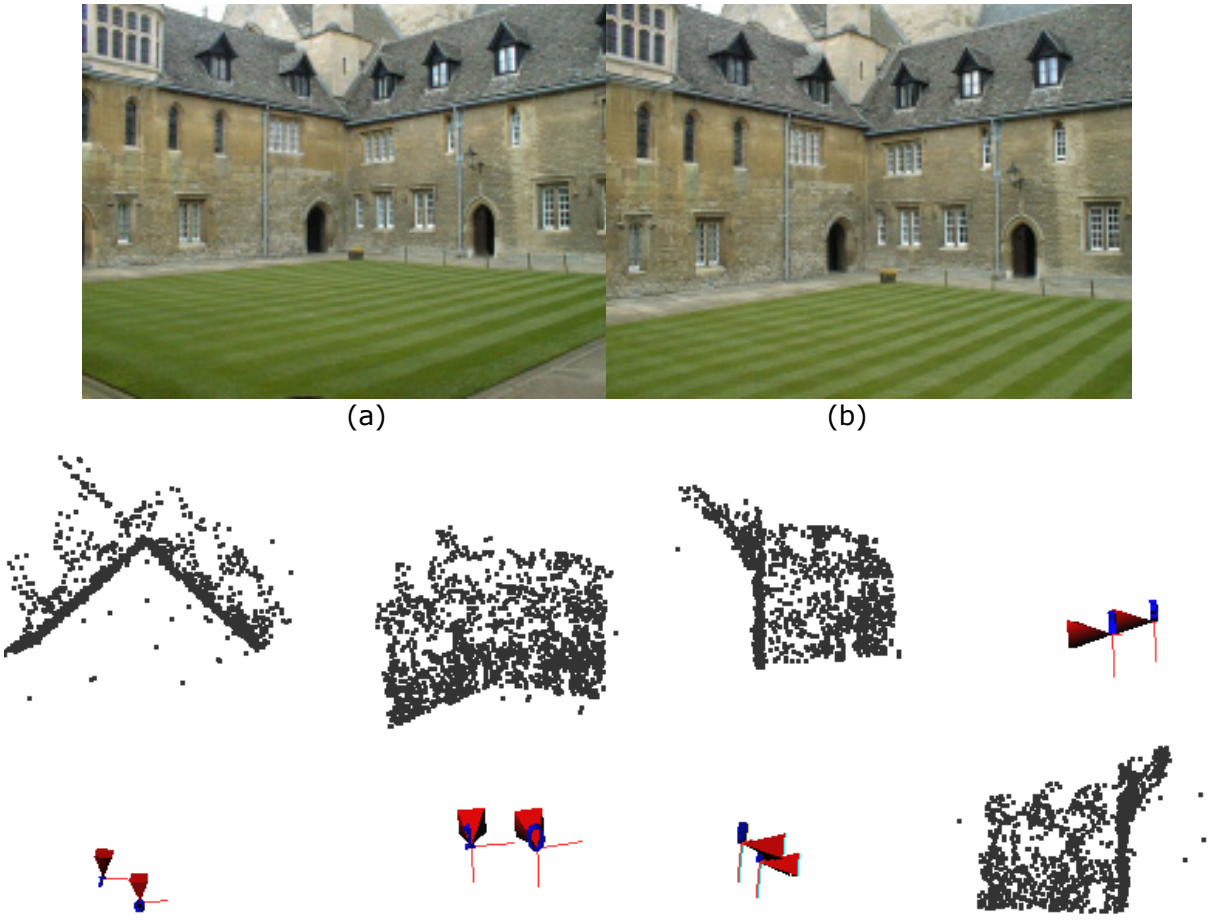


Figure 3.23: 3-D reconstruction results for Merton College[59]

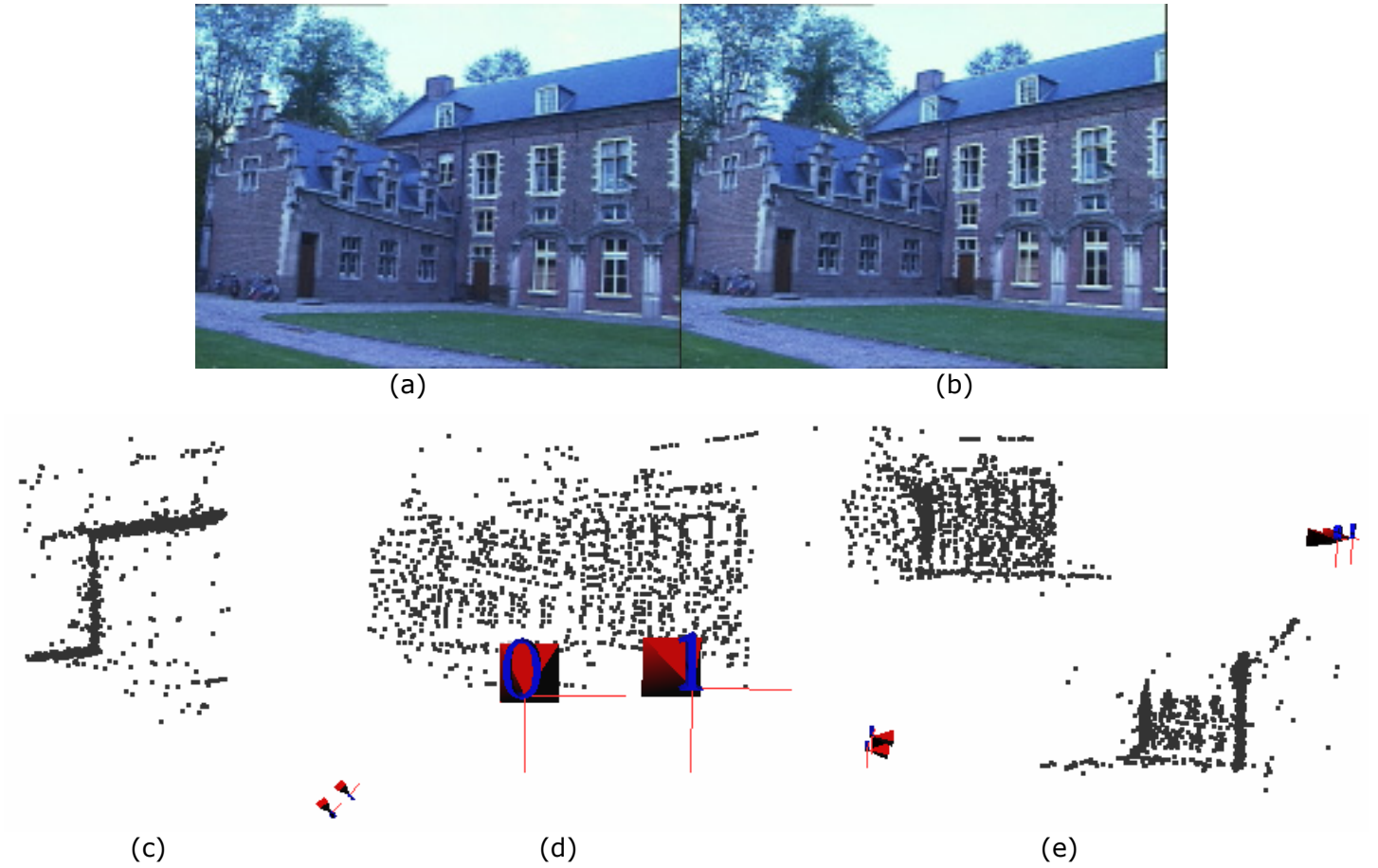


Figure 3.24: 3-D reconstruction results for Leuven Castle [24]

CHAPTER 4

3D RECONSTRUCTION FROM MULTIPLE VIEWS

The estimation of the 3D model of a scene is an ongoing research topic in computer vision. There are many applications of this research in robot navigation, visual automation, virtual reality and computer graphics. The aim of obtaining accurate models of a scene from, not only frame pairs but also a sequence of images has always obtained much attention. The method by Tomasi and Kanade [39] uses an affine factorization algorithm to extract the structure of the scene from image sequences. The most important restriction of the algorithm is that it makes an orthographic projection assumption. Beardsley et al. [38] and Pollefeys et al. [37], on the other hand, employ a sequential algorithm to extract and update a projective reconstruction of a scene. In these sequential algorithms, for every new frame, the location and orientation of the scene with respect to an initial reconstruction is re-calculated and some new 3-D points are initialized. In this way, the final structure and motion information is built up gradually. While the first approach [37] computes a projective reconstruction, the latter one [38] upgrades the structure to metric.

In this chapter, an iterative algorithm [37] to reconstruct a scene from several images is presented. The simplest case of this problem is the two view case, which is explained in the previous chapter. The problem might be defined as the process for combining information, which is gathered from images captured at different locations, orientations and even different viewing parameters. In order to find a solution, the following assumptions are made: the camera parameters of the images are known a priori in all of the images and the scene is completely stationary.

The algorithm starts with the initial reconstruction of a scene from two images in order to obtain a common structure. Next, the position and orientation for the further views is computed in this setup. At the addition of a new frame, the initial reconstruction is refined and upgraded. In this manner, the pose of the views that do not have any common features with the initial reconstruction can be calculated. After the estimation of motion and structure for all of the sequence frames, the estimation is further refined by using a procedure entitled, as *bundle adjustment* [40, 41].

4.1 Initial structure computation

The initial reconstruction step produces an initial framework that is used to build upon all other views. Two frames are chosen from the sequence and reconstruction is performed, as it is explained in the previous chapter. The reconstruction frames must be general enough to be compatible with other views. These frame pairs must not be formed of frames, containing dominant planes or rotation-only-configurations. For such degenerate cases, the reconstruction

might fail. The initial 3-D structure computation algorithm is already presented in the previous chapter.

4.2 Addition of a new view

In the previous section, the initial reconstruction is briefly explained. This section explains how to add a new view to the framework. First of all, the pose of the new view is detected and then new structure points are initiated to update the reconstruction through triangulation.

4.2.1 Pose estimation

The pose of the new frame with respect to the current framework can be obtained by utilizing the correspondences of the new view with a previous view and the structure points.

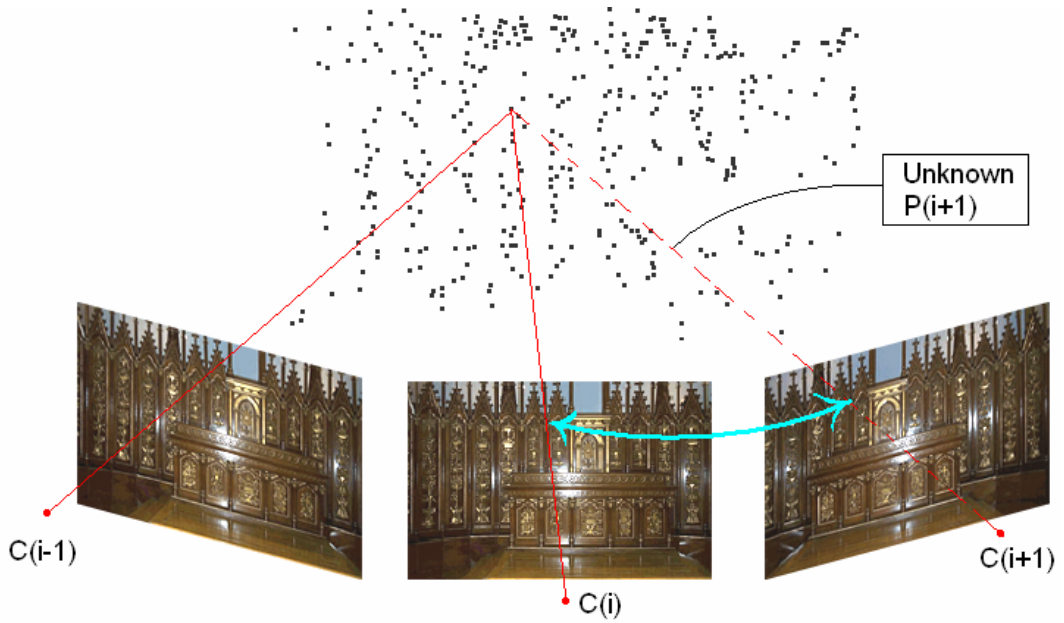


Figure 4.1: Pose estimation: 3D-2D correspondences are obtained by using the relation between the structure and the correspondences estimated from frames f_i and f_{i+1}.

First of all, the epipolar geometry between the new view and a previously inserted view is obtained by using the robust technique, which is explained in Section 3.2. As a next step, 2-D points, whose 3-D structure points are already calculated, are selected from the obtained correspondence set (see Figure 4.1).

From the above figure, it is observed that, during the addition of a new frame f_{i+1}, if a correspondence point between f_{i+1} and f_i is also matched to a point in the frame f_{i-1}, then one can form a set of points composed of 3D–2D projection pairs for f_{i+1}, since the location of the structure point associated to this point has already been calculated in the previous iteration by the relation between f_{i-1} and f_i. In this way, the projection information for the new frame can be calculated by a number of points with such property. The projection matrix of this new frame,f_{i+1} , is calculated by using a robust algorithm, similar to the one used in the computation of the fundamental matrix.

4.2.1.1 Computation of the projection matrix from 3D–2D correspondences

The relation between the elements of a projection pair $x_i \leftrightarrow X_i$ is given by the following relation:

$$m_i = PM_i \text{ where } \quad m_i = \begin{bmatrix} u_i \\ v_i \\ w_i \end{bmatrix} \text{ and } M_i = \begin{bmatrix} X_i \\ Y_i \\ Z_i \\ 1 \end{bmatrix} \qquad (4.2.1)$$

By rearranging (4.2.1), one can obtain

$$\begin{aligned} u_i &= P_1^T M_i \\ v_i &= P_2^T M_i \quad \text{where } P_k^T \text{ is the k}^{\text{th}} \text{ row of the P matrix} \\ w_i &= P_3^T M_i \end{aligned} \tag{4.2.2}$$

It is known that image plane coordinates of the *m* vector is obtained by the relations

$$x_i = \frac{u_i}{w_i} \text{ and } y_i = \frac{v_i}{w_i} \tag{4.2.3}$$

By using the relations given in (4.2.2), one can find easily

$$\begin{aligned} u_i P_3^T M_i - w_i P_1^T M_i &= 0 \\ v_i P_3^T M_i - w_i P_2^T M_i &= 0 \end{aligned} \tag{4.2.4}$$

and modifying the equation above

$$\begin{bmatrix} -w_i M_i^T & 0^T & u_i M_i^T \\ 0^T & -w_i M_i^T & v_i M_i^T \end{bmatrix} \begin{bmatrix} P_1 \\ P_2 \\ P_3 \end{bmatrix} = 0 \tag{4.2.5}$$

Hence, for a pair of 3D – 2D projection pairs, two homogeneous equations are found. Since the projection matrix has eight degrees of freedom, four pairs of projection pairs are sufficient to find a unique solution for P defined up to scale. Stacking all the equations obtained from projection pairs (possibly more than 4), a system of linear equations in the form of

$$Ap = 0 \tag{4.2.6}$$

can be obtained, where *A* is the measurement matrix and $p = \begin{bmatrix} P_1^T & P_2^T & P_3^T \end{bmatrix}^T$, projection matrix elements. The solution to this problem subject to $\|p\| = 1$ constraint (since scale does not matter) is, as indicated before, equal to the eigenvector associated with the smallest eigenvalue of the *A* matrix.

During the estimation of the projection matrix, as in the normalized 8-point algorithm, the normalization step is also applied to the data points in order to improve the conditioning of the problem. The normalization is applied on the 3D points as a translation in order to move the centroid to the origin and a scaling to make the variance of the distance of 3-D points to the origin $\sqrt{3}$. A similar normalization is also applied to the 2D points, whereas this time the variance is modified to be $\sqrt{2}$.

Algorithm 4.1.1: Normalized P-Matrix estimation from projection pairs

Given n>3 3D-2D correspondence pairs

1. Compute the mean and variance of the distances to the centroid for both 3D and 2D points.
2. Form matrices T_{2D} and T_{3D} such that the $T_{2D}m$ and $T_{3D}M$ are the normalized 2D and 3D coordinates, respectively.
3. Form the A matrix from the projection pairs according to Equation 4.2.5.
4. Find the SVD of the A matrix such that $A = USV^T$ and the solution vector is the column of the V matrix associated with the smallest diagonal entry of the S matrix (i.e., smallest singular value of A)
5. Compute the projection matrix P' for the denormalized data points as, $P' = T_{2D}^{-1} P T_{3D}$

4.2.1.2 Robust estimation of the projection matrix from projection pairs

Similar to the case during the estimation of the fundamental matrix, some robustness is required in order to ensure a correct computation of the projection matrix, in case of contaminated data. For this purpose, RANSAC-based computation of the projection matrix is adopted (for details of the RANSAC algorithm, refer to Section 3.2.3).

The error measure in order to decide whether a point is an inlier or not is decided by using the reprojection error, which is formally defined as:

$$\text{Reprojection Error} = d(m, PM)^2 \qquad (4.2.7)$$

where *d(m,PM)* returns the distance between the 2-D image point and the projection of 3-D scene point.

Algorithm 4.2.1: Robust P-Matrix Estimation

Given n>3 3D-2D correspondence pairs

Repeat N times

1. Select 4 pairs of 3D-2D correspondences randomly and estimate a projection matrix following the Algorithm 4.1.1
2. Find the number of pairs consistent with the estimated model using the reprojection error (Equation 4.2.7)
3. Choose P with the largest number of inliers, and reject those pairs which yield e > t for this particular P.
4. Recalculate the number of iterations N using the formula given in Equation 3.2.7.

4.2.1.3 Refinement of the projection matrix

After the robust estimation of the projection matrix, a nonlinear stage also exists in order to refine the projection matrix. Levenberg-Marquardt algorithm is used to minimize the reprojection error given in (4.2.7) with respect to the parameters of the projection matrix. However, direct minimization of the P matrix parameters will yield erroneous results, since the elements of P matrix are not independent from each other. Therefore, the minimization should be carried out on the individual rotation and translation parameters. In order to achieve this form, the rotation matrix should be represented in quaternion form (see Appendix E).

4.3 Initialization of new structure points

For the points, which have not been associated to a 3-D point, some new 3-D structure points should be estimated by using the calculated projection matrices for the current and the previous frames through triangulation (Section 3.4). This approach will ensure the estimation of the pose of the views, which do not have common features with the initial framework. Moreover, it is possible to initiate higher number of 3-D points for the scene for obtaining more information. It is observed during the simulations, choosing points that are present in at least more than 3 views ensures the elimination of spurious matches and improves the overall structure in the final reconstruction.

4.4 Refining structure and motion

Once the structure and motion has been computed for all of the frames in the sequence, a final global refinement is applied. For this purpose, *bundle adjustment* [40] method is used. Bundle adjustment is the problem of refining a visual reconstruction to produce *jointly optimal* 3D structure and viewing parameter (camera pose and/or calibration) estimates. This procedure is o*ptimal* in the sense that the parameter estimates are obtained by minimizing a model fitting error function. The estimation is also *joint* so that the solution is both optimal with respect the structure and camera variations at the same time. "Bundle" refers to the light rays joining the 3D points and the camera centers which are attuned optimally according to both feature and camera positions. In this method, all of the structure and camera parameters are adjusted together in one bundle.

The cost function for minimization can be selected as follows:

$$\min_{P_i, M_j} \sum_{i=1}^{m} \sum_{j=1}^{n} d(m_{ij}, P_i(M_j))^2 \qquad (4.4.1)$$

This cost function jointly minimizes the errors due to noise during model estimation and locations of the 3D points. Therefore, the minimization problem has a vast parameter space. The direct minimization of this cost will need quite a long time to converge. However, a sparse version of the bundle adjustment should improve the execution time considerably. Therefore, a sparse variant of the bundle adjustment is preferred [41]. More

information about *sparse bundle adjustment* is given in Appendix D.

The minimization over the projection matrix parameters is not performed directly, whereas the rotation and translation parameters are again utilized separately.

4.5 Multiple view reconstruction algorithm

Multiple view reconstruction algorithm is summarized in the below diagram. Briefly, the algorithm first estimates an initial reconstruction and then inserts each frame with respect to this framework. Finally, the overall reconstruction is refined employing a global bundle adjustment. In Figure 4.2, the structure of the multiple-view reconstruction algorithm is given.

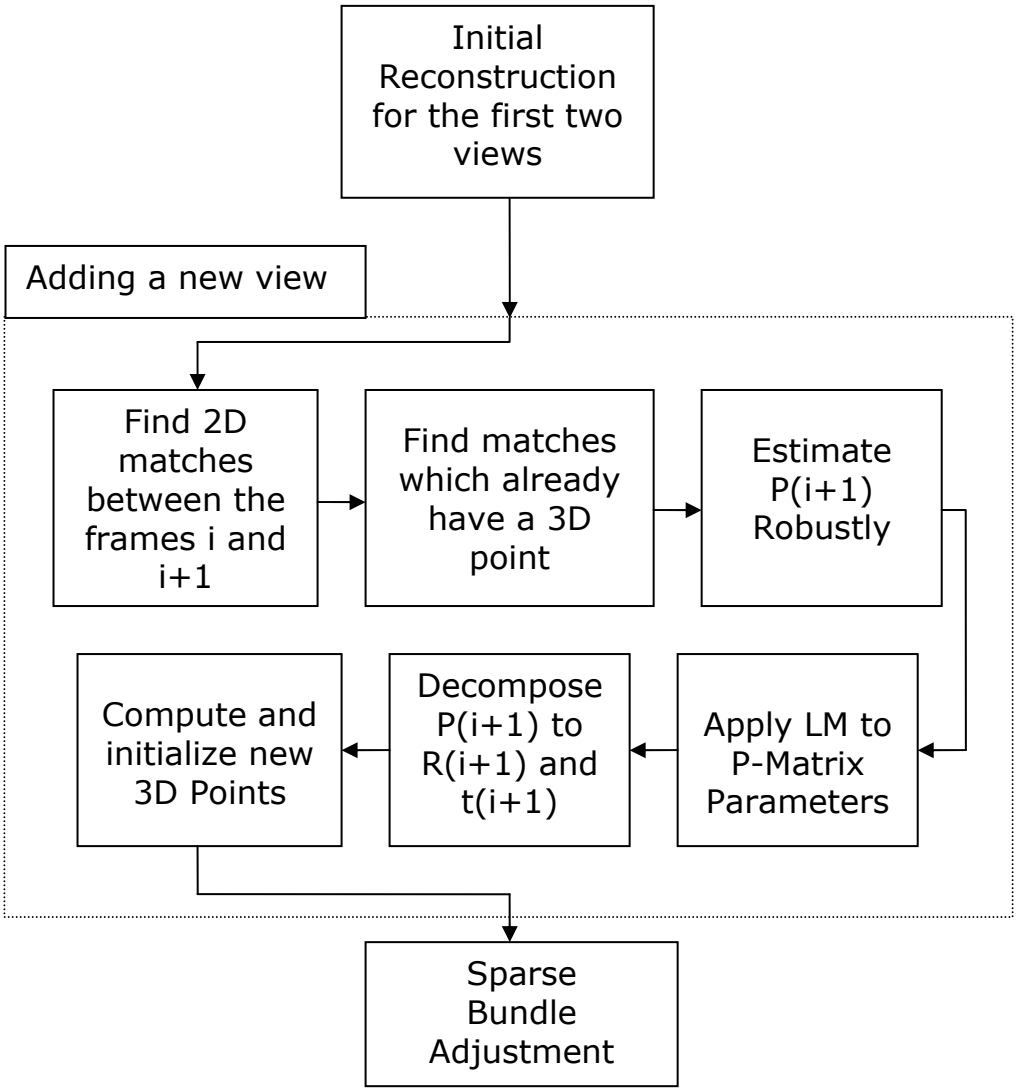


Figure 4.2: Multiple view reconstruction algorithm

4.6 Simulation results

In the figures below, some of the multiple view reconstruction results are presented. The results are given in different viewing angles (see Figure 3.20).

Figure 4.3: Leuven Castle Sequence

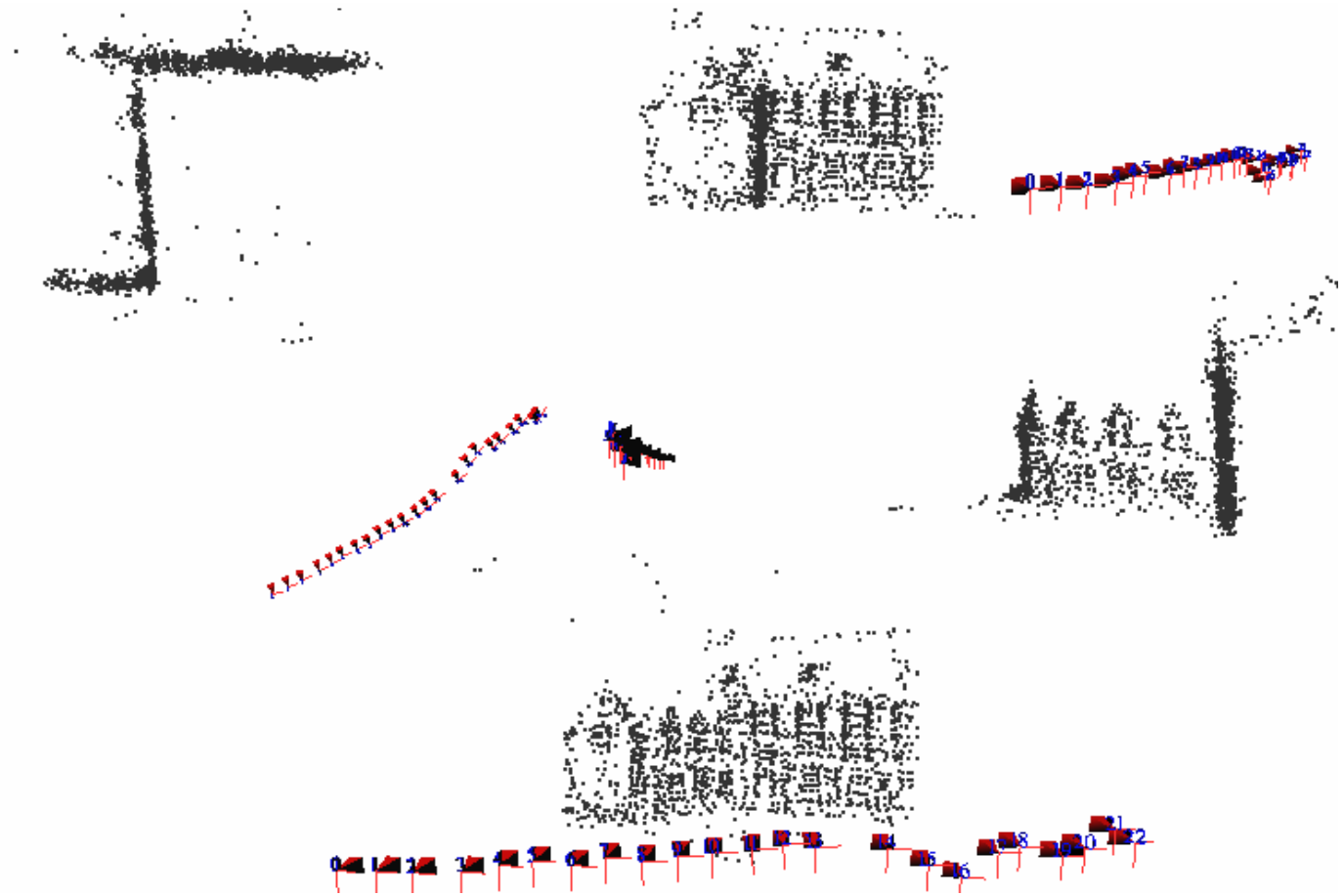


Figure 4.4: Leuven Castle Sequence results, illustrated from different viewing angles. Each triangle prism represent a camera location from which a picture is taken.

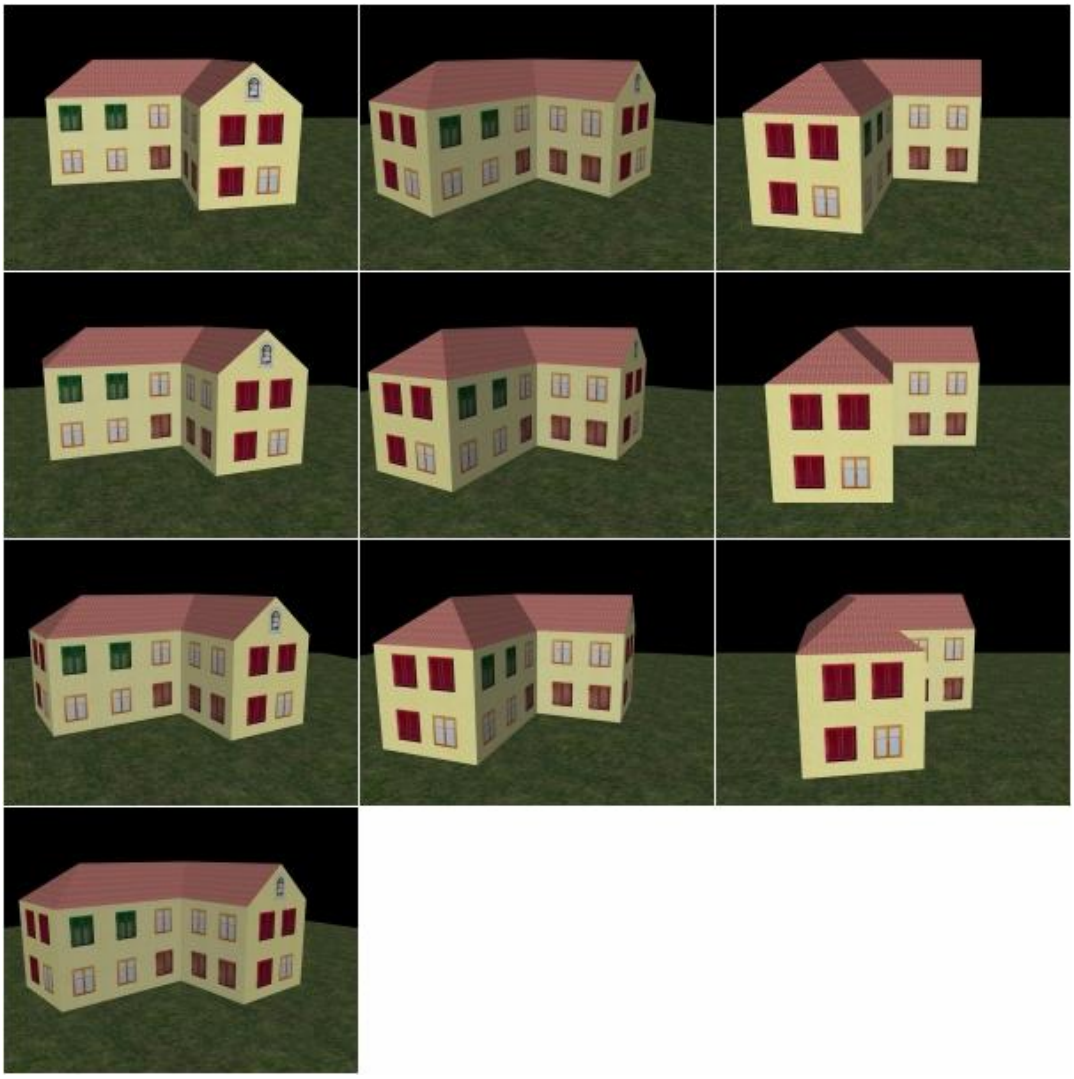


Figure 4.5: Model House Sequence

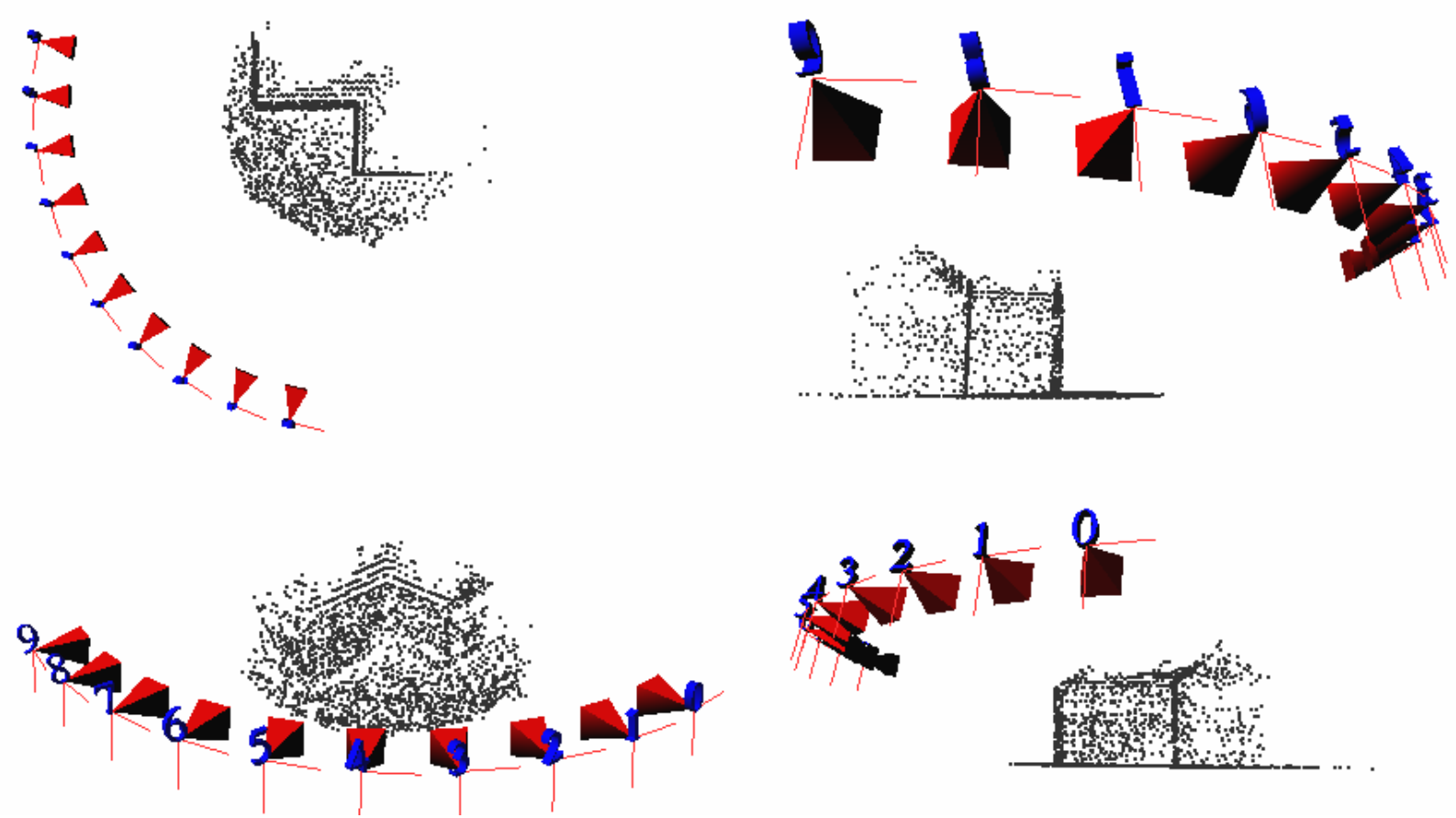


Figure 4.6: Model house sequence results

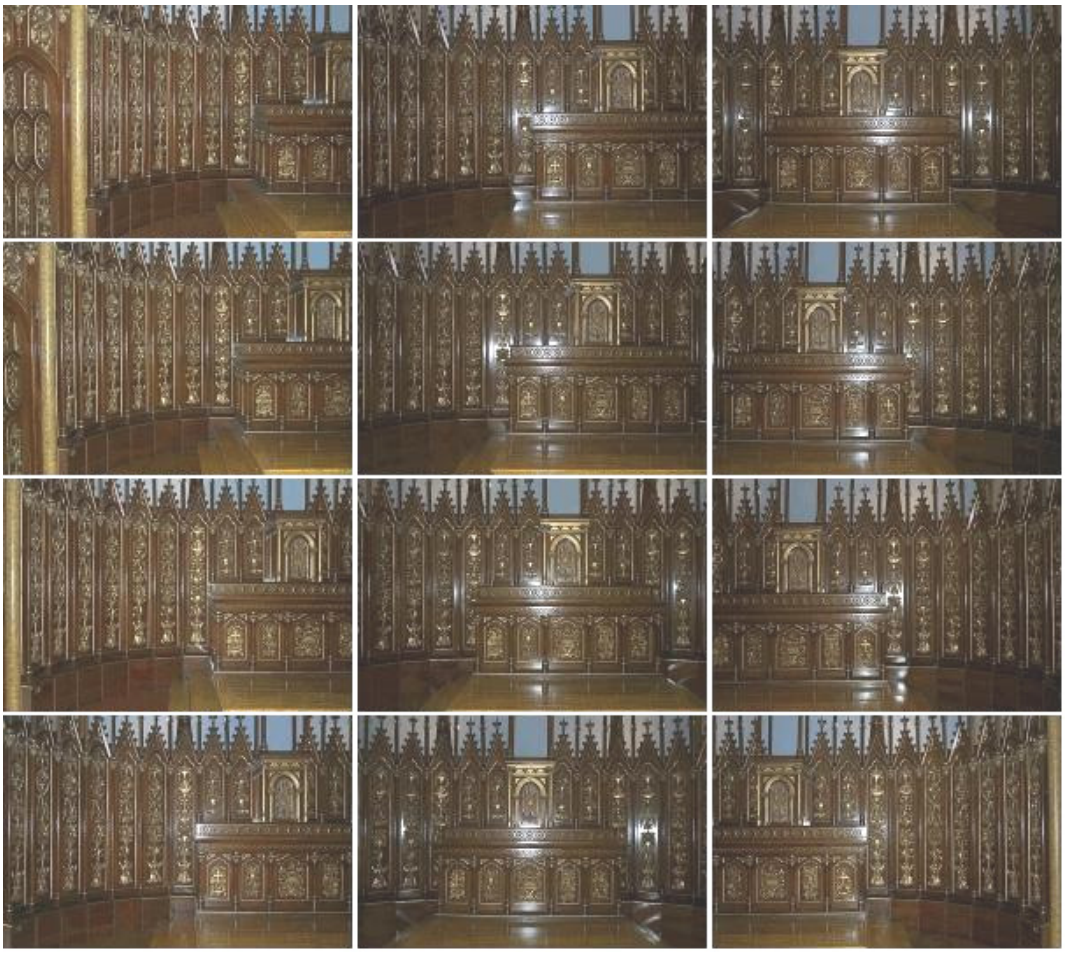


Figure 4.7: Chapel Sequence Images

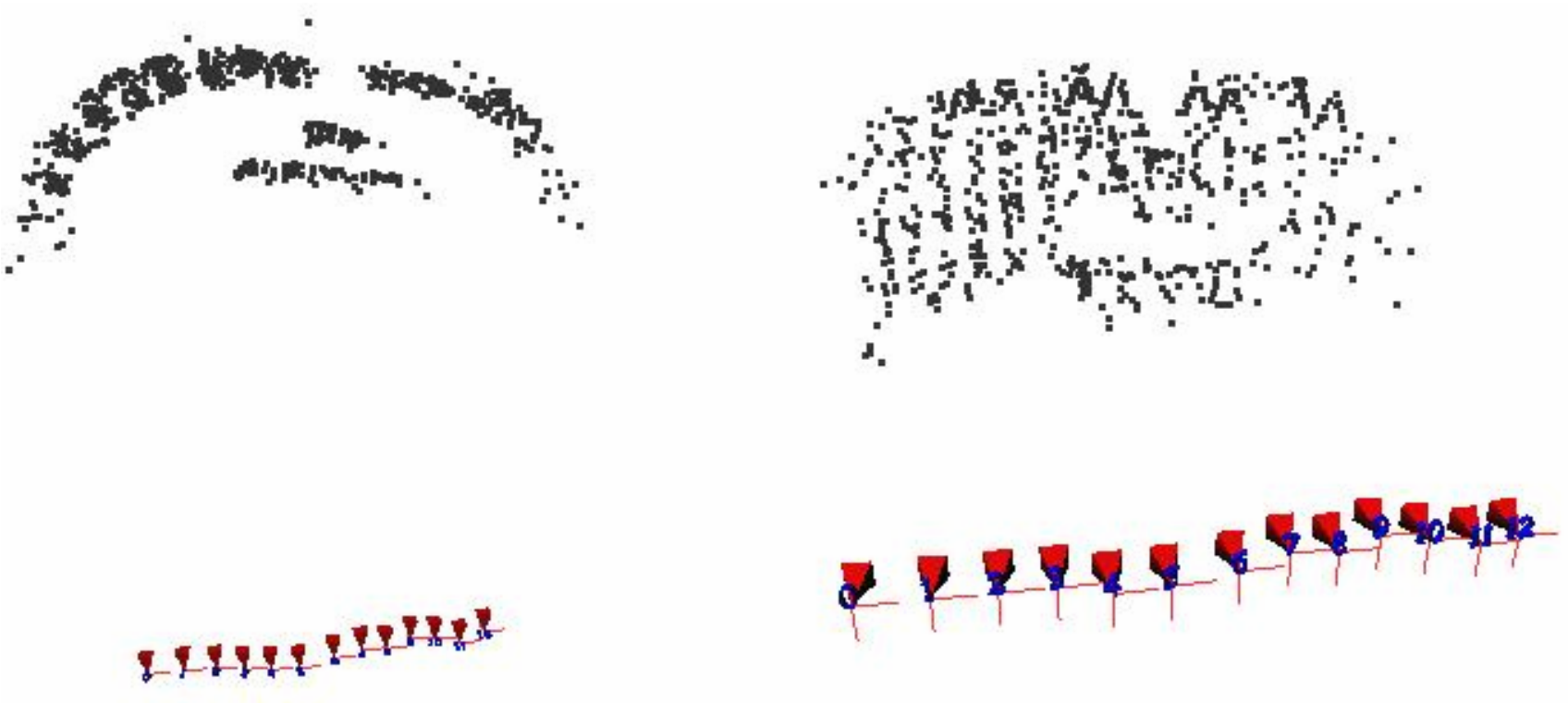


Figure 4.8: Chapel Sequence Results

CHAPTER 5

3D RECONSTRUCTION FROM MULTIPLE VIEWS CONTAINING INDEPENDENTLY MOVING OBJECTS

5.1 Introduction

In the previous chapters of this thesis, an algorithm is presented in order to estimate the fundamental matrix between two views robustly, while rejecting the correspondence outliers (Algorithm 3.3.4). The implemented algorithm is suitable for static environments. In this chapter, the performance of this algorithm in sequences which contain independently moving objects (IMO) is investigated. Moreover, a novel algorithm in order to improve the computation time of the outlier rejection is also proposed. For the sake of completeness, some background information is given in the following sections about parallax-based rigidity constraint, which is the backbone of the proposed algorithm.

5.2 Plane+Parallax Decomposition

3D parallax is the variations in the 2D motion vectors of the projected scene points due to changes in the depth of the scene structures, when the camera makes a significant translational motion [60]. There are single- and multi-layered approaches to

handle different situations where the parallax is not very significant [60]. However, if the parallax effect starts to increase, in case of more complex 3D scenes, then *plane+parallax decomposition* approach should be utilized, as suggested in [60].

In *plane+parallax decomposition*, motion vectors of the scene are decomposed into two components: plane and parallax. The 2D parametric registration process is performed by a single global 2D parametric transformation between a pair of images:

$$\begin{bmatrix} u(x,y) \\ v(x,y) \end{bmatrix} = \begin{bmatrix} p_1 x + p_2 y + p_5 + p_7 x^2 + p_8 xy \\ p_3 x + p_4 y + p_6 + + p_7 xy + p_8 y^2 \end{bmatrix} \qquad (5.2.1)$$

where $u(x,y)$ and $v(x,y)$ are the motion vectors at point (x,y). By estimating the parameters p_i in (5.2.1), the plane registration transformation is computed.

The plane registration step removes all the effects of camera rotation, zoom and calibration without explicitly calculating them [60, 61]. This result can also be understood from the fact that the planar motion caused by rotation or zoom does not depend on plane depth. In other words, all the planes at different depth layers will be registered also, once a plane is registered in terms of rotation and zoom of the camera. Therefore, the residual image motion after the plane registration should be due only to the translational component of the motion of the camera and to the deviation of the scene structure from a planar surface. Thus, the residual motion field is an *epipolar flow field.* An epipolar flow field is a field of vectors that are structured subject to an epipole [60]

(see Figure 5.2). These observations led to the so called *plane+parallax decomposition* of the scene.

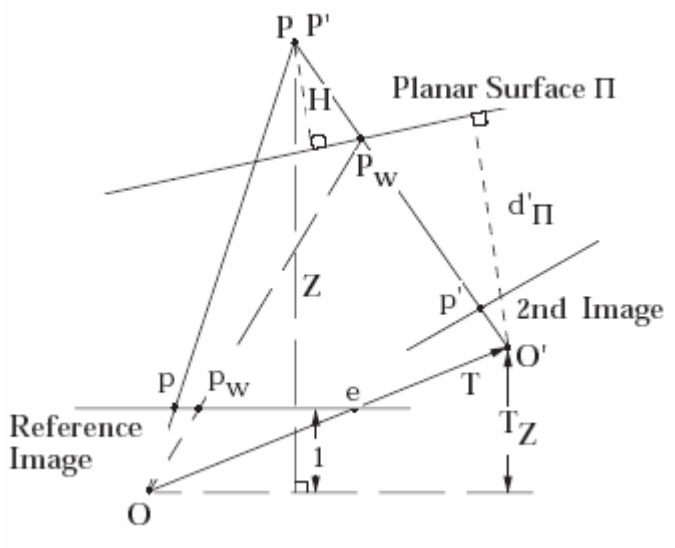


Figure 5.1: Geometric interpretation of the plane+parallax decomposition [60]

In Figure 5.1, the geometric interpretation of the plane+parallax decomposition is illustrated. In this figure, $P=(X,Y,Z)^T$ and $P'=(X',Y',Z')^T$ are the Cartesian coordinates of a scene point with respect to two different camera views and $p=(x,y)$ and $p'=(x',y')$ denote the projections of these points onto the camera planes, respectively. In Figure 5.1, Π denotes a real (or a virtual) planar surface in the scene, which is registered by a parametric registration approach. The 2D image displacement of the point P is then calculated as

$$u = p'-p = u_\pi + \mu \qquad (5.2.2)$$

where u_π is the planar part of the image motion and μ is the residual planar parallax in 2-D motion. The homography due to Π can be modeled as a 2-D parametric transformation, which is in

general a projective transformation, and an approximation to this transformation can be approximated by

$$\bar{u}_{\Pi} = \bar{p}' - \bar{p}_w \quad , \quad \bar{\mu} = \begin{cases} \gamma \dfrac{T_z}{d'_{\pi}} (\bar{e} - \bar{p}_w) & 'if \;\; T_z \neq 0' \\ \dfrac{\gamma}{d'_{\pi}} \bar{t} & 'if \;\; T_z = 0' \end{cases} \qquad (5.2.3)$$

where $\bar{p}_w$ is an image point in the first frame which results from warping the corresponding point $\bar{p}'$ in the second image by the 2D quadratic transformation of the plane Π. $\bar{e}$ denotes the epipole and d'_{π} denotes the distance of the second camera center from the plane. γ is called as the *projective 3D structure* of point *P* [60] and it is a measure of 3-D shape of point P. It is equal to the ratio of the perpendicular distance of point P to the planar surface Π to the depth of the point P with respect to the first camera (γ = H/Z see Figure 5.1). The final term $\bar{t} = (T_x, T_y, T_z)^T$ is the translation. For the derivation of this equation, the readers should refer to [60].

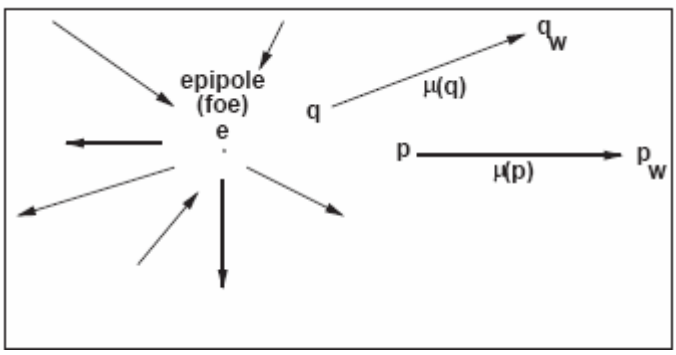


Figure 5.2: Epipolar field of the residual parallax displacements [60]

The parallax equation, given in (5.2.3), suggests the existence of an epipole, where all residual motion vectors expand from or diverge to. Therefore, if the epipole is recovered, all that remains

for detecting the moving objects is to identify the vectors, which do not obey this common rule. The estimation of the position of the epipole, therefore, strictly affects the performance of the independent moving object detection problem. However, it will be observed in the next section that without calculating the epipole explicitly, it is still possible to find a metric to detect IMO's.

5.3 Parallax-based rigidity constraint

It is explained in Section 5.2 the methodology to compute the 3-D projective structure of a point. The *relative 3D projective structure* of two points having γ_1 and γ_2 is defined as:

$$\frac{\gamma_1}{\gamma_2} = \frac{\bar{\mu}_2^T (\Delta \bar{p}_w)_\perp}{\bar{\mu}_1^T (\Delta \bar{p}_w)_\perp} \tag{5.3.1}$$

where, as shown in Figure 5.3, p_1 and p_2 are the image locations of two points and $\Delta \bar{p}_w = \bar{p}_{w2} - \bar{p}_{w1}$ is the vector connecting the warped coordinates ($v_\perp$ denotes a vector perpendicular to v).

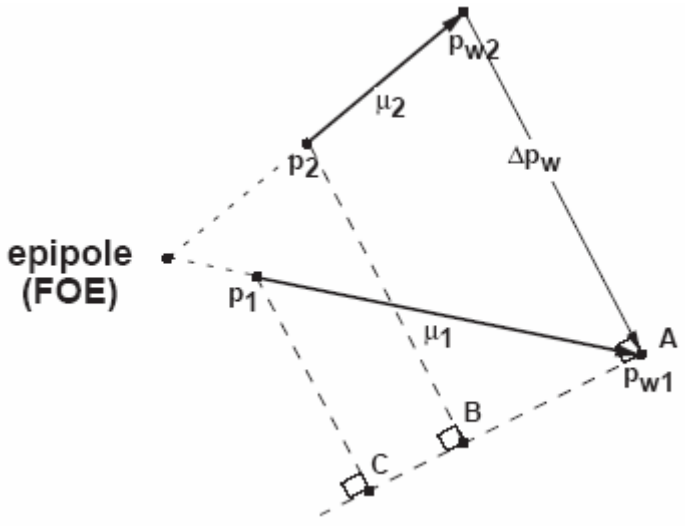


Figure 5.3: Pair-wise parallax-based shape constraint

This constraint (Equation 5.3.1) directly relates the relative projective structure of two points without an explicit epipole relation.

In [60], it is stated that, relative 3D projective structure of a pair of points does not change with respect to the camera motion. Therefore, by observing the value of this constraint, it is possible to detect independently moving objects. This constraint is defined formally as:

$$\frac{\bar{\mu}_2^{jT}(\Delta\bar{p}_w)_\perp^j}{\bar{\mu}_1^{jT}(\Delta\bar{p}_w)_\perp^j} - \frac{\bar{\mu}_2^{kT}(\Delta\bar{p}_w)_\perp^k}{\bar{\mu}_1^{kT}(\Delta\bar{p}_w)_\perp^k} = 0 \qquad (5.3.2)$$

where $\bar{\mu}_1^j, \bar{\mu}_2^j$ are the parallax displacement vectors of the two points between the reference frame and jth frame, $\bar{\mu}_1^k, \bar{\mu}_2^k$ are the parallax vectors between the reference frame and kth frame, and $(\Delta\bar{p}_w)^j, (\Delta\bar{p}_w)^k$ are the corresponding distances between the warped points.

By using this constraint, it is possible to discriminate between the background and IMOs in three frames, given a motion vector that must be selected from the background.

5.4 Algorithm to eliminate matches due to IMO's

As it is observed in the previous section, by the help of parallax-based rigidity constraint, it might be possible to detect independently moving objects in three consecutive frames.

However, in order to accomplish this, the constraint strictly requires a motion vector pair (one between the first two frames and another between the second and third frame), which must belong to a background point. In order to achieve this aim, a novel algorithm is proposed within the next sections.

5.4.1 Plane Registration

The plane registration process involves the estimation of eight parameters from the motion vectors of two images (Equation 5.2.1). However, all of the motion vectors cannot be used for this purpose, since there may be outliers as well as many non-planar surface vectors. The dominant plane estimation, therefore, has to be completed by using a robust procedure. Similar to the procedure for the estimation of the projection matrix in Section 4.2 (or the estimation of the fundamental matrix in Section 3.3, in the plane registration step), RANSAC is employed for the robust estimation of the "dominant plane". Once the parameters for the dominant plane are estimated, the residual parallax components of the motion vectors are calculated as the next step.

5.4.2 Background seed selection algorithm

Background seed selection is a critical step in removing IMO contributions from the correspondence set. Parallax-based rigidity constraint should be utilized for this purpose; it constrains 3-D structure of all stationary background points. The parallax-based rigidity constraint, although, forces the change in the relative 3-D structure to remain zero, this does not always hold due to noise. Therefore, only choosing a random vector and counting the

number of vectors that obey the constraint will not solve the problem of the background vector selection. Moreover, the errors in the parallax-based rigidity constraint differ, when one changes the support (background) vector of the constraint (μ_1 in Equation 5.3.2). Therefore, simple thresholding will not be the solution to this problem, since the threshold should also be changed adaptively for different scenes.

The proposed novel solution to this problem can be explained as follows: N different support vectors are chosen and the number of vectors that are outside a certain neighborhood around one of the support vectors (i.e. candidate background seed point), which obey the rigidity constraint within a small threshold, are counted. After testing all support vectors in this manner, the candidate seed point, yielding the maximum number of supports, is chosen as the *background seed*.

The support vectors are also selected according to the magnitude of the residuals. The magnitude range of the residual vectors is divided into N equal intervals and a support vector is selected from every interval (see Figure 5.4). This selection method is adopted due to the fact that the plane registration step usually leaves behind vectors with small residual from the dominant plane. Therefore, the vectors on this dominant plane must not be selected, since their small norm is due to noise. On the other hand, the vectors with large residuals are not reliable, since they might be outliers. Hence, in order to cover the whole range of vectors such a procedure is proposed.

Another important aspect of the proposed selection criteria is elimination of the vectors within the neighborhood of the support

vector, while calculating the number of vectors that obey the rigidity constraint. In this manner, it is possible to eliminate possible points belonging to an IMO, which mostly has its support vectors within its neighborhood. If this constraint is not used, one might find the change in the rigidity constraint still to be a small number to erroneously declare an IMO point to become a background seed, while, unfortunately, most of the support pixels are belonging to the IMO itself. On the other hand, this constraint reduces the number of the consistent vectors to an IMO-belonging support vector. This situation is not a problem for the background vectors, since they are not confined (i.e. localized) to a single region.

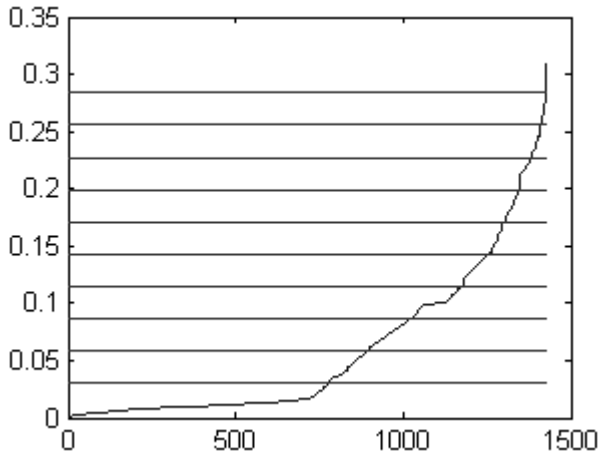


Figure 5.4: Residual motion vectors sorted according to their norms: y axis is the norm and x axis is the pixel number

5.4.3 Application of the parallax-based rigidity constraint by the background seed

At this stage, all the correspondence vectors are tested by using parallax-based rigidity constraint with the previously selected

background seed pixel. In order to increase the robustness of the algorithm, more than one background pixel can be used to discriminate between background and IMO vectors. A vector is decided to belong to a background point, if, out of M different supports, it is within the first p-percent of the sorted cost calculated according to (5.2.2) at least K times. (K < M and K is larger than some threshold). Hence, the following algorithm is obtained for rejecting IMO contributions, as well as any kind of outliers, in the correspondence set. A summary of the algorithm is given below:

Algorithm 5.4.1: Using parallax based rigidity constraint to reject IMO's

1. Apply plane registration to the motion vectors between the first two frames as well as the second and third frames by using RANSAC
2. Find the background seed
    a. Sort the residual motion vectors according to their norms
    b. Choose N support vectors with equal distance from each other in terms of their norm values
    c. Calculate the number of vectors that obey the parallax based rigidity constraint with threshold t_1 for each of the support vectors. Do not consider the vectors within d_1 distance to the support vector.
    d. Choose the vector with the maximum number as the background seed
3. Select M vectors yielding the smallest error with the background seed and calculate the parallax based rigidity constraint errors for each of these support vectors
4. Sort the elements of these sets according to their errors and select the vectors that are within the first p-percent of the sets.
5. Choose the vectors that are selected more than K times (K<M) as background pixels and discard the rest.

5.5 Simulation results

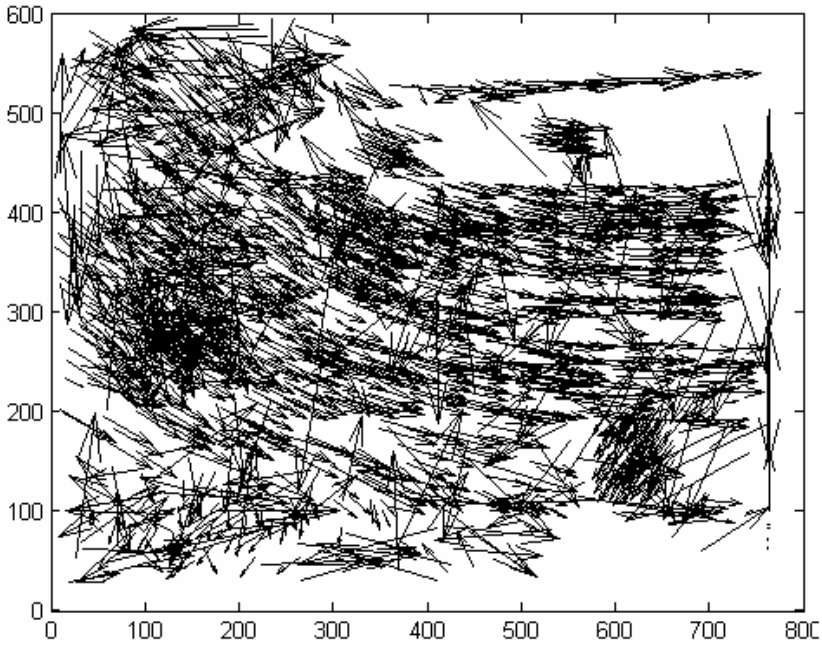
In this section, the results of the Algorithm 5.4.1 and the comparison tests of this algorithm with the outlier rejection technique explained in Section 3.3 (Algorithm 3.3.4), is presented. In the figures and table below, Algorithm 3.3.4 is denoted as *RANSAC* and Algorithm 5.4.1 is mentioned as *IMOR*. Another comparison is achieved by using both of the algorithms consecutively. This method is also abbreviated as *IMOR+RANSAC.*

In the implementation of *IMOR*, the following parameters are chosen N = 20, t_1 = 1e-5, d_1 = 60, p=0.7, M=10 and K = 6. During simulations, the following image sets are utilized: Figure 5.5 and 5.7 contain an artificial IMO, inserted into the scene, whereas Figure 5.9 includes a natural case. In these figures, the results are presented in the following manner: (a), (b) and (c) sub-figures are the input image triplets, where (d) depicts the resulting correspondence vectors found by the matching algorithm given in Algorithm 3.3.2 for the first two images. Subfigure (e) shows the resulting displacement vectors selected by *IMOR* as background and (f) shows the results of the *RANSAC* algorithm. Finally, the rejected vectors by IMOR are shown in (g).

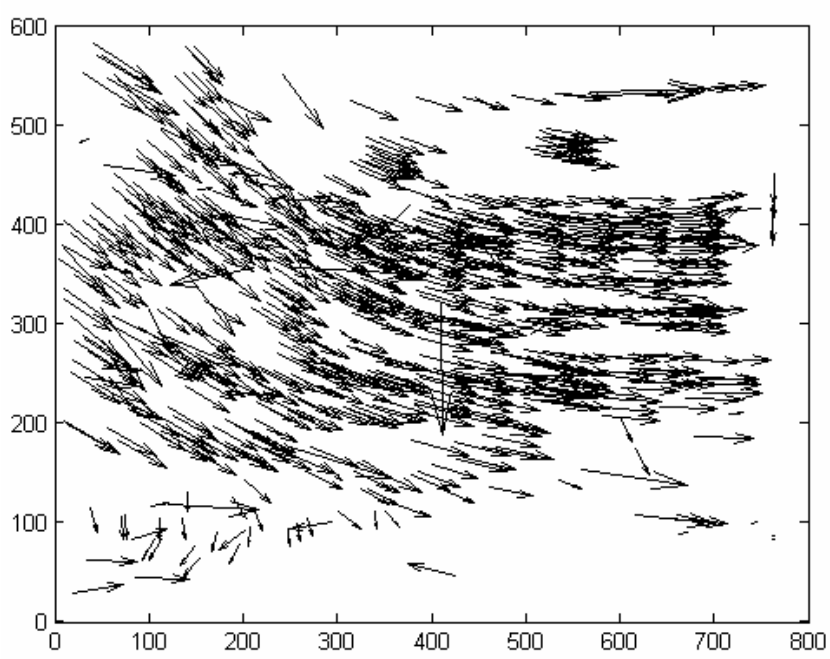
(a) image 1

(b) image 2

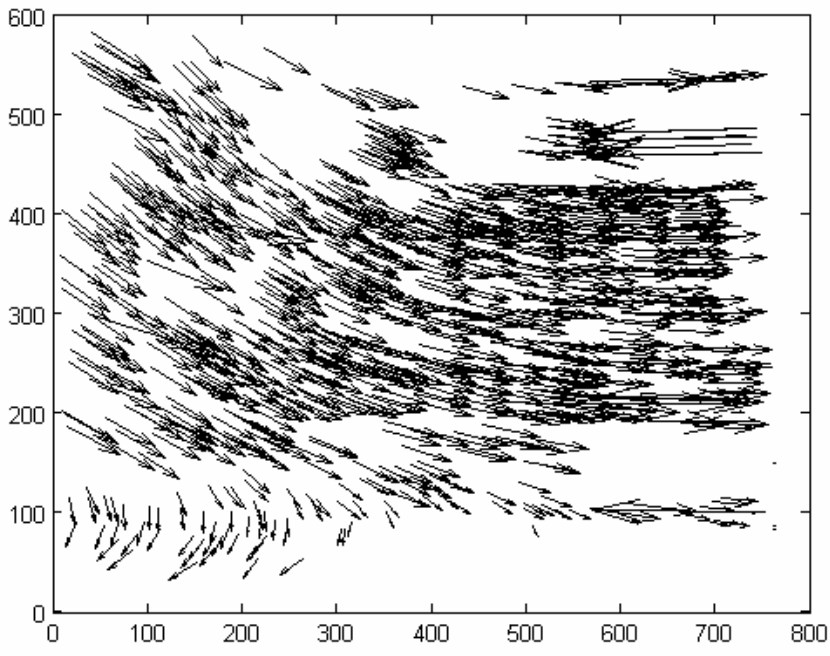
(c) image 3



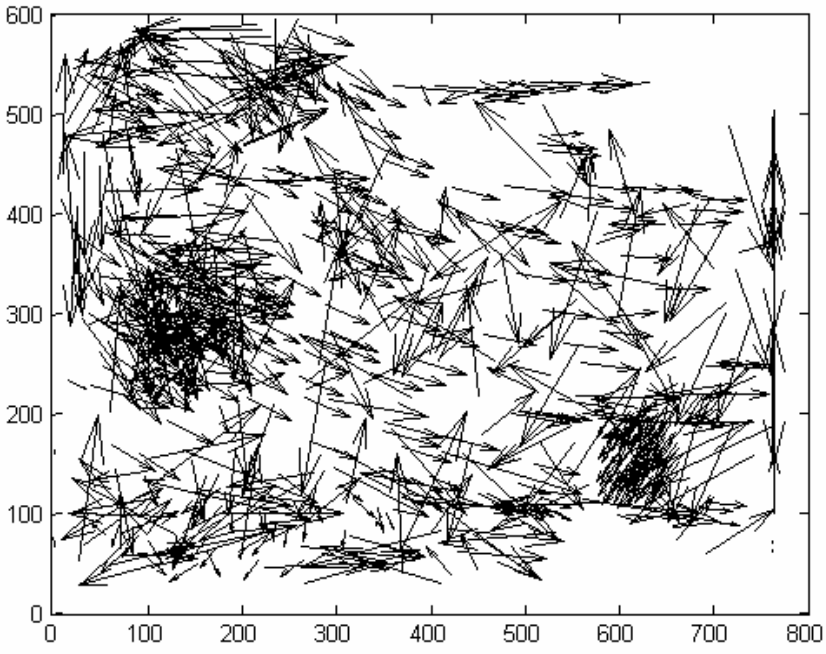
(d) input displacement vectors



(e) background vectors selected by ***IMOR***



(f) background vectors selected by ***RANSAC***



(g) rejected vectors by ***IMOR***

Figure 5.5 IMO Rejection Example 1

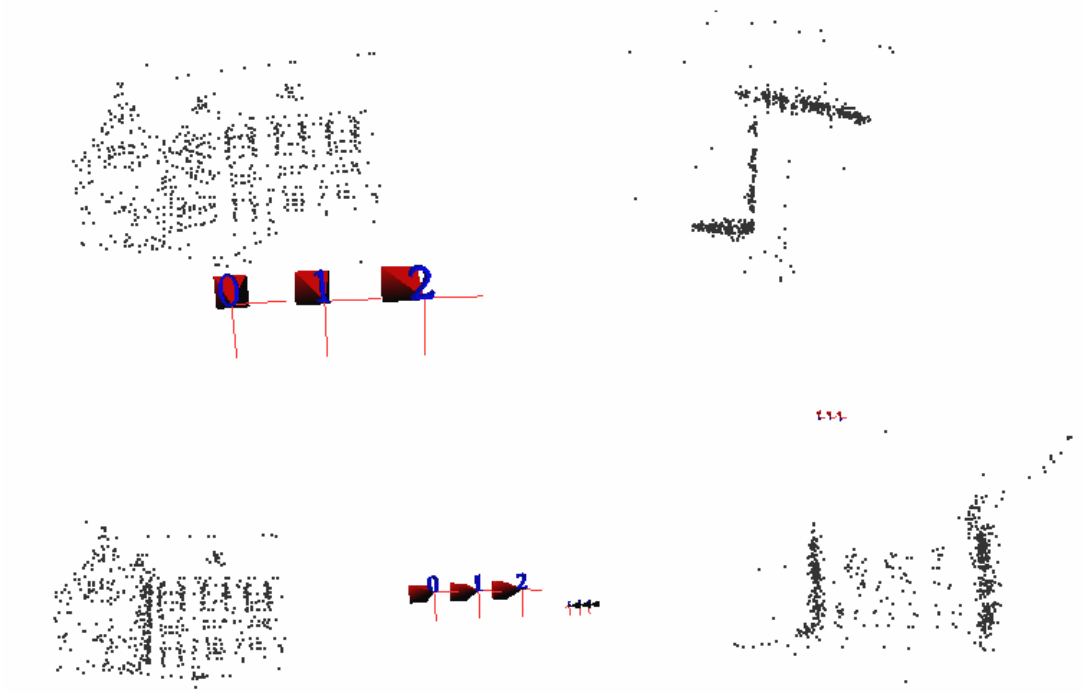
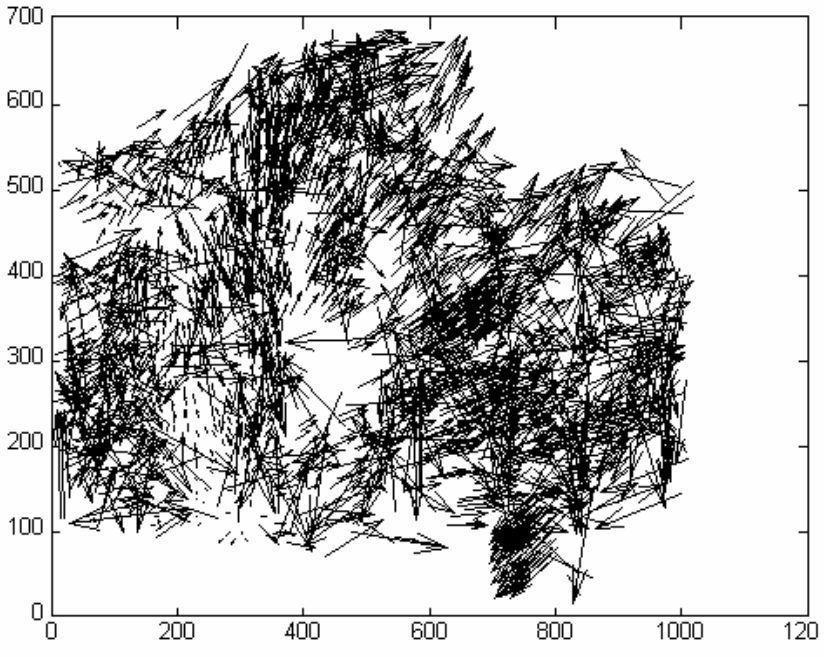


Figure 5.6: Reconstruction from images Figure 5.5 (a), (b), and (c) using IMOR+RANSAC

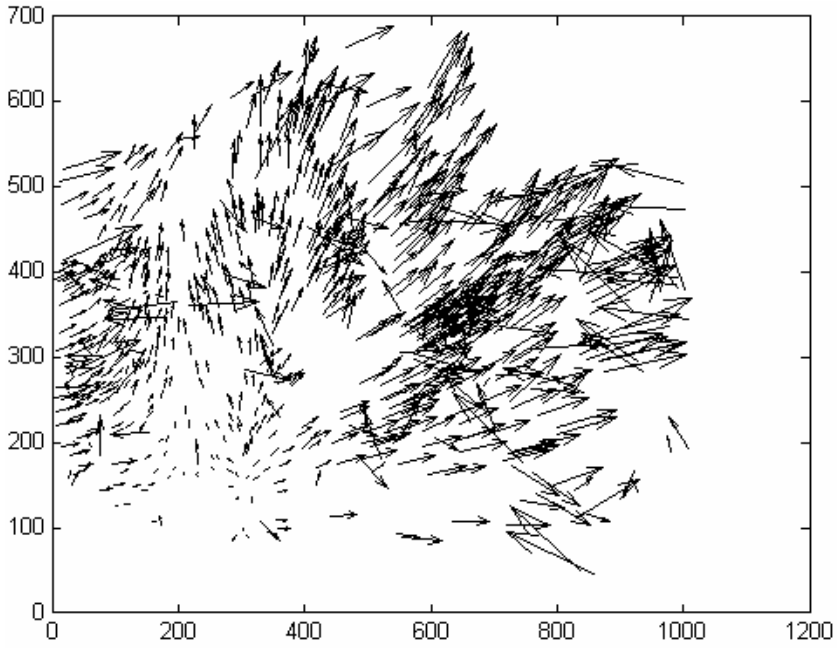
(a) image 1

(b) image 2

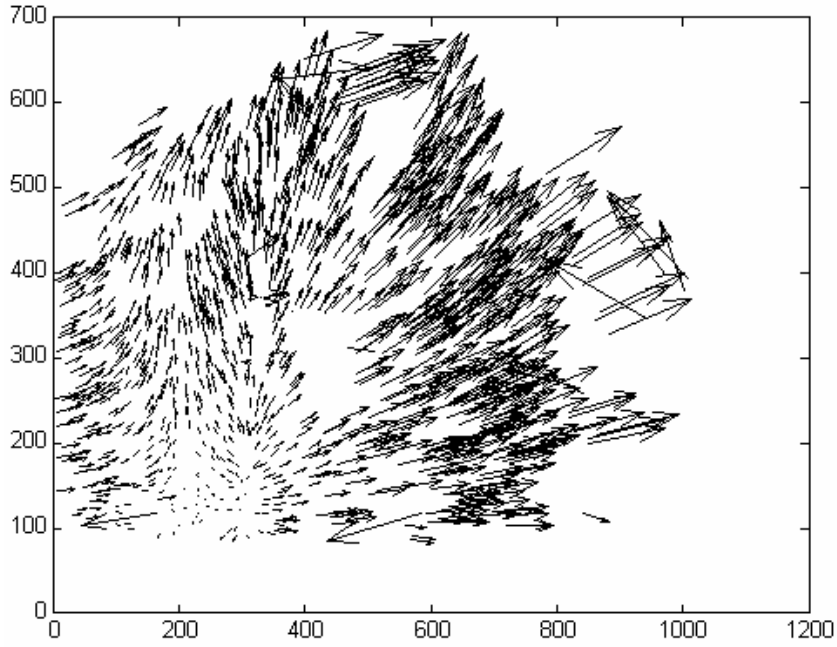
(c) image 3



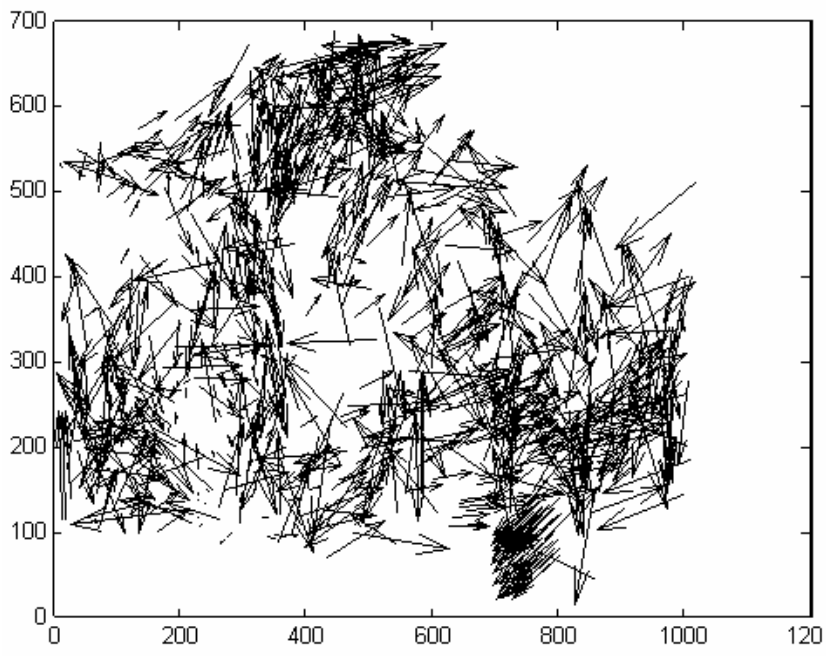
(d) input displacement vectors



(e) background vectors selected by ***IMOR***



(f) background vectors selected by ***RANSAC***



(g) rejected vectors by ***IMOR***

Figure 5.7: IMO Rejection Example 2

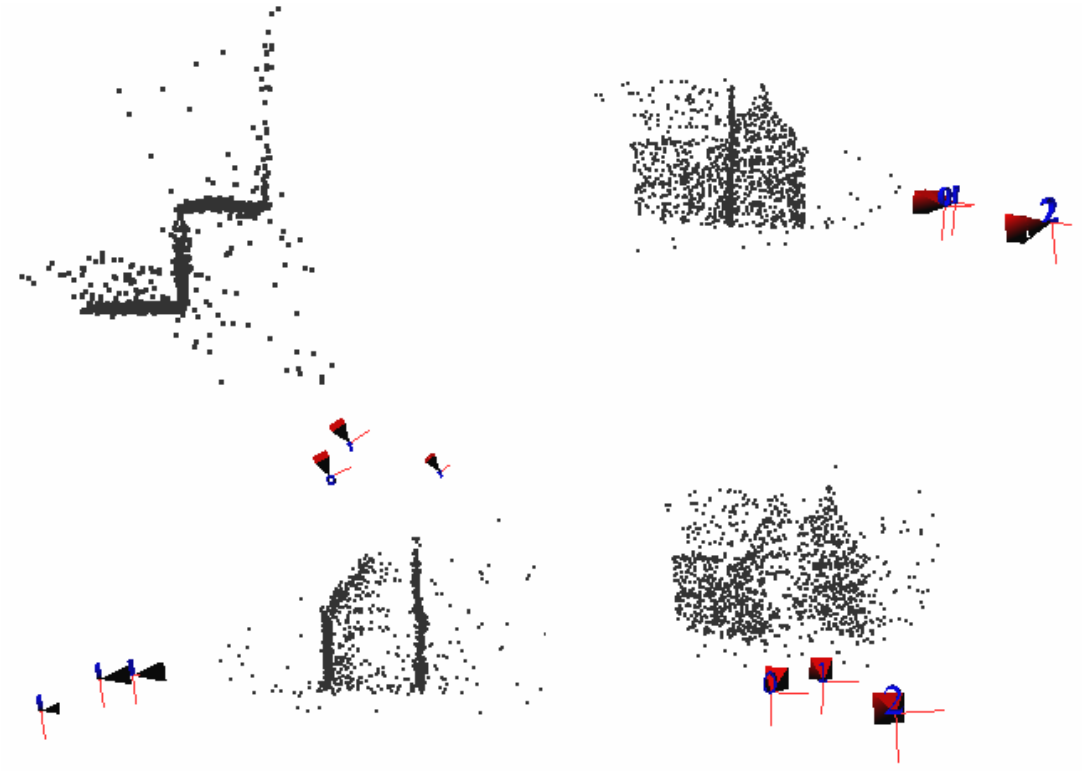
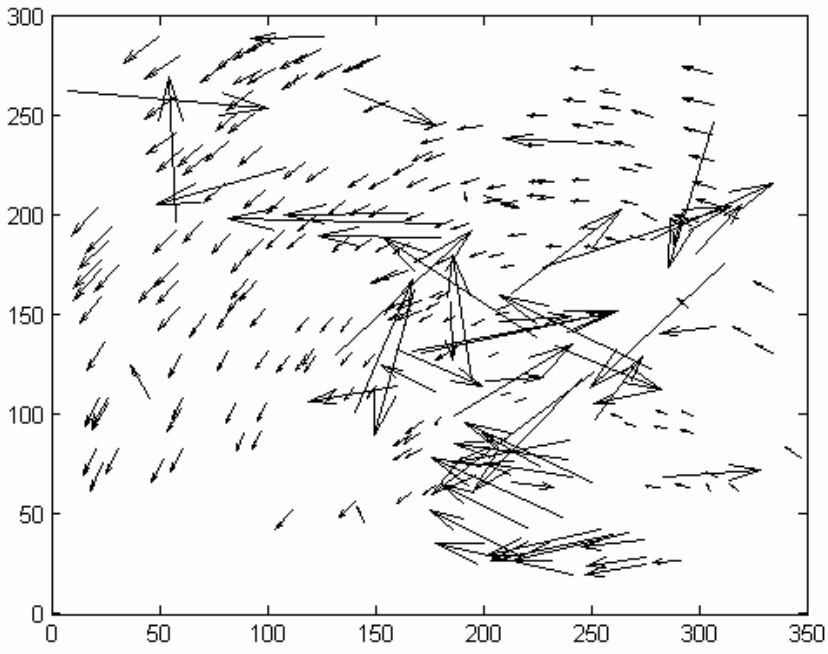


Figure 5.8: Reconstruction from images Figure 5.7 (a), (b) and (c) using IMOR+RANSAC

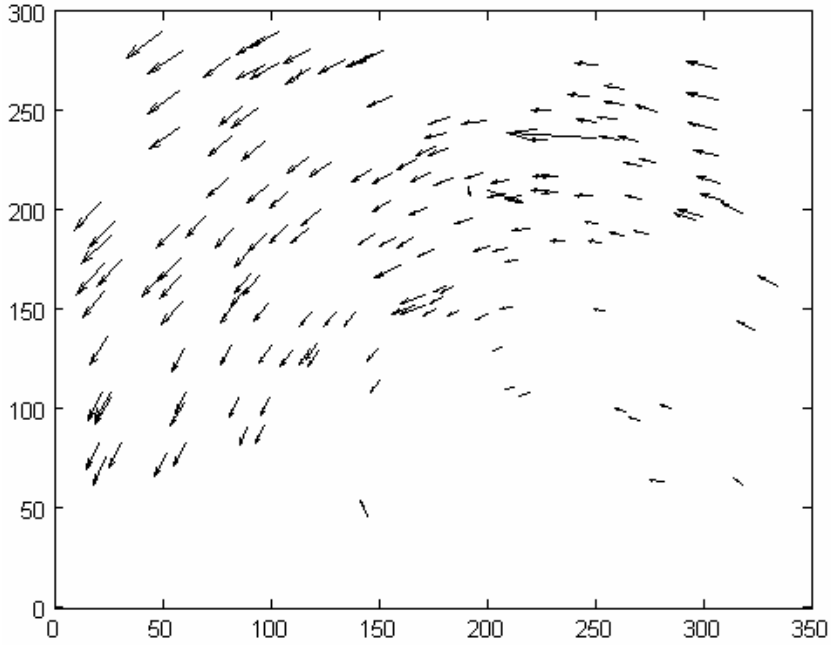
(a) image 1

(b) image 2

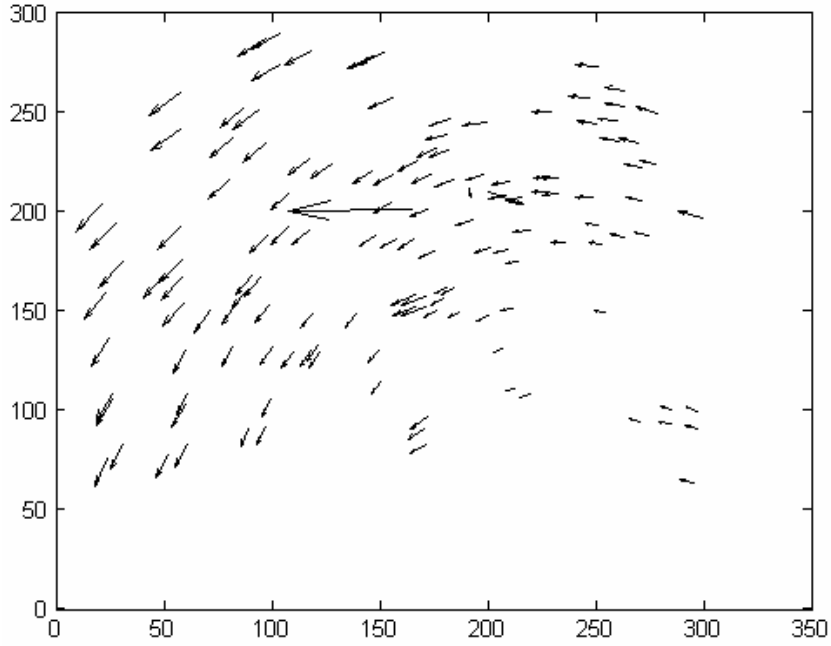
(c) image 3



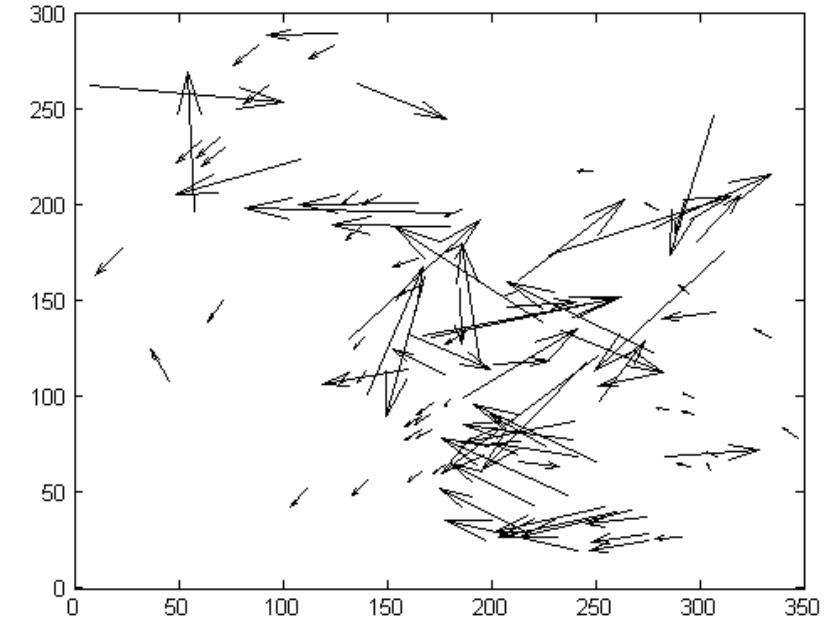
(d) input vectors



(e) background vectors selected by ***IMOR***



(f) background vectors selected by ***RANSAC***



(g) rejected vectors by ***IMOR***

Figure 5.9: IMO Rejection Example 2

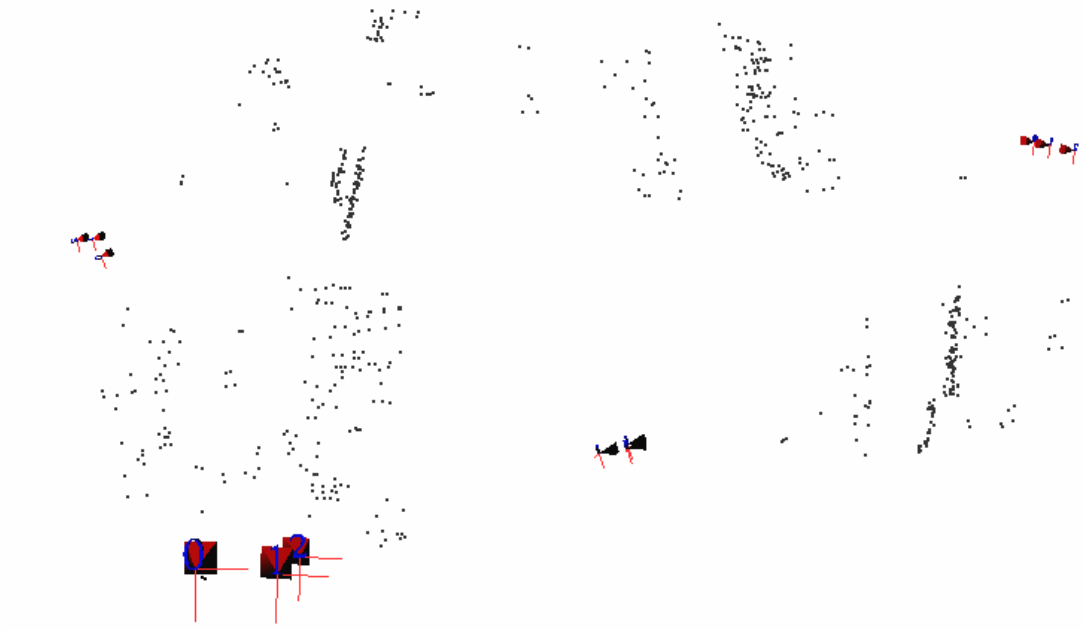


Figure 5.10: Reconstruction from images given in Figure 5.9 a, b and c using IMOR+RANSAC algorithm.

As it is observed from these results, *IMOR* algorithm gives comparable results with the *RANSAC,* although it cannot always

eliminate all of the outliers. However, *IMOR* is advantageous compared to *RANSAC* due to its much shorter execution time. It should be noted that *RANSAC* is an iterative algorithm with the number of iterations is not fixed, whereas *IMOR* is a single step approach. Hence, it is possible to use *IMOR* before *RANSAC* to eliminate most of the outliers and then use *RANSAC* to refine the results. In this manner, with a small number of iterations, a comparable reconstruction quality may be achieved in less time.

Table 5.1. Comparison Table: RANSAC vs IMOR+RANSAC

	Iteration Number	Duration (msec)	Wrong Rejections	Outliers Not Detected	Inlier Number	Total Vector Number
RANSAC	1626	4968	11	3	971	1651
IMOR	-	31	156	33	856	1651
IMOR+ RANSAC	21	112	158	1	824	1651

In Table 5.1, the results of aforementioned three algorithms are presented. The tests are performed over different data sets (7 different image triplets) and the results are calculated by simple averaging. "Wrong Rejections" column in the table refers to the number of true inliers that are labeled as outliers by the algorithms, whereas "Inlier Number" column refers to the number of correspondences, algorithms declare as inliers. The number of correct inliers detected by the algorithm can be found from the table by taking the difference of fifth and fourth columns.

It can be inferred from Table 5.1 that the *IMOR* algorithm cannot detect a large number of outliers, and therefore, the fundamental matrix estimate computed by using this contaminated set will give

inferior results. As expected, the reconstruction by using only the *IMOR* algorithm has been unacceptable during the performed simulations. Although, the results of the *RANSAC* algorithm alone yields very accurate reconstruction results, utilization of the IMOR algorithm as a preprocessing step before *RANSAC* decreases the execution time of the overall outlier rejection algorithm considerable, approximately 40 times. Therefore, it is proposed to jointly utilize the outlier rejection algorithms in a cascaded manner (*IMOR*+*RANSAC*). This combination yields quite improvement for execution time without losing much from the reconstruction quality.

CHAPTER 6

CONCLUSION

6.1 Summary of the thesis

In this thesis, structure from motion problem is addressed for calibrated scenes, including the cases containing independently moving objects. For this purpose, the reconstruction process is divided into sections and each stage is presented separately. The first stage is the fundamental problem of estimating the structure and motion by using only two views. Then, the method is further generalized for more than two view case: the multiple view reconstruction. Finally, multiple views containing independently moving objects (IMOs) are examined and a novel method is proposed by using the parallax based rigidity constraint in order to reject IMOs as well as outliers.

The first section is the computation of the scene structure and motion parameters from two calibrated images. This process starts by finding some point matches between two images. In order to match points for different images, it is necessary to extract salient features from these images. For this purpose, a modified version of the Harris corner detector is utilized. The detector is modified such that the results can be obtained in subpixel accuracy. After the extraction of salient features, a

moderately simple algorithm (in terms of computation complexity) is used in order to match these point features. The matching is performed by examining two main criteria: normalized cross correlation (NCC) and strength measure (SM). NCC is used to measure the similarity of image patches around the feature positions and SM is used to introduce smoothness to the motion vectors by using neighborhood information. Once a set of putative correspondences are determined, the next step is the estimation of the fundamental matrix, which encapsulates the motion parameters of the cameras. For this purpose, normalized eight point algorithm is utilized. This algorithm estimates the fundamental matrix linearly by using only eight correspondences. It is a fast, non-iterative algorithm and its results are comparable to other iterative methods. Although, the estimation results are satisfactory for a set, which is contaminated with outliers, it is necessary to use a robust method to improve its performance. In order to introduce the necessary robustness, a statistical method, random sample consensus (RANSAC), is exploited. RANSAC operates by estimating the model from small random sets of the input and testing the goodness of the model iteratively. The iterations are stopped, when the process is guaranteed to yield a good estimate statistically. The linear robust estimation of the fundamental matrix is followed by a nonlinear minimization algorithm in order to improve the estimate. Levenberg-Marquardt (LM) minimization algorithm is used over the whole set of points labeled as inliers by the RANSAC method. Once the fundamental matrix is estimated and refined, it is decomposed into its motion parameters. Since the calibration information is known, the essential matrix is computed and then, it is decomposed into rotation and translation parameters. By utilizing these parameters,

projection matrices are computed. Finally, using a triangulation algorithm, the positions of the 3D points are calculated. The utilized triangulation algorithm is a projective invariant method which involves finding the roots of a sixth degree polynomial.

The second section is the generalization of the two view algorithm for more than two views. This step is performed by first constructing an initial framework and then building-up the reconstruction of the remaining frames relative to this framework. The initial reconstruction is computed by using the two-view reconstruction algorithm. For adding a new view to the framework, the relative pose and location of this frame is estimated by a robust procedure. A set of projective pairs are formed by relating the framework points and matches computed with the new frame and a previously inserted frame. The projection matrix is then estimated from this projective pairs by using RANSAC. Once the projection matrix is found, by the help of triangulation, some new 3D points are initialized for the framework. Finally, the overall structure is refined via bundle adjustment. This adjustment involves the minimization of the total reprojection error over the whole camera and point locations.

The last section is devoted to the reconstruction from a sequence containing independently moving objects. In order to detect the moving objects, the parallax-based rigidity constraint is used. In the application of this constraint, a background pixel has to be presented to the system as an input with user intervention. For avoiding this interaction, a novel method is proposed for an automatic background pixel selection algorithm. Moreover, in

order to make the system more robust, the results of more than one background pixels are fused.

6.2 Discussions

In the feature detection part of the algorithm, it is observed from the experiments performed (Table 3.2) that for a relatively minor computational load, subpixel accurate feature-detection increases the performance considerably. The resolution is increased to subpixel level by biquadric polynomial fitting (see Appendix B) to the Harris cornerness surface in every local patch. The support rectangle size (N) of the fit is chosen as same with the size of the Gaussian filter used in the Harris detector (N=3). The fitting is tested for different values of the support size (N), and it is observed that if a local maximum exists within the support, the detection of the true maxima may be disrupted due to the inferior approximation of the fit. Therefore, it is recommended to use same sized filters and windows throughout the process in order to avoid local maximum.

Feature matching operation by using only the normalized cross correlation (NCC) measure has been found out to be insufficient, especially for the repetitive textured regions. In the performed experiments, the patches within the repeating regions still yield acceptable results for erroneous matches due to the nature of the measure (Table 3.1). For this reason, a neighbor-based matching measure together with NCC, called the strength measure (SM), is included to the algorithm. The results are improved to be satisfactory (see Figure 3.8 for repetitive textured region results). Although, the complexity of the matching increases by reducing

the number of false matches and increasing correct match number, the required number of iteration for the subsequent robust fundamental matrix estimation stage is reduced.

During the estimation of the fundamental matrix, it is observed that, by a non-linear minimization algorithm, the performance can be improved (Table 3.2). The Levenberg-Marquardt (LM) algorithm is selected for the minimization purposes. From the test results, it is obvious that LM minimization is a crucial part of the overall algorithm and it should not be skipped. Moreover, as a future study, in the robustification stage while incorporating RANSAC algorithm, the goodness of the fundamental matrix may be tested over a random set, instead of using all of the putative matches. This will surely decrease the computation time, however the performance of the system has to be considered.

In order to locate the position of the 3D points from the computed correspondences, four algorithms, namely midpoint method, linear-eigen method, linear least-square method and polynomial triangulation method, are tested. It is observed from Figure 3.18 that polynomial triangulation behaves best under projective transformation. On the other hand, the midpoint method gives the inferior results and should be avoided. As it is observed from Figure 3.19, almost all of the methods behave equally and can be used alternatively for Euclidean reconstruction problems.

Multiple view reconstruction method presented in Chapter 4 makes use of projection pairs in order to relate new frames with the current framework, i.e., projection pairs are used to calculate the projection matrix of the new view. However, it is not

guaranteed to have an outlier-free set of projection pairs and, therefore, it is required to use a robust method in the estimation as well. During the simulations involving the direct (non-robust) estimation of the projection matrix, the algorithm lost track of the cameras in most cases and the reconstructions are unacceptable due to the erroneous estimation of the orientation and location information.

Finally, a new approach for using parallax-based rigidity constraint, in order to reject outliers and also independently moving objects, is proposed. In the exploitation of this constraint, it is necessary to locate a pixel that is guaranteed to be on the background. By calculating the change of the projective 3D structure of a point from the residual parallax vectors with respect to this selected background point, the point is decided to be an inlier or an outlier. In the experiments, it is noticed that the selection of this background point is quite critical. It should not be selected on the dominant plane due to the fact that the remaining residual vectors on the dominant plane are mostly due to noise and errors made in the plane registration stage. Moreover, it is also noticed that a threshold based system will be inadequate to discriminate between background and foreground vectors due to the dependency of the errors to input scene conditions. Hence, it is proposed to use a selection algorithm for the best consistent matches. On the other hand, the main problem of this method is its requirement to specify the percentage of the background vectors to the overall set. If it is specified less than the correct value, some background vectors will be rejected and if it is more, some outliers will remain. However, this is not a serious problem

for large sets of vectors, since losing some background vectors by specifying a modest percent can be still tolerable.

6.3 Future Work

The consecutive frames of a video sequence have very small baseline distances. Therefore, the tested system could not calculate the rotation and translation parameters reliably for such consecutive video frames. In order to adapt the system to take video input, some measure to compute the distance between the frames might be included. In this manner, during the inclusion of a new frame to the system, the new frame might be related to more than one frame which are detected to be close. Another issue is the uncalibrated camera case. A self-calibration routine should be included to the system in order to have more flexibility with the input images. Finally, a dense matching and reconstruction may be incorporated in order to form more detailed reconstructions.

REFERENCES

[1] Hartley, R. and Zisserman A., *Multiple view geometry in Computer Vision*, Cambridge University Press, New York. 2000

[2] Faugeras O., *Three Dimensional Computer Vision: A Geometric Viewpoint,* MIT Press, Cambridge, Massachusetts, London, England 1993

[3] Maybank, S. J and Faugeras O., *A theory of self-calibration of a moving camera.* International Journal of Computer Vision, 8(2): 123-152, Aug 1992

[4] Tsai R.Y. , *A versatile camera calibration technique for high-accuracy 3D machine vision metrology using off-the-shelf TV cameras and lenses.* IEEE Journal of Robotics and Automation, 3(4):323-344, Aug 1987

[5] Hartley, R. *Self calibration from multiple views with a rotating camera*. Proc. Third European Conference on Computer Vision, J.-O. Eklundh, ed., vol. 800-801, pp.471-478, May 1994

[6] Zhang, Z., *A flexible new technique for camera calibration.* IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 22, no. 11, pp. 1330-1334, Nov 2000

[7] Zhang, Z., *Camera calibration with one-dimensional objects*, Proc. European Conf. Computer Vision, vol. 4, pp. 161-174, May 2002

[8] Zhang, Z., *Camera calibration with one-dimensional objects.* IEEE Trans. On Pattern Analysis and Machine Intelligence, vol.26, no.7, July 2004

[9] Horn, B. K. P., (1986) *Robot Vision,* MIT Press, Cambridge, Massachusetts and McGraw-Hill, New York

[10] Sturm, P., Maybank, S., *On plane based camera calibration: A general algorithm, Singularities, Applications.* Proc. IEEE Conf. Computer Vision and Pattern Recognition, pp432-437, June 1999

[11] Tapper M., McKerrow P.J., Abrantes, J. *Problems encountered in the implementation of Tsai's algorithm for camera calibration.* Proc. 2002 Australasian Conference on Robotics and Automation, Auckland, Nov 2002.

[12] Weng J, Huang T.S., Ahuja N., "Motion and Structure from Image Sequences", Springer Series in Information Sciences 29, 1993

[13] Harris, C., Stephens, M. A Combined Corner and Edge Detector. In 4th Alvey Vision Conference, S. 147–151. 1988

[14] Faugeras Olivier, Luong Quang-Tuan, Maybank S.J., "Camera Self-Calibration: Theory and Experiments", Proceedings of the 2nd European Conference on Computer Vision, 321-334, 1992

[15] Luong Quang-Tuan, Faugeras Olivier, "Self-Calibration of a Moving Camera from Point Correspondences and Fundamental Matrices", International Journal of Computer Vision, 22(3), 261-289, 1997

[16] Zeller Cyril, Faugeras Olivier, "Camera Self-Calibration from Video Sequences: the Kruppa Equations Revisited", INRIA, 1996

[17] Hartley Richard, "Kruppa's Equations Derived from the Fundamental Matrix", IEEE Transactions on Pattern Analysis and Machine Intelligence, 19, 2, 1997

[18] Lei Cheng, "A New Approach to Solving Kruppa Equations for Camera Self-Calibration", ICPR, 2002

[19] Fusiello A., "Uncalibrated Euclidean Reconstruction: A Review", Image and Vision Computing, 18, 555-563, 2000

[20] Lourakis Manolis, Deriche Rachid, "Camera Self-Calibration Using the Singular Value Decomposition of the Fundamental Matrix", Asian Conference on Computer Vision (ACCV), 2000

[21] Csurka G., Zeller C., Zhang Z., Faugeras O., *Characterizing the Uncertainty of the Fundamental Matrix*, INRIA Report no.2560, 1995

[22] Heyden A., Astrom K., "Euclidean Reconstruction from Constant Intrinsic Parameters", Proc. 13th International Conference on Pattern Recognition, pages 339-343, 1996

[23] Pollefeys Marc, Gool L. Van, " Self-calibration from the absolute conic on the plane at infinity", Proc. Computer Analysis of Images and Patterns, pages 175-182, 1997

[24] Pollefeys Marc, Gool L. Van, "A stratified approach to self-calibration", Proc. 1997 Conference on Computer Vision and Pattern Recognition, pages 407-412, 1997

[25] Pollefeys Marc, Gool L. Van, Oosterlinck A., "The Modulus Constraint: A new Constraint for Self Calibration", Proc 13th International Conference on Pattern Recognition, pages 349-357, 1996

[26] G. Xu and Z. Zhang, *Epipolar geometry in stereo, motion and object recognition*. Kluwer academic publishers, 1996

[27] H. C. Longuet-Higgins. *A computer algorithm for reconstructing a scene from two projections*. Nature. 293:133-135, September 1981

[28] H.C. Longuet-Higgins. The reconstruction of a scene from two projections – configurations that defeat the 8-point algorithm. In Proc. First Conf. Artificial Intelligence Applications, pages 395-397, Denver, Colorado, 1984.

[29] J. Weng, T.S. Huang, and N. Ahuja. Motion and structure from two perspective views: Algorithms, error analysis and error estimation. IEEE Trans. PAMI, 11(5):451-476, May 1989.

[30] Lourakis, M., *Levenberg-Marquardt non-linear least squares algorithms in C/C++,* http://www.ics.forth.gr/~lourakis/levmar/, 2004

[31] Hartley R., *In Defense of the Eight-Point Algorithm,* IEEE Transactions on pattern analysis and machine intelligence, vol 19, no. 6, June 1997

[32] 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, Volume 24 Number 6, June 1981

[33] Peter J. Rousseeuw and Annick M. Leroy. *Robust Regression and Outlier Detection*. Wiley, 1987.

[34] P. H. S. Torr and A. Zisserman. MLESAC: A new robust estimator with application to estimating image geometry. *Computer Vision and Image Understanding*, 78:138–156, 2000.

[35] Press, W. H. et al, *Numerical recipes in C: the art of scientific computing,* Cambridge University Press, New York, 1992

[36] Z. Zhang, R. Deriche, O. Faugeras, Q.T. Luong*, A Robust Technique for Matching Two Uncalibrated Images Through the Recovery of the Unknown Epipolar Geometry,* INRIA, Report No. 2273, 1994

[37] M. Pollefeys, L. Van Gool, M. Vergauwen, F. Verbiest, K. Cornelis, J. Tops, R. Koch, Visual modeling with a hand-held camera, International Journal of Computer Vision 59(3), 207-232, 2004.

[38] P.A. Beardsley, A. Zisserman and D.W. Murray, *Sequential Updating of Projective and Affine Structure from Motion*, International Journal of Computer Vision 23(3), 235–259 (1997)

[39] C. Tomasi and T. Kanade, "Shape and motion from image streams under orthography: A factorization approach", International Journal of Computer Vision, 9(2):137-154, 1992

[40] B. Triggs, P. McLauchlan, R. Hartley, A. Fiztgibbon, "Bundle Adjustment – A Modern Synthesis", In B. Triggs, A. Zisserman, R. Szeliski (Eds.), *Vision Algorithms: Theory and Practice*, LNCS Vol.1883, pp.298-372, Springer-Verlag, 2000.

[41] M.I.A. Lourakis and A. A. Argyros, *The Design and Implementation of a generic sparse bundle adjustment software package based on the Levenberg-Marquardt algorithm,* http://www.ics.forth.gr/ ~lourakis/sba/, 2004

[42] K. Mikolajczyk and C. Schmid. An affine invariant interest point detector. In *Proceedings of the 7th European Conference on Computer Vision, Copenhagen, Denmark*, volume I, pages 128–142, May 2002.

[43] Schmid, C., Mohr, R., Bauckhage, C., Comparing and Evaluating Interest Points. In IEEE International Conference on Computer Vision, S. 230–235. 1998.

[44] Bayram I. *Interest point matching across arbitrary views*, M.S. Thesis. June 2004

[45] Dufournaud Y., Schmid C., Horaud R. : Image matching with scale adjustment. Research Report, no. 4428, 2002

[46] Kanazawa Y., Kanatani K. : Robust image matching under a large disparity. In proceedings of Workshop on Science of Computer Vision, Okayama, Japan, pp. 46-52, September 2002.

[47] A. Rosenfeld, R.A. Hummel, and S.W. Zucker. Scene labeling by relaxation operations. IEEE Trans. SMC, 6(4):420{433, 1976.

[48] Maronna, R.A., 1976, Robust M-estimators of multivariate location and scatter. Ann. Stat. 4:51-67

[49] Canny, J. F.: A Computational Approach to Edge Detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Bd. 8(6):679–698, 1986.

[50] Deriche, R., Faugeras, O.: A Computational Approach for Corner and Vertex Detection. *International Journal of Computer Vision*, Bd. 1(2):167–187, 1993.

[51] Smith, S. M., Brady, J. M.: SUSAN - A New Approach to Low Level Image Processing. *International Journal of Computer Vision*, Bd. 23(1):45–78, 1997

[52] Tekalp, A. M., *Digital Video Processing,* Prentice Hall, 1995

[53] Faugeras, O., Luong, Q., Papadopoulo, T., *The Geometry of Multiple Images: The Laws that govern the formation of multiple images of a scene and some of their applications,* MIT Press, 2001

[54] Hartley, R. I, Gupta, R., Chang, T., *Stereo from Uncalibrated cameras,* IEEE Computer Society Conference on Proc. CVPR'92, 1992

[55] Hartley, R. I, and Sturm, P. *Triangulation,* Computer Vision and Image Understanding, Vol. 68, No.2, pp.146-157, 1997 Article No.IV970547

[56] Beardsley, P. A., Zisserman, A. and Murray, D. W., *Navigation using affine structure from motion, in Computer Vision,* ECCV'94, LNCS Series 801, pp.85-96, Springer-Verlag, Berlin/New York, 1994

[57] Haralick, R.M., Shapiro, L.G., *Computer and Robot Vision,* Addison-Wesley Pub. Co., 1992

[58] Strecha C., *Test Images*, http://www.esat.kuleuven.ac.be/~cstrecha /test images, 2004

[59] Oxford University, *Visual geometry group homepage*, http://www.robots.ox. ac.uk/~vgg/data2.html, 2004

[60] Irani, M. , Anandan, P. , "A Unified Approach to Moving Object Detection in 2D and 3D Scenes", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 20, No. 6, June 1998

[61] Lourakis, M.I.A. , Argyros, A.A., Orphanoudakis, S.C. , "Independent 3D Motion Detection Using Residual Parallax Normal Flow Fields", ICCV, pp. 1012-1017, 1998

APPENDIX A

ZHANG'S CAMERA CALIBRATION ALGORITHM

In this relatively recent method, a coplanar calibration pattern is captured a few times with different orientations by moving either the camera or the model plane. The world coordinate system is assumed to be aligned with the model plane, i.e. calibration pattern is on $z = 0$ plane and the x- and y-axes are parallel to the pattern features. The feature points are automatically detected from the captured images. As in [4], only this information is used in order to extract intrinsic, extrinsic and distortion parameters of the camera.

The estimation of the unknown calibration parameters in principle is quite similar to the method by Tsai [4]. The major difference is that no strict motion is defined for the camera to gather some depth information. The assumption of coinciding the z=0 plane with the calibration pattern simplified the formulation of the procedure a lot. A homography between 3D and 2D measured image coordinates of the system is defined, as

$$s\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = A\begin{bmatrix} r_1 & r_2 & r_3 & t \end{bmatrix}\begin{bmatrix} X \\ Y \\ 0 \\ 1 \end{bmatrix} = A\begin{bmatrix} r_1 & r_2 & t \end{bmatrix}\begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} = H\begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} \qquad \text{(A.1)}$$

where $H = \begin{bmatrix} h_1 & h_2 & h_3 \end{bmatrix}$ and

$$s\tilde{m} = H\tilde{M}, \qquad \tilde{m} = \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \text{ and } \tilde{M} = \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} \qquad \text{(A.2)}$$

Given an image of the model plane, a homography can be estimated easily [1]. An estimate of the H can be determined by using nonlinear least square methods, after minimizing,

$$\zeta = \sum_i \left\| m_i - \hat{m}_i \right\|^2 \text{ where } \hat{m}_i = \frac{1}{\bar{h}_3^T M_i}\begin{bmatrix} \bar{h}_1^T M_i \\ \bar{h}_2^T M_i \end{bmatrix} \qquad \text{(A.3)}$$

Such a minimization can be performed by using Levenberg-Marquardt method. However, an initial guess is required, as usual. This initial guess is obtained as the right singular vector of L, where L is equal to the concatenation of equations obtained by rearranging (A.2), i.e.,

$$\begin{bmatrix} \tilde{M}^T & 0^T & -\tilde{u}M^T \\ 0^T & \tilde{M}^T & -\tilde{v}M^T \end{bmatrix} x = 0 \qquad \text{(A.4)}$$

After finding the homography, this matrix is decomposed into A, R and t by the following procedure: from Equation A.1, one has,

$$[\, h_1 \quad h_2 \quad h_3 \,] = \lambda A [r_1 \quad r_2 \quad t] \tag{A.5}$$

Since columns of the R matrix are orthonormal, one should have

$$\begin{aligned} h_1^T A^{-T} A^{-1} h_2 &= 0 \\ h_1^T A^{-T} A^{-1} h_1 &= h_2^T A^{-T} A^{-1} h_2 \end{aligned} \tag{A.6}$$

It can be shown that $B = A^{-T} A^{-1}$ has five distinct parameters [6]. Performing the same strategy as it is achieved for the solution of H, one can compute the parameters of the B matrix easily. Once B is estimated, A matrix parameters are obtained by,

$$\begin{aligned} v_0 &= (B_{12}B_{13} - B_{11}B_{23})/(B_{11}B_{22} - B_{12}^2) \\ \lambda &= B_{33} - [B_{13}^2 + v_0(B_{12}B_{13} - B_{11}B_{23})]/B_{11} \\ \alpha &= \sqrt{\lambda / B_{11}} \\ \beta &= \sqrt{\lambda B_{11}/(B_{11}B_{22} - B_{12}^2)} \\ \gamma &= -B_{12}\alpha^2\beta/\lambda \\ u_0 &= \gamma v_0/\beta - B_{13}\alpha^2/\lambda \end{aligned} \tag{A.7}$$

Once A is determined, the extrinsic parameters for each image is readily computed

$$\begin{aligned} r_1 &= \lambda A^{-1} h_1 \\ r_2 &= \lambda A^{-1} h_2 \quad \text{with } \lambda = 1/\left\|A^{-1}h_1\right\| = 1/\left\|A^{-1}h_2\right\| \\ r_3 &= r_1 \times r_2 \\ t &= \lambda A^{-1} h_3 \end{aligned} \tag{A.8}$$

After the initial estimates are determined, all of the parameters including lens distortion are refined using a non-linear minimization technique over the cost function:

$$\Gamma = \sum_{i=1}^{n} \sum_{j=1}^{m} \left\| m_{ij} - \hat{m}\left(A, R_i, t_i, M_j\right) \right\|^2 \qquad \text{(A.8)}$$

where $\hat{m}\left(A, R_i, t_i, M_j\right)$ is the projection of the point M_j in image i, according to Equation A.1.

APPENDIX B

BIQUADRIC POLYNOMIAL FITTING TO THE CORNERNESS SURFACE

For a given corner pixel we would like to fit a biquadric polynomial. For every (x, y, R) pair, in the N-neighborhood of the pixel, there exist N equations of the form

$$K_1 x_i^2 + K_2 y_i^2 + K_3 x_i y_i + K_4 x_i + K_5 y_i + K_6 = R_i \qquad \text{(B.1)}$$

where (x_i, y_i) values are computed taking the (x, y) as the origin.

Stacking these equations in the form of $AX = B$ where

$$A = \begin{bmatrix} x_1^2 & y_1^2 & x_1 y_1 & x_1 & y_1 & 1 \\ x_2^2 & y_2^2 & x_2 y_2 & x_2 & y_2 & 1 \\ . & . & . & . & . & . \\ x_{NXN}^2 & y_{NXN}^2 & x_{NXN} y_{NXN} & x_{NXN} & y_{NXN} & 1 \end{bmatrix} \qquad \text{(B.2)}$$

$$B = \begin{bmatrix} R_1 & R_2 & . & . & . & R_{NxN} \end{bmatrix}^T \qquad \text{(B.3)}$$

$$X = \begin{bmatrix} K_1 & K_2 & K_3 & K_4 & K_5 & K_6 \end{bmatrix}^T \qquad \text{(B.4)}$$

one should get a system of linear equations. Solving for X using the pseudo inverse of the A matrix,

$$X = (A^T A)^{-1} A^T B \tag{B.5}$$

and computing the peak of the polynomial by taking the derivative of (B.1) with respect x and y separately and equating those equations to zero, the following relations are obtained

$$x_0 = \frac{K_3 K_5 - 2K_2 K_4}{4K_1 K_2 - K_3 K_3} \quad \text{and} \quad y_0 = \frac{K_3 K_4 - 2K_1 K_5}{4K_1 K_2 - K_3 K_3} \tag{B.6}$$

Hence, the final corner pixel value is (x+x0, y+y0)

APPENDIX C

LEVENBERG-MARQUARDT MINIMIZATION ALGORITHM

The Levenberg-Marquardt (LM) algorithm is an iterative technique that is used to solve non-linear least squares problems. It has become a standard technique and used extensively in many computer vision problems. LM is a combination of steepest descent and the Gauss-Newton method. By changing a single parameter, the algorithm swings between these two methods. When the current estimate of the solution is away from the correct one, algorithm operates in the steepest descent mode and when the solution is close to the correct one, it operates in the Gauss-Newton mode.

Let f be a functional which maps a parameter vector p to an estimated measurement vector $\hat{x}$, $\hat{x} = f(p)$. An initial parameter vector, p_0, and a measurement vector, X, is provided as well, and it is desired to find the best result, minimizing the squared distance $\|\varepsilon\|^2$ with error $\varepsilon = X - X_{cap}$. LM algorithm approximates the functional f with a linear function around the current parameter vector p. For a small $\|\delta_p\|$, the Taylor series expansion of f leads to

$$f(p + \delta_p) \approx f(p) + J\delta_p \tag{C.1}$$

where J is the Jacobian matrix. LM tries to find the best parameter vector p^+ iteratively. Hence, it is required to find a $\|\delta_p\|$ that minimizes the error

$$\|x - f(p + \delta_p)\| = \|x - f(p) - J\delta_p)\| = \|\varepsilon - J\delta_p\|$$

The solution to the minimization of problem of $\|\varepsilon - J\delta_p\|$ is

$$J^T(\varepsilon - J\delta_p) = 0 \tag{C.2}$$

$$J^T J\delta_p = J^T\varepsilon \tag{C.3}$$

The matrix $J^T J$ is an approximation to the second order derivatives, the Hessian matrix. Instead of solving (C.3), LM solves a modified version of this equation, denoted as the *augmented normal equations:*

$$(J^T J + \lambda I)\delta_p = J^T\varepsilon,\ \ \lambda > 0 \tag{C.4}$$

where λ is called as the *damping term*. The update of λ is performed according to the change in error term. If the update term causes the error to decrease, then the change is accepted and λ term is decreased. On the other hand, if the error increases, the damping term is increased and (C.4) is solved again with the new λ without accepting any change until the error is reduced. For the practical use of the LM algorithm, the method by Laurakis [30] is implemented. The pseudo-code for the algorithm is:

Algorithm C.1:

Input: Given a vector function $f : R^m \to R^n$ with $n \geq m$, a measurement vector $x \in R^n$ and an initial parameters estimate $p_0 \in R^m$

Output: A vector $p^+ \in R^m$ minimizing $\|x - f(p)\|^2$

$k := 0; v := 2; p := p_0;$
$A := J^T J; \varepsilon_p := x - f(p); g := J^T \varepsilon_p;$
stop $:= (\|g\|_\infty \leq \varepsilon_1); \mu := \tau \max_{i=1,\ldots,m}(A_{ii});$
while(not stop) and $(k < k_{\max})$
    $k := k + 1;$
    repeat
        Solve $(A + \mu I)\delta_p = g;$
        if($\|\delta_p\| \leq \varepsilon_2 \|p\|$)
            stop := true;
        else
            $p_{new} := p + \delta_p;$
            $\rho := (\|\varepsilon_p\|^2 - \|x - f(p_{new})\|^2) / (\delta_p^T (\mu \delta_p + g));$
            if $\rho > 0$
                $p = p_{new};$
                $A := J^T J; \varepsilon_p := x - f(p); g := J^T \varepsilon_p;$
                stop $:= (\|g\|_\infty \leq \varepsilon_1);$
                $\mu := \mu * \max(\frac{1}{3}, 1 - (2\rho - 1)^3); v := 2;$
            else
                $\mu := \mu * v; v := 2 * v;$
            endif
        endif
    until($\rho > 0$) or (stop)
endwhile

APPENDIX D

SPARSE BUNDLE ADJUSTMENT

This section shows the development of a sparse bundle adjustment algorithm, which is obtained by using the LM algorithm presented in Appendix C. The development and the notation mostly follow the technical report in [41].

Assume that n 3D points are seen in m views and let x_{ij} be the projection of the ith point on the jth image. Bundle adjustment (BA) is the refinement of a set of initial camera and structure parameter estimates for finding a set of parameters that accurately predict the locations of the observed n points in the set of m available images. Representing the jth camera parameters as a_j and ith point as b_i, the minimized cost function is the total reprojection error:

$$\min_{a_j, b_i} \sum_{i=1}^{n} \sum_{j=1}^{m} d(Q(a_j, b_i), x_{ij})^2 \qquad \text{(D.1)}$$

where $Q(a_j, b_i)$ is the predicted projection of the ith point on the jth image and $d(x, y)$ represents the Euclidean distance between inhomogeneous points, denoted by x and y. The projection

expression $Q(a_j, b_i)$ is defined general enough to allow any camera and structure parameterization. If the dimension of a_j is equal to d_1 and the dimension of b_i is equal to d_2, then the above minimization has a total dimension of nd_2+md_1, which is a quite large number, even for a moderate sized BA problem.

The formulation of the BA is given as,

A parameter vector containing the whole structure and motion parameters is represented as:

$$P = (a_1^T, a_2^T, \ldots\ldots a_m^T, b_1^T, b_2^T \ldots\ldots b_n^T)^T \qquad \text{(D.2)}$$

and the measurement vector containing all the measured image coordinates is represented as:

$$X = (x_{11}^T, \ldots\ldots x_{1m}^T, x_{21}^T, \ldots\ldots x_{2m}^T, \ldots\ldots x_{n1}^T, \ldots\ldots x_{nm}^T)^T \qquad \text{(D.3)}$$

Let the initial parameter vector be P_0 and for each parameter estimate, the estimated measurement vector be $\hat{X}$. The relationship between P_0 and $\hat{X}$ is given by

$$\hat{X} = f(P) \qquad \text{(D.4)}$$

with

$$\hat{X} = (\hat{x}_{11}^T, \ldots\ldots \hat{x}_{1m}^T, \hat{x}_{21}^T, \ldots\ldots \hat{x}_{2m}^T, \ldots\ldots \hat{x}_{n1}^T, \ldots\ldots \hat{x}_{nm}^T)^T \qquad \text{(D.5)}$$

where $\hat{x}_{ij} = Q(a_j, b_i)$

Thus, BA is equal to minimizing the squared error $\varepsilon^T\varepsilon$ where $\varepsilon = X - \hat{X}$ over P. This minimization problem may be solved using the LM algorithm in order to iteratively solve the augmented normal equations:

$$(J^T J + \lambda I)\delta_p = J^T\varepsilon, \quad \lambda > 0 \tag{D.6}$$

where J is the Jacobian of f and δ is the sought update to the P estimate.

The sparseness of the above problem will be explained by using n=3 points and m=2 views without losing any generality to keep the demonstration manageable.

The measurement vector is $X = (x_{11}^T, x_{12}^T, x_{21}^T, x_{22}^T, x_{31}^T, x_{32}^T)^T$ and the parameter vector becomes $P = (a_1^T, a_2^T, b_1^T, b_2^T, b_3^T)^T$. Notice that $\frac{\partial \hat{x}_{ij}}{\partial a_k} = 0, \forall j \neq k$ and $\frac{\partial \hat{x}_{ij}}{\partial b_k} = 0, \forall i \neq k$. Let A_{ij} and B_{ij} denote $\frac{\partial \hat{x}_{ij}}{\partial a_j}$ and $\frac{\partial \hat{x}_{ij}}{\partial b_i}$, respectively. The LM updating vector δ can be partitioned into camera and structure parameters as $(\delta_a^T, \delta_b^T)^T$ and further as $(\delta_{a1}^T, \delta_{a2}^T, \delta_{b1}^T, \delta_{b2}^T, \delta_{b3}^T)^T$. Using the outlined notation, the Jacobian can be calculated as

$$\frac{\partial X}{\partial P} = \begin{bmatrix} A_{11} & 0 & B_{11} & 0 & 0 \\ 0 & A_{12} & B_{12} & 0 & 0 \\ A_{21} & 0 & 0 & B_{21} & 0 \\ 0 & A_{22} & 0 & B_{22} & 0 \\ A_{31} & 0 & 0 & 0 & B_{31} \\ 0 & A_{32} & 0 & 0 & B_{32} \end{bmatrix} \tag{D.7}$$

From (D.7), it is clearly observed that the Jacobian matrix is a sparse matrix. Substituting this expression into the $J^T J$ term in the left hand side of (D.6),

$$J^T J = \begin{bmatrix} \sum_{i=1}^{3} A_{i1}^T A_{i1} & 0 & A_{11}^T B_{11} & A_{21}^T B_{21} & A_{31}^T B_{31} \\ 0 & \sum_{i=1}^{3} A_{i2}^T A_{i2} & A_{12}^T B_{12} & A_{22}^T B_{22} & A_{32}^T B_{32} \\ B_{11}^T A_{11} & B_{12}^T A_{12} & \sum_{j=1}^{2} B_{1j}^T B_{1j} & 0 & 0 \\ B_{21}^T A_{21} & B_{22}^T A_{22} & 0 & \sum_{j=1}^{2} B_{2j}^T B_{2j} & 0 \\ B_{31}^T A_{31} & B_{32}^T A_{32} & 0 & 0 & \sum_{j=1}^{2} B_{3j}^T B_{3j} \end{bmatrix} \quad \text{(D.8)}$$

Denoting the $\sum_{i=1}^{3} A_{ij}^T A_{ij}$, $\sum_{j=1}^{2} B_{1i}^T B_{1i}$, and $A_{ij1}^T B_{ij}$ by U_j, V_i and W_{ij}, the above matrix is equal to

$$J^T J = \begin{bmatrix} U_1 & 0 & W_{11} & W_{21} & W_{31} \\ 0 & U_1 & W_{12} & W_{22} & W_{32} \\ W_{11}^T & W_{12}^T & V_1 & 0 & 0 \\ W_{21}^T & W_{22}^T & 0 & V_2 & 0 \\ W_{31}^T & W_{32}^T & 0 & 0 & V_3 \end{bmatrix} \quad \text{(D.9)}$$

Using (D.7), the right hand side of (D.6) can be expanded as

$$\begin{bmatrix} \sum_{i=1}^{3} A_{i1}^T \varepsilon_{i1} \\ \sum_{i=1}^{3} A_{i2}^T \varepsilon_{i2} \\ \sum_{j=1}^{2} B_{1j}^T \varepsilon_{1j} \\ \sum_{j=1}^{2} B_{2j}^T \varepsilon_{2j} \\ \sum_{j=1}^{2} B_{3j}^T \varepsilon_{3j} \end{bmatrix} \tag{D.10}$$

Denoting $\sum_{i=1}^{3} A_{ij}^T \varepsilon_{ij}$ and $\sum_{j=1}^{2} B_{ij}^T \varepsilon_{ij}$ by ε_{a_j} and ε_{b_i} respectively, (D.7) can be written as

$$\begin{bmatrix} U_1 & 0 & W_{11} & W_{21} & W_{31} \\ 0 & U_2 & W_{12} & W_{22} & W_{32} \\ W_{11}^T & W_{12}^T & V_1 & 0 & 0 \\ W_{21}^T & W_{22}^T & 0 & V_2 & 0 \\ W_{31}^T & W_{32}^T & 0 & 0 & V_3 \end{bmatrix} \begin{bmatrix} \delta_{a1} \\ \delta_{a2} \\ \delta_{b1} \\ \delta_{b2} \\ \delta_{b3} \end{bmatrix} = \begin{bmatrix} \varepsilon_{a1} \\ \varepsilon_{a2} \\ \varepsilon_{b1} \\ \varepsilon_{b2} \\ \varepsilon_{b3} \end{bmatrix} \tag{D.11}$$

Denoting $U^* = \begin{bmatrix} U_1^* & 0 \\ 0 & U_2^* \end{bmatrix}$, $V^* = \begin{bmatrix} V_1^* & 0 & 0 \\ 0 & V_2^* & 0 \\ 0 & 0 & V_3^* \end{bmatrix}$ and $W = \begin{bmatrix} W_{11} & W_{21} & W_{31} \\ W_{12} & W_{22} & W_{32} \end{bmatrix}$ where * denotes the augmentation of the diagonal elements, allows the augmented normal equation to be further compacted to

$$\begin{bmatrix} U^* & W \\ W^T & V^* \end{bmatrix} \begin{bmatrix} \delta_a \\ \delta_b \end{bmatrix} = \begin{bmatrix} \varepsilon_a \\ \varepsilon_b \end{bmatrix} \tag{D.12}$$

The solution of the above equation may be found by left multiplying the equation with

$$\begin{bmatrix} I & -WV^{*-1} \\ 0 & I \end{bmatrix} \tag{D.13}$$

resulting

$$\begin{bmatrix} U^* - WV^{*-1}W^T & 0 \\ W^T & V^* \end{bmatrix} \begin{bmatrix} \delta_a \\ \delta_b \end{bmatrix} = \begin{bmatrix} \varepsilon_a - WV^{*-1}\varepsilon_b \\ \varepsilon_b \end{bmatrix} \tag{D.14}$$

From this equation, the first the update term δ_a is found from the upper equality and then substituting for δ_a, the value of the δ_b is found.

$$(U^* - WV^{*-1}W^T)\delta_a = \varepsilon_a - WV^{*-1}\varepsilon_b \tag{D.15}$$

$$V^*\delta_b = \varepsilon_b - W^T\delta_a \tag{D.16}$$

The rest of the algorithm is same as the LM algorithm outlined in Algorithm C.1. The only difference is the calculation of the update terms. They are calculated using Equations D.15 and D.16.

APPENDIX E

QUATERNION REPRESENTATION OF THE ROTATION MATRIX

A quaternion represents a three-dimensional rotation as a four-component row vector of unit length:

$$q = \left[n_x \sin(\theta/2) \quad n_y \sin(\theta/2) \quad n_z \sin(\theta/2) \quad \cos(\theta/2) \right] = [\underline{q_v} \quad q_s] \tag{E.1}$$

with $\|q\| = \|q_v\| + q_s^2 = 1$

This definition uses the axis-angle form of rotation information. In this form, a rotation is specified by an axis and a rotation angle. The axis is (n_x, n_y, n_z) and the rotation angle is θ. The rotation is performed according to the right-hand rule.

The relation between the rotation form and axis angle form is given as:

$$R = \exp(\theta[n]_x) = I + [n]_x \sin\theta + [n]_x^2(1 - \cos\theta) \qquad \text{(E.2)}$$

where

$$[n]_x = \begin{bmatrix} 0 & -n_z & n_y \\ -n_z & 0 & -n_x \\ n_y & -n_x & 0 \end{bmatrix} \qquad \text{(E.3)}$$