CAMERA CALIBRATION AND STRUCTURE FROM MOTION PROCEDURES ON HUMANOID ROBOT NAO

Ozan GÖNENÇ

Laboratoire Electronique, Informatique et Image (Le2i)
UMR CNRS 5158
Université de Bourgogne

Supervisor
Prof. David Fofi

A Thesis Submitted for the Degree of
MSc Erasmus Mundus in Vision and Robotics (VIBOT)

· 2012 ·

Abstract

The interest in computer vision applications of mobile robotics is significantly increased in past two decades. One of the most popular problems of computer vision is called Structure from Motion(SfM). SfM is the process of recovering the three-dimensional structure of a scene and related camera motion from a sequence of images. Effecting the success of many applications of computer vision, camera calibration is a critical process for Structure from Motion. For particular methods of SfM, camera calibration is a prerequisite and it is the process of finding the characteristics of the camera. Thus, the objective of this thesis is three-fold; first, we discuss the camera calibration problem in detail, and provide a review on the existing camera calibration methods and explain a particular camera calibration algorithm in detail. Then we explain the problem of SfM and provide a SfM algorithm. Finally, we discuss the applicability of the camera calibration and SfM methods, in the light of first two previous discussions, on a humanoid NAO and provide an application of camera calibration and SfM for NAO. Hence this thesis is a study of camera calibration and structure from motion methods of computer vision on the Humanoid NAO and it contributes by applying those challenging tasks on a mobile robot(NAO).

Contents

List of Figures

List of Tables

Acknowledgments

I would like to express my deep-felt gratitude to my advisor, Prof. David Fofi of the Lab of Electronics, Computer Science and Image Processing(Le2i) at The University of Burgundy, for his advice and constant support.

I also wish to thank, Dr. Ralph Seulin of the Lab of Electronics, Computer Science and Image Processing(Le2i) at The University of Burgundy. His suggestions, encouragement, comments and additional guidance were invaluable to the completion of this work. As a special note, Dr. Ralph Seulin was extremely helpful in providing the additional guidance and expertise I needed in order to complete this work.

Additionally, I want to thank Dr. Yohan Fougerolle showing me the way to complete my degree with his valuable advices and guidance.

Of course, I am also indebted to my parents, my brother for their great patience and support which was very important to me. Also, I would like to thank my friend Alper Akçöltekin. In addition to undergraduate years, we spent a lot of time with Alper during our graduate studies. He was always very helpful and supportive to me.

And finally, I would like to thank my girlfriend for putting up with me during the development of this work with continuing, loving support. I do not have the words to express all my feelings here, only that I love you Özge.

Chapter 1

Introduction

1.1 Motivation

The utilization of the humanoid robots is drastically increased in last 20 years both in research and industry. The aim of humanoid robotics research is to gain better understanding of human body structure and behaviours and the ultimate goal is to create a humanoid that reaches the capabilities of a human in terms of motion, perception, information interpretation and intelligence. On the other hand mobile and humanoid-like robots is being used and developed in robotics industry. Besides the production line robotics, human-like robots are being a part of daily life mostly in the area of personal robotics for various purposes.

Although the information processing mechanisms of the human brain is extremely complex, mankind was able to develop human like robots and come up with solutions that at least resemble the results of human information processing processes like recognizing objects, reconstructing layouts and understand the geometric space specifically with the usage of computer vision techniques. One of the most important sensor organ for human cognition[1] is the eyes. When it comes to humanoid robots cameras are considered to be the most fundamental and most used input sensors as it is a replacement for human eyes. Therefore, visual input and vision-based solutions for humanoids have been widely used for robot navigation, localization, pattern recognition and scene understanding as first steps of simulated human information processing for humanoid robots. Hence computer vision, a branch of artificial intelligence, is seeking the answers of how to resemble the human vision system. Although the inverse problem of obtaining 3-dimensional reconstruction from 2-dimensional images(as the output is the same, for humans; a 2D image on human retina and for cameras a 2D image) is a hard problem, hu-

[1] the field of study which is focused on how humans learn from sensory information in order to acquire perceptual and motor skills.

mans have the ability to solve this problem. On the other hand, as a computer vision problem reconstructing a 3-dimensional scene is a difficult problem.

In order to improve the capability of a humanoid robot, the visual sensor(camera) characteristics should be known, because the output of a camera is a two-dimensional image and the only way to go further in problems like depth-sensing and 3D reconstruction-as obtaining metric information of the environment that will lead a humanoid to accomplish more complicated tasks-is to know the characteristics of a camera.

Structure from Motion is the method of reconstructing 3D points and recovering the camera motion from 2D images that are taken from different position of the same scene. The reconstruction of rigid objects and motion estimation of the camera, from a sequence of images is a challenging and a popular problem in computer vision. An application of SfM procedures on a Humanoid will him to obtain information about the environment. Hence the goal of this project and this thesis is camera calibration and structure from motion using humanoid NAO(see Figure 1.1)

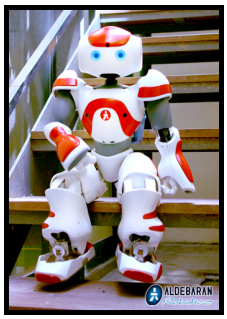


Figure 1.1: The Humanoid NAO developed by Aldebaran Robotics

1.2 Objectives

Camera calibration is one of the major objectives in computer vision because a lot of computer vision techniques are related to camera calibration. Some of these techniques might be structure from motion, stereo vision, robot localization and navigation and most of the existing solutions for these techniques assumes that the camera calibration parameters are already available. Thus there are two objective of this work;

- The primary objective of this work is to develop a module for NAO to obtain the camera parameters.

- The second objective is the application of sparse 3D reconstruction using structure from motion procedures.

1.3 Thesis Structure

- *Chapter 2* explains the camera calibration and discusses the existing calibration techniques and finally explains the method we have used for this project.
- *Chapter 3* explains the structure from motion procedure that we have employed to build a sparse 3D reconstruction of the environment.
- *Chapter 4* explains the various tests we have done for both calibration and structure from motion and shows the results we have obtained.
- *Chapter 5* includes the conclusion of this thesis and discusses the future improvements that can be achieved.
- *Appendix A* is the detailed information about the technical details of the humanoid NAO.
- *Appendix B* aims to give a background information for some of the procedures that we have explained mainly in chapter 2 and chapter 3.

Chapter 2

Camera Calibration

This chapter first gives the general definition of camera calibration problem, then discusses the different calibration techniques briefly and finally explains the calibration method we have used for calibration NAO's camera in detail.

2.1 Introduction

For most of the computer vision applications, camera calibration is the first step because camera parameters are required to measure the metric structure of a scene precisely. Camera calibration is the process of obtaining camera intrinsic and extrinsic parameters. Camera parameters are necessary to obtain knowledge about the euclidean structure of the 3D world from 2D images. Thus to be able to obtain metric information from images, one needs to know the calibration parameters of the camera.

There are different techniques for camera calibration and these techniques might be categorized by different criterion. Instead of discussing the different calibration techniques by grouping them by a specific criterion we will briefly explain some of very well known camera calibration methods. For more extensive and detailed comparisons and evaluations of the existing camera calibration techniques see [26] and chapter 2 of [20].

Finally it's worth mentioning the idea of self-calibration. Self-calibration is the process of obtaining the camera parameters using sequences of images from the camera without any known structure or using any specific apparatus. Providing on-the-fly camera calibration, self-calibration techniques([6], [10]) are useful but still there are robustness issues of existing self calibration methods. Most of the time these issues are related to the geometry of the scene image contents that will be used for recovering the camera calibration parameters. That's why, camera self-calibration is not used in this project.

2.2 An Overview of Camera Calibration Methods

There exists considerable amount of work(to cite a few [26], [28]) to compare the existing camera calibration methods in terms of different criterion. In this section before explaining the details of the calibration method that have been employed, we will briefly explain some of the existing camera calibration methods.

- **Direct Linear Transform(DLT)**
  DLT [12] is a two step calibration procedure and is based on the pinhole camera model. The first step is the solution of linear transformation from object coordinates to image coordinates. The matrix is represented by 3×4 matrix A and the equation can be written as

$$\begin{pmatrix} u_i w_i \\ v_i w_i \\ w_i \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \end{pmatrix} \begin{pmatrix} X_i \\ Y_i \\ Z_i \\ 1 \end{pmatrix}$$

  The parameters $a_{11}, ..., a_{34}$ of the DLT matrix can be solved by a homogeneous matrix equation below

$$Lp_i = 0$$

  By replacing the correct image points with observed values we can estimate the parameters $a_{11}, ..., a_{34}$ in a least squares fashion. Abdel-Aziz and Karara [1] uses the constraint $a_{34} = 1$ in order to avoid a trivial solution $a_{11}, ..., a_{34} = 0$. Then, the equation can be solved with a pseudo inverse technique.

- **R. Y. Tsai's Method**
  A very popular calibration method developed by R. Y. Tsai requires a planar pattern but also requires the exact displacement of the pattern(see Figure 2.1). Thus there is no difference between using a 3D object and a 2D planar pattern with a known displacement.

  Tsai's calibration method [33] assumes that some of the camera parameters are provided by the manufacturer in order to reduce the initial guess of parameters. It is based on a pinhole perspective projection model and the parameters to be estimated are focal length of the camera, radial distortion coefficient, rotation and translation components and co-ordinates of centre of radial lens distortion. The technique uses a 2 step approach. First step includes the computation of all the extrinsic parameters except t_z. This process is

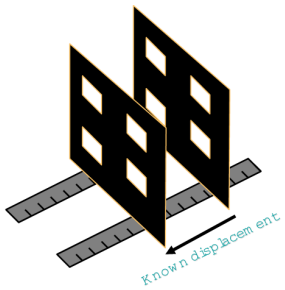


Figure 2.1: The pattern used for Tsai's method(Image courtesy [20])

a solution of a system of linear equations. The input of these equations are the coordinates of the points both in the calibration pattern and world coordinate systems. The second step is to estimate all the missing parameters by non-linear optimization using Levenberg-Marquadt algorithm [21].

- **The method of Faugeras**
  Another technique that uses a 3D object as apparatus for camera calibration is explained by O. Faugeras [5]. The setup is shown in figure 2.2 has two orthogonal planes with A 3D coordinate system with known coordinates and known the corners in this coordinate system.

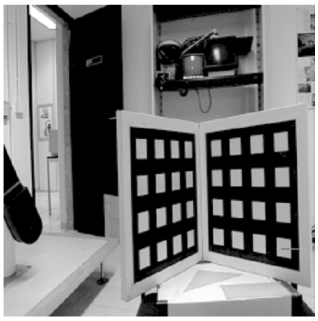


Figure 2.2: 3D object used by Faugeras(Image courtesy [20])

First the corners of the boards(see Figure 2.2) detected. Then using linear least squares method the camera projection matrix(P) is estimated and the intrinsic and extrinsic parameters are obtained from P. Finally the intrinsic and extrinsic parameters are refined using non-linear optimization.

2.3 Zhang's Camera Calibration Method

The method, developed by Z.Zhang [35] is an easy to implement technique that also requires an easy setup; only a planar pattern as an apparatus to be observed from different positions of the camera. The pattern could be anything with the condition of known metric on the plane but traditionally the chessboard pattern is used because it is easy to extract information from the chessboard pattern.

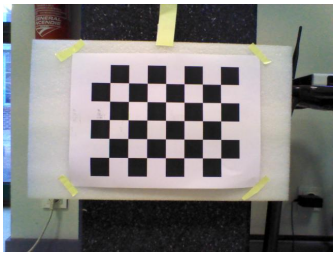


Figure 2.3: Chessboard pattern used for calibration

2.3.1 Notation

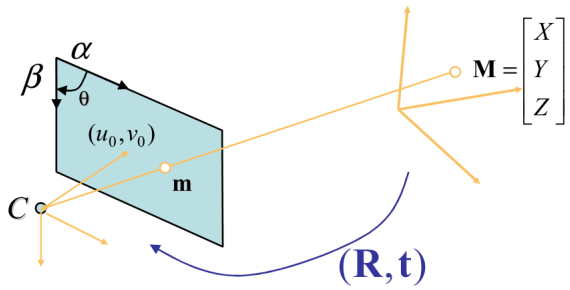


Figure 2.4: Pinhole Camera Model(Image Courtesy [20])

2D points are denoted by lower letters and 3D points are denoted by capital letters. The homogeneous coordinates of a 2D point is denoted by $\tilde{x}$ and a 3D point is denoted by $\tilde{X}$. Thus $m = [u, v]^T$ denotes a 2D point and $\tilde{m} = [u, v, 1]^T$ denotes the same 2D point with homogeneous coordinates. The notation is same for 3D points as well. A camera is modelled by the usual pinhole model(see Figure 2.4). Notice that 3D point(M), image point(m) and the optical centre(C) are collinear. For this reason, mathematically, there is no difference between

placing the image plane, to the front or the back of the optical centre. The relationship between a 3D point M and corresponding image projection m is given by

$$s\tilde{m} = A \begin{pmatrix} R & t \end{pmatrix} \tilde{M} \tag{2.1}$$

$$A = \begin{pmatrix} \alpha & \gamma & u_0 \\ 0 & \beta & v_0 \\ 0 & 0 & 1 \end{pmatrix}$$

where (R, t) of equation 2.1 is the extrinsic parameters of the camera and s is an arbitrary scale factor. A is the camera intrinsic parameters matrix. α and β of A are the scale factors in u and v direction respectively and u_0 and v_0 are the coordinates of the principal point. Note that $(A^{-1})^T$ or $(A^T)^{-1}$ is denoted by A^{-T}.

2.3.2 Methodology

Firstly, it is assumed that the model plane is placed on $Z = 0$ of the world coordinate system. Hence, when we denote the columns of rotation matrix R separately from equation 2.1 we get

$$s \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = A \begin{pmatrix} r_1 & r_2 & r_3 & t \end{pmatrix} \begin{pmatrix} X \\ Y \\ 0 \\ 1 \end{pmatrix}$$

$$= \underbrace{A \begin{pmatrix} r_1 & r_2 & t \end{pmatrix}}_{\text{H}} \begin{pmatrix} X \\ Y \\ 1 \end{pmatrix}$$

From above equations the homography H we obtain is

$$H = A \begin{pmatrix} r_1 & r_2 & t \end{pmatrix} \text{ thus } s\tilde{m} = H\tilde{M} \tag{2.2}$$

An homography can be estimated from the image of the model plane. The method utilizes the technique for estimation of homography based on maximum likelihood criterion. Thus from 2.2

$$\begin{pmatrix} h_1 & h_2 & h_3 \end{pmatrix} = \lambda A \begin{pmatrix} r_1 & r_2 & t \end{pmatrix}$$

where λ is an arbitrary scalar because H is defined up to a scale factor. Then by using the orthogonality property of r_1 and r_2, which indicates $r_1 r_2 = r_1^T r_2 = r_2^T r_1 = 0$, we can define

two constraints on the intrinsic parameters. The first one is

$$\begin{pmatrix} h_1 & h_2 & h_3 \end{pmatrix} = \lambda A \begin{pmatrix} r_1 & r_2 & t \end{pmatrix}$$

$$A^{-1} \begin{pmatrix} h_1 & h_2 & h_3 \end{pmatrix} = \lambda \begin{pmatrix} r_1 & r_2 & t \end{pmatrix}$$

$$\begin{pmatrix} A^{-1}h_1 & A^{-1}h_2 & A^{-1}h_3 \end{pmatrix} = \lambda A \begin{pmatrix} r_1 & r_2 & t \end{pmatrix}$$

Using the orthogonality property $r_1^T r_2 = 0$ we get

$$(A^{-1}h_1)^T (A^{-1}h_2) = 0$$

$$h_1^T A^{-T} A^{-1} h_2 = 0 \tag{2.3}$$

The second constraint is derived from the orthonormal property of r_1 and r_2 which indicates that $r_1 r_1 = 1$, $r_2 r_2 = 1$, $r_1^T r_1 = 1$ and $r_2^T r_2 = 1$. Hence again from 2.2 we have

$$\begin{pmatrix} A^{-1}h_1 & A^{-1}h_2 & A^{-1}h_3 \end{pmatrix} = \lambda A \begin{pmatrix} r_1 & r_2 & t \end{pmatrix}$$

$$\begin{aligned} &\text{Using } r_1^T r_1 = 1 \text{ we get } (A^{-1}h_1)^T (A^{-1}h_1) = 1 \\ &\text{Using } r_2^T r_2 = 1 \text{ we get } (A^{-1}h_2)^T (A^{-1}h_2) = 1 \end{aligned} \tag{2.4}$$

Using 2.4 we obtain the second constraint on the intrinsic parameters

$$h_1^T A^{-T} A^{-1} h_1 = h_2^T A^{-T} A^{-1} h_2 \tag{2.5}$$

Notice that for both constraints that $A^{-T}A^{-1}$ is the image of absolute conic(see Appendix B.5 for details on absolute conic and image of absolute conic)

2.3.3 Relation Between Intrinsic Constraints and Absolute Conic

Now we will explain the relationship between the two intrinsic constraints(2.3 and 2.5) and the absolute conic. The model plane in the camera coordinate system is defined by

$$\begin{pmatrix} r_3 \\ r_3^T t \end{pmatrix}^T \begin{pmatrix} x \\ y \\ z \\ w \end{pmatrix} = 0$$

For the points at infinity $w = 0$ and this plane intersects with the plane at infinity at a line. Hence we can define $\begin{pmatrix} r_1 \\ 0 \end{pmatrix}$ and $\begin{pmatrix} r_2 \\ 0 \end{pmatrix}$ are two points on that line. Therefore any point on that line can be written as a linear combination of these two points.

$$\begin{aligned} x_\infty &= a \begin{pmatrix} r_1 \\ 0 \end{pmatrix} + \begin{pmatrix} r_2 \\ 0 \end{pmatrix} \\ &= a \begin{pmatrix} ar_1 + br_2 \\ 0 \end{pmatrix} \end{aligned}$$

Now if we compute the intersection of the above line with the absolute conic(see Appendix B for details on absolute conic and the image of absolute conic) noting that by definition the point x_∞ is a circular point [29] x_∞ satisfies $(x_\infty)^T x_\infty = 0$ we have the two intersection points

$$x_\infty = a \begin{bmatrix} r_1 \pm i r_2 \\ 0 \end{bmatrix}$$

Then the projection of these two points, $\tilde{m}_\infty = A(r_1 \pm i r_2) = h_1 \pm i h_2$, on the image of absolute conic is given by $A^{-T}A^{-1}$, is

$$(h_1 \pm i h_2)^T A^{-T} A^{-1} (h_1 \pm i h_2) = 0$$

which gives us the two constraints 2.3 and 2.5

2.4 Solving Camera Calibration

This section provides the explanation of the closed-form solution of the camera calibration problem [35]. The analytical solution and the non-linear optimization technique will be explained.

The constraints of the intrinsic parameters include the image of absolute conic($A^{-T}A^{-1}$). Thus let

$$B = A^{-T}A^{-1} \equiv \begin{bmatrix} B_{11} & B_{12} & B_{13} \\ B_{12} & B_{22} & B_{23} \\ B_{13} & B_{23} & B_{33} \end{bmatrix} \tag{2.6}$$

Now notice that B is a 3×3 symmetric matrix with 6 independent entries. Then B can be defined as a 6D vector with the upper triangular elements of B

$$b = \begin{bmatrix} B_{11} & B_{12} & B_{22} & B_{12} & B_{23} & B_{33} \end{bmatrix}^T \tag{2.7}$$

Then we have

$$h_i^T B h_j = v_{ij}^T b \tag{2.8}$$

with

$$v_{ij} = \begin{bmatrix} h_{i1}h_{j1} \\ h_{i1}h_{j2} + h_{i2}h_{j1} \\ h_{i2}h_{j2} \\ h_{i3}h_{j1} + h_{i1}h_{j3} \\ h_{i3}h_{j2} + h_{i2}h_{j3} \\ h_{i3}h_{j3} \end{bmatrix}^T$$

where h_i denotes the i^{th} column vector of H. Now we can when we rewrite the two constraints 2.3 and 2.5 as

$$\begin{bmatrix} v_{12}^T \\ (v_{11} - v_{22})^T \end{bmatrix} b = 0 \tag{2.9}$$

Thus we have 2 equations for each image of the model plane. When we have n images we will stack all the equations and will obtain $2n \times 6$ matrix V as shown in equation 2.10.

$$Vb = 0 \tag{2.10}$$

V is a $2n \times 6$ matrix. Applying SVD(see appendix B) we get the solution to 2.10. Knowing b, and $B = \lambda A^{-T}A$ we can obtain all the intrinsic parameters. Finally the extrinsic parameters for each image can be computed from A. From 2.2

$$r_1 = \lambda A^{-1} h_1$$
$$r_2 = \lambda A^{-1} h_2$$
$$r_3 = r_1 \times r_2$$
$$t = \lambda A^{-1} h_3$$

where $\lambda = 1/||A^{-1}h_1|| = 1/||A^{-1}h_1||$

2.4.1 Summary of the Method

The method can be summarized as follows [35];

1. Take a few images(at least 2) of the chessboard pattern from different orientations.
2. Detect the feature points(chessboard corners) for each image.
3. Estimate the five intrinsic parameters and all the extrinsic parameters using the closed-form solution as described above.
4. Estimate the coefficients of the radial distortion by solving the linear least-squares.
5. Refine all parameters by minimizing [16].

2.5 Camera Calibration Procedure for NAO

This part explains a two-phased application method for the camera calibration to NAO. The first phase is the acquisition of chessboard pattern images from different positions of the camera and the second phase is calibration using the images acquired in the first phase.

In the first phase there are three main considerations and these are corner detection for correct image acquisition, motion and image acquisition. The procedure in this phase is repeated for a given number of times in order to obtain required number of images.

- **Chessboard Corner Detection**
  we have to be sure that the planar chessboard pattern is in front of the NAO's camera perspective and the corners of the pattern are extractable. Both of the goals are achieved by employing a corner detection method since when there is no pattern in front of the camera, there will not be any corners to detect. If we do not make sure that the corners are extractable then we may acquire irrelevant sequence of images that has no value for the calibration process. The corner detection method is the findChessBoardCorners method of OpenCV(see Appendix B.6 for details of the algorithm).

- **Motion**
  We need to move NAO in order to obtain an image from different positions. Two kinds of motion is programmed. The first motion is only body motion. Without moving NAO by walking, we will change the position of the camera. This motion is based on the knees and the pelvis.(see figureREFREFREFREF). The second motion is walking in a circular fashion(This motion is illustrated in Figure 2.5). The motion of NAO is programmed using the built-in functions of NAO and any motion that contains rotation and translation at the same time is valid.

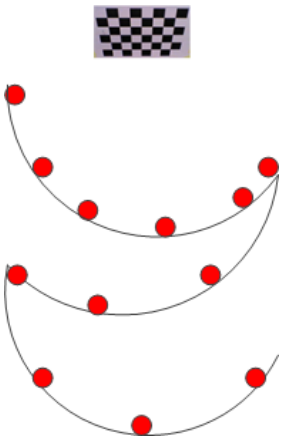


Figure 2.5: The illustration of motion of NAO in order to take different positions for image acquisition. The red points represent NAO.

- **Image Acquisition**
  After a particular motion, if the corners of the chessboard pattern is extractable, we will acquire the image and save it. OpenCV and Python Image Library is used for image acquisition and saving.

In the second phase of the calibration procedure we only need to read images that are saved, and employ the calibration procedure. Note that in order to obtain the pattern images there are two different options. One may either fix the pattern and move the camera or fix the camera and move the pattern. We have applied and tested both procedures but the goal of our calibration process in terms of motion is the former one. The technical details of the implementation of the method is explained in the following chapter.

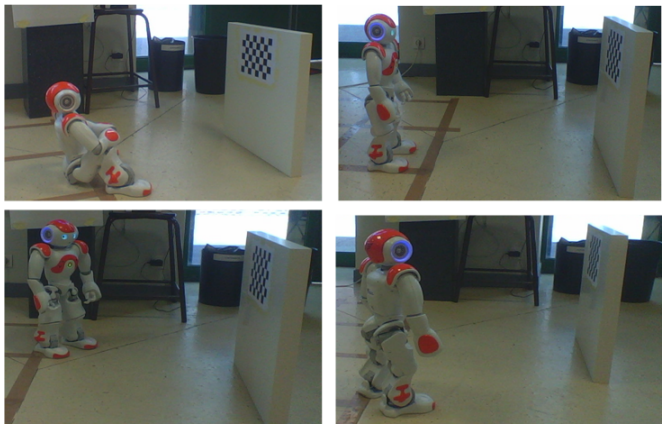


Figure 2.6: 4 different Poses of NAO in order to get images

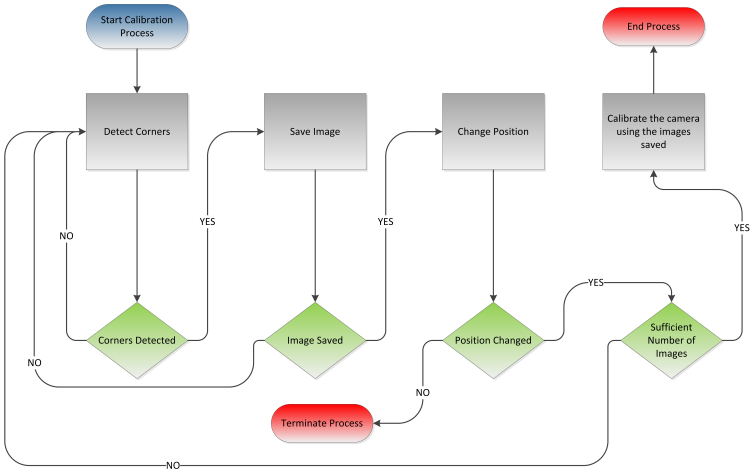


Figure 2.7: The calibration procedure

Chapter 3

Structure from Motion

In this chapter, first a general method is explained for 3D reconstruction and a classification of Structure from Motion(SfM) methods are discussed. Finally the adaptation of a particular SfM method to humanoid NAO is explained in detail.

3.1 Introduction

Structure from Motion is the method of reconstructing 3D points and recovering the camera motion from 2D images that are taken from different position of the same scene. The reconstruction of rigid objects and motion estimation of the camera, from a sequence of images is a challenging and a popular problem in computer vision. A definition of general methods such as triangulation, epipolar geometry estimation, etc., has to be addressed in order to explain the SfM. Since the general definition of 3D reconstruction proposes solutions to such problems, first we explained the general method of 3D reconstruction. Then a general classification of SfM methods is discussed briefly. Finally, the solution to the problem of motion estimation is explained in detail and affiliated with the methods explained in 3D reconstruction problem and the application of the SfM method to humanoid NAO is explained.

3.2 General Method of 3D Reconstruction

A 3-dimensional scene point can be reconstructed using image pairs. There are parameters that define type of reconstruction and these are; *the fundamental matrix* and *camera matrix*. Since this is a general method, it is assumed that there is no knowledge about the cameras or the scene and the method is explained.

Given an image pair of the same scene there exists correspondences $x_i \longleftrightarrow x_i^{'}$ where x_i and $x_i^{'}$ are the corresponding points in the images. Then for all i we have the following relationship between the corresponding points and the actual, unknown, 3D points X_i

$$x_i = PX_i \text{ and } x_i' = P'X_i \text{ for all i}$$

The method of 3-dimensional reconstruction can be summarized as;

1. *Compute the fundamental matrix from point correspondences*
   The fundamental matrix F satisfies $x_i'Fx_i = 0$ when the correspondences are known, and each point correspondence one linear equation for F, thus by knowing at least 8 corresponding point it is possible to solve the equation linearly for F. The equation for n pair of points $(x_n, y_n, 1)$ and $(x_n', y_n', 1)$ is

$$Af = \begin{bmatrix} x_1'x_1 & x_1'y_1 & x_1' & y_1'x_1 & y_1'y_1 & y_1' & x_1 & y_1 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_n'x_n & x_n'y_n & x_n' & y_1'x_n & y_n'y_n & y_n' & x_n & y_n & 1 \end{bmatrix} f = 0$$

   Generally we will have more than 8 points thus, the least square solution is the singular vector corresponding the smallest singular value of A.

2. *Compute the camera matrices from fundamental matrix*
   The camera will be chosen as $P = \left[I|0\right]$ and $P' = \left[[e']_\times F|e'\right]$

3. *Compute the 3D point X_i which corresponds to the relation $x_i \longleftrightarrow x_i^{'}$*
   The process is also known as *triangulation* and the solution of triangulation is; since $x = PX$ and $x' = P'X$, one should combine these two equations into a form of $AX = 0$. Then solve $AX = 0$ using SVD and picking the singular vector corresponding to the smallest singular value. Algebraically;

$$x = PX \longleftrightarrow x \times (PX) = 0$$

   where

$$x = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \text{ and } PX = \begin{bmatrix} P_1^T X \\ P_2^T X \\ P_3^T X \end{bmatrix}$$

$$y(P_3^T X) - (P_2^T X) = 0$$
$$x(P_3^T X) - (P_1^T X) = 0$$
$$x(P_2^T X) - y(P_1^T X) = 0$$

Notice that the third equation is a linear combination of the first two. Then the homogeneous system $AX = 0$ is

$$x = \begin{bmatrix} xP_3^T & -P_1^T \\ yP_3^T & -P_2^T \\ x^{'}P_3^T & -P_1^{'T} \\ y^{'}P_3^T & -P_2^{'T} \end{bmatrix} X = 0$$

Using SVD the equation above is solved thus the 3D points are recovered.

3.3 Review of Structure from Motion Methods

In this section a classification framework for SfM algorithms is discussed. Various criterions can be defined in order to classify the SfM methods. The most general classification of SfM methods is the *calibrated SfM* where the camera matrix of the camera is known and *uncalibrated SfM*. However, both calibrated and uncalibrated SfM shows similarity in terms of the techniques that they use. Thus in addition to the calibrated and uncalibrated SfM the following techniques are used for classification as well.

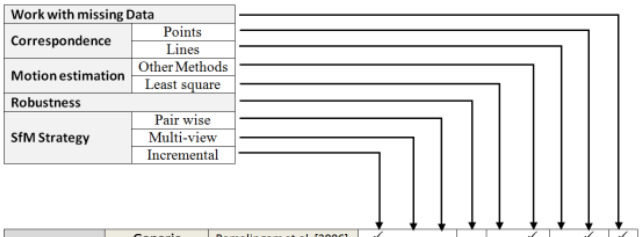
1. **Working with Missing Data**
   The techniques may be classified by their response to missing information. By missing information we refer to the problem of missing feature points in one of the views. Some methods are specifically developed to be able to handle the missing data, others are sensitive to the presence of noise and they fail.

2. **Feature Extraction and Correspondence**
   The features might be points(a general case), lines or edges. A classification case is the method that SfM algorithms follow in order to extract and match features. The most popular methods for feature extraction are Harris operator [8], SIFT [18] and SURF [2].

3. **Motion Estimation**
   A classification criterion might be about how the method computes the motion. The most popular computation of motion estimation is those which uses the linear least squares optimizations.

4. **Robustness**
   The response of the algorithm in the presence of noise is defined to be another criterion.

5. **SfM Strategy**
   The strategy is defined in terms data utilization and three different approaches exist, namely; Multi-view, Pair-wise and Incremental. By data utilization we mean whether, data from all views are used in the calculations at the same time (Multi-view), or only the data from pairs of images are considered (Pair-wise). The last type is using an incremental approach where it starts with a pair of images only, then incorporates the data from a new image(Incremental).



Calibrated SfM	Generic SfM	Ramalingam et al. [2006]	✓			✓	✓	✓
		Mouragnon et al. [2009]	✓		✓	✓	✓	✓
	Object space cost function	Mitachell et al. [2010]		✓	✓	✓	✓	
		Schweighofer & Pinz [2006]		✓	✓	✓	✓	
	Factorization	Tomasi a & Kanade [1998]		✓		✓	✓	
	SOCP + BA	Martinec & Pajdla [2006]		✓	✓	✓	✓	✓
		Martinec & Pajdla [2007]		✓	✓	✓	✓	✓
	EKF	Broida et al [1990]		✓	✓	✓	✓	
		Qian et al [2001]		✓	✓	✓	✓	
	Line-based	Taylor & Kriegman [1995]		✓		✓	✓	
		Weng and Huang [1992]		✓		✓	✓	
Un-Calibrated SfM	Bilinear	Ramachandran et al. [2011]		✓	✓	✓	✓	
	Factorization	Sturm and Triggs [1996]		✓		✓	✓	
		Oliensis and Hartly [2007]		✓	✓	✓	✓	
	DMS + BA	Sanvely et al.[2008]	✓		✓	✓	✓	
		Brown & Lowe [2005]	✓		✓	✓	✓	
	EKF	Azarbayejani & Pentland [1995]		✓	✓	✓	✓	
	Line-based	Bartoli & Sturm [2003]		✓		✓	✓	
	Explicit reconstruction	Rothwell et al.[1995]		✓	✓		✓	✓

Figure 3.1: The Classification of SfM Methods [14]

3.3.1 Calibrated SfM

Calibrated SfM results with euclidean structure of the scene, thus using calibrated SfM it is possible to obtain real measurements such as lengths and angles of the real scene. There are different approaches as shown in Figure 3.1. The most popular calibrated SfM methods are the factorization methods([4], [13], [32] to cite a few). The common point of all factorization methods is that they all allow to linearize the camera observation model and their computational cost is low, thus they provide fast results. Another method is called generic SfM. The main property of the generic SfM is that those techniques can be applied to all kinds of cameras(pinhole cameras, catadioptric, omnidirectional etc.). A generic camera calibration and SfM technique is defined by Ramalingam *et al.* [25]. The problem of the method is the failure of the correspondence algorithm(SIFT). Another generic SfM method is defined by Mouragnon *et al* [22]. Their method is a generic real-time SfM based on incremental 3D reconstruction.

3.3.2 Uncalibrated SfM

The uncalibrated SfM methods assume that the camera parameters are not known, thus using those methods one can only obtain projective reconstructions. It is impossible to obtain real measurements such as lengths and angles of the real scene. In order to obtain euclidean reconstruction, the camera parameters should be given. The most popular uncalibrated SfM is the factorization based methods, as it is for the calibrated SfM as well. A very popular method for factorization based uncalibrated SfM is proposed by Sturm and Triggs [31]. Note that for this method and initial guess of projective depth is necessary. In [19], in order to improve the results of [31] some iterative methods have proposed. Oliensis and Hartley [24] show that [31], [19] are unstable because of the convergence problem and in [24] they proposed a new iterative extension for [31] and [19]. There are more techniques based on bilinear programming, Direct metric structure and bundle adjustment, etc., and an extensive survey is done by [14] explaining each category of algorithm in more details.

3.4 Methodology

In the previous sections, 3D reconstruction problem is discussed and the SfM methods classified using certain criterions. Now, we will complete our discussion by explaining the details of our approach applied to NAO. Our method is a calibrated SfM method because the camera parameter are found by the method the method that we have explained in the previous chapter.

3.4.1 The Correspondence Problem

The correspondence problem is the problem of finding the corresponding points of two images that are acquired by a moving camera. After years of research on correspondence problem, there are a lot of different methods are developed and the most popular methods are those based on Harris'operator [8], SIFT[1] [18] and SURF[2] [2]

The method that we choose to find corresponding points in a sequence of images, is based on feature detection and feature descriptor creation. It is called speeded up robust features(SURF) [2].

Assume we have only two images and we want to find the corresponding points for these images. SURF first detects potential feature points in the first image and defines descriptors for each detected feature point. Then the same procedure is applied to the second image. The image descriptors are defined by dividing a local image into a grid(typically 4x4) and a orientation histogram is computed for each of these cells. Then, knowing each feature point's descriptor for both images, the best matches are found. Finally RANSAC [7] is used to filter positive false matches.

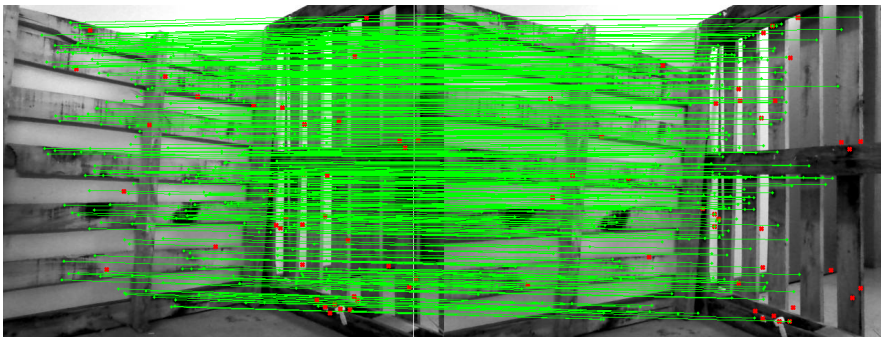


Figure 3.2: Feature point detection and matching using SURF

Note that the computational cost of SURF descriptor is low. Since the methods are applied to a humanoid(mobile robot) SURF descriptors are chosen because on a humanoid, in motion, the time of the computation is crucial.

3.4.2 Essential Matrix Estimation

The fundamental matrix(F) is a 3×3 the algebraic representation of the epipolar geometry and the notion of F is defined for any corresponding points $x \longleftrightarrow x'$ in two images as

[1]Scale-invariant feature transform
[2]Speeded up robust features

$$x'^T F x = 0 \tag{3.1}$$

The essential matrix constraint(a specialization of fundamental matrix constraint) we have

$$\hat{x}_i'^T E \hat{x}_i = 0 \tag{3.2}$$

and the essential matrix equation can be written as

$$E \equiv [t]_\times R \tag{3.3}$$

Normally F has 8 degree of freedom(DoF) but in the special case of essential matrix where the points are defined in the camera coordinate system by applying the K^{-1}(the inverse of camera intrinsic parameters matrix) to the corresponding points x_i and x_i', the DoF of the essential matrix becomes 5, 3 DoF for translation and 3 DoF for rotation matrices(see equation 3.3), since the essential matrix is defined up to a scale. This means that given at least 5 corresponding points in two image, one can estimate the essential matrix. The algorithm, for estimating the essential matrix, by using at least 5 points is called the five-point algorithm [23]. However, we do not employ five-point algorithm proposes a non-linear solution to the essential matrix estimation problem. A non-linear solution is not suitable for our purposes since computation cost of the non-linear estimation methods very high. The method is called by humanoid robot in motion, thus we need a faster and computationally inexpensive solution. Moreover, the non-linear estimation methods are sophisticated methods that are highly complex to implement.

Due to the reasons that we have explained the normalized eight-point algorithm [17], [11](see Appendix B.3 for details) is chosen to estimate the essential matrix. The normalized eight-point algorithm has advantages over five-point algorithm and it suits our purposes for the reasons such as; it solves the essential matrix estimation problem linearly consequently it is fast and computationally inexpensive and it is easy to implement.

3.4.3 Camera Extraction from Essential Matrix

The essential matrix is used to recover rotation(R) and translation(t) vectors.

> ***Theorem*** *Let the singular value decomposition of the essential matrix be* $E \sim Udiag(1,1,0)V^T$*, where* U *and* V *are chosen such that* $det(U) > 0$ *and* $det(V) > 0$*. Then* $t \sim t_u \equiv \begin{bmatrix} u_{13} & u_{23} & u_{33} \end{bmatrix}^T$ *and* R *is equal to* $R_a \equiv UDV^T$ *or* $R_b \equiv UD^TV^T$

As explained in the theorem by David Nipster [23], any combination of the R and t satisfies the epipolar constraint. Thus there is an ambiguity and to find the true combination the first camera matrix is assumed to be $[I|0]$ and t is unit length. There are four possible solutions except for

overall scale, and these are; $P_1 \equiv \begin{bmatrix} R_a & | & t_u \end{bmatrix}$, $P_2 \equiv \begin{bmatrix} R_a & | & -t_u \end{bmatrix}$, $P_3 \equiv \begin{bmatrix} R_b & | & t_u \end{bmatrix}$ and $P_4 \equiv \begin{bmatrix} R_b & | & -t_u \end{bmatrix}$.

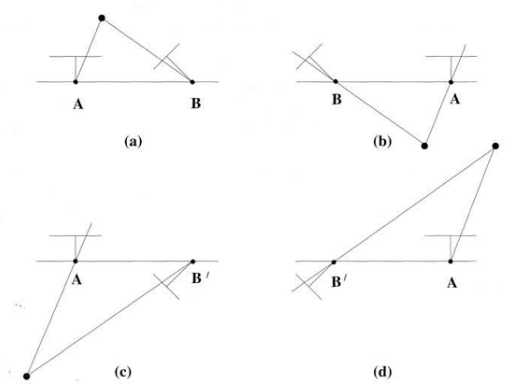


Figure 3.3: Four possible configurations are illustrated. The true configuration is (a) where the 3D point is in front of both cameras.(Image courtesy [9])

The true configuration is decided using the cheirality constraint[3]. Note that one point that satisfies the cheirality constraint, is enough to decide which configuration is the right one. In order to test the cheirality the point is triangulated. $([I|0], P_1)$ view pair is used first to find the 3D point Q. The points for the two views are indicated as Q_1 and Q_2. If $Q_1 Q_2 > 0$ and $(P_i Q_1) Q_2 > 0$ where i is one of the possible solutions, then the cheirality constraint is satisfied and the positions of the cameras are found. Finally, simple triangulation by back-projecting rays from the image points will fail because of the noise in the image points and the back-projected rays will not intersect. Thus triangulation requires definition and solution of a cost function. There are different solutions of triangulation [9]. We have employed a linear triangulation method which is a direct analogue from DLT method [9]. Since we have the relations $x = PX$ and $x' = PX'$ for the first and second image points these equations can be combined as $AX = 0$ which is an equation linear in X(see details of triangulation in the section 3.2).

3.5 Structure from Motion using NAO

The method explained in the previous section is applied to the images acquired by NAO in order to estimate the motion and reconstruct the 3D scene structure. The input for the method is the calibration matrix of the camera and the two sequential image of the scene. The system

[3]The scene points should be in front of the camera

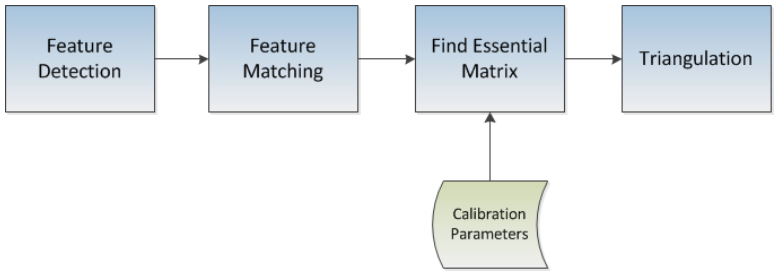


Figure 3.4: The system of SfM using NAO. The first part (feature detection and matching) is a *search problem* and the second part (essential matrix estimation and triangulation) is an *estimation problem*.

works for two sequential images of the environment. The output is the motion of the camera and a 3-dimensional point cloud of the scene. Any pair of images can be the input of this process and the output will be the projection matrix for the second pose of the camera and a 3D model of the scene.

Chapter 4

Implementation and Results

This chapter first explains the implementation details, then the test results for both camera calibration and structure from motion.

4.1 Implementation

The calibration method is implemented using python [34] scripting language. Python was chosen as it is supported by Naoqi framework. In addition to the built-in functions that Aldebaran Robotics provided, the python image library and opencv libraries have used for the calibration module. Only a prototype of the SfM procedures are implemented using Matlab(some of the functions are from Peter Kovesi [15]) and the 3D point clouds are visualized using Meshlab sofware. This part of the work needs to be ported to Aldebaran software development and robot manipulation IDE choregraphe(see Appendix A). The problems regarding to the implementation of this work was mainly related to SfM part because of the necessity of the usage of different libraries. The problems are raised when we wanted to embed the prototype of SfM module, as we did for calibration module, inside choregraphe because of some incomparability issues of array manipulation libraries such as numpy [34]. However, the SfM module can still be used with Matlab using the Matlab SDK of Aldebaran robotics.

One advantage of having a procedure as a module in choregraphe is that, it is an encapsulated module that provides abstraction for any user. Therefore, by using the choregraphe boxes any user can obtain results even without knowledge of methods.

4.2 Camera Calibration Results

Camera calibration test results are verified using widely used and recognized camera calibration toolbox [3] by the computer vision society, written by Jean-Yves Bouguet. First we have tested the camera calibration module fixing NAO and moving pattern in different rotation and translations to be sure that; camera calibration module is working and the programmed movements are not a limitation for the camera calibration procedure(the movement of NAO might be limited compared to the former rotations and translations). Therefore, we have to apply two different tests. The first one verifies that the implementation of the method has no errors. The second one verifies the calibration can also be done moving NAO. The images are shown in figure 4.1 are for the first test.

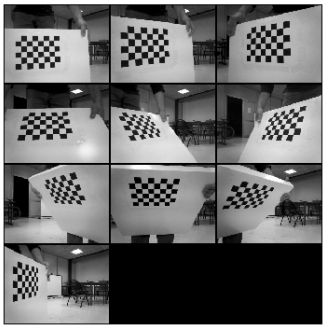


Figure 4.1: The images of the chessboard pattern in different rotations and translations

The results for intrinsic parameters of the camera by calibration toolbox is

$$K = \begin{bmatrix} 556.686109305625340 & 0.0 & 328.683955331234590 \\ & 557.590558351206710 & 231.765667870181740 \\ 0.0 & 0.0 & 1.0 \end{bmatrix}$$

The results for intrinsic parameters of the camera by our method is

$$K = \begin{bmatrix} 557.473449707 & 0.0 & 329.959350586 \\ 0.0 & 558.239074707 & 232.031509399 \\ 0.0 & 0.0 & 1.0 \end{bmatrix}$$

The results show the intrinsic parameters of the camera with both method is the same. Since the test condition to be verified for the first test is to obtain the same results with the calibration toolbox, we can verify that the method that we have implemented works.

The second test set is the images captured by fixing the calibration pattern and moving

NAO. The image set is shown and the movements are illustrated in figure 4.2. The results of our method for intrinsic parameters is

$$K = \begin{bmatrix} 574.608947754 & 0.0 & 320.613189697 \\ 0.0 & 575.934082031 & 230.169967651 \\ 0.0 & 0.0 & 1.0 \end{bmatrix}$$

As the results show that the calibration method works and the recovered intrinsic parameters are correct with a very slight change which can be ignored. Note that the more image of patterns we get the more accurate results we obtain.

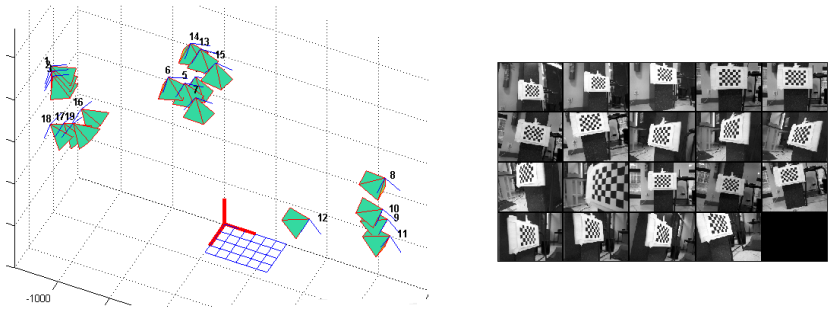


Figure 4.2: The motion of NAO(left) and the calibration images(right)

4.3 Structure from Motion Results

Structure from Motion module is tested for 3 different set of scene structures. The first test images are acquired from outdoor scenes(4.3). The first one is a pre-prepared environment with a lot of feature points and specific and solid structure(4.5). The third test set is the random sequence of images that have no pre-prepared structure.(4.7). In order to obtain a good 3D reconstruction we needed as much correspondence points between two sequential images as we can extract and match, thus the number of feature points detection and matching was crucial.

Figure 4.3: Two views of an outdoor scene. The green dots are representing the matched feature points.

Figure 4.4: The 3D point cloud of the library(4.3) from three different point of view.

Figure 4.5: An pre-prepared indoor environment with a lot of feature points.

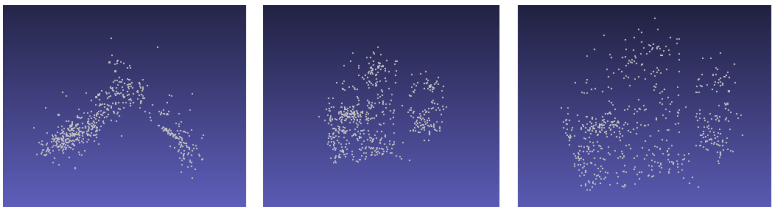


Figure 4.6: The 3D reconstruction of the scene shown in Figure 4.5.

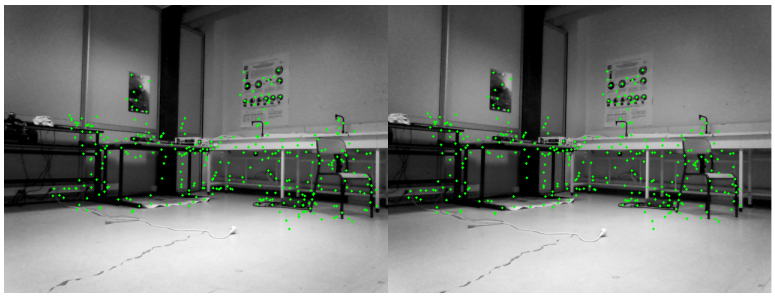


Figure 4.7: A random scene.

Figure 4.8: The 3D reconstruction of the scene in Figure 4.7.

The results of 3D reconstructions are shown above. Generally we had good reconstructions except when NAO was moving in an environment where there was less number of future points. Scenes like those are not easy to reconstruct and the method we have applied is not well responded to such environments. Since, there is no ground-truth of such applications we can visually inspect the results from the created point clouds as shown in above figures. The results of SfM also verifies that the calibration module results are correct. Obviously, with wrong camera parameters the results of 3D reconstructions would not be true.

Chapter 5

Conclusions and Future Work

The goal of this thesis was to develop methods that would allow humanoid robot NAO to calibrate it's camera and obtain a 3D reconstruction of the scene from two views and recover the pose of the camera. Also in terms of reusability we aimed to design the system for both calibration and Sfm in a way that it provides high level of abstraction.

In the first part of this thesis, camera calibration problem is explained and a brief review on the existing methods for camera calibration is provided. Then we gave the details of the camera calibration algorithm we have chosen to apply on NAO. Since the camera calibration was the first step of this work it was crucial to obtain accurate results for further tasks and development. The results and various tests that we have showed that the application for camera calibration of NAO is successful. Also it is provided as an encapsulated module in order to provide the abstraction of details as much as possible. In the second part, the concept of Structure from Motion procedures are discussed and a classification of SfM algorithms is provided. Then the SfM method is explained and we discussed why this particular method is chosen for this application. Finally the tests and the results for both procedures are shown, explained and verified.

In terms of reusability and modularity of the applications, for both considerations of future developments and ease of use, we have developed the calibration module as a stand alone package that any user, even without the knowledge of the details of the method, can employ this module and obtain the calibration results for all the works that require camera characteristics. The SfM module is developed as an external script instead of as a module of Nao.

In this project all the goals that are defined in the problem statement are achieved. The contribution of this work is the application of challenging tasks; camera calibration and structure from motion using a mobile robot(NAO).

There are variety of computer vision algorithms that can be applied on NAO using our work

as basis. Some of those may be listed as future works;

- The first future work of this thesis would be importing the SfM method we have already prototyped and tested as external script.
- A dense 3D reconstruction application may be developed over this work using more than two images as input for the scene and applying different correspondence algorithms.
- An Incremental SfM methods might be employed using our method as a basis in a way that each and every image points are registered to the initial point cloud. Moreover a real-time SfM might be an application to extend this work.
- A stand alone Robot localization and navigation application can be developed using the motion and structure information.

Appendix A

Humanoid NAO

A.1 General Description and Hardware

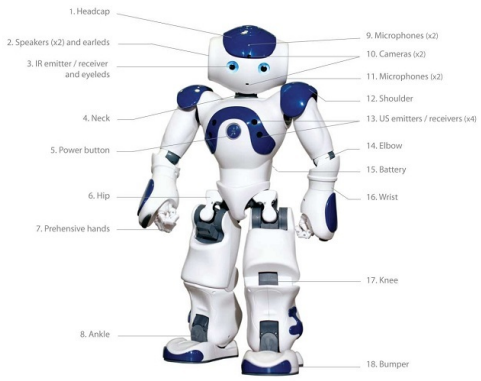


Figure A.1: Nao components(image courtesy [27])

NAO is a humanoid robot produced by Aldebaran Robotics [27](figure A.1), a French start-up company. NAO is 58 cm tall and weights approximately 5 kg. The version of NAO we have used for this research has 27 DoF; specifically 2 for head, 5 for each arm, 1 is the pelvis, 6 in each leg, and 1 in each hand(figure A.2).

The motion of NAO is based on DC Motor's and the autonomy is approximately 20 minutes. There are various sensors placed on NAO. These sensors include 2 CMOS cameras(figure A.3) on the head, 2 ultrasound sensors on the chest 1 bumper sensor on each arm, leg and head. Moreover there are 2 speakers placed on the ears and it has 4 microphones for speech recognition

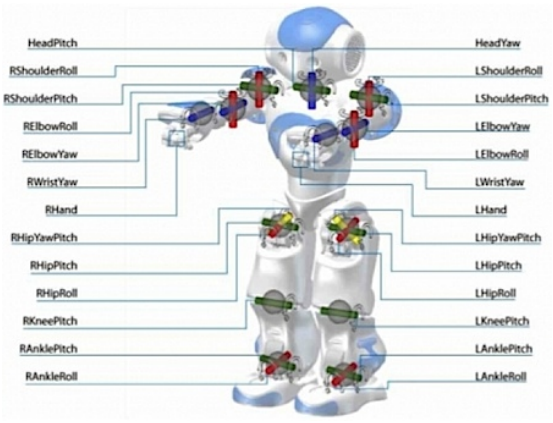


Figure A.2: NAO Joints(image courtesy:http://www.technovelgy.com/graphics/content10/nao-robot-dof.jpg)

purposes.

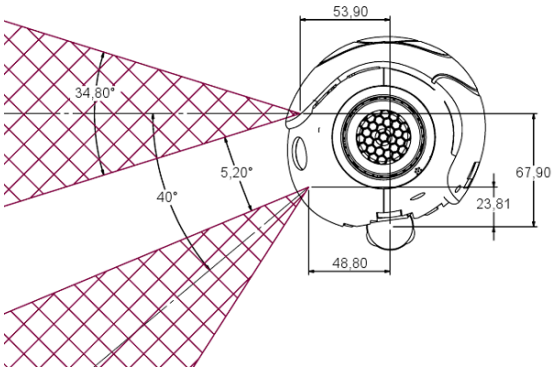


Figure A.3: NAO top and bottom cameras(image courtesy [27])

A.2 Software Architecture

The software architecture of NAO is called NaoQi framework by Aldebaran Robotics. There are three components of this architecture. These are OpenNao, NaoQi Deamon and Device control manager(DCM).

- **OpenNao**: Embedded GNU/Linux distribution based on Gentoo, specifically developed to fit the NAO needs

- **NaoQi Deamon**: The main software that runs on the robot and controls it and provides basic robotic needs like synchronization and parallelism
- **Device Control Manager**: Similar to NaoQi Deamon but DCM calls functions of controllers directly and by this mecanish the abstraction layer provided by NaoQi lost. DCM provides advantages and disadvantages according the purpose of the user. For example if one wants to implement real-time algorithms and requires fast responses, then using DCM is advantageous because using DCM one can get faster responses by controlling directly the hardware architecture. On the other hand, using DCM, one loses the balance constraints that are provided by NaoQi is lost.

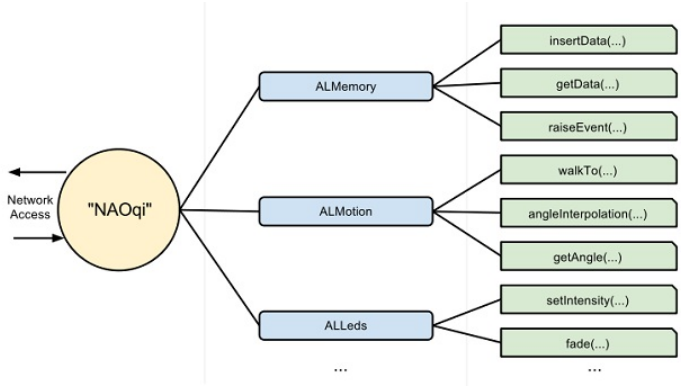


Figure A.4: NaoQi Deamon provides libraries(modules) and those modules provide their own functions

The process is as follows; once one started Nao, the OpenNao OS starts running and initializes the NaoQi Deamon. Here NaoQi Deamon(Also called **Broker**) employs the required and demanded libraries, which include basic functions. Note that NaoQi Deamon prevents us handling with low level hardware operations by providing an abstraction layer.

A.3 Programming NAO

The NAOqi API is the API developed by Aldebaran Robotics for programming NAO and is currently available in at 8 languages. Apart from some minor language-specific differences, the API is mostly the same across all languages, allows to bring knowledge from one language to another. One can only write a NAOqi module in C++ and Python, but have access to the full client API in all languages. Only C++ and Python are supported on the robot, other languages are only supported on computer to remotely access NAO.

A.3.1 Choregraphe and NAOSim

Choregraphe is the software entirely designed and developed by Aldebaran Robotics. It lets users create and edit movements and interactive behaviours simply. The language to program behaviours and functions inside Choregraphe is Python. *NAOSim* is the simulator where a virtual environment can be customized by inserting and modifying objects of various shapes. NAOSim includes physics thus all the behaviours programmed by a user can be tested with a simulated robot.

Appendix B

Background

This chapter aims to give the background information about some of the basic techniques, information and algorithms that have been used throughout this thesis.

B.1 Camera Modelling

This section explains how a 3D point is modelled on a 2D image the using pinhole camera model. We will, geometrically and algebraically, interpret this modelling and following chapters will use this very basic model. Images of this section is taken from [30].
In order to map a 3D world coordinate point, the model uses three different mappings.

1. **First Mapping - 3D world to Camera(Centre of Projection)**

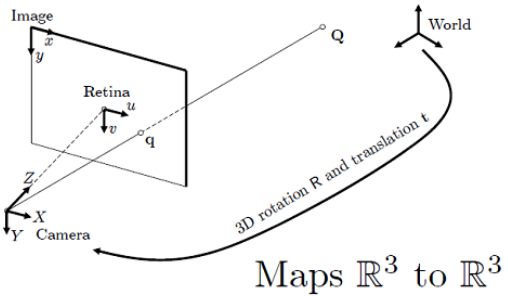


Figure B.1: World point to centre of projection mapping

Displacements are position and orientation. If we want to express this operation in ho-

mogeneous coordinates;

$$Q_c = \begin{pmatrix} R & t \\ 0 & 1 \end{pmatrix} Q \tag{B.1}$$

Where R is the 3x3 rotation matrix and Q is the world homogeneous coordinates $(\mathrm{X,Y,Z,1})^{\mathrm{T}}$. R has two properties and these are;

$$R^T R = I \tag{B.2}$$

$$det(R) = 1 \tag{B.3}$$

if we interpret this 2 equations; the R matrix is orthogonal and multiplication of R transpose and R is equal to I because this transformation preserves the angles. Determinant of R is 1 because we preserve the size as well.

2. **Second Mapping - Camera(Centre of Projection) to retina**

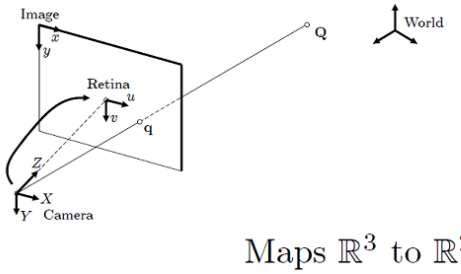


Figure B.2: Camera(Centre of Projection) to retina

The camera centred 3D coordinates Q_c is $(\mathrm{X}_c,\mathrm{Y}_c,\mathrm{Z}_c,1)^{\mathrm{T}}$ and u and v are retina centred 2D point coordinates. We can obtain u and v by;

$$u = fX_c/Z_c \tag{B.4}$$

$$v = fY_c/Z_c \tag{B.5}$$

in homogeneous coordinates;

$$\begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \begin{pmatrix} fX_c \\ fY_c \\ Z_c \end{pmatrix} \tag{B.6}$$

where f is the focal length of the camera. Now if we write it in matrix form;

$$\begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \begin{pmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} fX_c \\ fY_c \\ Z_c \end{pmatrix} \tag{B.7}$$

3. **Third Mapping - Retina to image plane**

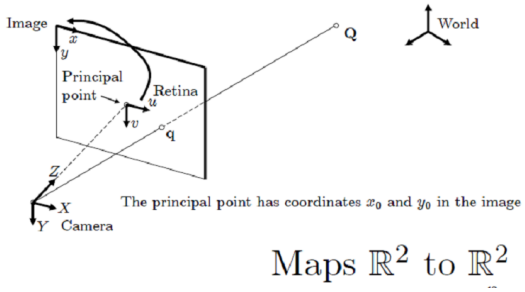


Figure B.3: Retina to image plane

Let k_x and k_y be the density of pixels along u and v. Then we have
$x = k_x u + x_0$ and $y = k_y v + y_0$. We express these two equations in matrix form as follows;

$$\begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} k_x & 0 & x_0 \\ 0 & k_y & y_0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} \tag{B.8}$$

Now if we put all the mappings we have done together we get;

$$q_{image} = \underbrace{\begin{pmatrix} fk_x & 0 & x_0 \\ 0 & fk_y & y_0 \\ 0 & 0 & 1 \end{pmatrix}}_{\text{K}} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} R & t \\ 0 & 1 \end{pmatrix} Q_{world} \tag{B.9}$$

K: Camera calibration matrix that contains intrinsic parameters of the camera.
R and t: Extrinsic parameters of the camera
Q_{world}: The 3D world coordinate
q_{image}: The image coordinate of Q_{world}

The above equation can also be written as;

$$q_{image} = \underbrace{(KRKT)}_{\text{P}} Q_{world} \tag{B.10}$$

$$q_{image} = PQ_{world} \tag{B.11}$$

$$P = K[R|t] \tag{B.12}$$

P: Perspective projection matrix

Finally, a skew parameter(s) can be added for non-orthogonal image retina axis.

B.2 Epipolar Geometry

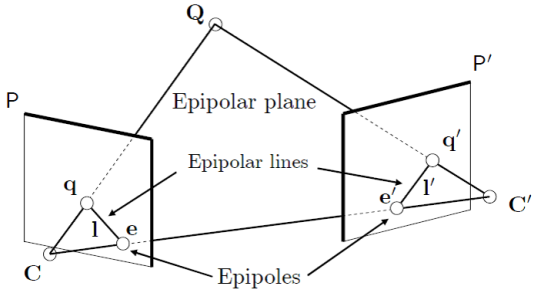


Figure B.4: The flow chart of the system

Epipolar Geometry is the internal geometry between two different views. For a visual representation of epipolar geometry, see figure B.4. The terminology related to epipolar geometry is

- *Epipole* is the point of intersection between the image planes and the line(baseline) defined by two camera centres.
- *Epipolar plane* is the plane containing the baseline.

- *Epipolar lines* are the lines that are at the intersection of the image plane and epipolar plane.

If the point q in one image is known, then the epipolar line l' is known and the 3D point Q projects into the right image, on the point q' which must lie on this particular epipolar line. This means that for each point observed in one image the same point must be observed in the other image on a known epipolar line. This is the definition of epipolar constraint and it means that it is possible to test if two points really correspond to the same 3D point.

The algebric representation of the epipolar geometry is the fundamental matrix(F) and for all corresponding points $x \longleftrightarrow x'$ the fundamental matrix constraint should be satisfied and this constraint is defined as

$$x'^T F x = 0 \tag{B.13}$$

Triangulation is the process of determining the position of a 3D point in 3D space from its projections on two or more images. There are two requirements in order the use triangulation. The calibration parameters and the point correspondences must be known. Then if the points q and q' are known, then their projection lines are also known. If the two image points correspond to the same 3D point X the projection lines must intersect precisely at Q which means that Q can be calculated from the coordinates of the two image points, and this process is known as triangulation.

B.3 The Normalized Eight-Point Algorithm

The eight point algorithm is introduced by C. Longuet-Higgins [17] in 1981 to estimate the essential matrix from a pair of image's corresponding points. The method then argued because of its sensitivity to the noise. In 1997 Richard I. Hartley proposed a normalized version of the algorithm [11]. Hartley showed normalization (translation and scaling) of the coordinates of the matched points is improving the performance of the original algorithm drastically. First the eight point algorithm, then the difference of normalized version is explained.

The corresponding image points x and x' are given as $x = (u, v, 1)^T$ $x' = (u', v', 1)^T$. From the fundamental matrix constraint $x'^T F x = 0$ we have

$$\begin{bmatrix} u & v & 1 \end{bmatrix} \begin{bmatrix} F_{11} & F_{12} & F_{13} \\ F_{21} & F_{22} & F_{23} \\ F_{31} & F_{32} & F_{33} \end{bmatrix} \begin{bmatrix} u' \\ v' \\ 1 \end{bmatrix} = 0 \tag{B.14}$$

The equation B.14 can be rewritten as

$$\begin{bmatrix} uu' & uv' & u & vu' & vv' & v & u' & v' & 1 \end{bmatrix} \begin{bmatrix} F_{11} \\ F_{12} \\ F_{13} \\ F_{21} \\ F_{22} \\ F_{23} \\ F_{31} \\ F_{32} \\ F_{33} \end{bmatrix} = 0 \tag{B.15}$$

Stacking eight corresponding points, find the solution of the system that minimizes $\sum_{i=1}^{N} (x_i^T F x_i^{'})^2$ under the constraint of $|F|^2 = 1$

$$\begin{bmatrix} u_1u_1' & u_1v_1' & u_1 & v_1u_1' & v_1v_1' & v_1 & u_1' & v_1' & 1 \\ u_2u_2' & u_2v_2' & u_2 & v_2u_2' & v_2v_2' & v_2 & u_2' & v_2' & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ u_8u_8' & u_8v_8' & u_8 & v_8u_8' & v_8v_8' & v_8 & u_8' & v_8' & 1 \end{bmatrix} \begin{bmatrix} F_{11} \\ F_{12} \\ F_{13} \\ F_{21} \\ F_{22} \\ F_{23} \\ F_{31} \\ F_{32} \end{bmatrix} = - \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \tag{B.16}$$

The problem of the algorithm is the poor numerical conditioning. This problem is fixed with rescaling(normalization) the data.

1. **Normalization**

   (a) The points are translated so that their centroid is at the origin

   (b) Then the points are scaled so that the average distance from the origin is equal to $\sqrt{2}$ and this transformation is applied to each of the two images independently.

2. **Essential Matrix Estimation**
   Using the algorithm given above compute F for normalized points

3. **Enforce Rank-2 Constraint** Calculate the SVD of F and make sure that the $d_{33} = 0$

in the equation below.

$$F = UDV^T = U \begin{bmatrix} d_{11} & & \\ & d_{22} & \\ & & d_{33} \end{bmatrix} \begin{bmatrix} v_{11} & \cdots & v_{13} \\ \vdots & \ddots & \vdots \\ v_{31} & \cdots & v_{33} \end{bmatrix}$$

4. **De-Normalization** The fundamental matrix is transformed back to original units. If one assumes that T and T' are the normalizing transformations for two image points, then the fundamental matrix of the original coordinates will be $T^T F T'$.

B.4 Singular Value Decomposition

Singular Value Decomposition(SVD) is a matrix factorization technique for real or complex matrices. Any m by n matrix A can be factorized into

$$A = Q_1 \Sigma Q_2^T = (orthogonal)(diagonal)(orthogonal)$$

The columns of Q_1(m by m) are eigenvectors of AA^T, and the columns of Q_2(n by n) are eigenvectors of $A^T A$. The r singular values on the diagonal of Σ(m by n) are the square roots of the non-zero eigenvalues of both AA^T and $A^T A$.

B.5 The Absolute Conic

The absolute conic(AC), Ω, is a conic on the plane at infinity. The AC is defined by a set of points satisfying the equations below

$$\begin{aligned} x_1^2 + x_2^2 + x_3^2 &= 0 \\ x_4 &= 0 \end{aligned} \tag{B.17}$$

Let x_∞ be a point on AC. By definition, we have $x_\infty^T x_\infty = 0$ and $\tilde{x}_\infty = \begin{bmatrix} x_1 & x_2 & x_3 & 0 \end{bmatrix}^T$ and $\tilde{x}_\infty^T \tilde{x}_\infty = 0$. This can be interpreted as the AC is a conic of purely imaginary points on the plane at infinity(see Figure B.5)

The AC is invariant to any rigid transformation. Now let a rigid transformation be $H = \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix}$ and x_∞ be a point on AC. Then the projective coordinates of the point is $\tilde{x}_\infty = \begin{bmatrix} x_\infty \\ 0 \end{bmatrix}$ with

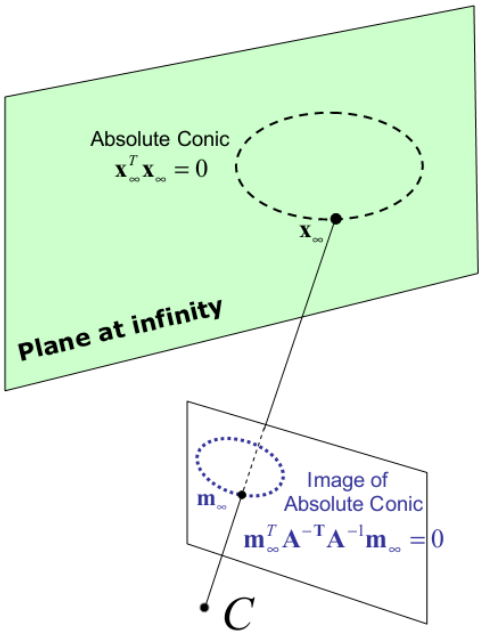


Figure B.5: The AC and the IAC(Image Courtesy [20])

$x_\infty^T x_\infty = 0$. Now if we denote the point after transformation x'_∞ we have

$$\tilde{x'}_\infty = H\tilde{x}_\infty = \begin{bmatrix} Rx_\infty \\ 0 \end{bmatrix} \tag{B.18}$$

Hence, x'_∞ is on the plane at infinity and on the AC because

$$x_\infty^{'T} x_\infty = (Rx_\infty)^T (Rx_\infty) = x_\infty^T (R^T R) x_\infty = 0 \tag{B.19}$$

The image of the absolute conic(IAC), ω, is determined only by the intrinsic parameters of the camera and does not depend on the extrinsic parameters of the camera. From B.5 notice that, the projection of x_∞, m_∞ is defined by

$$\tilde{m}_\infty = sA \begin{bmatrix} R & t \end{bmatrix} \begin{bmatrix} x_\infty \\ 0 \end{bmatrix} = sARx_\infty \tag{B.20}$$

Moreover, from B.20 we have

$$\tilde{m}^T A^{-T} A^{-1} \tilde{m} = s^2 x_\infty^T R^T R x_\infty = s^2 x_\infty^T x_\infty = 0 \tag{B.21}$$

Finally, the IAC is defined by $A^{-T}A^{-1}$. This means that if we can find the IAC, we can find the camera intrinsic parameters.

B.6 Extraction of the Chessboard Corners

OpenCV method(findChessBoardCorners) for chessboard corner detection is taking two inputs; the number of rows and columns of the target chessboard pattern. For example the input of the method for the pattern shown in 2.3 is 5 and 7 for rows and columns respectively. The method first applies a thresholding, then erosion. The aim for both processes is to break the connectivity between the squares. Then the method tries to obtain the input numbers for number of squares. The erosion process is repeated as many times as it requires to break the connectivity and obtain the required number of squares. Finally, the squares are brought back together to form the board pattern.

Bibliography

[1] Y I Abdel-Aziz and H M Karara. *Direct linear transformation from comparator coordinates into object space coordinates in close-range photogrammetry*, volume 1, pages 1–18. Falls Church: American Society of Photogrametry, 1971.

[2] Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. Speeded-up robust features (surf). *Comput. Vis. Image Underst.*, 110(3):346–359, June 2008.

[3] Jean-Yves Bouguet. Camera calibration toolbox. 2000.

[4] Stphane Christy and Radu Horaud. Euclidean shape and motion from multiple perspective views by affine iterations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18:1098–1104, 1996.

[5] Olivier Faugeras. *Three-dimensional computer vision: a geometric viewpoint*. MIT Press, Cambridge, MA, USA, 1993.

[6] Olivier D. Faugeras, Quang-Tuan Luong, and Stephen J. Maybank. Camera self-calibration: Theory and experiments. In *Proceedings of the Second European Conference on Computer Vision*, ECCV '92, pages 321–334, London, UK, UK, 1992. Springer-Verlag.

[7] Martin A. Fischler and Robert C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6):381–395, June 1981.

[8] Chris Harris and Mike Stephens. volume 15, pages 147–151. Manchester, UK, 1988.

[9] Richard Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, New York, NY, USA, 2 edition, 2003.

[10] Richard I. Hartley. Self-calibration from multiple views with a rotating camera. pages 471–478. Springer-Verlag, 1994.

[11] Richard I. Hartley. In defense of the eight-point algorithm. *IEEE Trans. Pattern Anal. Mach. Intell.*, 19(6):580–593, June 1997.

[12] J Heikkila and O Silven. *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 0(10):1106–1112, 1997.

[13] David W. Jacobs. Linear fitting with missing data for structure-from-motion. *Computer Vision and Image Understanding*, 82:2001, 1997.

[14] Ismael Kassem. Structure from motion and camera self-calibration. In *Master Thesis, Master of Science in Vision and Robotics*, University of Burgundy.

[15] Peter Kovesi. Matlab and octave functions. 2012.

[16] D. Liebowitz and A. Zisserman. Metric rectification for perspective images of planes. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, CVPR '98, pages 482–, Washington, DC, USA, 1998. IEEE Computer Society.

[17] H. C. Longuet-Higgins. Readings in computer vision: issues, problems, principles, and paradigms. chapter A computer algorithm for reconstructing a scene from two projections, pages 61–62. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1987.

[18] David G. Lowe. Object recognition from local scale-invariant features. In *Proceedings of the International Conference on Computer Vision-Volume 2 - Volume 2*, ICCV '99, pages 1150–, Washington, DC, USA, 1999. IEEE Computer Society.

[19] Shyjan Mahamud, Martial Hebert, Yasuhiro Omori, and Jean Ponce. Provably-convergent iterative methods for projective structure from motion, 2001.

[20] Gerard Medioni and Sing Bing Kang. *Emerging Topics in Computer Vision*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 2004.

[21] J More. *The Levenberg-Marquardt algorithm: implementation and theory*, volume 630, pages 105–116. Springer, 1978.

[22] E. Mouragnon, M. Lhuillier, M. Dhome, F. Dekeyser, and P. Sayd. Generic and real-time structure from motion using local bundle adjustment. *Image Vision Comput.*, 27(8):1178–1193, July 2009.

[23] David Nistér. An efficient solution to the five-point relative pose problem. *IEEE Trans. Pattern Anal. Mach. Intell.*, 26(6):756–777, June 2004.

[24] John Oliensis, Senior Member, Richard Hartley, and Senior Member. Iterative extensions of the sturm/triggs algorithm: Convergence and nonconvergence. In *In ECCV (4*, pages 214–227, 2006.

[25] Srikumar Ramalingam, Suresh K. Lodha, and Peter Sturm. A generic structure-from-motion framework. *Comput. Vis. Image Underst.*, 103(3):218–228, September 2006.

[26] Fabio Remondino and Clive Fraser. Digital camera calibration methods: considerations and comparisons. *International Archives of Photogrammetry Remote Sensing and Spatial Information Sciences*, 36(5):266–272, 2006.

[27] Aldebaran Robotics. Documentation. 2012.

[28] J Salvi, X Armangu, and J Batlle. A comparative review of camera calibration methods with accuracy evaluation. *Pattern Recognition*, 35:1617–1635, 2002.

[29] J.G. Semple and G.T. Kneebone. *Algebraic Projective Geometry*. Oxford Classic Texts in the Physical Sciences. Clarendon Press, 1998.

[30] Asst. Prof. Desire Sidibe. Computer vision, lecture notes. 2011.

[31] Peter Sturm and Bill Triggs. A factorization based algorithm for multi-image projective structure and motion. pages 709–720. Springer-Verlag, 1996.

[32] Carlo Tomasi. Shape and motion from image streams under orthography: a factorization method. *International Journal of Computer Vision*, 9:137–154, 1992.

[33] R Tsai. A versatile camera calibration technique for high-accuracy 3d machine vision metrology using off-the-shelf tv cameras and lenses. *IEEE Journal on Robotics and Automation*, 3(4):323–344, 1987.

[34] www.python.org. Python documantation. 2012.

[35] Zhengyou Zhang. A flexible new technique for camera calibration. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22(11):1330–1334, November 2000.