Western Graduate&PostdoctoralStudies

Western University
Scholarship@Western

Electronic Thesis and Dissertation Repository

1-16-2018 11:00 AM

Feature Based Calibration of a Network of Kinect Sensors

Xiaoyang Li, *The University of Western Ontario*

Supervisor: Steven Beauchemin, *The University of Western Ontario*
Co-Supervisor: Michael Bauer, *The University of Western Ontario*

A thesis submitted in partial fulfillment of the requirements for the Master of Science degree in Computer Science

© Xiaoyang Li 2018

Follow this and additional works at: https://ir.lib.uwo.ca/etd

Part of the Artificial Intelligence and Robotics Commons

Recommended Citation

Li, Xiaoyang, "Feature Based Calibration of a Network of Kinect Sensors" (2018). *Electronic Thesis and Dissertation Repository*. 5180.
https://ir.lib.uwo.ca/etd/5180

This Dissertation/Thesis is brought to you for free and open access by Scholarship@Western. It has been accepted for inclusion in Electronic Thesis and Dissertation Repository by an authorized administrator of Scholarship@Western. For more information, please contact wlswadmin@uwo.ca.

Abstract

The availability of affordable depth sensors in conjunction with common RGB cameras, such as the Microsoft Kinect, can provide robots with a complete and instantaneous representation of the current surrounding environment. However, in the problem of calibrating multiple camera systems, traditional methods bear some drawbacks, such as requiring human intervention. In this thesis, we propose an automatic and reliable calibration framework that can easily estimate the extrinsic parameters of a Kinect sensor network. Our framework includes feature extraction, Random Sample Consensus and camera pose estimation from high accuracy correspondences. We also implement a robustness analysis of position estimation algorithms. The result shows that our system could provide precise data under certain amount noise.

Keywords

Kinect, Multiple Camera Calibration, Feature Points Extraction, Correspondence, RANSAC

Acknowledgments

Foremost, I would like to express my sincere gratitude to my advisors Prof. Steven Beauchemin and Prof. Michael Bauer for their continuous support of my Master study and thesis, for their patience, motivation, enthusiasm, and immense knowledge. The door to Prof. Beauchemin’s office was always open whenever I ran into a trouble spot or had a question about my research or writing. I could not have imagined having a better advisor and mentor for my Master study.

Besides my advisors, I would like to thank the rest of my thesis committee: Prof. John Barron, Prof. Kostas Kontogiannis, and Prof. Ken Mclsaac, for their encouragement, insightful comments, and hard questions.

Last but not the least, I would like to thank my family: my parents Gan Li and Guixiang Li, for giving birth to me in the first place and supporting me spiritually throughout my life. This accomplishment would not have been possible without them. Thank you!

Table of Contents

Abstract i

Acknowledgments ii

Table of Contents iii

List of Tables vi

List of Figures vii

Chapter 1 1

1 Introduction 1

  1.1 Overview 1

  1.2 Kinect Mechanism 1

    1.2.1 Kinect Sensing Hardware 2

  1.3 Problem and Issues 5

  1.4 Thesis Contribution 6

  1.5 Thesis Contents 6

Chapter 2 7

2 Related Work 7

Chapter 3 11

3 System Description 11

  3.1 Stereo Vision Theory 12

    3.1.1 Disparity and Depth 12

    3.1.2 Coordinate Systems 14

    3.1.3 Pinhole Camera Model 15

3.2 Camera Calibration 17
    3.2.1 Intrinsic Parameters 17
3.3 Epipolar Geometry 21
3.4 Feature Detection 26
    3.4.1 SIFT Feature 26
    3.4.2 SURF Feature 30
3.5 Correspondence 34
    3.5.1 Overview 34
    3.5.2 Random Sample Consensus (RANSAC) 35
3.6 Camera pose estimation Algorithm 37
    3.6.1 Eight-Points Algorithm (2D-2D) 37
    3.6.2 Perspective-n-Point Algorithm (3D-2D) 41
    3.6.3 Point Set Registration (3D-3D) 43
Chapter 4 46
4 Methodology 46
4.1 Feature Selection 46
4.2 Correspondence Method Selection 46
4.3 Experiments with Real RGB-D Images 47
    4.3.1 Case 1 47
    4.3.2 Case 2 47
4.4 Robustness Analysis 48
Chapter 5 50
5 Experiments and Results 50

5.1 Results with Real RGB-D Data 51

5.1.1 Case 1 51

5.1.2 Case 2 55

5.2 Robustness Analysis 57

Chapter 6 64

6 Future Work 64

6.1 System Constraints 64

6.2 Device Limitations 64

References 67

Curriculum Vitae 72

List of Tables

Table 1. Comparative Specifications of Kinect v1 and Kinect v2. 4

Table 2 Comparison of results from different camera pose estimation algorithms 54

Table 3 Comparative results from different camera pose estimation algorithms 57

Table 4 Synthetic Data 59

Table 5 Theta at sigma=0.8 and sigma=0.9 59

List of Figures

Figure 1.1. Hardware configuration of Kinect v2. ... 3

Figure 1.2. Illustration of Kinect depth measurement. ... 4

Figure 3.1 Flow chart r the whole system ... 12

Figure 3.2 Stereo cameras projection. ... 13

Figure 3.3 Projection between different coordinate systems ... 15

Figure 3.4 A pinhole camera model. ... 17

Figure 3.5 Barrel Distortion and Pincushion Distortion ... 19

Figure 3.6 Points correspondence geometry (a) ... 22

Figure 3.7 Points correspondence geometry (b) ... 22

Figure 3.8 Illustration of Epipolar geometry ... 23

Figure 3.9 Epipolar constraint ... 26

Figure 3.10 Input image applied with different Gaussian kernel. ... 28

Figure 3.11 SIFT key point descriptor ... 30

Figure 3.12 Differences between SIFT (left side) and SURF (right side) when constructing a scale space. ... 32

Figure 3.13 Orientation assignment of SURF ... 33

Figure 3.14 Feature descriptor for SURF ... 34

Figure 3.15 Four solution form decompose Essential Matrix.. 40

Figure 3.16 Re-center dataset 44

Figure 5.1 RGB-D image structure. 51

Figure 5.2 Case 1: Feature Points.. 53

Figure 5.3 Case 1: Correspondences.. 54

Figure 5.4 Case 2: Feature Points 56

Figure 5.5 Case 2: Correspondences... 56

Figure 5.6 Gaussian Distribution with different values for sigma... 60

Figure 5.7 Comparative offsets of R for sigma $\in$ [0,1]... 61

Figure 5.8 Comparative offsets of T for sigma $\in$ [0,1] 61

Figure 5.9 Comparative offsets of R for sigma $\in$ [0,0.7]... 62

Figure 5.10 Comparable offsets of T for sigma $\in$ [0,0.7] 62

Figure 6.1 The reflection problem with a transparent glass... 65

Figure 6.2 Different RGB-D images caused by a change in ambient temperature................ 65

Chapter 1

1 Introduction

1.1 Overview

The robotics industry has been developing rapidly in the past few years. Typical robotic tasks, like simultaneous localization and mapping (SLAM), navigation, object recognition and many others, greatly benefit from having color and depth information fused together. Traditional high-cost 3D profiling cameras often result in lengthy acquisition and slow processing of massive amounts of information. With the invention of the low-cost Microsoft Kinect sensor, high-resolution depth and visual (RGB) sensing has created many opportunities for multimedia computing. Our objective is to design and implement a system for calibrating multiple Kinect sensors in different views.

1.2 Kinect Mechanism

Kinect contains a normal RGB camera, an Infrared Sensor and a four-microphone array. Combining these devices, the Kinect is able to provide RGB images, depth images and audio signals simultaneously, which encourages varied applications in different fields, such as image signal synchronization, human 3-D motion capture, human face identification.

1.2.1 Kinect Sensing Hardware

Kinect is a composite device consisting of a near-infrared laser pattern projector, an IR camera and a color (RGB) camera; Figure 1.1 shows the arrangement of the sensors on a Kinect. The laser pattern projector and the IR camera are used as a stereo pair to capture depth information in 3D space. The IR projector casts an IR speckle dot pattern into the 3D scene while the IR camera captures the reflected IR speckles. Due to the uniqueness of each projected dot, the depth of a point can be captured by relative left-right translation of the dot pattern. This translation is dependent on the distance of the object to the camera-projector plane. Such a procedure is illustrated in Figure 1.2. [1]

Each component of the Kinect hardware is described below.

1) RGB Camera: It delivers three basic color components of the video. The camera operates at 30 Hz, and can offer images at 1920×1080 pixels with 8-bit per channel. The angular field of view (FOV) is 84.1° horizontally and 54.8° vertically.

2) 3-D Depth Sensor: It consists of an IR laser projector and an IR camera. Together, the depth image is constructed by triangulation from the stereo pair. The sensor has a practical ranging limit of 0.5 m−8.0 m distance, and outputs video at 30 frames/s with the resolution of 512×424 pixels. The angular field of view (FOV) is 70.6° horizontally and 60.0° vertically.

Table 1 also provides comparative specifications of the Kinect v2, which is used in this thesis, and the previous Kinect v1.

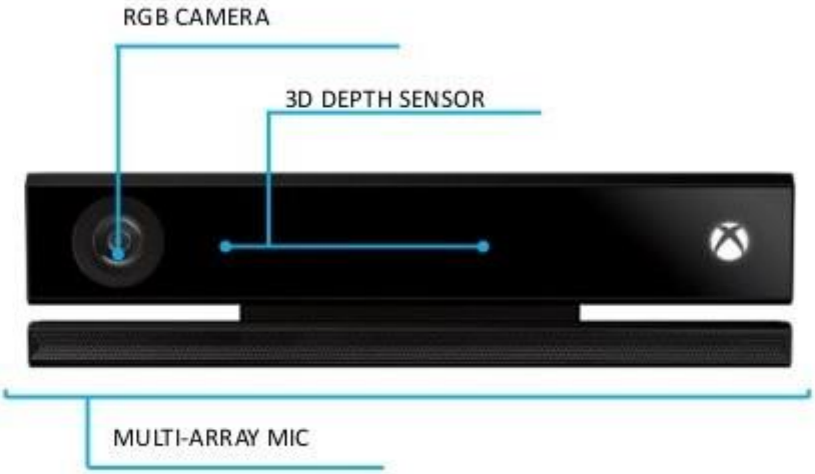


Figure 1.1. Hardware configuration of Kinect v2.

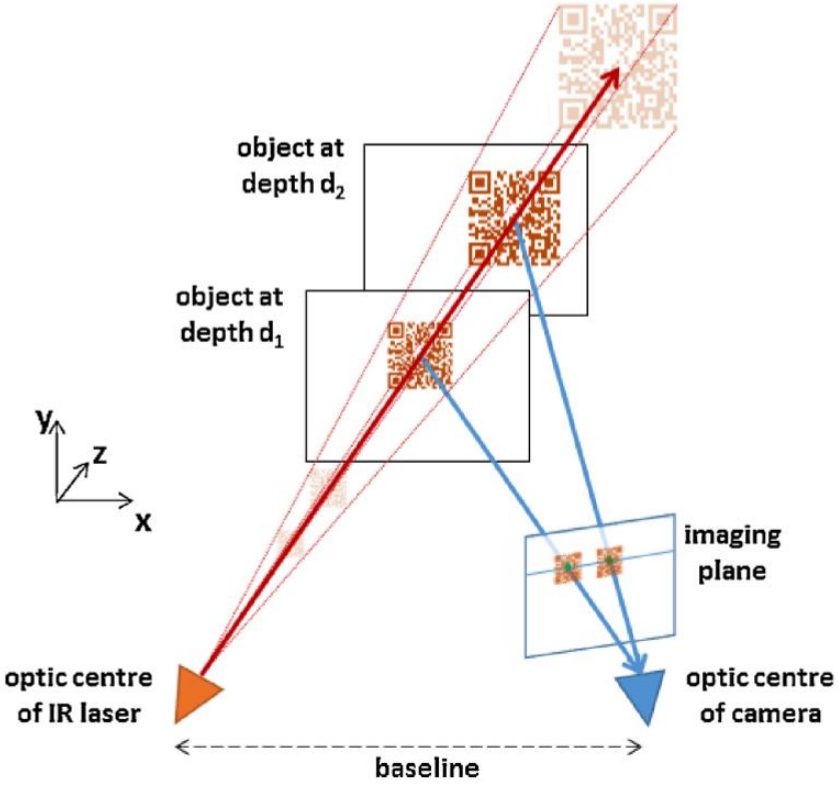


Figure 1.2. Illustration of Kinect depth measurement. IR camera could sense the depth by the unique project pattern.

Table 1. Comparative Specifications of Kinect v1 and Kinect v2.

	Kinect v1	Kinect v2
Resolution of RGB image	640 × 480(pixel)	1920 × 1080(pixel)
Resolution of IR image	320 × 240(pixel)	512 × 424(pixel)
Field view of RGB image	62° × 48.6°	84.1° × 53.8°
Field view of IR and depth image	57.5° × 43.5°	70.6° × 60°

Maximum skeletal tracking	2	6
Method of depth measurement	Lighting coding	Time of Flight
Working range	0.8m~3.5m	0.5m~8.0m

1.3 Problem and Issues

Multiple-camera systems have become increasingly prevalent in robotics and computer vision research. One of the most important tasks is how to combine the visual information in those cameras. This problem, in general, can be interpreted as a mathematical problem. Given two cameras, each with their own 3D coordinate system, they obtain information of the same scene from different views. There should be a rigid transformation between the two camera coordinates. Specifically, in 3D space, this transformation is determined by a rotation matrix R and a transit vector $\vec{\boldsymbol{t}}$.

This problem can be stated as follows:

Assume that we have two calibrated cameras C_l and C_r, such that the intrinsic matrices are K_l and K_r. However, the absolute positions expressed in a world coordinate system O_w between them are unknown. By giving the input RGB + Depth (RGB-D) images of the same scene from those two cameras, can we design a system that can automatically determine the unknown values of R and $\vec{\boldsymbol{t}}$.

1.4 Thesis Contribution

This thesis presents and evaluates a framework that integrates RGB-D data capture, feature points selection, correspondence optimization and camera pose reconstruction into one application. The thesis also includes a robustness analysis of the approach which demonstrates that the system is resistant to noisy data and provides precise estimation. Besides, this thesis is the first one to introduce 3D eight-points algorithm and numerically compare those three pose estimation algorithms.

1.5 Thesis Contents

This thesis is organized as follows:

Chapter 2 briefly describes the traditional methods of camera calibration as well as previous work on 3D sensor calibration and address the deficiency and constraints of each algorithm. Chapter 3 is the core of this thesis which explained in detail each stage of the processing within the system. With respect to feature detection and correspondence selection, we rely on the work of Lowe [2][3] and Bay and Herbert [4]. We also leverage the camera pose estimation algorithms proposed by Lepetit [5] and Hartley [6]. In Chapter 4, the methodology that is used to build the complete system is presented and the sets of experiments are described. The results presented in Chapter 5 illustrate the effectiveness of the system and demonstrate its robustness to noise. Finally, Chapter 6 lists some areas of future work for potentially improving the system and proposes new directions that overcome some of the inherent constraints in our current work.

Chapter 2

2 Related Work

Camera calibration, in general, is the process of estimating the parameters of a pinhole camera model approximating the camera that produced a given photograph or video. The extrinsic calibration of a camera from independent pairwise correspondences with multiple views is essentially the problem of estimating the camera positions between non-central cameras.

Many calibration methods have been proposed to do the calibration work using a calibration object with known world coordinates. One possible way is to use a calibration object, which consists of several easily detectable non-coplanar feature points with known relative 3D positions, such as a checkerboard pattern [7], [8], [9]. Corresponding points between 3D world points and image points could be accomplished by fixing the world coordinate system in the calibration object. Then, by optimizing the mapping from picture coordinates system to corresponding 3D world coordinates, both intrinsic and extrinsic parameters could then be estimated. This optimization problem could be solved linearly if 17 correspondences are given [10]. Apart from the time complexity, 6 pairs of correspondences, at a minimum, are also able to be used to find an acceptable estimation [11]. However, this 6-pairs correspondence method cannot output reliable results for some particular situations, for example, if one of the generalized views is a pin-hole. For the case of any arbitrary combination of correspondences between the 4 views of two stereo rigs, Vasconcelos et al. propose a non-minimal solution using 10 correspondences [12].

To conclude, these kinds of algorithms can solve the problem for calibrating a single camera efficiently. However, when it comes to a camera network, which contains multiple cameras, it is tedious and cumbersome to calibrate all the cameras simultaneously using these reference objects, as it is often extremely difficult to make all the points on the calibration object simultaneously visible in all views. Moreover, designing highly accurate tailor-made calibration patterns are often difficult and expensive.

Another option [13], [14] is to observe the object through planar mirror reflections in order to handle situations of little or no overlap in the focal views. They overcome the need for all cameras to have overlapping direct views by allowing them to see the object through a mirror. They first adopt standard calibration methods to find the internal and external parameters of a set of mirrored cameras poses and then estimate the external parameters of the real cameras from their mirrored poses by formulating constraints between them. However, those calibration procedures are explicit, in the sense that they require substantial human intervention, and are meant to be carried out as an initial off-line step before starting to operate the network of cameras.

People have also turned to 3D sensors, which offer depth information in more convincing and precise ways. An important task for multiple cameras is the calibration for 3D sensors.

Auvinet et al. [15] proposed a method for calibrating multiple depth cameras by using only depth information. Their algorithm is based on plane intersections and the Network Time Protocol for data synchronization. The calibration achieves good results even if the depth error of the sensor is 10 mm. A drawback of their implementation is that they have to manually select the plane corners and, above all, they only deal with depth sensors, thus avoiding the possibility to add the color information to the fused data.

Another approach to solve the calibration problem is the one proposed by Le and Ng [16]: they jointly calibrate groups of sensors. Each group is composed by a set of sensors that can provide a 3D representation of the world. They first calibrate the intrinsic parameters of the sensors individually, then they calibrate the extrinsic parameters of each group and the extrinsic parameters of each group with respect to all the others. In the end, they refine the calibration parameters of the entire system in one optimization step. Their experiments show that this method not only reduces the calibration error, but also requires little human intervention.

In 2010, the Kinect for Xbox 360 sensor was introduced by Microsoft as an affordable and real-time source for RGB-D data, which was dedicated to gesture detection and recognition in a game controller. Due to its acquisition speed, many researchers have recognized the potential of the Kinect's RGB-D imaging technology, which gives a significant rise to this field.

In order to merge data collected from different Kinect sensors, various approaches have been proposed for simultaneous calibration of Kinect's sensors. Burrus [17] proposes to use traditional techniques for calibrating the Kinect color camera involving manual selection of the four corners of a checkerboard for calibrating the depth sensor. Zhang et al. [18] propose a semi-automatic method. They automatically sample the planar target to collect the points for calibration of depth sensor and then manually select corresponding points between color and depth images. Combing them together, the extrinsic relationship within a single Kinect sensor is established. Gaffney [19] describes a technique to calibrate the depth sensor by using 3D printouts of cuboids to generate different levels in depth images. However, after that it requires an elaborate process to construct the target. To avoid the need for blocking the projector when calibrating, Berger et al. use a checkerboard where black boxes are replaced with mirroring aluminum foil [20].

In conclusion, traditional methods fail to use depth information effectively which makes them unable to determine the distance in the real world. A number of the previous 3D sensor calibrating algorithms require human intervention and thus are tedious and complex when calibration of multiple cameras is needed. This thesis provides a system that can not only determine camera poses with a high precision but it is also robust to noisy data. More importantly, it is an automatic system that requires no human intervention.

Chapter 3

3 System Description

In this Chapter, we present each component of our system in chronological order. We introduce the theory of RGB-D image formation, depth information, pinhole camera models and four coordinate systems. Following this, the problem of calibrating the camera system is proposed. This part contains the definition of intrinsic and extrinsic camera parameters and discusses the phenomenon of camera lens distortion. In order to introduce the Essential and Fundamental Matrices, the theory of Epipolar Geometry is also discussed.

We then discuss some common computer vison tasks such as, feature detection and matching. Two scale-invariant feature detection algorithms, SIFT and SURF, are presented. We also discuss a robust stochastic method (RANSAC) commonly used to match features from source images.

Three methods of estimating camera pose from correspondences are presented. These are: the Eight-Point algorithm (uses 2D correspondences), Perspective-n-Point (uses 3D-2D correspondences), and 3D Point Registration (uses 3D correspondences). The system is described as a flow chart in Figure 3.1

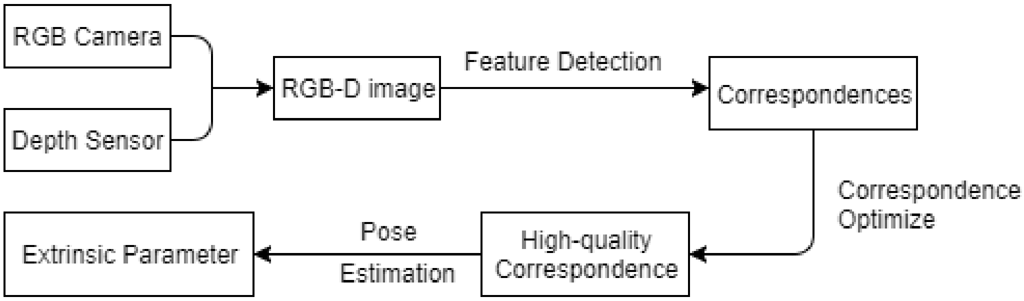


Figure 3.1 Flow chart r the whole system

3.1 Stereo Vision Theory

Stereo vision refers to the process of extracting 3D information from two (or more) 2D views of a scene. In this Section, common knowledge is discussed such as disparity and depth. In addition, the pinhole camera model as well as different coordinate systems are introduced in order to discuss stereo vison effectively.

3.1.1 Disparity and Depth

Human eyes are horizontally separated by about 50–75 mm, depending on each individual. Thus, each eye has a slightly different view of the world around. The term *binocular disparity* refers to geometric measurements related to the estimation of scene depth. In computer vision, binocular disparity is calculated from stereo images taken from a set of stereo cameras. The variable distance between these cameras, called the baseline, can affect the disparity of a specific point on their respective image plane

[21]. Figure 3.2 shows one point in the real world projected onto two image planes by two cameras in different positions.

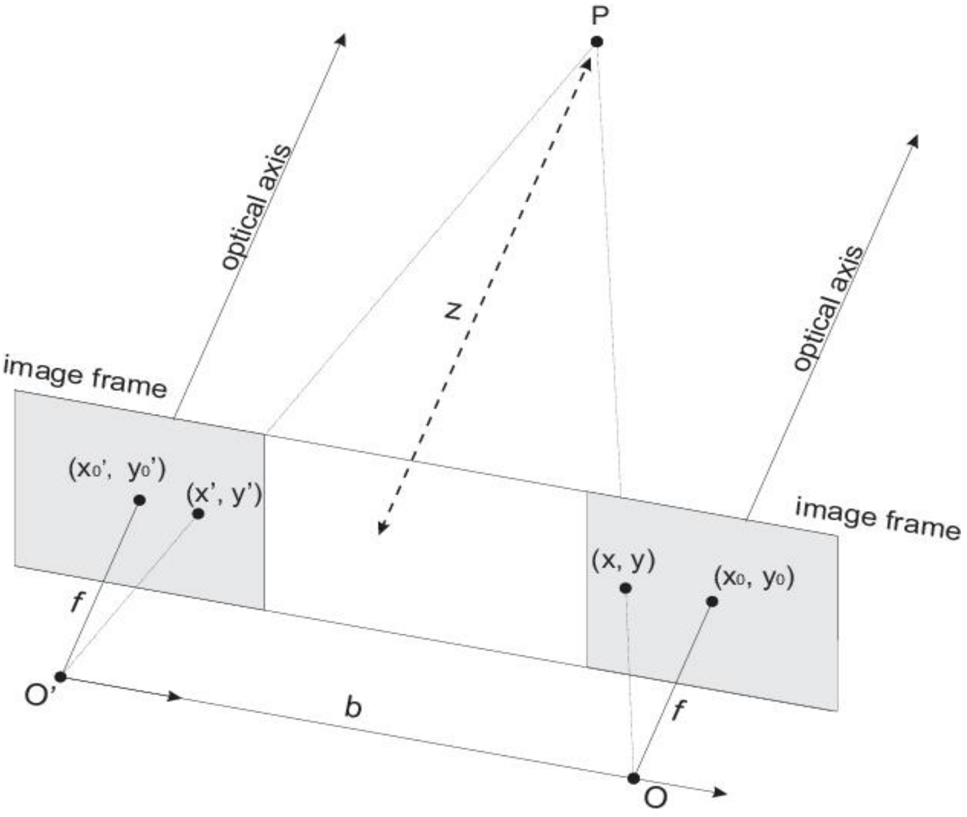


Figure 3.2 Stereo cameras projection. The quantity b refers to baseline, and $(x - x')$ is disparity.

Disparity can be used in the extraction of information from stereo images. One case that disparity is most useful is for 3D depth estimation. Disparity and distance from the cameras are inversely related. As the distance of an object from the cameras increases, the disparity decreases. According to triangulation, we have

$$\text{Disparity d} = x - x' = \text{b} \cdot \frac{f}{z}$$

Therefore

$$\text{Depth } z = \frac{b \cdot f}{x - x\prime} = \frac{b \cdot f}{d} \tag{3.1}$$

where x and x' are the distance between points in the image plane corresponding to the 3D scene point and their camera centers. b is the distance between two cameras and f is the focal length of camera.

3.1.2 Coordinate Systems

1) World Coordinate $(\mathrm{U}, \mathrm{V}, \mathrm{W})$

It is the absolute coordinate system of the world. In general, a 3D scene is expressed in this coordinate system.

2) Camera Coordinate $(\mathrm{X}, \mathrm{Y}, \mathrm{Z})$

This coordinate system is related to the pinhole camera model. In this coordinate system, Z is the optical axis (or line if sight), with the image plane located at f units away along the optical axis. f is known as the focal length.

3) Film Coordinate (x, y)

By forward projecting point from camera coordinates onto the image plane, a 2D coordinate is obtained, which is called a film coordinate.

4) Pixel Coordinate (u, v)

In images, the points are measured in pixel units, so the film coordinate needs to be digitalized into the pixel coordinate. In this coordinate, the center is normally located in the upper left corner of the image and the *u* and *v* axes are parallel to *x* and *y* axes in film coordinates.

Figure 3.3 shows the relationship between the four coordinate systems.

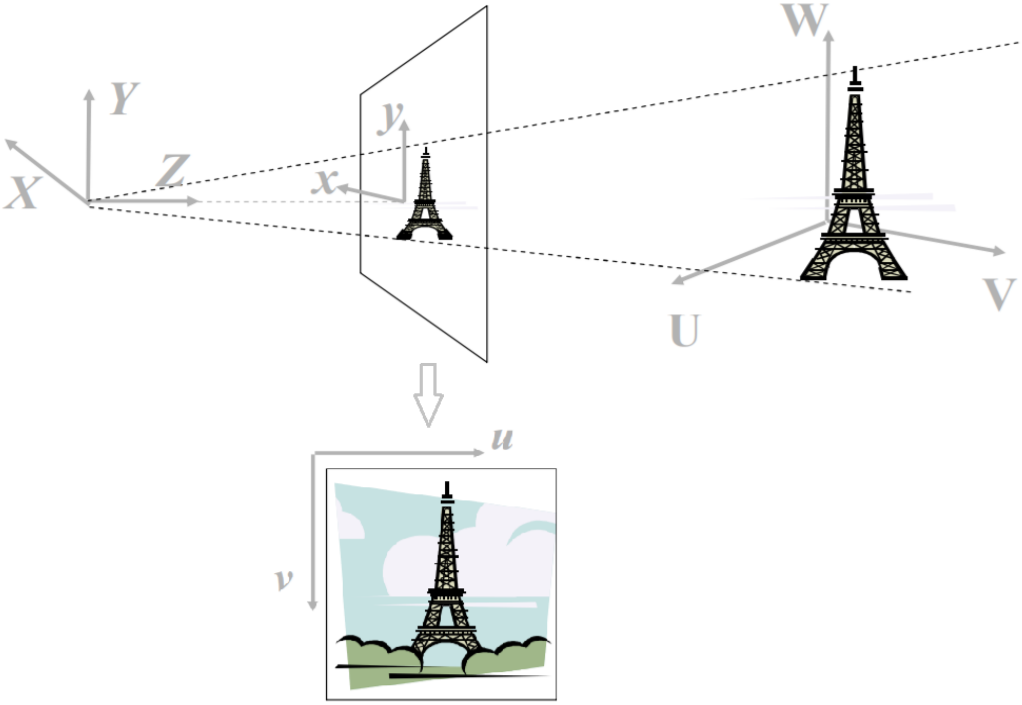


Figure 3.3 Projection between different coordinate systems

3.1.3 Pinhole Camera Model

Given a camera coordinate system with f as its focal length. The line from center of projection and perpendicular to the image plane is called principal axis of the camera and the intersection of this principal axis with the image plane is called the principal point.

Assume (X, Y, Z) is the 3D coordinates of an image point $p(x, y)$ on the camera image plane. Then, the 3D point and its 2D image point are related as:

$$x = f\frac{X}{Z}\,, y = f\frac{Y}{Z} \qquad (3.2)$$

This relation is known as perspective projection. However, *x* and *y* are expressed in the units of the world coordinate system, and need to be converted to pixel units. Transforming the coordinates requires knowing the column-wise and row-wise density of the pixels (pixels per millimeter), let them be $a_u$and a_v respectively. We also need to know the deviation of the principal point from the image center. Let its coordinates be $(-x_0, -y_0)$. Then p can be written as

$$u = a_u(x + x_0) = a_u f\frac{X}{Z} + a_u x_0 \qquad (3.3)$$

$$v = a_v(y + y_0) = a_v f\frac{Y}{Z} + a_v y_0\,.$$

The pinhole camera model is illustrated in Figure 3.4.

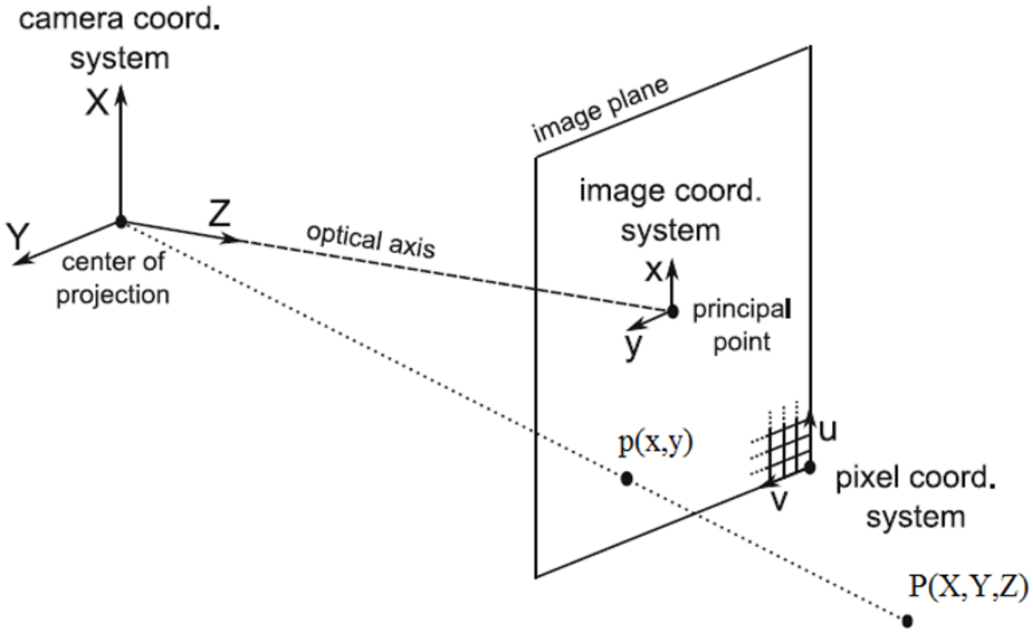


Figure 3.4 A pinhole camera model.

3.2 Camera Calibration

3.2.1 Intrinsic Parameters

Equation (3.3)

$$u = a_u(x + x_0) = a_u f \frac{X}{Z} + a_u x_0$$

$$v = a_v(y + y_0) = a_v f \frac{Y}{Z} + a_v y_0 \ .$$

can written in matrix form. First, let point *P* be expressed as:

$$P' = \frac{P}{Z} = \begin{pmatrix} X/Z \\ Y/Z \\ Z/Z \end{pmatrix} = \begin{pmatrix} X' \\ Y' \\ 1 \end{pmatrix} \tag{3.4}$$

Then (3.3) can be expressed as:

$$\begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \begin{pmatrix} a_u f & 0 & a_u x_0 \\ 0 & a_v f & a_v y_0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} X' \\ Y' \\ 1 \end{pmatrix} = KP' \tag{3.5}$$

and let

$$f_x = a_u f, f_y = a_v f, c_x = a_u x_0, c_y = a_v y_0.$$

Those four quantities are determined by the internal structure of the camera, and are known as intrinsic parameters. Therefore, the camera intrinsic parameter matrix K can be represented as:

$$K = \begin{pmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{pmatrix} \tag{3.6}$$

Normally, the intrinsic parameters of the camera are usually found by calibration [22]. For the Kinect cameras, they can be obtained using the Microsoft Kinect SDK v2.0 [23].

3.2.1.1 Distortion Coefficients

In the section 3.1.3, we introduced the pinhole camera model, disregarding the possibility of distortions introduced by the camera assembly and the lens itself. Because of such imperfections, when projecting points into the image plane, distortions are introduced.

There are two types of distortions: optical and perspective. Both result in some type of image deformations, lightly or noticeably. In optical distortion, there has been two most common and significant distortions: Barrel Distortion and Pincushion Distortion. Figure 3.5 shows how they deform images.

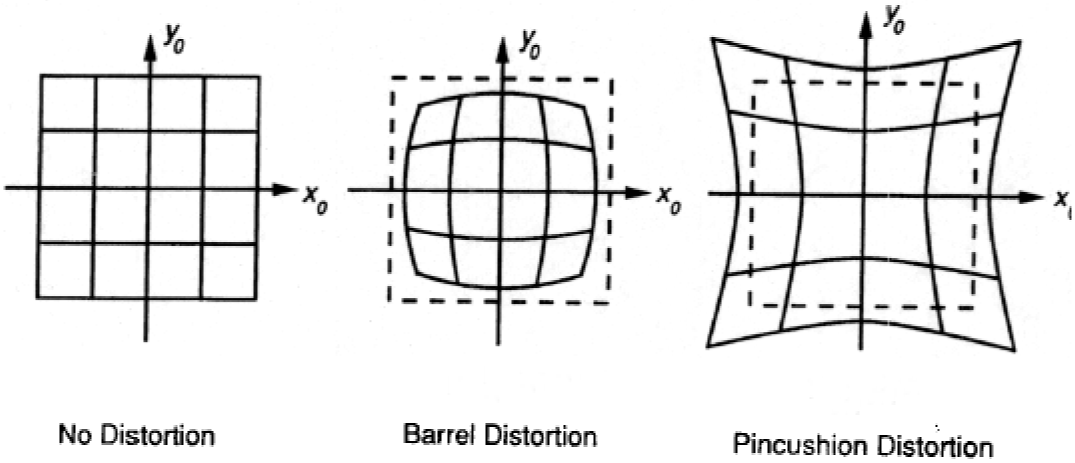


Figure 3.5 Barrel Distortion and Pincushion Distortion

These distortions are accounted for and corrected with relatively simple models [22], [24]. Let (x, y) be the ideal coordinates on image plane, and (x_d, y_d) the corresponding, real observed coordinates. Also, let $(0,0)$ denote the principal point, free of any distortion. Then we can write:

$$x_d = x \cdot (1 + k_1 r^2 + k_2 r^4 + k_3 r^6), \tag{3.7}$$

$$y_d = y \cdot (1 + k_1 r^2 + k_2 r^4 + k_3 r^6),$$

Where

$$\mathrm{r} = \sqrt{x^2 + y^2}$$

and k_1, k_2, k_3 are radial distortion coefficients. Zhang claims that the first two terms in these equations are sufficient to adequately undistort images in most cases [2].

3.2.1.2 Extrinsic Parameters

In the pinhole camera model, we have a prerequisite assumption that the center of the camera coordinate system is the center of world coordinate system. But in real life, it not always the case. We define a rotation matrix R and a translation vector $\vec{\boldsymbol{t}}$ to denote the coordinate system transformations from 3D world coordinates to 3D camera coordinates. Let $\mathrm{P}(U, V, W)$ to be a point in the world coordinate and $\mathrm{P}(X, Y, Z)$ corresponding to coordinates in the camera coordinate system. Then,

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = R \cdot \begin{pmatrix} U \\ V \\ W \end{pmatrix} + \vec{\boldsymbol{t}} = \begin{pmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{pmatrix} \cdot \begin{pmatrix} U \\ V \\ W \end{pmatrix} + \cdot \begin{pmatrix} t_x \\ t_y \\ t_z \end{pmatrix} \tag{3.8}$$

The parameters R and $\vec{\boldsymbol{t}}$ are known as extrinsic parameters. They describe the camera's location and orientation in the world coordinate system. The extrinsic parameters are often written in a matrix form:

$$[R|\vec{\boldsymbol{t}}] = \left[\begin{array}{ccc|c} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \end{array}\right], \tag{3.9}$$

known as the extrinsic matrix.

3.3 Epipolar Geometry

3.3.1.1 Overview

When using two cameras or the same camera in two different locations to image the same scene, the two resulting pictures are related by what is known as Epipolar Geometry. This type of geometry is independent of scene structure, and only depends on the cameras' internal parameters and relative pose [25].

Suppose X is a point in 3D world coordinates, where X is projected onto the two views, as x in the first view, and as x' in the second view. 3D Point X and the camera centers C and C' form what is known as the epipolar plane π. Figure 3.6, shows that the rays back-projected from x and x' intersect at X, and are lying in the plane π.

If we only know x, the position of x' is impossible to determine (See Figure 3.7). The plane π is determined by the stereo baseline and the rays defined by x. The point x' lies on a line l', which is the image in the second view of the ray back-projected from x. This constraint is quite crucial to stereo correspondence, since the search for the point corresponding to x need not cover the entire image plane but can be restricted to the line l', known as an epipolar line.

We introduce the terminology of Epipolar geometry.

- Epipole: the epipole is the point of intersection of the line joining the camera centers (the baseline) with the image plane.
- Epipolar plane: an epipolar plane is a plane containing the baseline.

- Epipolar line: an epipolar line is the intersection of an epipolar plane with the image plane. All epipolar lines intersect at the epipole.

The geometric entities involved in Epipolar geometry are illustrated in Figure 3.8.

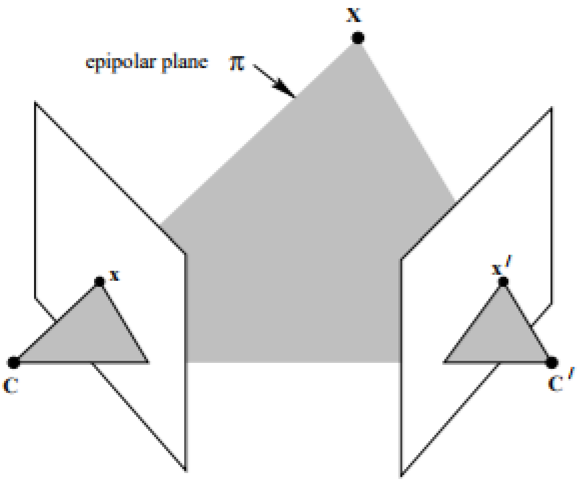


Figure 3.6 Points correspondence geometry (a)

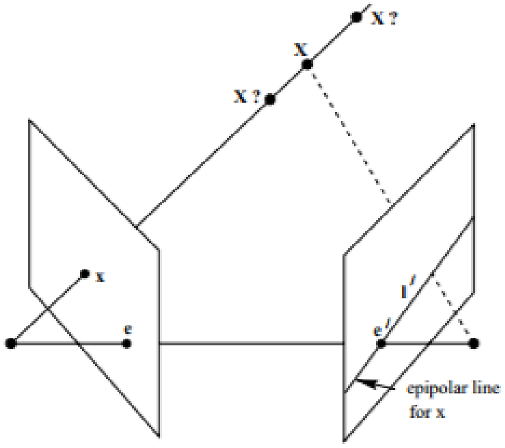


Figure 3.7 Points correspondence geometry (b)

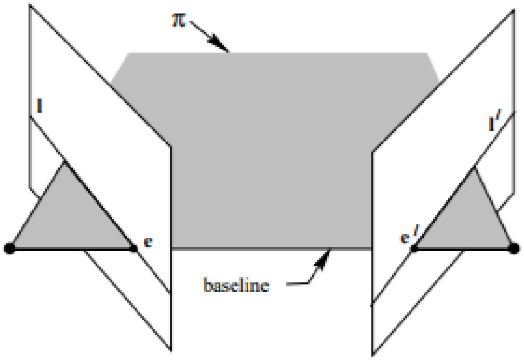


Figure 3.8 Illustration of Epipolar geometry

3.3.1.2 Fundamental and Essential Matrices

In 1981, Longuet-Higgins found that there is a special 3 × 3 matrix that associates two images from different perspectives of the same scene [26]. Then in the late 1980s and early 1990s, other scholars re-discovered this matrix in various fields. It came to be known as the Fundamental Matrix.

Let's consider two perspective images of a scene as taken from a stereo pair of cameras. (or equivalently, assume the scene is rigid and imaged with a single camera from two different locations) and suppose K_l and K_r are the intrinsic parameter matrices of these two cameras. Let P be a point in the world coordinate, and $P_l(P_{l_x}, P_{l_y}, P_{l_z})$ and $P_r(P_{r_x}, P_{r_y}, P_{r_z})$ be the coordinates of P in left and right camera coordinate system. The position and orientation of the two cameras are related by a rotation matrix R and a translation vector $\vec{\boldsymbol{t}} = (t_x, t_y, t_z)^T$ in the following way:

$$P_r = R \cdot P_l + \vec{\boldsymbol{t}} \tag{3.10}$$

Due to the epipolar constraint (see Figure 3.9) and outer product properties we can write:

$$P_r \cdot \left(\vec{\boldsymbol{t}} \times P_r\right) = 0 \tag{3.11}$$

Combining (3.10) and (3.11) yields:

$$P_r \cdot \left(\vec{\boldsymbol{t}} \times (R \cdot P_l + \vec{\boldsymbol{t}})\right) = 0 \tag{3.12}$$

Since, $\vec{\boldsymbol{t}} \times \vec{\boldsymbol{t}} = \mathbf{0}$, the equation is equivalent to:

$$P_r \cdot \vec{\boldsymbol{t}} \times R \cdot P_l = 0 \tag{3.13}$$

We have

$$\vec{\boldsymbol{t}} \times R = SR \tag{3.14}$$

where

$$S = \begin{bmatrix} 0 & -t_z & t_y \\ t_z & 0 & -t_x \\ -t_y & t_x & 0 \end{bmatrix},$$

is a skew-symmetric matrix of rank 2.

Thus, we write:

$$P_r^T \cdot SR \cdot P_l = 0\,. \tag{3.15}$$

Let $E = RS$:

$$P_r^T \cdot E \cdot P_l = 0. \tag{3.16}$$

The matrix E here called the Essential Matrix. The matrix E:

- has rank 2, and
- depends only on the extrinsic parameters ($\boldsymbol{R}$ and $\vec{\boldsymbol{t}}$).

Consider $x_l = (u_l, v_l)$ and $x_r = (u_r, v_r)$ as the projections of 3D point P onto the two image planes, with x_l and x_r defined as:

$$x_l = \begin{pmatrix} u_l \\ v_l \\ 1 \end{pmatrix} \quad , \quad x_r = \begin{pmatrix} u_r \\ v_r \\ 1 \end{pmatrix} \tag{3.17}$$

According to (3.4), we have

$$Z \cdot x_l = K_l X_l \, , \tag{3.18}$$

$$Z \cdot x_r = K_r X_r .$$

Combing (3.16) and (3.18)

$$Z^2 \cdot x_r^T \cdot K_r^{-T} \cdot E \cdot K_l^{-1} \cdot x_l = 0 \, , \tag{3.19}$$

which is equivalent to

$$x_r^T \cdot K_r^{-T} \cdot E \cdot K_l^{-1} \cdot x_l = 0 \tag{3.20}$$

The matrix $F = K_r^{-T} \cdot E \cdot K_l^{-1}$, is called Fundamental Matrix and expresses epipolar geometry in image coordinates (see Figure 3.9).

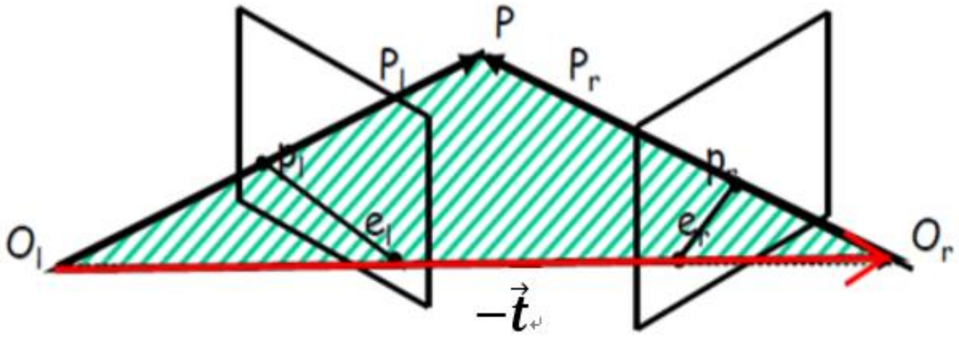


Figure 3.9 Epipolar constraint

3.4 Feature Detection

3.4.1 SIFT Feature

The Scale Invariant Feature Transform (SIFT) was first proposed by David Lowe in 1999 and improved in 2004 to identify points of interest in an image [2], [3]. The SIFT approach, for image feature generation, takes an image and transforms it into a "large collection of local feature vectors", with each of these feature vectors invariant to scaling, rotation or translation of the image.

The SIFT algorithm consists of a 4-stage filtering approach:

- Scale-space peak detection

This stage of the filtering attempts to identify those locations and scales that are identifiable from different views of the same object. The approach to achieve this is building a scale space by using Laplacian of Gaussian (LoG). The scale space is defined by the function:

$$L(x, y, \sigma_D) = G(x, y, \sigma_D) * I(x, y) \,,$$

where $I(x, y)$ represents the original image, $*$ is the convolution operator, and G is a Gaussian kernel. Figure (3.10) shows an input image to which a Gaussian kernel is applied in a successive manner.

There are various techniques to detect stable key point locations in scale-space. One of them is the Difference of Gaussians technique: locating scale-space peak $D(x, y, \sigma_D)$ by computing the difference between two images, one with scale k times the other. $D(x, y, \sigma_D)$ is then given by:

$$D(x, y, \sigma_D) = L(x, y, k\sigma_D) - L(x, y, \sigma_D)$$

To detect the local maxima and minima of $D(x, y, \sigma_D)$, each point is compared with its 8 neighbors at the same scale, and its 9 neighbors up and down one scale. If this value is the minimum or maximum of all these points then this point is an extremum.

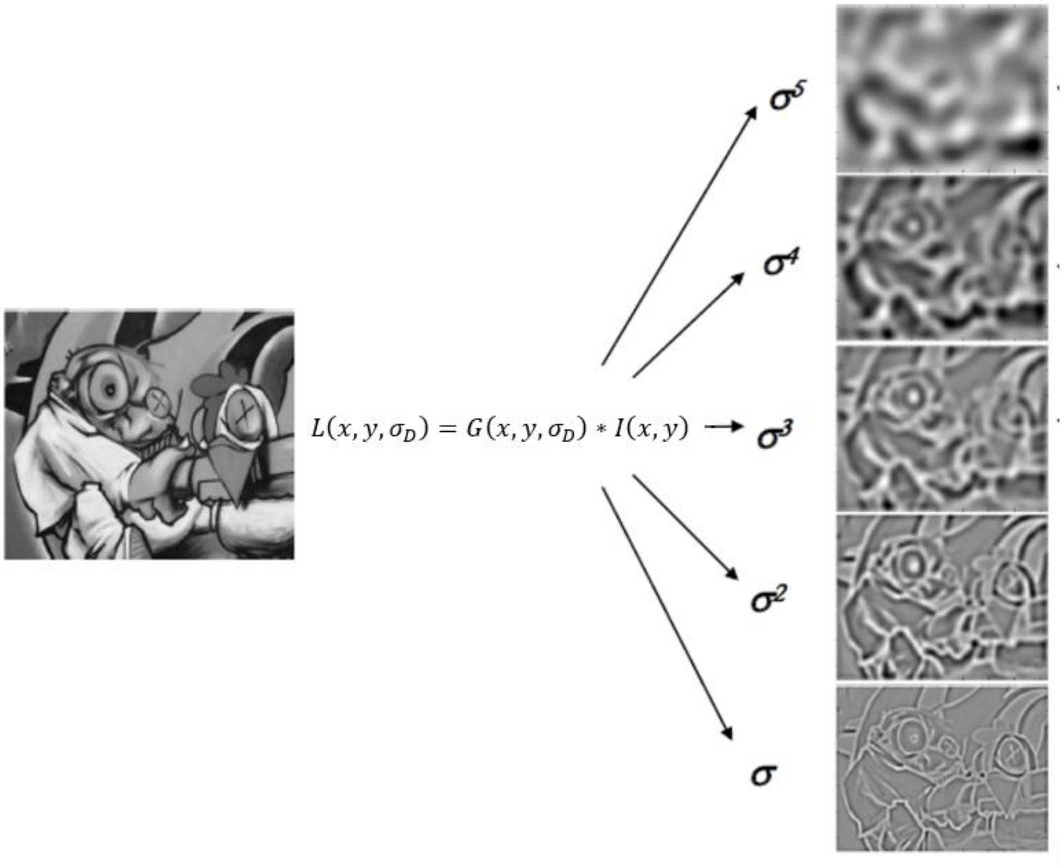


Figure 3.10 Input image applied with different Gaussian kernel. Applied with different scales factor, input image shows different details.

- Key point localization

The purpose of this step is to pinpoint the location of the feature key. To reach this goal, the Laplacian value for each key point found in stage 1 is calculated. The location of extremum, z is given by:

$$\mathrm{z} = -\frac{\partial^2 D^{-1}}{\partial X^2}\frac{\partial D}{\partial X},$$

If the function value at z is below a threshold value then this point is excluded. By doing this, low contrast extrema and poorly localized candidates are removed. It is noted for difference of Gaussian function that there is a large principle curvature across the edge but a small curvature in the perpendicular direction. If this difference is below the ratio of largest to smallest eigenvector, from the 2x2 Hessian matrix at the location and scale of the key point, the key point is rejected.

- Orientation assignment

This is the third step of SIFT. The purpose is to achieve the orientation invariance of the SIFT feature. The approach taken to find an orientation is:

- Use the key points scale to select the Gaussian smoothed image L
- Compute gradient magnitude m and orientation θ

$$m(x,y) = \sqrt{(L(x+1,y) - L(x-1,y))^2 + (L(x,y+1) - L(x,y-1))^2}\ ,$$

$$\theta(x,y) = \tan^{-1}((L(x,y+1) - L(x,y-1))/(L(x+1,y) - L(x-1,y)))$$

- For each key point, we select the 16 x 16 neighborhood and quantify the gradient of all pixels (256) in this window into the histogram of 36 bin.
- Locate the highest peak in the histogram. Use this peak and any other local peak within 80% of the height of this peak to create a key point with that orientation

- Key point descriptor

Once Completed the above three steps of SIFT, then we need to find a Local Image Descriptors at key points. Here we chose to use the gradient direction histogram to describe this key point. Key point descriptors typically use a set of 16 histograms, aligned in a 4x4 grid, each with 8 orientation bins, one for each of the main compass directions and one for each of the mid-points of these directions. This results in a feature vector in 128-dimensions, (see Figure 3.11).

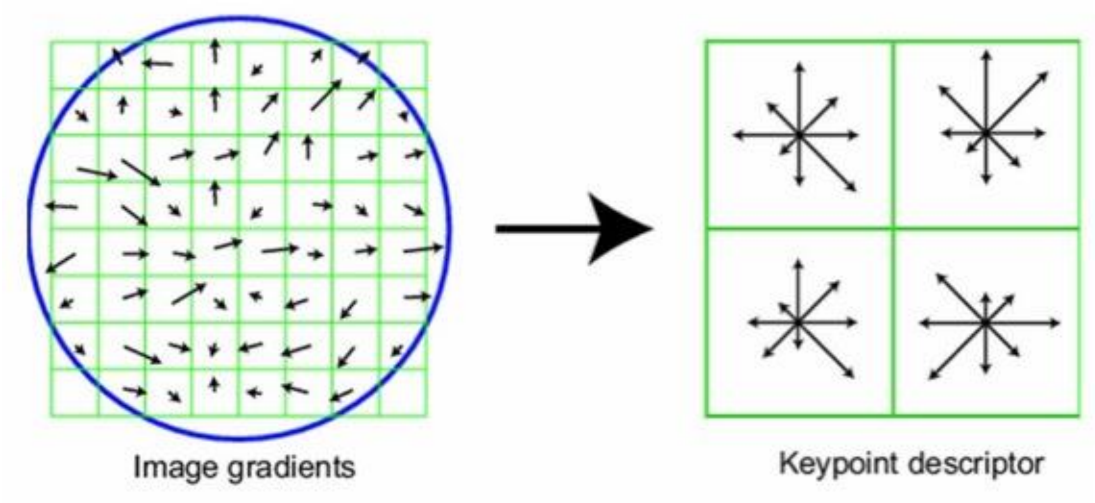


Figure 3.11 SIFT key point descriptor

3.4.2 SURF Feature

In computer vision, speeded up robust features (SURF) is a patented local feature detector and descriptor. This algorithm is based on the same principles and steps of SIFT, but it utilizes a different scheme [4].

- Hessian Matrix-based interest points

SURF uses a Hessian based blob detector [27] to find interest points. The determinant of a Hessian matrix expresses the extent of the response and is an expression of the local change around the area.

A Hessian Matrix is defined as:

$$H(x,\sigma) = \begin{bmatrix} L_{xx}(x,\sigma) & L_{xy}(x,\sigma) \\ L_{xy}(x,\sigma) & L_{yy}(x,\sigma) \end{bmatrix},$$

where

$$L_{xx}(x,\sigma) = I(x) * \frac{\partial^2}{\partial x^2} g(\sigma)$$

$$L_{xy}(x,\sigma) = I(x) * \frac{\partial^2}{\partial xy} g(\sigma)$$

The core of SURF detection is non-maximal-suppression of the determinants of the Hessian matrices. However, to calculate the convolutions is a time-consuming process. To speed it up, integral images and approximated kernels are being used.

An Integral image $I(x)$ is an image where each point $p(x,y)^T$stores the sum of all pixels in a rectangular area between center $O(0,0)^T$and p.

$$I(x) = \sum_{i=0}^{i<x} \sum_{j=0}^{j<y} I(x,y)$$

- Scale-space representation and interest point localization

As in the first stage of SIFT, interest points can be found at different scales, partly because the search for correspondences often requires the comparison of images where these points are seen at different scales. In the stages of building a scale pyramid, the SURF and SIFT use different strategies. For SIFT, the Gaussian filter size is kept unchanged, but the image size changes at different scales. The SURF technique is opposite, the image size remains unchanged but the filter size changes (see Figure 3.12).

In order to localize interest points in the image and over scales, a non-maximum suppression in a $3 \times 3 \times 3$ neighborhood is applied.

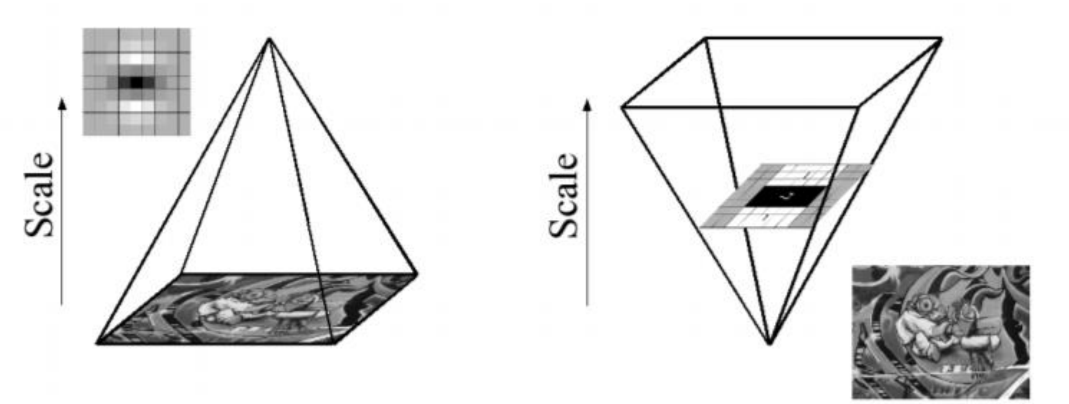


Figure 3.12 Differences between SIFT (left side) and SURF (right side) when constructing a scale space.

- Orientation assignment

In order to be invariant to image rotation, SURF builds the distribution of first-order Haar wavelet responses in the x and y direction for those interest points.

The dominant orientation is estimated by calculating the sum of all responses within a sliding orientation window of size $\pi/_3$, as in Figure 3.13.

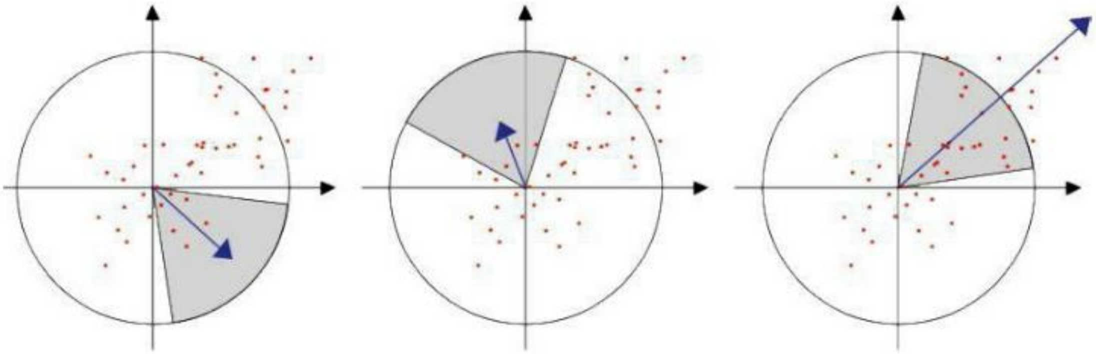


Figure 3.13 Orientation assignment of SURF. The right region has a strongest Haar Wavelet response, therefore such orientation serves as the dominant orientation for the interst point.

- Descriptor based on sum of Haar wavelet responses

For the extraction of the descriptor, the first step consists of constructing a square region centered around the interest point and oriented along the dominant orientation. The size of this window is 20*s*, where *s* is the scale at which the interest point was detected. The region is then split up regularly into smaller 4×4 square sub-regions. Then, the wavelet responses d_x, d_y,$|d_x|$ and $|d_y|$ are summed up over each sub-region and form a feature vector for each interest point, as in Figure 3.14.

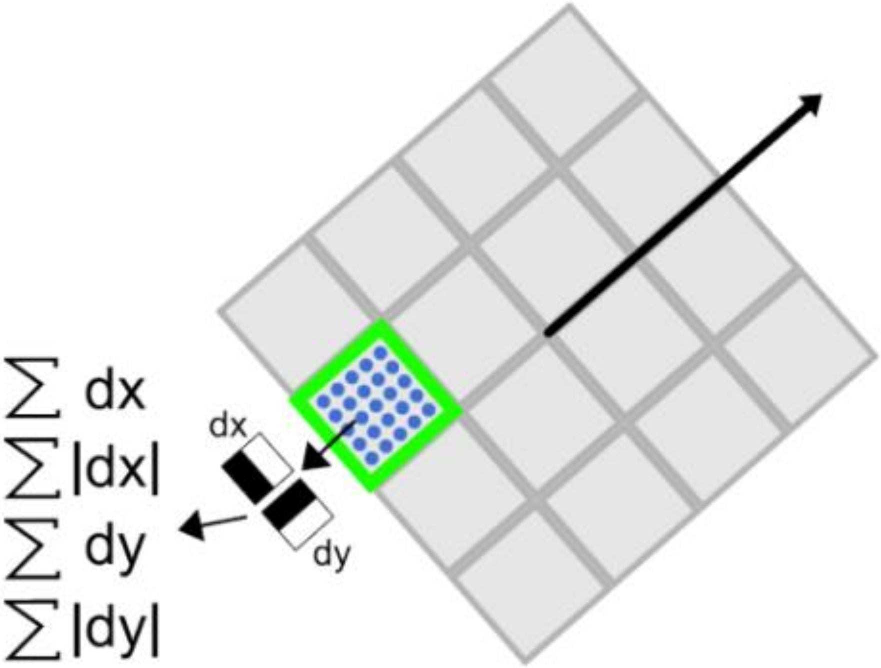


Figure 3.14 Feature descriptor for SURF

3.5 Correspondence

3.5.1 Overview

Given two or more images of the same 3D scene, taken from different points of view, the correspondence problem refers to the task of finding a set of points in one image which can be identified as the same points in another image.

There are two basic ways to achieve this:

- Correlation-based methods

The idea of the correlation is to check if one part in one image looks like another in another image. One simple method is to compare small patches between rectified images: A filter window is passed over a number of positions in one image to check how well it compares with the same location as well as several nearby locations in the other image.

- Feature-based methods

This kind of methods first finds features in one image and then associates them with features from a second image that are most similar. As mentioned in Section 3.4, SIFT, SURF and other feature point detection algorithms could be used to reach this goal. For example, rough correspondence could be found by just comparing two key points descriptor’s Euclidean distance.

3.5.2 Random Sample Consensus (RANSAC)

The RANSAC algorithm performs a robust fitting of models in the presence of many data outliers [27], [28],[29]. Given a dataset whose data elements contain both inliers and outliers, RANSAC uses the voting scheme to find the optimal fitting result. The implementation of this voting scheme is based on two assumptions:

- Few Outliers: the noisy features will not vote consistently for any single model.

- Few Missing Data: there are enough features to agree on a good model.

RANSAC algorithm is often used in computer vision. To be specific, it could be used to simultaneously solve the correspondence problem and estimate the fundamental matrix related to a pair of stereo cameras.

This algorithm works as follows:

```
Algorithm:

Input: source RGB-D image $I_l$ and $I_r$, threshold $\varepsilon$, Maximum iteration $\mu$

Output: high-quality correspondences set $S$ and Essential Matrix $E$

Step 1: Calculate Feature points set $F_l$ and $F_r$ from source image. (e.g. using SURF feature.)

Step 2: Calculate correspondence set $C$ from $F_l$ and $F_r$ by using feature based correspondence methods.

Step 3: Initialize $N = 0$.

Step 4: Randomly select a set $S_t$ , which contains 8 correspondences, from $C$ and compute $E_t$ with 8 points algorithms and initialize $N_t$=0.

Step 5: for correspondence $p \leftrightarrow q$ in $C$, compute $pE_tq$.

        if $pE_tq < \varepsilon$

        then $N_t = N_t + 1$

Step 6: if $N_t > N$

        then $N = N_t, S = S_t, E = E_t$.

Step 7: Repeat Step 4 ~ Step 6 $\mu$ times.
```

Step 8: output set S and Essential Matrix E.

3.6 Camera pose estimation Algorithm

3.6.1 Eight-Points Algorithm (2D-2D)

3.6.1.1 Computing the Fundamental Matrix

In Section 3.1.1.2, we introduced the Fundamental Matrix

$$x_r^T \cdot F \cdot x_l = 0$$

where x_r and x_l are projections of the same 3D point in two different image planes.

Given any pair of correspondence $p \leftrightarrow q$ where $p = (u, v, 1)^T$ and $q = (u', v', 1)^T$, the equation relating points p and q to the Fundamental Matrix F is:

$$uu'f_{11} + uv'f_{21} + uf_{31} + vu'f_{12} + vv'f_{22} + vf_{32} + u'f_{13} + v'f_{23} + f_{33} = 0\,.$$

Each correspondence gives rises to one linear equation in the unknown entries of F. Therefore at least 8 correspondences are needed to estimate matrix F from a homogenous set of linear equations:

$$\begin{bmatrix} u_1u_1' & u_1v_1' & u_1 & v_1u_1' & v_1v_1' & v_1 & u_1' & v_1' & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ u_8u_8' & u_8v_8' & u_8 & v_8u_8' & v_8v_8' & v_8 & u_8' & v_8' & 1 \end{bmatrix} \vec{f} = 0\,. \tag{3.21}$$

Where $\vec{f} = (f_{11}, f_{21}, f_{31}, f_{12}, f_{22}, f_{32}, f_{13}, f_{23}, f_{33})^T$,

Writing (3.21) as

$$\boldsymbol{A}\vec{f} = 0 \tag{3.22}$$

we are seeking $\vec{f}$ that minimizes $\|\boldsymbol{A}\vec{f}\|$ subject to the constraint $\|\vec{f}\| = \vec{f}^T \cdot \vec{f} = 1$. This represents a least-squares estimation of the Fundamental Matrix.

3.6.1.2 Extracting Camera Pose from the Essential Matrix

Once F is estimated, we can compute the Essential Matrix with the following equation:

$$E = K_r^T \cdot F \cdot K_l$$

We also have

$$E = \vec{\boldsymbol{t}} \times R = SR\,, \tag{3.23}$$

where R and $\vec{\boldsymbol{t}}$ are the rotation matrix and the translation vector describing the positions of the two cameras, relative to each other.

E has the following properties,

- rank 2
- $\det(E) = 0$
- its two non-zero singular values are equal.

Since the scale is arbitrary, we set a constraint stipulating $\|\vec{\boldsymbol{t}}\| = 1$.

E has a special form of Singular Value Decomposition (SVD):

$$E = U \cdot diag(1{,}1{,}0) \cdot V^T \tag{3.24}$$

We introduce two auxiliary matrices:

$$W = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad \text{and} \quad Z = \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

where the matrix W is a rotation and Z is skew symmetric. Furthermore, for these matrices we have:

$$ZW = \mathrm{diag}(1{,}1{,}0)\ , \tag{3.25}$$

$$ZW^T = -\mathrm{diag}(1{,}1{,}0)$$

Using (3.23) and (3.24), we have two solutions that fit $E = SR$, which are

$$S_1 = -UZU^T\ ,\ \ R_1 = UW^TV^T\ , \tag{3.26}$$

$$S_2 = UZU^T\ \ ,\ \ R_2 = UWV^T\ .$$

It’s easy to validate that both R_1 and R_2 are rotations and $S_1 = S_2$. We can extract $\vec{\boldsymbol{t}}$ from S_1(or S_2) such that $\vec{\boldsymbol{t}} = \pm(s_{32}, s_{13}, s_{21})^T$.

Therefore, we have four possible solutions of R and $\vec{\boldsymbol{t}}$, which are $(R_1, \vec{\boldsymbol{t}})$, $(R_2, \vec{\boldsymbol{t}})$, $(R_1, -\vec{\boldsymbol{t}})$, $(R_2, -\vec{\boldsymbol{t}})$, and only one of them is the right solution. Since $\vec{\boldsymbol{t}}$ is normalized to $\|\vec{\boldsymbol{t}}\| = 1$, the actual translation vector between the two cameras is determined up to a scale factor: $T = \lambda \cdot \vec{\boldsymbol{t}}$. To solve for the exact camera pose, we need to use 3D point depth information. Only one of these solutions puts the scene points in front of both cameras. So the correct solution can be identified by computing structure for all four cases by triangulation, and choosing the one solution that enforces most of the structure solution (allowing for a few reconstruction errors) to be in front of both cameras. In Figure 3.15, only solution 2 has positive depths, thus solution 2 is the correct one.

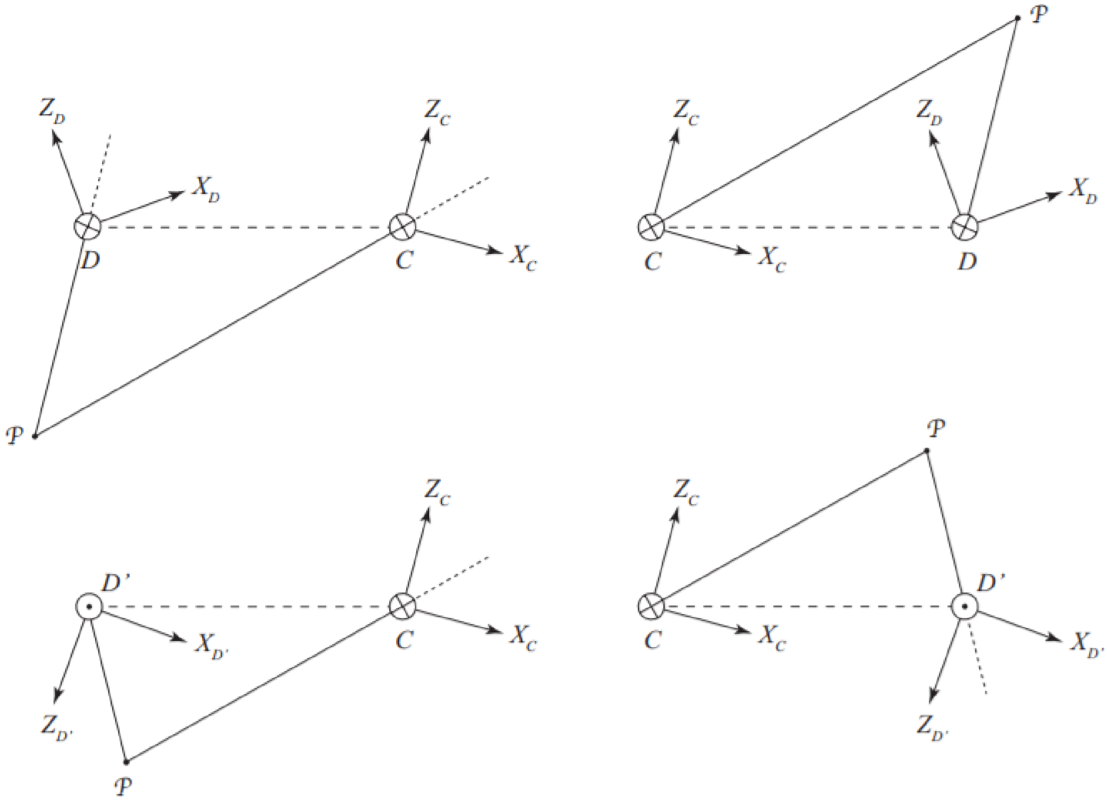


Figure 3.15 Four solution form decompose Essential Matrix. The correct solution(right top one) makes the world points P has a positive depth in both camera coordinate system.

3.6.2 Perspective-n-Point Algorithm (3D-2D)

Perspective-n-Point (PnP) [30] is the problem of estimating the pose of an intrinsically calibrated camera given a set of n 3D points in the world and their corresponding 2D projections in the image of the camera. Suppose we have $\boldsymbol{n}$ 3D points in the real scene and their corresponding 2D image projections. Assuming the intrinsic matrix of the camera is K, the perspective projection model for cameras can be described as follows:

$$sP_c = K\left[R\middle|\vec{\boldsymbol{t}}\right]P_w \tag{3.27}$$

where $P_w = (x, y, z, 1)^T$is the homogenous world point; $P_c = (u, v, 1)^T$ is the corresponding homogenous image point, s is a scale factor for the image point, R and $\vec{\boldsymbol{t}}$ are the desired rotation matrix and translation vector.

This equation also has an equivalent matrix form:

$$s\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix}\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}. \tag{3.28}$$

Efficient PnP (EPnP) which was developed by Lepetit [5], is a common method for solving the general problem of PnP for n ≥ 3.

They introduce the concept of virtual control points, and design the algorithm based on the notion that each of the n points, also known as reference points, can be expressed as a weighted sum of 4 non-coplanar virtual control points (only 3 for planar configurations).

Let those n points whose 3D coordinates are known in the camera coordinate system be $p_{\mathrm{i}}^{\mathrm{c}}, i = 1, \cdots, n$ and assume 4 virtual control points expressed as $c_j, j = 1, \cdots, 4$.

Then we can express each reference point as a weighted sum of control points:

$$p_i^c = \sum_{j=1}^{4} a_{ij} c_j^c \text{ ,with } \sum_{j=1}^{4} a_{ij} = 1 \tag{3.29}$$

Combining with (3.28), we obtain:

$$\forall i, \; s_i \begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \sum_{j=1}^{4} a_{ij} \begin{bmatrix} x_j^c \\ y_j^c \\ z_j^c \end{bmatrix} \tag{3.30}$$

Substituting this expression in the first two rows yields two linear equations for each reference point:

$$\sum_{j=1}^{4} a_{ij} f_x x_j^c + a_{ij}(c_x - u_i) z_j^c = 0 \tag{3.31}$$

$$\sum_{j=1}^{4} a_{ij} f_x y_j^c + a_{ij}\left(c_y - v_i\right) z_j^c = 0$$

Note that projective parameter s_i disappear in those equations. Grouping all references points, we generate a linear system:

$$\mathrm{M}x = 0 \tag{3.32}$$

Where $x = [c_1^{c^T}, c_2^{c^T}, c_3^{c^T} c_4^{c^T}]^T$ is a vector in R^{12} (the coordinates of those control points are unknown), and M is a $2\mathrm{n} \times 12$ matrix generated by arranging the coefficients of (3.31) for each reference point.

The solution is to the null space of M, which can be expressed as:

$$x = \sum_{i=1}^{N} \beta_i v_i \tag{3.33}$$

The solution is found efficiently by solving the null eigenvectors of matrix $M^T M$. Depending on n, the proper linear combination coefficients $\{\beta_i\}_{i=1,\cdots 4}$ is calculated.

In order to have an optimized result, those four coefficients $\beta = [\beta_1, \beta_2, \beta_3, \beta_4]$ are refined by choosing the values that minimize the change in distance between control points, using Gauss-Newton Optimization:

$$\mathrm{Error}(\beta) = \sum_{(i,j)\ s.t.\ i<j} \left(\left\| c_i^c - c_j^c \right\|^2 - \left\| c_i^w - c_j^w \right\|^2 \right) \tag{3.34}$$

where c_i^w are 3D world coordinates.

After determining the best β, x is solved. Therefore, the pose transformation between the world coordinates system and the camera coordinate system is easily formed.

This algorithm is found in the open source library OpenCV (routine name: solvePnP). Also, RANSAC can be used to deal with outliers in the data set to optimize the result (routine name, solvePnPRansc) [31].

3.6.3 Point Set Registration (3D-3D)

3D Registration is a method to seek for the optimal rotation and translation between two sets of corresponding 3D point data, that makes them well registered [32]. Suppose we have two sets of 3D points, dataset A and B. This method has three main steps:

- Finding the centroids

This is quite straightforward: the centroids are the average point and can be calculated as follows:

$$centroid_A = \frac{1}{N} \sum_{i=1}^{N} P_A^i \, , \tag{3.35}$$

$$centroid_B = \frac{1}{N} \sum_{i=1}^{N} P_B^i$$

Here, P_A and P_B are points in dataset A and B respectively.

- Finding the optimal rotation

To find the optimal rotation, we need to re-center the datasets so that their centroids are at the origin. This is shown in Figure 3.16.

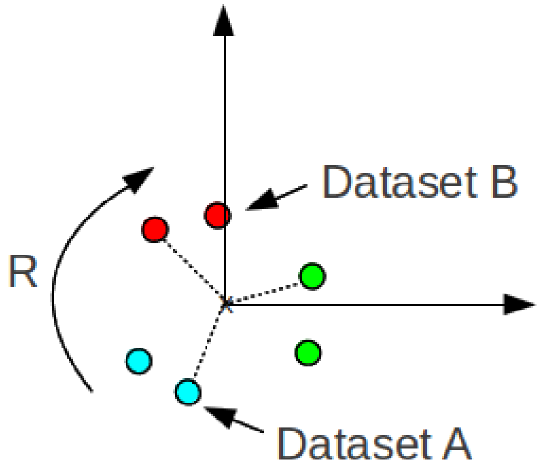


Figure 3.16 Re-center dataset

The next step involves the computation of a covariance matrix H, and using SVD to find the rotation as follows:

$$H = \sum_{i=1}^{N}(P_A^i - centroid_A)(P_B^i - centroid_B)^T \qquad (3.36)$$

$$[U, S, V^T] = SVD(H) \qquad (3.37)$$

The optimal rotation can be calculated as

$$R = V^T U^T \qquad (3.38)$$

Note:

R here has to be a rotation matrix, which means $\|R\| = 1$. We need to validate this property in case the SVD returns a reflection matrix, which is numerically correct but is nonsense for real cases. If *R* is the reflection matrix, then $\|R\| = -1$, and we multiply the third column of *R* by -1 as a remedy.

- Finding the translation vector

Since we have the optimal *R*, the translation vector $\vec{\boldsymbol{t}}$ is obtained with:

$$\vec{\boldsymbol{t}} = -R \cdot centroid_A + centroid_B \tag{3.39}$$

Chapter 4

4 Methodology

This thesis proposes to evaluate the algorithms described in the last Section to calibrate for the extrinsic parameters of two Kinect cameras. In this Section, we follow with appropriate justifications for our approach, and what the advantages are for each chosen method or process. Also, this Section justifies the reasons for our choices in the experiments that follow in the next Chapter.

4.1 Feature Selection

In section 3.4, we presented the SIFT and SURF algorithms. They are quite similar in the generating phase. In our case, SIFT has detected larger numbers of features compared to SURF but with slower speed. For its high speed and relatively high-quality performance, we choose SURF as our feature point detection approach [33].

4.2 Correspondence Method Selection

In the selection of correspondence methods, feature-based methods provide relatively accurate information about local regions of interest and thus achieve better matching compared to correlation-based methods, which are also computationally expensive. Feature-based methods work best with images taken with roughly the same point of view and either at the same time or with little to no movement of the scene between image captures, such as stereo images.

4.3 Experiments with Real RGB-D Images

Normally, camera pose estimation techniques are aimed at calibrating multiple cameras in different positions. However, one equivalent way to validate a calibration algorithm is to keep the scene unchanged and capture images from different positions with one camera. By doing so, we only need to calibrate intrinsic parameters once, with no synchronization issues. The experiments in this thesis are based on the second method.

4.3.1 Case 1

For this case, we manually set the pose of the Kinect camera and measure the distance of the two successive locations used to take images. Since the rotation is hard to physically measure, we did not rotate the camera. The objective for this case is to prove that the camera pose obtained by the algorithm is close to the physically measured distance.

4.3.2 Case 2

In this case, we investigate how our algorithm performs in random cases. So, the camera position is chosen with no particular preference, and the square error of the computed 3D correspondence sets is calculated to estimate the correctness of the results. Note that the choice of camera positions is constrained by the fact that there must be a scene overlap in the two images in order to obtain valid point correspondences.

4.4 Robustness Analysis

We are interested in finding out the behavior of the algorithms when confronted to noise. In order to compare the robustness of camera pose estimation algorithms, we used a synthetic set of 3D points, with a known rotation and translation to also generate the second 3D set. Noise is then added, and we compare the so obtained translation and rotation to the original one. We proceed to list the specifics of our robustness analysis.

- Gaussian Noise

Gaussian noise is a random form of noise with a probability density function equal to that of the normal distribution. In the field of Computer Vision, Gaussian noise is widely used since it is regarded as the most realistic simulation of noise for most circumstances.

- Synthetic Data

The goal of the robustness experiments is to analyze how algorithms behave when noise is present. To achieve this goal, we tested the algorithms without the use of the cameras, to avoid any unwanted noise such as that introduced when finding image correspondences. The use of synthetic 3D points sets and camera position and orientation reflects robustness in an unbiased fashion and allows us to know the real total amount of noise that is being introduced.

- Offsets of R and $\vec{t}$

The rotation matrix is completely specified by three angles about the *x*-, *y*-, and *z*- axes, which we denote as θ_x, θ_y and θ_z. We use $\theta_0 = (\theta_{x_0}, \theta_{y_0}, \theta_{z_0})^T$ and $t_0 = (t_{x_0}, t_{y_0}, t_{z_0})^T$

as the ground truth for the rotation angle vector and the translation vector, and $\theta = (\theta_x, \theta_y, \theta_z)^T$ and $t = (t_x, t_y, t_z)^T$as the results obtained from the algorithms under study.

To measure the system behavior towards noise, the offsets of R and $\vec{\boldsymbol{t}}$ is defined as follow:

$$\text{Offset\_R} = \sqrt{(\theta - \theta_0)^T \cdot (\theta - \theta_0)}$$

$$\text{Offset\_T} = \sqrt{(t - t_0)^T \cdot (t - t_0)}$$

which calculate the Euclidian distance between two points in R^3 space.

Chapter 5

5 Experiments and Results

To properly identify the correctness and the robustness of the algorithms, a series of experiments are performed in the indoor lab scene. The input for the first experiment are RGB-D images obtained from our Kinect sensor. This experiment aims to prove that our work is not only numerically correct but also well suited for general cases. The second experiment focuses on the robustness of the chosen camera pose estimation algorithms. Various levels of Gaussian noise are added to the synthetic data and the distance between the synthetic pose parameters and the output of each algorithm is calculated.

Figure 5.1 shows the type of RGB-D images used for the experiments. The left image is the RGB image from the color camera, which is resized to 512×424 (original resolution 1920×1080). The right image is the depth image-capture from the IR sensor, which also has a resolution of 512×424. The red points inside the image indicate invalid depth values. The intensity of the depth value is related to object distance: nearer surfaces are darker while further surfaces are lighter.

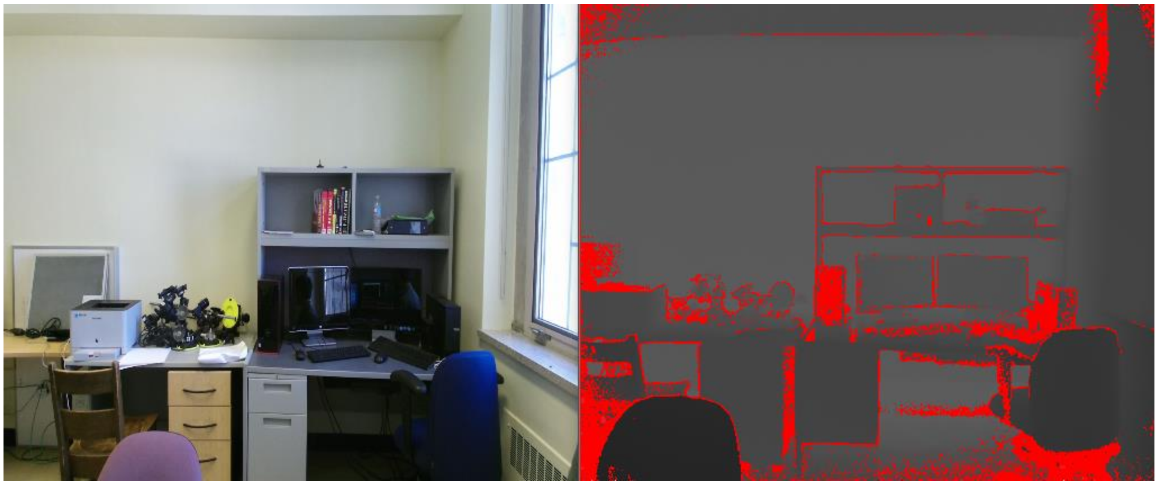


Figure 5.1 RGB-D image structure. The red dots indicate invalid depth value. RGB image and Depth image has slightly difference if you look carefully (only demonstrate the RGB-D image structure), they do have intrinsic mapping function from each RGB pixel to depth pixel.

5.1 Results with Real RGB-D Data

5.1.1 Case 1

In this case, the Kinect sensor first captured an indoor scene at one location and was then relocated along the x axis of the camera coordinate system by approximately 3 inches (76.2 mm). These two RGB-D images only contain a slight perspective difference, which makes feature points easy to match correctly. The SURF features (colored dots) from the image obtained at the second camera position are shown in Figure 5.2 and the optimized correspondences between the two RGB-D image sources are displayed in Figure 5.3.

Table 2 gives the rotation matrix and translation vector obtained from each specific algorithm (4 digits after the decimal point are retained). We conclude from the results that, in this case, all the algorithms give a nearly perfect solution for the rotation matrix, which is very close to identity matrix. However, for the translation vector, the Perspective-n-points algorithm and the 3D point registration algorithm both perform better than the Eight-point algorithm. These results support the correctness of the implementations. Moreover, it also indicates that the intrinsic parameters of the camera are well measured so that the points in the image plane could be adequately re-projected as 3D coordinates.

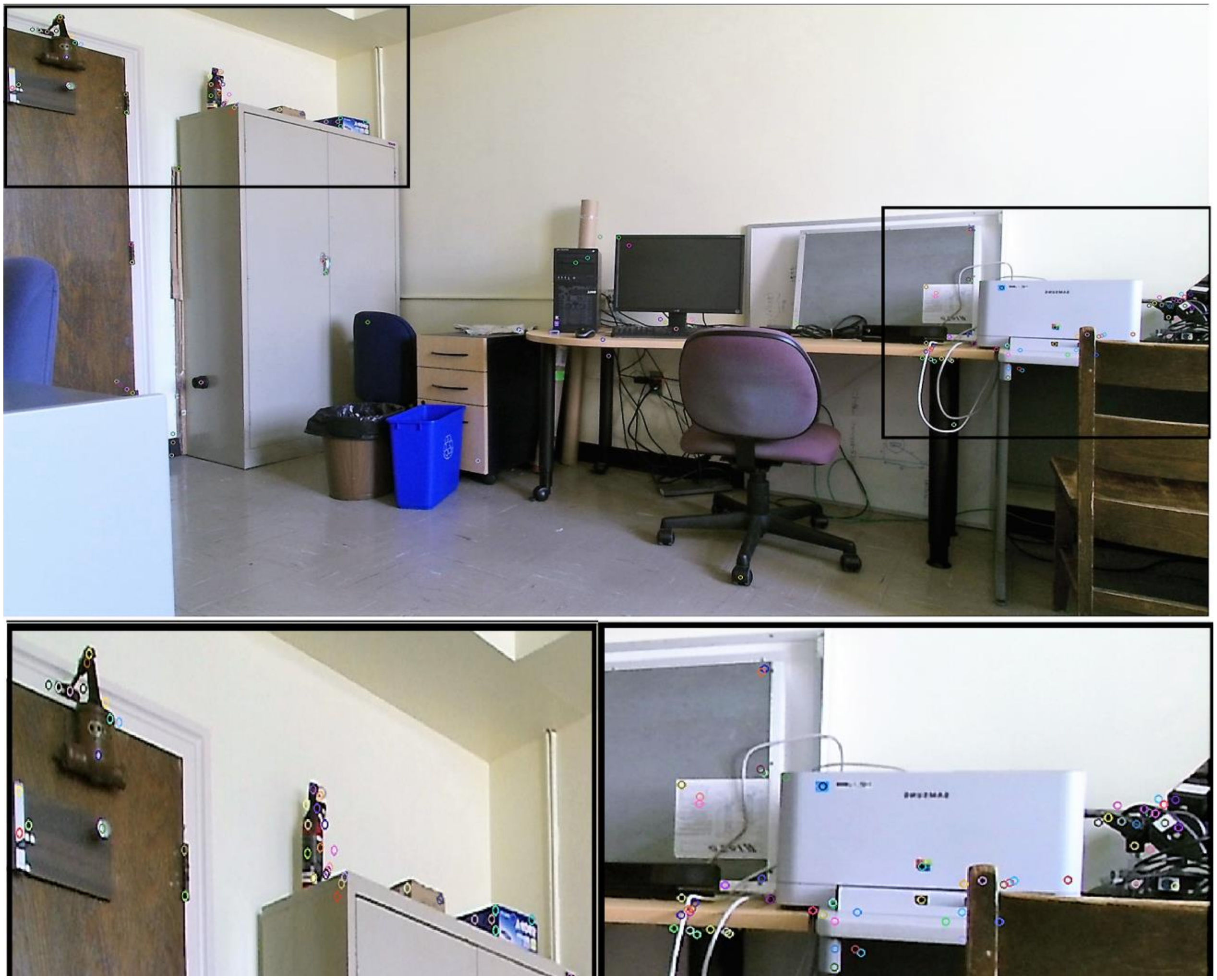


Figure 5.2 Case 1: Feature Points. The bottom images zoom in the black region of the top one. Those colorful dots are SURF feature points detected from feature detection phases.

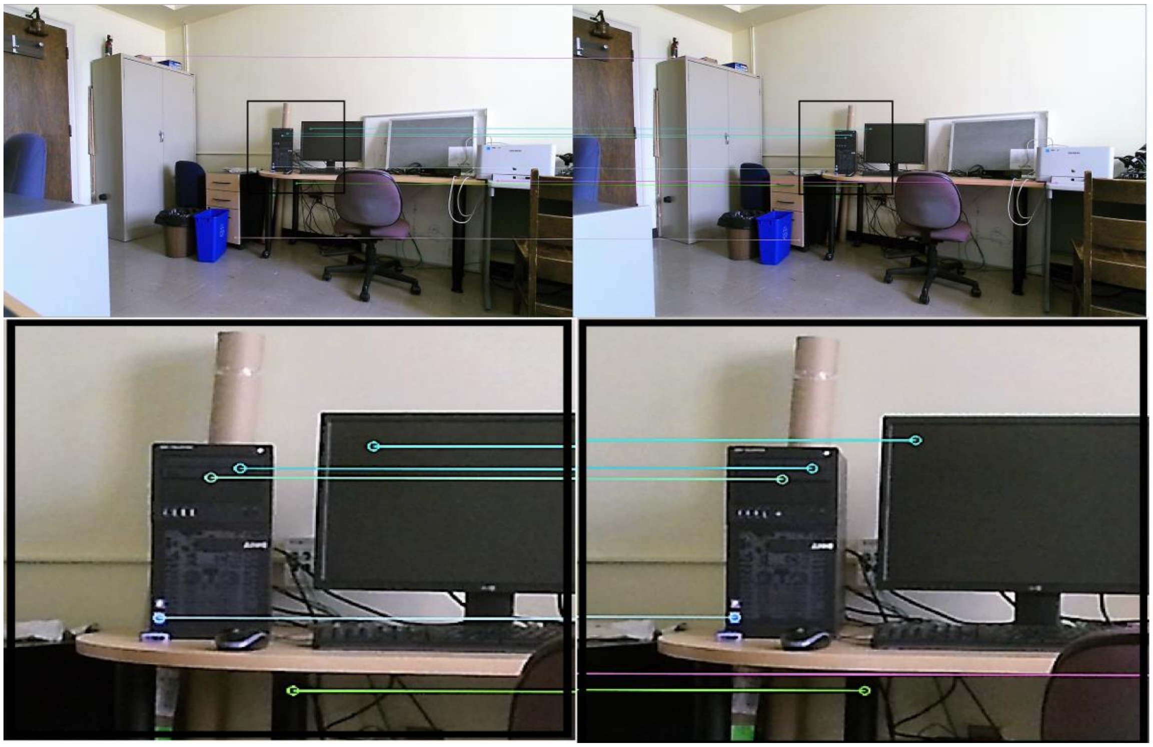


Figure 5.3 Case 1: Correspondences. The bottom images zoom in the black region of the top one, which shows the details of matching correspondence.

Table 2 Comparison of results from different camera pose estimation algorithms

Algorithms	Rotation Matrix	Translation Vector
Eight-Point Algorithm	$\begin{bmatrix} 0.9992 & -0.0050 & 0.0398 \\ 0.0053 & 0.9999 & 0.0074 \\ -0.0398 & 0.0076 & 0.9992 \end{bmatrix}$	$\begin{bmatrix} -0.0908 \\ 0.0215 \\ -0.0083 \end{bmatrix}$
Perspective-n-Point Algorithm	$\begin{bmatrix} 1.0000 & 0.0004 & -0.0008 \\ -0.0004 & 0.9999 & 0.0059 \\ 0.0008 & -0.0059 & 1.0000 \end{bmatrix}$	$\begin{bmatrix} -0.0804 \\ 0.0022 \\ -0.0018 \end{bmatrix}$

3D Point Registration	$\begin{bmatrix} 0.9999 & -0.0005 & -0.0028 \\ 0.0005 & 0.9999 & 0.0076 \\ 0.0028 & -0.0076 & 0.9999 \end{bmatrix}$	$\begin{bmatrix} -0.0734 \\ -0.0015 \\ -0.0030 \end{bmatrix}$
Pre-given Ground Truth	$\begin{bmatrix} 1.0000 & 0.0000 & 0.0000 \\ 0.0000 & 1.0000 & 0.0000 \\ 0.0000 & 0.0000 & 1.0000 \end{bmatrix}$	$\begin{bmatrix} -0.7620 \\ 0.0000 \\ 0.0000 \end{bmatrix}$

5.1.2 Case 2

In this case, the Kinect senor is placed at two random locations where the two perspectives have a visual overlap over the scene. Figures 5.4 and 5.5 show the feature points and their correspondences separately.

Comparative results from the camera pose estimation algorithms are displayed in Table 3. Since we no longer have synthetic pose data (rotation and translation), the average squared error is recorded to measure the performance of each algorithm. The average squared error is calculated in the following way: For all the matching pairs in the correspondence set, we use the rotation matrix and the translation vector produced by each algorithm to re-project the 3D matching points in the left image into a new 3D point set and then we average the squared error of the re-projection and with the right image matching points.

From Table 3 we conclude that the Perspective-n-points algorithm and the 3D point registration algorithm both have the same level of average squared error, under 0.005.

The result from the Eight-Point algorithm is significantly higher, which is around 0.0775. However, this error rate is still acceptable.

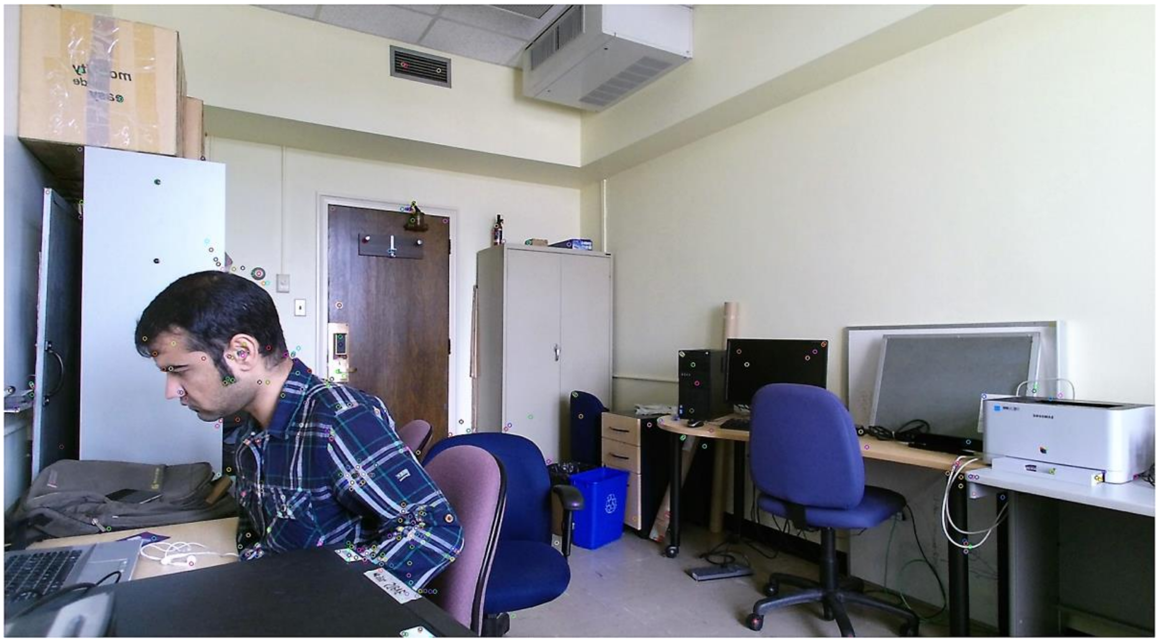


Figure 5.4 Case 2: Feature Points

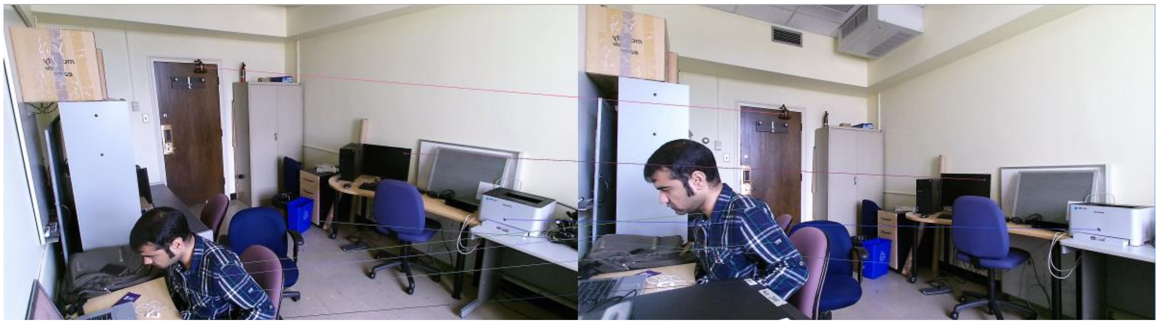


Figure 5.5 Case 2: Correspondences

Table 3 Comparative results from different camera pose estimation algorithms

Algorithms	Rotation Matrix	Transit Vector	Average Squared Error
Eight-Points Algorithm	$\begin{bmatrix} 0.9983 & -0.0237 & 0.0520 \\ 0.0097 & 0.9967 & 0.2538 \\ -0.0563 & -0.2528 & 0.9659 \end{bmatrix}$	$\begin{bmatrix} -0.1508 \\ -0.6120 \\ -0.0477 \end{bmatrix}$	0.07746700
Perspective-n-Point Algorithm	$\begin{bmatrix} 0.9985 & -0.0186 & 0.0506 \\ 0.0053 & 0.9679 & 0.2512 \\ -0.0537 & -0.2506 & 0.9666 \end{bmatrix}$	$\begin{bmatrix} -0.2034 \\ -0.5638 \\ -0.0517 \end{bmatrix}$	0.00492257
3D Points Registration	$\begin{bmatrix} 0.9986 & -0.0187 & 0.0501 \\ 0.0056 & 0.9674 & 0.2531 \\ -0.0533 & -0.2525 & 0.9661 \end{bmatrix}$	$\begin{bmatrix} -0.2033 \\ -0.5719 \\ -0.0488 \end{bmatrix}$	0.00473059

5.2 Robustness Analysis

In this experiment, we randomly generate a Kinect-like set of (u, v, d) points with $u \in (0,1920), v \in (0,1080), d \in (500,8000)$ (see Table 4). These points, along with the intrinsic parameters form a set of points S_1 in the world coordinate system.

In addition, we also defined a rotation matrix R and a translation vector $\vec{\boldsymbol{t}}$ as ground truth, where

$$R = \begin{bmatrix} 0.684718 & 0.136252 & 0.715959 \\ 0.492042 & 0.638267 & -0.592039 \\ -0.537639 & 0.757661 & 0.369991 \end{bmatrix}$$

$$\vec{\boldsymbol{t}} = \begin{bmatrix} 0.35 \\ -0.28 \\ 0.76 \end{bmatrix}$$

For all the points in the set S_1, we follow the convention that $q = R \cdot p + \vec{\boldsymbol{t}}$, forming set S_2 which is the perfect matching set for S_1.

The next step is to add the Gaussian noise in. We defined our levels of Gaussian noise with various standard deviations, but kept the mean to 0. Figure 5.6 shows how the Gaussian distribution behaves with different standard deviations.

We form the correspondence sets G_1 and G_2 by generating Gaussian noise and adding it to the points in S_1 and S_2. The noise could be interpreted as the error in dealing with real case usage, which may come from data gathering, parameter measuring, feature matching and so on.

We then input G_1 and G_2 into the camera pose estimation stage of our chosen algorithms and record the offsets between the outputs and the pre-set ground truth values. Note that, when sigma equals to 0, both algorithms give nearly error-free solutions, as expected.

The results are shown in Figures 5.7 and 5.8. From these, we conclude that each algorithm has a different tolerance to input noise. The 3D Registration algorithm performs best when faced with noise, as the offset for both R and T is close to 0 if sigma is less than 0.6. The Perspective-n-points algorithm's output is a relatively precise result if the noise level (sigma) is less than 0.4. The Eight-point algorithm is very sensitive to noise. Even a tiny amount of noise has a great influence on the results. Only when sigma is under 0.1, do we obtain results that we somehow could regard as reliable.

Past a certain noise level, it becomes clear that the offset changes irregularly (our error measure becomes meaningless). This is due in part to the fact that the space in which a rotation matrix can be wrong is bounded. For instance, there seems to be a peak noise value for the Eight-point algorithm at sigma=0.8, as seen in Figure 5.7. For these reasons, we focus our noise analysis to a sigma span of [0, 0.7] and use smaller intervals. Figures 5.9 and 5.10 report on these results.

Table 4 Synthetic Data

Index	u	v	d	Index	u	v	d
1	1185.36	441.60	2564	5	688.74	447.77	1986
2	236.50	875.36	1185	6	1122.33	842.32	4952
3	1542.77	992.21	5508	7	785.79	689.55	866
4	989.88	155.80	7421	8	908.11	258.49	3358

Table 5 Theta at sigma=0.8 and sigma=0.9

	θ_x	θ_y	θ_z	Offset_R
Sigma=0.8	$-41.3203°$	$12.8374°$	$61.4392°$	110.166
Sigma=0.9	$125.078°$	$-43.3872°$	$22.8942°$	98.2871
Ground Truth	$63.9722°$	$32.5231°$	$35.7012°$	

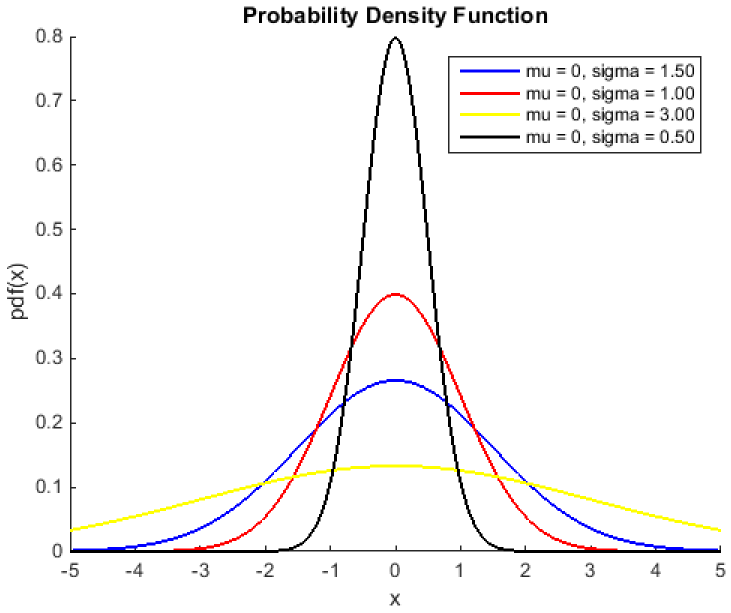


Figure 5.6 Gaussian Distribution with different values for sigma

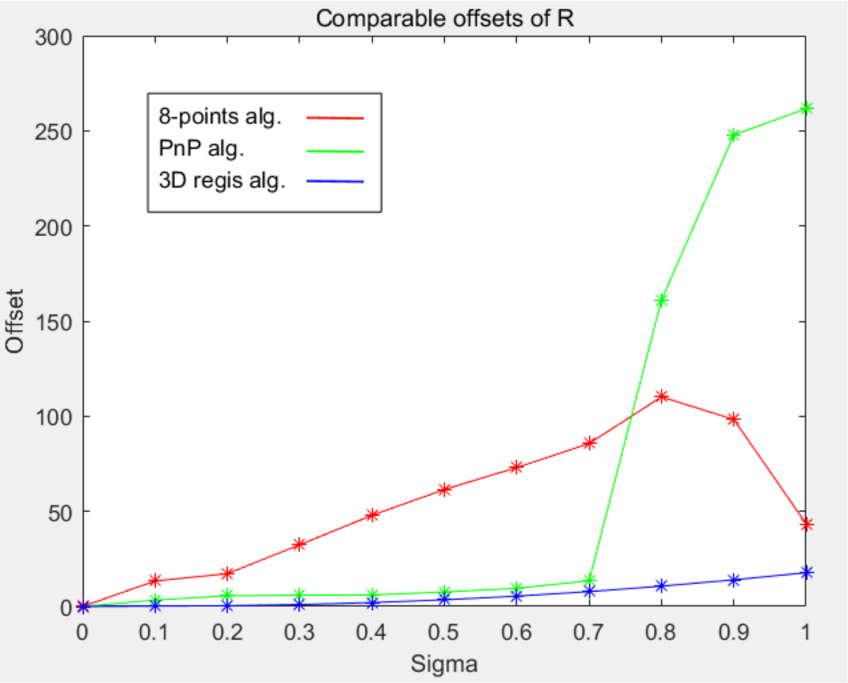


Figure 5.7 Comparative offsets of R for sigma∈ [0,1]

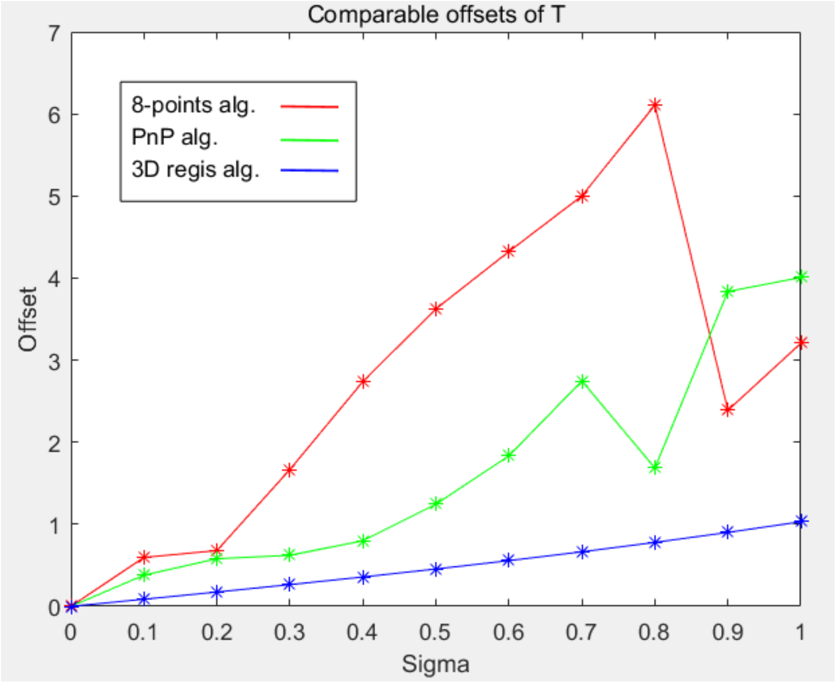


Figure 5.8 Comparative offsets of T for sigma∈ [0,1]

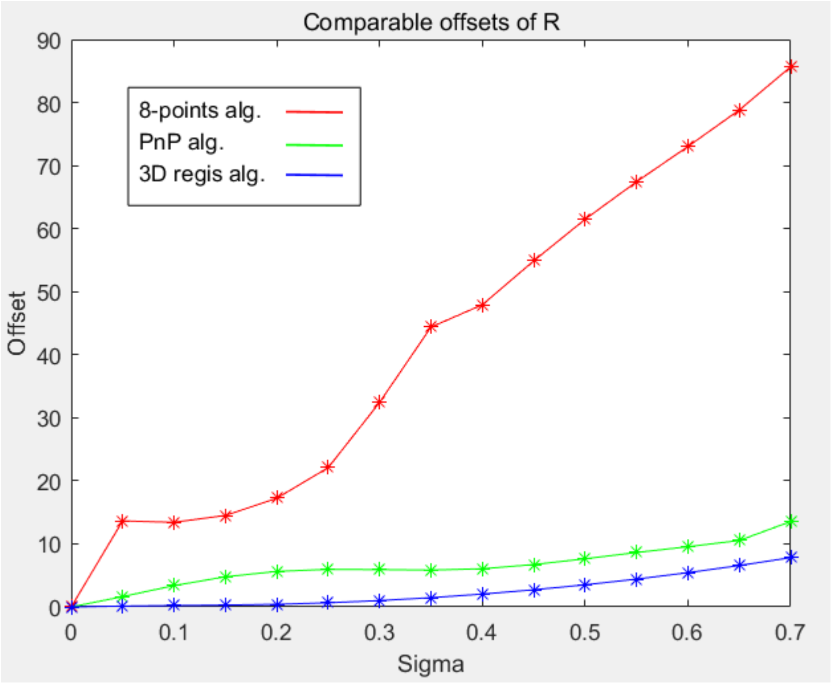


Figure 5.9 Comparative offsets of R for sigma∈ [0,0.7]

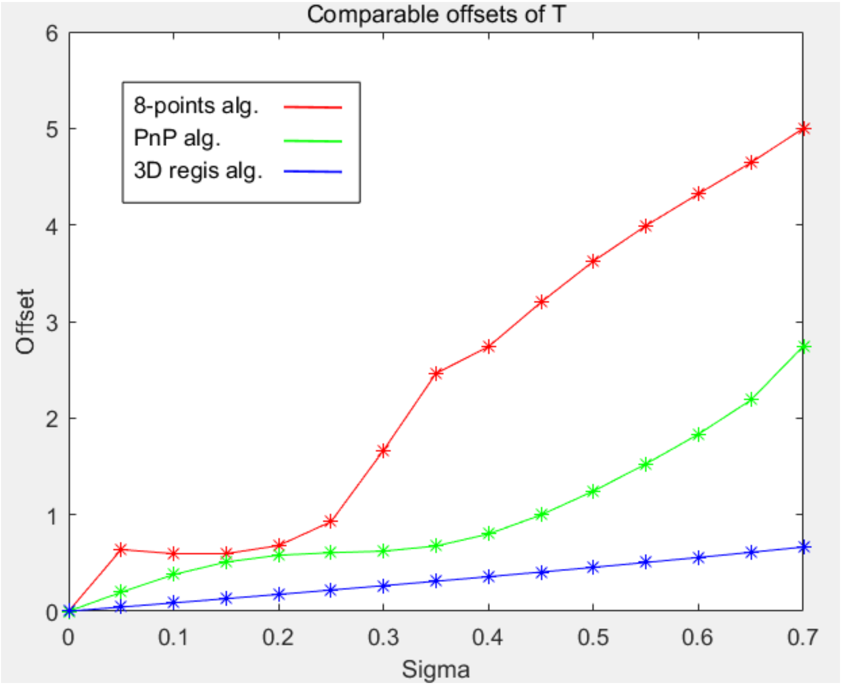


Figure 5.10 Comparable offsets of T for sigma∈ [0,0.7]

5.3 Conclusion

To conclude, this thesis integrates RGB-D data capture, feature points selection, correspondence optimization and camera pose reconstruction into one application to solve the problem of automatically calibrating the network of Kinect Sensors. And it firstly introduces 3D eight-points algorithm and numerically compare the robustness to noise data of three pose estimation algorithms, which are Eight-point algorithm, Perspective-n-point algorithm and 3D point set registration algorithm. This thesis finds that all the pose estimation algorithms output precise results in normal circumstance. Among them, 3D point set registration algorithm works best even when towards heavily noised data set.

Chapter 6

6 Future Work

6.1 System Constraints

Since our system is based on feature points correspondences, the inherent constraint for this system is that there be some significant view overlaps between the Kinect sensors. Therefore, additional sensors are required if we wish to calibrate for greater scene areas. To avoid this inherent constraint, Bok et al. [34] propose a practical solution to calibrate between a camera and a laser sensor system, without overlap. These ideas could be integrated in our framework.

6.2 Device Limitations

Kinect cameras have problems handling intricate light reflections very well, partially due to the method by which Kinect cameras use to estimate depth (infrared). Transparent objects such as glass, may partially reflect the infrared spectrum and cause significant errors in the depth images. In Figure 6.1, the drinking glass is almost not 3D-captured and the main effect is a distortion of depth information. In addition, if two Kinects are capturing a scene at the same time, their respective IR patterns may interfere, in turn causing errors when generating RGB-D images. More work is definitely needed at the sensor level.

What’s more, temperature as well as air draft may also have an influence on the process of range measurement of the Kinect [35]. Figure 6.2 indicates that various temperatures can cause slightly different results with all other parameters remaining constant.

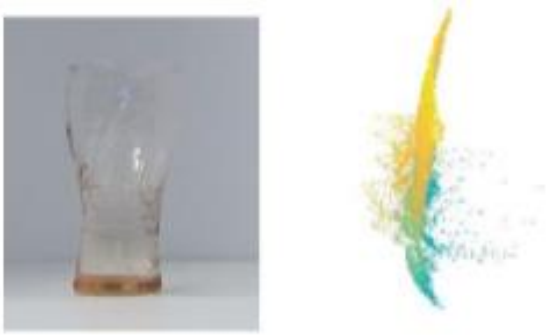
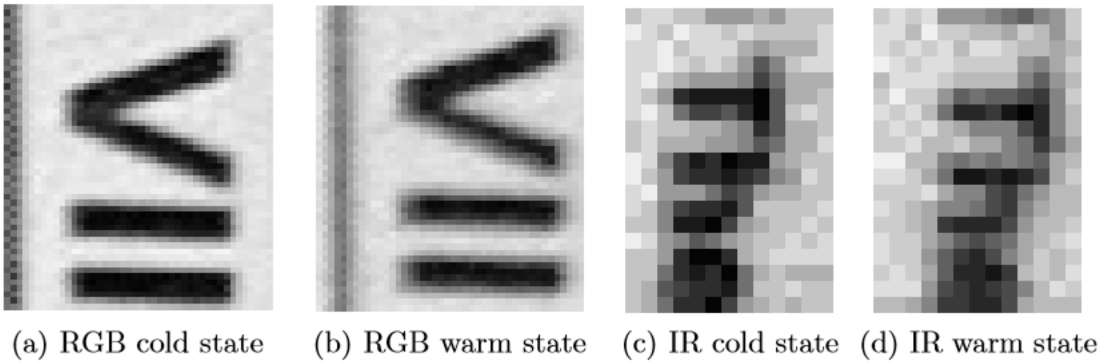


Figure 6.1 The reflection problem with a transparent glass. Different color in the right indicates the surface of a glass have different depth. This mismeasurement may cause problem in further algorithms.



(a) RGB cold state (b) RGB warm state (c) IR cold state (d) IR warm state

Figure 6.2 Different RGB-D images caused by a change in ambient temperature. In cold state, the Kinect was cooled down by an externally mounted fan which slowly streams air through the Kinect’s body and cools down its internal components to the

environmental temperature of 27.6℃. In warm state, the fan was deactivated and the Kinect was warmed up just by processing the color and depth image-stream for 45 minutes.

References

[1] Han, Jungong, et al. "Enhanced computer vision with microsoft kinect sensor: A review." *IEEE transactions on cybernetics* 43.5 (2013): 1318-1334.

[2] Lowe, D. G. (1999). Object recognition from local scale-invariant features. In Computer vision, 1999. The proceedings of the seventh IEEE international conference on (Vol. 2, pp. 1150-1157).

[3] Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. International journal of computer vision, 60(2), 91-110.

[4] Bay, Herbert, et al. "Speeded-up robust features (SURF)." Computer vision and image understanding 110.3 (2008): 346-359.

[5] Lepetit, V., Moreno-Noguer, F., & Fua, P. (2009). Epnp: An accurate o (n) solution to the pnp problem. International journal of computer vision, 81(2), 155-166.

[6] Hartley, R. I. (1997). In defense of the eight-point algorithm. IEEE Transactions on pattern analysis and machine intelligence, 19(6), 580-593.

[7] Zhao, Z., & Liu, Y. (2008, June). Practical multi-camera calibration algorithm with 1d objects for virtual environments. In Multimedia and Expo, 2008 IEEE International Conference on (pp. 1197-1200).

[8] Shen, E., & Hornsey, R. (2011). Multi-camera network calibration with a non-planar target. IEEE Sensors Journal, 11(10), 2356-2364.

[9] Courchay, J., Dalalyan, A., Keriven, R., & Sturm, P. (2010, May). A global camera network calibration method with linear programming. In Proceedings of the International Symposium on 3D Data Processing, Visualization and Transmission.

[10] Kim, J. H., Li, H., & Hartley, R. (2010). Motion estimation for nonoverlapping multicamera rigs: Linear algebraic and l∞ geometric solutions. IEEE Transactions on Pattern Analysis and Machine Intelligence, 32(6), 1044-1059.

[11] Stewénius, H., Nistér, D., Oskarsson, M., & Åström, K. (2005, October). Solutions to minimal generalized relative pose problems. In Workshop on omnidirectional vision (Vol. 1, No. 2, p. 3).

[12] Vasconcelos, F., & Barreto, J. P. (2009). Towards a minimal solution for the relative pose between axial cameras. Int. J. Comput. Vision, 84(3), 237-256.

[13] Kumar, R. K., Ilie, A., Frahm, J. M., & Pollefeys, M. (2008, June). Simple calibration of non-overlapping cameras with a mirror. In Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on (pp. 1-7).

[14] Rodrigues, R., Barreto, J. P., & Nunes, U. (2010, September). Camera pose estimation using images of planar mirror reflections. In European Conference on Computer Vision (pp. 382-395). Springer, Berlin, Heidelberg.

[15] Auvinet, E., Meunier, J., & Multon, F. (2012, July). Multiple depth cameras calibration and body volume reconstruction for gait analysis. In Information Science, Signal Processing and their Applications (ISSPA), 2012 11th International Conference on (pp. 478-483).

[16] Le, Q. V., & Ng, A. Y. (2009, October). Joint calibration of multiple sensors. In Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on (pp. 3651-3658).

[17] Burrus, N. (2011). Kinect RGB demo. Manctl Labs. Available online: http://rgbdemo. org/(accessed on 21 January 2017).

[18] Zhang, C., & Zhang, Z. (2014). Calibration between depth and color sensors for commodity depth cameras. In Computer Vision and Machine Learning with RGB-D Sensors (pp. 47-64). Springer International Publishing.

[19] Gaffney, M. (2011). Kinect/3D scanner calibration pattern. Available online: http://www.thingiverse.com/thing:7793.

[20] Berger, K., Ruhl, K., Schroeder, Y., Bruemmer, C., Scholz, A., & Magnor, M. A. (2011, October). Markerless motion capture using multiple color-depth sensors. In VMV (pp. 317-324).

[21] Binocular disparity. (n.d.). In Wikipedia. Retrieved October 4, 2017, from https://en.wikipedia.org/wiki/Binocular_disparity

[22] Zhang, Z. (2000). A flexible new technique for camera calibration. IEEE Transactions on pattern analysis and machine intelligence, 22(11), 1330-1334.

[23] Microsoft, “Camera intrinsics structure.” Retrieved October 6, 2017, from https://github.com/Kinect/Docs/tree/master/Kinect4Windows2.0/k4w2/Reference/Kinect_for_Windows_v2/Kinect/CameraIntrinsics_Structure

[24] Valgma, L. (2016). 3D reconstruction using Kinect v2 camera(Doctoral dissertation, Tartu Ülikool).

[25] Hartley, R., & Zisserman, A. (2000). Epipolar geometry and the fundamental matrix. Multiple view geometry.

[26] Longuet-Higgins, H. C. (1981). A computer algorithm for reconstructing a scene from two projections. Nature, 293(5828), 133-135.

[27] Bolb detection. (n.d.). In Wikipedia. Retrieved October 4, 2017, from https://en.wikipedia.org/wiki/Blob_detection

[28] Random sample consensus. (n.d.). In Wikipedia. Retrieved October 6, 2017, from https://en.wikipedia.org/wiki/Random_sample_consensus

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

[30] Perspective-n-Point (n.d.). In Wikipedia. Retrieved October 6, 2017, from https://en.wikipedia.org/wiki/Perspective-n-Point

[31] OpenCV 2.4.13.4 Documentation. Camera Calibration and 3D Reconstruction

. Retrieved from http://docs.opencv.org/2.4/modules/calib3d/doc/camera_calibration_and_3d_reconstruction.html

[32] Besl, P. J., & McKay, N. D. (1992). A method for registration of 3-D shapes. IEEE Transactions on pattern analysis and machine intelligence, 14(2), 239-256.

[33] Panchal, P. M., Panchal, S. R., & Shah, S. K. (2013). A comparison of SIFT and SURF. International Journal of Innovative Research in Computer and Communication Engineering, 1(2), 323-327.

[34] Bok, Y., Choi, D. G., Vasseur, P., & Kweon, I. S. (2014, September). Extrinsic calibration of non-overlapping camera-laser system using structured environment. In Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on (pp. 436-443).

[35] Fiedler, D., & Müller, H. (2013). Impact of thermal and environmental conditions on the kinect sensor. In Advances in Depth Image Analysis and Applications (pp. 21-31). Springer, Berlin, Heidelberg.

Curriculum Vitae

Name:	Xiaoyang Li
Post-secondary Education and Degrees:	Tianjin University Tianjin City, Tianjin, China 2012-2016 B.A.
	The University of Western Ontario London, Ontario, Canada 2016-2017 M.A.
Honours and Awards:	Western Graduate Research Scholarship 2016-2017 Mitacs Globalink Graduate Fellowship Award\ 2016-2017
Related Work Experience	Teaching Assistant The University of Western Ontario 2016-2017