AUTONOMOUS NAVIGATION AND MAPPING OF MOBILE ROBOTS BASED ON 2D/3D CAMERAS COMBINATION

Vom Fachbereich Elektrotechnik und Informatik der Universität Siegen

Zur Erlangung des akademischen Grades

Doktors der Ingenieurwissenschaften
(Dr. -Ing.)

genehmigte Dissertation
von

M.Eng. Chanin Joochim

1. Gutachter : Prof. Dr. -Ing. Hubert Roth
2. Gutachter : Prof. Dr. rer. nat. Klaus Schilling
Vorsitzender : Prof. Dr. -Ing. Robert Mayr

Tag der mündlichen Prüfung: 27.05.2011

Kurzfassung

Aufgrund der tendenziell zunehmenden Nachfrage an Systemen zur Unterstützung des alltäglichen Lebens gibt es derzeit ein großes Interesse an autonomen Systemen. Autonome Systeme werden in Häusern, Büros, Museen sowie in Fabriken eingesetzt. Sie können verschiedene Aufgaben erledigen, beispielsweise beim Reinigen, als Helfer im Haushalt, im Bereich der Sicherheit und Bildung, im Supermarkt sowie im Empfang als Auskunft, weil sie dazu verwendet werden können, die Verarbeitungszeit zu kontrollieren und präzise, zuverlässige Ergebnisse zu liefern. Ein Forschungsgebiet autonomer Systeme ist die Navigation und Kartenerstellung. Das heißt, mobile Roboter sollen selbständig ihre Aufgaben erledigen und zugleich eine Karte der Umgebung erstellen, um navigieren zu können.

Das Hauptproblem besteht darin, dass der mobile Roboter in einer unbekannten Umgebung, in der keine zusätzlichen Bezugsinformationen vorhanden sind, das Gelände erkunden und eine dreidimensionale Karte davon erstellen muss. Der Roboter muss seine Positionen innerhalb der Karte bestimmen. Es ist notwendig, ein unterscheidbares Objekt zu finden. Daher spielen die ausgewählten Sensoren und der Register-Algorithmus eine relevante Rolle. Die Sensoren, die sowohl Tiefen- als auch Bilddaten liefern können, sind noch unzureichend. Der neue 3D-Sensor, nämlich der „Photonic Mixer Device“ (PMD), erzeugt mit hoher Bildwiederholfrequenz eine Echtzeitvolumenerfassung des umliegenden Szenarios und liefert Tiefen- und Graustufendaten. Allerdings erfordert die höhere Qualität der dreidimensionalen Erkundung der Umgebung Details und Strukturen der Oberflächen, die man nur mit einer hochauflösenden CCD-Kamera erhalten kann. Die vorliegende Arbeit präsentiert somit eine Exploration eines mobilen Roboters mit Hilfe der Kombination einer CCD- und PMD-Kamera, um eine dreidimensionale Karte der Umgebung zu erstellen.

Außerdem wird ein Hochleistungsalgorithmus zur Erstellung von 3D Karten und zur Poseschätzung in Echtzeit unter Verwendung des „Simultaneous Localization and Mapping“ (SLAM) Verfahrens präsentiert. Der autonom arbeitende, mobile Roboter soll ferner Aufgaben übernehmen, wie z.B. die Erkennung von Objekten in ihrer Umgebung, um verschiedene praktische Aufgaben zu lösen. Die visuellen Daten der CCD-Kamera liefern nicht nur eine hohe Auflösung der Textur-Daten für die Tiefendaten, sondern werden auch für die Objekterkennung verwendet. Der „Iterative Closest Point“ (ICP) Algorithmus benutzt zwei Punktwolken, um den Bewegungsvektor zu bestimmen. Schließlich sind die Auswertung der Korrespondenzen und die Rekonstruktion der Karte, um die reale Umgebung abzubilden, in dieser Arbeit enthalten.

Abstract

Presently, intelligent autonomous systems have to perform very interesting tasks due to trendy increases in support demands of human living. Autonomous systems have been used in various applications like houses, offices, museums as well as in factories. They are able to operate in several kinds of applications such as cleaning, household assistance, transportation, security, education and shop assistance because they can be used to control the processing time, and to provide precise and reliable output. One research field of autonomous systems is mobile robot navigation and map generation. That means the mobile robot should work autonomously while generating a map, which the robot follows.

The main issue is that the mobile robot has to explore an unknown environment and to generate a three dimensional map of an unknown environment in case that there is not any further reference information. The mobile robot has to estimate its position and pose. It is required to find distinguishable objects. Therefore, the selected sensors and registered algorithms are significant. The sensors, which can provide both, depth as well as image data are still deficient. A new 3D sensor, namely the Photonic Mixer Device (PMD), generates a high rate output in real-time capturing the surrounding scenario as well as the depth and gray scale data. However, a higher quality of three dimension explorations requires details and textures of surfaces, which can be obtained from a high resolution CCD camera. This work hence presents the mobile robot exploration using the integration of CCD and PMD camera in order to create a three dimensional map.

In addition, a high performance algorithm for 3D mapping and pose estimation of the locomotion in real time, using the "Simultaneous Localization and Mapping" (SLAM) technique is proposed. The flawlessly mobile robot should also handle the tasks, such as the recognition of objects in its environment, in order to achieve various practical missions. Visual input from the CCD camera not only delivers high resolution texture data on depth volume, but is also used for object recognition. The "Iterative Closest Point" (ICP) algorithm is using two sets of points to find out the translation and rotation vector between two scans. Finally, the evaluation of the correspondences and the reconstruction of the map to resemble the real environment are included in this thesis.

Acknowledgements

I would like to acknowledge my gratitude towards my supervisor Prof. Dr.-Ing. Hubert Roth for giving me a chance as a Ph.D. student. I am grateful for his guidance, inspirations and motivations during the different stages of my work. I really appreciate his confidence and his support to participate in many international conferences. It is a great pleasure for me to see the world and to get to know how big it is.

I would like to thank especially my second supervisor Prof. Dr. rer. nat. Klaus Schilling for his kindly helpful and very rapid response to all questions. His Telematics research community inspired me and gave me the brilliant idea to improve my research. I am grateful to the committee chair Prof. Dr. –Ing. Robert Mayr for my final examination management.

I am especially grateful to the German Academic Exchange Service (DAAD) for giving me the opportunity to pursue my doctoral study by supporting the scholarship and all helpfulness to me and my wife during our stay in Germany.

I wish to thank my colleague and officemate, Dipl. Ing. Christof Hille for the assistance in several things, i.e. discussing programming, providing ideas and for all his effort to explain things to me and for his suggestion. A special thank to Dr. -Ing. Chayakorn Netramai, Dipl. Ing. Matthias Mende and Dipl. Ing. Peter Will not only for the terrific programming idea but also for the fantastic coffee time and for sharing gorgeous stories as well as fabulous German traditions and histories. Furthermore, I am very thankful to all RST team members for many helpful discussions, recommendations and for sharing experiences.

I want to thank Ms. Judy Arndt and Ms. Frauke Koch for being faithful proofreaders and correcting the English writing. I would like to say a special thank to the exchange students from the European Region Action Scheme for the Mobility of University Students (Erasmus Programme) for supporting some parts of my research.

I would like to thank my parents for providing me with high quality genetic materials, enthusiasm, inspiration, and also a big thank you to all my family members for their love, encouragement, sacrifices, patience and care.

Finally, I wish to thank my wife, Orapadee Joochim, for being with me. She has supported and discussed my work and left encouraging comments during difficult times. Thanks for being a part of my life, sharing and struggling all obstacles together. Her love sustains me through my ordeal of research works. I am never alone.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

Autonomous systems nowadays have to perform very interactive tasks due to trendy increases in facility support demands of human living. Autonomous systems have been used in various houses, offices, museums as well as in factories. The autonomous system can to operate several kinds of applications such as cleanings, house helpers, deliveries, safety, educations, supermarket helpers, receiving officers because it can be used to control the processing time, and to provide precise and reliable outputs. The autonomous processes do not need to be paused. They can work all day without pause or rest. Therefore, they are expected to fulfill many human task demands. Reliability is not only the main target in adapting the autonomous system instead of human works but also safety issues are important. Industrial robots have a variety of uses in industries. They are used, instead of humans for material handling, spot and arc welding, as well as for carrying heavy items.

1.1 The mobile robotics candidates

The mobile robot is one subset of the autonomous system. Mobile robots have been extensively studied for many decades. The mechanical constructions and modern control strategy have been developed at the same time. Energy efficiency is one of the most challenging issues in the design of mobile vehicle systems. The mechanical parts have been improved by many researchers over centuries, e.g. a small vehicle, robots with two legs (humanoid) and many legs (insects), as well as flying robots. Several kinds of modern robots can simulate and imitate particular natural living things. *ASIMO* was created by Honda. It has an appearance resembling a human being. It can walk or run on two feet at speed of up to 6 km/h. *MSR-H01* is hexapod robot. It can control the 3 DOF freedoms smoothly, and walks on six legs. It has generated a great

deal flexibility of movement. The *Blue fin* robotics is an autonomous underwater vehicle (AUV). It travels underwater, and makes detailed maps of the seafloor before starting to build a subsea infrastructure. The *Foster-Miller TALON* is a small unmanned ground vehicle (UGV). It is an extension of human capability in the battle field. It is generally capable of operating outdoors and over a wide variety of terrain. The *md4-100* is the unmanned aerial vehicle (UAV), quad rotor helicopter. It can explore the sky, do aerial photography and plant inspection. Previous examples mentioned above are minority applications of the modern mobile robots. There are still many fantastic mobile robotics that we do not cite. In particular, the improvement of mechanical technique is as interesting as the vision system. Mobile vision is similar to human vision. The human eye sends the environment vision to the brain. The brain recognizes what the objects are that the human eye is watching. Those objects can be caught, eaten, or can walk away. Further, humans have a left and a right eye. The small different focus between the two eyes is sent, and calculated in brain. The distance between the object and the human body can be precisely estimated.
The capability of an unmanned ground vehicle (UGV) to determine its location is a fundamental issue of autonomous navigation. The high performance navigation system should enable strategic path planning. The goal location is to reach the region of coverage, as well as exploring and avoiding obstacles. This should make the following trajectories possible. The general problems to solve are to let a mobile robot explore an unknown environment using range sensing and to build a map of the environment from sensor data. The sensors data with an alignment method can be used to represent the world model. If merging several scans from different locations, it is essential to align the scanned data. The difficulty is that using only odometry information alone is inadequate for determining the relative poses. When each frame of sensor data is obtained, it is aligned to a previous frame or to a cumulative global model. To be able to resolve inconsistency once, we need to maintain the local frames of data together with their estimated poses.

1.2 Simultaneous Localization and Mapping, SLAM

The principle problems of mobile robot navigation are three basic questions. Where am I? Where am I going? How do I go there? Figure 1.1 shows the fundamental problem of mobile robot locomotion. To answer these questions, the mobile robot has to have a model of the environment, to perceive and analyze the environment, to find its position within the environment and to execute the movement. Thus, mobile robot navigation requires competencies in modeling, perception, planning and controlling. The vision system and the high performance of mobile robot controller are the most important keys to solve these problems. For mobile robots, applications are indispensable to recognize 3D objects.

The question is whether or not it is possible for mobile robots to start in an unknown location in an unknown environment, and at the same time to build a map of the environment while simultaneously using this map in the mobile robot locomotion. The SLAM lets the mobile robots operate in an environment without previous information. The SLAM increases the range of potential mobile robots application. The eventual SLAM output makes mobile robots truly autonomous operation. The research output represented by the SLAM technique can solve indeed those problems.

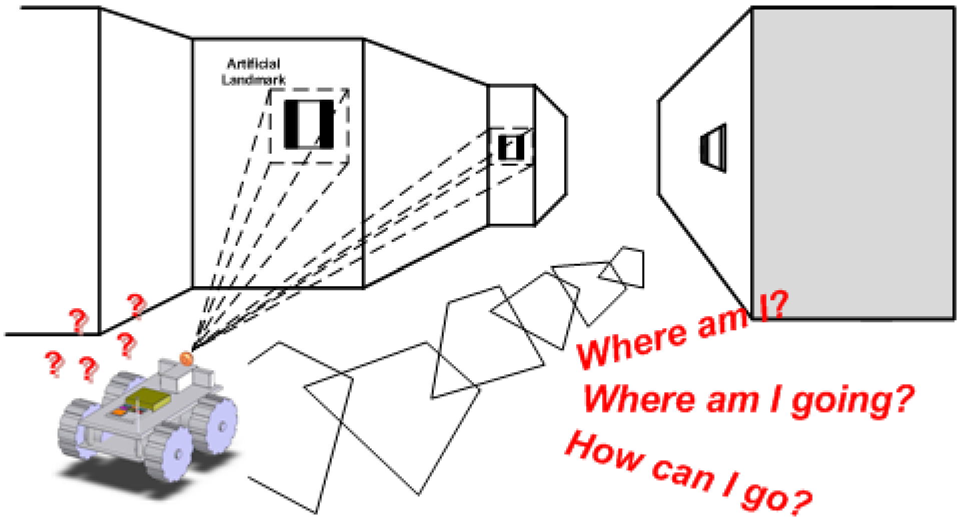


Figure 1.1 The fundamental of mobile robot locomotion

The real world environment is very complex. The estimation of six dimensions for robot pose by using visual sensors is a challenging problem. The six dimensions are x, y and z directions as well as roll, yaw and pitch angles coordinates. When the robot moves in the natural environment, these degrees of freedom have to be considered. The good sensors plus the highly efficient algorithm are necessary. The sensors, which SLAM applications use, separate into two types, the passive and active sensors.

Passive sensors : 2D camera, stereo camera, GPS, RFID
Active sensors : Laser range finder, ultrasonic, radar, infrared, sonar, time of flight camera.

The passive sensors do not contact the object, nor are they irritable to the environment, as they run on low energy and have low costs. They further have the ability of a fast data acquisition rate. On the other hand though, the active sensors are bigger, and consume much energy but are very robust and reliable. Nevertheless, the requirements are dependent on the pros and cons of the sensors. The prominent keys, which have to be considered, are

Speed : data acquisition speed should be high enough for the accuracy of mobile robot locomotion.
Robustness : output of sensors shouldn’t be varying in the different environments high/low temperature, humidity, dust, fog, light and surface.
Energy : low energy consumption can extend the operating period of the mobile robot tasks.

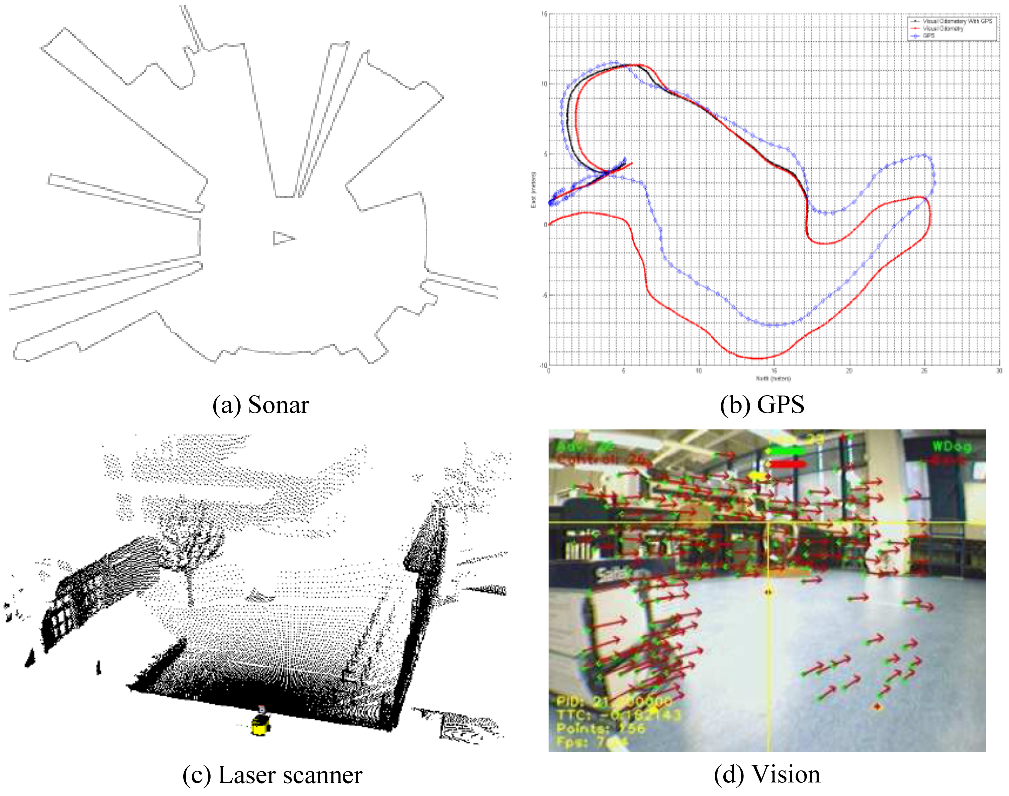

(a) Sonar (b) GPS (c) Laser scanner (d) Vision

Figure 1.2 Sensors candidates (a) Sonar [1], (b) GPS from outdoor environment [2], (c) 3D laser scanner output represents as a point cloud [3], (d) Vision output from single camera [4].

Figure 1.2 shows the output from several sensors (a) [1] used sonar sensor for mobile robot navigation. Sonar outputs provide a pretty nice output and low cost construction. The useful operating range is 20 feet with a typical range resolution of 1 inch. Moreover, the emitting source is an infrared LED, therefore potential eye safety problems are avoided compared with a laser source. The two dimension path planning can be created easily from a sonar sensor. However, the sonar’s rate of data acquisition is limited by the speed of sound, which is 343.2 m/s at 20 degrees Celsius. It should be possible to obtain and process 100 returns per second in tracking a target 1 meter away. A very high vehicle position update rate should be possible, if directed sensing strategies are combined with a faster firing capability. (b) One interesting sensor is GPS, [2] presented a real-time low cost system to localize a mobile robot in outdoor environments. Their system relies on stereo vision fusion on a low-cost GPS sensor using a Kalman filter to prevent long-term drifts. The experimental results were presented for outdoor localization in moderately sized environment over 100 meters. (c) [3] proposed a framework for analyzing the results of a SLAM approach based on a metric for measuring the error of the corrected trajectory using laser scanner. (d) [4] presented a vision based navigation algorithm. He used the output from the single camera to operate an autonomous mobile robot based on a frontal optical flow estimation algorithm. He proposed the control strategy by using the main input from vision sensor. The mobile robot could be built at a low cost, but could also provide the high performance of autonomous strategy image, and obstacle avoidance.

In an outdoor environment the Global Positioning System (GPS) is very well known, and provides the high reliability. However, indoor mobile robots cannot use the Global Positioning System (GPS) to reset dead-reckoning error growth unless they surface, which is usually undesirable. In this research, we aim to use the novel active sensor, the time of flight camera, to create the three dimension mapping. The new 3D sensor namely the Photonic Mixer Devices (PMD) purposes real-time capturing the surrounding volume. The PMD is becoming more attractive because it has its own modulated light source. It can work independently from the environmental lights. The PMD cameras can measures the depth of the objects by using the time of flight which emits from the transmitter, and fly back to the receiver. The main drawback of the current PMD camera is the low output resolution (64x48) compared to other types of cameras available. However, we do not use the PMD sensor on its own. We propose to integrate the passive and active sensor (high resolution and time of flight camera) in order to compensate the drawback from both kind of sensors. SLAM is expected to build a map at the same time and to estimate the pose of the locomotion in real time. The flawlessly mobile robotic application tasks meanwhile should be able to recognize objects in the environment in order to achieve a variety of practical missions. The high resolution image (2D camera) is registered on the 3D volume (PMD) subsequently rescaling and calibrating both sensors. Visual input from the 2D camera not only delivers high resolution texture data on 3D volume but also is used for object recognition. In the object detection technique, we apply the open source library from the "Intel Open Source Computer Vision Library, OpenCV". The OpenCV is the open source visual C++ code library. It is very famous in real time image processing, and gives a very powerful vision processing.

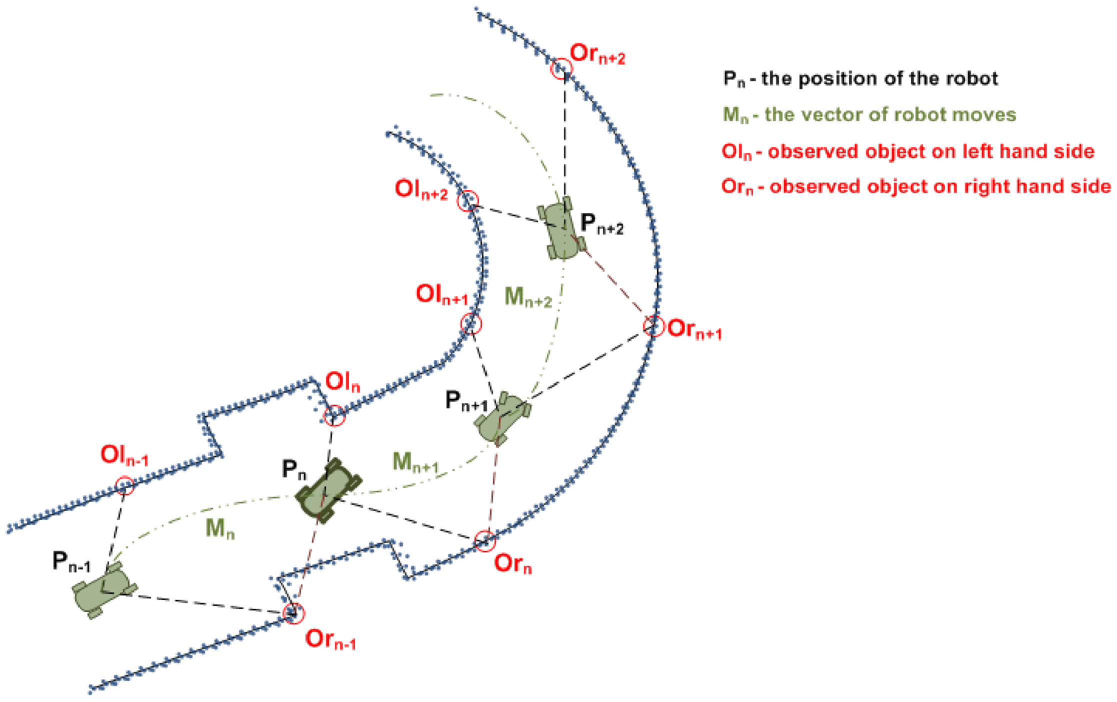

P_n - the position of the robot
M_n - the vector of robot moves
Ol_n - observed object on left hand side
Or_n - observed object on right hand side
Or_{n+2}
Ol_{n+2}
P_{n+2}
Ol_{n+1}
M_{n+2}
Or_{n+1}
Ol_n
P_{n+1}
Ol_{n-1}
P_n
M_{n+1}
M_n
Or_n
P_{n-1}
Or_{n-1}

Figure 1.3 Mapping problem

Robotic mapping addresses the problem of acquiring spatial models of physical environments through mobile robots. Figure 1.3 shows the fundamental mapping problem. To complete the overall mapping, it is necessary that each single step is mapped. This in turn means that in order to solve the overall mapping problem, each step needs to be solved as each step proposes a problem. Figure 1.3 expresses the mobile robot moved through the narrow path.

Where

n	: discrete time index $\{n = 1,2,3,\dots\}$
P_n	: currently robot position
M_n	: robot pose vector
Ol_n	: observed objects on the left hand side
Or_n	: observed objects on the right hand side

The observation of the environmental object is the first problem that needs to be solved in order to commence with the mapping. The observed objects on the left hand side (Ol_n) and right hand side (Or_n) of the robot path are used to be the reference positions for the next robot pose (P_n). The object detecting technique requires an accuracy method. The maximum robot movement vector (M_p) is limited, and must not control beyond the observed objects, as it will otherwise cause the next robot pose to be raveled. Therefore, the history states $\{n, n-1, n-2, \dots\}$ are necessary to keep in the databank, and let them be used in the future steps. The robot starts at P_n, and is moving to the nearby position P_{n+1}, P_n subsequently. In general, the result of matching pairwise scans is a complex and possibly conflicting network pose relations. Spatial relations between local frames are derived from matching pairs of scans or from odometry measurements. Maps are commonly used for robot navigation (e.g., localization) and in wide variety application areas. The robotic mapping problem is commonly referred to as SLAM (simultaneous localization and mapping) or CML (concurrent mapping and localization).

Since robotic maps are needed in many applications, techniques for building a map as quickly as possible would be highly beneficial. Survey class AUVs must maintain forward motion for controllability. Hence, the ability in adaptively choosing a sensing and motion strategy, which obtains the most information about the environment, is especially important. The ICP algorithm is taken over the theatrical image registration due to yield the real time data frames registration. Figure 1.4 shows the principle ideas of ICP algorithm. Assuming that the motion between the two frames is small or approximately known, we can apply the approximate poses of the motion between the two frames to the first one. Then, the motion between the intermediate frame and the second frame can be small. The motion of robot should not be bigger than the range of the maximum pattern distance. ICP algorithm is the method to find rotation and translation relationship between two frames. After we the relation, the two frames can merge, and build the 3D maps afterwards.

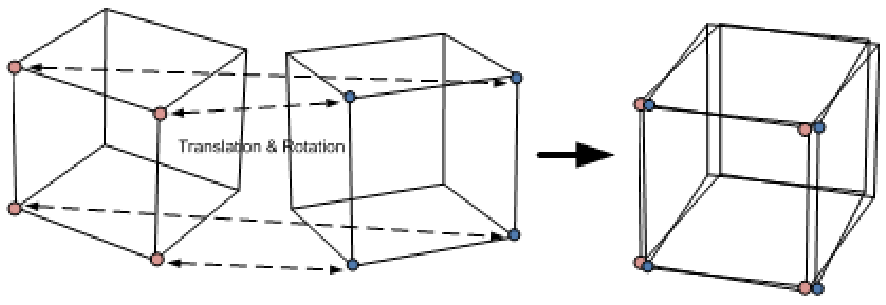


Figure 1.4 Basic ICP concept

Regarding enhancing 3D real time applications, several researchers have purposed. Prasad et. al [5] presented the first step to combine a high resolution two dimension camera and PMD camera. They showed the idea to setup the mechanical platform. Their outputs could enhance the 3D vision and express output in real time. Schiler et. al [6] presented a calibration model between the PMD and CCD cameras. Since the field of view and resolution of PMD are low, the traditional calibration has difficulty providing the correct output. However, an intensive study about the use of such a combination system for a motion estimation task is still missing. Their work therefore seeks a way to combine the output from PMD and the stereo camera for the 6D motion estimation task. A data combination method, which compensates the strong and weak features from each camera, was presented. The suggested method is implemented on the real combined camera system. Carefully designed experiments are presented in order to demonstrate and evaluate the improvement of the result over the conventional single camera system.

If the robot poses were known, the local sensor inputs of the robot, i.e., local maps, could be registered into a common coordinate system to create a map. Unfortunately, any mobile robot's self localization suffers from imprecision. Therefore, the structure of the local elements, e.g., of single scans, needs to be used to create a precise global map. On the other hand, it is relatively easy to localize a robot, if the environment is known, i.e., a map is provided. The nature of SLAM is like the chicken-egg problem, namely the coordinates of the elements of the map depends on the robot's position, and the robot's position is estimated with by the use of the map navigation.

1.3 Thesis overview

In this thesis, the aim is to present the 3D mapping approach using new 3D camera sensors. To achieve this, several kinds of knowledge have to be combined. We separate all details in 7 chapters as shown below.

Chapter 2 presents the literature review of researchers. Their studies involve intelligent mobile robotics, vision, image processing as well as SLAM. Several essential fundamental concepts are included and discussed in this chapter. The related works, methods, sensors etc., which relate to or use the same background, are discussed for their pros and cons. Many ideas are also applied and cited in our thesis.

Chapter 3 demonstrates the mobile robot, MERLIN and all hardware agriculture designs. The construction, details and prominent points of each sensor are described. The control of mobile robot has been separated into two modes i.e., the manual and autonomous mode. The software used to control the mobile robot from the work station is LABVIEW. The C programming language is used to program the microcontroller. The image processing code is developed by using Visual C++. All software is explained in detail relating to the functions and techniques used. The software examples are demonstrated.

Chapter 4 presents the fundamental concepts of image processing. The introduction of how to capture the raw data from 2D/3D cameras is included. The image from high resolution 2D camera is combined with the 3D data from the PMD camera in order to yield the 3D scenario with the texture information. The data from 2D camera is not only used to increase the texture to 3D data but is also used for object detection. The camera calibration method is proposed to combine data between 2D/3D data. The data relevance is important for the high quality output. The sharewares OpenCV and OpenGL are the main keys for capturing 2D and showing the combination output respectively.

Chapter 5 introduces the principle of SLAM. The robot navigations have to be calculated for the pose estimation. In general, robot poses consist of the translation and rotation matrices. The ICP is a key algorithm to calculate the translation and rotation. The translation and rotation are afterwards used to build the real world mapping and to show mobile robot path.

Chapter 6 the real-time object detection and SLAM results are presented. The real implementation is tested and presented in this chapter. The experiments are separated in sections. Firstly, the motion estimation is tested by evaluating the output of the stereo and PMD camera in order to understand the performance of the navigation system. The results conclude that PMD camera provides an excellent translation result but rotation results are worse than the results of the stereo camera. The combination approach is the optimize solution to yield better results. Secondly, the object detection in a 3D environment is presented. The experiments prove the proposed method can detect the artificial landmark in real time. The 2D/3D cameras combination approach is used and presented the results. The 3D map buildings are presented in several environments, with and without 2D combination. The output from 2D/3D camera combination approach shows the reliability of the three dimension map building. The suggestions regarding pros and cons of using a new 3D camera are concluded in this chapter.

Chapter 7 concludes and summarizes the works. This research aims to use the novel 3D sensors, PMD, for mobile robot applications. Many approaches are presented to improve the performance of 3D map building. Future works focus on increasing the field of view of 3D image, mobile robot hardware, moving speed and processing time. The matching approach is also inquired to improve a step in this direction.

Some parts of this thesis were published at international conferences. The symposium areas varied in topics, but were all related in the fields of telematic, sensors network, mobile robot applications as well as image processing. The following is a summarization of the according publications.

1. Jens Bernshausen, Chanin Joochim, Hubert Roth, ***"Mobile Robot Self Localization and 3D Map Building using a 3D PMD-Camera"***, 18th World Congress of the International Federation of Automatic Control (IFAC), August28 – September2, 2011. Milan, Italy.

2. Chanin Joochim, Hubert Roth, ***"Mobile Robot Exploration Based On Three Dimension Cameras Acquisition"***, The Second IFAC Symposium on Telematics Applications (TA 2010), October 5-8, 2010, Timisoara, Romania.

3. C. Joochim, H. Roth, ***"The Indoor SLAM using Multiple Three Dimension Sensors Integration"***, The IEEE 5th International Workshop on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS'2009), September 21-23, 2009, Rende (Cosenza), Italy.

4. C. Joochim, C. Netramai and H. Roth, ***"Coordination of SLAM and Artificial Landmark Recognition using 2D/3D Sensors for Mobile Robot Vision"***, The Tenth International Conference on Pattern Recognition and Information Processing (PRIP'2009), May 19-21, 2009, Minsk, Belarus.

5. C. Joochim, H. Roth, ***"Development of a 3D Mapping using 2D/3D Sensors for Mobile Robot Locomotion"***, The 2008 IEEE International Conference of Technologies for Practical Robot Applications, TePRA", 10-11 November 2008. Woburn, Massachusetts, USA.

6. Hubert Roth, Anatoly Sachenko, Vasyl Koval, Chanin Joochim, Oleh Adamiv, Viktor Kapura.***"The 3D Mapping Preparation using 2D/3D cameras for Mobile Robot Control"***, The Scientific-theoretical journal "Artificial Intelligence", State University of Informatics and Artificail Intelligence, No.4 2008, Ukraine.

7. C. Netramai, O. Melnychuk, C. Joochim and H. Roth, ***"Combining PMD and Stereo camera for Motion Estimation of a Mobile Robot"***, The 17th IFAC (International Federation of Automatic Control) World Congress, International conference, 6 - 11 July 2008. Seoul, South Korea.

8. C. Netramai, O. Melnychuk, C. Joochim and H. Roth, ***"Motion Estimation of a Mobile Robot using different types of 3D Sensors"***, The international conference, ICAS2008 (The Fourth International Conference on Autonomic and Autonomous System) at Gosier, Guadeloupe on March 16-21 2008.

Chapter 2

Literatures Review

The development of 3D structure from data set of 2D cameras has been largely studied for several years and is known in the computer vision community as "structure from motion". The SLAM has most often been used in combination with other sensors rather than with regular cameras. The mapping and localization of the mobile robot community has been proposed for a decade. These topics are interesting for new applications with novel mobile robots. The sensors and algorithms to generate the tasks should cooperate appropriately. Many researchers have adapted the novel sensors in order to use it in mobile robot applications. Well known sensors such as cameras, laser scanners, ultra sonic, infrareds, gyroscopes, global positioning systems and electronic compasses have always been used. This chapter introduces the principle of localization sensors and the literature review of related works. A literature review discusses published information in a particular subject area.

2.1 Mobile robot localization and sensor candidates

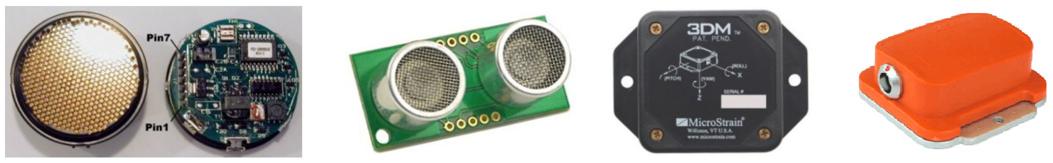
Sensors are important inputs to control the performance of the mobile robot. Normally, sensors used in mobile robot localization are sonar, stereo cameras, laser scanners and video cameras. This section introduces the behavior of each sensor, its pros and cons, and the reasons why the PMD camera is selected for the purpose of our work. There are several different types of sensor candidates which can be used in a mobile robot. This section compares the pros and cons of the PMD camera with other sensing alternatives, and discusses the future trend of the PMD camera.

2.1.1 Sonar

Sonar or ultrasonic sensor stands for Sound Navigation and Ranging, and was initially developed for underwater applications. In Figure 2.1 (a), SRF04 Ultra Sonic sensor [7] is an ultrasonic range finder that can measures a wide range from 3cm to 3m. This ranger finder is a perfect for your robot or any other projects requiring accurate ranging information. As shown in Figure 2.1 (b), SonaMini-S Model is a complete sonar transmitter/receiver systems. It is designed to be used as a rangefinder but can be hacked for a variety of other purposes (communications, etc). The SonaMini is controlled by a small on-board microprocessor and can be used in a variety of modes with a wide range from 15 cm-5.8 meters. Urick [8] described airborne ultrasonic range sensing, in which a single transducer acts as both transmitter and receiver. Sonar sensors use the principle time of flight. Sonar sensor emits sound from the transmitter to the object, and counts the time till the sound echoes back to the receivers. Sonar rate data acquisition is limited by the speed of sound which is 343.2 meters per second at 20 degrees Celsius. The limitation of sound principle is that it depends on the ambient temperature and modulation frequency. It is well know that using sonar sensor is for intelligent obstacle avoidance and bumpers. Gough et al. [9] consider frequency modulation techniques, which developed for underwater and airborne applications. Mobile robot navigation using sonar has been the subject of much research over the last decade. Sonar is heralded as a cheap solution to mobile robot sensing problems because it provides direct range information at a low cost. Initially, sonar sensor was popular for the usage as mobile robot obstacle sensors but it now has also been adapted to mobile robot 3D mapping techniques as well. Leonard et al. [1] adapted the sonar sensor in mobile robot navigation. They analyzed the sonar acquiring results, displaying, and processing a large amount of sonar data acquired in a variety of indoor scenes, with several different robots. They developed the conviction, contrary to popular belief, that sonar is in fact a very good sensor. The to be considered characteristics of the sonar sensor are the data ranges which can be seriously corrupted by reflections [9]. Poor directionality that limits the accuracy in determination of the spatial position of an edge to 10-50 cm, depends on the distance and angle between the obstacle surface and the acoustic beam.

2.1.2 Orientation sensors

Odometry sensor provides the three main outputs that are the rotation angle of pitch, yaw and roll. The odometry sensor is excellent to provide the precise rotation angle. Odometry sensor fusion with encoder can serve for finding actual position and orientation. The advantage of odometry is to provide orientation in matrix, quaternion and Euler formats.



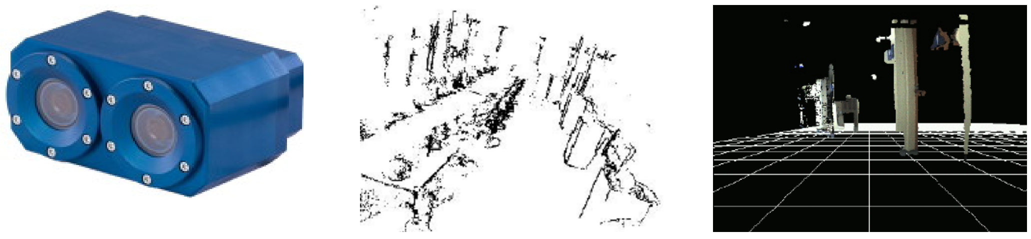
(a) (b) (c) (d)

Figure 2.1 (a) SRF04 Ultra Sonic sensor (b) SonaMini-S Model
(c) Micro Strain [10] (d) MTi [11]

Figure 2.1 (c) 3DM is a 3-axis orientation sensor capable of measuring: +/-180˚ of yaw heading, +/-180˚ of pitch, and +/-70˚ of roll. Orthogonal arrays of magnetometers and accelerometers are used to compute the pitch, roll and yaw (also referred to as heading or azimuth) over a wide angular range. Figure 2.1 (d) MTi is low-power signal processor providing drift-free 3D orientation as well as calibrated 3D acceleration, 3D rate of turn and 3D earth-magnetic field data. The MTi is an excellent measurement unit (IMU) for the stabilization and the control of cameras, robots, vehicles and other (un)manned equipment. Nister et al. [12] presented a system to estimate the motion by using conjunction with information from several sources such as GPS, inertia sensors, wheel encoders. [13] localized robot in map and matching data by taking advantage of whatever surface features. Their research concentrated on metrically precise maps that are derived from dense range readings, such as those provided by sonar arrays, scanning laser range finders or stereo vision systems.

2.1.3 Stereo and single camera

The general camera provides the two-dimensional image which can only be used for object detection and obstacle avoidance. The depth information cannot be received from the normal vision. However, the technique, which improves the vision can measure the depth data namely the stereo camera. This technique uses two CCD cameras to fix the position. If we know the intrinsic parameters as well as the base line between two cameras, the depth data then can be measured from the triangle formula. The stereo camera is also very well known in mobile robot application. However, the limitation of stereo is that it is not an active sensor. It calculates the depth from a passive technique. That means the accuracy depends on the ambient light. If at that time, the ambition light is not sufficient, the depth data is in question and cannot calculate accurately. The stereo camera is one of the most well known depth enabled camera which exists. It has been widely used in robotics and automation applications for decades, particularly the short baseline stereo camera, which is small in size and can be easily integrated into many applications. Whereas the problem of motion estimation using a stereo camera is perfectly feasible, the depth measurement precision of the stereo camera is usually limited due to its stereoscopic design and the software computational complexity. High precision depth measurement can be achieved using large baseline stereo system with high resolution imagers and complex algorithms. Nevertheless, this would hinder the use of such systems on many applications where space and computational power are limited. Therefore, one always has to come to a good compromise in order to derive the most benefits from the stereo camera system. Due to the fact that the PMD and stereo camera do share one important purpose in common, namely their purpose to acquire the depth of the 3D scene, there have been some works which compare and combine the output of both camera systems in several ways to gain advantages over the use of a single camera system. Figure 2.2(a) the stereovision camera namely the MobileRanger C3D [14]. It could measure the depth for demanding applications such as mobile robot navigation object tracking, gesture recognition, 3D surface visualization and advanced human computer interaction. MobileRanger can be used as a 3D sensor in robot navigation or object sensing. Specular reflections occur when the angle between the wave front and the normal to a smooth surface are very large [15]. Figure 2.2 (b) shows his result, the 120 frames captured in a meeting room. [16] presented a hybrid technique for constructing geometrically accurate, visually realistic planar environments from stereo vision information. The technique is unique for estimating camera motion from two sources, i.e. range information from stereo and visual alignment of images. Figure 2.2 (c) shows his output that the reconstructed 3D mapping from stereo camera. The output is not only provides visually realistic images but also demonstrates the depth information.



(a) (b) (c)

Figure 2.2 (a) Stereovision [14] (b) 3D map from stereo camera [15]. (c) Reconstructed 3D mapping [16]

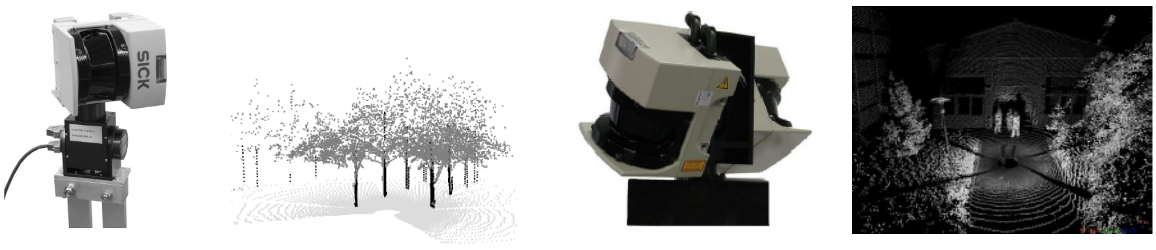
[17] used a mobile robot to generates a mapping in a consistent, globally correct map for outdoor environments. They use the stereo camera to demonstrate a complete system for off-road navigation in unstructured environments. Their robot could run at an average speed 1.1 m/s. [18] presented an algorithm for creating globally consistent three-dimensional maps from depth fields produced by stereo camera based range measurement systems. The point-to-plane variant of ICP is used for local alignment, including weightings that favor nearby points and a novel outlier rejection strategy that increases the robustness for this class of data while eliminating the burden of user-specified thresholds. [19] Employed edge points to perform SLAM with a stereo camera. The edge-point based SLAM is applicable to non-textured environments since plenty of edge points can be from even a small number of lines

obtained. They proposed a method to estimate camera poses, and build detailed 3D maps robustly by aligning edge points between frames using the ICP algorithm. [20] used the Rao-Blackwellised Particle Filter (RBPF) for the class of indoor mobile robots equipped only with stereo vision. They purposed to construct dense metric maps of natural 3D point landmarks for large cyclic environments in the absence of accuracy of landmark position measurements and motion estimates. Their work differs from other approaches because landmark estimates are derived from stereo vision and motion estimates are based on the sparse optical flow. [21] [22] [23] presented a new algorithm for determining the trajectory of a mobile robot, and created a detailed volumetric 3D model. The algorithm exclusively utilizes information provided by a single stereo vision system. The algorithm can deal with both planar and uneven terrain in a natural way without requiring extra processing stages or additional orientation sensors. However, the SLAM process is unstable in non-textured environments, where sufficient corner-like features cannot be extracted. Since many environments are non-textured, the importance of alternative feature forms such as lines is indicated in [24] [25].

In addition to the stereo camera, the single camera can also be used in SLAM application. [26] presented a method to incorporate 3D line segments in vision based SLAM. A landmark initialization method that relies on the Plücker coordinates to represent a 3D line is introduced: a Gaussian sum approximates the feature initial state and is updated as the new observation that is gathered by the camera. [27] used line features in real-time visual tracking applications, which are commonplace when a prior map is available. They described how straight lines can be added to a monocular Extended Kalman Filter Simultaneous Mapping and Localisation (EKF SLAM) system. In addition, they also presented a fast straight-line detector that hypothesizes, and tests straight lines connecting detected seed points. [28] described how to initialize new landmarks to observe mapped landmarks in subsequent images, and to deal with the data association challenges of edges. Initial operation of these methods in a particle-filter based SLAM system is presented. [29] described a principled, Bayesian, top-down approach to sequential SLAM, and demonstrated robust performance in an indoor scene. The image features used, being 2D patches, are limited in viewpoint-variance. The algorithm would benefit from the use of features such as planar 3D patches whose change in image appearance that could be better predicted from different viewpoints.

2.1.4 Laser scanner

The laser scanner rangefinder appears to be the most popular one, as many researchers have used it in real time mobile robot exploration. The laser range finder is a powerful sensor to obtain the depth data. The laser range finder is mainly used with mapping and localization techniques. The laser beams are transmitted, and their echoes are received, which is quite similar to time-of-flight concept. The laser scanner source is based on light, and the exact value speed of light travels in vacuum is 299,792,458 m/s. The laser scanner has a very precise resolution, 10 mm. On the other hand, the laser beam can provide an accuracy depth over hundred meters, on the other hand, its limitation on output power makes it seem inaccurate though, and the sizes are comparatively large and heavy. One disadvantage of scanning laser range finders is the slow data acquisition rate due to the mechanical scanning mechanism. Hence, the power is lost for moving the mechanical parts. [30] [31] [32] [33] [34] [35] proposed the real time accuracy 6D SLAM using laser scanner with a powerful fast variant of the ICP algorithm registers 3D scan taken by a mobile robot into a common coordinate system. Measurement rate is about 3.4 seconds $(181 \times 256\, points)$. [36] used the data from laser scanner to present an alternative matching 2D range scans the Normal Distributions Transform. He subdivided the 2D plane into cells. Each cell is assigned a normal distribution. The result of the transformation is a piecewise continuous and differentiable probability density, that can be used to match another scan using Newton's algorithm. One drawback from laser scanner is the field of view. The conventional laser scanner provides only one row with 180°, which is not sufficient for 3D mapping. Wulf et al. [37] [38] presented the modification technique to raise up the field of view and performance of laser scanner. They improved the infrastructure basement by integrating the servo motor, actuator and processing unit. Sensor output can provide 3D geometric in real time with scan times 1.6 second for an apex angle of $180° \times 180°$ to 5 seconds for $360° \times 180°$ without distortion. Figure 2.4 (a)-(b) shows the scanner models, which are used, and output. Figure 2.4 (c) shows the figure from the Fraunhofer IAIS 3DLS-K, and consists of two SICK LMS 291 2D laser range finders mounted on a rotatable carrier [39].



(a) (b) (c) (d)

Figure 2.3 (a) 3D scanner consisting of a 2D laser range sensor and a servo drive [37] (b) Yawing scan in outdoor application [37] (c) The Fraunhofer IAIS 3DLS-K scanner [39] (d) Output from the Fraunhofer IAIS 3DLS-K scanner [40]

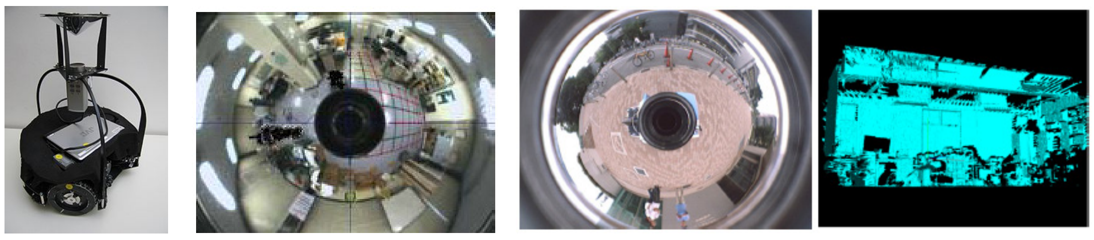
This carrier is continuously rotated around the vertical axis. Depending on the current orientation, the 2D laser range scans of the scanners are transformed into a sensor-centric coordinate frame. [40] used this scanner to show transformed scans that are aggregated to form a local 3D point cloud as Figure 2.4 (c). They presented a sensor setup for a 3D scanner that is especially appropriate for a fast 3D perception.

Their robot moved and used fast 3D perception for collision avoidance. The representation of these maps is defined in a way that they can be used with any algorithm for SLAM or collision avoidance, which is operating on 2D laser range scans. [41] extracted a line map from the sequence of points in each laser scan using a probabilistic approach, and then computes *virtual corners* between two lines in the same line map. The movement of the robot is estimated from correspondences of virtual corners between the two line maps. [42] considered the computation of motion strategies to efficiently build polygonal layouts of indoor environments using a mobile robot equipped with a range sensor. They handled the model that was being built for the robot which was going to be used for the next sensing operation performance. The paper argues that the polygonal layout is the convenient intermediate model to perform other visual tasks. [43] created maps for natural environments, it is however necessary to consider the 6DoF pose case. They demonstrated the functionality of estimating the exact poses and their covariance in all 6DoF, leading to a globally consistent map. The correspondences between scans are found automatically by use of a simple distance heuristic.

2.1.5 Omnidirectional camera

The omnidirectional camera [44] is a camera that provides 360 horizontal degree field of view, whereas the normal 2D camera in generally has less than 180 horizontal degree. The Omnidirectional sensor can capture 360 horizontal degrees without calibration. This reduces the maintenance and hidden costs. It applies in many applications such as traffic transportation, outdoor navigation and the robotics community. The high efficiency video visual odometry output using a single camera has been a goal for many years. [45] presented a robust SLAM framework using a single camera catadioptric stereo system that composed of vertically aligned two hyperboloidal mirrors and a CCD camera. They improved the conventional stereo cameras with a narrow field of view by fusing a single camera catadioptric stereo system, and proposed the rectification algorithm that aligns the mirrors and a camera parallel to one another in order to satisfy the single viewpoint (SVP). The omnidirectional camera dominates the panoramic area and robotics. The concept of omnidirectional camera is to capture the light from all direction falling onto the focal point from a full sphere shape. The output of omnidirectional camera can provide the panoramic scenario in real time without any register image frame technique. Thus, the output gives the better quality and frame rate in the same field of view. The omnidirectional camera is always use in visual odometry as well as simultaneous localization and mapping, SLAM.

Scaramuzza et al. [48] proposed the motion estimation of the vehicle, which is estimated by using the property of the ground plane projection into two different camera views. The goal is to estimate the motion of the vehicle in outdoor environments over long distances. Due to the large field of view (FOV) of the panoramic camera, interesting points (all around the car) are extracted, and matched from the pairs of consecutive frames. Kiyokawa et al. [47] proposed a visualization and interaction technique for remote surveillance using both 2D and 3D scene data acquired by a mobile robot equipped with an omnidirectional camera and an omnidirectional laser range sensor.



(a) (b) (c)

Figure 2.4 (a) Mobile robot with omnidirectional cameras [44] (b) Image is projected on camera [46] (c) Output from omnidirectional cameras [47]

An egocentric-view is provided using high resolution omnidirectional live video on a hemispherical screen. As depth information of the remote environment is acquired, additional 3D information can be overlaid onto the 2D video image such as a passable area and the roughness of the terrain in a manner of video see-through augmented reality. A few functions to interact with the 3D environment through the 2D live video were provided, such as path-drawing and path-preview. The path-drawing function allows planning a robot's path by simply specifying 3D points on the path on screen. In addition, a miniaturized 3D model is overlaid on the screen providing an exocentric view which is a common technique in virtual reality. In this way, their technique allows an operator to recognize the remote place and to navigate the robot intuitively by seamlessly using a variety of mixed reality techniques.

2.1.6 PMD camera

The PMD camera is the novel 3D sensor. The base sensor that uses the Photonic Mixer Devices entitled PMD sensors as a smart pixel can be an integration element for a 3D imaging camera on a chip based on standard CCD- or CMOS-technology. The main feature is an array sensor which can measure the distance to the target in parallel without scanning. The key execution is based on time of flight principle. A light pulse is transmitted from a sender unit, and the target distance is measured by determining the turn around time back to the receiver. According to the speed of light, the interval distance can easily be calculated. The detail of the PMD camera will be more thoroughly described in the next chapter.

2.2 Conclusion

In this chapter, the principle of several sensors for the purpose of mobile robot localization is introduced. The well-know sensors, which have been used, are presented. Many types of sensors, for example the Sonar, Stereo, Vision, Laser scanner, Odometry, Omnidirectional camera as well as the PMD camera have been presented. Each sensor has its pros and cons. The sonar has the lowest cost for generating the robot navigation. The output range is acceptable, but it is still to follow at various ambient temperatures. Stereo and vision systems are attractive sensors, which can be used in obstruction avoidance, and can show the environment texture. It should be noted, however that these sensors have the limitation of the ambient light. If the ambient light is not sufficient, both sensors will give the wrong data or worst case and they cannot provide any information. The Omnidirectional camera is a very creative novel sensor that they can capture 360 degree by using only one camera, however, the high processing compensate mathematic are required. Moreover, the ambient light also affects the quality of the data. The Laser scanner is the longest and most accurate distance detection, but the acquisition for whole volume requires the mechanical part and is time consuming. The PMD is the newest sensor using the time of flight principle. The capturing frame rate is high and accurate. Nevertheless, the PMD output volumetric is still limited, and the several kinds of textures are affected by noise disturbances. For those reasons, the fusion sensor seems to be the best solution in order to receive the best sensor information. This means that we combine every pro and reduce cons to any sensor. In the next section, we will present the fusion between the PMD and 2D camera in order to use these fusion output in the mobile robot navigation system.

Chapter 3

Mobile Robot Hardware

This chapter presents an overview of mobile robots, which are used in this thesis. The details of the mobile robot construction, sensors, wireless communication as well as the software, which are used to handle all features including the server PC and robot client are introduced. The Mobile Experimental Robots for Locomotion and Intelligent Navigation, MERLIN, have been designed based on several models, and adapted in many environmental missions. The previous prototypes were designed and tested using radio wireless radio communication protocol, speed and steering control algorithm. The compactable software and hardware are improved in order to enhance the reliability and suitability for 3D mapping mission.

3.1 Data communication and interfacing method

Data communication plays a decisive role in the communication between mobile robots and users. Several kinds of data interfaces are very attractive such as WLAN, radio link, blue tooth and RS-232/485 communication. The blue tooth and WLAN are powerful for the big and fast data transfer package. The radio link can easily communicate with the microcontroller and can also reduce the process steps. RS-232 is prominent in simple access and can easily find the interface tools, which are included in every window version. The overall process has the server station (computer based user) and client station (computer on mobile robot). The mini computer is needed because the image processing and wireless data package are too complex to use only microcontrollers. The server station uses LABVIEW to create Graphic User Interface (GUI) for interfacing between the user and the robot. LABVIEW is used to receive commands from a joystick (manual mode) via USB, the keyboard and the mouse. It sends the user commands to RS-232 wireless module via serial communication. The 8 bits data command the robot movement. The 3DM sensor is mounted on the top in order to avoid the electromagnetic noise from the motor and high current from the battery. The PMD camera is processed in real-time by using an onboard mini-computer. Appendix A shows the detail of the mini-PC, MICROSPACE-PC41. Since the mini-PC has a high performance and efficient input/output ports, it was suitable to be used in my research.

Serial port	: micro controller communication
USB	: 2D camera interface
Firewire	: PMD camera interface
Wireless LAN	: remote desktop monitoring

The goal of this research is to have an affordable autonomous mobile robot for indoor environments that is able to generate 3D geometry maps of natural scenes from the scratch. It should work completely autonomous with or without manual control, with landmarks or beacon.

3.1.1 Three orthogonal magnetometers, 3DM

Generally, the pose estimation can be calculated by using the sensor or image processing. The image processing approach uses the optical flow estimation to test the pose in each angle. The three dimension orthogonal magnetometer, namely 3DM is the sensor, which can measure the absolute angle from the three axis: roll, pitch and yaw. The measurement refers to the magnetic earth's pole and gravitation. The output is capable to measure angles from 0° to 360° degrees on yaw, 0°-360° degrees on pitch and -70°-+70° degrees on roll. It calculates the yaw angle using the magnetic field from the earth and compensates the errors using the accelerometers. The data is sent via the serial communication (COM Port) with a board rate of 9600 bit/sec. The 3DM sensor is used to compensate the pose estimation error by optical flow approach. Moreover, the pose estimation from the image cannot be calculated because there is an error due to the bad environment. This sensor is used for position estimation during mobile robot navigation. It is also sometimes used to describe the distance traveled by a vehicle. The 3DM is used by some robots independent of the fact whether they are legged or wheeled robot. It is used to estimate their positions relative to the starting location. Odometry is the use of data from the movement of actuators to estimate change in position over time. This method is often very sensitive to errors. Rapid and accurate data collection, equipment calibration and processing are required in most cases for odometry to be used effectively. Appendix B shows the data sheet and the acquired data via RS-232 source code based on VC++ programming.

3.2 Mobile robot candidates

The research topics in our group intend to co-operate multi distributed mobile robots in order to generate the real-time outdoor 3D mapping. The mobile robots in our group are separated in two groups i.e. flying airship and ground based mobile robots. The flying airship investigates the predictability of the overall terrain as viewed from the top. The information of the terrain such as flat, roughly, barbed wire as well as dead end are transferred to the ground based mobile robots. The 3D mapping then is built efficiently and is faster than only to ground based mobile robots. Moreover, the flying airship is one kind of ground based mobile robots, which is suitable for such terrain. These following section presents an overview of mobile robot candidates for this research. The distributed multi cooperative outdoor SLAM is the adapted concept from indoor SLAM that has been developed. Our task is one part of this big project, which is in one part of the cooperative project. The mobile robot, which will join the future project, has four types of different prominent tasks. We describe the kind of those robots.

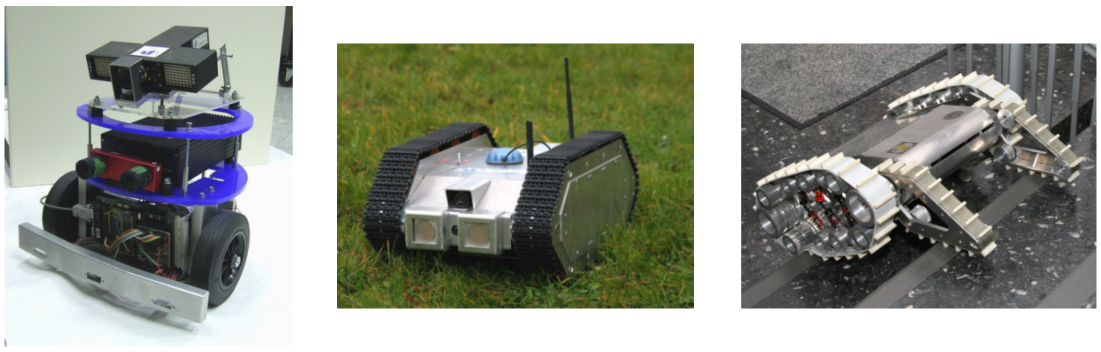

(a) TOM3D (b) Tank MERLIN (c) Track MERLIN

Figure 3.1 The types of ground based mobile robots

(a) TOM3D is a two wheels drive mobile robot. It is equipped with a very high resolution encoder, which provides a high precision locomotion. It can rotate by itself at 360° without dead area and with high performance for autonomous obstacle avoidance. The mini-computer receives the image data from the cameras, meanwhile calculating the best path for the robot movement. TOM3D consists of two main layers. The lower layer is equipped with an embedded PC, a 16-bit microcontroller and a driving/steering motor. The top layer equips with 2D and PMD camera. Both cameras connect the embedded PC which is on the below layer. TOM3D inherits the distinctive features of these building blocks by the integration and adds some features that are useful for this purpose but not available for the manual wireless joystick control.

(b) Tank MERLIN is an armored vehicle, adapted from a military tank, which is used in military. It can be used to climb over barbed wire and rough terrain. The ultrasonic sensors are used for an autonomous locomotion. The GPS is also used in order to get the outdoor position. Moreover, the installed 2D camera is used for image processing requirements.

(c) Track MERLIN has the special flip arms in front of the robot. It has prominent arms used to climb up sharp slopes such as hills or stairs. The main power drive uses the high performance I2C motor with installed high resolution encoder in order to precisely control the arms and moving track. All three types of the ground based mobile robots distribute their own suitable tasks and terrain.

3.3 Mobile Experimental Robots for Locomotion and Intelligent Navigation (MERLIN)

The goal of this research is to have an affordable autonomous mobile robot for indoor environments that is able to generate 3D geometry maps of natural scenes from the scratch. It should work completely autonomous with or without manual control. MERLIN inherits the distinctive features of these building blocks by the integration and adds some useful features for this purpose. MERLIN is a four wheels drive, high speed movement. Its tasks can be used in smooth and rough areas, wide surfaces and few obstacles. It communicates to a workstation via a wireless communication module. The mobile robot can run into two modes. In the first mode, i.e. the manual mode, the microcontroller receives the command from the joystick via the RS-232 wireless module. The data is sent as package, 8 bits each for controlling the direction of the robot by user command. In the second mode, i.e. the autonomous mode, the mobile robot localizes autonomously without any command from the user. It locates by itself using only the information from the sensors, the encoder and 2D/3D images. Since the digital 3D models of the environment are important in order to localize the position of the mobile robot itself in an unknown environment, the PMD and high resolution 2D camera are equipped. The 3D mapping and object recognition can be achieved by using the MERLIN. The MERLIN is the selected robot, which is used in our project. We will explain the details in the next section.

The RST, "Institut für Regelungs- und Steuerungstechnik" research works [49] are focusing on the unmanned autonomous mobile robots, flying robots, industrial robots and intelligent control. The mobile robot applications in our group aim to enhance the intelligent potential operation. The case studies of the obstacle avoidance using classical, fuzzy and neural network control as well as the image processing have been studied, and implemented simultaneously. The robot vision is one of our attractive points. Not only the object detection but also the 3D SLAM for the purpose of mobile robot exploration is implemented. The stereo camera as well as the Photonic Mixer Devices (PMD) camera are our concentrated vision sensor research goal. We realize that the automatic control and mobile robot applications are very attractive tasks, which requires researches and developments continuing forward, as the high performance mobile robots nowadays should operate by themselves in many applications. The variety of sensors such as wireless communication, GPS, ultrasonic sensors, infrared sensors, intelligent obstacle avoidance, vision systems as well as self localization techniques are generally used on the mobile robots in order to accomplish all the tasks. Many control procedures have been studied, implemented and tested strictly in real mobile robot applications. The integration between the passive sensors and visions sensor are cooperated in a suitable way in order to yield a robust and reliable output.

3.3.1 Data communication structure

Figure 3.2 depicts the process of data communication. For the *Client robot,* the microcontroller sends the PWM signal to the driving circuit (full H-bridge) to control the DC motor and stepping motor for driving and steering operation, respectively. The encoder signal is sent back to the microcontroller via counter port to calculate the robot speed. Microcontroller exchanges the data to the mini-PC via serial communication port RS232. The mini-PC is the main processing core. The mini-PC required two serial communication ports for connecting with 3DM, COM1 and microcontroller, COM2.

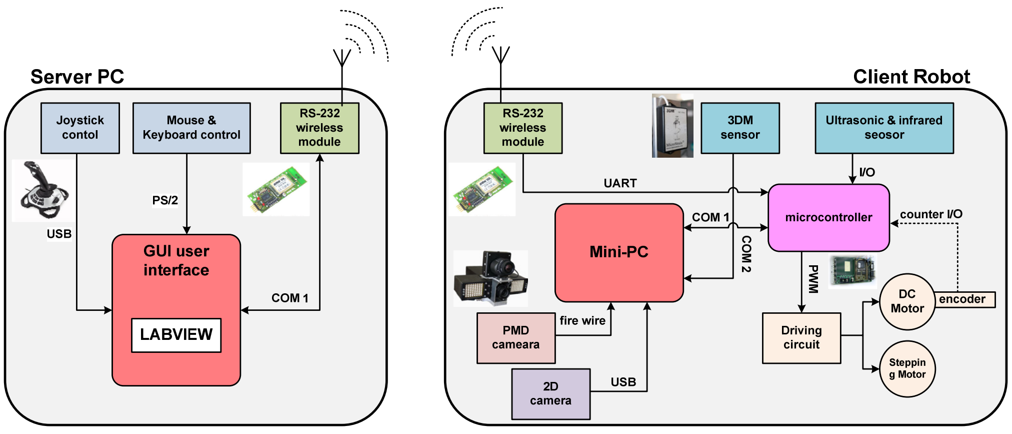


Figure 3.2 Overall control system structure

The PMD and the 2D camera send the image information via firewire and USB, respectively. *Server PC* is the main computer station for the user interface. LabVIEW is the most important software to control all features. The Graphic User Interface (GUI) is built in order to be a tool to interface between the user and low level hardware. LabVIEW is the powerful tool to generate the easy GUI and to interface with hardware. The most appropriate way to freely control the robot movement is to use joystick control. The joystick connects with the computer server via a USB port. LabVIEW receives the XY axis joystick command and twelve buttons with reserve using in any demands. Mouse and keyboard also can also transmit the command to control the robot. Finally, all data are sent to the robot client via RS-232 wireless module.

3.3.2 LabVIEW and joystick interface

LabVIEW is a program development environment from National Instruments, which is not a *text-based* language of code but a *graphical* programming language like a block diagram programming form. LabVIEW is like C or BASIC, which is a general-purpose programming system with extensive libraries of functions for any programming task. LabVIEW includes libraries for data acquisition and serial instrument control, data analysis, data presentation and data storage. In particular, conventional program development tools and hardware interface are included. One prominent feature of LabVIEW is that it is easy to interface with hardware that could reduce the working time for programming, including acquiring data from the joystick via the USB port. Several buttons and axis controls can easily be captured on one block library.

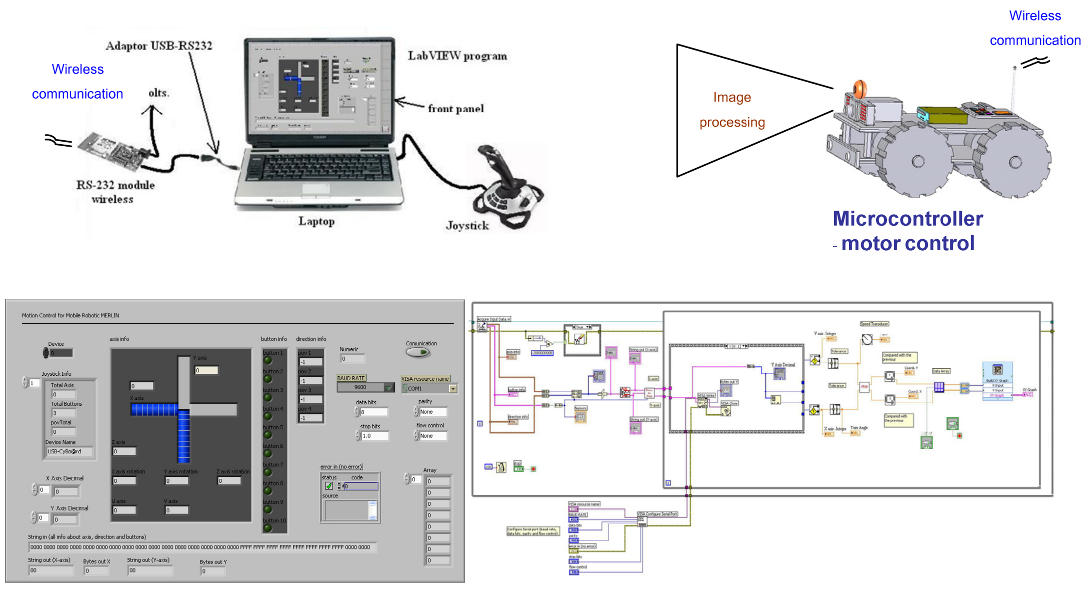


Figure 3.3 LabVIEW Interface

Thus, for server part, we use LabVIEW, which is the main core program to be interactive between users and mobile robot. Figure 3.3 show some parts of the LabVIEW program. The main purposes to use LabVIEW are to capture the data from the joystick and to send this command to serial connection. LabVIEW uses the VISA protocol for sending the data to the serial port. The important thing is the configuration (baud rate, start/stop bit, data) that must be synchronized to the receiver part (microcontroller). The data is sent in packages, each package contains 8 bits. The microcontroller uses the serial interrupt routine to sort out the data in the package e.g. steering direction and robot speed. All data packages are sent via RS232 wireless module to the RS-232 wireless module, connected with the microcontroller on the robot. Moreover, the LabVIEW page also shows the speed and basics of the mobile robot behavior. All details in each LabVIEW block modules can be seen in [50] and [51].

3.3.3 Microcontroller

The Infineon C167CR-LM, is a 16-bit microcontroller of the Infineon C166 family. Figure 3.4 illustrates the Infineon C167CR-LM. It was designed to meet the high performance requirements of real-time embedded control applications. In addition, it also supports C language programming by Keil compiler, which is suitable for developing the complex systems. The architecture of this family has been optimized for high instruction. The Infineon C167CR-LM is an improved representative of full featured 16 bit single-chip CMOS microcontrollers.

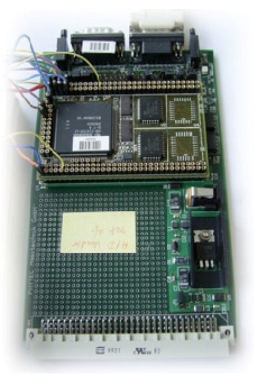


Figure 3.4 Microcontroller Infineon C167CR-LM

The architecture combines the advantages of both RISC and CISC. The sum of the features, which is combined that result it is a high performance microcontroller. The Infineon C167CR-LM does not only integrate a powerful CPU core and a set of peripheral units into one chip but it also connects the units in a very efficient way. Furthermore, since the mobile robot system is complex, the microcontroller has to receive the data from the mini-computer, surrounding sensors and send commands to the driving dc motor. As a result, the microcontroller must be fast enough for the calculation. Therefore, it should not be lower than 16 bit. Thus, the Infineon C167CR-LM is used in this research.

3.3.4 RS-232 wireless module

The Wireless “WI Z-434-SML-I.A” module (Figure 3.5) is a transceiver and receiver for point-to-point data transfer in the full-duplex mode. They are the right solution for all those applications in which the so called “virtual cable” has to be implemented, like this case, the RF connection of two systems with RS232 output (for instance two PCs.). The “WI Z-434-SML-I.A” module is available in a 12V and 5V version voltage supplied through the input 10 of the data connectors. The W232 adapter can be connected to the module. It foresees inside an integrated circuit which shall care for the conversion of the electric signals from RS232 logic to TTL logic and vice versa.

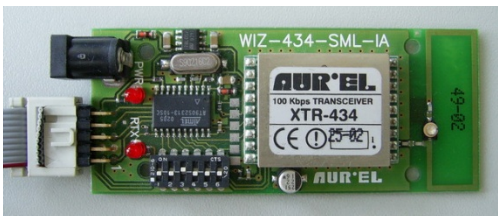


Figure 3.5 RS-232 wireless Module

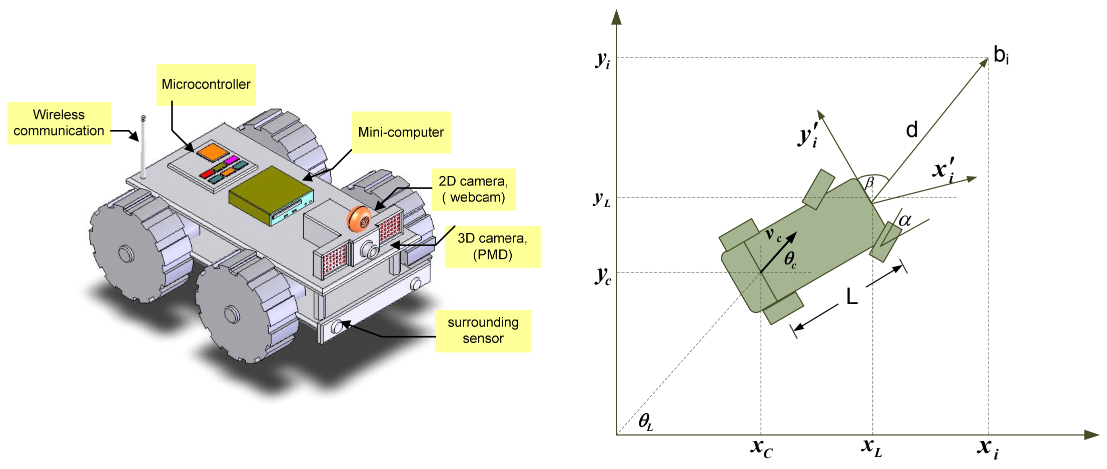
Transceiver modules which are suitable to replace a serial wire connection with RF wireless [433.92 Mhz] Half-duplex. The transmission speed is selectable at 9600, 19200, 57600 and 115200 bps. Max allowed data packet length is 96 bytes. The module has small dimensions (4 x 9 cm) and its card integrates a 100 kbps XTR transceiver, a microprocessor which administrates the RF synchronizing protocol and a tuned antenna realized on a printed circuit. Besides, this component also has a selected dip-switch with six positions and two indication LED for changing the baudrate. The module is enclosed in a special case, which complies with the EN 661000-4-2 rule.

3.3.5 Merlin model

MERLIN is sorted in wheeled robot types. It means the power source to move the robot use motorized wheels. This robot type is simpler than using legs because it is easy to design, to construct and certainly to program. It is suitable to use in flat terrains. Even though, wheeled robots locate themselves excellently, they still have areas lacking in this ability. They cannot locate well in cut and obstructed terrain such as rocky areas, in collapsed buildings or low friction surfaces. Thus, there are several kinds of wheeled robots. A popular one is differential steering. Robots can have many wheels, but the minimum for stabling the dynamic balance is three wheels. Additional wheels can be added for more robustness or more payloads efficiently. Wheeled robots separately use driven wheels for movement, named differential steering [52]. It can change the direction by rotating each wheel at different speed. They may have other wheels that are used only for keeping the balance.

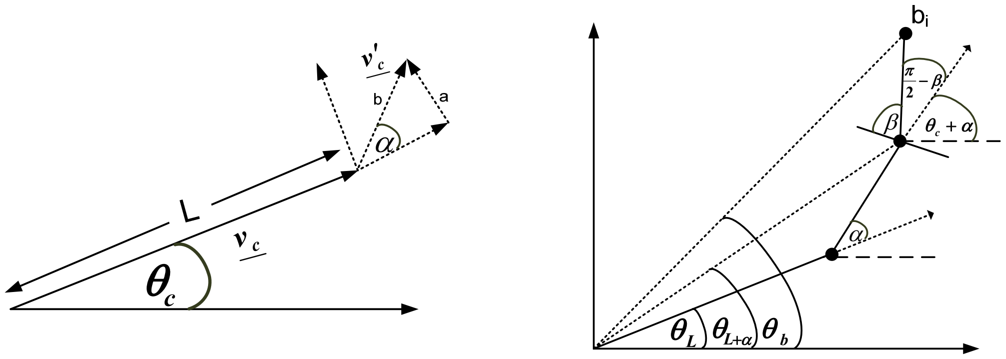
Two wheeled robots: are harder to balance than other types because they must keep moving to remain upright. The center of gravity of the robot body is kept below the axle. Usually, this is accomplished by mounting the batteries below the body. They can have their wheels parallel to each other, these vehicles are called bicycles, or one wheel in front of the other, tenderly placed wheels. Two wheeled robots maintain their upright position by driving in the direction that the robot is falling. For balancing, the base of the robot must stay under its center of gravity. For a robot that has the left and right wheels, it needs at least two sensors. A sensor is used to determine angle and wheel encoders, which keep the track of the position of the platform of the robot.

Three wheeled robot: The three wheeled robot has two types of driving. Firstly, different two steered wheels and one additional free rotating wheel are used to keep the body in balance. Second, two wheels are powered by a single source and one additional wheel is the steering control. In the case of different steered wheels, the robot direction may be changed by varying the relative rate of the rotation of the two separately driven wheels. If both wheels are driven in the same direction and at the same speed, the robot will go straight forward. Otherwise, depending on the speed of rotation and its direction, the center of rotation may fall anywhere in the line joining the two wheels.



(a) Prototype of MERLIN robot (b) Car-like model

Figure 3.6 Car-like mathematical model



(a) Angular velocity (b) World coorinate transformation

Figure 3.7 World coordination and robot position

Four wheeled vehicles: (two powered, two free rotating wheels), this type is similar to the differential drive but they have two front rotating wheels for a more reliable balance. This is more stable than in three wheeled robots because the center of gravity is really a rectangle shape instead of a triangle. It has a more stable movement, especially when moving in a sharp area (more roll, yaw angles). The center of gravity remains inside the rectangle form; moreover, it provides more useful space. The car-like steering method allows the robot to turn in the same way a car does. It provides a lacking area, but it is very stable for a fast movement.

$$\begin{aligned} \tan(\alpha) &= \frac{a}{\left|\underline{v_c}\right|} \\ \dot{\theta} &= \frac{a}{L} = \frac{\left|\underline{v_c}\right|}{L}\tan(\alpha) \end{aligned} \tag{3.1}$$

$$\begin{bmatrix} \dot{x}_c \\ \dot{y}_c \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \left|\underline{v_c}\right| cos\theta_c \\ \left|\underline{v_c}\right| sin\theta_c \\ \frac{\left|\underline{v_c}\right|}{L}\tan(\alpha) \end{bmatrix} \tag{3.2}$$

$$\underline{z} = \begin{bmatrix} Translation \\ Rotation \end{bmatrix} = \begin{bmatrix} d \\ \beta \end{bmatrix} = \begin{bmatrix} \sqrt{[\acute{x}_i{}^2 + \acute{y}_i{}^2]} \\ atan\left(\frac{\acute{y}_i{}^2}{\acute{x}_i{}^2}\right) \end{bmatrix} \tag{3.3}$$

$$\theta_L = \beta - \frac{\pi}{2} + \alpha \tag{3.4}$$

$$\underline{z_{bi}} = \begin{bmatrix} \sqrt{[x_i{}^2 + y_i{}^2]} \\ \theta_L \end{bmatrix} \tag{3.5}$$

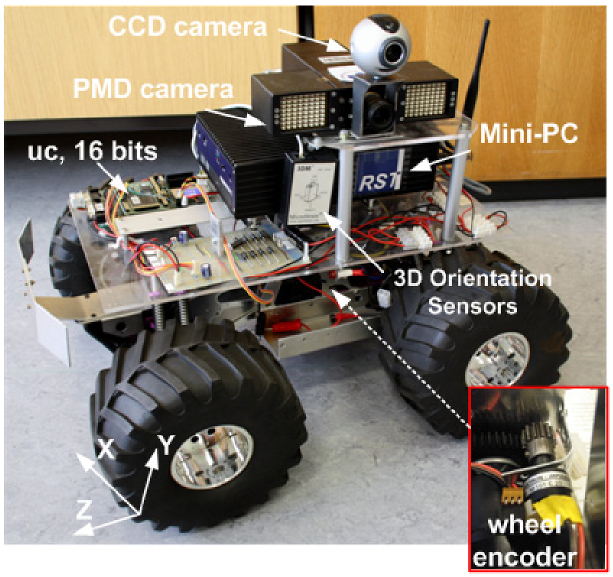
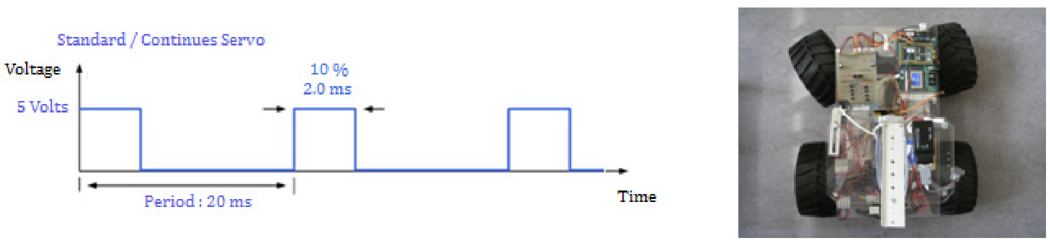


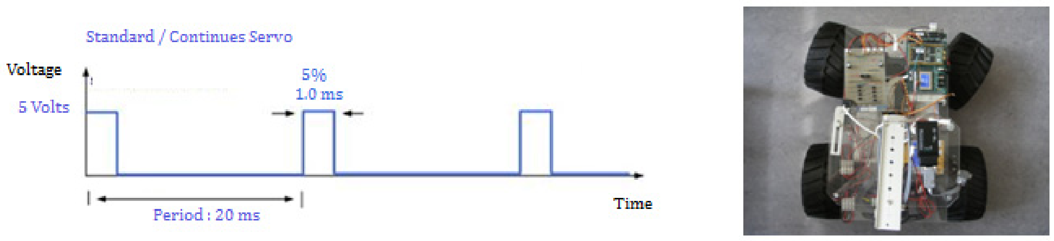
Figure 3.8 MERLIN robot

Figure 3.6(b) shows the mathematical model of the car-like model. In order to understand how to calculate the locomotion of the car-like model. The position of the robot consists of two parameters, i.e. translation and rotation. Eq. (3.2) $\dot{x}_c, \dot{y}_c$ are the vector of translation and θ is the rotation of the robot, which is controlled by the front steering angle(α). From eq.(3.3) we can use to find the centroid position of the robot. Due to the fact that the robot has a long dimension (L), the centroid position is different from the robot front. The next step is to find the update front robot position, (d) in eq. (3.4) is the front robot position, which comes from the relation of the translation and rotation of previous position. Eq. (3.5) explains the final equation to find the position of the robot. The related position of the detected objects and the robot MERLIN in Figure 3.8 is equipped with a PMD, a CCD camera, a 16-bit microcontroller, a RS-232 wireless module, a 3D compass and an embedded PC. All image data from CCD and PMD camera are processed in the embedded PC. The output from image processing is sent to the microcontroller. It sends commands to control a mobile robot in order to autonomously avoid the obstacles. Furthermore, MERLIN can be controlled in manual mode by receiving the command from the joystick which is handled in the LabVIEW core program.

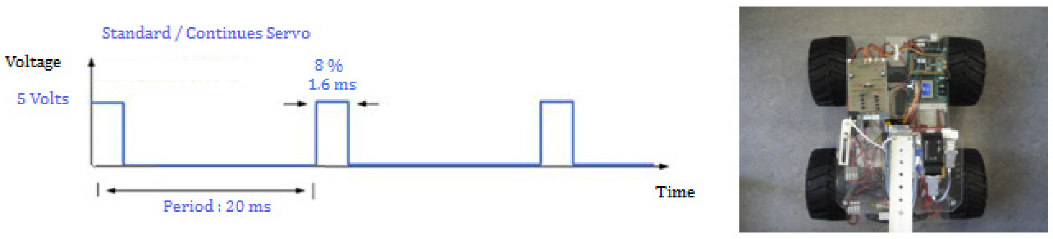
3.3.6 Steering control using servo motor



(a) Turn left maximum 10% duty cycle



(b) Turn right maximum 5% duty cycle



(c) Straight forward, 8% duty cycle

Figure 3.9 Steering control

The servo motor is the main source used for steering the two front wheels. Pulse-width modulation (PWM) is the command to control the position of the robot. The position is controlled by adjusting the duty cycle and speed is controlled by adjusting the frequency of PWM. The PWM could be generated from interrupting routine in the microcontroller. To control MERLIN, the PWM frequency is set to 50 Hz. The servo motor can rotate from 0°-180° degree but the ideal robot steering is also fixed by mechanical part. The maximum angle is an acquisition from the experiment. Then, the maximum left angle is obtained with a duty cycle of 10%, 2 ms. The maximum angle to the right is 5%, 1ms and with a duty cycle of 8% in order to keep the straight direction. The following condition represents the output of the steering control.

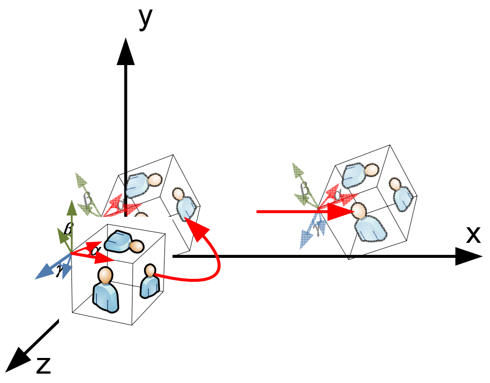
3.4 Conclusion

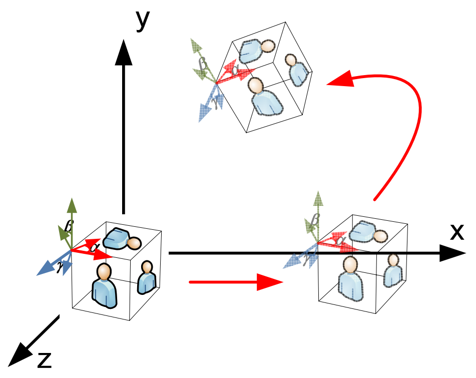
This chapter describes the overall hardware system, how a mobile robot works and how the mobile robot can be controlled. The controller unit is separated into two parts; i.e. the server and the client. The server part is the main computer that interacts with the user. The client part is the computer unit on the mobile robot. Several kinds of mobile robot candidates are presented. Each robot has its pros and cons. In this research, MERLIN has been selected because we attempt to generate the 3D mapping, for which the accurate control is required. MERLIN is a car-like model, which obviously has high speed locomotion. The mathematical model of car-like is described in order to clearly understand how we can calculate the robot position and its locomotion. Steering uses two front wheels. The basic assistance circuit is also introduced for the 3DM, RS-232 wireless module, microcontroller and minicomputer. LabVIEW is used for the main program in the server part because it acquires the data from the joystick, and sends the command data via the RS232-wireless module.

Chapter 4

Three Dimensional Images Enhancement and Object Detection

This chapter describes the fundamental concepts of image processing and the preparation of the raw data before approving the SLAM technique. The raw data from PMD camera has to approve and reduce noises before use. The 2D data is fused to the 3D data in order to get more reliability of 3D mapping. The 2D information from fusion is useful for users in order to distinguish which kind of scenario we want. Moreover, the high quality 2D images are used for objects detection applications which are crucial in mobile robots nowadays. The three dimension images enhancement and object detection method (Haar-like features) will be proposed. The PMD camera is a main sensor used. The PMD camera models and principles will be described in this chapter. Then theory will be presented in order to understand the aim of this thesis. It is indispensable to be aware of the three basis dimensional geometric theories, in order to comprehend the pose estimation phenomenal. In this chapter, the mobile robot locomotion and the vision processing are mentioned. The key regulations are the rotation and translation matrix between two imaging frames. The first basic concept is a rotation matrix. A rotation matrix is a matrix that multiplies with vector rotations while preserving its length. One of the toughest computer graphics problems is understanding the effects of combined three dimensional transformations. Figure 4.1 shows the 3D cube, the cube rotates on itself around the z-axis, and translates to the positive x-axis. If we rotate the object first, the translation of the final destination is still on the x-axis. On the other hand, if we put the same rotation and translation matrix to the cube but translate through the x-axis and rotate around the z-axis, the final position will be different from the first condition. To implement the three dimension mapping, a clear understanding of the three dimensional mathematical structure is required.



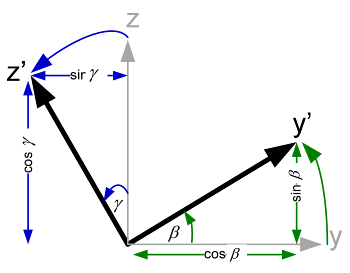


(a) Rotate then translate (b) Translate then rotate

Figure 4.1 Principle of 3D transformation

4.1 Rotation representation using Euler angles

The Cartesian coordinates x, y, z and $\theta_x, \theta_y, \theta_z$ are essential terms to describe the object pose (position and orientation $\mathbb{R}^3$ respectively). Figure 4.2-Figure 4.4 describe the orientation on three axis: roll, pitch and yaw [53].

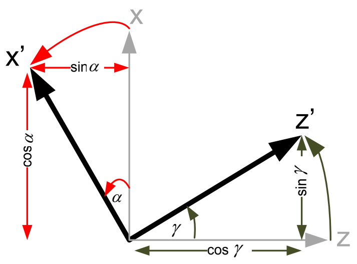


$$R_x = \begin{bmatrix} \cos(\theta_{x',x}) & \cos(\theta_{y',x}) & \cos(\theta_{z',x}) \\ \cos(\theta_{x',y}) & \cos(\theta_{y',y}) & \cos(\theta_{z',y}) \\ \cos(\theta_{x',z}) & \cos(\theta_{y',z}) & \cos(\theta_{z',z}) \end{bmatrix}$$

$$R_x = \begin{bmatrix} \cos(0°) & \cos(90°) & \cos(90°) \\ \cos(90°) & \cos(\beta) & \cos(\frac{\pi}{2} + \gamma) \\ \cos(90°) & \cos(\frac{\pi}{2} - \beta) & \cos(\gamma) \end{bmatrix}$$

$$R_x = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\beta) & -\sin(\gamma) \\ 0 & \sin(\beta) & \cos(\gamma) \end{bmatrix}$$

Figure 4.2 Roll rotation (X-axis)

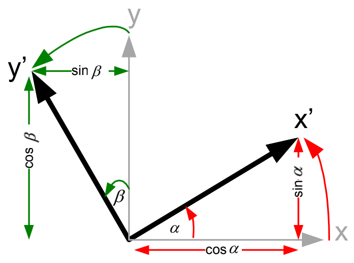


$$R_y = \begin{bmatrix} \cos(\theta_{x',x}) & \cos(\theta_{y',x}) & \cos(\theta_{z',x}) \\ \cos(\theta_{x',y}) & \cos(\theta_{y',y}) & \cos(\theta_{z',y}) \\ \cos(\theta_{x',z}) & \cos(\theta_{y',z}) & \cos(\theta_{z',z}) \end{bmatrix}$$

$$R_y = \begin{bmatrix} \cos(\alpha) & \cos(90°) & \cos(\frac{\pi}{2} - \gamma) \\ \cos(90°) & \cos(0°) & \cos(90°) \\ \cos(\frac{\pi}{2} + \alpha) & \cos(90°) & \cos(\gamma) \end{bmatrix}$$

$$R_y = \begin{bmatrix} \cos(\alpha) & 0 & \sin(\gamma) \\ 0 & 1 & 0 \\ -\sin(\alpha) & 0 & \cos(\gamma) \end{bmatrix}$$

Figure 4.3 Pitch rotation (Y-axis)



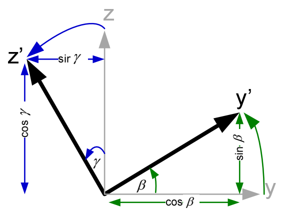
$$R_z = \begin{bmatrix} \cos(\theta_{x',x}) & \cos(\theta_{y',x}) & \cos(\theta_{z',x}) \\ \cos(\theta_{x',y}) & \cos(\theta_{y',y}) & \cos(\theta_{z',y}) \\ \cos(\theta_{x',z}) & \cos(\theta_{y',z}) & \cos(\theta_{z',z}) \end{bmatrix}$$

$$R_z = \begin{bmatrix} \cos(\alpha) & \cos(\frac{\pi}{2} + \beta) & \cos(90°) \\ \cos(\frac{\pi}{2} - \alpha) & \cos(\beta) & \cos(90°) \\ \cos(90°) & \cos(90°) & \cos(0°) \end{bmatrix}$$

$$R_z = \begin{bmatrix} \cos(\alpha) & -\sin(\beta) & 0 \\ \sin(\alpha) & \cos(\beta) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

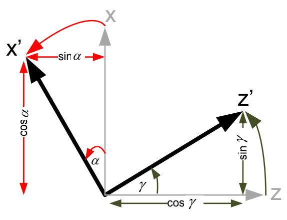
Figure 4.4 Yaw rotation (Z-axis)

roll rotation



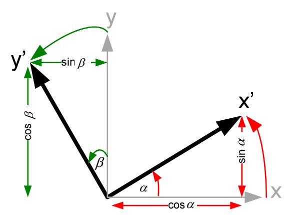
$$R_x = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\beta) & -\sin(\alpha) \\ 0 & \sin(\beta) & \cos(\alpha) \end{bmatrix}$$

pitch rotation



$$R_y = \begin{bmatrix} \cos(\alpha) & 0 & \sin(\gamma) \\ 0 & 1 & 0 \\ -\sin(\alpha) & 0 & \cos(\gamma) \end{bmatrix}$$

yaw rotation



$$R_z = \begin{bmatrix} \cos(\alpha) & -\sin(\beta) & 0 \\ \sin(\alpha) & \cos(\beta) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Figure 4.5 Fundamental rotation matrix

The final formulas of orientation angles are concluded in Figure 4.5. Figure 4.5 shows the matrices of relation from each roll, pitch and yaw. However, for the real time applications, the objects rotate every angle at the same time. The overall orientation is computed by the cross product of each of the elements. The final output equation is in terms of 3×3 matrix, R_{3D}.

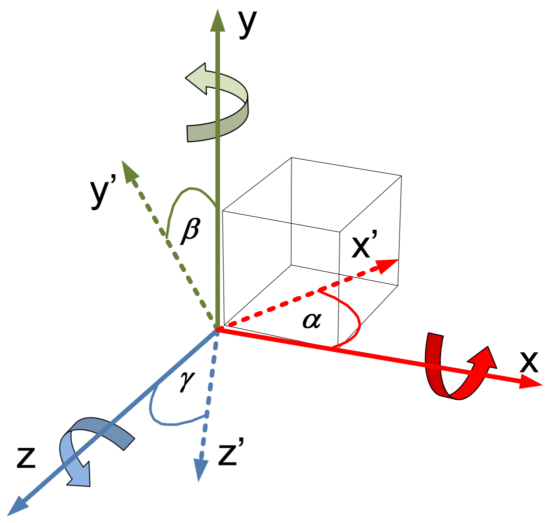


Figure 4.6 Three Dimensional Rotation

$$R_{3D} = R(\alpha).R(\beta).R(\gamma)$$
$$= \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\beta) & -\sin(\alpha) \\ 0 & \sin(\beta) & \cos(\alpha) \end{bmatrix} . \begin{bmatrix} \cos(\alpha) & 0 & \sin(\gamma) \\ 0 & 1 & 0 \\ -\sin(\alpha) & 0 & \cos(\gamma) \end{bmatrix} . \begin{bmatrix} \cos(\alpha) & -\sin(\beta) & 0 \\ \sin(\alpha) & \cos(\beta) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$
$$= \begin{bmatrix} \cos(\beta)\cos(\alpha) & \sin(\gamma)\sin(\beta)\cos(\alpha) + \cos(\gamma)\sin(\alpha) & -\cos(\gamma)\sin(\beta)\cos(\alpha) + \sin(\gamma)\sin(\alpha) \\ -\cos(\beta)\sin(\alpha) & -\sin(\gamma)\sin(\beta)\sin(\alpha) + \cos(\gamma)\cos(\alpha) & \cos(\gamma)\sin(\beta)\sin(\alpha) + \sin(\gamma)\cos(\alpha) \\ \sin(\beta) & -\sin(\gamma)\cos(\beta) & \cos(\gamma)\cos(\beta) \end{bmatrix}$$

4.2 OpenGL (Open Graphics Library) introduction

The 3D scenario output is the main output of our work, the well known programming interfaces are OpenGL and DirectX. OpenGL is a software interface to graphics hardware. The OpenGL is one of the Application Programming Interfaces

(APIs), freeware developed by Silicon Graphics, Inc (SGI). It interfaces to graphic hardware that is 3D modeling library which is highly portable and very fast, elegant and can create beautiful 3D graphics. OpenGL provides the prepackaged functionality. It can be used for a variety of purposes e.g. for CAD engineering, architectural modeling, computer three dimension demonstrations and video games. The OpenGL provides the 3D modeling library that is highly portable and very fast. It is based on the Visual C runtime library which provides some package functionality. It can easily be used with a program based on Visual C++. All the command library can insert in the main C program by including only header files #include<gl/gl.h> and #include<gl/glu.h>. Setting up requires a few steps for complier tools to link to the correct OpenGL library Windows, Macintosh and Linux. In addition, there is no fee for using OpenGL for non profit research. It can create elegant and beautiful 3D graphics in high quality. The OpenGL is intended to be used with computer hardware that is designed and optimized for the display and manipulation of 3D graphics. An OpenGL-based application is used for showing the 3D scenario of 3D mapping. OpenGL is a powerful three dimensional graphic presentation programming. Moreover, it also has the useful library assistant to easily render and smooth the 3D shape. All output results in this research use OpenGL is use for display all output results in this research.

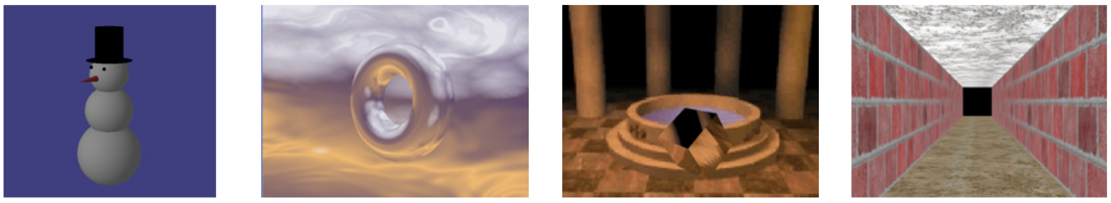


Figure 4.7 OpenGL output examples

It is possible to use any programming language (VB,VC++, etc) can be used in combination with the OpenGL libraries. In this research, we use VC++ for access to API. It also has the useful library assistant to render, draw and smooth complex 3D image. For the mapping in this research, OpenGL is used to be a display. The raw data from PMD and 2D cameras are corrected and analyzed in order to make a 3D mapping. Figure 4.7 shows the examples of OpenGL output [54] [55]. From the figure, we can see that OpenGL can generate several high texture 3D images which are attractive. OpenGL is intended for use with computer hardware that is designed and optimized for the display and manipulation of 3D graphics. OpenGL-based application is used for showing the 3D scenario of 3D mapping. Table 4.1 expresses the preprocessing of how to include the herder file in VC++ programming. This example expresses the empty frame (800 × 600) with blue background color.

```
#include <gl/gl.h>
#include <gl/glu.h>
#include "glut.h"

///////////////////////////////////////////////////////////
// Setup the rendering state
void SetupRC(void)
    {
    // Set clear color to blue
    glClearColor(0.0f, 0.0f, 1.0f, 1.0f);
    }

//////////////////////////////////////////////////////////////
// Called by GLUT library when the window has chanaged size //
//////////////////////////////////////////////////////////////
void ChangeSize(int w, int h)
        {
        GLfloat aspectRatio;
        // Prevent a divide by zero
        if(h == 0)
           h = 1;
        // Set Viewport to window dimensions
      glViewport(0, 0, w, h);
        // Reset coordinate system
        glMatrixMode(GL_PROJECTION);
        glLoadIdentity();
        // Establish clipping volume (left, right, bottom, top, near, far)
        aspectRatio = (GLfloat)w / (GLfloat)h;
      if (w <= h)
        glOrtho (-100.0, 100.0, -100 / aspectRatio, 100.0 / aspectRatio, 1.0, -1.0);
      else
        glOrtho (-100.0 * aspectRatio, 100.0 * aspectRatio, -100.0, 100.0, 1.0, -1.0);
        glMatrixMode(GL_MODELVIEW);
        glLoadIdentity();
        }

int main(int argc, char* argv[])
    {
      glutInit(&argc, argv);
      glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
      glutInitWindowSize(800, 600);
      glutCreateWindow("Simple Example");
      glutReshapeFunc(ChangeSize);
       SetupRC();
       glutMainLoop();
      return 0;
    }
```

Table 4.1 The simple example for starting the OpenGL frame

In Figure 4.8 the approach of how to show the 3D data in OpenGL is demonstrated. Figure 4.8 is separated into two main parts, i.e. PMD depth data and 2D image capturing. The 2D camera is used with OpenCv library to capture the image frames. Meanwhile, PMD camera use “pmdaccess.dll”, which is provided from PMD software, CamVisPro. Additionally, the 2D camera is used for demanding and controlling the behavior of image registration. Thus, many algorithms are applied before yielding the final output. The PMD camera is used for providing the exact depth data to generate the high performance SLAM in real-time. Both data are fused, and showed the output on the monitor screen based on OpenGL library. Figure 4.9 shows a shot at capturing the output from PMD camera which was taken of the corridor. The output uses OpenGL in a mode cube plot to show the depth data.

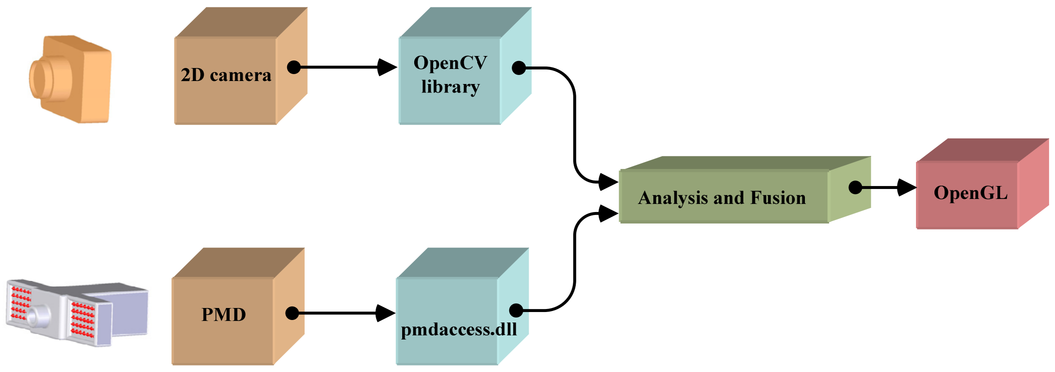


Figure 4.8 Using OpenGL to express 3D data

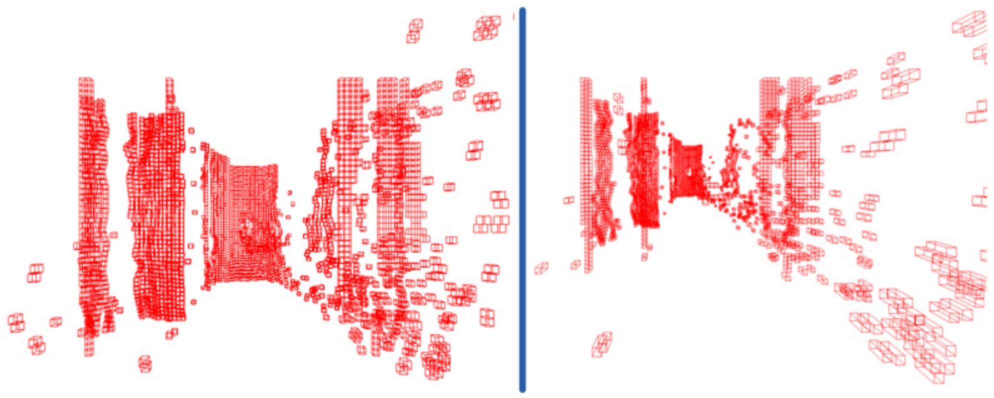


Figure 4.9 One shot PMD capturing by using OpenGL plotting

4.3 The PMD camera principle

4.3.1 The principle of PMD

The PMD camera stands for **P**hotonic **M**ixer **D**evices. It is the product of PMDTechnologies Company [56] which is located at the same area as the University of Siegen, Germany. The first PMD camera prototype was a research product from the Center for Sensor Systems (Zentrum für Sensorsysteme, ZESS) at the University of Siegen. The PMD has been researched over decade of scientific research on 3D time-of-flight imaging. The PMD has been subject of decades of scientific research on 3D time of flight imaging. The scientific research was detached from the mass product, namely PMDTec. ZESS and PMDTec, which support each other. The PhD research work attempts to improve the scientific work. PMDTec supports not only the hardware, but also deals with issues associated with reality from industrial relations. The PMD camera as a smart pixel can be an integration element for a 3D imaging camera on a chip based on standard CCD- or CMOS-technology and time-of-flight principle [57] [58] [59] [60]. Thus, the complete 3D information is captured in parallel without needing excessive calculation power. The smart-pixel sensor is able to capture a complete 3D scene in real time without any moving part. PMD based 3D imaging can be deployed in a wide range of indoor and outdoor applications. The Photonic Mixer Devices entitled PMD sensor as a smart pixel can be an integration element for a 3D imaging camera on a chip based on standard CCD- or CMOS-technology. The main feature is an array sensor, which can measure the distance to the target in parallel without scanning. The key execution is based on time of flight principle. A light pulse is transmitted from a sender unit and the target distance is measured by determining the return time back to the receiver. According to the speed of light, the interval distance can easily be calculated.

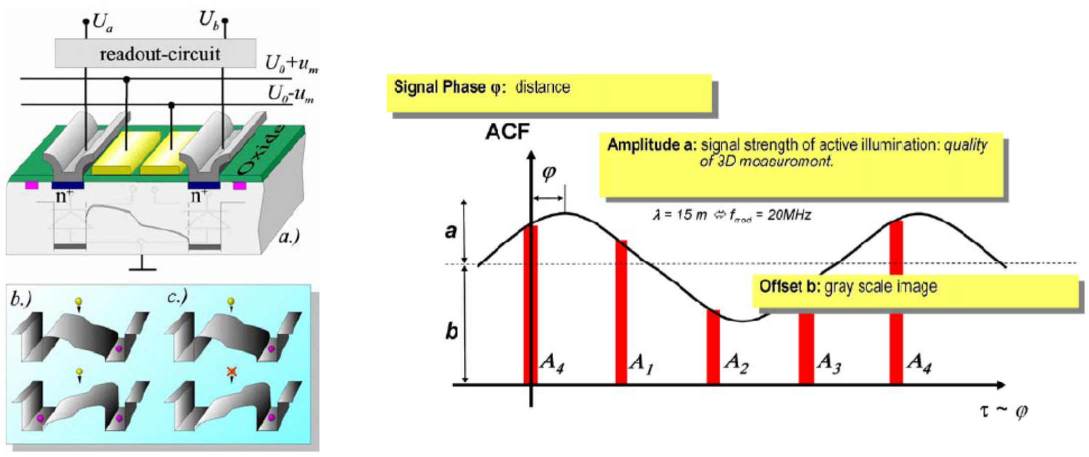


Figure 4.10 Principle of PMD sensor [60]

In Figure 4.10, the principle of time of flight principle based on PMD camera is illustrated. It is a single pixel PMD sensor element. There are two conductive diodes inside the pixel construction. The light sensitive Photo gates influences diodes to conduct the bias voltage. The voltage output generates the readout-circuit. If the incident light is constant and the modulation is the rectangular signal with a duty cycle of 50%, the phase shift to diode bias gate is the main influence to generate the differential voltage at the readout-circuit.

4.3.2 Distance analysis

To calculate the distance between obstacles and camera, the relative function of electrical and optical signal uses a phase-shift algorithm. The main component is an array sensor. It can measure the distance to the target in parallel without scanning. A light pulse is transmitted from a sender unit and the target distance is measured by determining the return time back to the receiver. According to the speed of light, the interval distance can easily be calculated. [60] shows the models and the principle time of flight principle based on PMD camera. Figure 4.11 expresses an easy understanding to calculate the distance. A source light emits a pulse and at the same time also starts the highly accurate stopwatch. If we can calculate a period based on the stopwatch, the distance can be calculated based on the following equation:

$$d = \frac{c.t}{2} \tag{4.1}$$

The equation (4.1) is the distance analysis method based on the time of light principle. d is the measured distance, c is the speed of light and t is duration time after emitting until receiving back. Dividing by 2 because time (t) goes both ways (out-backward). Therefore, the precise measurement depends on the accuracy of the sensor. Since the time of flight of light is 299,792,456 meter per second. The accuracy timer counter is required. The measured time is applied to use the phase shift determination method. In order to send the modulated signal to the target, the phase is shifted when it returns back to the receiver. The modulated light is measured for the intensity and phase shift, equation (4.2). The cross correlation method can determine the relation between the transmitted and received signal. Where a reference signal is $g(t)$ and optical signal is $s(t)$. $c(\tau)$ is the correlation function and the internal phase delay is τ [58].

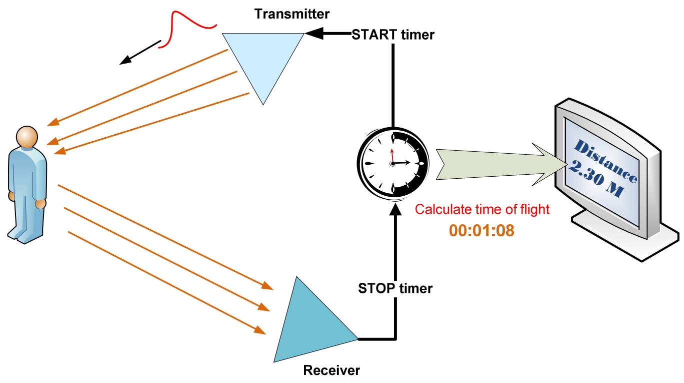


Figure 4.11 Principle distance calculation

$$c(\tau) = s(t) \otimes g(t) = \lim_{T \to \infty} \frac{1}{T} \int_{-\frac{T}{2}}^{\frac{T}{2}} s(t) g(t+\tau) dt \tag{4.2}$$

The PMD sensor uses the reference sinusoidal signal $g(t) = \cos(\omega t)$ and the optical received signal is $s(t) = k + a\cos(\omega t + \varphi)$. Where ω is the modulation frequency, φ is the phase shift of feedback signal, a is modulation amplitude. The correlation function $c(\tau)$ can be calculated

$$\begin{aligned} c(\tau) &= [1 + \mathrm{a}\cos(\omega t - \varphi)] \otimes [\cos(\omega t)] \\ &= \lim_{T\to\infty} \frac{1}{T} \int_{-\frac{T}{2}}^{\frac{T}{2}} [1 + a\cos\,(\omega t - \varphi)][\cos\,(\omega t + \omega\tau)]dt \\ &= \frac{\mathrm{a}}{2}\cos\,(\varphi + \omega\tau) \end{aligned} \tag{4.3}$$

The sending signal is the difference in phase (ωt), selecting $(\omega t_0) = 0^{\circ}$, $(\omega t_1) = 90^{\circ}$, $(\omega t_2) = 180^{\circ}$, $(\omega t_3) = 270^{\circ}$. By the correlation function at four selected points, we can determine the phase φ and amplitude a of the received optical signal $s(t)$ as

$$\begin{aligned} \varphi &= \arctan\left(\frac{c_{(\tau 3)} - c_{(\tau 1)}}{c_{(\tau 0)} - c_{(\tau 2)}}\right) \\ a &= \frac{\sqrt{\left[c_{(\tau 3)} - c_{(\tau 1)}\right]^2 + \left[c_{(\tau 0)} - c_{(\tau 2)}\right]^2}}{2} \\ h &= \frac{c_{\tau 0} + c_{\tau 1} + c_{\tau 2} + c_{\tau 3}}{4} \end{aligned} \tag{4.4}$$

The amplitude a is the measurement for the quality of the distance. h is offset of sinusoidal signal obtain by four sampling points $c_{\tau 0}, \dots, c_{\tau 3}$. The distance between the target and camera (d) can easily be calculated, see equation (4.5). Where c_0 is the speed of light, φ is the phase shift and f_{mod} is the modulation frequency.

$$d = \frac{c_0 \varphi}{4\pi . f_{mod}} \tag{4.5}$$

Many applications use the PMD camera such as machine vision. Range measurements are obtained exclusively via time-of-flight (TOF) information. The PMD camera provides three data, i.e. the distance, amplitude and 8 bits gray scale value. The PMD chip is the prominent component, which provides depth information in each pixel of the corresponding point in the object plane. Additionally, the PMD camera has the advantage of fast image mapping. It has been used in the automotive sector, security and surveillance, medical technology and life sciences. This camera enables fast optical sensing and demodulation of incoherent light signals in one component. It also gives both intensity and distance for each pixel. The PMD can be used to get the excellent depth information as well as gray scale value of the scene. A common modulation frequency is 20 MHz, which results in an unequivocal distance range of 7.5 to 40 meters. The depth data is obtained from the phase shift of the out-coming and incoming signals. The cross correlation is a measurement method of the similarity of two waveforms as a function of the shifting time. It applies in pattern recognition and signal electron analysis.

4.3.2.1 Review the related PMD camera researches

The PMD camera has been researched in several applications. This section reviews the related works of PMD cameras, which are separated into the following topic:

Combination and noise reduction: Prasad et al. [5] presented the first step of the combination of the high resolution with two dimension camera and PMD camera. They showed the idea to setup the mechanical platform. Their outputs could enhance the 3D vision and express output in real time. Wiedemann et al. [61] analyzed and proposed the characteristics of the PMD camera in mobile robot applications. They proposed that the integration time of the PMD camera is affected by the distance of the objects. By adjusting the integration time, an increasing reliability output can be achieved. They also used their work for autonomous movements, map building and obstacle avoidance. Beder et al. [62] compared the performance between the PMD and stereo camera using an oriented planar 3D patch. Their outputs showed that the PMD could provide the better accuracy but the low resolution is the main drawback of PMD. They suggested combining the PMD and stereo in order to yield better and potentially benchmarking vision surface reconstructions. Reulke et al. [63] combined the distance data with the high resolution image. The object coordinates the depth from PMD transformed into the lab coordinate frame. Schiller et al. [6] proposed the use of a planer calibration pattern for calibrating the PMD and standard 2D CCD camera. They found the new calibration methods, because the traditional method could not be used due to the small field-of-view and low pixel resolution of the PMD. The aim for the precise calibration is to find the intrinsic parameters, lens distortions and the systematic range error. Lindner et al. [64] proposed the data fusion between the PMD and RGB image. The final output could express the high depth with texture. The hidden surface removal was expressed by the mechanical setting up. Huhle et al. [65] discussed the denoising algorithms for the depth data and introduced a novel technique based on the NL-Means filter. Their works could remove the outliers from the depth data and accordingly achieve an unbiased smoothing result. Huhle et al. [66] presented a scene acquisition system, which allows for fast and simple acquisition of arbitrarily large 3D environments. They processed the frames consisting of depth and color information at interactive rates. A novel registration method was introduced that combines geometry and color information for enhanced robustness and precision. Santrac et al. [67] combined the 3D time of flight camera with a standard 2D camera to improve the robustness of the segmentation and to lower its computation complexity.

Mobile robotics: Ruangpayoongsak [49] used the PMD camera to detect the artificial landmark for the mobile in order to localize a mobile robot. The 3D output could sort out the object e.g. the bottle or the chair. Vacek et al. [68] performed the collision avoidance for the autonomous vehicles. The PMD and video cameras were assembled on the top of the vehicle, and showed how it could detect the obstacles. The depth data had to be filtered, afterward segmented into regions of equal depth. In the next step, a part of the segmented regions is compared with previous regions stored in the object list. At last, the updated objects are written into the real-time database so that the obstacle information can be achieved by the situational interpretation.

Machine learning: Ringbeck et al. [69] applied 3D data for object detection. Moreover, they presented the clarified 2D/3D output.

4.3.3 The PMD camera models

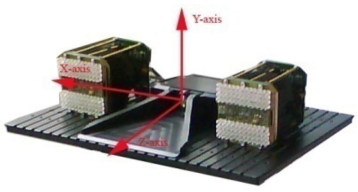
(a) The PMD A2: A2 is very powerful for the long distance detection with the maximum of 40 meters detecting distance. It provides 64×16 pixels with an angle of view of $52^{\circ} \times 18^{\circ}$. The model is approved for the automotive application for the safety in road traffic. It can recognize the relevant and non relevant objects.

(b) PMD [vision]® S3: S3 is a universal 3D camera with high robustness against background illumination. It provides 64×48 pixels. The maximum sampling rate is 20 Hz. It is emphasized in games, automation and automotive applications.

(c) The PMD[vision]® O3 is a universal, compact and lightweight 3D camera. It is robust and has excellent background illumination suppression properties. The resolution is 64×48 pixelswith 25 Hz max sampling rate. The interface with Ethernet is 10Base-T/100Base-TX. The camera can be used with both indoor and outdoor applications. It can also be applied to autonomous guided vehicles and robotics, security and protection as well as to automotives.

(d) The PMD[vision]® CamCube 3.0 is the highest resolution of time-of-flight camera worldwide (since 2010) The optical sensor 200×200 pixels, provides both of the depth and gray scale information with standard measurement of 0.3-7 meters. The field of view is $40^{\circ} \times 40^{\circ}$. USB 2.0 interface. This sensor increases the frame rate and integrated SBI technology. It can be used in indoor and outdoor environments. The embedded motion blur resistance allows for detecting fast moving objects.

(e) The PMD PhotonICs® 3k-S and 19k-s, the configuration is the same as before. But the array size output 3k-S provides 64×48 pixels, 19k-S provides 160×120 , and does not have the suppression of the background illumination (SBI). The SBI is included on the PMD chip, which can reduce the infrared light from the environment. The light source generates wavelength 870 nm with 3 watt energy consumption. The modulated light reaches its maximum at 20MHz with an expected distance of 7.5 meters.



(a) The PMD A2

(b) PMD[vision]® S3

(c) PMD[vision]® O3

(d) PMD[vision]® CamCube 3.0

(e) 3k-s and 19k-s

Figure 4.12 PMD camera models

The data is sent via the Ethernet (IEEE802.3u) and FireWire (IEEE 1394). The voltage range is 9-18 V. The 3k-S imager is in its third generation, and is fully certified in line with the relevant industry requirements. Every imager features SBI and can be used indoors as well as outdoors. Moreover, there is another kind of PMD, which is PhotonICs® 41k-S. It has 204 × 204 pixels and a high modulation bandwidths. The imager opens up a host of possibilities across a variety of applications, such as robotics, manufacturing, consumer electronics, gaming and medical technology. It offers excellent distance measurement capabilities without compromising resolution and boasts the following uniqueness. The PhotonICs® 1k-S chip has been approved for the deployment in automotive industry and is now in its third generation. The sensor has a resolution of 64 × 48 pixels ToF pixels. It reaches the automotive qualification according to ZVEI recommendations for Robustness Validation and can be deployed immediately in product development. Every imager is equipped with SBI, enabling indoor as well as outdoor use.

4.3.4 Starting connection

For the example of getting the raw data from FireWire (IEEE 1394), the header file “pmdmsdk.h” has to be included in the main file. A software development kit (SDK) is used for retrieving the data in variable dat. It contains 3072 pixels for the depth data. To calculate PMD data, we have to arrange the data into matrix by using two for loop.

```
#include "pmdmsdk.h"
        PMDHandle hnd;
        unsigned int res, width, height;
        double * dat;
         #define Width 64
         #define Heigh 48
         float PMDdata[Heigh][Width];
void GetPMD(void)
{
        int x,y;
        res = pmdConnectFireWire (&hnd);            // connect to camera
        if(res !=PMD_OK)
           printf("Could not connect\n");
        else
           printf("Access complete\n");

         pmdSetIntegrationTime (hnd, 3000);         //set the integration time
         pmdSetModulationFrequency (hnd, 30);       //set the modulation frequency
         res = pmdUpdate (hnd);
        if (res !=  PMD_OK)
            exit(3);

         res = pmdGetDistances (hnd, (void**)&dat); //retrieve depth data
        if (res != PMD_OK)
            exit(4);

            for (y=0;y<Heigh;y++)                   //arrange data to matrix 64x48
                 for(x=0;x<Width;x++)
                     PMDdata[x][y]=dat[(64*y)+x];    //PMDdata contains depth [64][48]
}
```

4.3.5 PMD camera characteristics

This section explains the pros and cons of each camera model. Table 4.2 shows the conclusion of the PMD camera's characteristics [61]. It expresses the detail of the resolution output (pixels), the field of view (height × width, degree), the sampling rate (Hz), the suppression of background illumination (SBI) as well as connection methods. Each model has its own pros and cons. It is used in different tasks depending on the required applications. PMD camera has two parameters, which can adjust the modulation frequency (Hz) and integration time (μs). Both parameters can tune by software which is provided from PMD library. The integration time is effected by the distance between the object and camera. It is used to calculate the distance. In particular, the PMD also relates to other parameters like the time of light and the phase shift on which both parameters are fixed. Figure 4.13 depicts the different integration time that is adjusted. When the integration time is shot (100 μs), the overall image is noisy but the camera can provide the faster data, the near objet (book) is acceptable but the far object (chair) is not.

Model	Resolution (pixels)	Field of view (degree)	Sampling rate (Hz)	SBI	Connection
PMD[vision]® 1k-S	64x16	45x16		Yes	Ethernet / FireWire
PMD[vision]® 3k-S	64x48	40x30	120	Yes	Ethernet / FireWire
PMD[vision]® 19k-S	160x120	40x30		No	Ethernet / FireWire
PMD[vision]® S3	64x48	40x30	20		Ethernet
PMD[vision]® O3	64x48	40x30	25		Ethernet
PMD[vision]® A2	64x16	52x18	100	Yes	Ethernet
CamCube 2.0	204x204	40x40			USB 2.0

Table 4.2 PMD camera characteristics

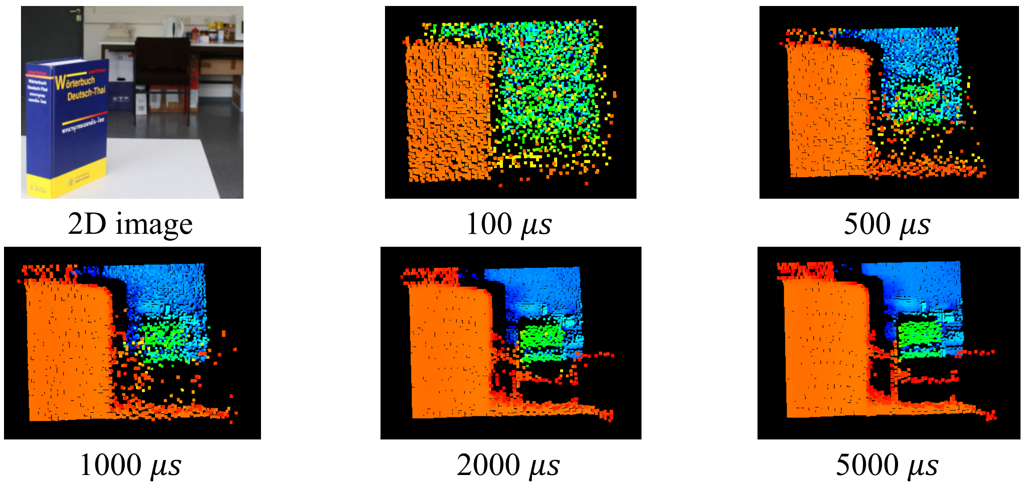
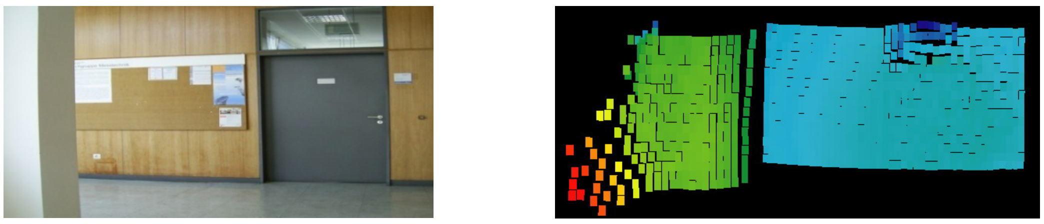


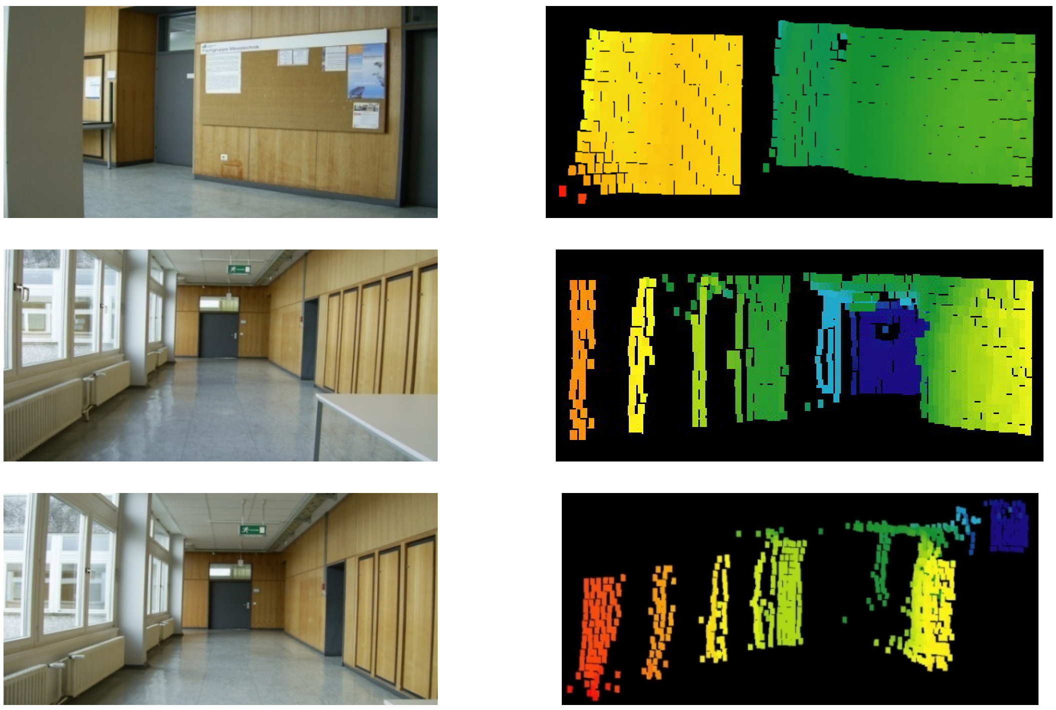
Figure 4.13 Changing integration time effect to the noise

When increasing the integration time, the output quality is increased step by step (noises reduce) but the giving data speed from the camera is slower. The high integration time ($5000\,\mu s$) has less noise than the low integration time. If we consider the high quality output the high integration time is suitable but in mobile robot applications the speed of the data capturing cannot be ignored. Wiedemann et al. [61] proposed the optimum integration time that related to the distance objects. They suggested that the integration time should adjust according to the distance. Especially the integration time has to be adapted for near the near object($< 1m$). They proposed a simple adaptation in three steps. Firstly, the average amplitude of each frame(Ave_{amp}) is calculated. Secondly, the deviation (dev) of the average amplitude and (Amp) each amplitude depth pixel ($dev = Ave_{,amp} - Amp$) are calculated. Thirdly, the new integration time ($Int_{new} = Int_{old} + dev.Int_{old}$) is adapted.

4.3.6 PMD camera acquisition

The reason to use the PMD camera for mobile robot application is that it provides the fast 3D volume compared to the frame size of other sensors. It does not require the additional moving mechanism. However, many drawbacks of PMD still have to be improved such as the field of view and maximum detected range. Figure 4.14 shows the one scan data that takes in the corridor using the A2 PMD camera.





(a) Image from 2D camera (b) Depth data from PMD camera

Figure 4.14 PMD scan raw data in the corridor

4.4 Median filtering

The raw data from the PMD camera probably contains the unpredictable noise because of the varied environment scenes and adaptations of the integration time. The main occurring of disturbing noise are the reflecting surfaces, e.g. the transparency glass and pure white color surface. The PMD camera is based on the light source that sometimes has an effect on the glass surfaces, strongly absorbing and reflecting surfaces. In this section, the simple algorithm to reduce the random noise and salt pepper is discussed. Noise reducing and filter algorithm are necessary to improve the raw data. The main idea to reduce the noise is to find the threshold data points, which are close (neighbor data) and join them together into one data. The filter removes the outliers by replacing the old data point with the new value of their surrounding points. The noises, which may occur, are defined as Gaussian noise, salt and pepper noise. The fundamental consideration concerning noise reduction is separated into two steps. Firstly, the raw data is filtered by checking the intensity of each point. The intensity of each pixel expresses the quality of data. Secondly, the interpolation of the neighbor data points to their surrounding points and join each point. Beyond the threshold length data has to be eliminated. The simple method uses the convolution technique implementing weighting kernels, but it is suitable for linear neighborhood operations. A median filter is one of the interesting topics to remove salt and pepper noise as well as outliner data. The median filter performs the neighboring pixels that are ranked according to intensity. Each output pixel contains the median value in the neighborhood around the corresponding pixel. The problem of unpredictable noise sometimes occurs.

The mean filtering [70] is the simplest type of the low-pass filter to replace each pixel value in an image with the mean value of its neighbors, eliminating the unrepresentative pixels, and attenuating the image noise. It uses the convolution filter based on a kernel, which often uses 3×3 square kernel. The characteristic of low-pass filter is that it can remove the high frequency information. It affects edges, lines and points that are smoothed.

$$v(m,n) = \sum_{(k,l)\in W} \sum a(k,l) y(m-k, n-l) \tag{4.6}$$

Where $y(m,n)$ and $v(m,n)$ are the input and output of the images, respectively. $W$is a suitably chosen window, $a(k,l)$ are the filter weights. A common class of spatial averaging filters that has all equal weights is

$$v(m,n) = \frac{1}{N_W} \sum_{(k,l)\in W} \sum y(m-k, n-l) + \bar{\eta}(m,n) \tag{4.7}$$

Where $a(k,l) = \frac{1}{N_W}$, N_W is the number of pixels in the window W and $\bar{\eta}(m,n)$ is the white noise average with zero mean and variance $\frac{\sigma_\eta^2}{N_W}$. The noise power is reduced by a factor equal to the number of pixels in the window. Figure 4.15 (a) shows the common mean filter mask coefficients. Median filter [71] is a nonlinear process that is useful for impulsive reduction, random noise, salt and pepper noise. Impulsive or salt-pepper noise can occur due to a random bit error. Median filter slides a window along the image. The media intensity value of pixels within the window becomes the output intensity of the pixel being processed. The Median Filter does somewhat the same, instead of taking the mean or average, it takes the median. The median is gotten by sorting all the values from low to high and then taking the value in the center. If there are two values in the center, the average of these two is taken. A median filter gives better results to remove salt and pepper noise because it completely eliminates noise. With an average filter, the color value of the noise particles are still used in the average calculations. When taking the median we only keep the color value of one or two healthy pixels. The median filter also reduces the image quality however.

$$v(m,n) = median\{y(m-k, n-l), (k,l) \in W\} \tag{4.8}$$

Where W is a suitably chosen window. The algorithm for median filtering requires arranging the pixel values in the window in increasing or decreasing order and picking the middle value. The window size is chosen that N_W is odd. If N_W is even, the median is taken as the average of the two values in the middle. Figure 4.15 (b) shows the simply median value. If we arrange the data in $v(m,n)$ window and the ranking data as $\underset{1^{st}}{\overset{5}{\uparrow}}, \underset{2^{nd}}{\overset{10}{\uparrow}}, \underset{3^{rd}}{\overset{10}{\uparrow}}, \underset{4^{th}}{\overset{12}{\uparrow}}, \underset{5^{th}}{\overset{15}{\uparrow}}, \underset{6^{th}}{\overset{21}{\uparrow}}, \underset{7^{th}}{\overset{21}{\uparrow}}, \underset{8^{th}}{\overset{25}{\uparrow}}, \underset{9^{th}}{\overset{30}{\uparrow}}$ the median of this window is 15 (5^{th}) because the 5^{th} is the median of all 9 value.

$\frac{1}{9}$	$\frac{1}{9}$	$\frac{1}{9}$
$\frac{1}{9}$	$\frac{1}{9}$	$\frac{1}{9}$
$\frac{1}{9}$	$\frac{1}{9}$	$\frac{1}{9}$

Mask 1

$\frac{1}{10}$	$\frac{1}{10}$	$\frac{1}{10}$
$\frac{1}{10}$	$\frac{1}{5}$	$\frac{1}{10}$
$\frac{1}{10}$	$\frac{1}{10}$	$\frac{1}{10}$

Mask 2

$\frac{1}{16}$	$\frac{1}{8}$	$\frac{1}{16}$
$\frac{1}{8}$	$\frac{1}{4}$	$\frac{1}{8}$
$\frac{1}{16}$	$\frac{1}{8}$	$\frac{1}{16}$

Mask 3

(a) Mean filter

10	5	21
15	12	30
25	10	21

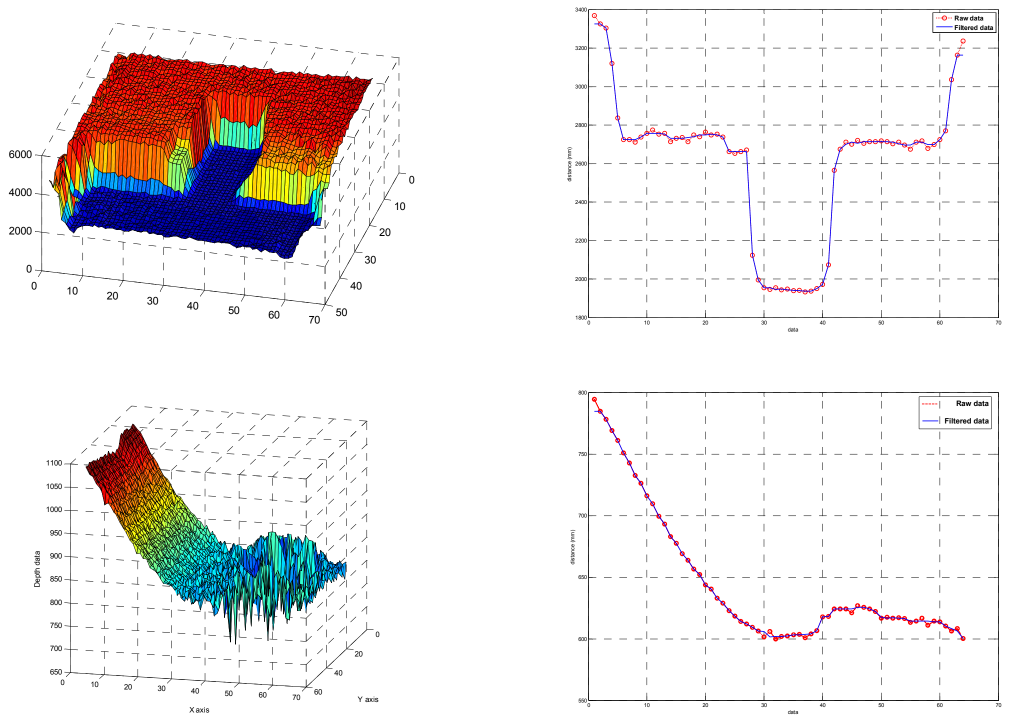
Median = 15

(b) Median filter

Figure 4.15 Filter mask coefficients

4.4.1 Data filtering results

This section shows the output after using the median filter. Figure 4.16(a) shows the one shot of PMD camera, which does not have many noises that disturb the system. Figure 4.16(b) compares the raw data (circle red dash) and filtered data (blue line). The blue line is the average data in every column. The results show that if the noise is under the threshold, the filtered data can still track the raw data. Figure 4.17 and Figure 4.18 show the salt-pepper noise which occur in the raw data. Top left is the raw data, top right is the filtered data. The bottom right is the error data that is eliminated and the bottom right compares the raw data (circle red dash) and filtered data (blue line). The figure shows the salt-pepper noise that could be eliminated while the filtered data will be used to build the map.



(a) (b)

Figure 4.16 (a) Raw PMD data (b) Outliner removal using media filter

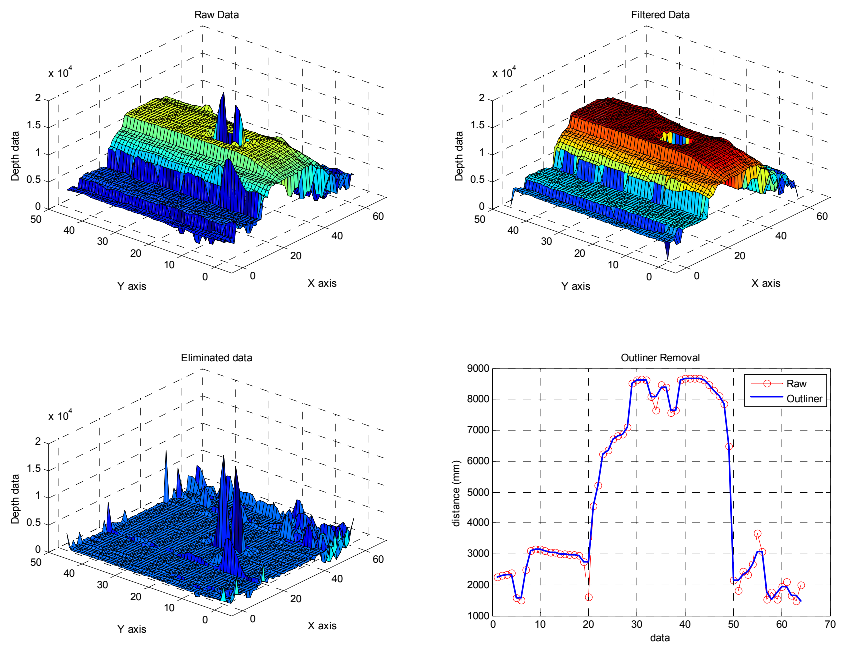


Figure 4.17 Median filter, remove the outliner

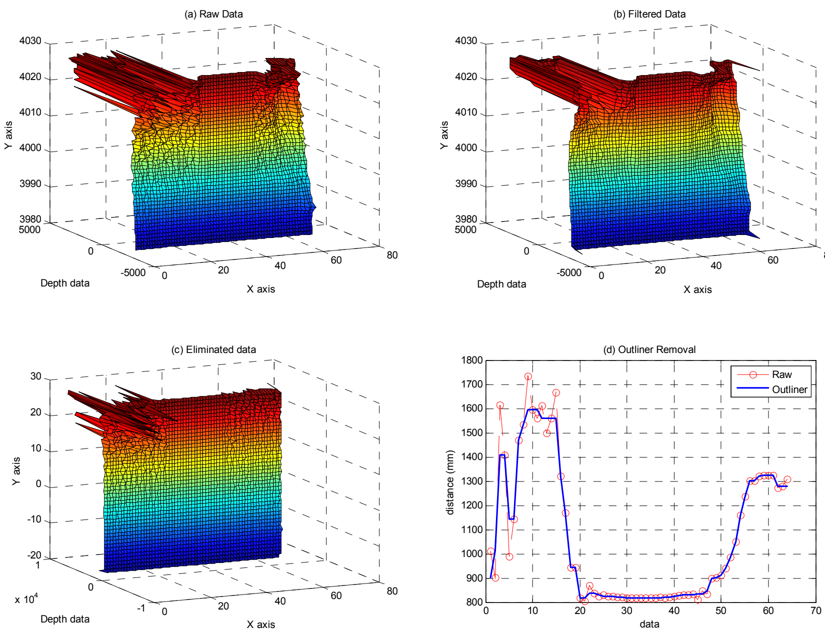


Figure 4.18 Median filter, remove the outliner

4.5 Camera calibration

The PMD camera provides precise depth data, however for the mobile robot application the multi tasking is necessary. The depth PMD data can be applied very well for moving object application but for non-moving objects is difficult. Figure 4.19 explains the moving and non moving object. The figure shows two dimension data on top left hand side and depth data from PMD camera on below right hand side. As figure, human being moves in the room (red square covering). The depth information changes continuously, thus to detect the moving object is not difficult to deal. The segmentation, background subtraction and classification the particles in the image plane can be done [72]. On the other hand, jacket is non-moving object (yellow square covering). Use only depth data cannot detect the jacket because the depth of jacket and the wall are quite same. Classification the jacket (non-moving object) is too complicated. The most efficiency and easiest to classify the non-moving object is to use the information from high resolution camera. Object detection is an important element of various computer vision areas. The goal is to find an object of a pre-defined class in a static video frame. Sometimes this task can be accomplished by extracting certain image features, such as edges, color regions, textures contours etc. Then use heuristics to find configurations and combinations of those features specific to the object of interest. However the complex objects, such as human faces are hard to find features and heuristics that will handle the huge variety of instances of the object class. Inspire by this problems the Haar-like features is use for the complex object detection. Before handle this algorithm, the calibration two cameras is processed in order to acquire the trustable data.

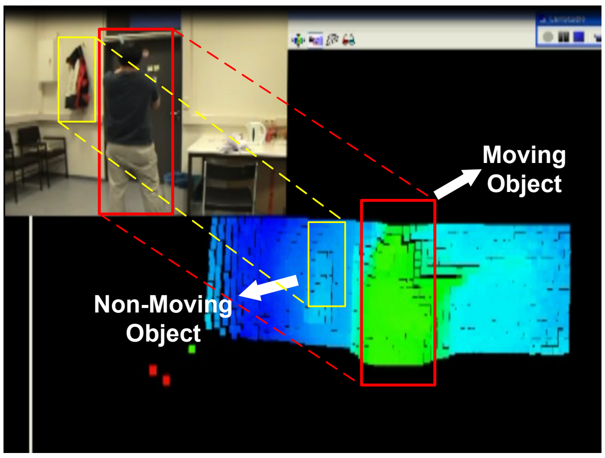


Figure 4.19 Object detection motivation

The Camera calibration is an essential step before handling image processing tasks in order to extract the metric information from image frames, especially the calculation between two cameras. Camera calibration is important for the measurement in real three dimensions and relation between camera and world coordinate. This research acquires the image data from two cameras, i.e. PMD and 2D camera. The camera calibration is an essential step in 3D computer vision in order to extract the metric information from 2D image. Ordinarily, calibration technique can be classified into two categories, photogrammetric and self calibration. *Photogrammetric calibration* uses an observing calibration object in 3D geometry space, which provides the good precision output. The calibration objects consist of couple related orthogonal planes. This method requires an expensive calibration equipment and elaborate setup. *Self calibration* does not use any calibration object. A moving camera in a static scene is required. Coordinate of scene provides the internal parameters from one camera displacement by using image information alone. If the images are taken by the same camera with fixed internal parameters, correspondences between three images are approved to find out the internal and external parameters in order to construct the 3D geometry. This approach is very flexible. However, there are many parameters that have to be estimated, and reliable results cannot be obtained. Lindner et al. [64] presented a precise calibration approach using the transformation methodology of 2D camera → World coordinate → PMD camera. This would increase the reliability of comprehensive results and we probably will attach in the future steps. A CCD frame is transformed onto the PMD frame through a rotation and translation matrices. Zhang [73] proposed a flexible technique to calibrate cameras using cameras to observe a planar pattern shown at a few different orientations. This approach combines the photogrammetric and self calibration techniques. Only 2D metric information was used instead of 3D one. The process of camera calibration gives us both a model of the camera's geometry and a *distortion* model of the lens. These two informational models define the *intrinsic parameters* of the camera. The camera calibration is an essential preliminary step before solving the vision tasks. The 2D camera and PMD camera are set up in the proper angle. Figure 4.20 shows the assumption that both of cameras point out to the same object. However, the extrinsic parameters such as focal length, image center, etc. are unlike. Thus, the intrinsic matrix between both cameras has to be found out by calibration. Fundamentally, PMD camera provides two data, depth and intensity. The intensity of PMD camera output is used for calibration only. Prasad et al. [5] presented the fundamental enhancement in the 3D vision output. They placed the 2D and 3D camera in a special structure and assumed that the fields of view of both cameras are almost the same. The real 3D range information with high intensity was generated. Lindner et al. [64] described a fast combining approach for the high resolution RGB image with PMD distance data. The RGB data is projected onto the geometry reconstructed depth information from the PMD camera. Mischler [74] reconstructed the 3D model of stereo and PMD data by using an OpenCV library.

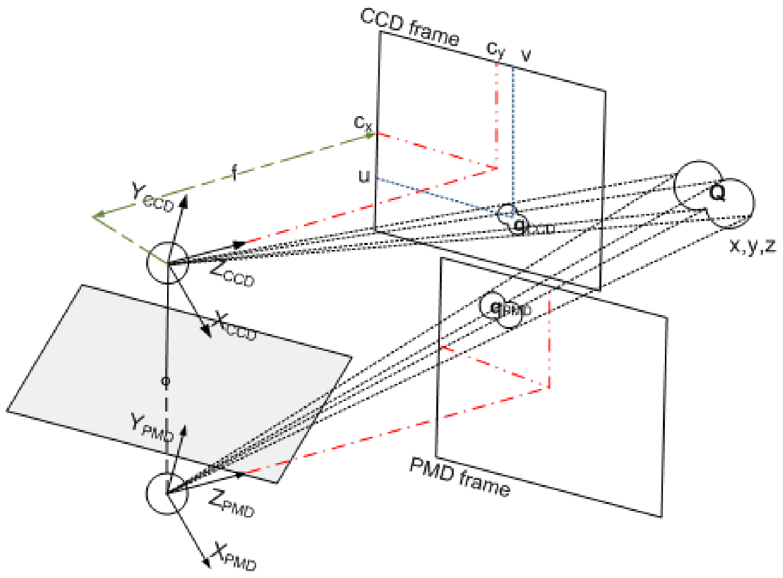


Figure 4.20 Camera calibration model

This thesis followed the hardware setting up principles and projective texture on geometry. Nevertheless, we are still not considering hidden surface removal and distortion factor. As shown in Figure 4.20 denotes that (q_x, q_y, q_z) are the coordinates of point $\underline{q}$ in the world space. It was assumed that the object is pointed out by the CCD camera, $\underline{q}_{CCD} = (q_{CCD\_a}, q_{CCD\_b})^T$ is the projected coordinate onto an artificial camera frame, if the original of the image coordinate does not align on $\underline{z}$ image plan axis. Denote (a_0, b_0) is the center of the artificial frame (principal point) and f is the focal length of the camera [75].

$$\begin{gathered} \frac{f}{Z} = \frac{u}{X} = \frac{v}{Y} \\ u = c_x + \frac{fX}{Z} \\ v = c_y + \frac{fY}{Z} \end{gathered} \tag{4.9}$$

By using the perspective transformation, we can define that using homogeneous coordinates, can be performed Q in a matrix.

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix} \begin{bmatrix} q_x \\ q_y \\ q_z \\ 1 \end{bmatrix} \tag{4.10}$$

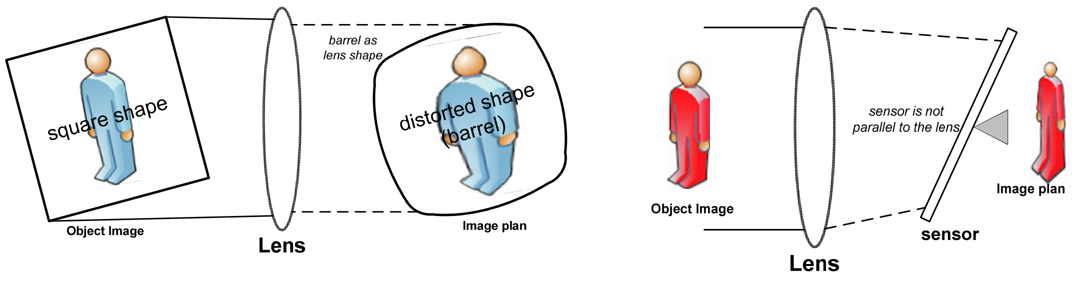
Where f_x and f_y are the focus lengths in x and y axis respectively. Finally, a matrix can be arranged as

$$\begin{gathered} \underline{\mathrm{U}} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \end{bmatrix} \\ \underline{\mathrm{E}} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix} \end{gathered} \tag{4.11}$$

Where $\underline{\mathrm{U}}$ is an intrinsic matrix of a camera. $\underline{\mathrm{R}}$ is an extrinsic matrix of a camera. The projection from point $\underline{\mathrm{Q}}$ onto the camera screen $\underline{\mathrm{m}}$ is

$$\underline{\mathrm{m}} = \underline{\mathrm{UEq}} \tag{4.12}$$

In practice, every lens has its distortions. The major components of the lens distortion are the radial distortion and slight tangential distortion. The radial distortion is a result of the lens shape. Because the lens shape is barrel, the distortion can be characterized by the terms of a Taylor series expansion [76]. The conventionally called k_1, k_2 and k_3. The image point location is distorted following equation.



(a) (b)

Figure 4.21 (a) Radial distortion (b) Tangential distortion

$$\begin{aligned} x_{u\_rad} &= x_d(1 + k_1r^2 + k_2r^4 + k_3r^6) \\ y_{u\_rad} &= y_d(1 + k_1r^2 + k_2r^4 + k_3r^6) \end{aligned} \tag{4.13}$$

The (x_d, y_d) is the original location of the distorted point and (x_{u_rad}, y_{u_rad}) is the corrected location which compensated by radial distortion factors. The tangential distortion is a manufacturing defects process. The lens is not being exactly parallel to the image sensor. The distortion can be described from two parameters p_1 and p_2 as.

$$\begin{aligned} x_{u\_tan} &= x_d + [2p_1y + p_2(r^2 + 2x^2)] \\ y_{u\_tan} &= y_d + [p_1(r^2 + 2y^2) + 2p_2x] \end{aligned} \tag{4.14}$$

The (x_{u_tan}, y_{u_tan}) is the corrected location which compensated by tangential distortion factors. The total lens distortion comes up the summation of radial and tangential distortion. The above model is extended as:

$$\begin{aligned} x_u &= x_d(1 + k_1r^2 + k_2r^4 + k_3r^6) + x_d + [2p_1y + p_2(r^2 + 2x^2)] \\ y_u &= y_d(1 + k_1r^2 + k_2r^4 + k_3r^6) + y_d + [p_1(r^2 + 2y^2) + 2p_2x] \end{aligned} \tag{4.15}$$

Where: (x_u, y_u) is undistorted location point. (x_d, y_d) is distorted location point. (x_c, y_c) is center of distortion (principal point). p_n is tangential distortion coefficient. k_n is radial distortion coefficient, $|\underline{r}| = \sqrt{(x_d - x_c)^2 + (y_d - y_c)^2}$. When the calibration is done, the distortion parameters are known and we can compensate for the data lost through distortion.

4.5.1 Jean-Yves Bouguet camera's calibration toolbox for MATLAB

As already mentioned in the previous section, the C implementation can easily be found by OpenCV which opens the source library based on C language. Nevertheless, the same method is implemented in MATLAB by Bouguet [77]. He created the MATLAB toolbox, which can find the intrinsic parameters (camera model) accurately. The advantage of his toolbox is that it does not only provides the accurate intrinsic and geometrical output, but also the graphical three dimensional fundamental behaviors of the camera that are easy to understand. This calibration method requires the checkerboard which has symmetrical shapes and knows the exact dimension. The used checkerboard is window size $30 \times 30\ mm$, 5 squares per side. Figure 4.22 shows the setting up CCD/PMD camera. Checkerboard places about $60\ cm$ far from the camera. The CCD camera is fixed above the PMD camera. The mechanical part has to be forced on the accurate position because changing mechanical setting will directly affect the calibration matrix, which effects the fused data. Figure 4.23 shows the field of view (FOV) of CCD and PMD cameras. The field of view depicts that both cameras are different. Thus not only the projected point, but also the resolution is dissimilarity as well. Thus, this figure strongly shows that the calibration is very important before fusing the images data.

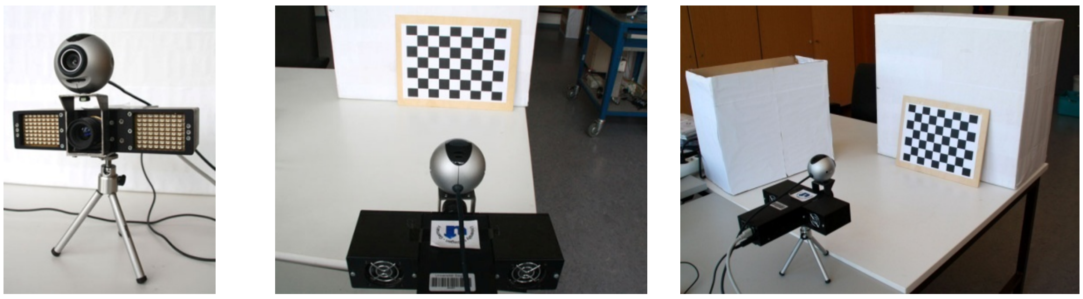
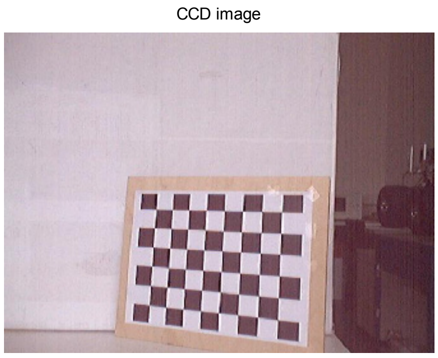
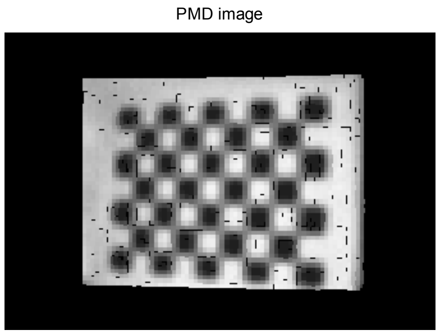


Figure 4.22 Calibration setting up



(a) CCD camera



(b) PMD camera

Figure 4.23 Images from CCD/PMD camera on the setting up position

Bouguet's toolbox is very powerful for calibrating the single camera, the stereo camera as well as the fish eye camera. The stereo camera calibration is applied in this work to find the relation of 2D and PMD camera in order to find the rotation and translation between both cameras. Then, the geometrical relation is used to fuse CCD and PMD data. Bouguet's method separates calibration into three steps.

Firstly, take checkerboard images from CCD and PMD 14 pairs. These images are used for calibration the coordinate.

Secondly, extract grid corners from the checkerboard. By extracting the corner from the CCD camera, the corner can be accurately found. On the other hand, the PMD camera has a problem. Since the resolutions of the PMD camera are small, the corner is rarely detected. Setting the ambient light is important. The experiments had to be corrected many times before getting a satisfactory output. The collected data from CCD and PMD have to be captured at the same time with the ambient environment.

Thirdly, calibrating the output of calibration provides five terms, as shown below. The intrinsic parameters [78] [79]consist of

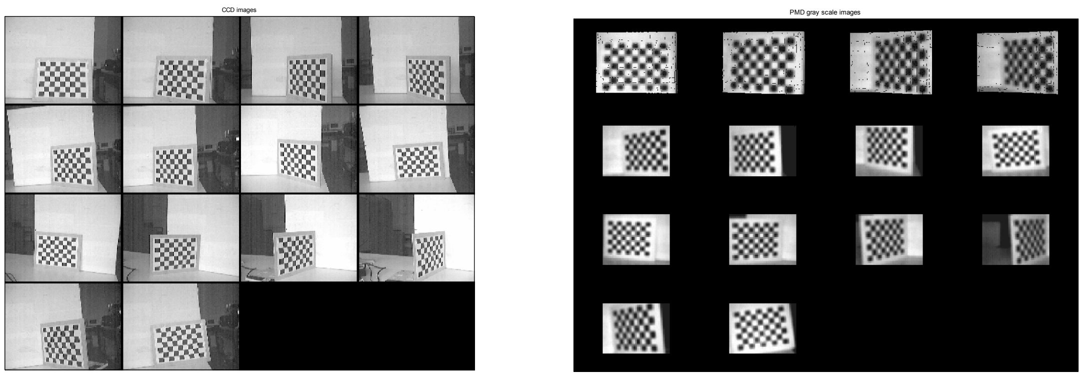
Focal length$(\underline{f})$: the focal length in pixels $[x_{axis}, y_{axis}]$

Principal point$(\underline{cc})$: the principal point coordinates $[a_0, b_0]$

Skew coefficient$(\underline{alpha_c})$: the skew coefficient defining the angle between x and y pixel axes.

Distortions$(\underline{kc})$: the image distortion coefficients (radial and tangential distortions) $[k_1, k_2, l_1, l_2]$

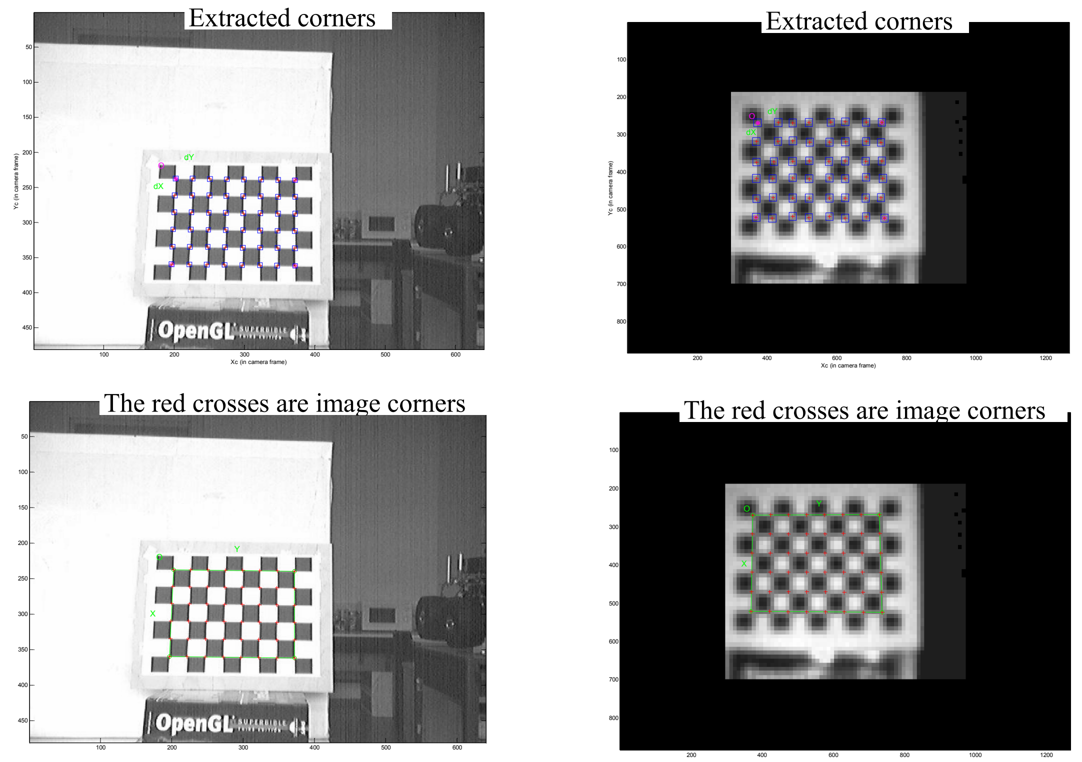
Pixel error$(\underline{err})$: the possibility of an error in each pixel $[x_{axis}, y_{axis}]$



(a) CCD images (b) PMD images

Figure 4.24 First step, take checkerboard images pairs

Moreover, this toolbox creates the graphical view of the camera extrinsic parameters. The extrinsic parameters show both camera centered point and world centered point, which is easy to understand. Figure 4.24 shows the first step, image pairs of CCD and PMD. Each image pair has to be recorded at the same time at the same condition; otherwise the related coordinate of calibration will be wrong. From the figure, we can see that the dissimilarity resolution of both cameras that is huge (10 times, data sheet), and field of view has a small wrinkle, which has to be ironed out. Figure 4.25 shows the second step, extracting corners from the checkerboard in order to use the coordinate to calibrate intrinsic parameters afterward. Table 4.3 shows the summary of the intrinsic parameter of CCD camera. The intrinsic parameter is necessary in order to find the project image in the processor chip. Figure 4.26 shows the graphical view of extrinsic parameter in camera/world centered mode, making it an accessible understanding for the calibration method. Figure 4.26(a) shows 14 graphical square images. They represent the random rotation of checkerboard. Trapezoidal red square represents the model of CCD camera. Trapezoidal peak to square base represents the CCD focusing point.



(a) CCD images (b) PMD images

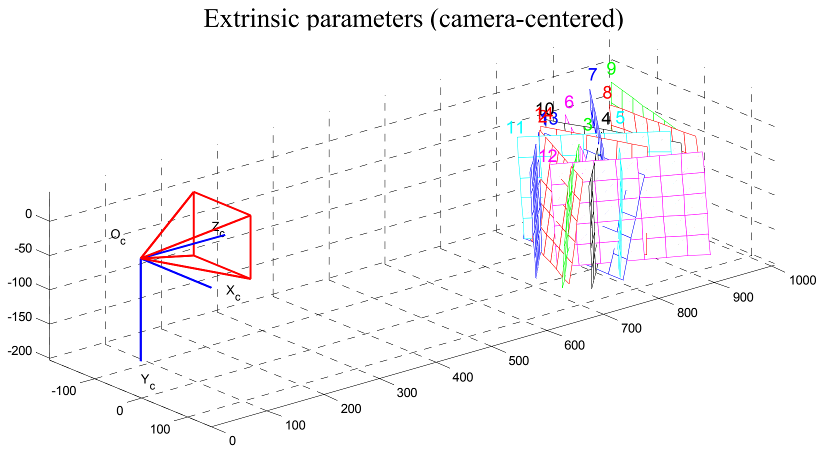
Figure 4.25 Second step, extract corners

Figure 4.26(b) adapts the data from (a), arranging new coordinates between the camera and checkerboard. The checkerboard is fixed to the one position, but the camera is rotated around the checkerboard instead. Figure 4.26(c) shows a lens pixel error. Ideally, the lens should not have the distortions. The main reason is the quality of the manufactured product. The high ideal quality lens is the parabolic shape, but almost low quality lens are the spherical shape instead because it is easy to build.

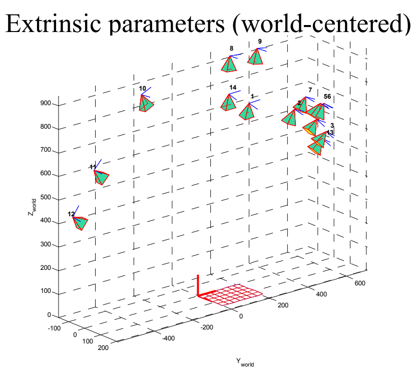
Lens distortion occurs in the rays farther from the lens center. It is bent because the projected image on CCD sensor is distorted. This distortion is the radial distortion. This distortion occurs from the shape of lens. The distortion in the center of the image is zero and is increased by the distance toward to periphery. The second distortion is the tangential distortion. It comes from the lens position and is not exactly parallel to the image sensor. The projected image is bent out of focus. The combination of both distortions is the complete distortion. Figure 4.27 visualizes the distortion error of CCD cameras: (a) radial distortion (b) tangential distortion and (c) complete distortion that is the combination of radial and tangential distortion. The tangential distortion of the CCD camera is not so good. The complete distortion is then showing as an amorphous shape.

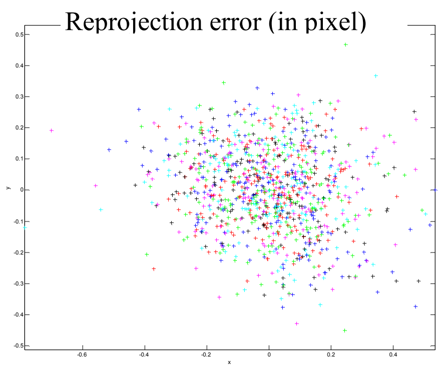
Intrinsic parameters of CCD camera:	
Focal Length: $\underline{f}$	[780.037 777.844] $\pm$ [6.28701 6.224]
Principal point: $\underline{cc}$	[353.244 223.784] $\pm$ [10.114 9.484]
Skew: $\underline{alpha\_c}$	[0.000] $\pm$ [0.000]=> angle of pixel axes = 90.000 $\pm$ 00000 degrees
Distortion: $\underline{kc}$	[0.024 0.015 -0.006 0.0109 0.000] $\pm$ [0.058 0.393 0.005 0.004 0.000]
Pixel error: $\underline{err}$	[0.179 0.142]

Table 4.3 CCD camera intrinsic parameters



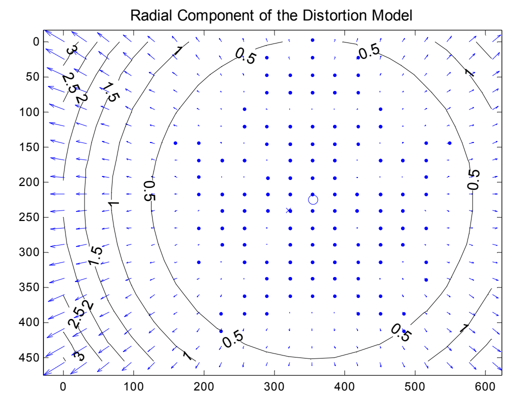
(a) Extrinsic parameter camera centered





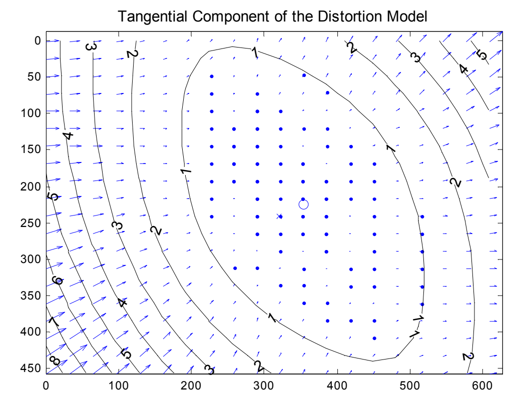
(b) Extrinsic parameter world centered (c) pixel error

Figure 4.26 CCD geometrical graphic related to checkerboard parameters



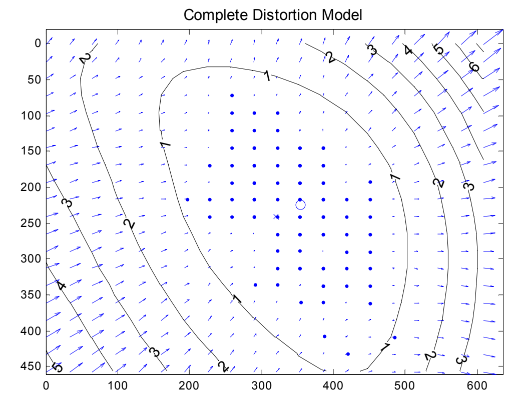
```
Pixel error            = [0.1795, 0.143]
Focal Length           = (780.037, 777.844)         +/- [6.287, 6.225]
Principal Point        = (353.244, 223.784)         +/- [10.11, 9.484]
Skew                   = 0                          +/- 0
Radial coefficients    = (0.02406, 0.01588, 0)      +/- [0.0585, 0.3939, 0]
Tangential coefficients = (-0.006842, 0.01079)      +/- [0.005877, 0.004345]
```

(a)



```
Pixel error            = [0.1795, 0.143]
Focal Length           = (780.037, 777.844)         +/- [6.287, 6.225]
Principal Point        = (353.244, 223.784)         +/- [10.11, 9.484]
Skew                   = 0                          +/- 0
Radial coefficients    = (0.02406, 0.01588, 0)      +/- [0.0585, 0.3939, 0]
Tangential coefficients = (-0.006842, 0.01079)      +/- [0.005877, 0.004345]
```

(b)



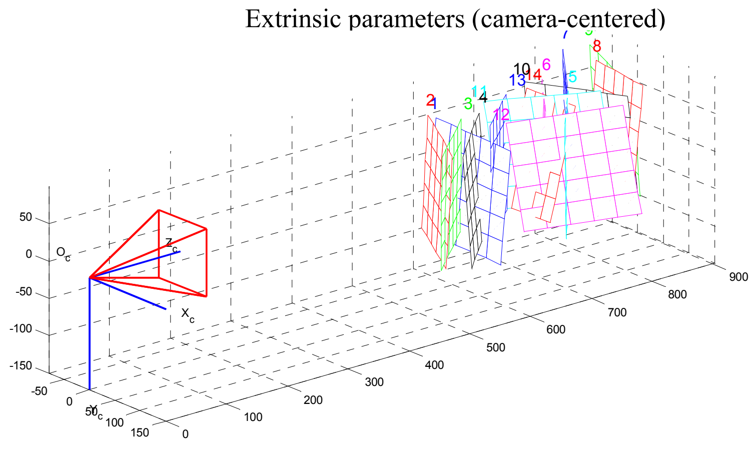
```
Pixel error            = [0.1795, 0.143]
Focal Length           = (780.037, 777.844)         +/- [6.287, 6.225]
Principal Point        = (353.244, 223.784)         +/- [10.11, 9.484]
Skew                   = 0                          +/- 0
Radial coefficients    = (0.02406, 0.01588, 0)      +/- [0.0585, 0.3939, 0]
Tangential coefficients = (-0.006842, 0.01079)      +/- [0.005877, 0.004345]
```

(c)

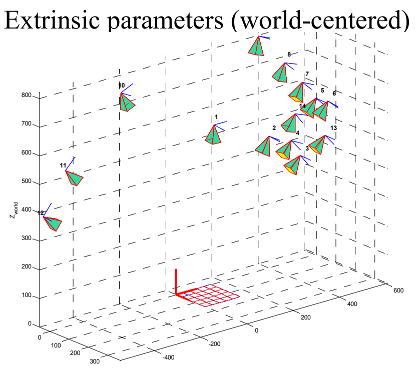
Figure 4.27 CCD Camera distortion visualization

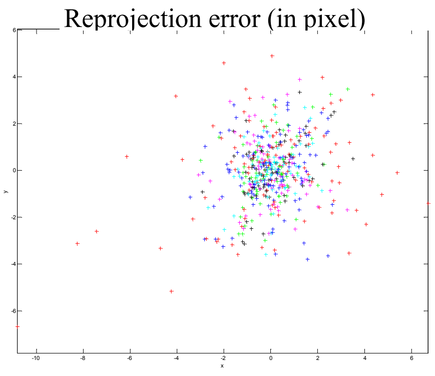
Intrinsic parameters of PMD camera:	
Focal Length: $\underline{f}$	[2049.807 1460.607]$\pm$[89.570 60.338]
Principal point: $\underline{cc}$	[571.588 432.630] $\pm$ [87.242 41.540]
Skew: $\underline{alpha\_c}$	[0.000] $\pm$ [0.000]=>angle of pixel axes = 90.000$\pm$0.000 degrees
Distortion: $\underline{kc}$	[-0.245 7.826 -0.023 -0.020 0.000]$\pm$[0.372 8.167 0.015 0.015 0.000]
Pixel error: $\underline{err}$	[1.560 1.504]

Table 4.4 PMD camera intrinsic parameters



(a) Extrinsic parameter camera centered





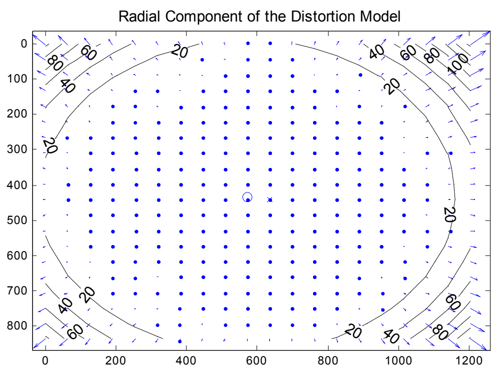
(b) Extrinsic parameter world centered (c) pixel error

Figure 4.28 PMD geometrical graphic related to checkerboard parameters

Table 4.4 shows the intrinsic parameters of the PMD camera in the same condition of the CCD camera. Figure 4.28 (a-b) shows the graphic between camera and checkerboard, we can see that the coordinate of PMD and checkerboard compared to Figure 4.26(a-c) are nearly the same. This confirms that the setting of the position of calibration method is collectedly done. Figure 4.29 shows the visualization distortion error of PMD camera: (a) radial distortion (b) tangential distortion and (c) complete distortion that is the combination of radial and tangential distortion.

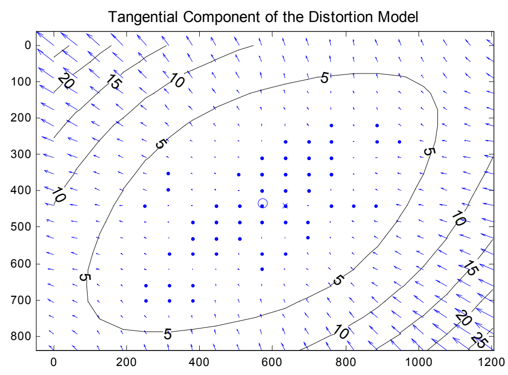
4.5.2 Two cameras calibration

Human eyes receive images from objects from two different positions. The human brain knows the focusing point of each eye and can differentiate between the focusing points of each eye (base line). The triangular principle can help us to calculate the distance from objects to the camera. This idea inspired the stereo camera system. A stereo camera is a type of camera with two lenses with separated image sensors, which can provide three dimensional images or movies. To calculate the stereo system, the essential things that have to be considered are as follows. *Firstly,* the radial and tangential lens distortions (undistortion factor) have to be removed. *Secondly,* the angle and the distance between the cameras are fixed. The output is two coplanar with exactly aligned rows. *Thirdly,* the correspondent features are found in the left and right camera views. *Fourthly,* the geometric of the camera turns into distances using the triangular principle. This method is called reprojection.



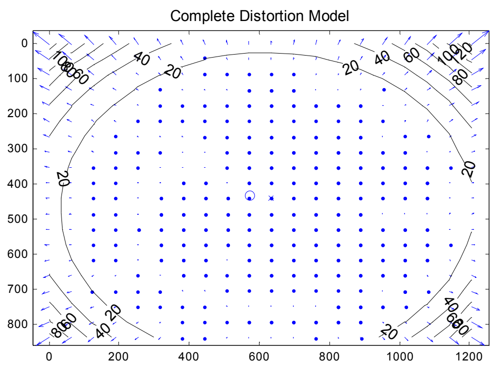
```
Pixel error              = [1.56, 1.504]
Focal Length             = (2049.8, 1460.61)          +/- [89.57, 60.34]
Principal Point          = (571.594, 432.632)         +/- [87.24, 41.54]
Skew                     = 0                                     +/- 0
Radial coefficients      = (-0.2454, 7.827, 0)        +/- [0.3723, 8.168, 0]
Tangential coefficients  = (-0.02321, -0.02043)       +/- [0.01551, 0.0156]
```

(a)



```
Pixel error              = [1.56, 1.504]
Focal Length             = (2049.8, 1460.61)          +/- [89.57, 60.34]
Principal Point          = (571.594, 432.632)         +/- [87.24, 41.54]
Skew                     = 0                                     +/- 0
Radial coefficients      = (-0.2454, 7.827, 0)        +/- [0.3723, 8.168, 0]
Tangential coefficients  = (-0.02321, -0.02043)       +/- [0.01551, 0.0156]
```

(b)



```
Pixel error              = [1.56, 1.504]
Focal Length             = (2049.8, 1460.61)          +/- [89.57, 60.34]
Principal Point          = (571.594, 432.632)         +/- [87.24, 41.54]
Skew                     = 0                                     +/- 0
Radial coefficients      = (-0.2454, 7.827, 0)        +/- [0.3723, 8.168, 0]
Tangential coefficients  = (-0.02321, -0.02043)       +/- [0.01551, 0.0156]
```

(c)

Figure 4.29 PMD Camera distortion visualization

The combination of CCD and PMD cameras is similar to a stereo camera system. Then, we apply the stereo theory for fusing CCD/PMD cameras. After getting the initial intrinsic parameters, CCD and PMD are assumed to point in the same direction, which has an overlap area of both cameras. On the other hand, some areas are not overlapped. The extrinsic parameters of both cameras are shown in the previous section. Therefore we can use these parameters to sort out the relative matrix, which is shown in Table 4.5. This table shows the $\underline{R}$ and $\underline{T}$, which relates two cameras to each other. We can see that the CCD camera is far from the PMD center approximately 14.31, 63.96 and 70.83 mm in the x, y and z axis, respectively. The rotation from the PMD center with an angle 0.05058, 0.07940 and -0.01120 in rows, pitch and yaw, respectively. It can be easily understood from Figure 4.31 that calibration data is matched with physical setup in Figure 4.22.

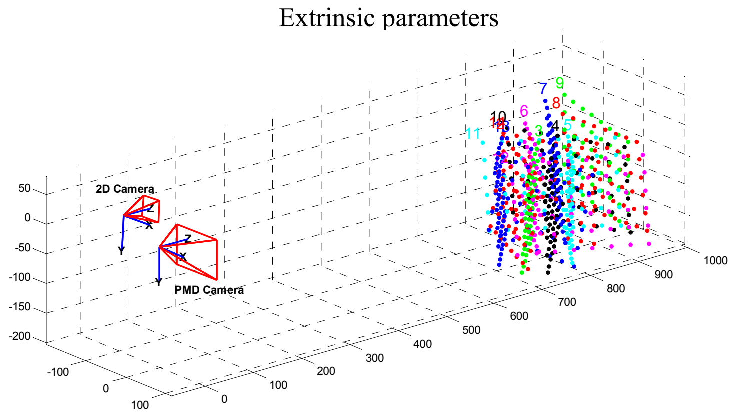


Figure 4.30 Extrinsic parameter between CDD/PMD cameras

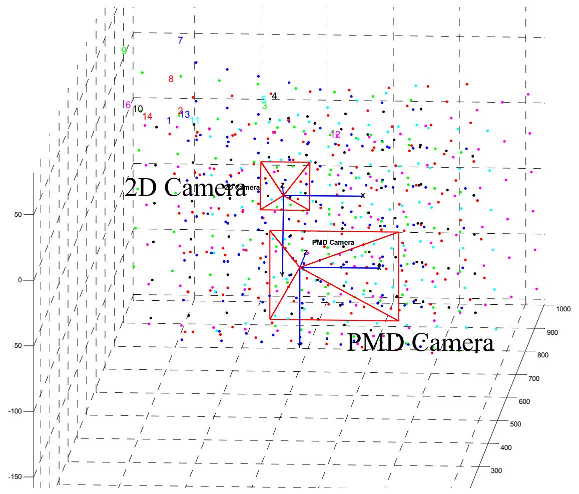


Figure 4.31 Relative coordinate between CCD/PMD cameras

The two coordinate vectors $\underline{x_L}$and $\underline{x_R}$ are respectively the left and right camera reference frames that are related to each other through the rigid motion transformation $\underline{x_R} = \underline{Rx_L} + \underline{t}$,where $\underline{R}$ is the 3x3 rotation matrix corresponding to the rotation vector from the calibration result. It is assumed that the object is projected onto the CCD frame and the PMD frame in different orientations. A CCD frame is transformed into the PMD frame through a rotation and translation matrix. Denote $(x_{PMD}, y_{PMD}, z_{PMD})$ are the coordinates of the PMD camera on artificial frame and $(x_{CCD}, y_{CCD}, z_{CCD})$ are the coordinates of the CCD camera frame. Then, the CCD frame can be projected onto the PMD frame by the equation shown in equation (4.16)

Relative vector (cm)	
Rotation vector ($\underline{r}$)	[0.05058 0.07940 -0.01120]
Translation vector ($\underline{t}$)	[14.31872 63.96501 70.83500]^T

Table 4.5 Relative matrix between CCD/PMD

$$\begin{bmatrix} x_{PMD} \\ y_{PMD} \\ z_{PMD} \end{bmatrix} = [\underline{R} \vdots \underline{t}] \begin{bmatrix} x_{CCD} \\ y_{CCD} \\ z_{CCD} \\ 1 \end{bmatrix} \tag{4.16}$$

$$\underline{q_{PMD}} = [\underline{R} \vdots \underline{t}]\underline{q_{CCD}} \tag{4.17}$$

4.6 Data fusion

The depth 3D information is a consistent and dominates point of the PMD camera, but the gray scale output still has pattern lacking. To achieve this, the high resolution 2D camera is operated to register 2D/3D vision providing more realistic correlative 3D color images, which are prepared for the future SLAM raw sources. The 3D scene is captured from PMD while the texture is mapped by the 2D camera. The resolution of the 3D data from the selected PMD camera is 64×48 pixels, which is approximately ten times less than the 2D camera (640x480 pixels). Huhle et al. [80] and Prasad et al. [5] presented the capability of these cameras to attain the high resolution, gray scale image and depth data. The gray scale image from the PMD camera is only used for the first time to calibrate the camera, after that this data is ignored. The interpolation method is then used for adjusting PMD image size in order to be equal with 2D data set. Figure 4.32 illustrates the interpolated depth and 2D data. Each depth data registers to the nearby 2D 10 pixels. The bunch of pixels has the same depth data [81].

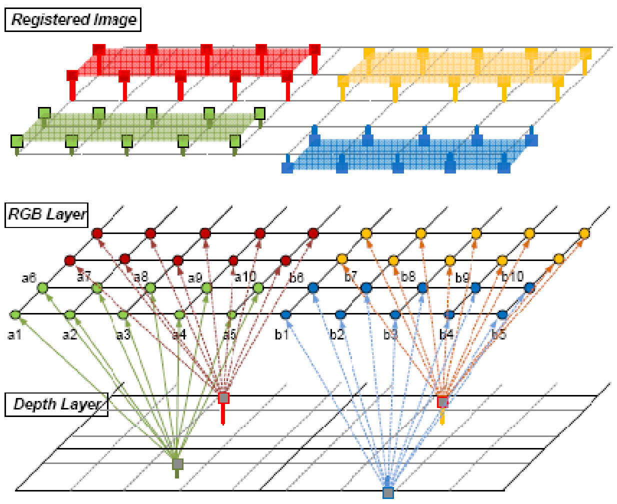


Figure 4.32 Registration and rescale image sets

$$P_{PMD}(x_n, y_m) = Q_{2D}(i_n^{n+10}, j_m^{m+10})$$
$$m, n \le Q_{\max\_\text{size}} \tag{4.18}$$

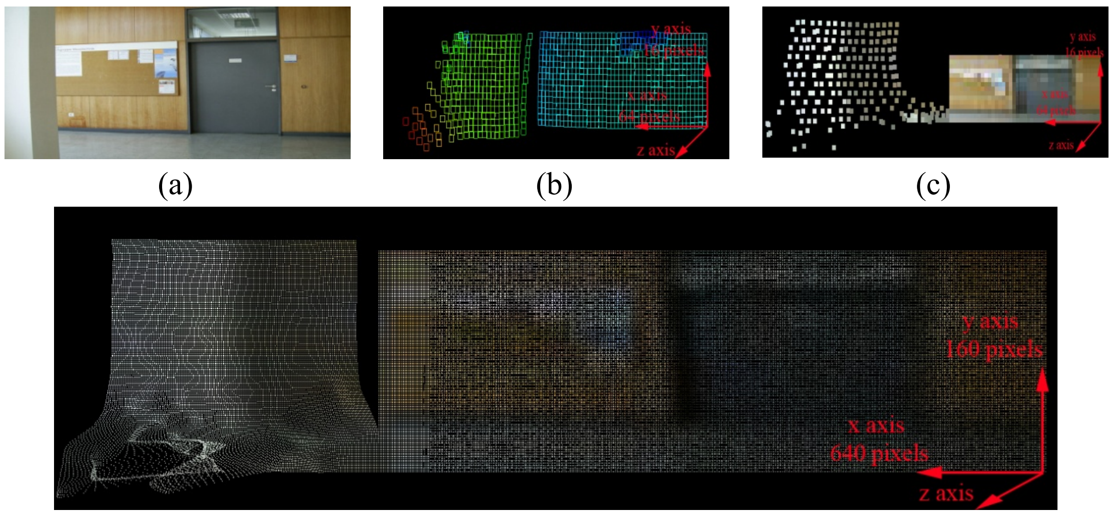


Figure 4.33 Output of (a) 2D camera (b) PMD camera (c) Fusion 2D/3D data (d) Rescaling fusion 2D/3D data

The bunch of pixels has the same depth data as in equation (4.18). P_{PMD} is the new matrix depth data and Q_{2D} is the RGB data from the 2D camera. The 3D mapping yields the texture for the 3D model. The OpenGL is subsequently used to display the entire 3D mapping output. The proposed method straightforwardly presents the achievement of the 3D mapping building. This ensures an easy

calibration of both cameras and there is only a small loss of information. Figure 4.33 (a-d) shows the output of the image before and after enhancement.
One experiment uses PMD A2 [82]. The depth data has to be rescaled by interpolating point to point. The raw data resolution is up from 64*x*16 pixels to 1280*x*320 pixels (20 times). Figure 4.34 shows the scenario around the second floor stair. This area has the complex information. (a) shows the depth data from the PMD camera. The data without processing is difficult to recognize or sort out the scenarios. (b) Categories depth data using the color range difference. The results look better, but can roughly recognize the scenario. (c) shows the fusion of 2D/3D data, the output can look clearly at what the scenario is because of the texture information from the 2D camera. Moreover, the depth can show the volumetric scenario. The area around the windows has the most erroneous data because of the glass reflection. However, for the rescale approach, we only suggest one frame capturing. In real time, 3D mapping is not necessary because it consumes the processing time. It has an effect on the real time application. Compared to the mapping quality which we will receive, it is worse.

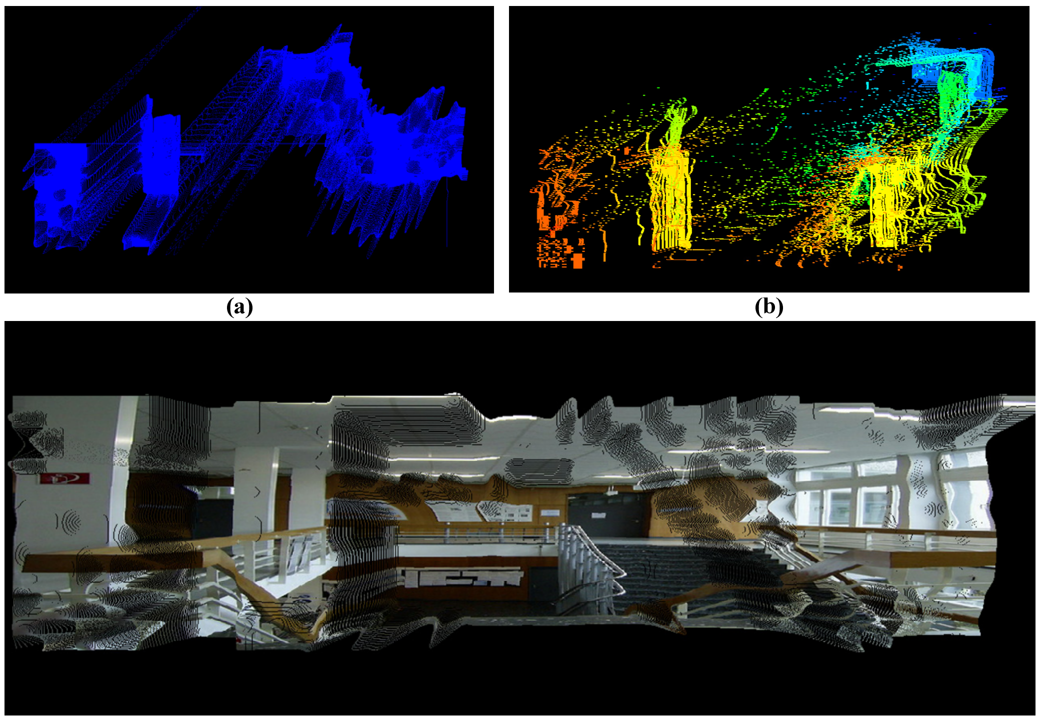


Figure 4.34 (a) Raw depth PMD data (b) Category depth using color (c) Fusion 2D/3D

4.7 Object detection

Nowadays, many complex functions are included in one complex mobile robot such as a wireless communication, Global Positioning System (GPS), smart obstacle avoidance, vision systems as well as self localization techniques. On the other hand, the real time three dimensional map building task to obtain the realistic visual appearance of particular environmental volume is also a very challenging task for modern mobile robots. The Simultaneous Localization and Mapping (SLAM) is a technique used for map building. SLAM is employed on a mobile robot it can be used for localization task and helps the robot to move autonomously through an unknown environment. SLAM can be represented by various active sources, e.g. ultrasonic sensors and laser scanners or by passive sources, i.e. vision sensors from cameras. Cameras can be used to find unique characteristics of features based on the pixels in and around the features, whereas the active sources are not able to do that. Besides the high performance for 3D mapping and estimating the pose of the locomotion in real time, the flawlessly mobile robot should also handle tasks such as the recognition of objects in its environment in order to achieve various practical missions. This topic hence aims to fulfill the high performance robot tasks by the recognition of the objects using fusing the data from the 2D/3D sensors. The 2D high resolution image is registered on the PMD 3D volume. Subsequently rescale and used this data for camera calibrating. The visual input from the 2D camera does not deliver only high resolution texture data on 3D volume, but it is also used for object recognition.

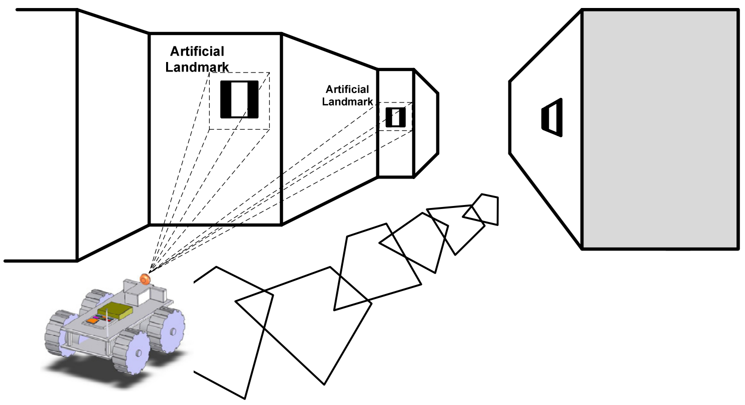


Figure 4.35 Coordination of SLAM and landmark recognition

The object detection is an important subfield element of various mobile robotic areas. Many tasks can be accomplished by extracting certain image features such as edges, color regions, textures and contours. A heuristic algorithm is used to find configurations, combinations of those features and specific objects of interest. However, it is difficult for artificial landmarks to recognize the features in a complex surrounding environment. Moreover heuristics handle the huge variety of instances of the object class. This topic describes algorithms for object detection based on object detection Haar-like features. The aims of object detection are developing navigation and vision systems for mobile robots applications. The object detection is used to detect the artificial landmarks which are on the robot's path. Mobile robots can move autonomously by following the hanging artificial landmarks. In mobile robotic tasks we want mobile robots to go to the door way autonomously, while detecting the artificial landmarks. The artificial landmarks are any figures or any symbols which are established by users. It should be unusual or totally different from the surrounding environment. It is probably easy for recognition and learning.

We attempt to reduce and assist the surrounding sensor which is used in mobile robots. Therefore, the CCD camera based on the object detection Haar-like features method, is capable of processing images extremely rapid and achieving high detection rates. The PMD camera is capable of capturing reliable depth images directly in real-time and it is a compact size with an affordable price which makes it attractive for versatile applications including surveillance and computer vision. Furthermore, the resulting gray scale image from the PMD camera can also be used for the basic vision recognition task. For these reasons, PMD is a novel and attractive tool for implementing the SLAM. However, the high performance SLAM is not only implemented only for a 3D map in real time, but it should also include recognizing objects and avoiding obstacles as well as estimating the trajectory simultaneously. The prominent output from the PMD camera provides spatial depth measurement from every pixel. The high resolution 2D camera is then combined with the PMD camera output in order to serve the complicated image processing tasks and support the complex machine learning requirements. Nevertheless, searching the objects within a complex environment is not an easy task. The Haar-like Features represent one method in which object searching takes place in real-time. This method can provide accurate and robust detection of the desired objects. It can be applied to detect any features, patterns, shapes and colors which might be present within complex environments. Figure 4.35 demonstrates an idea for the coordination of SLAM based on artificial landmark recognition. The different pixel resolutions and overlapping positions due to the machine setup are also considered in this work.

4.7.1 Rapid object detection using Haar-like features

Lienhart et al. [83] describes a visual object detection framework which is capable of processing images at extremely rapid detection rates and improved by Viola et al. [84]. Their method has been used widely for such things as faces, hands and body detection. We applied their algorithm to detect the artificial landmark in any environment without color needed. By now, their research has been included in the OpenCV library. Their algorithm is very powerful and can detect the objects with a high speed rate. The steps of the process are, first, a classifier is trained with thousands of sample views of a particular object, so called the positive samples and the negative samples. The Negative samples are the images which do not contain the interested objects. After a classifier is trained, it will be applied to a region of interest in an input image. The classifier output is "1" when that area is likely to show the object and "0" if another. To search for the objects in the whole image, one can move the search window though the image and check every location using the classifier. The classifier can be resized so that it is able to find the objects of interest of different sizes which is more efficient than resizing the image itself. They introduced a new image representation called the *"Integral Image"* which allows the features used by our detector to be computed very quickly. The second is a learning algorithm based on *AdaBoost,* which selects a small number of critical visual features from a larger set and yields extremely efficient classifiers. The third contribution is a method for combining increasingly more complex classifiers in a *"cascade"* which allows background regions of the image to be quickly discarded while spending more computation on promising object-like regions. Each classifier is the sum of rectangular followed by a selected threshold. In each round of boosting one feature is selected as the lowest weighted error. An input window is evaluated for every classifier of the cascade in which if the classifier returns false, that window evaluation is finished and the detector is returns false. If the classifier returns true, the window is passed to the next classifier in the cascade. The next classifier is evaluated in the same way. If the window passed through every classifier, then that window is respectable and the detector returns true. The processing time in windows that look like the expected object to evaluate and classify take a longer processing time than the non look like objects. The non look like object images are rapidly rejected in the beginning of the classification.

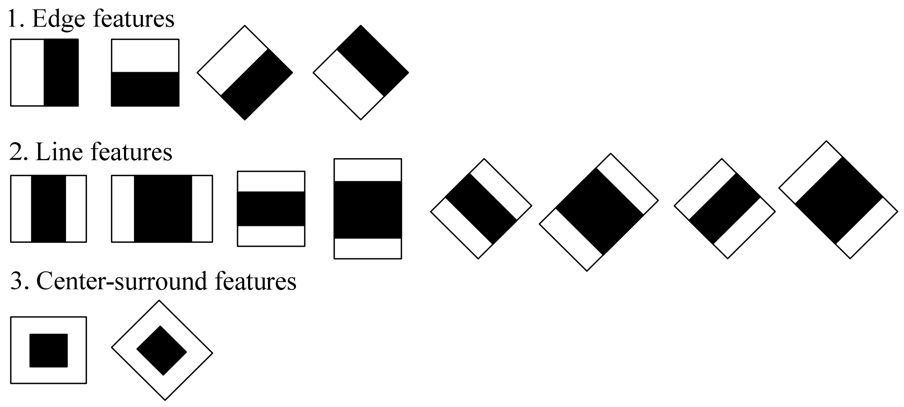


Figure 4.36 Feature prototypes of simple haar-like features. Black areas have a negative, and white areas have positive weights [85]

4.7.1.1 Feature detection using integral images

The learning algorithm principle uses features or raw pixel as input. The main reason to use the features instead of raw pixel is to operate much faster. A general pool of simple Haar-like features combined with feature selection can increase the capability of the learning algorithm. Figure 4.36 (a) shows the basic Haar-like features i.e. edge, line, diagonal and center surround features.

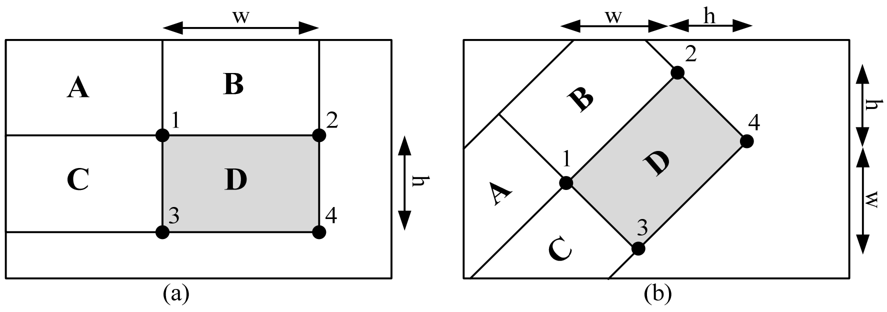


Figure 4.37 (a) Upright summed area (b) Rotated summed area [83]

A rectangle of pixels with top left corner (x, y), width w, height h and orientation $\alpha \in \{0°, 45°\}$. A rectangle is inside a window and specified by $\underline{r} = (x, y, w, h, \alpha)^T$. The summation of pixel is $PixSum(\underline{r})$. The set of used features is

$$\omega_1 PixSum(r_1) + \omega_2 PixSum(r_2) \tag{4.19}$$

Where the weights $\omega_1, \omega_2 \in \underline{R}$ is compensated value of different area size between the rectangles r_1 and r_2. Rectangle features can compute rapidly, an integral imaged I is an intermediate representation for the image and contains the sum of gray scale pixel values of image N with height y and width x

$$I = (x, y) = \sum_{\acute{x}=0}^{x} \sum_{\acute{y}=0}^{y} N(\acute{x}, \acute{y}) \tag{4.20}$$

The integral image can be computed by

$$\begin{gathered} I(x,y) = I(x,y-1) + I(x-1,y) + N(x,y) - I(x-1,y-1) \\ I(-1,y) = I(x,-1) = I(-1,1) = 0 \end{gathered} \tag{4.21}$$

Then only one scan over the input data is required. This intermediate representation $I(x,y)$ allows the computation of a rectangle feature value at (x,y) with height and width (h,w). The pixel sum of any upright rectangle $r = (x,y,w,h,0)$ can be determined by using four references, see figure Figure 4.37(a).

$$PixSum(r) = I(x,y) + I(x+w,y+h) - I(x,y+h) - I(x+w,y) \tag{4.22}$$

For the computation of the rotated feature, the sum of pixels of the rectangle rotated by 45° with the right most corner at (x,y) and extending till the boundaries of the image, see Figure 4.37(b).

$$I_r(x,y) = \sum_{\acute{x}=0}^{x} \sum_{\acute{y}=0}^{x-|\acute{x}-y|} N(\acute{x},\acute{y}) \tag{4.23}$$

From equation (4.23) the rotated integral image I_r can be computed by

$$\begin{gathered} I_r(x,y) = I_r(x-1,y-1) + I_r(x+1,y-1) - I_r(x,y-1) \\ + N(x,y) + N(x,y-1) \\ I_r(-1,y) = I_r(x,-1) = I_r(x,-2) = I_r(-1,-1) = I_r(-1,-2) = 0 \end{gathered} \tag{4.24}$$

The pixel sum of any rotated rectangle $r_r = (x,y,w,h,45°)$ can be determined by

$$\begin{gathered} PixSum(r_r) = I_r(x+w-h,y+w+h-1) + I_r(x,y-1) \\ - I_r(x-h,y+h-1) - I_r(x+w,y+w-1) \end{gathered} \tag{4.25}$$

The features are compositions of rectangles. The computation with several lookups and subtractions weighted with the area of the black and white rectangles is required. Threshold is automatically determined during a fitting process. The return values (α,β) of the feature are determined. An error of the examples is minimized. The examples are given in a set of images that are classified as positive or negative samples.

4.7.1.2 Learning classification functions

An AdaBoost approach is used to select a small set of features and train the classifier. In particular, AdaBoost learning approach is used to boost the classification performance of a simple learning approach. The key insight is generalization performance related to the margin of the examples. AdaBoost achieves large margins rapidly, using over 117,000 rectangle features with each image 24× 24 sub window. The weak learning algorithm is designed to select the single rectangle feature which best separates the positive and negative examples. Each feature allows the weak learner to determine the optimal threshold classification function. AdaBoost is a method to select a low number of good classification functions, namely "weak classifiers",$h_j(x)$ to form a final "strong classifier" which is a linear combination of the weak classifiers. θ_j is a threshold and p_j a parity indicating if f_j is bigger or less than the threshold for a positive classification.

Input: Training example $(x_i, y_i),\ i = 1, \dots, N$ where positive $(y_i = 1)$ and negative $(y_i = 0)$.

Initialize weights $w_{1,i} = \frac{1}{2m}, \frac{1}{2l}$ where m is number of negatives and l is the number of positive examples.

For $t = 1, \dots, T$

1. Normalize the weights, where $\omega_{t,i}$ is a probability distribution.
$$w_{t+1,i} = w_{t,i}\beta_t^{1-e_i}$$
2. For each feature, j train a classifier h_j which is restricted to usi a single feature. The error is evaluated with respect
3. to choose the classifier, h_t with the lowest error ϵ_t
4. Update the weights:
with $e_i = \begin{cases} 0: & x_i\ correctly\ classified \\ 1: & otherwise \end{cases}$ and $\beta_t = \frac{\epsilon_t}{1-\epsilon_t}$
5. The final strong classifier is:
$$h(x) = \begin{cases} 1: & \sum_{t=1}^{T} \alpha_t h_t(x) \geq 0.5 \sum_{t=1}^{T} \alpha_t \\ 0: & otherwise \end{cases}$$
where $\alpha_t = \log\left(\frac{1}{\beta_t}\right)$

Table 4.6 The AdaBoost algorithm for classifier learning

$$h_j(x) = \begin{cases} 1 : if\ p_j f_j(x) < p_j\theta_j \\ 0: otherwise \end{cases} \tag{4.26}$$

In practice no single feature can perform the classification task with low error. Features which are selected in early rounds of the boosting process had error rates between 0.1 and 0.3.

4.7.1.3 Cascade of classifiers

The cascade of classification is increased by the detection performance rate and a reduced computation time. The boosted classifiers can enhance the rejected negative sub window rate and detect the rate of the positive sub window. Classifiers are used to reject the majority of sub windows before doing the next classifiers step. It can enhance the low false positive rates.

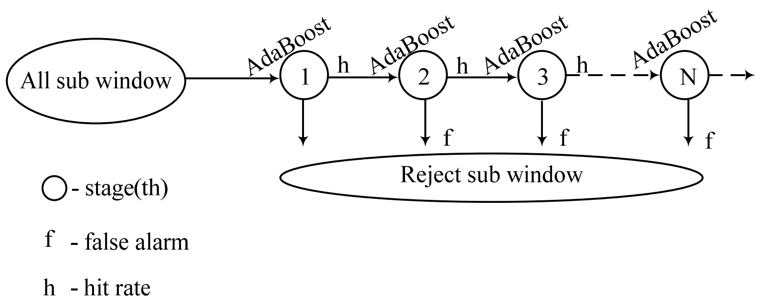


Figure 4.38 Cascade of classifiers with *N* stages. In each stage, the classifier is trained to achieve a hit rate of *h* and a false alarm rate of *f*

The principle of the cascade of classifiers is the generation of the decision tree where each stage is trained to detect all objects of interest and reject a certain fraction of the non-object patterns. Each stage uses the AdaBoost approach (Table 4.6) for the principle training algorithm. AdaBoost is a machine learning algorithm which can make a strong classifier based on a large set of weak classifiers by reweighting the training samples. Increasing the stage number and the number of weak classifiers is needed to achieve the desired false alarm rate at the given hit rate [84] [86].

4.7.2 Experiment description

In this work a scheme adaptation of the Haar-like features [76] is applied. The Haar-like Features are the real-time, accuracy and robustness approach for detecting the object. It can be used to detect any features, patterns, shapes, colors even in a complex environment. The artificial landmarks contrasts from the environment are easy to notice. Some kinds of artificial landmarks which we invent are shown in Figure 4.39. They consist of the mathematical shape e.g. square, circle and triangle with black or white edges. In research, we used square figures (the first left hand side).

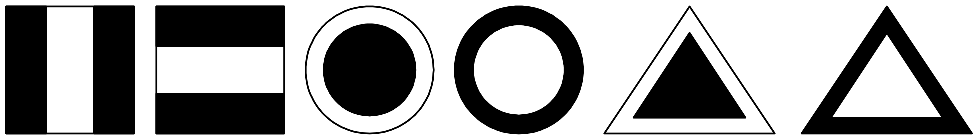


Figure 4.39 The samples of artificial landmarks

Object detection is an important element of various mobile robotic areas. Some tasks can be accomplished by extracting certain image features such as edges, color regions, textures, contours etc. However, for artificial landmarks with complex surrounding environments, it is tough to find the features. The applications of object detection are examples of autonomous moving follow the artificial landmarks. The robot can autonomously move by tracking the artificial landmarks which are hanged on working path. In industrial works, mobile robots were used to carry stuffs from the starting point to the destination point e.g. carry the tool boxes, heavy machines, spare parts and garbage.

Figure 4.40 Non-artificial landmarks samples (7170 negative samples)

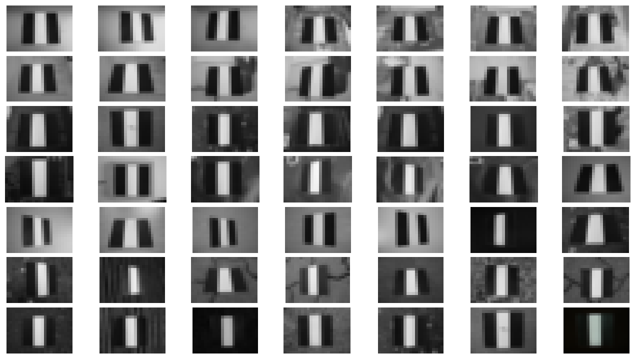


Figure 4.41 Artificial landmarks samples (1500 positive samples)

Regarding the real industrial applications, the mini electrical cars and folk lift trucks can be used by these applications. They use an electrical energy for the main sources. We can use the mobile robot instead of the human control vehicles. The collected set of positive examples which is artificial landmarks under different viewing conditions and a set of negative examples which are from random cropped image regions and do not contain the artificial landmarks. To yield the lower false rates, we successively collected misdetections from an image which does not contain the landmark into the negative sample set. The negative set has to be much more than the positive set in order to keep the false positive rate. All training examples had the size 20x20 and were normalized to have zero mean a standard deviation of 1.0. The cascade of classifiers has to collect the negative and positive samples [87]. The negative samples are the images that must not contain any object inside for detecting requirement and artificial landmark. In this experiment, the 7170 negative samples are used (see Figure 4.40). The positive samples are the images that must contain the artificial landmark. There are 1500 positive samples which were used in the experiments. Figure 4.41 shows the positive samples using the classifiers. This work uses the Intel Pentium 4, 3 GHz, 3GB RAM for training. The training times are approximately 80 hours.

4.8 Conclusion

This chapter explains the principle of Euler angles that is an important theory in order to understand the behavior of three dimensional locomotion. Euler angles explain the fundamental of rotation in each axis row, pitch, yaw as well as rotation in the three axis at the same time. The 3×3 matrices output describes can be left out the degree of freedom of both cameras and objects. Fundamentals of PMD cameras which are used for the main capturing sensor was described. The principle of the light source and the PMD chip were explained in order to clearly comprehend the PMD camera processing. Several types of PMD cameras were explained and the overall characteristics were concluded. The pros and cons between the PMD camera and other candidate sensors (laser scanner, range finder, sonar, vision) were compared. The PMD camera was the best sensor in terms of fast data volume capturing rate and precise data providing. However, the main cons are low resolution and effected by reflecting surfaces. The idea of the combination of the PMD and CCD camera was then proposed. In addition, median filtering was implemented in order to eliminate pepper-salt noise which is an unpredictable occurrence from the PMD camera capturing because the scenario in which the robot moves is very varied. The median filtering could eliminate the outliner noises from the grouping data. However, the noise and low resolution of the PMD camera is still the expression. The idea of fusion high resolution to PMD's depth data then was proposed in this chapter as well.

First of all, the calibration process has to be handled in order to get the intrinsic/extrinsic parameter of both cameras. Camera parameters are used to find out the relation between CCD and PMD cameras. Output from the relative matrix is used to optimize the position of RGB frame and the depth frame. The depth frame is fixed and the RGB frame projects to the depth frame with the calibration matrix from the calibration output. The MATLAB calibration toolbox is used in this work.

To enhance the performance of mobile robot applications, the object detection is included in these works. High resolution RGB data is used to sort out the object detection. When objects are detected, we get to know the coordinate from the objects to the camera from the PMD data because both images have already been registered. Eventually, the object detection in 2D and 3D were shown in the experiments.

Chapter 5

Simultaneous Localization and Mapping, SLAM

In Chapter1, the Simultaneous Localization and Mapping (SLAM) was previewed. This chapter again presents in depth the technique of SLAM and considers the three dimensional mapping problems. As a simple mobile robot behavior, all parts have to cooperate with each other perfectly. The set of robot wheels is connected to motors, microcontroller, computer, cameras, actuators and sensors for controlling the locomotion of the robot. The sensor for controlling the robot moves properly, and the image for the camera is used remotely by an operator to map inaccessible places. The camera provides the visual information for the operator to understand what the surrounding objects are, and how the robots can recognize them. Determining the locating of objects in an unknown environment and meanwhile locating the robot position related to these objects, is an example of SLAM. The aims of SLAM are to let robots make the maps without any previous map information or human assistance whatsoever. Maps could be made in areas, which are dangerous or inaccessible to humans such as deep-sea, mine, cave and unstable structures.

Concerning the development of the SLAM, we will first of all examine the localization and mapping individually to better understand how and why we should find a simultaneous solution. If a solution to the SLAM problem could be found, it would open the door for innumerable mapping possibilities where human assistance is cumbersome or impossible. A solution to SLAM should obviate outside localization methods like GPS or man-made beacons. It makes robot navigation possible in places like damaged (or undamaged) space stations and other planets. Even in the locations where GPS or beacons are available, a solution to the SLAM problem would be invaluable. Currently GPS is only accurate to within about one half of a meter, which is often more than enough to be the difference between successful mapping and getting stuck. Placing man-made beacons is expensive in terms of time and money. In situations where beacons are an option, simply mapping by hand is almost always more practical.

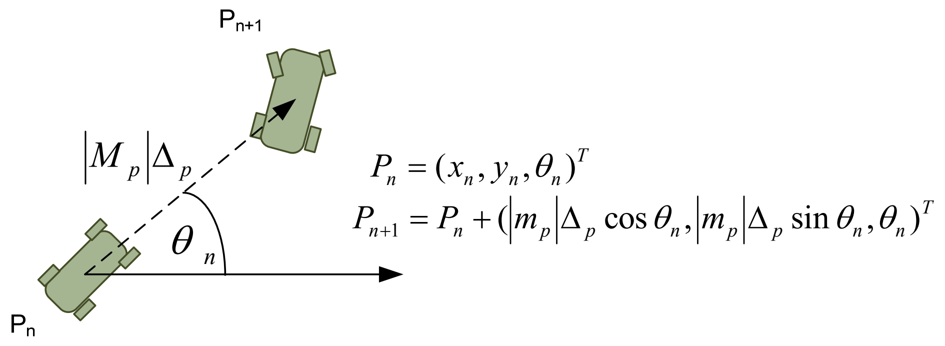
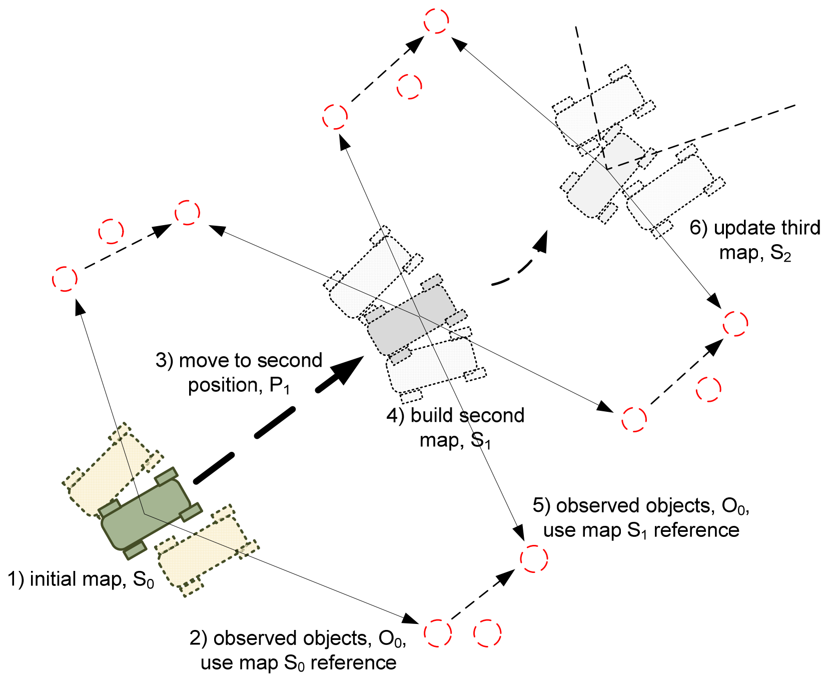

P_{n+1}
$|M_p|\Delta_p$
θ_n
P_n
$P_n = (x_n, y_n, \theta_n)^T$
$P_{n+1} = P_n + (|m_p|\Delta_p \cos\theta_n, |m_p|\Delta_p \sin\theta_n, \theta_n)^T$

Figure 5.1 Principle of robot locomotion

Figure 5.1 shows a principle of how to calculate the present position of the robot. Where P_n is the present position with the original x_n, y_n and θ_n are starting coordinates. P_{n+1} is the next position with accumulative distance from the previous position. M_p and θ_n are two main parameters to change the robot position. M_p is the vector from the previous position to the present position. θ_n is the steering angle related to the previous position to the present position as well. But the real experiments of the triangle principle cannot be used accurately because slipping between the robot wheel and ground, the complex environment, mechanical errors as well as the real experiment is a three dimensional system. The considering of the x and y axis only is not sufficient, the z axis has to be included in the conditions. Biber et al. [88] presented a method for the n scan matching technique that is built upon existing pairwise scan matching algorithms. They presented a theoretical motivation and showed a basic solution to include n scan matching into the SLAM technique. Zhao et al. [89] used two 2D laser range finders for acquiring 3D data. One laser scanner is mounted horizontally and another vertically. The latter one grabs a vertical scan line, which is transformed into 3D points based on the current robot pose. Since the vertical scanner is not able to scan sides of objects, Zhao et al. used two additional, vertically mounted 2D scanners, shifted by 45 to reduce occlusions

5.1 The principle of Simultaneous Localization And Mapping, SLAM

One of the major problems to develop the SLAM is the accumulation of the pose error. The accumulation exists because an error in localization has a universal effect on the perceived location of all features. Understanding and utilizing the relationship among errors in feature locations and robot pose is at the core of SLAM research. It is the major motivation behind solving localization and mapping concurrently. The SLAM is a technique to use autonomous mobile robots to create a map within an unknown environment without a priori knowledge, or update a map within a known environment from a given priori map to keep track of their current location, whether the robot has the priori map or generates the map itself. The final maps are used to determine the localization in an environment. Maps hence support the assessment of the actual location as obtained by means of navigation.



6) update third map, S_2
3) move to second position, P_1
4) build second map, S_1
5) observed objects, O_0, use map S_1 reference
1) initial map, S_0
2) observed objects, O_0, use map S_0 reference

Figure 5.2 Principle of mapping

The precision of any locating steps supports a locally relevant map. That improves the assessment of the real location. Some researchers have defined the SLAM as a chicken or egg problem, which one came first. Concerning the SLAM problem, the map or robot localization should come first. An unbiased map is needed for the localization; meanwhile an accurate pose estimate is needed to build that map as well. For the iteration of the measured distance, the direction travelled in has an effect to inherent imprecision and accumulates the sensor error. Thus, every feature that has been located in the map will contain corresponding errors. There are various techniques to compensate errors such as Kalman filters, particle filters and scan matching of range data. The SLAM is therefore defined as the problem of building a model leading to a new map or repetitively improving an existing map while localizing the robot within that map at the same time. In practice, the answers to the two characteristic questions cannot be delivered independently of each other. Before a robot can contribute to answer the question of what the environment looks like given a set of observations, it needs to know, e.g., when a robot starts to scan features; it is capable of estimation the distance between the feature and itself as well as the direction of the feature. The distance can be estimated in a number of ways. If a feature is of known physical dimensions, the perceived size of the feature can indicate the distance. Another method of estimating the depth is to use multiple cameras and to employ stereo vision techniques to triangulate the distance. The distance estimation, like the feature tracking, is a nontrivial problem, which resides outside the scope of this project. In a simple case, the pose is a vector containing the x and y coordinates of the robot along with the orientation. In reality, the pose of the vehicle can be more complicated.

Furthermore, because we wanted to perform the visual odometry in city streets, flat terrains as well as on motorways where buildings or the 3D structure are not always present, we chose to estimate the motion of the vehicle by tracking the ground plane.

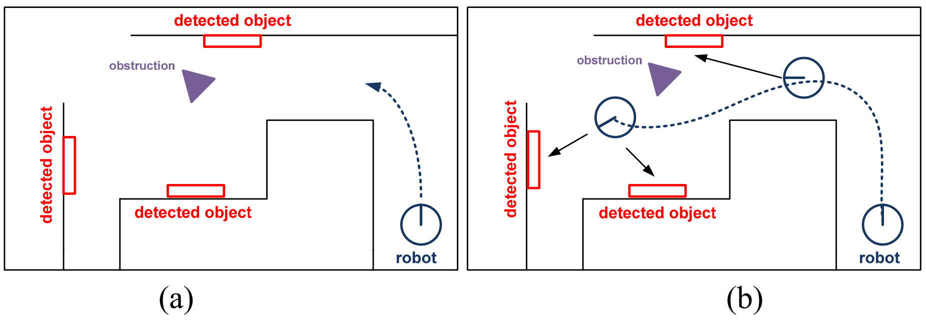
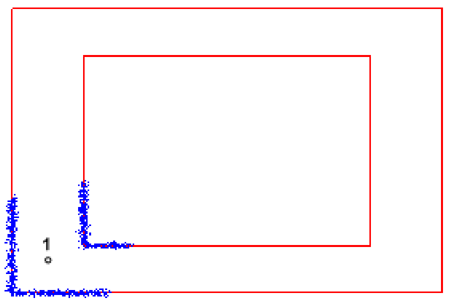


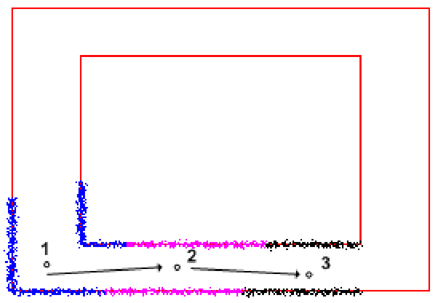
Figure 5.3 Autonomous robot tasks

Figure 5.3 expresses the autonomous robot tasks. The robot is standing in a corridor with the interested objects. The detected objects are, for example, chairs, tables, artificial landmarks and human beings. The robot starts from the starting point, and moves autonomously along the doorway. If the environment has obstructing objects, how does the mobile robot decides whether the object is the obstruction or the interested object? In the experiments, the mobile robot considers the obstruction as first priority. Otherwise, the mobile robot probably cashes into obstruction, and damages itself. Later, the robot decides what the interesting objects are, and then the robot must react early enough to calculate all objects in the environment. In the following section, the navigation technique which is integrated into the object detection algorithm will be proposed. The navigation technique predicts the future motions, avoids the obstruction, and detects the interesting objects.

5.2 Problem definitions

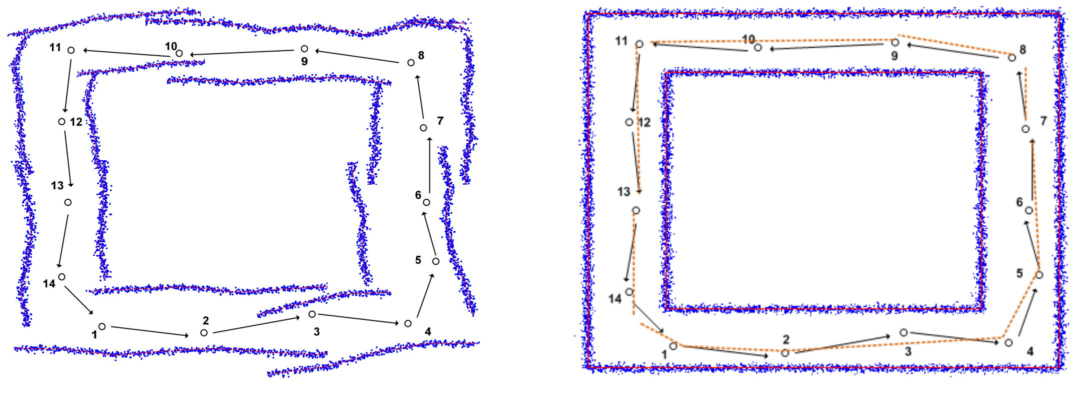
One general problem of mobile robot exploration in an unknown environment is to build the world model or mapping by using the range data. The out coming range data could use the reference distance from the robot itself to the surrounding environment in order to build the model. The easiest way to build the world model is to correct every moving step. The presented data is combined with the previous data continuously, thus the world model can be easily aligned. By merging many such scans taken at different locations, a more complete description of the world can be obtained. In order to integrate all sensor data, it is essential to find the replacing aligning data formulation. Moreover, the error of each scanning data is accumulated in the raw data that trends the data to the wrong direction. The spatial relationships are derived from the matching pair wise scan data. When each frame of sensor data is obtained, it is aligned to a previous frame to a cumulative global model. Figure 5.4 (a) shows the one scanning data from the range sensor in the robot, e.g., the laser range scanner, the sonar, the PMD and stereo camera. Figure 5.4 (b) shows the multi scanning data. The figure shows the ideal scanning data being received because each scanning data can be aligned continuously. The major problem of this approach is that the resulting world model may eventually become inconsistent as different parts of the model are updated independently.





(a) (b)
Figure 5.4 (a) One scanning data from range sensor
(b) Multi scanning data from range sensor

For the simulation example in Figure 5.5, it is assumed that the robot starts at position 1, 2,…,n to the destination in the close loop area. The robot uses 14 scanning times along the way. But ideally Figure 5.5 (a) can occur, if we have a very accurate sensor, and there is no slip between the ground and the wheel, as well as no obstacles in the way. If obstacles are laid in the way, basically the mobile robot has to avoid them because the robot cannot align the trajectory. The output then comes out as shown in Figure 5.5 (b), this figure shows the target alignment based on a network of relative pose constraints. The original posts are represented in the solid line and the dash line shows the new suitable trajectory that will provide the new pose constraints. The world model adjusts the new trajectory line. Spatial relations between local frames are derived from matching pairs of scanning measurement.



(a) (b)

Figure 5.5 An example of consistently aligning a set of simulated scans.
(a) The original scans badly misaligned due to accumulated pose error
(b) Result of aligning these scans based on a network of relative pose constraints

The alignment methods have two estimates, i.e. local and global. Local estimation is the fundamental estimate between the neighbors pairwise. It eliminates the fundamental error and registers the scan pair which constrains the limited range of scanning. The global estimation is a network of relative pose estimation, which calculates the overall relation from every pose data. However, the error may occur from a slip between the floor and robot wheel. For example, the processor commands the robot wheel meanwhile the encoder reads the feedback data for counting the distance of the robot. In practice, the robot should operate in the close loop area, but if the floor is very slippery, the encoder data, which is sent to the processor, is wrong and that affects the trajectory like a zigzag wave, and this is unacceptable. This is unacceptable. In Figure 5.5 (a), the constraints are indicated by the line connecting pair of poses. Figure 5.5 (b) generates the world model from using sensors only, which is usually inadequate for determining the relative scan poses because of the accumulating error.

5.2.1 Lu and Milios Global consistent alignment approach

The classical globally consistent range scan alignment was presented by Lu and Milios, 1997 [90]. Their method was very powerful to align the n-scanning matching for two dimension data sets. The phenomenal aspect of the exploration is that the robot scans a horizontal locomotion from a single robot, yaw angle. The range scan represents a partial view of the environment. It is assumed that the robot travels along a path from position $\underline{x_1}, \underline{x_2}, \ldots, \underline{x_n}$. At each pose $\underline{x_i}$, the robot captures the surrounding environment. Matching two scanning neighbor data at different poses is essential.

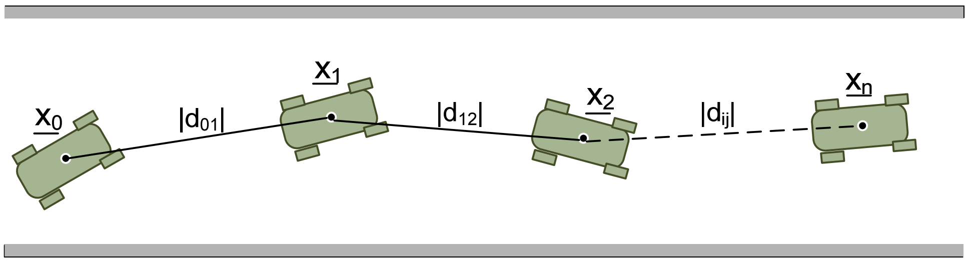


Figure 5.6 Robot moving steps

Figure 5.6 considers each robot move from $\underline{x_n}$ to $\underline{x_{n+1}}$ and the spatial distance $\left|\underline{d_{ij}}\right| = \left|\underline{x_i} - \underline{x_j}\right|$. The optimal estimation of all poses to build a consistent map of the environment can be calculated.

$$\left|\underline{d_{ij}}\right| = \left|\underline{x_i} - \underline{x_j}\right| \tag{5.1}$$

The observation $\underline{d_{ij}}$ could be modeled as $\underline{d'_{ij}} = \underline{d_{ij}} + \underline{\Delta d_{ij}}$ where $\underline{\Delta d_{ij}}$ is a random Gaussian error distribution, which mean is zero, and the covariance matrix $\underline{C_{ij}}$ is assumed to be known. Assume that all observation errors are Gaussian and independently related. To minimize estimation in the most optimal way, the observation is $\underline{d'_{ij}}$ by using the Mahalanobis distance.

$$\underline{w_{ij}} = \sum_{(i,j)}^{n} \left(\underline{d_{ij}} - \underline{d'_{ij}}\right)^T \underline{C_{ij}^{-1}} \left(\underline{d_{ij}} - \underline{d'_{ij}}\right) \tag{5.2}$$

Consider the simple linear term of the estimation problem. Without any losses of the network connection, $\underline{x_i} - \underline{x_j}$ that is assumed to be connected in every pairwise. If the $\underline{d_{ij}}$ is out of range, $\underline{C_{ij}}$ is assumed zero. Eq (5.2) can be represented as

$$\underline{w_{ij}} = \sum_{(0\le i\le j\le n)}^{n} \left(\underline{x_i} - \underline{x_j} - \underline{d'_{ij}}\right)^T \underline{C_{ij}^{-1}} \left(\underline{x_i} - \underline{x_j} - \underline{d'_{ij}}\right) \tag{5.3}$$

Where $\underline{w_{ij}}$ is represented as a function of all positions $\underline{x_n}$. Represent the measurement equation in a matrix form as

$$\underline{D} = \underline{HX} \tag{5.4}$$

Where $\underline{X}$ is the nd-dimensional matrices which is the concatenation of $\underline{x_1}, \underline{x_2}, \ldots, \underline{x_n}$. $\underline{D}$ is the concatenation of all position difference of $\underline{d_{ij}}$ and $\underline{H}$ is the incidence matrix with all entries being 1, -1 or 0.

$$\underline{W} = (\underline{D'} - \underline{HX})^T \underline{C^{-1}} (\underline{D'} - \underline{HX}) \tag{5.5}$$

Where $\underline{D'}$ is the concatenation of all the observations $\underline{d'_{ij}}$ and $\underline{C}$ is the covariance of $\underline{D'}$. Then the solution for $\underline{X}$ which minimizes $\underline{W}$ is given by

$$\underline{X} = \underbrace{\left(\underline{H^T C^{-1} H}\right)}_{\underline{G}}{}^{-1} \underbrace{\underline{H^T C^{-1} D'}}_{\underline{B}} \tag{5.6}$$

The covariance of $\underline{X}$ is

$$\underline{C_X} = (\underline{H^T C^{-1} H})^{-1} \tag{5.7}$$

Denote matrix $\underline{G} = \underline{H^T C^{-1} H}$ is defined, and $\underline{B} = \underline{H^T C^{-1} D'}$ simplifies the notation of the solution. $\underline{G}$ consists of

$$\begin{aligned} \underline{G_{ij}} &= \sum_{j=0}^{n} \underline{C_{i,j}^{-1}}, (i = j) \\ \underline{G_{ij}} &= -\underline{C_{i,j}^{-1}}, (i \neq j) \end{aligned} \tag{5.8}$$

The entries of $\underline{B}$ is obtained by

$$\underline{B_i} = \sum_{j=0, j \neq i}^{n} \underline{C_{i,j}^{-1} D'} \tag{5.9}$$

Solving the linear optimal estimation problem (5.9) is equivalent to solve the following linear equation system

$$\underline{X} = \underline{C_x B} \tag{5.10}$$

After solving $\underline{X}$, using $\underline{X}$ is the solution to update the new pose and each covariance. By merging each time scan, the complete 3D mapping can be obtained. When each frame of the sensor data is obtained, it is aligned to the previous frame or to a cumulative global model. The major problem of this approach is that the result of the environment model may eventually become inconsistent as different parts of the model are updated independently. Moreover, it may be difficult to resolve such inconsistency, if the data frames have already been permanently integrated. To be able to resolve the inconsistency, it is necessary to maintain the local frames of data together with their estimated pose. Each local frame is defined as the collection of sensor data measured from a single robot pose. The idea of the approach is to maintain all the local frame data. Spatial relations between local frames are derived from matching pairs of scans or from odometry measurements. New sensor data are matched to the current model of individual object frames. If some objects, which have been discovered earlier is observed again, its object frame pose is updated. A scalar random variable represents the uncertainty of a three degree of freedom pose. When a previously recorded object is detected again, the system only attempts to update the pose along the path between the two instances of detecting this object.

5.2.2 Pose estimation

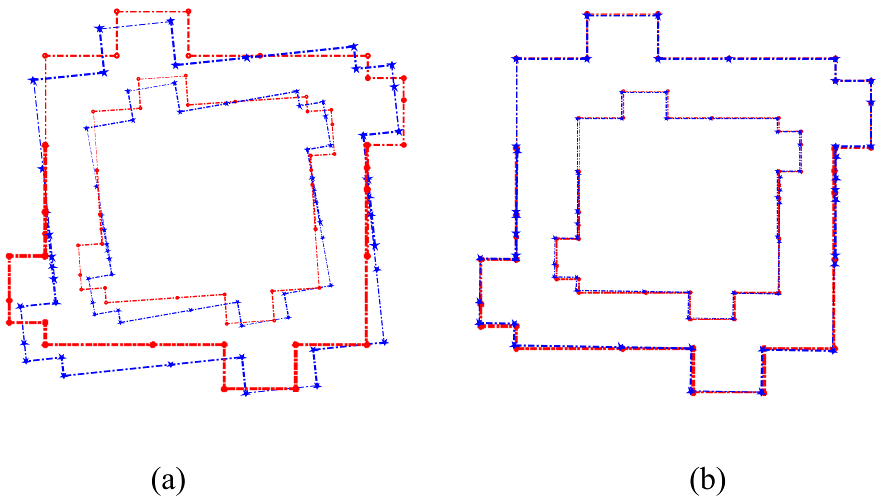

(a) (b)

Figure 5.7 The raw scanning data

For the scan pose estimation, the mobile robot moves and scans through the environment. The estimation of the motion has to be accurate. To understand the fundamental concept of algorithm, the simulation has to be implemented. This section shows the simulation output from Lu and Milios's approach. In the first simulation, it is assumed that the robot moves in a close area. The robot travels around a central object, and forms a loop enclosing area. The random sensor's data error is included in the supposing robot path. In Figure 5.7 (a) show the raw scanning data where the pose of each scan is obtained by the range data. According to the accumulation error of the sensor data, the shape of the map seems poor, and is not in alignment. Figure 5.7 (b) shows the result after correcting the pose errors and scanning data. The final output can obtain a good mapping and alignment.

5.3 Feature tracking and correspondence finding

In the first step for SLAM, it is assumed that the mobile robot moves from the first position to another. The correspondence points between two frames have to be found in order to merge those frames. The good feature tracking is a feature point extracted from an image. The Open Source Computer Vision Libraries (OpenCV) are a very famous collection of algorithms for image processing and computer vision. Some libraries are used in basic image analysis such as corner detection, canny edge detection and non-maxima suppression [91]. The good feature tracking library from OpenCV is used to find the correspondence points in image frames. In Figure 5.8, green points show the corresponding points between two image frames. In each step of generation mapping, the previous and next frames must have the same corresponding points; otherwise each step map cannot merge to the large map.

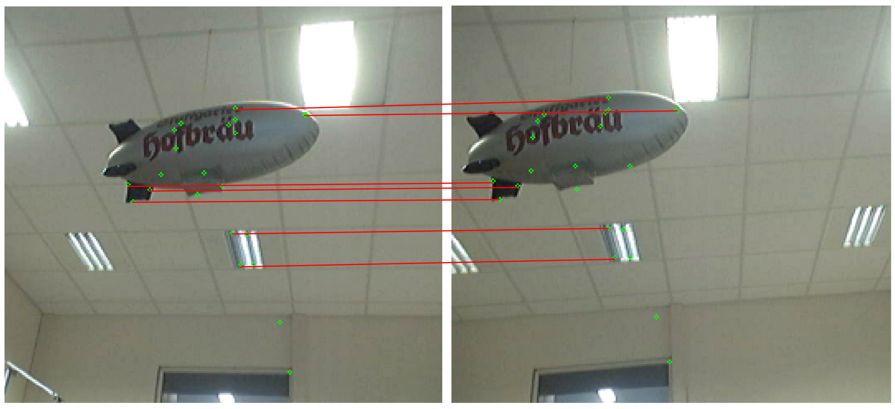


Figure 5.8 Corresponding points of two image frames

5.3.1 The optical flow

The unpredictable mobile robot locomotion for obstacle avoidance is a complex task for generating 3D mapping. In addition, posing the robot, for instance turning left, right, going forward, backward and climbing steep inclines is an arduous achievement of 3D mapping. In fact, the direction that should merge the frame is very important. The optical flow is hence perused to seek the pose estimation of a mobile robot. One of the famous optical flow techniques is the motion template. It was proposed by [92] [93]. They proposed effective methods to track the moving object using the different edge segmentation from the camera image frame to frame.

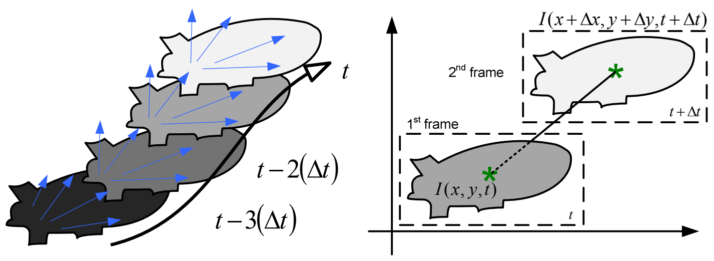


Figure 5.9 Optical flow motion

A model of zeppelin airship movement is demonstrated in Figure 5.9. The white indication is the current zeppelin position. Coming up of new silhouettes is captured in the next frame and overlaid on the white position. Each step of fading sequences of silhouettes is recorded, and referred to as the motion history image (MHI). The image input of the camera frame should sufficiently contain texture information in order to estimate the correspondence points between two frames. It is assumed that in Figure 5.9, the first frame is captured at time t and point(x, y). This point contains the color intensity, $I(x, y, t)$. When the zeppelin changes the position, a camera captures the second frame at the same time, but the same point is still kept in the second frame. The intensity in the present frame is the same as in the previous frame.

$$I_2(x + \Delta x, y + \Delta y, t + \Delta t) = I_1(x, y, t) \tag{5.11}$$

Where x, y are the coordinates on x-axis and y-axis, respectively. The Taylor series represents the (5.11) as a summation term.

$$\begin{aligned} &I_2(x + \Delta x, y + \Delta y, t + \Delta t) \\ &\qquad = I_1(x, y, t) + \frac{\partial I}{\partial x}\Delta x + \frac{\partial I}{\partial y}\Delta y + \frac{\partial I}{\partial t}\Delta t + \cdots \end{aligned} \tag{5.12}$$

It is assumed that the higher order terms are very small and can be ignored. Equation (5.12) is equal to equation (5.13).

$$I_2(x+\Delta x, y+\Delta y, t+\Delta t) = I_1(x,y,t) + \frac{\partial I}{\partial x}\Delta x + \frac{\partial I}{\partial y}\Delta y + \frac{\partial I}{\partial t}\Delta t \tag{5.13}$$

$$\frac{\partial I}{\partial x}\Delta x + \frac{\partial I}{\partial y}\Delta y + \frac{\partial I}{\partial t}\Delta t = 0$$
$$\frac{\partial I}{\partial x}\frac{\Delta x}{\Delta t} + \frac{\partial I}{\partial y}\frac{\Delta y}{\Delta t} + \frac{\partial I}{\partial t}\frac{\Delta t}{\Delta t} = 0 \tag{5.14}$$

$$\frac{\partial I}{\partial x}v_x + \frac{\partial I}{\partial y}v_y + \frac{\partial I}{\partial t} = 0 \tag{5.15}$$

Where $v_x = \frac{\Delta x}{\Delta t}$ and $v_y = \frac{\Delta y}{\Delta t}$ are the image velocity or optical flow at pixel (x, y). The $\frac{\partial I}{\partial x}, \frac{\partial I}{\partial y}$ and $\frac{\partial I}{\partial t}$ are the point intensity gradients. This equation can be rearranged concisely.

$$I_x = \frac{\partial I}{\partial x}, I_y = \frac{\partial I}{\partial y}, I_t = \frac{\partial I}{\partial t} \tag{5.16}$$

$$I_x v_x + I_y v_y = -I_t \tag{5.17}$$

$\underline{v}$ is the image velocity at pixel (x, y). Figure 5.10 shows the optical flow output. The camera moves from the right to the left side. The arrow length expresses the direction of mobile robot locomotion. The directions of optical flow are used to find spatial distance for merging the image frames in order to generate SLAM in real time.

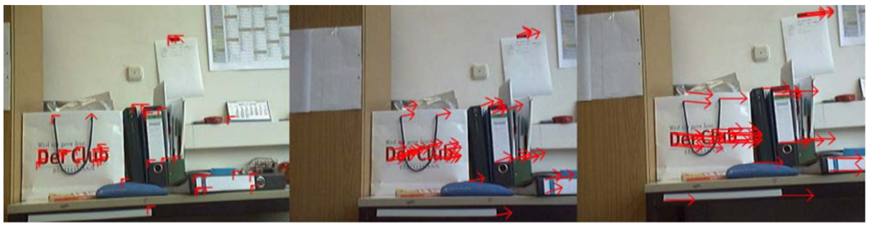
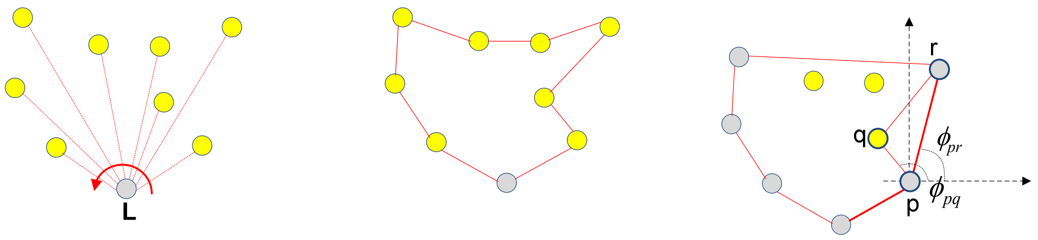

position 1 position 2 position 3

Figure 5.10 Optical flow output

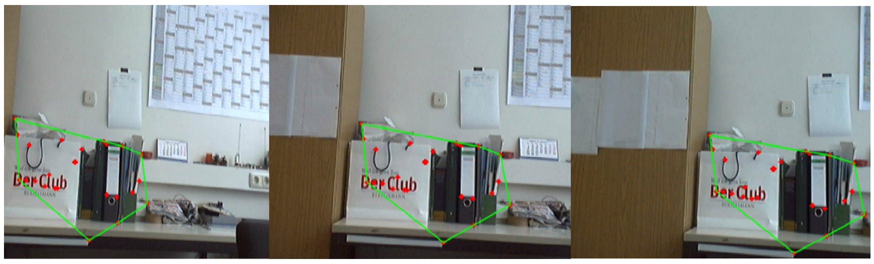
5.3.2 Convex set

After realizing the direction of the robot, the next issue is how to find the overlapping of image frames. It can be noticed from Figure 5.10 that the camera moves from the previous position (1^{st}) to the next position (2^{nd}) and the last position(3^{rd}). Corresponding points still appear on every frame. The frame is used to merge and create the map frame. The frame is important to calculate and keep it before the robot moved out of range. To determine the map frame, the fixed boundary of corresponding points is important to be found out. The Quick hull, Graham's scan, Jarvis March and Convex hull are very famous for determining the smallest interested convex of clown points. Figure 5.11 demonstrates the steps to find the convex hull. *Step1:* finding the lowest point (L) within all set points. *Step2:* sorting the set points in counterclockwise direction and network for each point. *Step3:* sorting by calculating the relative angle. The orientations of three points p, q and r are the ordering points in the network. Angles $\emptyset_{pq}$ and $\emptyset_{pr}$ in counterclockwise are determined. The smaller angle is a convex vertex. In the third step, it can be seen that the three nonvertex points are out of boundary. Figure 5.12 shows the boundary of the corresponding area when the camera changes its positions. The green area is the overlapping area, which has to keep and register this area by using the matching algorithm, which will be presented in the next section.



step 1st step 2nd step 3rd

Figure 5.11 Convex hull computation



position 1st position 2nd position 3rd

Figure 5.12 Corresponding area

5.4 Interactive Closest Point (ICP) matching algorithm

The ICP (Iterative Closest Point) algorithm is a method to search the closest of three dimensional points when a relative pose estimate is available. The basic concept is to capture the closest corresponding points by the calculation transformation and rotation matrix. In general, the iterative closest point (ICP) is widely used for the registration of 3D clouds. Umeyama et al. [94] proposed to find the similarity transformation parameters in m-dimensional space that gives the least mean squared error between these point patterns. Wen [95] presented to solve the special absolute orientation problem, searching a reduced gradient algorithm together with its convergence proof and generalizing it to the case with weighted errors, based on the classic absolute orientation technique. Surmann et al. [31] [30] justified an applied ICP algorithm to generate mapping based on locally consistent 3D laser range scans. The results were excellent for solving the SLAM problem.

5.5 Algorithm overview

The ICP algorithm was proposed by Besl and McKay [96]. They used to register two given point sets in a common coordinate system. The output can provide the parameter of the rotation and translation matrix. In each interaction step, their algorithm selected the closest points as correspondences, and calculated the transformation. The basic method to compute the close between two points on any geometrics view is to find the Euclidean distance $|\underline{d}|$ between two points $\underline{p} = (x_p, y_p, z_p)$ and $\underline{q} = (x_q, y_q, z_q)$. $|\underline{q} - \underline{p}| = \sqrt{(x_q - x_p)^2 + (y_q - y_p)^2 + (z_q - z_p)^2}$. The closest point method mentions in $\underline{n}$ dimension form. The quaternion based algorithm is normally expressed over the single value decomposition, SVD. The SVD is based on the cross covariance matrix of two cloud point distributions. The multiple scans registration and pose relation are an estimated spatial relation between the two poses, which can be derived from matching two range scans. We estimate all the poses by solving an optimization problem. Since a robot pose defines the local coordinate system of a scan is used, the pose relation between scans can be directly obtained from the odometry, which measures the relative movement of the robot. More accurate relations between scan poses are derived from aligning pairwise scans of points. After aligning two scans, it can record a set of corresponding points on the two scans. This correspondence set will form a constraint between the two poses. The amount of overlap between two scans is estimated empirically from the spatial extent of the matching parts between the two scans. The pose is defined as a three dimension vector $(x, y, \theta)^t$ consisting of 2D robot position. The two types of link between two pairs of nodes are defined. *Firstly*, two poses are very close or on sufficient overlap data. It is called *a weak cloud* pairwise data between the two nodes. *Secondly*, the two pose data is not too close, and has sufficient overlap information. It is called *a strong cloud* pairwise data. For each strong cloud pairwise data, a constraint on the relative pose is determined by the set of corresponding points on the two scans. It is assumed that the robot starts on the pose P_{ref}, and takes one scan S_{ref}. Afterward, the robot moves to the new pose P_{new} and takes another scan S_{new}. The approximation between two poses is implemented easily from the odometry information. Unfortunately, the providing information is sometimes incorrect due to wheel slippage between the robot's wheels and surface environment. The relative optimization problems of the P_{ref} and P_{new} have to be solved. The scanning crowded data represents the contour shape of the passing robot environment. Due to the existence of accumulated sensor noise and slippage of the wheel, it may sometimes happen that two scan data cannot align suitably. Generally, the adopted scan matching data is used to find the best alignment of the overlapping part based on the minimum error. We can estimate the matching criterion as the minimization of a distance between the two scans, as the function of the rotation and the translation.

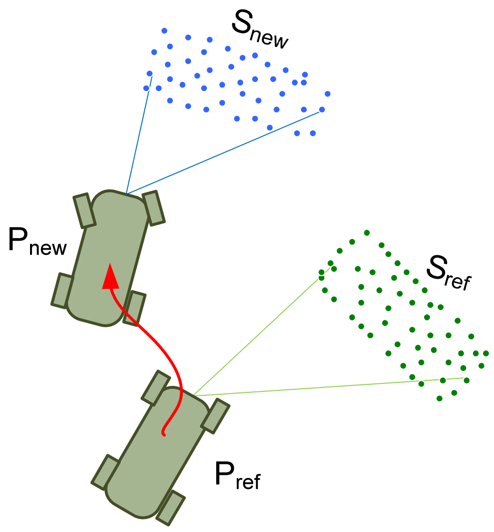


Figure 5.13 Robot movement and scan data

[97] concluded the least-squares rigid motion using SVD which assumed point set $\underline{p}$=($p_i, \dots, p_n$) and $\underline{q}$=($q_i, \dots, q_n$) are interaction cloud points between two robot positions, $\forall i \in 1, \dots, N$. Searching rotation $\underline{R}, \underline{\Delta\theta}$ and translation $\underline{t}, \underline{\Delta t}$ between the two data frames, reference and corresponding clown points are the main problems. The aim is to properly estimate the next position, getting an ideal minimum error.

$$\underline{e} = \min_{\boldsymbol{R},\boldsymbol{t}} \sum_{i=1}^{n} \underline{\omega_i} \left\| \underline{p_i} - (\underline{R}\underline{q_i} + \underline{t}) \right\|^2 \tag{5.18}$$

Where $\underline{\omega_i} > \underline{0}$ is the weight for each point pair.

$$\underline{p} - \underline{e} = \underline{q} \tag{5.19}$$

$$\underline{e} = \frac{1}{n}\sum_{i=1}^{n} \left\| \underline{p_i} - \left(\underline{R}\underline{q_i} + \underline{t}\right) \right\|^2 \qquad (5.20)$$

Where $\underline{e} = (e_1, \dots, e_n)$ is an error vector. Equation (5.20) is an observation equation, which is achieved by least square minimization and $(\underline{\mathrm{R}}, \underline{\mathrm{t}})$ represents the optimum rotation and translation.

5.5.1 Translation calculation

First step, calculation the centroid of the cloud points, it is assumed that the $\underline{\overline{p}}$ and $\underline{\overline{q}}$ are the centroid of two cloud points.

$$\underline{\overline{p}} = \tfrac{1}{n}\textstyle\sum_{i=1}^{n}\underline{p_i} \quad , \quad \underline{\overline{q}} = \tfrac{1}{n}\textstyle\sum_{i=1}^{n}\underline{q_i} \qquad (5.21)$$

The movement of all points to the artificial center is accomplished by subtracting with each centroid.

$$\begin{aligned} \underline{p_i'} &= \underline{p_i} - \underline{\bar{p}} \\ \underline{q_i'} &= \underline{q_i} - \underline{\bar{q}} \end{aligned} \qquad (5.22)$$

The summation of each point in centroid is zero

$$\textstyle\sum_{i=1}^{n}\underline{p_i'} = \underline{0} \ \text{ and } \ \textstyle\sum_{i=1}^{n}\underline{q_i'} = \underline{0} \qquad (5.23)$$

The error between two cloud points is

$$\underline{e} = \underline{p_i'} - \underline{R}\underline{q_i'} - \underline{t'} \qquad (5.24)$$

The summation of the absolute error is

$$\underline{e} = \sum_{i=1}^{n} \left\| \left[\underline{p_i'} - \underline{R}\underline{q_i'}\right] - \underline{\acute{t}} \right\|^2 \qquad (5.25)$$

From Eq.(5.25) we can rearrange the new equation, and minimize an error in order to find the $\underline{R}$**.** In order to minimize the error, $\underline{e}$. The middle term is zero because summation of $\sum_{i=1}^{n} \underline{p_i^{'}} = \underline{0}$ and $\sum_{i=1}^{n} \underline{q_i^{'}} = \underline{0}$ are zero. Therefore, there are only the first and the third term left. The first term does not depend on the parameter $\underline{\acute{t}}$. The third term must be not less than zero. The total error, which then leaves only the third term must be greater than zero. A special solution can be.

$$\begin{gathered} 0 \leq \underline{t'} - \underline{p_i'} + \underline{R}\underline{q_i'} \\ \underline{t'} \approx \underline{p_i'} - \underline{R}\underline{q_i'} \end{gathered} \tag{5.26}$$

The translation between two cloud points then depends on each centroid. We denote in terms of simplified equation referring to the artificial centroid.

$$\underline{e} = \underline{p_i'} - \underline{R}\underline{q_i'} \tag{5.27}$$

If $\underline{\acute{t}} = \underline{0}$, the total error to be minimized is just

$$\sum_{i=1}^{n} \left\| \underline{p_i'} - \underline{R}\underline{q_i'} \right\|^2 \tag{5.28}$$

5.5.2 Rotation calculation

From equation (5.28), we can minimize the error. Where $\|x\| = \sqrt{\underline{x^T x}}$. Thus

$$\begin{aligned} &\left\| \underline{p_i^{'}} - \underline{R}\underline{q_i^{'}} \right\|^2 = \left(\underline{p_i^{'}} - \underline{R}\underline{q_i^{'}} \right)^T \left(\underline{p_i^{'}} - \underline{R}\underline{q_i^{'}} \right) \\ &= \left[\underline{p_i^{'T}} - \underline{q_i^{'T}}\underline{R^T} \right] \left[\underline{p_i^{'}} - \underline{R}\underline{q_i^{'}} \right] \\ &= \underline{p_i^{'T} p_i^{'}} - \underline{q_i^{'T}}\underline{R^T}\underline{p_i^{'}} - \underline{p_i^{'T}}\underline{R}\underline{q_i^{'}} + \underline{q_i^{'T}} \underbrace{\underline{R^T R}}_{\underline{I}} \underline{q_i^{'}} \\ &= \underline{p_i^{'T} p_i^{'}} - \underline{q_i^{'T}}\underline{R^T}\underline{p_i^{'}} - \underline{p_i^{'T}}\underline{R}\underline{q_i^{'}} + \underline{q_i^{'T} q_i^{'}} \end{aligned} \tag{5.29}$$

Where the rotation matrix implies $\underline{R^T R} = \underline{I}$, ($\underline{I}$ is the identity matrix). Which $\underline{q_i^{'T}}\underline{R^T}\underline{p_i^{'}}$ is a scalar, for any scalar $a = a^T$, therefore

$$\underline{q_i'}^T \underline{R}^T \underline{p_i'} = (\underline{q_i'}^T \underline{R}^T \underline{p_i'})^T = \underline{p_i'}^T \underline{R} \underline{q_i'} \tag{5.30}$$

Equation (5.30) is substituted into equation (5.29)

$$\left\| \underline{p_i'} - \underline{R}\underline{q_i'} \right\|^2 = \underline{p_i'^T p_i'} - 2\underline{p_i'}^T \underline{R} \underline{q_i'} + \underline{q_i'^T q_i'} \tag{5.31}$$

Equation (5.31) is minimized in order to find the rotation term

$$\begin{aligned}
\min_{\underline{R}} \sum_{i=1}^{n} \underline{\omega_i} \left\| \underline{p_i'} - \underline{R}\underline{q_i'} \right\|^2 \\
&= \min_{\underline{R}} \sum_{i=1}^{n} \underline{\omega_i} (\underline{p_i'^T p_i'} - 2\underline{p_i'}^T \underline{R}\underline{q_i'} + \underline{q_i'^T q_i'}) \\
= \min_{\underline{R}} \sum_{i=1}^{n} \underline{\omega_i} \underbrace{(\underline{p_i'^T p_i'})}_{\underline{0}} - \min_{\underline{R}} \sum_{i=1}^{n} \underline{\omega_i} \left(2\underline{p_i'}^T \underline{R}\underline{q_i'}\right) \\
&+ \min_{\underline{R}} \sum_{i=1}^{n} \underline{\omega_i} \underbrace{\left(\underline{q_i'^T q_i'}\right)}_{\underline{0}}
\end{aligned} \tag{5.32}$$

$$= \min_{\underline{R}} \textstyle\sum_{i=1}^{n} (-2\underline{\omega_i} \underline{p_i^T} \underline{R}\underline{q_i'})$$

Where $\sum_{i=1}^{n}(\underline{p_i'^T p_i'})$ and $\sum_{i=1}^{n}(\underline{q_i'^T q_i'})$ are the equation zero because these terms do not depend on $\underline{R}$. Where $\underline{W} = diag(\omega_1, \dots, \omega_n)$ is a $n \times n$ diagonal matrix. $Trace(\underline{A}) = a_{11} + a_{22} + \cdots + a_{nn} = \sum_{i=1}^{n} a_{ii}$.

$$\sum_{i=1}^{n} (\underline{\omega_i}\, \underline{p_i'}^T \underline{R}\underline{q_i'}) = Trace(\underline{W}\underline{p}^T \underline{R}\underline{q}) \tag{5.33}$$

Where $Trace(\underline{AB}) = Trace(\underline{BA})$, therefore

$$\begin{aligned} Trace\left(\underline{WP^T}\underline{RQ}\right) &= Trace\left(\left(\underline{Wp^T}\right)\left(\underline{Rq}\right)\right) \\ &= Trace((\underline{Rq})(\underline{Wp^T})) \end{aligned} \tag{5.34}$$

From the single value decomposition theory, it is assumed that $\underline{S} = \left[\underline{R}(\underline{q}\underline{W}\underline{p^T})\right] = \underline{R}(\underline{U\Sigma V^T})$. The substitution is in equation (5.34).

$$Trace\left(\underline{R}\underline{q}\underline{W}\underline{p^T}\right) = Trace\left(\underline{RU\Sigma V^T}\right) = Trace(\underline{\Sigma V^T RU}) \tag{5.35}$$

A $\underline{\Sigma}$ is a diagonal matrix. All elements is in the diagonal and cannot be less than zero $(\sigma_1, \sigma_2, \dots, \sigma_n \geq 0)$. Thus

$$\begin{aligned} &Trace\left(\underline{\Sigma M}\right) \\ &= Trace\left(\begin{bmatrix} \sigma_1 & 0 & \cdots & 0 \\ 0 & \sigma_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \sigma_n \end{bmatrix}\begin{bmatrix} m_{11} & m_{12} & \cdots & m_{1n} \\ m_{21} & m_{22} & \cdots & m_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ m_{n1} & m_{n2} & \cdots & m_{nn} \end{bmatrix}\right) \\ &= \sum_{i=1}^{n} \underline{\sigma_i m_{ii}} \leq \sum_{i=1}^{n} \underline{\sigma_i} \end{aligned} \tag{5.36}$$

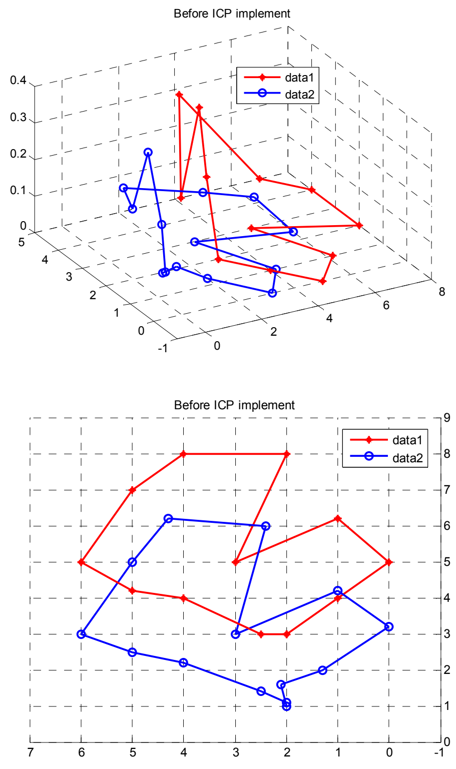
It can be notice that $\underline{V}, \underline{R}$ and $\underline{U}$ are orthogonal matrices. If $\underline{M} = \underline{V^T RU}$ is defined, then $\underline{M}$ also is an orthogonal matrix meaning that $\underline{M}$ is the identity matrix.

$$\begin{aligned} \underline{I} &= \underline{M} = \underline{V^T RU} \\ \underline{V} &= \underline{RU} \\ \underline{R} &= \underline{VU^T} \end{aligned} \tag{5.37}$$

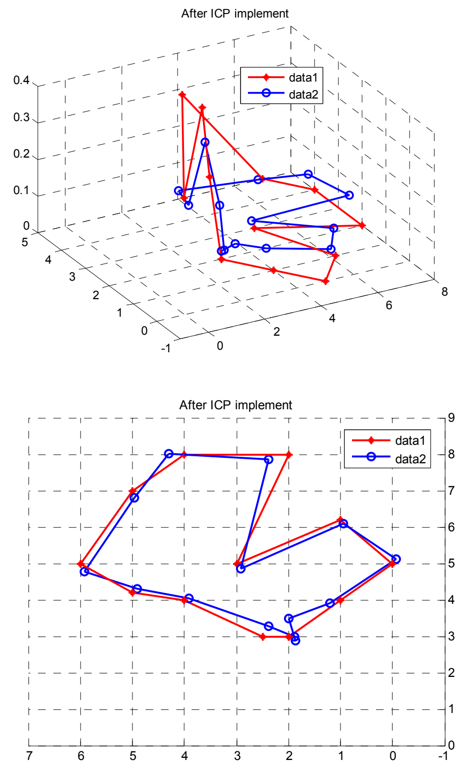
Thus, the rotation can be calculated from equation (5.37).

5.6 Simulation results

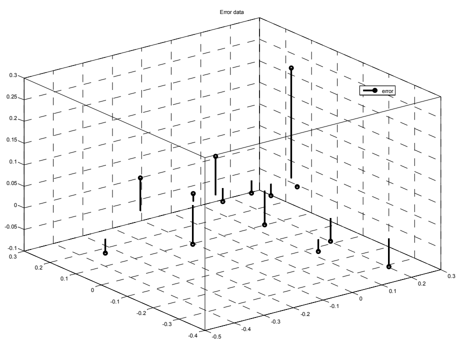
The matching approach is aimed at finding the minimum error of rotation and translation matrices between two scans and to search for an appropriate transformation matrix. In order to prove the previous suggested method, the simulation results using MATLAB are being discussed in this section. The simulation presents two crowning points with differential rotation and translation elements. Figure 5.14 (a) top, shows the principle of the simulation result. In this simulation, we want to prove the ICP technique that is proposed. The two data $(data_1, data_2)$ are assumed as shown below. Two data consist of twelve random elements in x, y and z axis, respectively. Figure 5.14 (a) below, illustrates the rotation of the field of view into two dimensions. We can clearly see that the two data are totally different translation and rotation. After applying the ICP algorithm, the rotation and translation matrices are acquired. The matrices are multiplied to $data_1$, then to the new $data_2$. Figure 5.14 (b) top, shows in the three dimension data that we can see the Euclidean distance to be minimized. Figure 5.14 (b) below, shows the rotation of the field of view into two dimensions. We can clearly see that the two data come close together. The rotation and translation are minimized. The rotation and translation matrices are also shown below. The rotation matrix is firstly multiplied to $data_1$, after that the translation is multiplied to $data_2$. Figure 5.14 (c) shows an error of each point. Steam plotting is an error value with the difference between $data_1$ and $data_2$. Moreover, the error data from each point is shown in the data below as well.



(a)



(b)



(c)

Figure 5.14 Simulation result (a) Before matching (b) After matching (c) Error between two frames

$$\underline{data1} = \begin{bmatrix} 1 & 1 & 2 & 2.2 & 3 & 5 & 6 & 6 & 3 & 4.2 & 3 & 2 & 1 & 1 \\ 2 & 2.5 & 4 & 5 & 6 & 5 & 4 & 2 & 3 & 1 & 0 & 1 & 2 & 2 \\ 0 & 0.2 & 0.3 & 0 & 0.2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

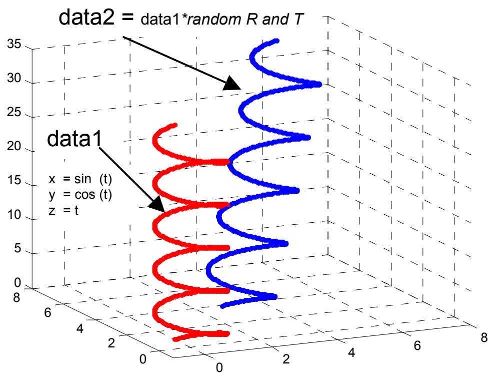
$$\underline{data2} = \begin{bmatrix} 1.1 & 1.4 & 2.2 & 2.5 & 3 & 5 & 6.2 & 6 & 3 & 4.2 & 3.2 & 2 & 1.6 & 1 \\ 2 & 2.5 & 4 & 5 & 6 & 5 & 4.3 & 2.4 & 3 & 1 & 0 & 1.3 & 2.1 & 2 \\ 0 & 0.1 & 0.2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

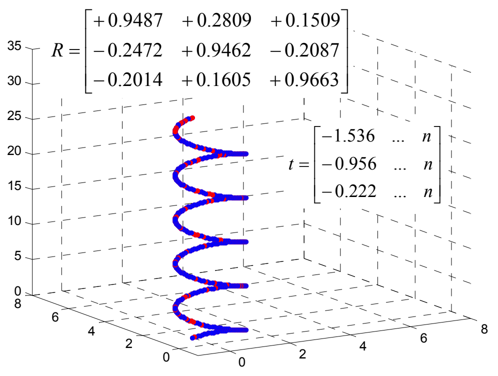
$$\underline{R} = \begin{bmatrix} 0.999679407929872 & -0.02352058301192 & -0.009373555129161 \\ 0.023707853016949 & 0.99951087031582 & 0.02039504414903 \\ 0.008889266916162 & -0.020610732526854 & 0.999748057581709 \end{bmatrix}$$

$$\underline{t_0} = \begin{bmatrix} 1.926531041153385 \\ -0.149405483316221 \\ 0.062140637535334 \end{bmatrix}$$

$$\underline{error} = \begin{bmatrix} 0.020 & -0.266 & -0.029 & -0.108 & 0.215 & 0.192 & -0.023 & 0.131 & 0.145 & 0.098 & -0.125 & 0.104 & -0.476 & 0.120 \\ 0.1274 & 0.115 & 0.094 & 0.092 & 0.081 & 0.033 & -0.295 & -0.391 & 0.079 & 0.050 & 0.073 & -0.197 & 0.012 & 0.126 \\ -0.030 & 0.077 & 0.090 & 0.018 & 0.254 & -0.003 & -0.028 & -0.066 & -0.027 & -0.078 & -0.090 & -0.053 & -0.033 & -0.029 \end{bmatrix}$$

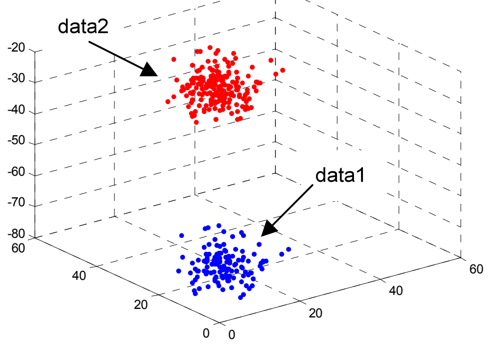
In Figure 5.15 (a), the equation $x = \sin(t), y = \cos(t)$ and $z = (t)$ are assumed as the first data set, where t is the various time. The $data_1$ is multiplied with the random rotation matrix $(\underline{R})$ and the center translation of the mass $(\underline{t_0})$. The ICP algorithm calculates the translation and rotation matrix. The registration data are shown in Figure 5.15 (b). Figure 5.16 (a) and (b) show the random data using a pseudo-random generated function. The simulation results prove that the translation and rotation matrix are found and matched with two different corresponding points. The translation and rotation data matrices are also shown. The simulation results give the appropriate output, which can be applied in the next step of experiments. Figure 5.16 shows the spatial distance enhancement before and after the correction of the data. Figure 5.17 the blue line, shows the spatial distance between $data_1$ and $data_2$. The spatial distances are about 28 units far. The red line represents the spatial distance after the correction. The average is nearly zero that means the simulation is going towards a suitable result.

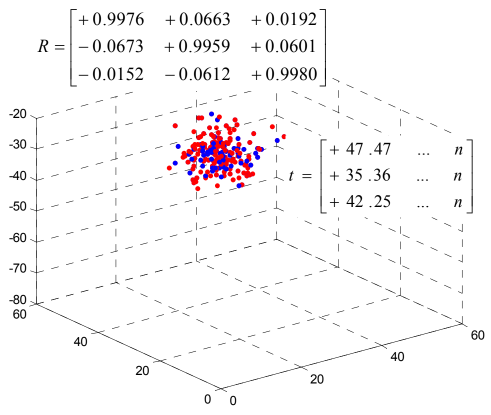




(a) Before matching (b) After matching

Figure 5.15 Simulation results1





(a) Before matching (b) After matching

Figure 5.16 Simulation results2

The next section is tested from the real PMD data. One PMD data frame is taken (red points), and another frame is taken (blue points) with the differential position. The simulation is implemented like the previous experiment. Figure 5.18 (a) top, shows the results from the PMD camera. The results are implemented before the ICP algorithm. Figure 5.18 (a) bottom, shows the rotation of the field of view in two dimensions. We can see that the two frame data are all different in three dimension positions.

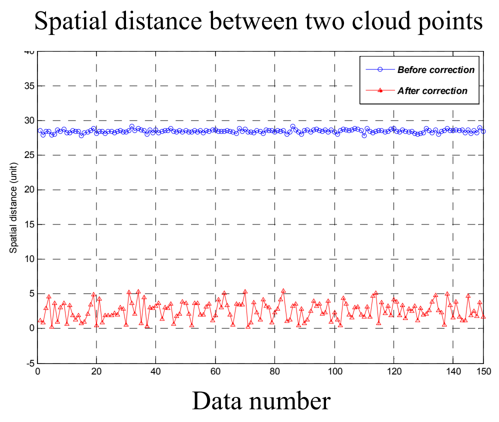


Figure 5.17 Spatial distance enhancement before and after correction

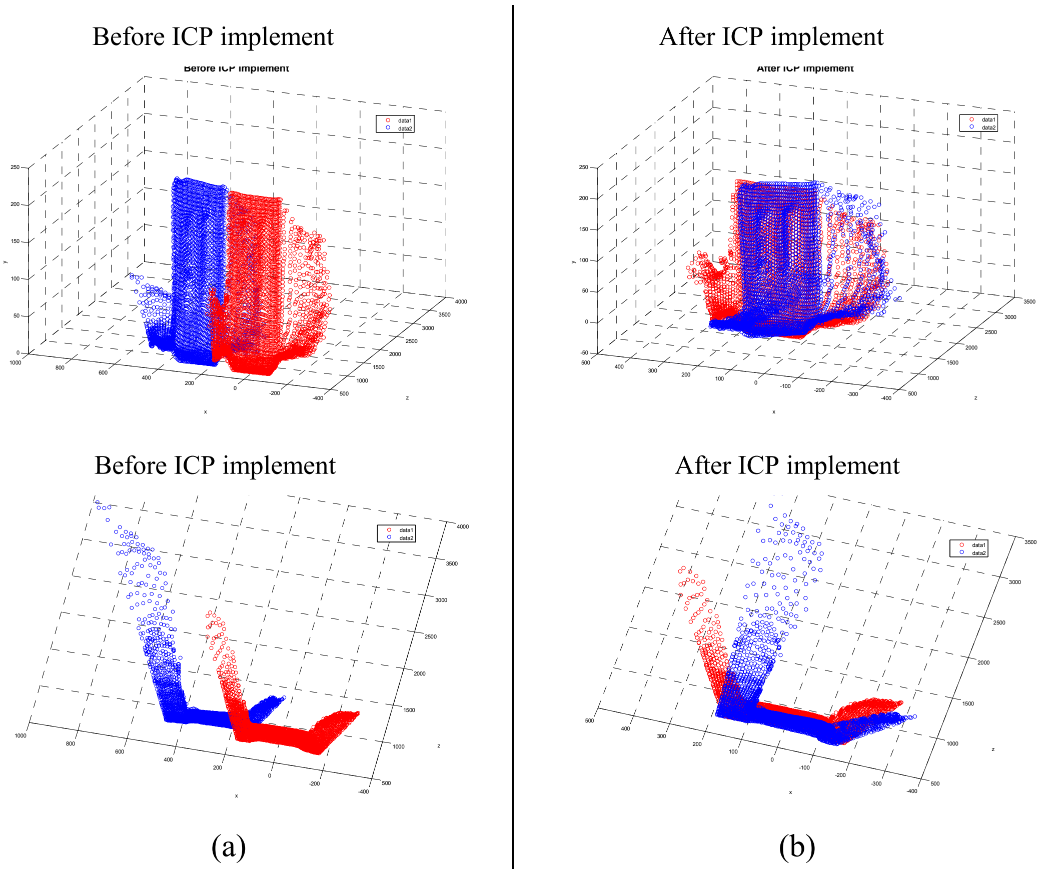


Figure 5.18 Result from PMD

$$\underline{R} = \begin{bmatrix} 0.988379315576389 & 0.151925487309443 & 0.0049974840208 13 \\ -0.151622806487188 & 0.983006013456833 & 0.103487690382259 \\ 0.01080986094724 & -0.103042825143209 & 0.99461817955073 \end{bmatrix}$$

$$\underline{t}_0 = \begin{bmatrix} -161.0640108074832 \\ -149.64520664621597 \\ 20.949632012753312 \end{bmatrix}$$

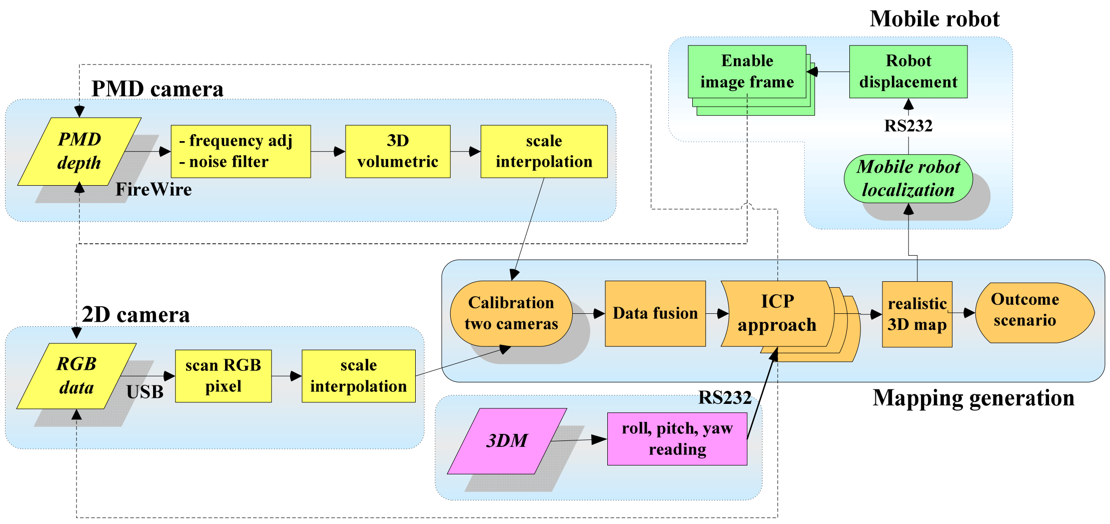


Figure 5.19 Diagram for 3D mapping

Figure 5.18 (b) shows the result after the implementation of ICP algorithm. Figure 5.18 (b) top, shows that the data can be registered between two frames. Figure 5.18 (b) bottom, shows the rotation of the field of view in two dimensions. We can see that the two frame data can be merged to very close points. The rotation and translation matrices are also shown in the matrices data below. Two matrices are relative vectors of each other. For this method, a real-time experiment will be used in the next chapter.

The overall process for generating 3D mapping is shown in Figure 5.19 that is separated into 4 blocks. *Firstly,* capture PMD depth data, the depth data is connected to a computer via firewire interface. The depth data has to eliminate the pepper-salt noise after getting the data. The enhanced data, which has the low noise, will calculate the three dimensional coordinates in the next step. *Secondly,* RGB data capturing, the color data is sent via the USB interface. Scale interpolation is handled. *Thirdly,* data fusion block, the depth and RGB data are fused using the calibration model. In this block, the ICP matching approach has been done and has generated the three dimensional map. This block is also the main command loop to send the trickle signal to the new capturing RGB/PMD data as well as control the mobile robot. Otherwise, if the mobile robot does not wait for the command from this block, the mobile robot will move randomly, and the mapping outcome cannot be predicted. The detail about the fusion is proposed in the previous section. *Fourthly,* the mobile robot localization block, the final block, is the mechanical control part. The synchronization of the image processing and mechanical parts has to be appropriately handled.

5.7 Conclusion

This chapter introduced and discussed SLAM techniques. The discussion showed that the SLAM technique is one of the most interesting mobile robot applications. The SLAM can locate itself while building the map in real time. This technique integrates various knowledge e.g., knowledge about control system, image processing, machine learning, sensor designs, data fusion, mechanical design as well as mathematical optimization. The main problems are to match the scanning data between each localization step. The alignment methods have to be estimated. The local estimation is the fundamental estimation between the neighbors pairwise. The global estimation is a network of relative pose estimation that will process the overall data again after the scanning is finished. The searching corresponding points between two frames are the first step to find the frame relative. Those corresponding points between two frames are used to find the intersection area. The intersection area is limited by a convex set technique. The convex set area evaluates the intersection area, because when the robot moves far away to the next frame, no intersection. The convex set will evaluate the next step frame. The optical flow is used to calculate the translation of the image frame. The optical flow uses a principle of comparing the pixel intensity. If the pixel intensity changes within the sampling period, the calculation of the image movement is feasible.

The aim of this research is to propose the novel 3D sensor, i.e., the PMD camera for mobile robot applications. It is expected to become an outstanding sensor within the next decade, especially for mobile robot 3D vision. The classical image register, ICP, is proposed in order to combine each robot step. The ICP simulation results are shown in order to clearly understand. However, the local registration does not suffice to yield 3D mapping reliability. The global registration then is also included in this research.

Chapter 6

Experimental Results

In the previous chapters, the concepts of object detection and SLAM are presented. To test the capability of those approaches, the real implementation is tested and presented in this chapter. The experiments are tested. Firstly, the motion estimation is tested by an evaluation between the stereo and PMD camera in order to understand the performance of the navigation system. Secondly, the object detection in 3D environment is presented. The 2D/3D camera combination approach from chapter 4 is used and the results are presented in this chapter. The 3D map buildings are presented in several environments, with and without 2D combination. The 2D/3D combination output shows the reliability improvement of the three dimension map buildings.

6.1 Motion estimation

The first part of this chapter introduces an improvement of the motion estimation solution by comparing the Photonic Mixer Device (PMD) and Stereo Camera. For the main feature of the stereo camera, the stereo camera has one sensor, which can detect and provide the depth data from the object to the camera by using the triangle principle of the focus lens between two cameras. The stereo camera provides the higher resolution of the 2D image compared to the one from the PMD camera, whereas the depth information derived from a PMD is usually far superior to the result from a stereo camera. This section proposes a combination scheme for the PMD and stereo camera in order to improve the results of the motion estimation. The combined setup is placed on a mobile robot and carried on the motion estimation task using a provided artificial landmark. Figure 6.1 shows the setting up method for the experiment. Both cameras are calibrated, and evaluate the position. It is assumed that the cameras point to the landmark. The next step is the feature extraction, template matching. The experiments are fulfilled using 1) stereo 2) PMD and 3) combination of stereo and PMD camera. The results of these three arrangements are compared and the outcome of the comparison is presented. Moreover, the output of object detection in 2D and 3D environment is presented. The expressions of the 3D mapping by only using the PMD camera and combination 2D and PMD camera are also proposed. The mapping experiments are tested in the closed room and long way corridor.

The robot moved along the circle path while the combined setup simultaneously captures the 3D images of the landmark from each new position, Figure 6.2. The motion is estimated based on the matching of the captured 3D images between two successive positions. The classical singular value decomposition (SVD) algorithm is used to solve the matching problem. The referent points for the SVD algorithm are extracted from the landmark using a robust corner detection algorithm [98] [99].

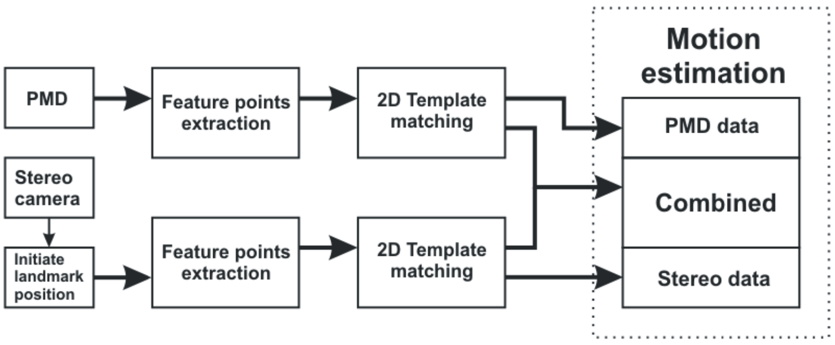
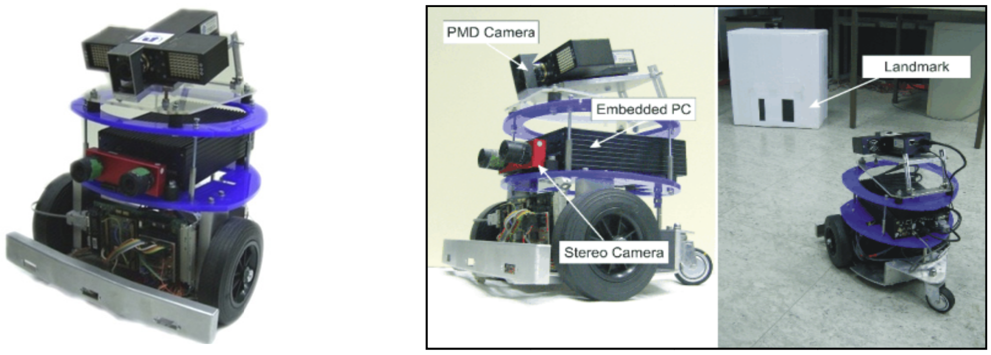


Figure 6.1 Setting up the experiment



(a) (b)
Figure 6.2 Motion estimation between stereo and PMD camera

During the experiment, the robot is moved to different positions within the allowed space while both camera systems are allowed to acquire images of the landmark at each new position. Figure 6.3 shows the images captured from PMD and stereo camera view. Figure 6.3 (a) illustrates the gray scale and depth data from the PMD camera.

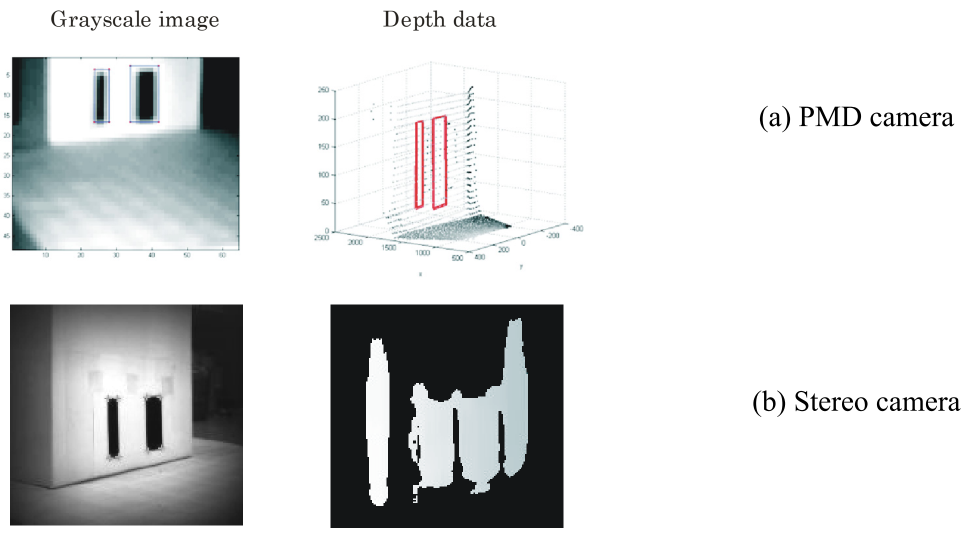


Figure 6.3 Estimate of the artificial landmark

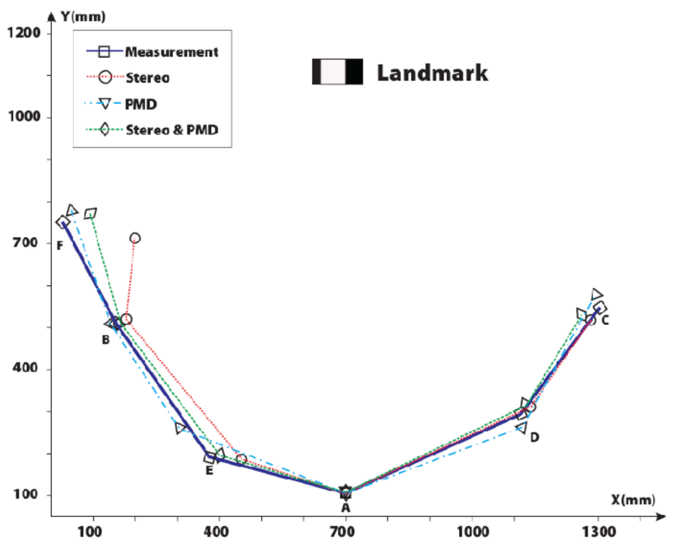


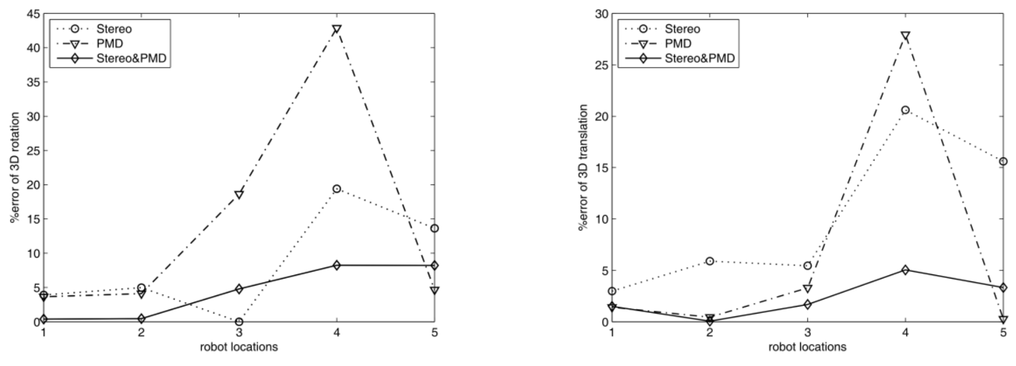
Figure 6.4 Result of trajectory curve movement

At the same position, Figure 6.3 (b) shows the gray scale and the depth data from the stereo camera. The experiment simulates a movement of a robot along a half circle around the landmark. The relationship between each rotation angle on the translation axes of the robot along this movement is illustrated by Figure 6.4. This movement provides a complete experiment result of 3D rotation and translation. The starting point of the experiment is on point A. Hence, the mobile robot moves to point B and C up to point F, respectively. Therefore the relative movement between A and

B, A and C up to F has five different positions. The motion between two successive positions is calculated by using the landmark as a reference. The hand measurement data that tells the true translation and rotation of the mobile robot is used as a reference value for evaluating the motion estimation result. The motion estimation and the measurement of the robot's movement from the stereo, the PMD camera and the combination of both cameras are illustrated in Figure 6.4.

An Analysis for stereo data: can be seen from the experimental result in Figure 6.5 (a) that the stereo camera seems to provide a slightly better performance due to its higher optical resolution, which helps for locating the exact position of the feature points on the landmark. However, Figure 6.5 (b) shows how the stereo camera has failed to defeat the PMD in terms of depth measurement due to its limitation of the stereoscopic design, which only delivers modest results compared to the excellent depth measurement when using the time of flight principle.

An Analysis for PMD data: The low resolution of the PMD camera does not allow the correct detection of the corners of the artificial landmark. A consequence is that the robot rotation has a higher error rate compared with the results from the stereo camera. However, the depth measurement performs better results for the translation.



(a) (b)

Figure 6.5 Percentage error of rotation and translation

This section compares the results of the motion estimation by using PMD and stereo camera systems. A test environment with an artificial landmark is constructed for the experiments. The gray scale images and the depth data from both cameras are used as the input. Three motion estimation results are obtained using only the PMD camera, stereo camera and the combination of both cameras. The results from the stereo, the PMD and the combination of stereo and PMD investigation show that the stereo and PMD camera provide results with almost comparable accuracy. The PMD camera can thoroughly measure 3D points with better depth accuracy than a stereo camera but the low resolution of gray scale image from the PMD camera (64×48 pixels) is not excellent enough to precisely locate the corners within the artificial landmark. The best results of motion estimation are obtained from using a combination of both PMD and stereo, where the advantages of both cameras are combined, that is, the precise corner detection from 2D high resolution stereo camera and the accurate depth data from the PMD camera. Each figure shows the distance in cm as well. A reliable tracking distance from the camera to the artificial landmarks ranges from 20cm up to 2.2 m. The limit of 2.2 m is because of the extreme wide-angle lenses mounted on the camera, so that the objects that are far away have an extremely small size in the projected image plane and could not be tracked.

6.2 Objects detection

6.2.1 2D object detection

The algorithm is tested in real world environment. The experiments of the artificial landmark detection are tested in many distances and rotational angles. From the experiments presented, the artificial landmarks could be detected even in a cluttered environment. Figure 6.6 shows the output of artificial landmark detection. Moreover, if we know the dimension of artificial landmarks we can easily calculate the distance from the landmarks to the camera using the normal rectangular ratio theory. Figure 6.7 shows the measurement distance that can be measured by calculating the area of the detecting object.

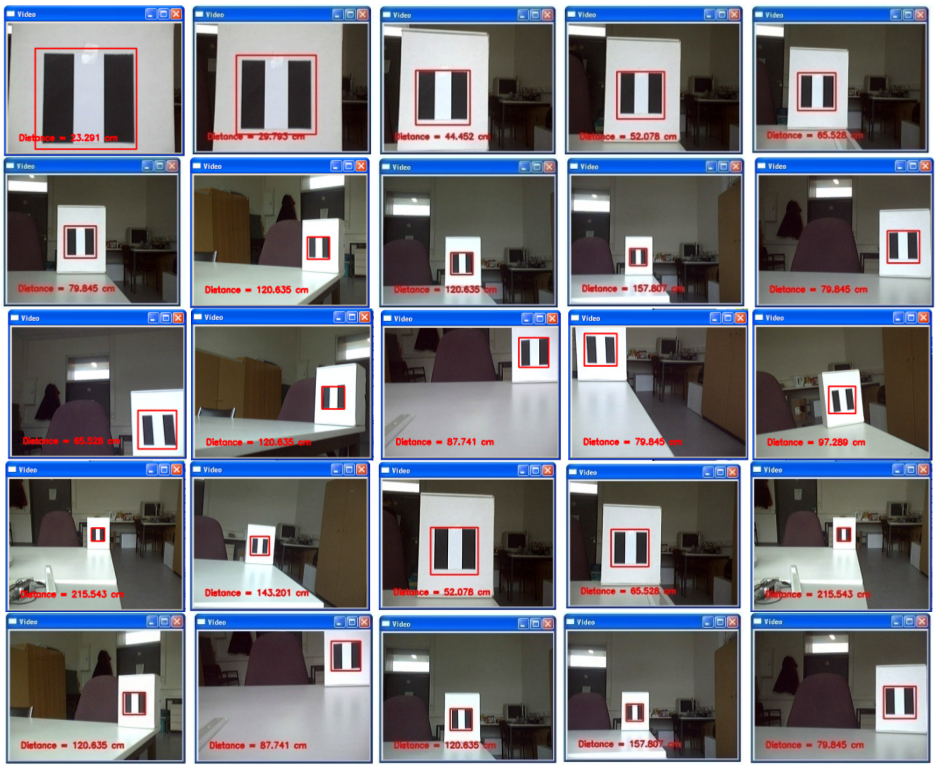


Figure 6.6 Artificial land mark detection output from 2D data

No.	real measurement (mm)	object detection (mm)	error (%)
1	20	23.21	16.05
2	30	29.79	0.70
3	45	44.45	1.22
4	50	52.07	4.14
5	80	79.84	0.20
6	100	97.28	2.72
7	120	120.63	0.50
8	140	143.20	2.28
9	160	157.81	1.36
10	215	215.54	0.25

Table 6.1 Different distance between real measurement and object detection

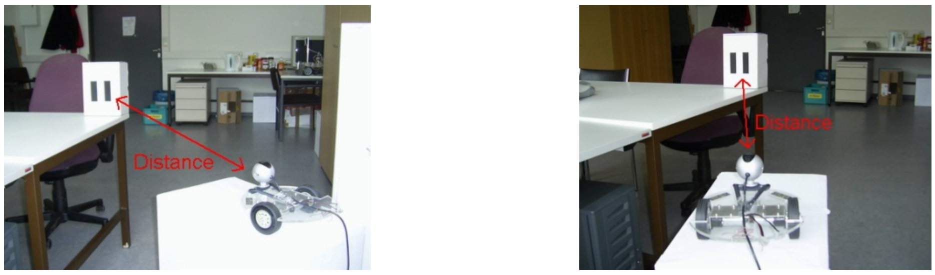


Figure 6.7 Distance measurement

6.2.2 3D object detection

The next experiment is designed to detect objection in 3D environments. In chapter 4 the calibration approach between the CCD and the PMD camera is described. After 2D and 3D data are registered, the coordinates of both pixels are registered. It can be assumed that each pixel of the CCD and PMD camera have the same coordinate. The object detection technique acquired from the CCD camera is transferred to the 3D coordinate.

A proper distance calibration is absolutely necessary due to the algorithm in dependence on an accurate geometric reconstruction. In Figure 6.8, the acquisition of an artificial landmark scenario is illustrated. The artificial landmark is detected in a 3D environment that can be extended to the future three dimensional mapping in real time. Figure 6.9 shows one application of moving object detection from 3D data. The artificial landmark is stuck behind the human being. The mobile robot can run automatically following the moving human being. This application can be applied in a supermarket. A cart can follow a human being who is shopping, or an incapacitated person who cannot move the cart by himself.

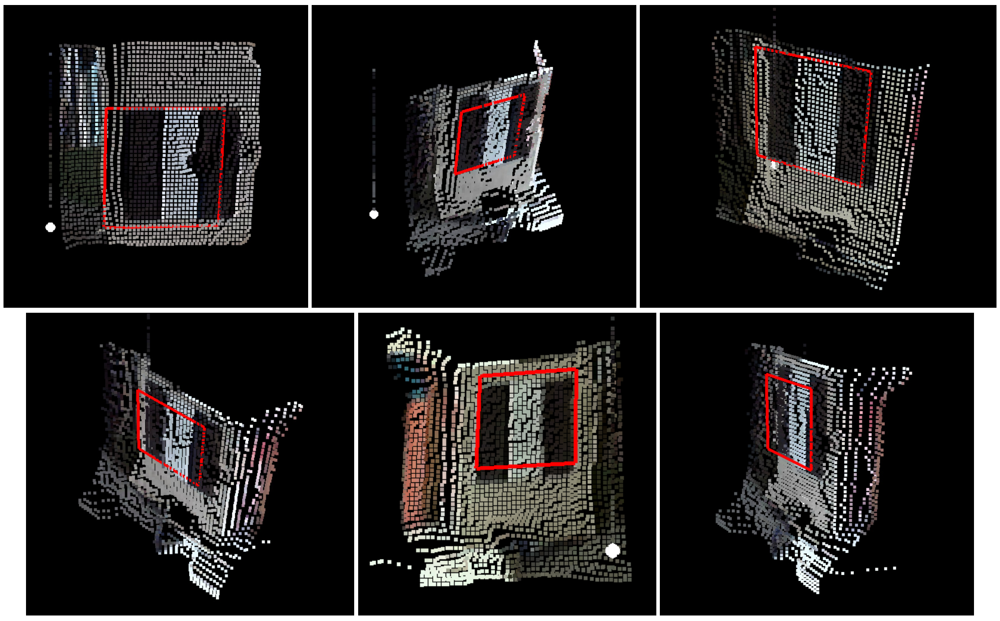


Figure 6.8 3D artificial landmark detection

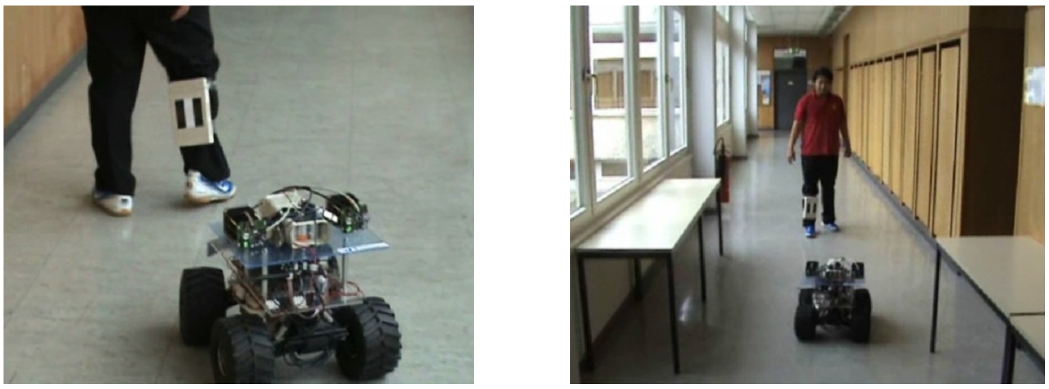
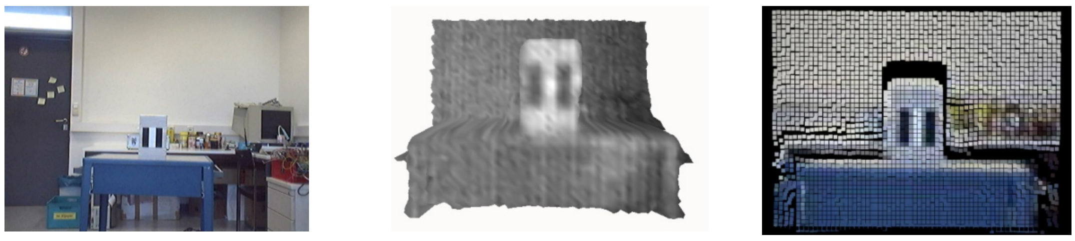


Figure 6.9 Following the moving objects

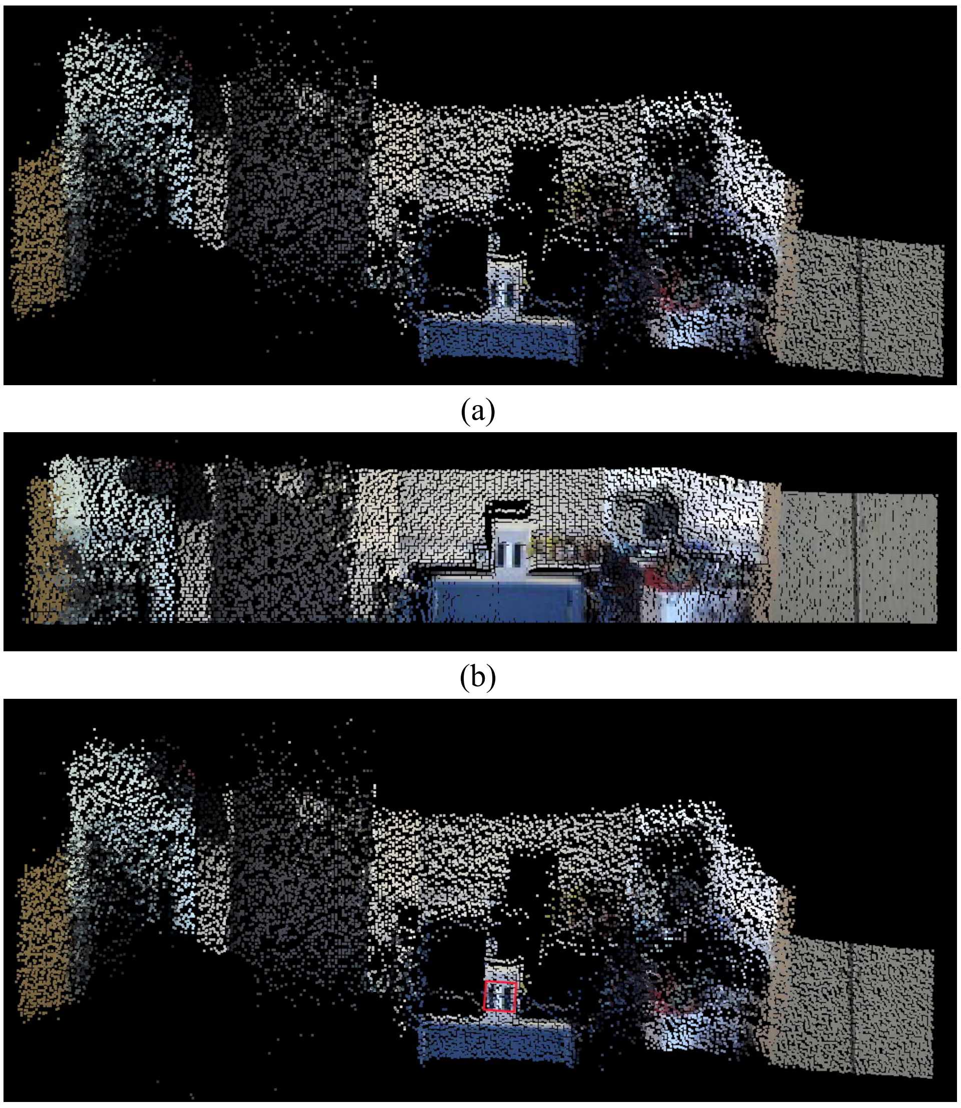
6.3 Off line 3D mapping operation

To understand the 3D mapping, the 3D mapping off-line operation is tested. The PMD camera, which is used in the experiment, is the 3k-S model with the resolution of 64× 48 pixels. The horizontal field of view (FOV) is approximately 10.0° and the vertical 12.5°. The 2D camera has a resolution of 640 × 480 pixels, 45° and 34° are the values of the horizontal and vertical FOV.



(a) 2D image (b) gray scale PMD (c) combination

Figure 6.10 One snap shot for 3D mapping



(a)

(b)

(c)

Figure 6.11 Off-line 3D mapping and object detection

The different FOV and mounting position between both cameras impact on the overlap between the output frames. Figure 6.10 shows the one snap shot used to generate 3D mapping of differential field of view from PMD and 2D cameras [100]. Figure 6.10 (a) is the field of view of 2D image. (b) is the depth data (shown in gray scale) from the PMD camera. (c) is the combination data from 2D/3D camera. One snap shot in (a) and (b) expresses that the field of view of both camera is really different. Thus, the calibration model is essential. The off-line testing is tested by snap data shot to shot. The rotation angle is known, which means that the overlapping of each frames is too small. The matching approach then could generate very precise 3D mapping. Figure 6.11 shows the 3D map which is approximately 7 meters long. Not only depth volume is shown, but also the scenario contour is shown. The scenario (left to right) consists of a cabinet, door, artificial landmark, computer monitor and cabinet again. This data can be applied to mobile robot intelligent locomotion and the user also knows what those certain scenarios are. Furthermore, the real-time object detection is implemented in the experiment. In Figure 6.11 (c), we can see the red square that is in the position of an artificial landmark, which is detected.

6.4 Real time operation

6.4.1 Mapping without the combination of 2D/3D cameras

This section shows the real time operation of 3D mapping without the combination with the 2D camera [101]. Figure 6.12 demonstrates the testing when the camera position (translation) is fixed, but the camera rotates around the room 180 degree. The 3D map output can show the dimension of the closed room, but users probably do not understand the detail of this scenario. Actually, this output is sufficient for autonomous mobile robot navigation. Figure 6.13 shows another environment on the indoor corridor. The output could also show the map of mobile robot trajectory. Nevertheless, the qualities of the map output are not precise like the off-line operation because the robot movement speed and mapping approach have to be related. The synchronization is then necessary for providing the high quality output.

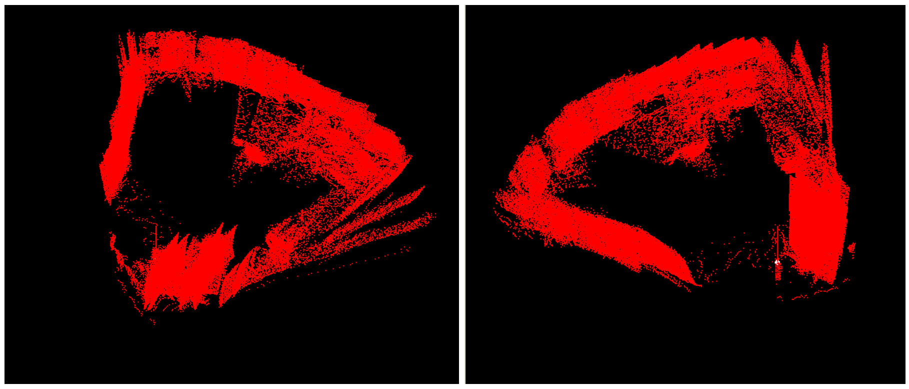
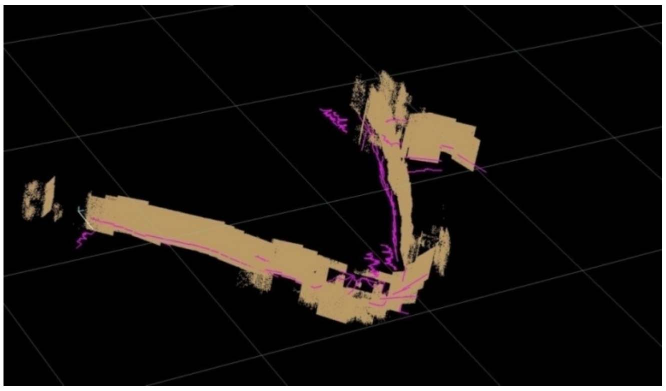


Figure 6.12 Closed room



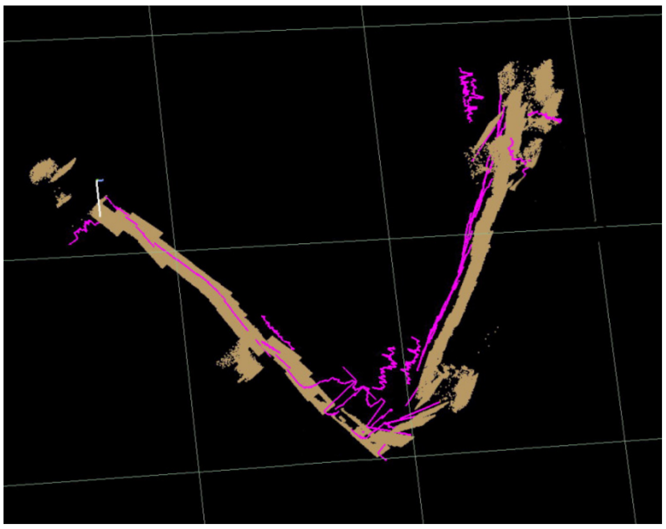
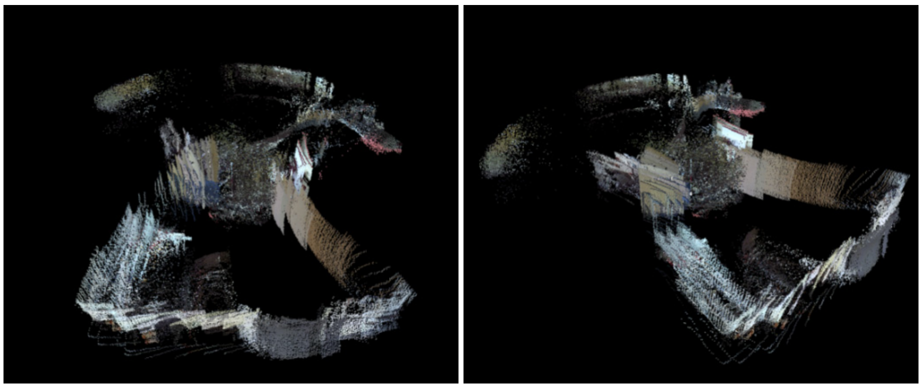


Figure 6.13 3D mapping at the corridor

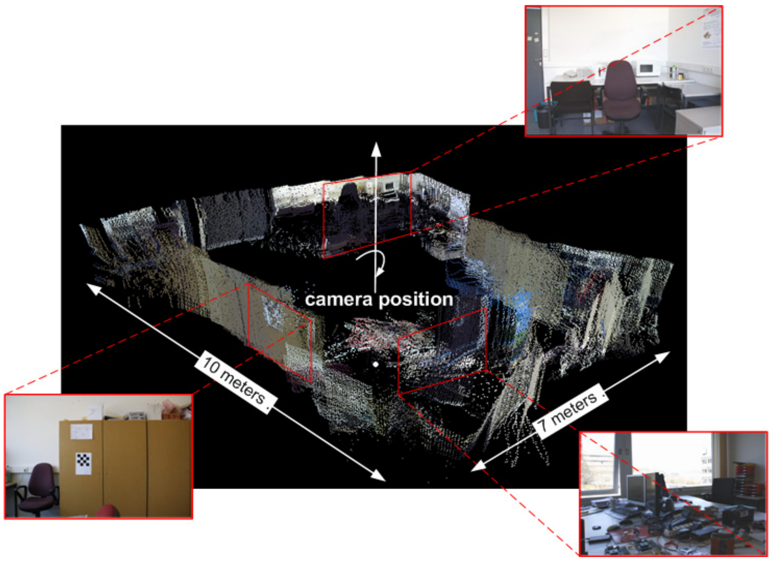
6.4.2 Mapping with the combination of 2D/3D cameras

Figure 6.14 shows the experiment in the parallelogram closed room. The testing fixed the position (translation) of the camera but rotated the camera 360 degrees clockwise. It can be seen that the shape of the closed room is shown. However, the top area of 3D map image shows the shape of the room is distorted because that area has windows and is a reflecting area. The sunshine from outside comes into the room and that affects the PMD depth data. This testing explains that the PMD camera still has a problem with the outdoor environment. The closely wavelength of PMD led source and sunshine is directly affected to the PMD depth data. Thus, testing should avoid strong sunshine areas. Thus, if we use the PMD camera with very strong sunshine, the PMD depth data is affected. Therefore, that recommends that the PMD camera be used only for indoor applications. Figure 6.15 shows testing in the same scenario again, but the windows are covered by a curtain. The sunshine could not come into the room, but the influence of light is still seen. The experiment shows that the influence of light has not much effect on the quality of PMD depth data. We can see that the shape of the room looks better. The size of the room is around 7×10 meters. The expanded red squares show the original image from 2D camera compare with the

3D scenario. However the 3D scenario can show the dimension of parallelogram closed room, even through the outliner noise is still occur.


(a) (b)
Figure 6.14 The open room with strongly sunshine

The next section shows the testing in order to generate the three dimensional mapping by combination of 2D and 3D cameras. The output is described in detail and gives the user more understanding of the 3D environment in detail, instead of using only the PMD camera. Figure 6.16 closed-up in a short distance in the map along the 90 degree corner around 10 meters. We can see the texture of environment and the detail of that scenario. Figure 6.17 show the testing on along the distance while the mobile robot moves in the closed loop indoor corridor. Figure 6.18 shows the schematic diagram of the tested environment. Figure 6.19 shows the output from testing. The 3D mapping shows the dimension of environment 25×20 meters with 5.30% and 5.00% error compare with the schematic diagram.



(a) (b)
Figure 6.15 The parallelogram closed room obtained during rotating the camera 360 degree clockwise

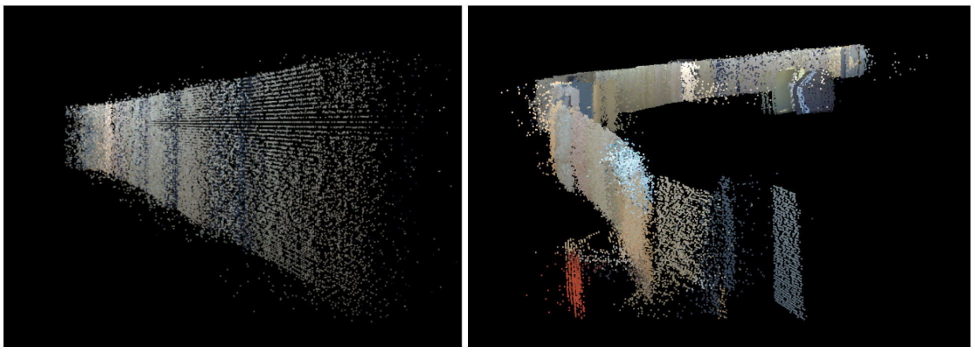


Figure 6.16 Closed-up short distance testing at the 90 degree corner

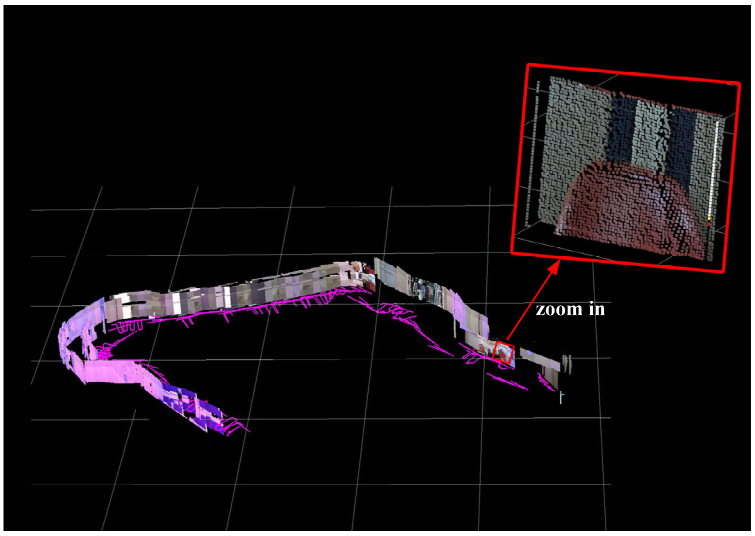


Figure 6.17 First corridor scenario and a close-up view of the 3D map

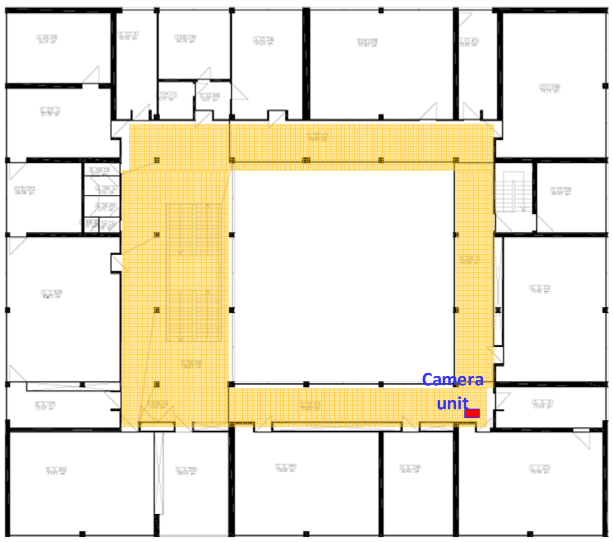


Figure 6.18 The schematic diagram of tested corridor dimension 26.40 x 21.00 meter

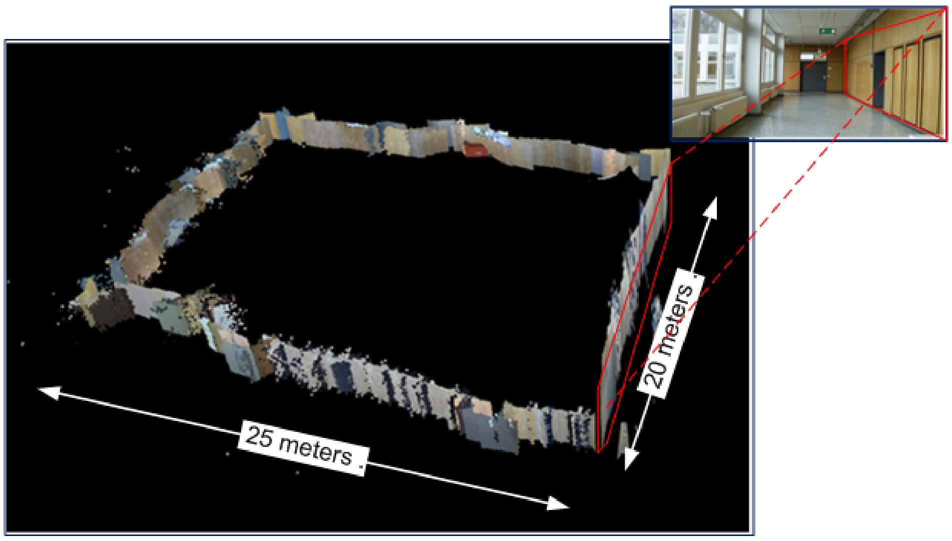


Figure 6.19 Second corridor obtained during the mobile robot moves along the indoor closed scenario

6.5 Conclusion

This chapter describes the evaluation of motion estimation, the output of the 3D mapping, the combination of 2D/3D mapping, and the object detection results. The stereo camera is a well-know inactive 3D sensor. It is used for the evaluation of the motion estimation. The results show that the translation result from the PMD camera is better, but that the stereo camera rotation result is an advantage. Eventually, the combination sensor is the best solution for using those sensors in the motion estimation tasks. The object detection approach can detect the artificial landmark in the 3D environment real time. The results are satisfactory and can be applied in future variety applications. One lacking of the PMD output is the small field of view which is the reason that the map can be generated on one side of the robot only. The new PMD camera model, which has a bigger field of view can enhance the quality of the output. Alternatively, the construction of the mechanical rotating part to rotate a camera from 0°-180° can solve this problem but the speed of robot movement may be decreased. Another solution is using two cameras. To equip a camera point to the left and right hand side, thus the mobile robot can move at the same speed while generating the 3D mapping in real time.

Chapter 7

Conclusions

7.1 Conclusions and summaries

This thesis presents the autonomous navigation system by modeling the small indoor mobile robot. One of the interesting problems of mobile robot applications is the map building to detect the interested objects in any environment. The car-like robot model is used in the experiments because it could locate smoothly and it was easy to control the steering angles. Furthermore, the speed control is flexible for obstacle avoidance behavior. The main embedded computer is powerful enough to process the image frame from the 3D sensors. The servo motors are used to steer the front wheel, and the DC motor is the main power source to drive the robot. All sensor data are calculated in the 16 bit microcontroller. The 3D magnetometer is also equipped for measuring the rotating 3D axis. Several sensors are investigated for designing, which sensors are suitable to acquire the depth information from scenario. The PMD has the capability to capture the real-time depth data in matrix frame data.

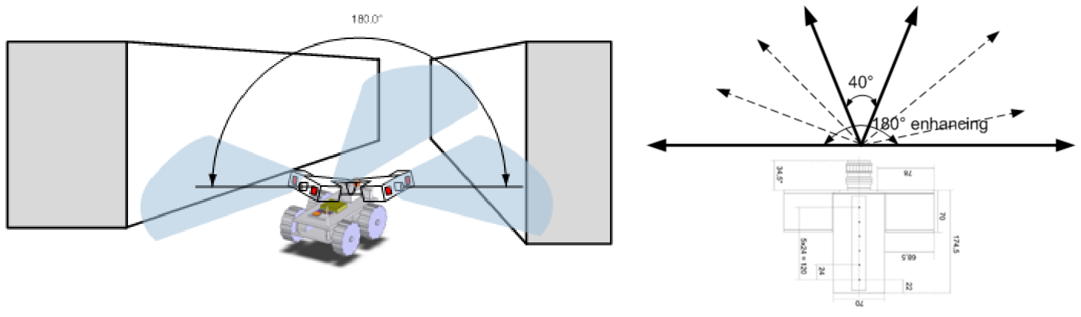
This thesis aims to propose and focus on the new 3D depth sensors, PMD camera, which is still new in the mobile robot application communities. In addition, the PMD camera itself has pros and cons in its ability to capture the 3D data. The fast data frame rate is excellent for detecting the moving objects. However, with non-moving objects, it is normally difficult to distinguish what the objects are. The solution to this problem is the high resolution 2D, CCD camera, employing the PMD for depth data. The output result from the combination is the depth data and contour information of the 3D environment. The high resolution from the 2D camera is used for detecting the objects in the clutter area or non-moving objects. The Haar-like features are also studied. This approach has a very high accuracy and performance, as well as a fast tracking rate. We apply the Haar-like feature technique to detect the artificial landmarks that are hung in the environment. The artificial landmarks are assumed to be any object which users want to detect. From this task, the mobile robot can detect the interesting objects. If we attach the artificial landmark to a human being (back or leg), it can be applied in a way that the mobile robot interactive movement follows the humans or moving objects. The main interests of this thesis are not only to build the autonomous mobile robot, but also to present the application to build the map of mobile robot trajectory. The main problem of high quality output is the image processing time. The detecting distance sensor and noises directly affect the processing time of the system. If the detecting distance is too short, the mobile robot is often required to update the new distance. If the sensor is affected by the ambient environment, noise occurs. The noise filter is then required to reduce the outliner points. The median filter is a well-known approach to reduce the 3D outliner point. It is used for the principle of the convolution window with a constant coefficient factor to reduce the salt-pepper noise from the 3D sensor data. The main effort put into this research is the integration of the image processing time and the robot locomotion. The robot speed and the map building have to match in order to create the high quality three dimensional map.

Eventually, the mobile robot, MERLIN is able to avoid the unknown obstacles and escape from unpredictable objects. It can also be controlled with the joystick interface from the user work station via wireless RS232 module. However, because of the distance limitation from the PMD camera, the MERLIN can operate at approximately 0.5 meter/second. In the final experiment, the 3D mapping and object detection approaches are shown. The system demonstrates that the objects could be detected in the 3D scenario, even though they are in a clutter environment. The experiment shows that the system is robust and efficient within the real unexpected scenario.

7.2 Future works

The works in this thesis are intended to demonstrate the car-like autonomous mobile robot applications and the 3D map building. Some parts of this work can be improved upon for future work.

The performance of the system is focusing on the new 3D sensor, PMD camera. Nowadays, the technology of the PMD camera is improving step by step. The newest model has a frame rate of 40 fps with 200 × 200 pixels, field of view 40° × 40°. This can increase the 3D mapping versatility. Moreover, the field of view of the PMD camera we use is approximately 10.0° horizontal and 12.5° vertical. This field of view is rather small for covering all mobile robot application. The camera field of view can be enhanced by rotating the camera from 0°- 180°. Moreover, the further suggestions can improve by following.



(a) Rotate the camera or use two cameras (b) Increasing the field of view

Figure 7.1 Enhancement of field of view

- The mobile robot hardware can be improved for more performance by including the increment of the number of sensors, e.g., the inferred on robot rear, left-right side, energy checking system, increased high precise encoder.
- For the co-operative multi robot operation; the speed of map building can be increased by using the multi robots co-operation. If the system has several robots, the work will independent and send the information to the same map server. The shared data can improve the speed of map building when the map covers a large area.
- For the outdoor environment, the grounding robot and flying Zeppelin are in the plans for operation. The co-operation between multi distributed mobile robots is performed in order to generate the real-time outdoor 3D mapping. The mobile robots are separated in two groups, i.e., flying airship and ground based mobile robots. The flying airship predictably investigates the overall terrain from the top view. The information of the terrain, i.e., flat, roughly, barbed wire as well as dead end is transferred to the ground based mobile robots. The 3D mapping then is built efficiently and faster than with only ground based mobile robots. Therefore, the flying airship solves one problem by making ground based mobile robots more suitable for all types of terrains.

It would be of interest to test the mobile robot in several environments. Referring to cooperative mobile robots, the performance of small robot communities will hopefully be enhanced in the next step. The speed of exploration should be increased, but the mathematical model is also more complex. The mentioned aspects will be challenging tasks for future works.

Appendix A

Mini Computer MICROSPACE-PC41

Feature

- Pentium M-738 Standard
- 512MB Memory
- 80GB disk
- DVD/CDRW Drive
- 5x USB V2.0
- LAN-Port A • 2x IEEE1394
- LPT and COM1
- DirectX 9, 64MB video Memory, VGA, DVI, TV In/Out
- MiniPCI connector
- Wake up from LAN and Boot over LAN (RPL/PXE)
- Fanless operating temperature +5°C to +50°C
- Small Dimensions: 159 x 245 x 66mm
- MPEG2 Encoder

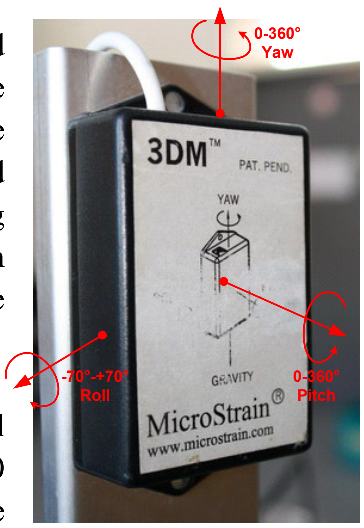
Ordering Information Description

Item Codes	Part#	Description
Microspace-PC41	812470	For MPEG2 Video-streaming
Options		
Wireless LAN	813087	WLAN (must be ordered with unit)
1GB	807363	Upgrade to 1GB DDR memory
DVDRW	807371	Upgrade to DVDRW
DVD/CDRW	807376	Remove DVD/CDRW
CF upgrade	807373	Compact Flash TypeII
Celeron-M300	807356	Replace CPU with CeleronM300
Celeron-M373	807357	Replace CPU with CeleronM373
Pentium-M745	807358	Replace CPU with Pentium M745
Pentium-M755	807358	Replace CPU with Pentium M755**
US Power Cable	100-9639	US Power Cable
Active cooler	814190	USB 5VDC fan
MPC-AC-cooler	814195	Mechanical interface to a vehicle A/C

The MICROSPACE PC normally comes equipped with a hard disk, memory, video, USB ports, LAN ports and has been factory tested to assure functionality. Each model has a unique feature incorporated to differentiate it from the other models. The unit can come pre-configured with the operating system and drivers in- stalled.
The MICROSPACE PC41 may be installed horizontally or vertically as well as mounted in a vehicle - it is very versatile. Its Remove DVD/CDRW compact size makes the computer easy to install in places where limited space is available. The MICROSPACE PC also works well in extremely dusty environments, since no dust particles are drawn in (fan-less design) as it is protected by the book size aluminum case. It is designed for maintenance free long-term operating in a rugged environment. These are the special features of the MICROSPACE PC which make it the smallest, quietest PC we know of in the market with this much processing power. Each MICROSPACE PC model is equipped with unique features and a wide range of options. Please see backside for details and contact

Appendix B

3DM Magnetometer



The 3DM is three magnetometers and accelerometers to calculate pitch, roll and yaw. We use the magnetic compass from MicroStrain. The three angles are relative to the earth's magnetic and gravitational field. The output is capable of measuring angles from 0-360 degrees on yaw, 0-360 degrees on pitch and -70-+70 degrees on roll. It calculated the yaw angle using the magnetic field from the earth and compensates the errors using the accelerometers. The data is sent via serial communication (COM Port) with a board rate of 9600 bit/sec using the VC++ to receive the raw data. The 3DM is a three magnetic compass using magnetometers and accelerometers to calculate pitch, roll and yaw angles. The three orientation angles are provided via serial RS232 or RS485. The device is capable of detecting three orientation angles as illustrated in table B.

Angle	**Degree**
Yaw	0 – 360
Pitch	0 – 360
Roll	-70 - +70

Table B 3DM magnetic specification

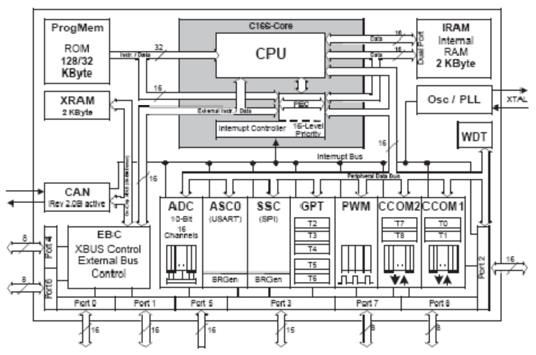
The 3DM is sensitive to magnetic field and electromagnetic compatibility (EMC), thus mounting is required to avoid EMC sources such as chassis, motors as well as batteries. The mounting position is then made of aluminum approximately 20 cm higher than the robot body. The source code below shows the fundamental program to read the three angles via serial RS-232 port by using the Visual C++. The table below shows an example reading the data source code via the serial RS-232 port, (COM1). The absolute roll, pitch and yaw can be read in real time.

```
using namespace std;

FILE *stream;
DCB dcb = {0};
HANDLE hComm;
BOOL fSuccess, fSuccess2;
unsigned char buffer2, buffer3[14];

       bool write_status;
        unsigned char szBuffer[14];
       DWORD dwRead, dwWritten;
       OVERLAPPED ovlr = {0}, ovlw = {0};
       COMMTIMEOUTS cto;

       char message;
       char *lpBuffer=&message;

       int roll, pitch, yaw, DeltaYaw;

void initialRS232()
{
    ovlr.hEvent = CreateEvent(NULL, TRUE, FALSE, NULL);
    ovlw.hEvent = CreateEvent(NULL, TRUE, FALSE, NULL);

    DWORD num_bytes_written = 0;

    // configulation COM 1
    hComm  =  CreateFile(TEXT( "COM1"),  GENERIC_READ  |  GENERIC_WRITE,    0,  NULL,
OPEN_ALWAYS, FILE_ATTRIBUTE_NORMAL|FILE_FLAG_OVERLAPPED, 0);

       dcb.BaudRate = CBR_9600;   // BaudRate 9600
       dcb.ByteSize = 8;          // number of byte
       dcb.StopBits = ONESTOPBIT; // StopBits
       dcb.Parity   = NOPARITY;

       dcb.DCBlength = sizeof(dcb);
       GetCommState(hComm, &dcb);
       SetCommState(hComm, &dcb);

               cto.ReadIntervalTimeout         = 1000;
               cto.ReadTotalTimeoutConstant    = 1000;
               cto.ReadTotalTimeoutMultiplier  = 1000;
               cto.WriteTotalTimeoutConstant   = 1000;
               cto.WriteTotalTimeoutMultiplier = 1000;

       szBuffer[dwRead] = 0;
}

void get3DM_Data()
{
    int i;
       message = 0x74;//'t';        //send 't'= 0x74 to 3DM for 'Poll-Mode'
       WriteFile(hComm, &message ,1 , &dwWritten, &ovlw);
       Sleep(35);

    //buffer2 = '4';//0xC0;       //send '1100 0000' = 0xC0 to 3DM for 'Command 4'
    message = 0xC0;
       WriteFile(hComm, &message ,1 , &dwWritten, &ovlw);
       Sleep(35);

       ReadFile (hComm, &buffer3, 1, &dwRead, &ovlr);
       if(buffer3[0]!=0x44)
               printf("Error 0x44\n");


    for(i=0; i<6; i++)
               buffer3[i]=0;

       ReadFile (hComm, &buffer3, 6, &dwRead, &ovlr);
} /* End of 'get3DM_Data()' */

void get_RollPitchYaw() {

    roll  =  (int)buffer3[0]*256  +  buffer3[1];                        //Alternative:
roll=(int)buffer3[0];
    if(roll>=49151) roll = (int)(0.0055*roll-361);
    else roll = (int)(0.0055*roll);


    pitch  =  (int)buffer3[2]*256  +  buffer3[3];                       //Alternative:
pitch=(int)buffer3[2];
    if(pitch>=32768) pitch = (int)(360-0.0055*pitch);
    else pitch = (int)(-0.0055*pitch);

    yaw  =  (int)buffer3[4]*256  +  buffer3[5];                         //Alternative:
yaw=(int)buffer3[4];
    yaw = (int)(360-0.0055*yaw);


} /* End of 'get_RollPitchYaw()' */

void GetCompass (int yaw)
{
 static int State = 1;
 static int LastReading;
 int CurrentReading;

 switch ( State )
 {
 case 1:
     LastReading = yaw;
     State = 2;
     break;

 case 2:
         State = 1;
     CurrentReading = yaw;
     DeltaYaw = CurrentReading-LastReading;
     break;
 }
}

void main(void)
{
     initialRS232();
     get3DM_Data();
     get_RollPitchYaw();
     printf("\nRoll=%i Pitch=%i Yaw=%i\n", roll, pitch, yaw);
     GetCompass (yaw) ;
     CloseHandle(hComm);
}
```

Appendix C

Microcontroller Infineon C 167

The rapidly growing area of embedded control applications represents one of the most time-critical operation environments for today's microcontroller. The microcontroller is a very important part of a mobile robot. It can be compared to the brain of a human being. It is used to compute every activity which the robot should do. It receives the output signal of the sensor, the command from the computer and the wireless communication data. On the other hand, it can also send the controlling commands to the motor control circuit. Presently, there are many kinds of microcontrollers. Many companies produce many high benefit microcontrollers, for example PIC, 8051, ARM, 68HC, Z80, BASIC STAMP, MPS430, Infineon. Each type of microcontroller has different characteristics i.e., I/O port, analog to digital converter, pulse width modulation, CAN interface capacity of RAM and ROM, speed of processor. Infineon is one type of microcontroller which consists of important parts for the construction of a high efficient mobile robot. For this reason my thesis focuses on microcontroller Infineon C167CR-LM.



(a) C167 Functional Block Diagram

(b) Microcontroller

Instruction

Infineon C167CR-LM is a 16-bit microcontroller of the Infineon C166 family, the C167-class. It has been designed to meet the high performance requirements of real-time embedded control applications. The architecture of this family has been optimized for high instruction. The C167CR-LM is an improved representative of full featured 16 bit single-chip CMOS microcontrollers. The architecture combines the advantages of both RISC and CISC. The C167CR not only integrates a powerful CPU core and a set of peripheral units onto on chip, but it also connects the units in a very efficient way.

The PHYTEC miniMODUL-167

Before using microcontroller C167CR-LM, this chip consists of an accessory circuit which is used for communication to a computer or interfacing with input/output port. The PHYTEC miniMODUL-167 is belonging to PHYTEC Meßtechnik GmbH. They combine with C167 to create a suitable accessory circuit. The miniMODUL-167 is intended for the use in memory-intensive applications and running within a CAN-bus network and is fitted with two RS-232 transceivers, a CAN-bus interface and an additional UART to provide the second asynchronous serial interface and a CAN-bus interface. These pins provide a 16-bit bidirectional I/O port, including 16 analog inputs with 10-bit resolution. The useful features of the miniMODUL-167 follow.

- 16-bit microcontroller, running at a 20 MHz clock speed.
- Delivering instruction cycles in 100 ns.
- 400/300 ns multiplication (16-bit x 16-bit), 800/600 ns division (32-bit/16-bit).
- Integrated on-chip memory (2 KB internal RAM, 2 KB on-chip high-speed RAM,
  and 128 KB or 3 KB on-chip ROM).
- 256 KB (to 2 MB) external SRAM.
- 256 KB (to 2 MB) external Flash.
- Flash supports on-board programming via RS-232 interface.
- Up to 1 MB optional EPROM.
- 16-parity-level interrupt system.
- 16-channel 10-bit A/D converter with programmable conversion time (7.76 *ms)*
  minimum) with auto scan mode.
- 4-channel PWM unit.
- Asynchronous/Synchronous serial channel (USART).
- Five general purpose16-bit timers/counters.
- 16-bit bi-directional I/O port.
- High-level language support.

Keil Software

Software development tools for the C16x, ST10, and XC16x support every level of developer from the professional applications engineer to the student just learning about embedded software development. The industry-standard Keil C Compiler, Macro Assembler, Debugger, Real-time Kernel, and Single-board

Computers support ALL C16x derivatives and help you getting your projects completed on schedule. These higher languages would then be compiled automatically into a machine language, which you can then upload into your robot. Probably the easiest language to learn would be **BASIC**, with a name true to itself. The BASIC Stamp microcontroller uses that language. But BASIC has its limitations, so if you have any programming experience at all, I recommended you to use the program in **C**. This language was the precursor to **C++**, so if you have already gained programming experience in C++, it should be really simple for you to learn. This aspect is complicated by the fact that is no existing standard to program microcontrollers. Each microcontroller has its own features, its own language, its own compiler, and its own uploading to the controller method. This is why I do not go into too much detail because there are too many options out there to talk about. The supporting documents that are delivered with the controllers should answer your specific questions. In addition, if you decide to use a PIC, you have to be aware of the fact that the compiler program (at least the good ones) can cost hundreds of dollars.

Appendix D

2D Image Processing Fundamental

The 2D image processing in this thesis was handled by using the OpenCV open source library. The OpenCV was used to capture the image frames from CCD camera and also process the fundamental data e.g. camera frame capturing, edge detection, point detection and pixel color reading. The fundamental setting up e.g. library files, include files, sources files installation can be found in OpenCVWiki [102]. Following examples show the beginning program for capturing image frame via USB, Canny edge detection, save data to hard drive, read pixel color,

<u>Beginning with OpenCV, VC++</u>

Reading data from 2D camera: Read image frames from the USB camera. The image frames are captured in pointer *cv_cap

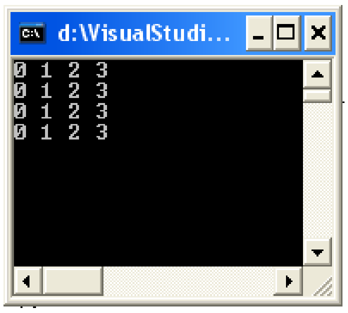
```
#include "stdafx.h"
#include <windows.h>
#include <cv.h>
#include <highgui.h>

void main(int argc,char *argv[])
{
        int c;
        IplImage* color_img;
        CvCapture* cv_cap = cvCaptureFromCAM(-1);
        cvNamedWindow("Video",1);                   // create window
        for(;;)
        {
                color_img = cvQueryFrame(cv_cap);   // get frame strucre
                if(color_img != 0)
                cvShowImage("Video", color_img);    // show frame
                c = cvWaitKey(10);                  // wait 10 ms or for key stroke
        }
cvReleaseCapture( &cv_cap );
cvDestroyWindow("Video");
 }
```

Edge detection: Detect the image edge by using the Canny edge detection. The output is shown in pointer “Myimage”. The output shows the edges which have in the original image.

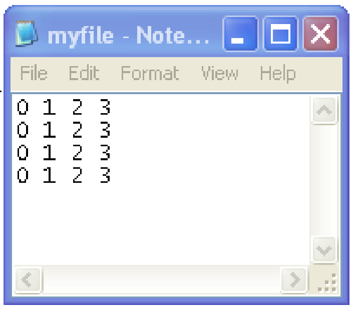
```
#include "stdafx.h"
#include<stdio.h>
#include<cv.h>
#include<highgui.h>
#include<stdlib.h>

int main()
{

        IplImage *image = 0;
        IplImage *img1,*img2;
        IplImage *gray_scale;

        double LowThreshold;
        double HighThreshold;
        int ApertureSize;

        IplImage* Myimage = cvLoadImage("D:/VisualStudio2005/ko2.jpg",0);
        cvNamedWindow("My Window", CV_WINDOW_AUTOSIZE);
        cvCanny(Myimage,Myimage,140,80);
        cvShowImage("My Window", Myimage);
        cvSaveImage("ImageEdge.jpg", Myimage);

        cvWaitKey(0);
        cvDestroyWindow("My Window");
        return 0;
}
```

Safe data to hard drive: The data after processing have to be saved in the hard drive in order to recall them again when we want to show. The format files are saved in text file.

```
#include "stdafx.h"
#include<stdio.h>
#include<cv.h>
#include<highgui.h>
#include<stdlib.h>

int main()
{
    int ApertureSize[4][4],i,j;
    FILE * pFile;
    pFile = fopen ("myfile.txt","w");
        for(int i=0;i< 4 /*Myimage->width*/;i++)
        {   pFile = fopen ("myfile.txt","a");
                for(int j=0;j<4 /*Myimage->height*/;j++)
                        {
                        ApertureSize[i][j]=j;
                        fprintf (pFile, "%d ",ApertureSize[i][j]);
                        printf("%d ",ApertureSize[i][j]);
                        }
                 fprintf (pFile, "\n");
                 fclose (pFile);
                 fprintf(stream, "\n");
        }
         return 0;
}
```



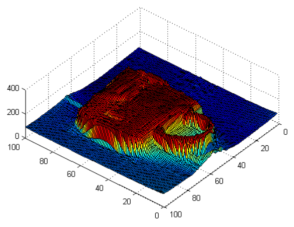
```
0 1 2 3
0 1 2 3
0 1 2 3
0 1 2 3
```



```
0 1 2 3
0 1 2 3
0 1 2 3
0 1 2 3
```

Read pixel color: Read pixel data is important for basically data processing. CvGet2D is the library from OpenCV that allow reading the data in each image pixel. The pixel data are stored in pointer "s"

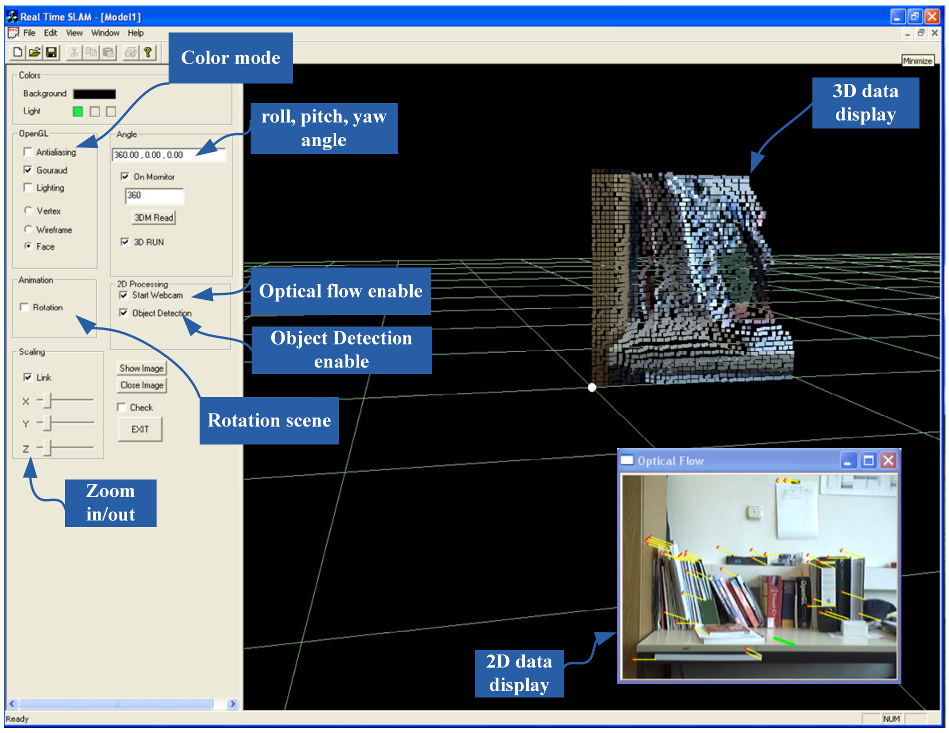
```
#include "stdafx.h"
#include "cv.h"
#include "highgui.h"
#include <stdio.h>

int main ()
{       int i,j;

    FILE * pFile;
    pFile = fopen ("myfile.txt","w");
    IplImage* Myimage = cvLoadImage("D:/VisualStudio2005/cup2.jpg",0);

for (i=0; i< Myimage->height ;i++)
        {  pFile = fopen ("myfile.txt","a");
           for (j=0; j< Myimage->width ; j++)
                        {
                       CvScalar s;
                       s=cvGet2D(Myimage,i,j);
                       printf("%3.0f ",s.val[0]);
                       fprintf (pFile, "%3.0f ",s.val[0]);
                       }
                       printf(";\n");
                       fprintf (pFile, ";\n");
                       fclose (pFile);
               }

      printf("width=%d\n height=%d\n nChannels=%d\n widthstep=%d\n",Myimage-
>width,Myimage->height,Myimage->nChannels,Myimage->widthStep );
        cvNamedWindow("My Window", CV_WINDOW_AUTOSIZE);
        cvShowImage("My Window", Myimage);
        cvWaitKey(0);
        cvReleaseImage(&Myimage);
        cvDestroyWindow("My Window");
        return (0);
```



Appendix E

OpenGL Based on MFC

OpenGL is the library to be used for showing the 3D data. Using only OpenGL library and VC++ are low level programming. Users can not interact real-time to the program. Thus, user interactive program is required in order to communicate and update data to the users. Graphic User Interface, GUI, is used to enhance the performance of user interactive program. In this thesis selected Microsoft Foundation Classes, MFC, to be a GUI based program. MFC is a library that wraps portions of the Windows API in C++ classes, including functionality enables to use a default application framework. The fundamental template of OpenGL based on MFC has been provided from Alliez [103]. We applied that template to this thesis which is shown below.



The MFC page combines OpenGL, OpenCV and MATLAB engines in order to generate the 3D map. The GUI page can show the 3D data in real-time, the 2D data, enable/disable optical flow and object detection functions, automatic rotation scene, change color mode, zoom in/out and three rotation angles (roll, pitch, yaw) from the 3DM sensor.

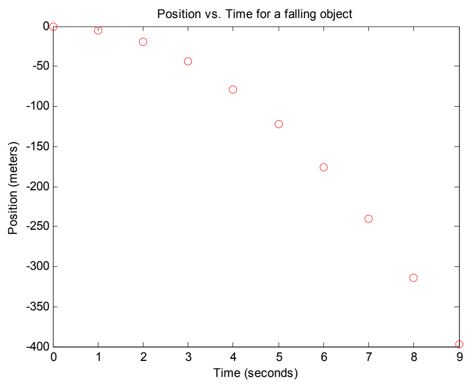
Appendix F

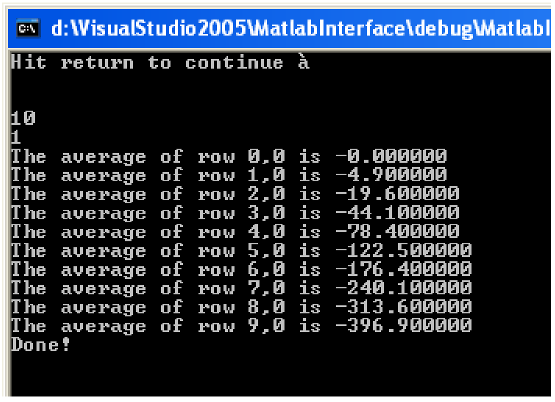
MATLAB Interfacing Engine

Mathematic calculation is importance part for calculation all processes. That is part which consume the most processing time. The one well-known library is GNU library. The GNU Scientific Library (GSL) is a routines collection for numerical computing. The routines have been written form scratch in C. The GSL provides several numerical calculations in C library code. It can calculate complex numerical problem very fast. However, the language structure is still low level. The rearrange source code and understanding are rather difficult. The second choice is using MATLAB interfacing engine. As known, MATLAB is a powerful calculating program. The program structure is easy to comprehend, even though the calculation time is slower than GNU. The MATLAB engine interfacing [104] is then used. The method is VC++ main program sends data to MATLAB via MATLAB engine. The numerical problems are calculated by using MATLAB library. After getting the result, results are also returned back to VC++ main loop via interfacing engine. The following example program shows sending the array [10][1] to MATLAB. Use this data put in the non-linear equation and plot two dimension graph on MATLAB. Return those variable back to the VC++ main loop and show on the command page.

```
#include "engine.h"
#include "C:/Program Files/MATLAB/R2006a/extern/include/engine.h"
#pragma comment (lib,"C:/Program
Files/MATLAB/R2006a/extern/lib/win32/microsoft/libeng.lib")
#pragma comment (lib,"C:/Program
Files/MATLAB/R2006a/extern/lib/win32/microsoft/libmx.lib")
#pragma comment (lib,"C:/Program
Files/MATLAB/R2006a/extern/lib/win32/microsoft/libut.lib")

using namespace std;

int main()
{
  Engine *ep;
  mxArray *T = NULL, *d = NULL;
  double *xValues,avg;
  int rowLen, colLen,i,j;
  double time[10] = { 0.0, 1.0, 2.0, 3.0, 4.0, 5.0, 6.0, 7.0, 8.0, 9.0 };
  T = mxCreateDoubleMatrix(1, 10, mxREAL);
  memcpy((void *)mxGetPr(T), (void *)time, sizeof(time));
  engPutVariable(ep, "T", T);  // Place variable T into the MATLAB workspace
  engEvalString(ep, "D = .5.*(-9.8).*T.^2;");        // Execute a MATLAB command
  engEvalString(ep, "plot(T,D,'or');");
  engEvalString(ep, "title('Position vs. Time for a falling object');");
  engEvalString(ep, "xlabel('Time (seconds)');");
  engEvalString(ep, "ylabel('Position (meters)');");
  d = engGetVariable(ep, "D");
```

```
   xValues = mxGetPr(d);
   rowLen = mxGetN(d);
   colLen = mxGetM(d);
       printf("%d\n",rowLen);
       printf("%d\n",colLen);

       for(i=0;i<rowLen;i++)
          {
           avg=0;
           for(j=0;j<colLen;j++)
              {
               avg = xValues[(i*colLen)+j];
               printf("The average of row %d,%d is %f\n",i,j,avg);
              }
          }
   mxDestroyArray(T);                   /* Free memory */
   engEvalString(ep, "close;");
   engClose(ep);                        /* Close MATLAB engine */
   printf("Done!\n");
   scanf("%d");
   return EXIT_SUCCESS;
 }
```





```
Hit return to continue à


10
1
The average of row 0,0 is -0.000000
The average of row 1,0 is -4.900000
The average of row 2,0 is -19.600000
The average of row 3,0 is -44.100000
The average of row 4,0 is -78.400000
The average of row 5,0 is -122.500000
The average of row 6,0 is -176.400000
The average of row 7,0 is -240.100000
The average of row 8,0 is -313.600000
The average of row 9,0 is -396.900000
Done!
```

(a) Using MATLAB engine interfacing (b) Returning data to main VC++ program

Bibliography

[1] John J. Leonard and Hugh F. Durrant-Whyte, "Directed Sonar Sensing for Mobile Robot Navigation," *Cambridge,MA*, January 1992.

[2] Motilal Agrawal and Kurt Konolige, "Real-time Localization in Outdoor Environments using Stereo Vision and Inexpensive GPS," in *Pattern Recognition, 2006. ICPR 2006. 18th International Conference on* , 2006, pp. 1063-1068.

[3] Rainer Küummerle et al., "On Measuring the Accuracy of SLAM Algorithms," *Auton. Robots*, vol. 27, no. 4, pp. 387-407, 2009.

[4] Maksym Usov, "Vision Based Mobile Robot Navigation," Enschede, Netherlands, 2006.

[5] T.D. Arun Prasad, Klaus Hartmann, Wolfgang Weihs, Seyed Eghbal Ghobadi, and Arnd Sluiter, "First steps in Enhancing 3D vision technique using 2D/3D sensors," *Computer Vision Winter Workshop*, February 2006.

[6] Ingo Schiller, Christian Beder, and Reinhard Koch, "Calibration of a PMD-Camera using a planer calibration pattern together with a multi-camera setup," 2008.

[7] Devantech SRF04 Ranger. [Online]. http://www.acroname.com/robotics/parts/R93-SRF04.html

[8] R. Urick, *Principles of Underwater Sound*. New York: McGraw-Hill, 1983.

[9] P. Gough, A. de Roos, and M. Cusdin., "Continuous transmission FM sonar with one octave bandwidth and no blind time.," in *Autonomous Robot Vehicles*, 1990.

[10] 3DM: MicroStrain Pitch, Roll & Yaw sensor. [Online]. http://www.microstrain.com/3dm.aspx

[11] 3D Motion Tracking - Xsens. [Online]. http://www.xsens.com/

[12] David Nister, Oleg Naroditsky, and James Bergen, "Visual Odometry for Ground Vehicle Applications," Sarnoff Corporation, Princeton NJ,.

[13] J.-S. Gutmann and K. Konolige, "Incremental mapping of large cyclic environments ," in *Computational Intelligence in Robotics and Automation, 1999. CIRA '99. Proceedings*, 1999, pp. 318-325.

[14] MobileRobotss Inc. [Online]. http://www.mobilerobots.com/Mobile_Robots.aspx

[15] J. Borenstein and Y. Koren., "Real-time obstacle avoidance for fast mobile," *IEEE Trans. Systems, Man, and Cybernetics*, no. 19, pp. 1179-1189, September 1989.

[16] Luca Iocchi, Kurt Konolige, and Max Bajracharya, "Visually Realistic Mapping of a Planar Environment with Stereo," in *Proc. of Seventh International Symposium on Experimental Robotics (ISER)*, 2000.

[17] Motilal Agrawal, Kurt Konolige, and Robert C. Bolles, "Localization and Mapping for Autonomous Navigation in Outdoor Terrains : A Stereo Vision Approach," in *IEEE Workshop on Applications of Computer Vision (WACV'07)*, 2007.

[18] J. Diebel, K. Reuterswad, S. Thrun, J. Davis, and R. Gupta, "Simultaneous Localization and Mapping with Active Stereo Vision," in *Proceedings of 2004 IEEElRSJ International Conference on Intelligent Robots and Systems*, Smndal, Japan, 2004.

[19] Masahiro Tomono, "Robust 3D SLAM with a Stereo Camera Based on an Edge-Point ICP Algorithm," in *2009 IEEE International Conference on Robotics and Automation Kobe International Conference Center*, Kobe, Japan, May 12-17, 2009, pp. 4306-4311.

[20] P. Elinas, R. Sim, and J. J. Little, "σ SLAM: Stereo Vision SLAM Using the Rao-Blackwellised Particle Filter and a Novel Mixture," in *Proc. of ICRA2006*, 2006, pp. 1564–1570.

[21] M. A. Garcia and A. Solanas, "3D Simultaneous Localization and Modeling from Stereo Vision," in *Proc. of ICRA2004*, 2004.

[22] S. Se, D. Lowe, and J. Little, "Local and Global Localization for Mobile Robots using Visual Landmarks," in *Proc. of IROS2001*, 2001.

[23] David Schleicher, Luis M. Bergasa, Rafael Barea, Elena Lopez, and Manuel Ocafia, "Real-Time Simultaneous Localization and Mapping using a Wide-Angle Stereo Camera and Adaptive Patches," in *Proceedings of the 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Beijing, China, October 9 - 15, 2006.

[24] D.C. Herath, K.R.S. Kodagoda, and G. Dissanayake, "Stereo Vision Based SLAM Issues and Solutions," ARC Centre of Excellence for Autonomous Systems,University of Technology, Sydney, Sydney, Australia, ISBN 978-3-902613-01-1,.

[25] Woo Yeon Jeong and Kyoung Mu Lee, "Visual SLAM with Line and Corner Features," in *Proc. of IROS2006*, 2006.

[26] Thomas Lemaire and Simon Lacroix, "Monocular-vision based SLAM using Line Segments Robotics and Automation," in *2007 IEEE International Conference on*, Roma, Italy, 10-14 April 2007.

[27] Paul Smith, Ian Reid, and Andrew Davison, "Real-Time Monocular SLAM with Straight Lines," in *Proc. of BMVC2006*, 2006.

[28] Ethan Eade and Tom Drummond, "Edge Landmarks in Monocular SLAM," in *Proc. of BMVC2006*, 2006.

[29] "Real-time simultaneous localisation and mapping with a single camera," in *Proceedings of the Ninth IEEE International Conference on Computer Vision (ICCV'03)*, 2003.

[30] Hartmut Surmann, Andreas Nüchter, and Joachim Hertzberg, "An autonomous mobile robot with a 3D laser range finder for 3D exploration and digitalization of indoor environments," *Robotics and Autonomous Systems 45*, pp. 181-198, 2003.

[31] Andreas Nüchter, Kai Lingemann, Joachim Hertzberg, and Hartmut Surmann, "Heuristic-Based Laser Scan Matching for Outdoor 6D SLAM," , Koblenz, Germany, September 2005.

[32] Andreas Nüchter, Kai Lingemann, Joachim Hertzberg, and Hartmut Surmann,

"6D SLAM with Approximate Data Association," in *ICAR2005, IEEE Robotics and Automation Society*, 2005.

[33] Kai Lingemanna, Andreas Nüchtera, Joachim Hertzberga, and Hartmut Surmannb, "High-speed laser localization for mobile robots," *Robotics and Autonomous Systems 51*, 2005.

[34] Maurice Müller, Hartmut Surmann, Kai Pervölz, and Stefan May, "The Accuracy of 6D SLAM using the AIS 3D Laser Scanner," in *2006 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems*, Heidelberg, Germany, September 3-6, 2006.

[35] Henrik Andreasson and Achim Lilienthal, "Vision Aided 3D Laser Based Registration," in *Proc. European Conference on Mobile Robots: ECMR 2007*, 2007, pp. 192-197.

[36] Peter Biber, "The Normal Distributions Transform: A New Approach to Laser Scan Matching," University of Tübingen, Tübingen,.

[37] Oliver Wulf and Bernardo Wagner, "FAST 3D SCANNING METHODS FOR LASER MEASUREMENT SYSTEMS," Institute for Systems Engineering, University of Hannover, Hannover, Germany,.

[38] Oliver Wulf, Andreas Nüchter, Joachim Hertzberg, and Bernardo Wagner, "Benchmarking Urban Six-Degree-of-Freedom Simultaneous Localization and Mapping," , 2007.

[39] Fraunhofer IAIS. 3DLS. [Online]. http://www.3d-scanner.net/index.html

[40] Dirk Holz, David Droeschel, Sven Behnke, Stefan May, and Hartmut Surmann, "Fast 3D Perception for Collision Avoidance and SLAM in Domestic Environments,".

[41] Carlos Lara and Leonardo Romero, "Robust Local Localization of a Mobile Robot in Indoor Environments Using Virtual Corners," in *CIARP 2007*, Morelia Mich. Mexico, 2007, pp. 901-910.

[42] H. Gonzalez-Banos, E. Mao, J.C. Latombe, T.M. Murali, and A. Efrat, "Planning Robot Motion Strategies for EfficientModel Construction," Stanford University, Stanford, CA 94305, USA,.

[43] Dorit Borrmann, Jan Elseberg, Kai Lingemann, Andreas Nüchter, and Joachim Hertzberg, "Globally consistent 3D mapping with scan matching," in *Robotics and Autonomous Systems 56*, 2008, pp. 130–142.

[44] Omnidirectional camera. [Online]. http://en.wikipedia.org/wiki/Omnidirectional_camera

[45] Jungho Kim, Kuk-Jin Yoon, Jun-Sik Kim, and Inso Kweon, "Visual SLAM by Single-Camera Catadioptric Stereo," in *SICE-ICASE International Joint Conference 2006*, Busan, Korea, 2006.

[46] Tsuyoshi Tasaki and Fumio Ozaki, "Obstacle Classification and Location by Using a Mobile Omnidirectional Camera Based on Tracked Floor Boundary Points," *The 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5222-5227, October 11-15 2009.

[47] Kensaku Saitoh, Takashi Machida, Kiyoshi Kiyokawa, and Haruo Takemura, "A 2D-3D Integrated Interface for Mobile Robot Control Using

Omnidirectional Images and 3D Geometric Models," *ASIAGRAPH 2007 in Tokyo*, vol. 1, no. 2, pp. 35-40, October 2007.

[48] Davide Scaramuzza and Roland Siegwart, "Appearance-Guided Monocular Omnidirectional Visual Odometry for Outdoor Ground Vehicles," *IEEE TRANSACTIONS ON ROBOTICS*, vol. 24, no. 5, OCTOBER 2008.

[49] Niramon Ruangpayoongsak, "Development of autonomous features and indoor localization techniques for car-like mobile robots," *PhD thesis, University of Siegen*.

[50] Fernando Samaniego González, "Remote Control of Mobile Robots using LabVIEW," University of Siegen, Siegen, Master Thesis 2008.

[51] Guillermo Venero Gómez, "The Remote Operation of the Autonomous Mobile Robot," University of Siegen, Siegen, Master Thesis 2009.

[52] Wikibooks. Robotics/Types of Robots/Wheeled. [Online]. http://en.wikibooks.org/wiki/Robotics/Types_of_Robots/Wheeled#3-wheeled_vehicles

[53] Berthold K.P. Horn, Hugh M. Hilden, and Shahriar Negahdaripour, "Closed-form solution of absolute orientation using orthonormal matrices," *Journal of the optical Society of America A*, vol. 5, p. 1127, July 1988.

[54] Jr. and Benjamin Lipchak Richard S. Wright, *OpenGL SuperBible, Third Edition*.: Sams Publishing, July 2004.

[55] NeHe Productions. [Online]. http://nehe.gamedev.net/

[56] PMD Technologies. [Online]. http://www.pmdtec.com/

[57] H. Kraft et al., "3D-camera of hight 3D-frame rate, depth-resolution and background light elimination based on improved PMD (photonic mixer device)-technologies," *OPTO*, 2004.

[58] R. Lange, "3D time-of-flight distance measurement with custom solid state image sensors in CMOS/CCD-technology," University of Siegen, Siegen, PhD dissertation 2000.

[59] Z. Xu, R. Schwarte, H. Heinol, B. Buxbaum, and T. Ringbeck, "Smart pixel-photonic mixer device (PMD)," in *Proc. Int. Conf. on mechatron & Machine Vision*, 1998, pp. 259-264.

[60] T. Ringbeck, T. Möller, and B. Hagebeuker, "Multidimensional measurement by using 3-D PMD sensors," *Advances in Radio Science*, pp. 135-146, 2007.

[61] Matthias Wiedemann, Markus Sauer, Frauke Driewer, and Klaus Schilling, "Analysis and characterization of the PMD camera for application in mobile robotics," *The International Federation of Automatic Control*, July 2008.

[62] Christian Beder, Bogumil Bartczak, and Reinhard Koch, "A Comparison of PMD-Cameras and Stereo-Vision for the Task of Surface Reconstruction using Patchlets," *Computer Science Department, University of Kiel, Germany*.

[63] R. Reulke, "Combination of Distance Data with High Resolution Images".

[64] Marvin Lindner, Andreas Kolb, and Klaus Hartmann, "Data-Fusion of PMD-Based Distance-Information and High-Resolution RGB-Images," *Computer Graphics Group, Institute for Vision and Graphics, University of Siegen, Germany*.

[65] Benjamin Huhle, Timo Schairer, Philipp Jenke, and Wolfgang Straßer, "Robust Non-Local Denoising of Colored Depth Data," in *Computer Vision and Pattern Recognition Workshops, 2008*, 2008.

[66] Benjamin Huhle, Philipp Jenke, and Wolfgang Straßer, "On-the-Fly Scene Acquisition with a Handy Multisensor-System," WSI/GRIS, University of Tuebingen, Tuebingen,.

[67] Neven Santrac, Gerald Friedland, and Raul Rojas, "High Resolution Segmentation with a Time-of-Flight 3D-Camera using the Example of a Lecture Scene," Department of Computer Science, Freie Universitaet Berlin, Berlin, Technical Report 2006.

[68] Stefan Vacek, Thomas Schamm, Joachim Schröder, and Rüdiger Dillmann, "Collision Avoidance for Cognitive Automobiles Using a 3D PMD Camera," *Institute of Computer Science and Engineering, University Karlsruhe, Germany*.

[69] Thorsten Ringbeck and Bianca Hagebeuker, "A 3D Time of Flight Camera for Object Detection," *Optical 3-D Measurement Techniques*, 2007.

[70] Anil K. Jain, *Fundamentals of Digital Image Processing*, Colleen Brosnan, Ed. University of California, Davis, United States of America: Prentice-Hall, Inc., 1989.

[71] Jae S. Lim, *Two-Dimensional Signal and Image processing*, Raein Maes, Ed. New Jersey, United States of America: Prentice-Hall, Inc., 1990.

[72] Sébastien Piérard, Jérôme Leens, and Marc Van Droogenbroeck, "REAL-TIME PROCESSING OF DEPTH AND COLOR VIDEO STREAMS TO IMPROVE THE RELIABILITY OF DEPTH MAPS," in *Proceedings of 3D Stereo MEDIA*, Lìge, Belgium, November 2009.

[73] Zhengyou Zhang, "A Flexible New Technique for Camera Calibration," Microsoft Research, Microsoft Corporation, One Microsoft Way, Redmond, WA, Technical Report MSR-TR-98-71, 1998.

[74] Antoine Mischler, "3D Reconstruction from Depth and Stereo Images for Augmented Reality Application," Technische Universität Berlin, Berlin, Master Thesis 2007.

[75] Jiang Zhou, "Tutorial - Camera calibration," 2010.

[76] Gary Bradski and Adrian Kaehler, *Learning OpenCV*, 1st ed., Mike Loukides, Ed. CA 95472, united States of America: O'Reilly Media, Inc., 2008.

[77] Jean-Yves Bouguet. (2008, 2 June) Camera Calibration Toolbox for Matlab. [Online]. http://www.vision.caltech.edu/bouguetj/calib_doc/index.html

[78] Janne Heikkilä and Olli Silvén, "A Four-step Camera Calibration Procedure with Implicit Image Correction," in *Proceedings of the 1997 Conference on Computer Vision and Pattern Recognition*, 1997, p. 1106.

[79] Jean-Yves Bouguet, "Visual methods for three-dimensional modeling," California Institue of Technology, Pasadena, California, PhD Thesis 1999.

[80] Benjamin Huhle, Sven Fleck, and Andreas Schilling, "Integrating 3D Time-of-Flight Camera Data and High Resolution Images for 3DTV Applications," University of Tübingen WSI/GRIS, Tübingen,.

[81] Hubert Roth et al., "The 3D Mapping Preparation using 2D/3D cameras for Mobile Robot Control," in *The Scientific-theoretical journal "Artificial Intelligence"*, State University of Informatics and Artificail Intelligence, Ukraine., 2008.

[82] Hubert Roth Chanin Joochim, "Development of a 3D Mapping using 2D/3D Sensors for Mobile Robot Locomotion," in *The 2008 IEEE International Conference of Technologies for Practical Robot Applications, TePRA*, Massachusetts, USA, 10-11 November 2008.

[83] Rainer Lienhart and Jochen Maydt, "An Extended Set of Haar-like Features for Rapid Object Detection," Intel Labs, Intel Corporation, Santa Clara, CA 95052 USA,.

[84] Paul Viola and Michael Jones, "Rapid Object Detection using a Boosted Cascade of Simple Features.Conference on Computer Vision and Pattern Recognition," *Conference on Computer Vision and Pattern Recognition*, 2001.

[85] Rainer Lienhart, Alexander Kuranov, and Vadim Pisarevsky, "Empirical Analysis of Detection Cascades of Boosted Classifiers for Rapid Object Detection," in *DAGM'03, 25th Pattern Recognition Symposium*, Madgeburg, Germany, Sep, 2003, pp. 297-304.

[86] José Barreto, Paulo Menezes, and Jorge Dias, "Human-Robot Interaction based on Haar-like Features and Eigenfaces," in *in Proc. 2004 IEEE Int. Conf. on Robotic and Automation* , 2004, pp. 1888-1893.

[87] C. Joochim, C. Netramai, and H. Roth, "Coordination of SLAM and Artificial Landmark Recognition using 2D/3D Sensors for Mobile Robot Vision," in *The Tenth International Conference on Pattern Recognition and Information Processing (PRIP'2009)*, Minsk, Belarus, May 19-21,2009.

[88] Peter Biber and Wolfgang Straßer, "nScan-Matching: Simultaneous Matching of Multiple Scans and Application to SLAM," in *Proceedings of the 2006 IEEE International Conference on Robotics and Automation*, Orlando, Florida, 2006.

[89] Huijing Zhao and Ryosuke Shibasaki, "Reconstructing Textured CAD Model of Urban Environment Using Vehicle-Borne Laser Range Scanners and Line Cameras," in *ICVS 2001*, Heidelberg, 2001, pp. 284-297.

[90] F. Lu and E. Milios, "Globally consistent range scan alignment for environment mapping," in *Autonomous Robots*, 1997.

[91] Jianbo Shi and Carlo Tomasi, "Good Features to Track," *IEEE Conference on Computer Vision and Pattern Recognition (CVPR94) Seattle*, June 1994.

[92] A. Bruhn, J. Weickert, and C. Schnorr, "Lucas/Kanade Meets Horn/Schunck: Combining Local and Global Optic Flow Methods," *Internatinal Journal of Computer Vision*, pp. 211-231, 2005.

[93] Bruce D. Lucas, "An Iterative Image Registration Technique with an Application to Stereo Vision," *Proceedings DARPA Image Understanding Workshop*, pp. 121-130, April 1981.

[94] Shinji Umeyama, "Least-Square Estimation of Transformation Parameters Between Two Point Patterns," *IEEE Transactions on pattern analysis and Machine intelligence*, vol. 13, no. 4, April 1991.

[95] Gaojin Wen, Zhaoqi Wang, Shihong Xia, and Dengming Zhu, "Least-squares fitting of multiple M-dimensional point sets," *The Visual Computer: International Journal of Computer Graphics*, vol. 22, pp. 387-398, June 2006.

[96] Paul J. Besl and Neil D. McKay, "A Method for Registration of 3-D Shapes," no. Vol. 14, No.2 February, 1992.

[97] Olga Sorkine, "Least-Square Rigid Motion Using SVD," New York, 26 February 2009.

[98] C. Netramai, O. Melnychuk, C. Joochim, and H. Roth, "Combining PMD and Stereo camera for Motion Estimation of a Mobile Robot," in *The 17th IFAC (International Federation of Automatic Control) World Congress, International conference*, South,Korea, 6 - 11 July 2008.

[99] C. Netramai, O. Melnychuk, C. Joochim, and H. Roth, "Motion Estimation of a Mobile Robot using different types of 3D Sensors," in *The international conference, ICAS2008 (The Fourth International Conference on Autonomic and Autonomous System)*, Gosier, Guadeloupe, March 16-21 2008.

[100] C. Joochim and H. Roth, "The Indoor SLAM using Multiple Three Dimension Sensors Integration," in *The IEEE 5th International Workshop on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS'2009)*, Cosenza, Italy, 2009.

[101] Chanin Joochim and Hubert Roth, "Mobile Robot Exploration Based On Three Dimension Cameras Acquisition," in *The Second IFAC Symposium on Telematics Applications (TA 2010)*, Timisoara, Romania, 2010.

[102] OpenCV Wiki. [Online]. http://opencv.willowgarage.com/wiki/

[103] Pierre Alliez. Codeguru. [Online]. http://www.codeguru.com/Cpp/G-M/opengl/article.php/c2693

[104] Calling MATLAB from C and Fortran Programs. [Online]. http://matlab.izmiran.ru/help/techdoc/matlab_external/ch_java.html