國立臺灣大學電機資訊學院資訊工程學系
碩士論文

Department of Computer Science and Information Engineering
College of Electrical Engineering and Computer Science
National Taiwan University
Master Thesis

運用單具二維LIDAR在立體環境中定位與建地圖
3D Localization and Mapping Using One 2D LIDAR

竇菲
Dopfer Andreas

指導教授：王傑智 博士
Advisor: Chieh-Chih Wang, Ph.D.

中華民國 九十八 年 八 月
August, 2009

國立臺灣大學碩士學位論文
口試委員會審定書

運用單具二維 LIDAR 在立體環境中定位與建地圖

3D Localization and Mapping using one 2D LIDAR

本論文係寶菲君（學號 R96922144）在國立臺灣大學資訊工程學系完成之碩士學位論文，於民國 98 年 7 月 27 日承下列考試委員審查通過及口試及格，特此證明

口試委員：

Chieh-Chih Wang （指導教授）

Han-Pang Huang

Ta-Te Lin

Shih-Chung Kang

系 主 任 呂育道

Acknowledgements

WHEN I first came to Taiwan almost four years ago I would have never dreamed that I am going to spend such a large and important part of my life here. Now looking back I am very glad that I can say with confidence that the decision to stay in Taiwan and to pursue a Master degree at National Taiwan University was best for me. There are many people I would like to thank for making this thesis possible, for helping and supporting me along my way over the last years.

First I would like to thank Bob Wang for being a great advisor. He helped me with many great ideas and fruitful discussions, and I want to thank him as well for all his support and encouragement in countless academic as well as personal matters over the last two years. Much gratitude goes to all members of the Perception and Learning lab for allowing me to become one of them, despite the cultural differences and the language barrier. In particular I want to thank Jeff Chen, Der-Yeuan Yu for all their help in many occasions. Also I want to thank Casey, Any, Jimmy, Jim, Chen-Han, Yu-Chun, Zhen-Yu, Chung-Han, Shao-Chen, Hao-Hsueh, and Alan for being reliable and very helpful colleagues during my time in the lab. I also want to thank Kuo-Hwei, Stanley, Leo, Atwood, Wei-Chi as well as Yi-Kuang and Chi-Hao who are previous members of the lab and have been a great support as well.

I specially want to thank my friends Hao-Ju Pang, Kathy Ko and Thijs Velema for all their support, encouragement and help throughout my last years in Taiwan. My friend and roommate Christian Ivancsits also has to be mentioned. I want to thank him for many long and rich discussions about our research and robotics in general. Also want to thank my long-standing friend Michale Eckl who was a great support during countless coffee-breaks and long discussions.

Last but not least I have to thank my family for all the support they gave me through this period of my life. Even though I know they would love to have me around them, they still supported my decisions to study far away from home. I also want to mention all my friends in Austria who stayed close to me over the years, and welcome me back like I have never left whenever I get home.

Andreas Dopfer, Taipei, July 19th, 2009

摘要

在這篇論文中，我們提出一種只利用一台二維 LIDAR 讓機器人在三維空間中做定位和建立三維空間資訊的方法。我們同時利用事前的環境知識和機器人本人的運動來完成這件事。我們主要是達成一種在取得三維空間資訊的同時，也提供了比得上使用水平二維 LIDAR 定位效能的方法。

3D LOCALIZATION AND MAPPING USING ONE 2D LIDAR

Andreas Dopfer

Department of Computer Science and Information Engineering
National Taiwan University
Taipei, Taiwan

August 2009

Submitted in partial fulfilment of
the requirements for the degree of
Master of Science

Advisor:
Chieh-Chih Wang

Thesis Committee:
Han-Pang Huang
Ta-Te Lin
Shih-Chung Kang

ABSTRACT

MUCH work on localization and mapping using LIDAR has been done in mobile robotics. While earlier work was done only in the two dimensional domain, a recent shift towards three dimensional localization and mapping using laser rangefinder can be seen. Three dimensional representations allow a more accurate modeling of the real world, allowing more sophisticated path planning and leading to better obstacle avoidance. Also the performance of localization can be improved, and three dimensional data allows better object recognition than 2D data.

Techniques capturing 3D data involve either multiple 2D LIDARS, one 2D LIDAR that is nodded or rotated using an external actuator together with highly accurate orientation sensing and synchronization, or an integrated, expensive 3D scanning system. In this thesis we propose a technique to capture 3D data only using one 2D LIDAR. To do so the robots motion is utilized together with reasonable assumptions. It is assumed that the ground the robot is moving on is flat and visible in the scan, that the sensors height is known and that the environment has vertical structures.

First an initial calibration procedure using a camera together with the LIDAR is performed to reveal the extrinsic parameters between robot and the sensor. The localization problem is divided into two steps. The LIDARs sensing plane is tilted away from the robots direction of motion towards the floor (or another known flat structure in the environment). The detection of the floor allows to estimate the angular orientation of the sensor in two dimensions. Using these estimates the range data can be transformed, so that known methods to estimate the missing parameters of the full LIDAR pose can be adopted. Being able to accurately estimate the three dimensional displacement between two consecutive scans allows to build an accurate three dimensional map of the environment.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

Introduction

To allow a mobile robot safe and efficient operation within the real world, their perception needs to answer two fundamental questions. One of them is: "How does the world (around me) look like?". To interact with the physical world a robot needs knowledge about the environment. In which form this information is required relates strongly to the robots purpose. For a robot that moves in a static environment it might be sufficient to know which areas it can traverse, but for a robot that is expected to do more sophisticated tasks such as interacting with the environment a higher level representation of the environment is required. The problem of deriving a world representation from sensors readings is called mapping.

The other question to answer is: "Where am I?". Just to know how the environment looks like is not enough. A robot also needs to know how its location in this environment and how it is located with respect to a certain object. The problem of finding the robots location is called localization. The sensors used for both can be cameras, sonars, laser rangefinder or any other sensor that can sense relevant information of the environment. In this thesis we focus on Light Detection and Ranging (LIDAR) sensors which use a laser beam to sense the distance to objects. The characteristics of this class of sensors will be discussed in section 3.1.

Localization

When talking about localization we can distinguish two different ways. One is to localize the robot given that a map of the environment is given. If the initial location in the map is known the task gets easier, since the region within which the robot can be is

bounded. If no initial location is given the problem get harder because more possible locations where the robot can be exist. The other way that is also named as localization is to keep track of the robot's subsequent movement. Rather than using a map of the whole environment, the previous sensor reading is used together with the current reading to estimate the change of position. Adding up these position changes localizes the robot with respect to its starting position. Following the taxonomy of (Thrun et al., 2005) this category of localization problems is called position tracking.

Mapping

Mapping is the problem of deriving a world representation given sensor measurements. Many possible map representations have been discussed like grid based approaches, topological ones or direct approaches that use the raw measurements in some sense. Given the position from which the sensor reading was recorded the reading can be integrated in the map in several ways. Caution has to be taken since in real world applications neither the sensor reading nor the position will be perfectly known.

Thesis objective

This thesis describes a method to localize a robot in 3D and generate a map of its environment using only one 2D LIDAR. To do so prior knowledge about the environment is utilized together with the robots motion. This work aims to reach comparable localization performance to 2D method while being able to capture richer 3D information.

Overview

Chapter 2 of this thesis will give an overview about existing mapping and localization techniques and will discuss different 3D mapping methods. Chapter 3 will introduce different methods and algorithms that are used to complete the work of this thesis. Algorithms to process the raw LIDAR data will be presented together with a calibration method to find the extrinsic parameters between a camera and a laser rangefinder which will be used to calibrate the LIDAR position and orientation. Chapter 4 will introduce our approach for 3D localization and mapping using one 2D LIDAR while chapter 5 will show detailed results. The last chapter is going to conclude this thesis and point out future possible research directions.

CHAPTER 2

Related Work

THE problem of modeling physical structures has been long studied in different research areas as geoscience (Schulz, 2007), photogrammetry or computer vision. Recently mapping using LIDARs has become popular in robotics, popular to the extend that nowadays most robots that are intended to move in either indoor or outdoor environments use one. Often no prior map is given, so the robot needs to build one in order to navigate in the environment and interact with it.

2.1. Mapping

First work on autonomous map building using a grid based approach was done by (Elfes, 1987) who proposed using occupancy grid maps for mapping. Due to limits of sonars as range sensors and the lack of computational power at this time his maps did not extend the size of a few thousand grid cells. The work on occupancy grids was later extended to 3D using Stereo vision by (Martin & Moravec, 1996). Various versions of grid based mapping are presently widely used because they are intuitive to use and easy to implement. Higher level features like loop closing can only be achieved with extra effort.

Feature based approaches extract, like the name suggests, features from the raw scan and then keep track of them and the robots location by using an Extended Kalman Filter (EKF). First described by (Cheeseman & Smith, 1986) EKF based SLAM allows loop closure, but introduces a data association problem and has to deal with the burden of the quadratic growth of the covariance matrix used by the EKF. Much work has been done and is done in this area since the seminal paper. (Thrun et al., 2002) introduced an Information Filter based (IF) technique that addresses this issue while (Burgard et al., 1999) uses a hybrid

approach generating local submaps using occupancy grids and then using Expectation Maximization (EM) Algorithm to form the global maps out of the submaps.

Direct methods use the raw measurements returned by the sensor to estimate the robots pose and represent the map. To find the location of the mobile robot the spatial constraints between two consequent scans are estimated. Assuming that parts of the two scans show the same environment, algorithms like the Iterative Closest Point Algorithm (ICP) can be used to match the scans. ICP was introduced by (Besl & McKay, Besl & McKay) and is heavily used in 2D as well as 3D. Various improvements to this algorithm have been proposed, (Rusinkiewicz & Levoy, 2001) gives an excellent summary describing the effects on performance and accuracy of the different variations. The map representation is typical a set of points, or a list of scans together with the corresponding robot pose in which the scan was taken. Parts of this thesis work use direct methods to estimate the robots pose and to describe the map.

2.2. Mapping in 3D

While most work mentioned before was done in 2D, the methods used can be extended to the three dimensional world. 3D EKF based techniques are often used in visual methods while range data based ones often use direct or gridmap based approaches.

Few sensors exist that can capture 3D data of its environment - one kind of these sensors are Time of Flight (TOF) cameras which use an array of flashing IR LEDs to get depth information. Their range is limited to a few meters, and the data returned is rather noisy.

Several techniques for acquiring three-dimensional data using 2D range scanners mounted on mobile robots have been proposed. One approach is to mount two scanners with different orientations as done by (Thrun et al., 2000) and (Thrun et al., 2003) another one to use one scanner and pan, tilt or rotate it (Harrison & Newman, 2008).

(Wulf & Wagner, 2003) shows the influence of different ways of tilting/rotating one scanner in space on the returned data. Drawback of these methods is that they require extra effort to precisely synchronize the tilting/rotating actuator with the scanner in order to determine the orientation in which each scan was taken. In case the robot is moving during the scanning the robots trajectory also needs to be taken into account. Several robotic systems avoid that problem by stopping for the the scanning. (Nuechter et al., 2005) and (Ryde

& Hu, 2007) use this 'stop-and-go' method for capturing data. Additional sensors as an Odometer or an IMU can be used to estimate the path traveled during the scan. (Harrison & Newman, 2008) describes in detail methods for the synchronization of scanner/tilting platform, as well as optimisation techniques using prior knowledge about the structure of the environment (for example: walls are known to be vertical) to allow capturing more accurate 3D information using a tilting scanner under general vehicle motion.

To capture 3D data with two LIDARs one is oriented with its scanning plane parallel to the plane in which the robot moves, while the other one is pointing towards the objects of interest having another orientation than the first one. Since the extrinsic parameters between the two scanners are fixed one can be used to determine the robots position in 2D assuming movement only in two dimensions, as the other 'sweeps' over the environment while passing thus generating the 3D data. (Frueh & Zakhor, 2004) used this approach to generate a city scale 3D model using 2 laser scanners mounted on a car while driving through the city under normal traffic conditions. These type of approaches have the drawback that the 3D information is captured at the moment the robot is at a location, not prior to that therefore the 3D data cant be used for navigation or obstacle avoidance purposes.

CHAPTER 3

Foundations

THIS chapter will introduce basic methods that are applied within this thesis. After a short description of the main sensor utilized for this work, techniques that are used to process raw LIDAR data are introduced. Then the ICP algorithm will be discussed which is employed to match two consecutive scans. Finally a method to find the extrinsic parameters between a camera and a laser scanner will be introduced.

3.1. LIDAR

LIDAR (Light Detection and Ranging) is the notation for a class of optical sensors that utilize light to measure distance to a remote target. While large high energy versions of this technology are used in meteorology and astrophysics there is a large number of small, portable sensing devices that can be used in industrial applications, or mounted on mobile robots. We will introduce the latter type of this class of sensors.

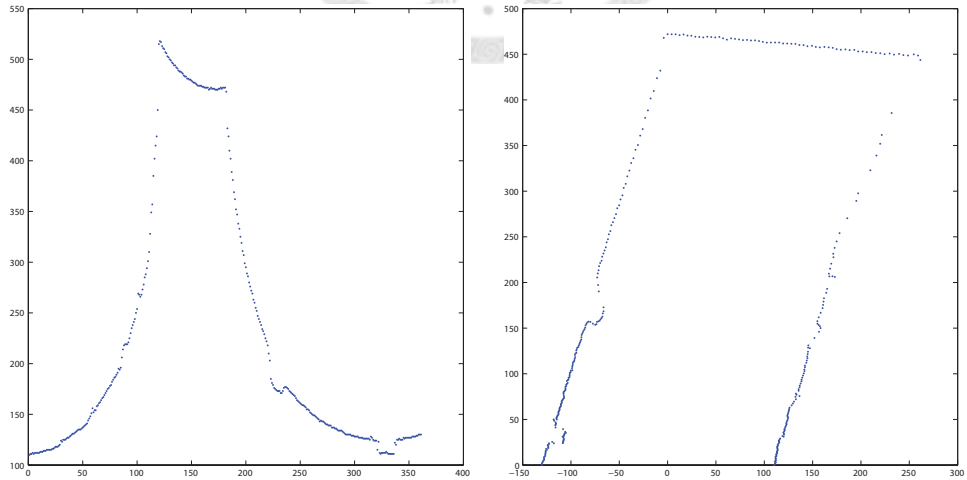
(a) Hokuyo laser scanner (b) Sick LMS 291 (c) SICK S200

Figure 3.1. Examples of various LIDAR sensors

A LIDAR device sends out a focused beam of light and measures the properties of the scattered light returned by an object to the scanner. The properties of the returned light, like time difference between sending out the beam and its return or phase shift, are then used to estimate the distance to the target. For the sensors used in mobile robotics the light source used is an eye-safe, invisible laser. To allow sensing the distance in more than one direction a rotating mirror is used to sweep the beam in a 2D plane over the environment. Figure 3.1 shows various models of laser range finders.

One typical characteristic for these sensors is a maximum range between 4 and 100 meters with an accuracy of a few centimeters or below. The amount of data points recorded varies depending on the model within a range of 1000 to 15000 measurements/second. The angular spacing between the measurements within the scanning plane ranges from 1° to 0.25°.

Figure 3.1(a) shows a Hokuyo LIDAR which has a maximum range of 4 meters, an angular resolution of 360/1024 degree and an opening angle of 240 deg. It is one of the smallest rangefinders on the market with a 4x4 centimeter footprint, a height of 7 centimeters and a weight of 160 grams. Figure 3.1(b) shows the widely used SICK LMS 291. Experiments in this thesis were performed using this sensor. Figure 3.1(c) shows a SICK S200.



(a) raw scan data with the range in centimeters on the vertical axis and the beam index (equivalent to 0.5 deg) on the horizontal axis

(b) same scan as in (a) converted to carthesisan coordinates. Both axis in centimeters.

Figure 3.2. Example of scans taken with a SICK LMS 291.

3.2. Raw Scan Processing

LIDAR measurements return the distance to an object together with the angle under which the measurement was made. This corresponds to a two-dimensional polar coordinate system where a point is defined by $(\phi_i, dist_i)$. These coordinates can easily be transferred to a Cartesian coordinate system (x_i, y_i).

To educe higher level information from a scan it is desirable to group the points to segments. While distance criteria in polar or cartesian coordinates allow basic segmentation, line segmentation is particularly helpful since in man-made environments most structures tend to have planar surfaces which are represented as lines in a scan.

3.2.1. Linear Least Square Line fitting

Given a set of n points defined by pairs (x_i, y_i) of Cartesian coordinates the line that minimizes the squared distance to all points can be calculated in a closed form

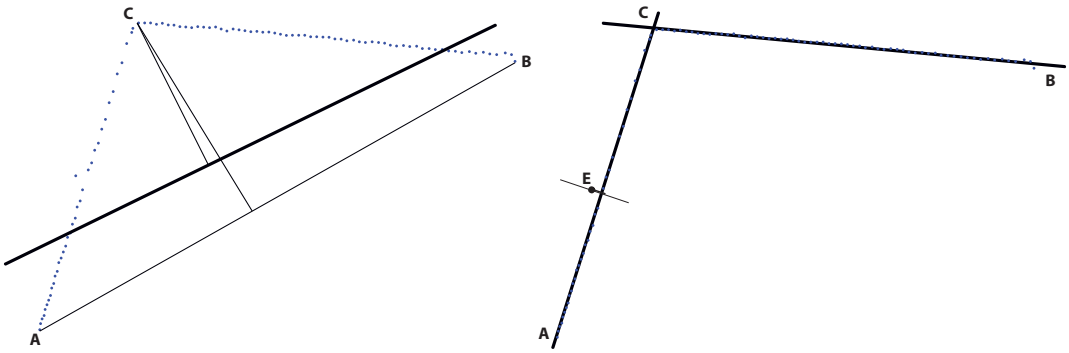
$$\tan 2\varphi = \frac{-2\sum_i (\bar{x} - x_i)(\bar{y} - y_i)}{\sum_i [(\bar{y} - y_i)^2 - (\bar{x} - x_i)^2]} \tag{3.1}$$

$$r = \bar{x}\cos(\varphi) + \bar{y}\sin(\varphi) \tag{3.2}$$

where $(\bar{x}, \bar{y})$ denotes the centroid of all points given by $\bar{x} = \sum_i x_i$ and $\bar{y} = \sum_i y_i$. φ is the angle of the normal and r the normal distance to the origin. Unfortunately a scan or even a part of an already segmented scan can be composed of multiple lines. To deal with this issue the following algorithm is applied.

3.2.2. Line splitting

The basic idea of the 'split and merge' algorithm is to recursively subdivide the point set into segments that can be better approximated by a line. Given a segment that should be split into lines, a line is fit through all points using Equations 3.1 and 3.2. Then the point with the maximum distance to the line is calculated. If the distance is below a defined threshold the fitted line is returned. Otherwise a new line defined by the first and last point in the set is defined. The point with the largest distance to this line is found and selected as splitting point, separating the point set into one subset consisting of all points from the first point to the splitting point (including) and a second subset containing the rest of the points.



(a) First step of the line splitting

(b) Two Lines were found

Figure 3.3. First the thick line in (a) is calculated using Equations 3.1 and 3.2. Point **C** is found as point with the largest normal distance to the line. The distance is above a defined treshold, so a new line **AB** is defined and the point with the largest distance is found again (**C**). The pointset is splitted into two sets. Again lines are fitted through both sets using the Linear Least Square method. The distance of **E** to the fitted line is below the defined treshold, as well as all points of the second set are within the treshold. Two lines were found.

The algorithm is performed subsequently on the subsets until no splitting occurs, meaning that all points are located within a certain threshold to a line. Figure 3.3 illustrates the procedure and ListingAlgorithm 1 summarizes the line splitting algorithm.

Algorithm 1 Line splitting algorithm

1: given a set of n points (x_i, y_i)
2: calculate line L_{LSq} (r, φ) using Equations 3.1, 3.2
3: find max distance d_{LSq} line L_{LSq} - point (x_i, y_i)
4: **if** distance $d_{LSq} >$ threshold **then**
5: define line by first and last point of the set L_{FL}
6: find point p_{split} with max distance from line L_{FL}
7: split set into $p_1 \dots p_{split}$, $p_{(split+1)} \cdots p_n$
8: start algorithm with both new pointsets
9: **end if**
10: **return** all lines $\leftarrow$ line

3.3. Iterative Closest Point (ICP) Algorithm

This algorithm was proposed by (Besl & McKay, Besl & McKay) to match two point sets. It aims to find the transformation $\mathbf{T} = (T, R)$ from a point set $\mathbf{B} = \{p_i^B ... p_n^B\}$ to another set $\mathbf{A} = \{p_i^A ... p_m^A\}$. The ICP algorithm uses the raw data without the need to extract corresponding features which makes it robust and easy to use. While in this thesis the algorithm is employed in 2D it works as well in 3D or higher dimensional data.

First for each point p^A in the current point set $\mathbf{A}$ the closest point p^B from the point set $\mathbf{B}$ is computed. This is a computationally demanding step requiring N comparisons with M points. Where the point set $\mathbf{A}$ consists of N points and $\mathbf{B}$ of M. Various strategies can be applied like kD-trees or sub sampling to speed up this step.

Given the two point sets $\mathbf{A}$ and $\mathbf{B}$ we consider the points p_i^A and p_i^B as a corresponding pair. We then aim to find the rotation and translation $\mathbf{T} = (R, T)$ that minimizes the following function

$$e(R,T) := \frac{1}{N}\sum_{i=1}^{N} \parallel p_i^B - (Rp_i^A + T) \parallel^2 \tag{3.3}$$

Minimizing this function for the 2D case leads to the following closed form solution for $\mathbf{T} = (\varphi, T_x, T_y)$. This solution follows (Lu & Milios, 1997).

$$\begin{aligned} \varphi &= \arctan \frac{\Sigma_{A_yB_x} - \Sigma_{A_xB_y}}{\Sigma_{A_xB_x} + \Sigma_{A_yB_y}} \\ T_x &= \overline{p}^{A_x} - (\overline{p}^{B_x}\cos\varphi - \overline{p}^{B_y}\sin\varphi) \\ T_y &= \overline{p}^{B_y} - (\overline{p}^{A_x}\cos\varphi + \overline{p}^{A_y}\cos\varphi) \end{aligned} \tag{3.4}$$

where

$$\overline{p}^{A_x} = \frac{1}{N}\sum_{i=1}^{N} p_i^{A_x}, \quad \overline{p}^{A_y} = \frac{1}{N}\sum_{i=1}^{N} p_i^{A_y}, \quad \overline{p}^{B_x} = \frac{1}{N}\sum_{i=1}^{N} p_i^{B_x}, \quad \overline{p}^{B_y} = \frac{1}{N}\sum_{i=1}^{N} p_i^{B_y}$$

and

$$\begin{aligned} \Sigma_{A_yB_x} &= \sum_{i=1}^{N}(p_i^{A_y} - \overline{p}^{A_y})(p_i^{B_x} - \overline{p}^{B_x}) \quad , \quad \Sigma_{A_xB_y} = \sum_{i=1}^{N}(p_i^{A_x} - \overline{p}^{A_x})(p_i^{B_y} - \overline{p}^{B_y}) \\ \Sigma_{A_xB_x} &= \sum_{i=1}^{N}(p_i^{A_x} - \overline{p}^{A_x})(p_i^{B_x} - \overline{p}^{B_x}) \quad , \quad \Sigma_{A_yB_y} = \sum_{i=1}^{N}(p_i^{A_y} - \overline{p}^{A_y})(p_i^{B_y} - \overline{p}^{B_y}). \end{aligned}$$

$p_i^{A_x}$ denotes the x-coordinate if the ith point from set **A** while $p_i^{B_y}$ denotes the y-coordinate if the ith point from set **B**.

Different ways exists to calculate **T**, as they all minimize equation 3.3. (Besl & McKay, Besl & McKay) gave a formal proof that the ICP algorithm converges for a local optimum. It is also mentioned that a good initial condition increases the chance that the algorithm converges to the global optimum. We will discuss how we get a good initial guess in Section **??**.

The complete ICP algorithm is summarized in Algorithm 2

Algorithm 2 Iterative Closest Point algorithm (ICP)

given two pointsets $\mathbf{A} = \{p_1^A ... p_n^A\}$ and $\mathbf{B} = \{p_1^B ... p_m^B\}$
$\mathbf{T} \leftarrow 0$
score $\leftarrow \infty$
while score $>$ iterationThreshold & iterations $<$ maxIterations **do**
  transform **A** using $\Delta\mathbf{T}$
  find n closest point pairs $< p_i^A, p_j^B >$
  calculate $\Delta\mathbf{T}$ using 3.4
  $\mathbf{T} \leftarrow \mathbf{T} + \Delta\mathbf{T}$
  calculate score as $f(\Delta\mathbf{T})$
  iterations ++
end while
return Transformation **T**

3.3.1. Improvements over the Naive Implementation

Various improvements and variants of this algorithm have been proposed since the seminal paper. They increase the robustness as well as the speed of ICP. (Rusinkiewicz & Levoy, 2001) gives an excellent summary over various possible changes to the naive implementation. We will introduce the ones relevant to our 2D ICP implementation.

Spatially equal point selection. The idea behind this variation is that for each Object within the field of view of the laser scanner the number of points used for matching the scans should be related to the size of the object, not to the distance between the LIDAR and the object. If all points are used and they are weighted the same, a small object very close to the scanner might dominate the matching process simply because of the high number of measurements from this object.

To avoid this issue a minimum (euclidean) distance between points is defined. Points that are to close are removed from the point set. This procedure is performed before the finding of the closest-point-pairs on one of the point sets.

Outlier removal. Newly detected objects and erroneous measurements tend to occur at positions that are spatially far from previous measurements, while new measurements of already observed objects should be near to the previous measurements. Making use of this information the matching result can be improved by refusing pairs with a distance over a certain threshold. The distance used was the maximum distance possible between two scans which was derived from the robots maximum speed.

Further improvements. Other strategies include reduction of the point sets by randomly sub sampling or using other criteria to choose which points to use. This is not necessary in the 2D case since the number of points is low compared to the typical 3D case. The weighting of pairs is another possible improvement over the naive implementation. Possible criteria for the weighting is the point to point distance or the direction of the link in relation to the directions to the neighbouring points within the point sets.

3.4. Clustering

By clustering or cluster analysis we mean to divide a a set of observations into several subsets. Here it is not known how many clusters exist or if one exists at all, therefore methods like k-means which require prior knowledge about the number of subsets are not suitable.

We use a method similar to the QT (Quality Threshold) algorithm. Instead of defining the number of expected clusters we define a distance threshold within which all points of a cluster should be. For each point we build a candidate cluster by adding the closest point, then the next closest and so on until the next point added would exceed the threshold. Then the cluster with most points is saved as first true cluster. After all points from this cluster are removed from the point set the procedure is repeated until all points are assigned to clusters.

Since the data we work on is likely to have outliers and we are only interested in agglomerations of points - we do not aim to assign every piont to a cluster - we stop the algorithm once the number of data points in a new cluster falls below a certain threshold.

3.5. LIDAR calibration

n this section a method to calibrate the LIDARs pose is introduced. By calibration we mean to seek the LIDARs position and orientation in 3D. Finding this position and location *only* using its own data is hard, since the data returned is sparse and only taken within a 2D plane.

A two step method that utilizes an additional camera is proposed. In the first step the extrinsic parameters between the camera and LIDAR are estimated. The second step finds the cameras position with respect to a placed calibration board. Combining both results allows to estimate the LIDARs pose in 3D.

Figure 3.4 summarizes the approach.

Figure 3.4. Overview of the calibration procedure. First the extrinsic parameters between camera and LIDAR are estimated. Then the camera is located using a checkerboard. Finally the LIDARs pose is found by combining the two previous results.

3.5.1. Extrinsic Camera - LIDAR calibration

To fuse the data of two sensors the extrinsic parameters between them need to be known. Here we introduce a calibration method to estimate these parameters between a LIDAR and a camera. Several papers address this calibration problem, but only few address the problem using a 2D laser scanner with beams that are invisible to the camera. While (Pless & Zhang, 2004) use the distance between the laser points and the plane of a calibration object visible in the camera (Zhao et al., 2007) proposed the use of multiple

video streams and a 3D model built from LIDAR data in which manually identified features visible in both data sets are used to to perform calibration. The calibration approach described here follows the work of (Li & Liu, 2007) which establishes constraints between lines visible in the camera image and points in the laser scan for calibration purposes.

First we assume that the camera itself is calibrated, meaning its intrinsic parameters are known and the pictures used are corrected. A pinhole camera can be described by a projection from the world coordinates $P = [X, Y, Z]^T$ to the image coordinates $p = [u, v]^T$. This can be represented as follows

$$p \sim K(RP + T) \tag{3.5}$$

where K is the intrinsic camera matrix, R is an orthonormal 3x3 matrix representing the cameras orientation and T a vector representing the cameras position.

All laser points are recorded in a plane. A laser coordinate system is defined with its origin in the LIDAR's center - and without loss of generality the scan plane is defined as $Y = 0$. Let P_L be a point in the laser coordinate system, so the transformation between the camera and laser coordinate system can be described as follows

$$P_L = \Phi P + t \tag{3.6}$$

where Φ is a 3x3 matrix representing the cameras orientation relative to the lasers orientation and t a vector representing their relative spatial position. Using Equation 3.5 and 3.6 a point in the Laser coordinate system can be projected into the image plane.

Possible features for calibration would be points, lines and planes. Different features introduce different constraints between camera and laser data. Point features would lead to the strongest constraints, but since the light of the LIDAR is not visible in the camera image it is less feasible to use point features.

Line features are easily detectable in the camera image. In LIDAR data changes in depth are easily detectable. The edges of a calibration board are therefore visible as lines in the camera image and visible as depth changes in the LIDAR data. The calibration target is designed as a rectangular triangle as shown in Figure 3.7. The rectangular shape gives the strongest constraints (Li & Liu, 2007).

Several poses of the checkerboard visible in both sensors are recorded. After identifying the edges AB and AC from the calibration target as lines ab and ac in the cameras image

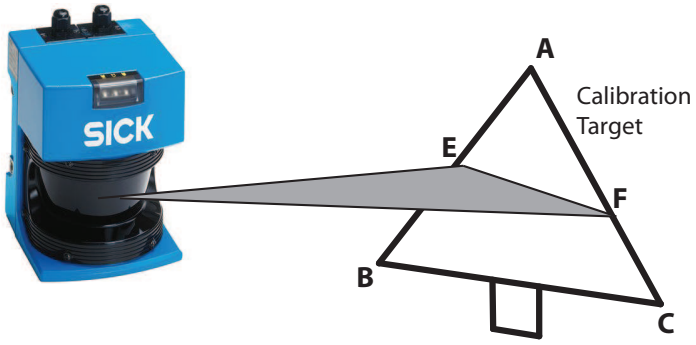
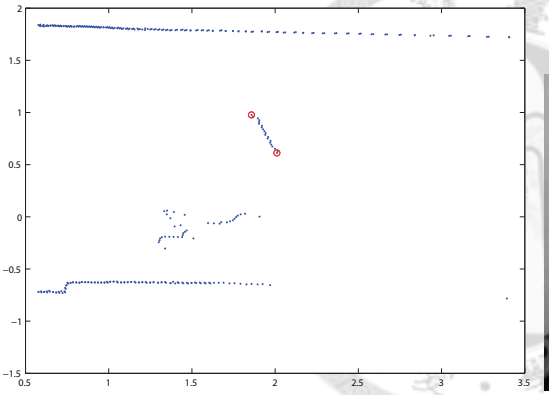
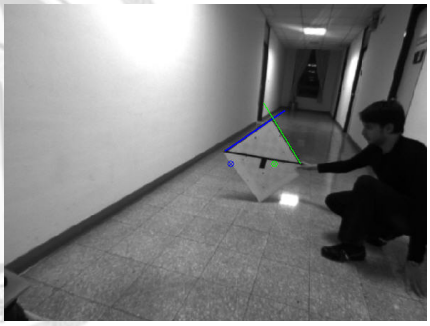


Figure 3.5. Calibration target intersecting with laser slice plane. Points E and F are visible in the scan and known to lie on the lines AB respectively AC which are visible in the camera.



(a) A scan with the points corresponding to the calibration boards edges marked in red circles



(b) A camera image with lines marked and the two laser points projected to the image using the initial guess for Φ and t. The object in the left lower corner of the camera image is the LIDAR sensor

Figure 3.6. Sample pair of camera image and laser scan used for calibration

plane we find the points E and F in the laser scan and project them to the camera plane as e and f using Equation 3.5 and 3.6. Figure 3.6 shows one pair of image and scan used for calibration. The distance between a point e and a line ab is defined by

$$dist(e,ab) = \frac{\| \vec{eb} \times \vec{ab} \|}{\| \vec{ab} \|}. \tag{3.7}$$

Using the distance between the lines and the projected points we can define the following error function that we minimize in order to find Φ and t.

$$\min \left\{ \sum_{i=1}^{N} (dist(e_i, a_i b_i) + dist(f_i, a_i c_i))^2 \right\}. \tag{3.8}$$

N is the number of corresponding camera images and laser scans. In order to calculate Φ and t we solve the nonlinear optimization problem using the Levenberg-Marquard algorithm. Algorithm 3 summarizes the proposed calibration method. Figure 3.7 shows the calibration results by projecting the Laser data onto the image.

Algorithm 3 Extrinsic Camera LIDAR calibration

1: given N measurement pairs from camera and LIDAR
2: set initial Φ and t by estimation or rough measurement
3: for each pair detected the two intersection points of the laser slice plane with the calibration targets edges and transform them to the cameras image plane
4: detect projections of the calibration targets edges to the image plane
5: calculate Φ and t by minimizing Equation 3.8 using the Levenberg-Marquard method
6: **return** Φ and t

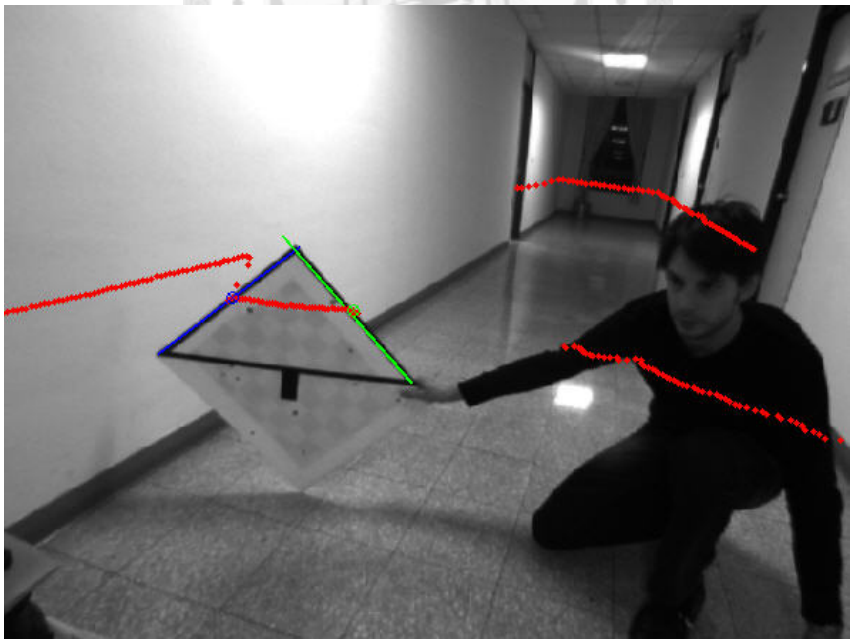


Figure 3.7. The camera image after the calibration procedure with the laser data projected on the image. The position of the laser scanner is visible in the lower left corner. The points projected on the authors head appear since the head is occluding the wall behind. The points actually hit the wall behind the head.

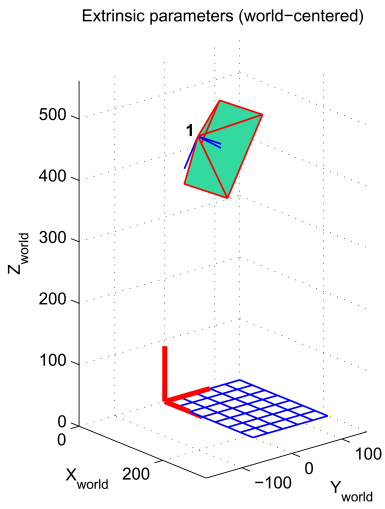
3.5.2. Calibrating the LIDAR's position

Estimating the LIDARs' position in space is a difficult task since the data returned from the sensor is relatively sparse. Calibration objects would need to be large bulky objects. To our best knowledge no method has been published that only uses the LIDAR data to uniquely calibrate the LIDARs position in 3D space. To avoid the hard problem of directly calibrating the LIDAR, we propose to utilize the camera for the calibration procedure.

In an intermediate step a checkerboard is used as calibration object to find the cameras position in space. By placing the checkerboard on a well defined location like the ground preferable with one orientation parallel to the an edge of the cameras image plane the parameters of interest (height, tilt and roll angle) can be estimated. Given the camera position and orientation is found the results of the camera-LIDAR calibration procedure introduced in the previous section can be used to estimate the LIDAR's position in space.

Figure 3.8(a) shows an example of a picture from a checkerboard with axis found. Figure 3.8(b) shows the estimated camera pose.

(a) Calibration pattern is placed on the ground to calibrate the LIDAR's pose with respect to the ground.



(b) Calibration results for the camera. Position 1 marks the camera position, z-axis is the hight above ground. All units in millimeter.

Figure 3.8. Calibrating camera with respect to the ground

CHAPTER 4

3D Localization and Mapping using one 2D LIDAR

In this chapter we will describe our approach to perform 3D mapping and localization using one 2D LIDAR. First, three assumptions for this approach are formulated and discussed. Then a two-step localization strategy is described in detail in two variants. One allows to perform the method online, the second one includes an offline optimization step.

4.1. Assumptions

The Assumptions made will be described and justified first, and later on, after describing our approach, we will discuss up to which extend they can be attenuated or violated without breaking the system.

ASSUMPTION 1. *The robot is moving on flat ground which is visible in the scan.*

For the case of a mobile robot operating in indoor environments, the first part of the assumption will hold in most cases. Man-made buildings generally tend to have a flat floor. Robots moving are under the influence of gravity so they will move within that floor-plane. Typicall wheeled robots are not able to climb stairs - so they stay on the ground. In urban environments where the robot moves on streets the assumption will locally hold in many cases, while downward or upward slopes of the street might violate it. For any off-road case this assumption will most likely not hold.

The second part of the assumption - that the floor is visible in the scan - can be easily achieved by mounting the sensor in that way on the robot. Tilting the LIDARs sensing plane sufficiently away from the direction of motion towards the floor will make sure the

floor is visible in the scan. Sufficiently meaning that after expected changes in the robots pitch and roll angle due the robots suspension or its elastic structure the floor is still visible.

ASSUMPTION 2. *The height of the Laser scanner is known.*

Given an initial calibration that reveals the sensors position in relation to a defined starting point that is located on the floor (as described in Section 3.5.2) the robots initial height is known. Unless in rough outdoor terrains where a robot is likely to move over uneven terrain a wheeled indoor robot is unlikely to change its height. For urban environments with the sensor on a car like robot the assumption will hold on flat terrain driving at constant speed. Rapid changes of speed are likely to lead to small height changes due to the suspension. These changes are bounded and could be estimated given a sophisticated motion model. For a legged robot the height is likely to change during walking, but given the control input together with the mechanical configuration it is possible to estimate the sensors height for each time.

ASSUMPTION 3. *The environment has* sufficient *vertical structures.*

This assumption is reasonable within most man-made structures as well as in urban areas since they are also man-made. Buildings tend to have vertical walls, supporting pillars, power poles or just poles for traffic signs are mostly vertical. Looking at a smaller scale in buildings we can also find many vertical structures - chairs or tables have vertical poles, closets or storage racks have vertical parts. Even trunks of a trees in nature tend to be vertical. In most environments a robot might operate in, this assumption will hold, only outdoor terrains moderately covered with vegetation will violate this assumption.

The meaning of *sufficient* will be explained later in detail. To illustrate, one defined, vertical, wall like structure properly visible from the scanner would be sufficient. Two visible vertical poles together with the ground (following Assumption 1) would also be sufficient.

4.2. Estimating the Orientation

Given the previously made assumptions together with an initial calibration, the segment of one LIDAR scan, that represents the ground, can be found and its characteristics allow to estimate the orientation of the LIDAR sensor with respect to the floor.

4.2.1. Identifying the ground line

Assuming the floor is flat, all scan points representing the ground will lie on a line. Applying the methods described Section 3.2 the scan is reduced to a set of lines represented by n pairs (φ_i, r_i) where φ is the angle of the normal and r is the normal distance of the line to the origin.

With the use of Assumption 1 and 2 the parameters of the line representing the ground solely depend on the LIDARs pitch and roll angle. The robots design can give bounds on these two angles. Using these thresholds together with the initial calibration the line representing the floor can be estimated.

The algorithm to find the ground line out of all line segments is listed as Algorithm 4.

Algorithm 4 ground line selection

score $\leftarrow -1$
for all line segments (r_i, φ_i) in scan **do**
  if r_i is within distanceBounds & φ_i is within orientationBounds **then**
    if previous ground line exists **then**
      distance score $\leftarrow f(r_i, r_{prev})$
      orientation score $\leftarrow f(\varphi_i, \varphi_{prev})$
    else
      distance score $\leftarrow f(r_i, r_{expected})$
      orientation score $\leftarrow f(\varphi_i, \varphi_{expected})$
    end if
    if distance score + orientation score $>$ score **then**
      score $\leftarrow$ distance score + orientation score
      ground line $\leftarrow [r_i, \varphi_i]$
    end if
  end if
end for
if score $= -1$ **then**
  return -1
else
  return groundline (r_{GL}, φ_{GL})
end if

Where $(r_{expected}, \phi_{expected})$ are the values obtained from the initial calibration. Figure 4.1 shows an example scan segmented with the result of a groundline detection.

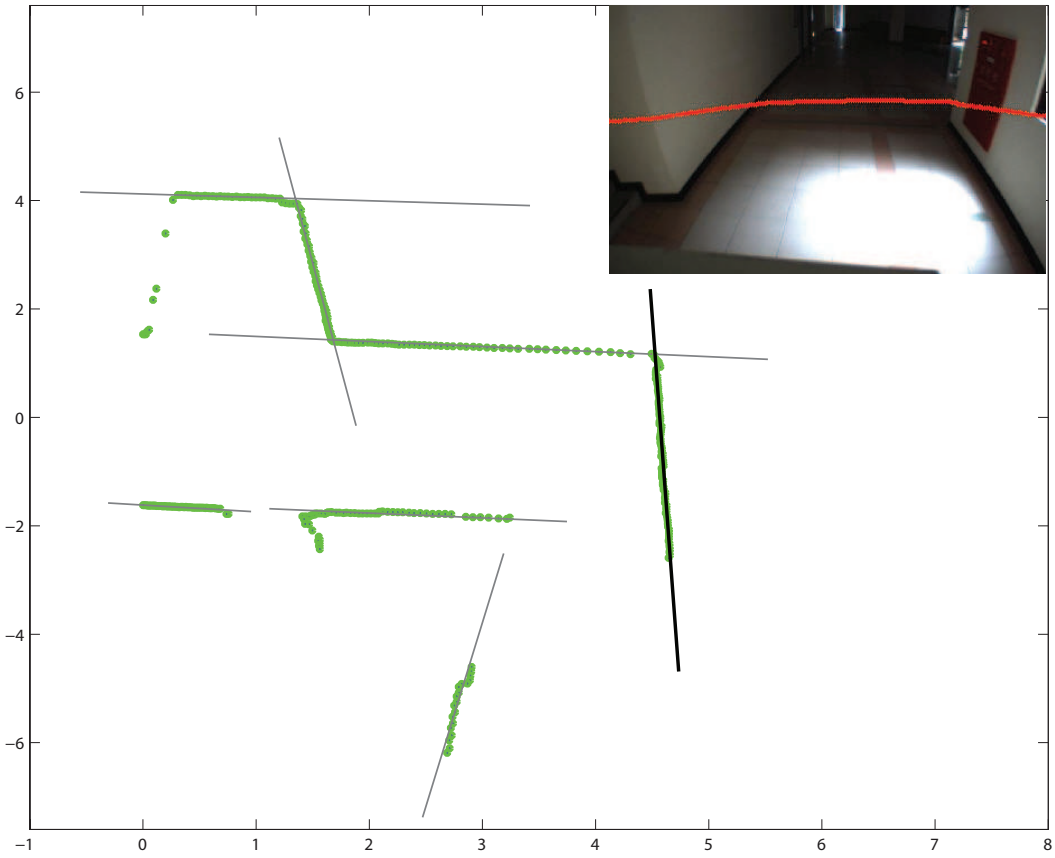


Figure 4.1. Segmented scan with detected line in gray and groundline estimate marked as a thick black line. Units on both axis in meters.

4.2.2. Selection Issues

Following the first assumption that the floor is visible in the scan the ground line can be etimated. Depending on the robots assembly, the bounds within the line is expected can be very tight. Experiments showed that often only one candidate exists within the bounds, making the selection process robust.

One possible issue is that no groundline exists at all. Even it violates the first assumption, the following localization steps can still be performed using the previous estimates instead of current ones. It clearly decreases the accuracy of the localization, but allows to

deal with periods without a detectable groundline. The case that there is no groundline can happen when the robot is moving towards a long wall with a moving direction normal to the wall. In all experiments concluded this happened only once.

Another possible approach is to keep all possible groundlines for each scan storing them all as valid hypotheses, and later do an offline optimization step picking the candidate for each scan that fits best in the groundplane fitted through all groundlines.

4.2.3. Calculating Pitch and Roll

A laser rangefinder measures the distance to a number of points. All the vectors from the center of the scanner to the measured point lie within a plane. Following Assumption 1 the ground is flat and is therefor also a plane. The intersection of two non-parallel planes is a line - here this line is exactly the groundline identified in the previous step. Given the parameters of this intersecting line, as well as the height of the scanner the orientation of the scanning plane can be calculated.

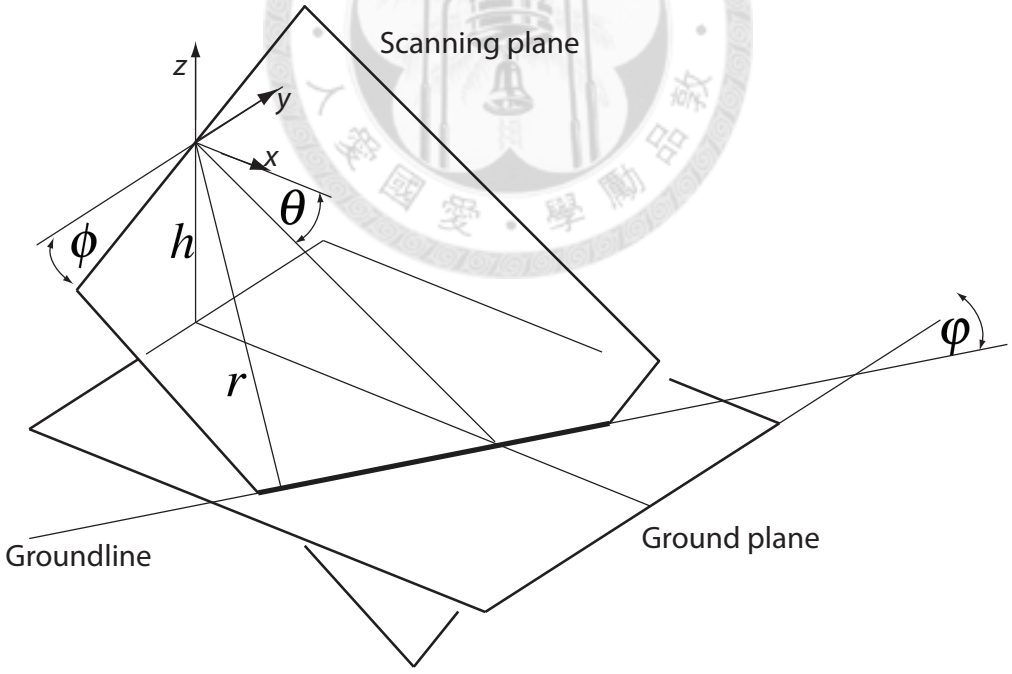


Figure 4.2. Pitch and roll angles

Let h be the height of the scanner above ground, r the normal distance between the intersecting line of the ground plane and the scanning plane and φ the orientation of the

intersecting line. Using fundamental geometry we can see from Figure 4.2 that the pitch angle θ is

$$\theta = \frac{\pi}{2} - \cos^{-1}\left(\frac{h}{r}\right) \tag{4.1}$$

the roll angle ϕ is:

$$\phi = \arctan\left(\frac{\sin\theta}{\tan\varphi}\right) \tag{4.2}$$

4.3. Estimating the Robot Motion

In the previous section we showed how roll and pitch orientations of the LIDAR can be estimated using assumptions about the environment. The LIDARs full pose is defined by 6 degrees of freedom (DOF) - with roll, pitch and height 3 out of these 6 are known. In this section we describe how we estimate the missing 3 parameters.

4.3.1. Scan Projection

Here we introduce two local coordinate systems. Let the robot's coordinate system be defined by three vectors: one normal to the ground plane, another is the projection of the 0° beam of the LIDAR on the ground plane and the third is normal to the previous two with its direction chosen so a that a right-handed or positive coordinate system is formed. The origin is chosen to be in the center of the laser rangefinder.

The second coordinate system is the LIDAR's own coordinate system, defined by a vector in the zero degree beam direction, one in the 90° direction and a third one that is normal to the previous two also forming a right-hand coordinate system.

The x axis is defined in both by the forward (0°) beam of the scanner while the z axis is defined in the robots coordinate system by the normal of the ground plane, and in the LIDARS coordinate system by the vector normal to the scanning plane. The rotation between the two coordinate systems can be defined by the previously calculated pitch and roll angles so that for $\theta = 0^\circ$ and $\phi = 0^\circ$ the two coordinate systems coincide.

A rotational matrix is defined that projects each point in the LIDAR's coordinate system p_i to the xy plane of the robots coordinate system

$$p_i^{xy} = R \cdot p_i \tag{4.3}$$

where R is composed from $R_x(\theta) \cdot R_y(\phi)$ and p_i^{xy} denotes a projected point. The new projected scan is parallel to the floor the robot is moving on. Vertical structures appear in the same x-y location independent of the height at which the scanner actually measured them. This allows to adopt known techniques to estimate the displacement within the groundplane between two consecutive scans given by the parameters x, y and ψ. The displacement of the two scans is equivalent to the robots movement.

4.3.2. Scan Matching Challenges

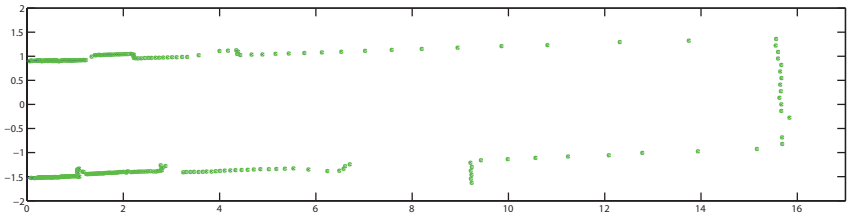
To estimate the displacement of two consecutive, projected scans the Iterative Closest Point algorithm (ICP) which was introduced in Section 3.3 is used. While the naive implementation works reasonably good on scans that are captured within the same plane in a static environment, the projected data of a tilted LIDAR introduces several challenges.

First the points representing the ground need to be removed from both point sets used for matching. This is done by calculating the line-point distance for each point to the ground line and rejecting the ones below a certain threshold. Removing these points reduces the number of points available for matching, but is necessary for the matching procedure.

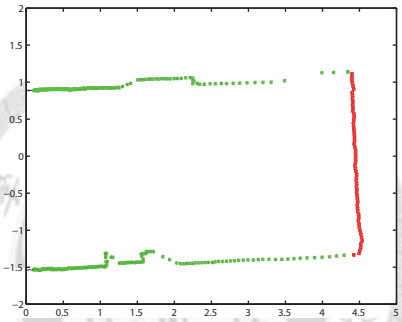
Tilting the LIDAR towards the ground reduces the maximum range in the scanners x-direction. While a commonly used SICK LMS 291 has 81 meters maximum range, the range in x-direction decreases down to a few meters when tilting the rangefinder. This makes it harder to estimate the forward movement of the robot. Figure 4.3 shows the differences between a horizontal and tilted scan that is reprojected of a hallway.

A scanner in horizontal orientation senses structures up to its maximum range within the full field of view which allows good matching, a tilted LIDAR sees less objects due the reduced forward range.

Strong features for estimating rotation and displacement in y-direction are usually available for a tilted scanner, while there is a lack of features that allow to estimate the forward movement. For example a long hallway without doors leads after removal of the groundpoints to two lines of points on both sides of the robot which allow excellent estimation of y and ψ but do not give any constraints on the forward direction.



(a) horizontal scan of a hallway



(b) The projected scan of the same scene captured with LIDAR tilted 20 degree towards the ground. Red points belong to the ground and are removed before matching.

Figure 4.3. Example of a horizontal and projected scan. After removing the points representing the ground in the tilted scan (right figure, red points) less features in moving direction of the robot are available.

Vertical structures in two tilted scans will allow proper matching, non-vertical structures might degrade the result and lead to *ambiguities*. Horizontal objects in indoor environments like tables or low walls outdoor will have constant projected locations for every point on their horizontal surface. Using them for matching would lead to a false result. Given enough vertical structures as postulated in Assumption 3 this issue can be avoided.

The reduced number of points in each scan and few features that establish constraints in moving direction together with the ambiguities introduced by non vertical structures can decrease the accuracy of ICP results. To deal with these issues, a good initial guess for the ICP algorithm is needed. To find this good initial guess a constant velocity motion model is used as described in the next section.

4.3.3. Constant Velocity Motion Model

To find a good initial condition for the scan matching procedure, we use the previous estimates for the robots velocity to project where the robot might be located at the time the next scan is taken.

$$v = \begin{pmatrix} \frac{T_x}{dt_p} \\ \frac{T_y}{dt_p} \\ \frac{\varphi}{dt_p} \end{pmatrix} \qquad T_i = v \cdot dt_n$$

$\mathbf{T} = (\varphi, T_x, T_y)$ is the result of the previous matching. dt_p is the time difference between the two previous scans used for matching and T_i is the initial guess for the next matching. dt_n is the time difference between the current and previous scan.

This simple approach assumes that the velocity is changing at relatively slow rates. Slow meaning that the changes between two consecutive scans are small, which is a reasonable assumption.

4.3.4. Sampling based Approach

As described in Section 3.3, ICP requires a good initial guess to work since it is prone to converge towards local minima. Ambiguities in the scan data can introduce several local minima which might lead ICP to give an incorrect matching result. To make the approach more robust to outliers and to aventually cover ambiguities, the ICP algorithm is performed several times with the same two scans, but using different initial conditions.

Figure 4.4 shows an example of 200 initial conditions where all results clustering towards one maxima (red dots). In Figure 4.5 also 200 initial conditions are displayed, but the results in red agglomerate around various maxima. Having *sufficient* vertical structures will make sure that at least one of these clusters corresponds to the correct result. To find the actual clusters from the set of the ICP results, the algorithm described in Section 3.4 is used.

When the environment is sufficiently structured and has few ambiguities, all initial guesses tend to converge towards the same result. If the environment is not sufficiently structured, has ambiguities or the two scans overlap only in small regions, multiple clusters are likely to occur. In case of multiple solutions these solutions can be used to estimate the uncertainty of the localization, or they can be saved as *mulitple hpotheses* and later an

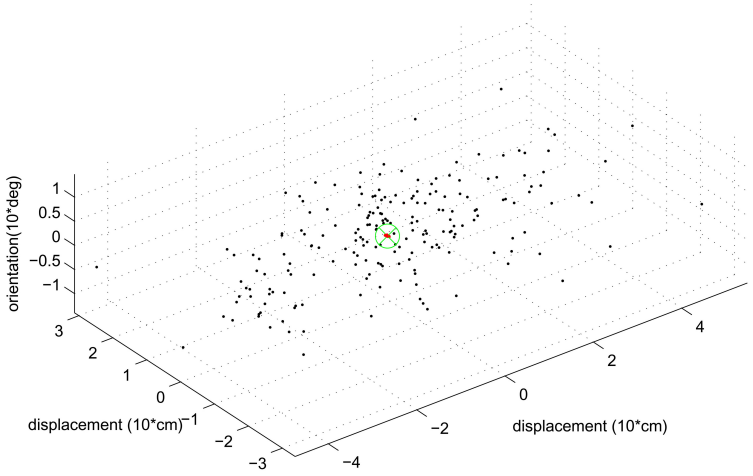


Figure 4.4. Sampling based results with no ambiguities. The black points correspond to initial guesses, the red ones are the results and the green circle with cross is the center of the clustering result

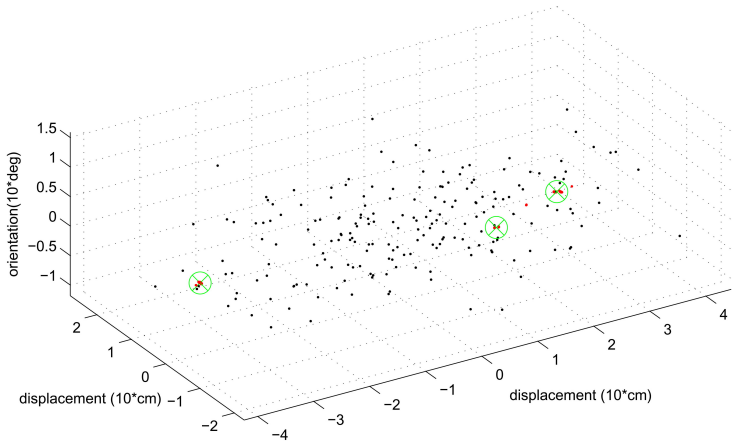


Figure 4.5. Sampling based results with ambiguities. The black points correspond to initial guesses, the red ones are the results and the green circle with cross is the center of the clustering result. 3 possible solutions can be identified.

offline optimization process tries to find the best solution of each timestep. The multiple hypotheses approach will be discussed in section 6.2.

4.4. Uncertainty Modeling

A particle filter based approach is choosen to keep track of the uncertainty over the scan matching procedure. A set of n particles each consisting of a possible robots pose is initialized at the starting position. For each two consecuitive scans the ICP algorithm is used to match the scans. The algorithm is initialized m times, with different, uniformly distributed initial conditions. For each of the n particles, one of the ICP results is randomly choosen and applied for each matching step. Here n = 10,000 particles were used with m = 100 ICP runs.

Algorithm 5 uncertainty estimation

```
1: initialize n particles at the starting location
2: for all measurements z_i do
3:    match scans z_i, z_{i-1} m times with different initial conditions
4:    for every Particle do
5:       randomly choose a matching result and move particle accordingly
6:    end for
7: end for
```

This approach represents the uncertainties by a distribution of samples. It allows to track multimodal distributions that occur when having ambiguities in the sensor data.

4.4.1. Uncertainty Comparison

To evaluate the previously described approach, data was captured with a tilted LIDAR and a horizontally oriented one. Both laser scanners were mounted close to each other on the robot. As discussed in previous sections, we expect the performance for the tilted scanner is comparable for two parameters (T_y, φ) while the accuracy in moving direction (T_x) might be less compared to the results of the horizontal scanner.

Figure 4.6 shows an example of uncertainty measurements evaluated with the approach described in the previous section. The data was recorded in a long hallway. The tilted scanner has few strong features to estimate dhe forward direction, while the horizontal one can sense a wall in front and can therefore better estimate the movememnt in this direction. After moving straigt for about 15 meters a 90° turn was made. The results are as expected - in moving direction the uncertainty is larger for the tilted scanner.

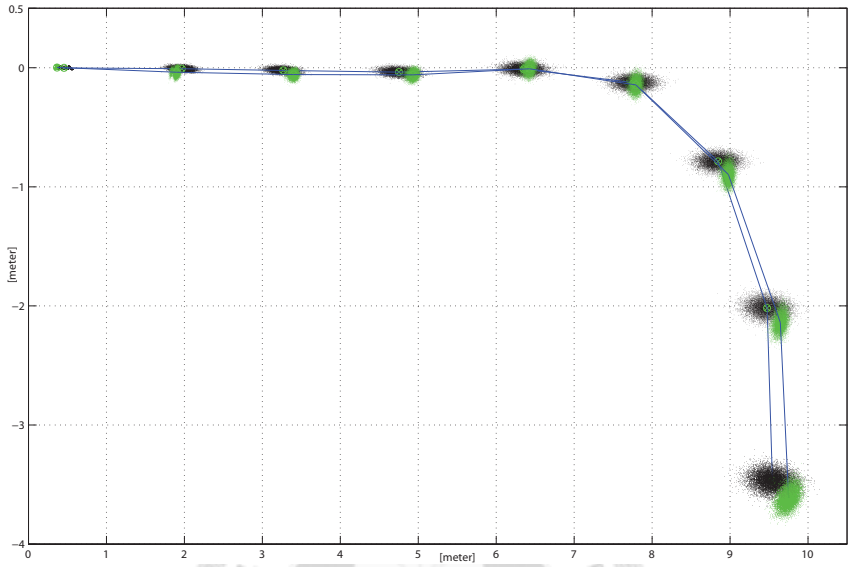


Figure 4.6. Uncertainty comparision between tilted and horizontal scan data. Each set of points represents the particle distribution after 10 matchings applied to the particles. The green point sets represent the horizontal scan, the black ones the tilted. Each set consists of 10,000 particles.

CHAPTER 5

Experimental Results

In this chapter we will introduce results of our approach. First the robotic platform used and its configuration is introduced, followed by detailed results for two different indoor scenarios. The performance of our method will be evaluated and known issues will be discussed.

5.1. Hardware

For our experiments the existing PAL1 robot was modified. Originally based on an electric wheelchair a frame is mounted on this platform in order to allow to fix a LIDAR at a height of around 1.5 meters above ground with an exchangeable tilting angle. Figure 5.1 shows a the used robot. The sensor on top of the frame is a SICK LMS 291-S05 Laser-rangefinder. A second LIDAR of the same type is mounted on the robot with a horizontal orientation to get additional data for performance comparisons. The detailed characteristics of the rangefinder's model used are listed in Table 5.1 (SICK AG Product Specifications).

SICK LMS 291 Specifications	
field of view	up to 180 deg
scanning rate	up to 75 Hz
angular resolution	1, 0.5 or 0.25 deg
maximum range	81 meters
accuracy	few centimeters
dimensions	18.5 x 15.6 x 21.0 cm
weight	4.5 kg
supply	20W @ 24VDC

Table 5.1. Sick LMS 291 Specifications

Figure 5.1. Modified PAL1 robot used for experiments. The top SICK LMS was the main sensor used in the experiments. The horizontally oriented LIDAR in the front was used to collect data for performance comparison.

A camera equipped with a wide angle lens is mounted close to the scanner with a forward orientation so that the field of view from both sensors partially overlap. The camera used is a Dragonfly2 from Point Gray Research Inc. It has a maximal resolution of 640x480 pixels and is using an IEEE1394a interface. All sensor data was recorded on a standard Laptop-PC running on Linux. The data from both SICK LMS were transferred using RS422 with a rate of 500 kbaud through a serial-to-USB converter.

5.1.1. Extrinsic camera-LIDAR calibration

Calibration between the LIDAR and camera was performed following the method introduced in Section 3.5.1 resulting in the following extrinsic parameters

$$T = \begin{pmatrix} 0.036 \\ -0.1 \\ -0.05 \end{pmatrix} \text{meter} \quad R = \begin{pmatrix} 0.1811 \\ -0.01238 \\ -0.01736 \end{pmatrix} \text{radian}$$

Figure 5.2. Camera-LIDAR calibration results The LIDAR is located below the camera slightly shifted to the left.

Even though no ground truth is available, the manually measured translation is consistent with the one the calibration process returned. Projection of the rangefinder data onto the camera image shows excellent compliance for the rotational parameters as shown in Figure 5.2.

5.1.2. LIDAR calibration

The calibration of the LIDARs position is performed as describes in Section 3.5.2. Camera images of a checkerboard placed on the ground are captured and the cameras pose is estimated. To improve the result, several images were captured, height, roll and pitch were calculated and the mean was taken. Then the LIDAR's pose was estimated using the results of the previous section. The following parameters were estimated

$$h = 1.46 \text{ meter} \quad \phi = 0.381^\circ \quad \theta = 18.92^\circ$$

5.2. Software Implementation

The drivers used to access the LIDAR and camera were integrated using linux. To operate the SICK LMS laser range finder the *The Sick LIDAR Matlab/C++ Toolbox* was used which is an open source toolbox branched from the source code used by one of the teams participating in the DARPA urban challenge (Derenick et al., 2009). The camera was controlled using the *libdc1394* package which is an open source driver for fire wire cameras that follow the IIDC standard. The data was recorded and stored with timestamps to allow synchronization.

The actual implementation of the methods described in the previous chapters was done using matlab. In order to speed up the computationally expensive ICP algorithm, this part was implemented in C++ and linked to matlab using a mex-functions. The time to record the two data sets that will be introduced in in Section 5.3 and 5.4 was around 3 minutes while the processing took less than 30 seconds on a standard laptop computer.

5.3. 3D-map of the CSIE Building: Fourth floor

A 3D map using one 2D LIDAR was generated during a run with the PAL1 robot in the fourth floor of the Computer Science and Information Engineering Building on NTU campus. It consists of $\sim$175,000 points generated from 515 scans with a path length of approximately 135 meters. The complete map is shown in Figure 5.3. If color information from the camera was available it was mapped to the corresponding point. Since the opening angle of the camera is significantly smaller than the 180° field of view from the LIDAR a part of the points does not have color information. These points are displayed in green.

Interesting details in the 3D model are marked by capital letters on the map. First to mention is the matching performance. The environment has one large loop so the accuracy can be evaluated how good the locations that close the loop match together. Figure 5.4 shows the region. The robot was starting on the left side and after completing the loop entering the same area again. Orientation and position match reasonably good, given that ICP can introduce minor errors for each matching which accumulate over time, as discussed in previous chapters. The loop is almost closed after processing of more than 500 scans.

Figure 5.5(a), 5.5(b) and 5.5(e) correspond to the area of the hallway marked **A** in the overall map. On the left several door are clearly visible, on the right a balcony is visible

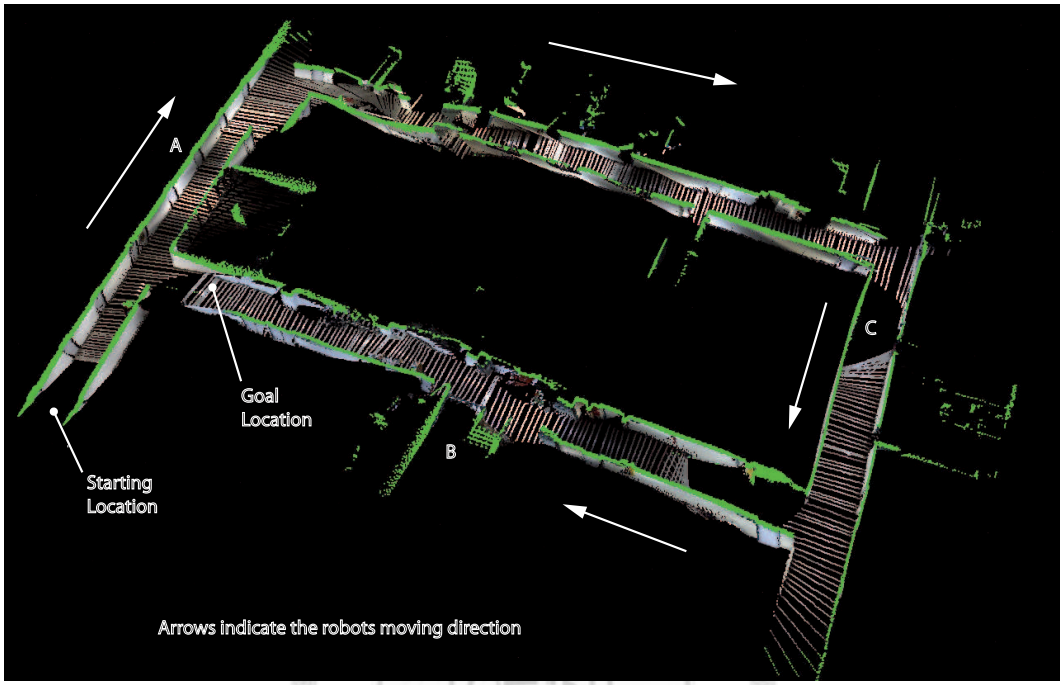


Figure 5.3. 3D map of the 4th floor, CSIE building generated using only one 2D LIDAR. Color information is added when available - otherwise points are shown in green. The longer side of the rectangle measures approximately 40 meters, the shorter one 30 meters.

through large glass doors. The map was generated during night, therefore the balcony appears dark and the different artificial lights lead to different colors depending on the lightning. Figure 5.5(d) corresponds to 5.5(f) - an area with several Boxes stacked in the hallway. Figure 5.5(c) shows stairs leading to the next floor. Boxes and Stairs are both close to the area marked by **B** in the large map. The missing data of the ground in area **C** originate in the turning around the narrow corner. Since the LIDAR 'sees' the ground approximately 4.5 meters in front of the robot turns with too small turning radius lead to this effect.

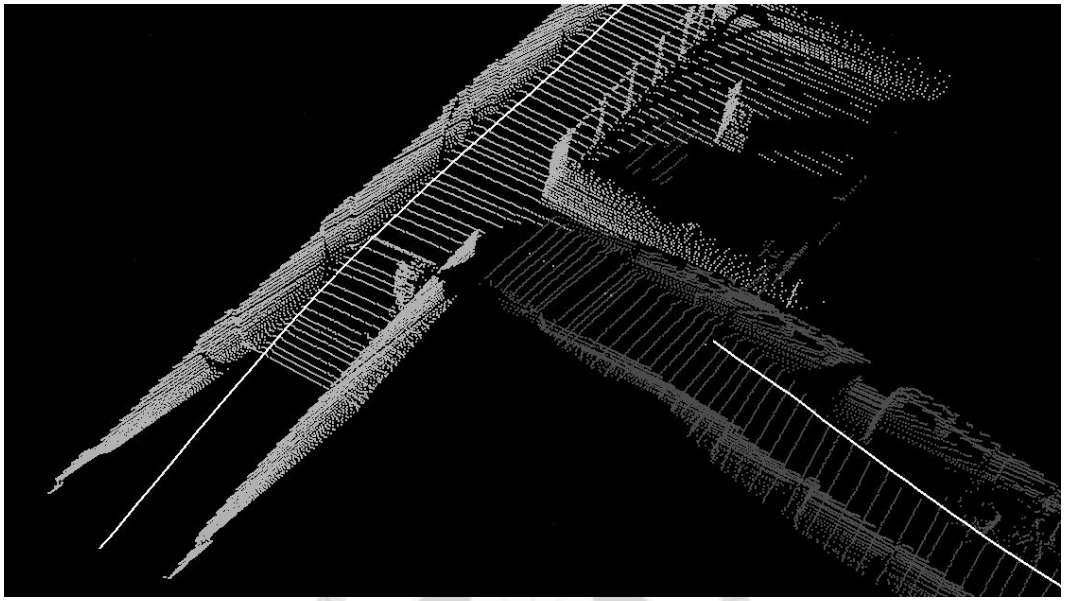
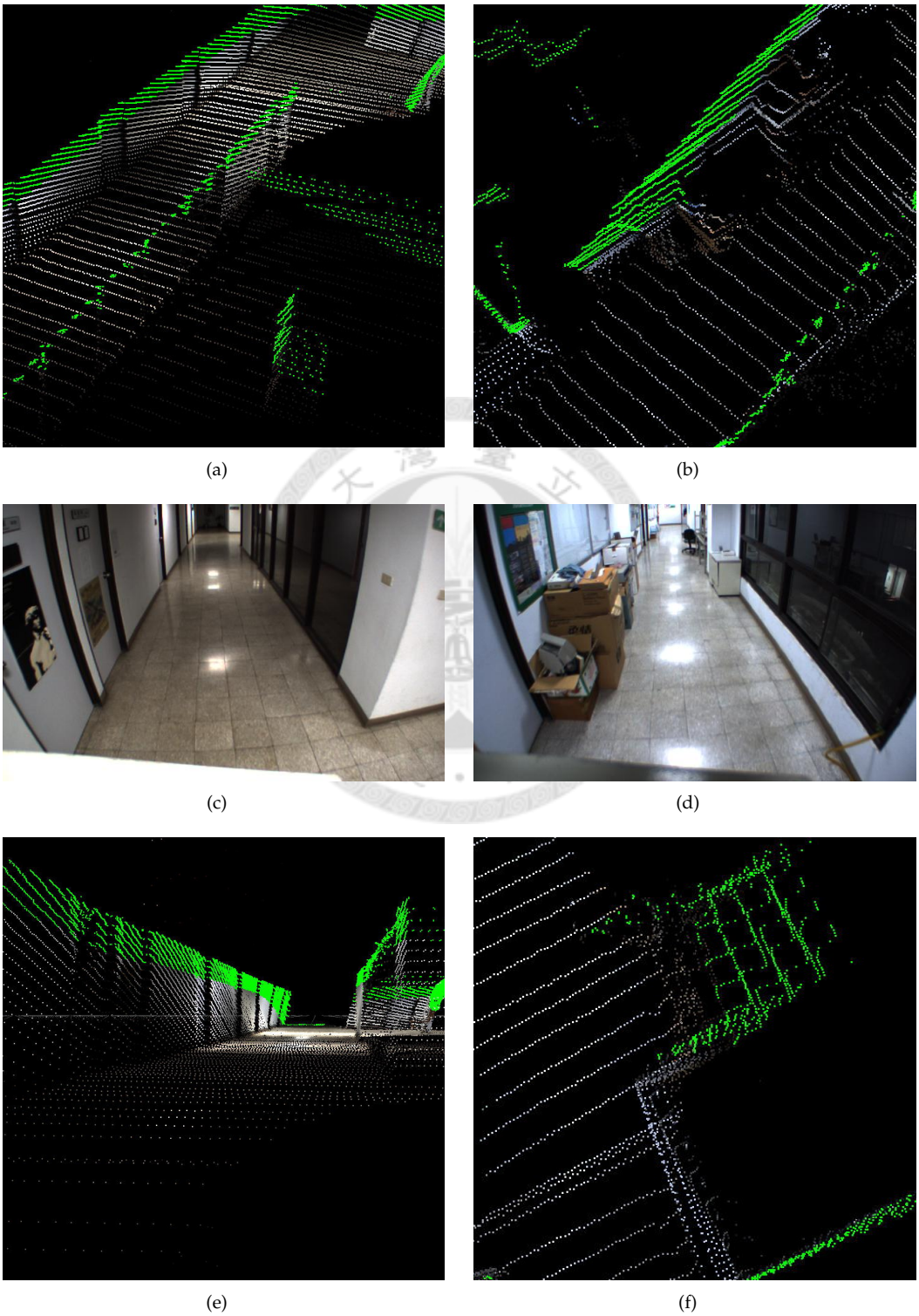


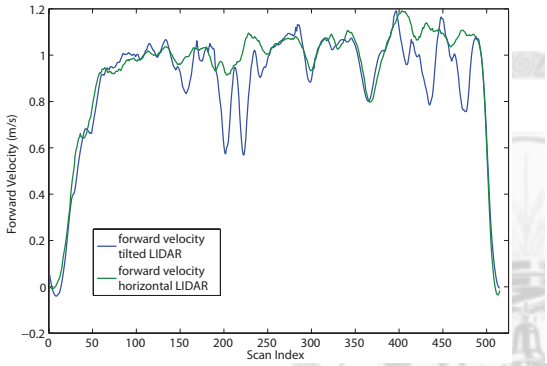
Figure 5.4. Fourth Floor CSIE building, area of loop closing. Robots starting location was on the left, and after completing the loop it comes back from the right side. Points belonging to the start of the path in light grey. Points recorded once robot returns to the same position in dark grey. The while line indicates the LIDARs trajectory. Orientation and Position match reasonably good after around 500 scan matching steps.



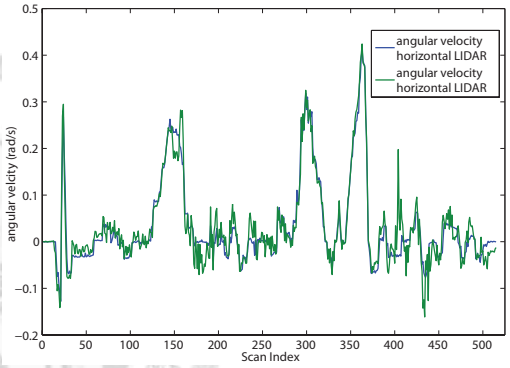
(a) (b) (c) (d) (e) (f)

Figure 5.5. Details from the 3D map generated in the fourth floor of the CSIE building. (a), (c) and (e) show the same scene, as well as (b) and (e).

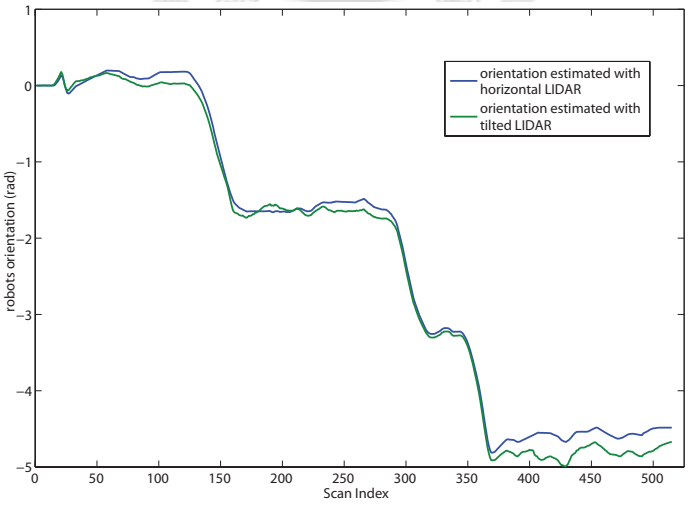
Performance Comparison. Besides the loop close we evaluate the performance by comparing the estimated velocity of the robot using the results from the tilted scanner and the horizontally oriented one. Figure 5.6(a) shows the forward velocity. The tilted scanner underestimates the velocity several times for short periods when passing areas with not enough vertical structures. The angular velocities as shown in figure 5.6(b) are very similar, just the ones from the tilted scanner being slightly more noisy due to the shorter range. At last the orientations are plotted in figure 5.6(c) which match good, only leaving a slight difference after more than 500 scan matchings.



(a) Comparision of forward velocity between tilted and horizontal LIDAR



(b) Comparision of angular velocity between tilted and horizontal LIDAR



(c) Comparision of the robots orientation between tilted and horizontal LIDAR

Figure 5.6. Comparisons between tilted and horizontal LIDAR.

5.4. 3D-map of the CSIE Building: First floor

A 3D model from the first floor of the Computer Science and Information Engineering Building on NTU campus was created by moving the robot once through the environment. It consists of more than 600 scans and $\sim$ 205,000 points. This area is very interesting since it includes a variety of different challenging passages like narrow hallways with sparse vertical features, hallways with windows on one side, areas with convex walls, areas with low structures like benches and also areas with negative obstacles. Figure 5.7 gives an overview over the area

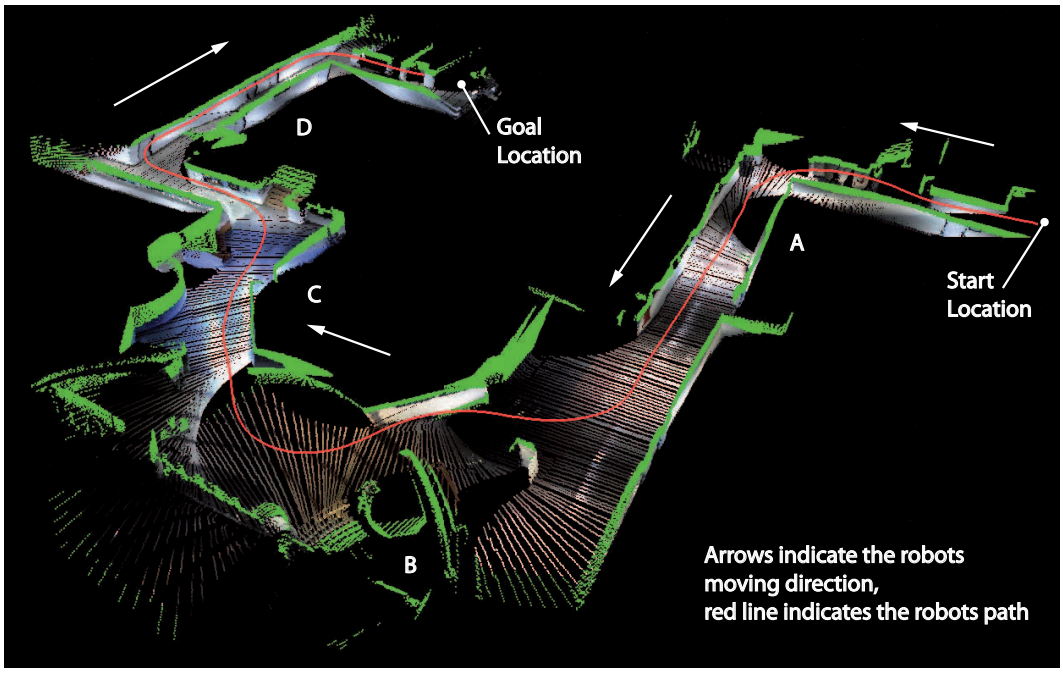


Figure 5.7. 3D model of the first floor, CSIE building. Color data from the camerea was mapped to the image where possible

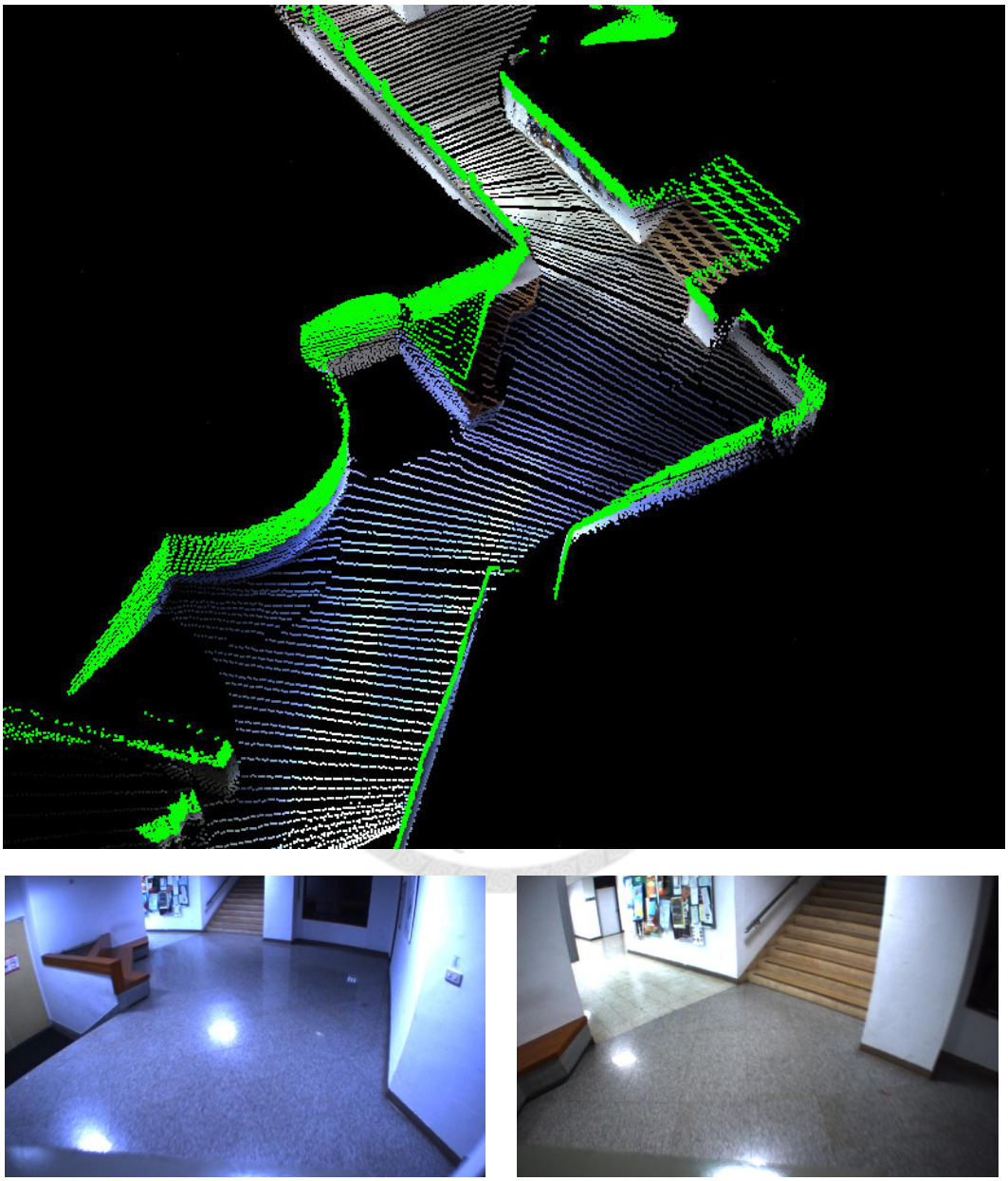


Figure 5.8. Detail of the 3D Model, first floor, CSIE building. Color data from the camera was mapped to the image where possible. The two camera images below the 3D model show the same scene as in the large picture. Details like the stairs, posters on the wall left to the stairs and a bench are clearly visible in the 3D map.

Figure 5.8 and Figure 5.9 show several details from the 3D map generated. For each of the snapshots of the 3D map a corresponding camera image is shown below. Figure 5.8 shows details of the map like stairs, a bench and even posters on a wall are visible besides the stairs. The first image of Figure 5.9 shows an area close to the CSIE buildings main entrance (marked as **B** in the overall map). Two flags in front of a wall are clearly visible. The second image in the first row shows a hallway. In the foreground windows are visible (including erroneous total reflections at the height of the sensor). Left of a door in the middle of the image is a fire extinguisher that is visible in the 3D model as well as in the camera image. Even small details are captured in the map. The two snapshots of the map below show the entrance of the building from different angles.

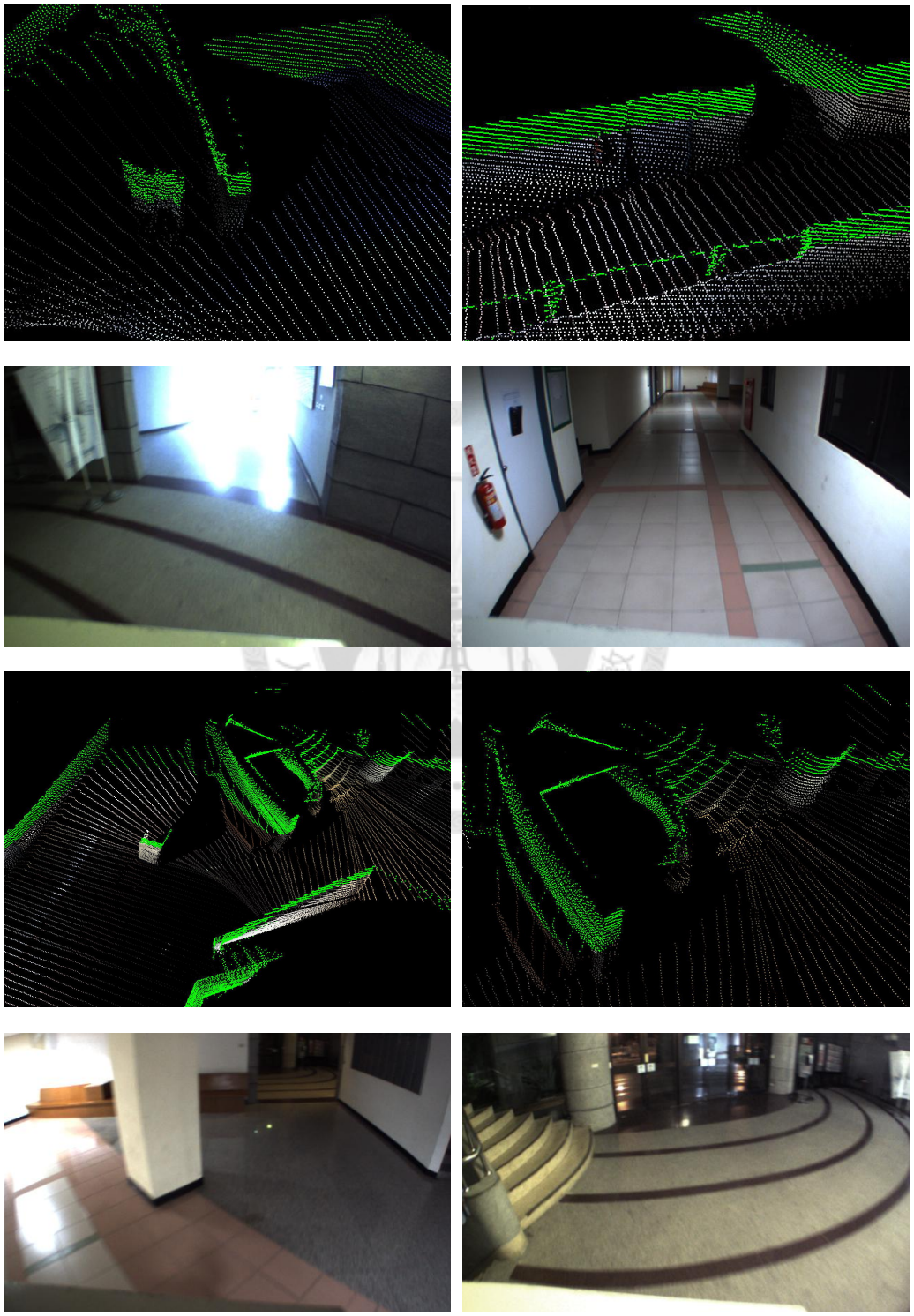


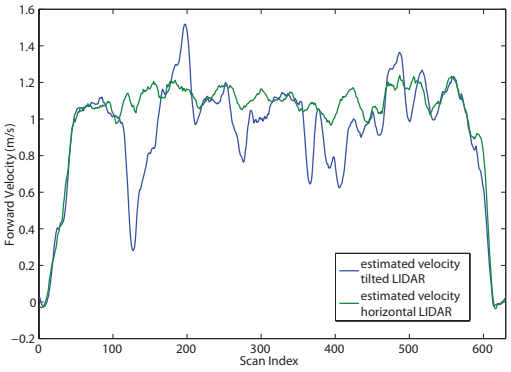
Figure 5.9. Details from the 3D map generated in the first floor of the CSIE building. First and third row show the 3D model generated, the camera images below show the same scene for comparison purposes. Details in the Text.

5.4.1. Performance

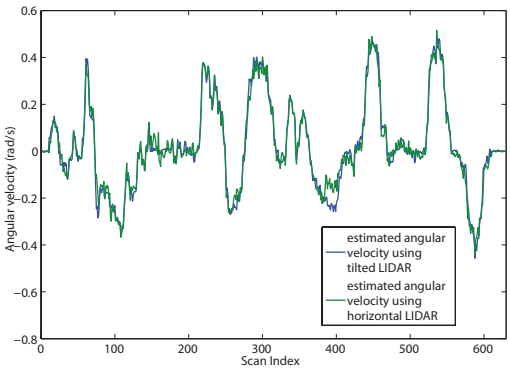
To evaluate the performance we compare the results from both horizontal and tilted LIDAR. Figure 5.10(a) plots the estimated speed during the track in the first floor. The problematic area marked by **A** in the map corresponds roughly to the scans 100-150 which shows that in this area the tilted scanner has problems estimating the forward movement. The angular velocities are shown in figure 5.10(b). They are comparable with the results from the tilted one being slightly more noisy, which is within expectations since only a shorter segment of the walls on both sides are visible. Figure 5.10(c) shows the robots orientation - around index 100 a small difference between the both scanners occur (corresponding to area **A** while the difference gets larger around index 400.

5.4.2. Issues

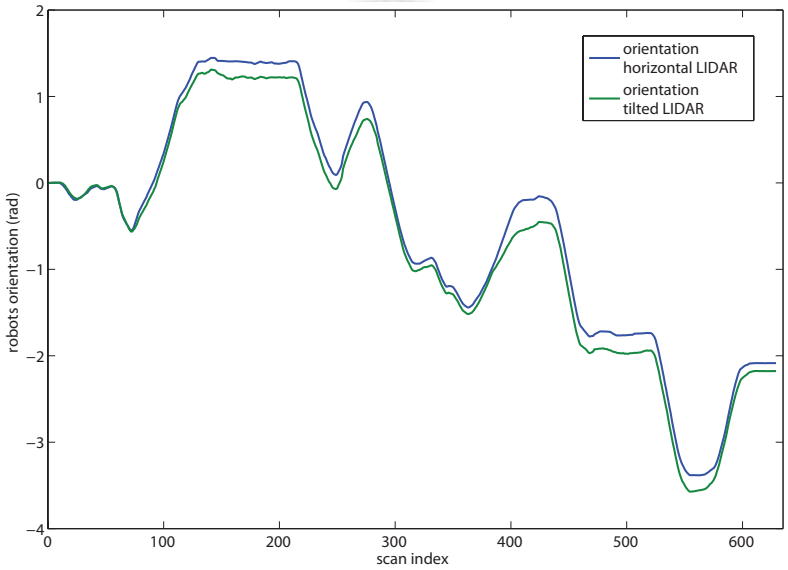
The most difficult region to map in the first floor is the area marked with **A** in the model. This region has windows on the right side while on the left side there is a featureless wall. Windows might return false LIDAR readings since the light of the laser might either go thru, be completely or partially reflected. Therefor in this area the robot receives erroneous measurements from one side, and the other side has no features. While the orientation error introduced is not too large the estimation of the forward movement looses accuracy as clearly seen in Figure 5.10(a).



(a) Comparision of forward velocity between tilted and horizontal LIDAR



(b) Comparision of angular velocity between tilted and horizontal LIDAR



(c) Comparision of the robots orientation between tilted and horizontal LIDAR

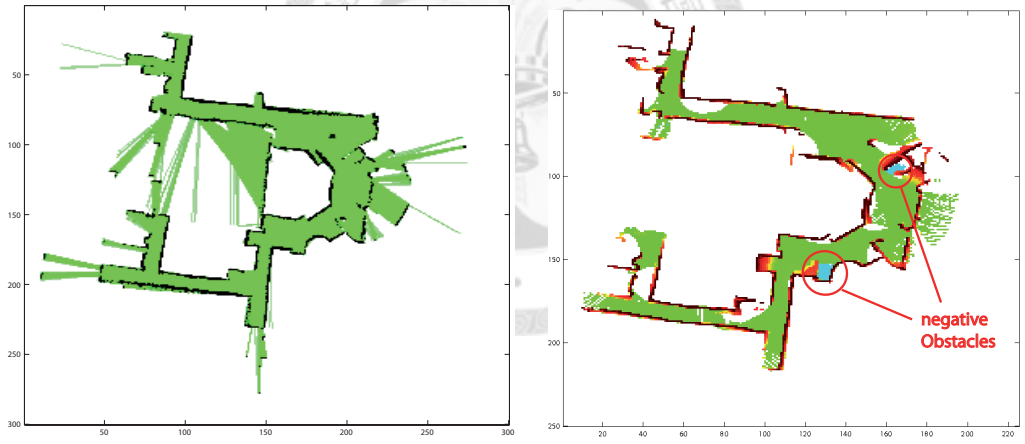
Figure 5.10. Comparisons between tilted and horizontal LIDAR.

5.5. Gridmaps

Figure 5.11 show two grid maps derived from the scanner data. The left figure is calculated from the 3D model generated by the tilted scanner by finding the highest point for each grid. The map is a surface map representing the height of the environment in each grid.

The map on the right was generated by performing a ray casting operation for each laser beam of the horizontal LIDAR. Each ray traversing a grid decreases the probability that this grid cell is occupied, while the endpoint of the beam increases the probability. The colors indicate the probability that a grid is occupied or not (and therefore traversable).

Given that both type of maps can be used for robot navigation one can clearly see that the one generated by the horizontal LIDAR can not see several objects (negative and positive ones) that are perceived by the tilted one.



(a) Occupancy grid map generated by the horizontal LIDAR. For each beam a ray casting operation changes the probability in the grids traversed or hit accordingly. Black corresponds to obstacles, green to traversable grids and white to unknown grids.

(b) Grid map generated by finding the highest point in each grid from the 3D point model built by the tilted LIDAR. Negative obstacles are marked in blue. The y were found by identifying beam-endpoints that were below ground level. For these beams a raycasting operation was used to identify gridcells that are below groundlevel.

Figure 5.11. Comparison of grid maps generated by the tilted and horizontal LIDAR. The right map corresponds to the horizontal oriented scanner and shows only one 2D slice thru the environment. Negative obstacles and obstacles out of this sliceplane are not visible in this map, while the gridmap generated with the tilted scanner shows all these obstacles. Gridsize in both maps is 0.2 meters.

CHAPTER 6

Conclusion

PERCEPTION is a key challenge in mobile robotics. This thesis introduced a new method to generate three-dimensional environment models using only one fixed 2D Laser rangefinder. Localizing the sensor in 3D was made possible by making assumptions which postulate prior knowledge about man-made environments.

6.1. Conclusion

This thesis proposed a method to localize a mobile robot in 3D while constructing a 3D environment map using only one fixed 2D LIDAR. Compared with localization and mapping methods that use one horizontally oriented LIDAR, this approach allows to sense objects that are out of the horizontal sensors slice plane as well as negative obstacles. Our approach has a localization performance comparable with 2D approaches while allowing to capture richer information of the environment.

The 3D environment representation perceived allows object recognition which is hard in 2D due the sparse nature of the data. Compared to other 3D mapping methods this method requires less hardware therefore significantly reducing weight, size, cost and power consumption and allows continuous capturing of data.

6.2. Future Work

Several aspects of this work can be further explored. One aspect is the possible release of some assumptions made by adding additional sensors.

Relieve of Assumptions. The *known-height of the LIDAR* assumption could be dropped given an Inertial Measurement Unit (IMU) is added to the robot. If the IMU could sense

the LIDARs orientations accurately, the sensors height for each scan could be estimated by using geometrical constraints given flat ground that is visible in the scan.

Taking the robot from indoor environments to urban environments might uncover new interesting challenges. When leaving buildings it might be necessary to drop the assumption of flat ground. Given an gradually built elevation map, fixed height of the laser rangefinder and an IMU the robots pose in 3D could still be estimated even it is not operating on flat ground. Since the estimation of the ground height is likely to be only locally accurate, prior knowledge like a large-scale street map with elevation information could be used to correct accumulated errors.

Loop Closing. Currently the work introduced is not able to detect and close loops. This is one important building block to enable reliable large scale SLAM. A Graph-SLAM based approach could be implemented. A future loop detection mechanism might work better than the normal 2D case due the richer 3D information available.

Multiple Hypotheses. The sampling based approach introduced in Section 4.3.4 can be used to keep track of multiple hypotheses in the matching stage. Ambiguities in the environment can lead to several results, each of them providing a good matching result. Treating all these results as valid hypotheses allows to use them in offline optimization method. Assuming a locally smooth trajectory (temporal as well as spatial) could allow to pick the correct hypotheses in an offline stage. Further prior information about the environment could be applied to pick the hypotheses that models the environment best.

Optimization using prior knowledge. This thesis makes already use of prior knowledge about the environment when assuming the ground is flat and that there are vertical structures. The 3D point clouds generated show that the representation of walls is locally accurate, but small errors accumulate leading to slightly curved or inclined walls. Walls in buildings are almost exclusively vertical planes - therefore optimization methods could use this information to correct localization errors - and could reduce the expensive point-representation into a representation by planes.

BIBLIOGRAPHY

Besl, P. & McKay, H. A method for registration of 3-d shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligenc*.

Burgard, W., Fox, D., Jans, H., Matenar, C., & Thrun, S. (1999). Sonar-based mapping of large-scale mobile robot environments using em. *International Conference on Machine Learning*.

Cheeseman, P. & Smith, R. (1986). On the representation and estimation of spatial uncertainty. *International Journal of Robotics Research*.

Derenick, J., Miller, T., Spletzer, J., Kushleyev, A., Foote, T., Stewart, J., Bohren, A., & Lee, D. (2009). The sick lidar matlab/c++ toolbox: Software for rapidly interfacing/configuring sick lidars with applications to real-time experimental robotics. *IEEE/RSJ International Conference on Intelligent Robots and Systems*.

Elfes, A. (1987). Sonar-based real-world mapping and navigation. *IEEE Transactions on Robotics and Automation*.

Frueh, C. & Zakhor, A. (2004). An automated method for large-scale, ground-based city model aquisition. *International Journal on Computer Vision*.

Harrison, A. & Newman, P. (2008). High quality 3d laser ranging under general vehicle motion. *IEEE International Conference on Robotics and Automation*, 7–12.

Li, G. & Liu, Y. (2007). An algorithm for extrinsic parameters calibration of a camera and a laser range finder using line features. *International Conference on Intelligent Robots and Systems*.

Lu, F. & Milios, E. (1997). Robot pose estimation in unknown environments by matching 2d range scans. *Journal of Intelligent and Robotic Systems*, *18*(3), 249275.

Martin, M. & Moravec, H. (1996). Robot evidence grids. Technical report, Carnegie Mellon University.

Nuechter, A., Lingemann, K., Hertzberg, K., & Surman, H. (2005). 6d slam with approximate data association. *International Conference on Advanced Robotics*.

Pless, R. & Zhang, Q. (2004). Extrinsic calibration of a camera and laser range finder. *International Conference on Intelligent Robots and Systems*.

Rusinkiewicz, S. & Levoy, M. (2001). Efficient variants of the icp algorithm. Technical report, Stanford University.

Ryde, J. & Hu, H. (2007). Mobile robot 3d perception and mapping with multi-resolution occupancy lists. *IEEE International Conference on Mechatronics and Automation*.

Schulz, W. H. (2007). Landslide susceptibility revealed by lidar imagery and historical records, seattle, washington. *Engineering Geology*, *89*(1-2), 67 – 87.

Thrun, S., Burgard, W., & Fox, D. (2000). A real time algorithm for mobile robot mapping with applications to multi robot and 3d mapping. *IEEE International Conference on Robotics and Automation*.

Thrun, S., Burgard, W., & Fox, D. (2005). *Probabilistic Robotics*. Cambridge, Massachusetts: The MIT Press.

Thrun, S., Ghahramini, Z., Koller, D., Durrant-Whyte, H., & Ng, A. (2002). Simultaneous mapping and localization with sparse extended information filters. *Proceedings of the Fifth International Workshop on Algorithic Foundations of Robotics*.

Thrun, S., Haehnel, D., Burgard, W., Ferguson, D., Montemerlo, M., Triebel, R., Baker, C., & Whittaker, W. (2003). A system for volumetric robotic mining in abandoned mines. *IEEE International Conference on Robotics and Automation*.

Wulf, O. & Wagner, B. (2003). Fast 3d scanning methods for laser measurement systems. Technical report, Institute for Systems Engineering, University of Hannover, Germany.

Zhao, H., Chen, Y., & Shibasaki, R. (2007). An efficient extrinsic calibration of a multiple laser scanners and cameras' sensor system on a mobile platform. *IEEE Inteligent Vehicle Symposium*.

Document Log:

Manuscript Version 5 — 3 August 2009)

Typeset by LaTeX — 5 August 2009

ANDREAS DOPFER

THE ROBOT PERCEPTION AND LEARNING LAB., DEPARTMENT OF COMPUTER SCIENCE AND INFORMATION ENGINEERING, NATIONAL TAIWAN UNIVERSITY, NO.1, SEC. 4, ROOSEVELT ROAD, DA-AN DISTRICT, TAIPEI CITY, 106, TAIWAN, *Tel.* : (+886) 2-3366-4888 EXT.407

E-mail address: r9622144@ntu.edu.tw