Robot Vision Applications using the CSEM SwissRanger Camera

Sigurjón Árni Guðmundsson

Kongens Lyngby 2006
IMM-THESIS-2006-65

Technical University of Denmark
Informatics and Mathematical Modelling
Building 321, DK-2800 Kongens Lyngby, Denmark
Phone +45 45253351, Fax +45 45882673
reception@imm.dtu.dk
www.imm.dtu.dk

IMM-THESIS: ISSN 0909-3192

Abstract

The SwissRanger is new type of depth vision camera using the Time-of-Flight (TOF) principle. It acquires in real time both normal intensity images and 3D range images. It is an active range finder with a harmless light source emitting near infrared light at under 1W. Most other active range finders are laser based and have much higher latency. The SwissRangers usefulness is proved here by solving two diverse robot vision applications: The mobile robot localization problem and the 3D object pose estimation problem.
The robots localization is found by segmenting range images; into planar surfaces. The segmentation is done by calculating the local surface normals at each pixel, grouping the image into regions and robustly fitting to planes using RANSAC. From these planes a map of the robots environment is constructed.
For a robot to handle an object it has to recognize the objects pose or orientation in space. This is approached by using a dimensionality reduction method called Local Linear Embedding (LLE). A dataset, with range images of an object can be seen as points in a very high dimensional pixel space. It has been shown that for 3D objects such points lie on nonlinear manifolds in the high dimensional space. The LLE technique reduces the dimensionality down to a true dimensionality of the manifold and reveals the separating characteristic in each point namely its pose. The pose of a new objects can then be detected by mapping it to this low dimensional space.

Keywords: mathematical modelling, 3D imaging, time-of-flight, range images, robot localization, pose estimation, range image segmentation, non-linear dimension reduction, Local Linear Embedding.

Preface

The work leading to this Master thesis was carried out within the *Image analysis and Computer Graphics* group at the *Informatics and Mathematic Modelling (IMM)* department of the *Technical University of Denmark (DTU)* between the 1^{st} of February and the 1^{st} of August.

The thesis serves as a requirement for the Master of Science degree in engineering, M.Sc.Eng. The extent of the project work is equivelant to 30 ECTS credits and was supervised by professor Rasmus Larsen and co-supervised by Jens Michael Carstensen.

I would very much like to thank my supervisors for suggesting to me this new exciting technology and for all their help during the work period. Also I'd like to thank Bjarne Ersbøll and Henrik Aanæs for all their help during the period, Lars Kai Hansen for his valuable input on machine learning issues, and fellow student Yin Yin for our sharing and discussions of the SwissRanger connected problems.

Furthermore I want to thank my family whom I love so much; María Rán Guðjónsdóttir and Guðjón Teitur Sigurjónsson, this thesis is dedicated to them.

Sigurjón Árni Guðmundsson, Lyngby, July 2006.

sigurjon_arni@yahoo.com

Nomenclature

For consistency the following notation is used in the report.

$\mathbf{X}$	matrix
$\mathbf{x}$	vector
$\vec{n}$	surface normal
$\hat{a}$	optimal solution
$\bar{x}$	mean of vector x
$p(\cdot)$	probability density function
$P(\cdot)$	probability
ε	costfunction (error function)
θ	Horizontal angle in spherical coordinates
ϕ	Polar angle in spherical coordinates
μ	Mean value of a Gaussian density
σ^2	Variance of a Gaussian density
Σ	Covariance matrix of a Gaussian density

Abbreviations

PCA	Principal component analysis
SVD	Singular value decomposition
EM	Expectation Maximization algorithm
GMM	Gaussian mixture model
LLE	Local linear Embedding

Contents

Part I

Chapter 1

Introduction

1.1 3D Imaging in Robotics

Robots rely on sensors as we rely on our senses. This reaction to its senses has given robotics a dramatic feel - fiction writers have found this fascinating: copying gods work in making a sensible being. A theme that can be seen in classics from Mary Shelleys *Frankenstein* and Isaac Asimov's science fiction novels to todays children's television, where adventures of friendly and evil robots are endless. Robot sensors are various; tilt sensors and compasses for measuring the robots pose, proximity and pressure sensors for picking up items, motion sensors for surveillance etc. For robots the most important is the vision sensing and furthermore the most complicated.
Mobile robots that are intended to move around in an area that is subject to change, must be able to perceive the environments 3D structure just so they can move around safely. Also, industrial robots that maneuver objects must know where and how the objects lie in space. For these types of problems and more robots need depth or range sensors; devises that can measure the distance from the robots to a surface. Such sensors have been designed in all sorts of sizes utilizing numerous types of technology.
One category of range sensors are active point-sensors that emit some sort of modulated wave e.g. sound (sonar), radiowaves (radar), lasers (ladar) or infrared light to find distances by measuring the time it takes for the wave to

reflect back. The range and accuracy of such sensors is connected with what kind of wave is used. The sonar ranger propagates high frequency audio signals in a cone from the emitter and thus gives inaccurate spatial information of the reflecting surface in that cone. Infrared point sensors usually only have sending power just to measure a few centimeters while radar sensors can be used at very long range sensing but can be useless up close.

Other sensors give detailed range information of everything in their field of

Figure 1.1: *A Range Image.* The dark regions are close and the lighter are further away. The scale is in meters. The background wall is past the non-ambiguity range and therefore appears 7.5m closer. (Captured with the Swiss-Ranger)

view, producing images where each pixel corresponds to a distance measurement. These kind of images are called range images or depth maps, sometimes 2.5D images (see figure 1.1). This kind of 3D information for short range measurement are much more informative than the point sensor measurements. If the images can be developed fast enough they are without a doubt the best solution for all sorts of applications. But this latency is a problem most range finder suffer from. In the following a few of these techniques are introduced.

1.1.1 Range Finder Methods

Range finders can basically be categorized into two types: Active methods and passive methods. Active methods propagate energy into the scene to be measured while passive methods use multiple images to calculate the range by triangulation.

Most active methods emit some sort of modulated wave and detect the depth by measuring the time it takes for the wave to reflect back. These emitted waves are modulated either in amplitude or frequency which makes the time-of-flight

(TOF) measurement possible[1].
Laser Range Finders (LRF) Most laser range finders emit a pulsating laser onto mirrors or optics which then mechanically move, sweeping the field of view. The scene is then triangulated from the mirror angle information. LRFs measure the TOF and can produce images with sub *mm* accuracy at short range (figure 1.2 shows one such commercial product). The fact that these cameras have to sweep the scene mechanically and do triangulation calculations afterwards results in quite long processing time. Such long latency makes LRFs unsuitable for most robot applications and they are mainly used for scanning purposes: 3D modeling for industry, architecture, archeology etc. Horizontal LRFs are on the other hand widely used for fast depth measuring, i.e. sensors that make 180° horizontal sweeps very quickly. They can be used for finding walls and obstacles etc., but without any vertical information this cannot be compared with range images.
Passive methods usually produce depth images by using two or more 2D inten-

Figure 1.2: *Three Commercial Range Finders.* The *Minolta* Vivid 910 LRF, 3 *TYZX*Deep Sea Stereo Cameras with different baselines and the *3D-shape* BodySCAN that uses stereo and structured light for its measuring (from [23, 24, 22]).

sity images calculating from the depth from distance between the image origins. These methods include, stereo imaging and multiple view imaging, structure from motion and more.
Stereo Vision
Stereo vision uses two high resolution cameras that are calibrated together in a rig. In a way its depth perception is similar to how we perceive depth: Our eyes send the brain two images at slightly different angles, from this the brain calculates the depth map. In computer vision the images can be matched using various methods, and the depth is calculated using epipolar geometry and triangulation. Stereo vision has proven quite robust in the fields of mobile robotics and 3D object perception. Its main problem is the heavy computations involved

[1]The AM TOF principle will be discussed in detail in chapter 2.1

and it relies on texture and contrast information to in the image matching; if the scene is uniform the measurement fails.

Structured Light
Lately, a new method called structured light has been popular in the range image research field. It is an active triangulation method where different coded light is projected onto the subject and from the borders, light and shadows the depth can be triangulated (figure 1.3). It is related to the multible view methods and solves the contrast problem by simply projecting contrasts into the image. This method is also computationally expensive; it usually uses up to 30 coded images to generate images, and it is not very mobile for use in robotics.

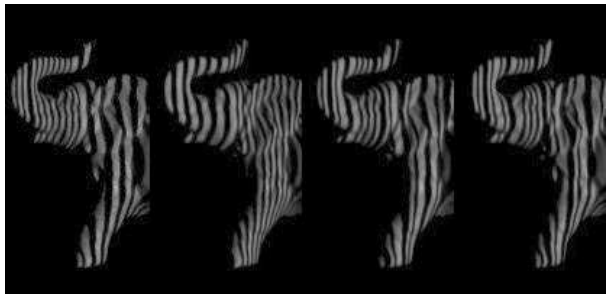

Figure 1.3: *The Structured Light Method.* From several coded light projections onto an object the range can be triangulated (from [12])

1.1.2 TOF Camera

A TOF camera is a range finder using active TOF measurements acquiring range and intensity images at almost real time. The SwissRanger SR-3000 is such a camera and will be explained and used for the experiments in this project.

In the following, an introduction to the problem considered in this thesis as well as the aim of the project and the structure of the report is presented.

1.2 Motivation and Background

Annually the *IMM* department at *DTU* invites companies, organizations, and institutions interested in image analysis, vision systems, and computer graphics

to three days of presentations and demonstrations. At the *Industrial Visiondays* in 2005, Thierry Oggier of *Centre Suisse d'Electronique et de Microtechnique* (CSEM) presented a new range finder camera design. A camera built on the TOF principle, a devise that "sees" distances and delivers 3D range images at almost real time, which is a quality valuable in numerous machine vision scenarios. *IMM* purchased a CSEM TOF camera to try its qualities and to be leading in the use of this exciting technology.
Robot vision is a field within image analysis where such a camera could prove ideal. In *IMMs'* Image lab there is a Cartesian robot well suited to test various Robot scenarios. The problem is thus to make and solve robot vision application problems for the TOF camera mounted on the Cartesian Robot.

1.3 Problem Description

The problems considered in this thesis are robot localization and pose estimation. Thus the camera is evaluated in two dissimilar applications within the diverse field of robot vision.
For a Robot to move to a target it must know where it is and be able to evaluate its surroundings. This can be done by segmenting the range image finding the floor and walls and obstacles in the robots environment and in that way making a map of its milieu, localizing it in a scene it may or may not have some prior knowledge of.
For a robot to handle an object it has to be able to recognize how it is oriented in space, i.e. how it has been rotated relative to some position. This is a classic problem in tracking, recognition and classification of objects and in robot tasks such as bin picking.
The main objective of this thesis is to develop methods that provide good solutions to these vision problems and show how well the TOF camera is suited for them. Such robotic problems also involve other fields in engineering such as control theory and automation. Here simple solutions are chosen for such problems, as more sophisticated methods are not in the scope of this work.

1.4 Thesis Overview

The structure of the thesis is as follows.

- Chapter 2 gives an overview of the TOF camera. Its characteristics, the technical principles it follows, its advantages and some problems that were

run into on the way.

- Chapter 3 describes the Cartesian robot in the image lab. How the camera is mounted on it and how it can be controlled.
- Chapter 4 introduces several common approaches in robot localization. A suitable algorithm for the problem is presented and the theoretical background of the methods are given.
- Chapter 5 covers the pose estimation theory in the same fashion as chapter 4.
- Chapter 6 presents experiments and results for the robot localization problem. The data is introduced and the algorithm is tried in different scenarios.
- Chapter 7 first introduces the datasets used in the pose estimation experiments. The results of the experiments are then analyzed.
- Chapter 8 summarizes the results of this project and discusses what improvement can be achieved in the future.

Chapter 2

The CSEM SwissRanger SR-3000 Camera

The SwissRanger SR3000 is developed by the Swiss Center for Electronics and Microtechnology (CSEM)(see figure 2.1). It is a state-of-the-art, solid-state, time-of-flight imaging devise that delivers depth maps (range images) as well as intensity (gray-level) images. The SwissRanger is an active range finder as it emits light and measures the time-of-flight (TOF) i.e. the time the light takes to travel from the camera to an object and back again. The distance is R and then the TOF is:

$$TOF = \frac{2R}{c} \tag{2.1}$$

Where c is the speed of light ($3 \times 10^8 m/s$). The camera is designed on the criteria to be a cost-efficient and eye-safe range finder solution.

2.1 The Time-of-Flight Distance Measurement Principle

Figure 2.2 shows the structure of the SwissRanger.

Basically it has a amplitude modulated light source and a two dimensional

Figure 2.1: The CSEM SwissRanger SR-3000 Camera

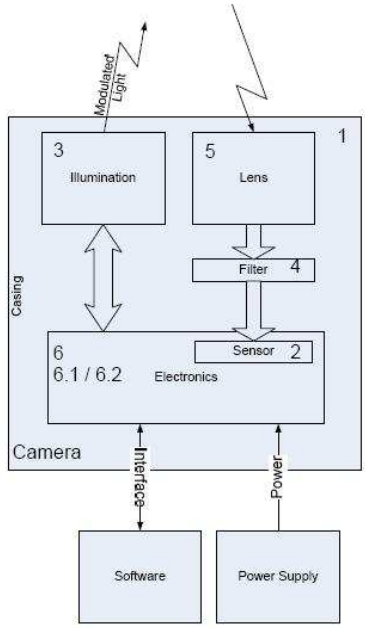

Figure 2.2: Block Diagram of the Camera. 1. Casing, 2. High Speed Demodulation imager, 3. Near-Infrared Illumination Unit, 4. Optical Band-Pass Filter, 5. Lens, 6. Electronics: 6.1. USB communication board, 6.2. Image Processing Board. (From [6])

sensor. The light source is an array of 55 near-infrared (n-ir) diodes (wave length $850nm$) that is modulated by a sinusoidal at $20MHz$. This is light is invisible to the naked eye see figure 2.3. Further specifications can be found in appendix A.

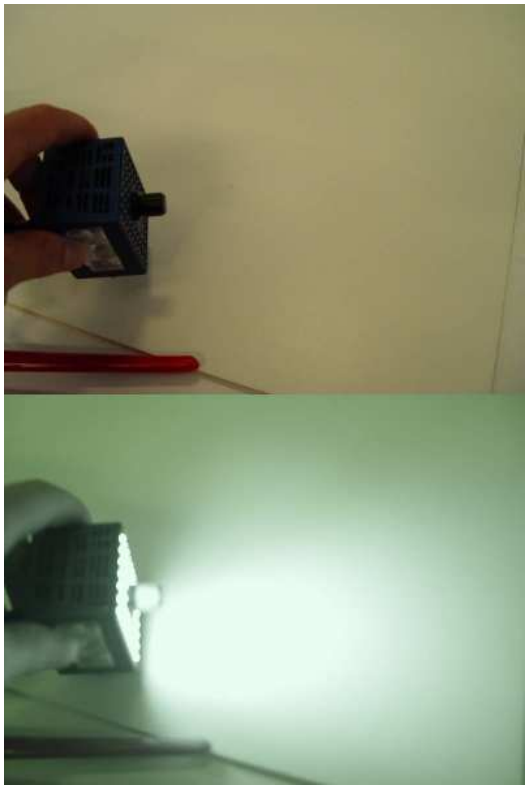

Figure 2.3: The $850nm$ emitted light can be detected using the nightvision feature on many consumer cameras.

The sensor is a 176×144 custom designed $0.8\mu m$ CMOS/CCD chip where each pixel in the sensor demodulates the reflected light by a lock-in pixel method, taking four measurement samples (m_1, m_2, m_3, m_4) $90°$ apart for every period. From these samples the returning signal can be reconstructed as shown in figure 2.4). The reconstructed signal is characterized by its offset:

$$B = \frac{m_1 + m_2 + m_3 + m_4}{4} \tag{2.2}$$

The amplitude:

$$A = \frac{\sqrt{(m_3 - m_1)^2 + (m_4 - m_2)^2}}{2} \tag{2.3}$$

And most importantly the phase shift between the emitted- and the returning light:

$$\phi = \arctan\left(\frac{m_4 - m_2}{m_1 - m_3}\right) \tag{2.4}$$

From the phase shift the distance is calculated:

$$R = R_{max}\frac{\varphi}{2\pi} \tag{2.5}$$

where R_{max} is the non-ambiguity range of the sensor, found by: $R_{max} = c/2f_m$ where f_m is the modulation frequency of the light source and c is the speed of light[1]. The sensor is a so called 2-tap sensor which means that it can store two of the four samples needed and consecutive measurements have to be performed to register the returning signal. The measuring is also integrated over many periods to improve the results.

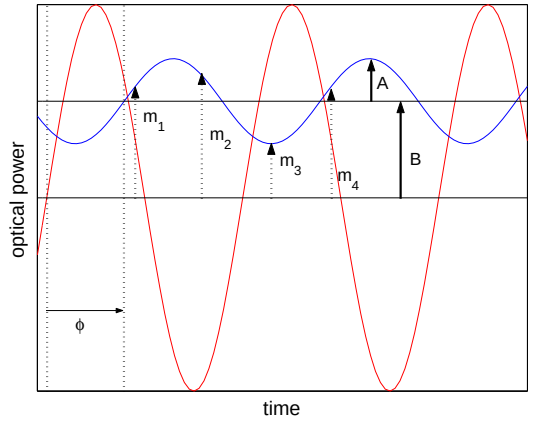

Figure 2.4: TOF Measurement Principle. The emitted(red) and the reflected (blue) signals. The phase shift ϕ, the offset B and amplitude A of the reflected signal is found from the four intensity measurement samples m_i.

Reconstructing the signal for each pixel means the camera is able to return two images at the same time: the range and the intensity per pixel.

[1] R_{max} is 7.5m in the SwissRangers default configuration

2.1.1 Limitations of the Distance Resolution

The limitations of a solid state sensor is noise. The sensor has noise sources such as thermal noise, quantization noise, reset noise, electronic shot noise etc. Most of these noise inputs can be greatly attenuated or eliminated except the electronic shot noise which is explained in quantum physics by when the finite number of photons is so small that statistical fluctuations in a measurement are detected. The noise is Poisson distributed (discrete Gaussian) and scales as the square root of the average intensity The standard deviation of the shot noise is the square root of photons or photo generated charge carriers.
As the standard deviation of the noise in the integrated sampling points m_i is $\sqrt{m_i}$ then the error propagates to an error with the standard deviation:

$$\sigma_R = \frac{R_{max}}{\sqrt{8}} \frac{\sqrt{B}}{2A} \tag{2.6}$$

For maximum range accuracy the offset must be minimal and the amplitude as high as possible.
The offset is mainly caused by background light i.e. other light sources than the SwissRanger's. To minimize this effect an optical bandpass filter attenuates all light frequencies except for the near infrared light from the diodes. The background light is then further suppressed by signal processing on chip, which corresponds to 50% of the maximal sunlight (specifications are given in appendix A). The amplitude is mainly effected by the objects distance from the sensor and its reflection properties, if it is far away the amplitude is smaller and also if it is e.g. dark colored then the reflection is weaker. This problem is further issued later in this chapter.

2.2 Practical Limitations of the SwissRanger Camera

2.2.1 Noise, Blur and Overflow

Noise, blur and overflow are the tradeoffs when the integration time is chosen. If the integration time is short the results are very noisy and if it is long the results are smoother with moving objects getting blurred, if the integration time is very long overflow starts to contaminate the results especially with objects close to the camera. Figure 2.5 shows the difference in the results. The camera has an auto-integration time feature which finds automatically a time that lowers the noise sufficiently for most applications without smoothing the image to much.

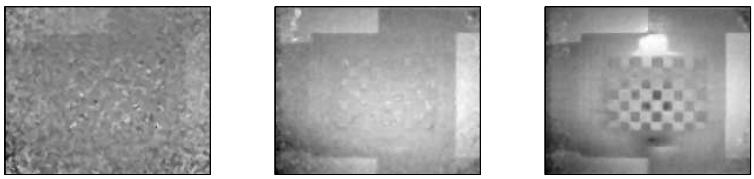

Figure 2.5: Changing the Integration Time. The integration time set to 1 msek, 10 msek (auto), and 25 msek , the measured distance in the center pixel is also given.

2.2.2 Reflection Properties

Different materials with various textures and colors have very different reflection properties i.e. how strongly the light is reflected back. This affects the amplitude and intensity of the reflected light which in turn affects the depth resolution as equation 2.6 clearly states.

Figure 2.6 shows that when the integration time gets longer the accumulated

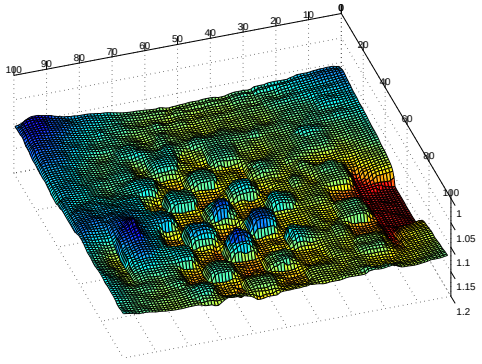

Figure 2.6: Chessboard Depth Image. Depth resolution in the dark area is worse than in the white areas.

amplitude measurements on the black pixels are much lower than in the white ones, resulting in different range accuracy. The black squares in the planar chessboard pattern appear as holes in the plane!

2.2.2.1 Multipath

The last practical problem run into was the multipath problem. In areas such as room corners the multipath phenomena problem occurs (figure 2.7). Light is reflected twice and the shorter direct reflection cannot be separated from the longer path; this creates an error, that smoothes these edges and adds to the distances.

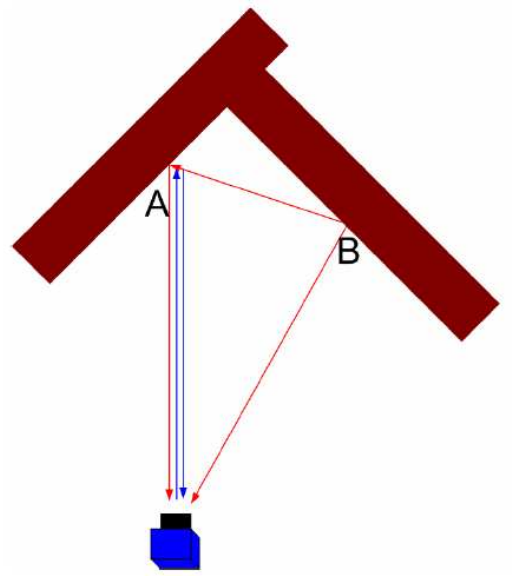

Figure 2.7: Multipath. The light reflects twice giving false measurements

2.3 Advantages

The SwissRanger has many advantages and the low latency is probably its strongest. None of the competing range finders can offer such a fast range acquisition and especially not in such a small package. This makes the SwissRanger extremely valuable in many practical fields where these qualities outweigh the cameras misgivings. In the world of range finding this is typical, all the different methods have their strengths and their weaknesses.

2.3.1 Comparison with Competing Technologies

Laser based depth measurement is limited to controlled industrial environments where eye and other health hazards are not a issue. In other scenarios the straight forward choice would be multiple view (e.g. stereo vision) or using a devise such as the SwissRanger. In table 2.1 a brief comparison is done on these methods:

	SwissRanger	**Stereo Vision**	**Laser Based**
Best Depth Accuracy	sub cm	sub mm Needs contrasts in the scene to match.	sub mm
Latency	Minimal Needs longer integration time for longer range.	Possible Triangulation is computationally expensive	Yes Slow mechanical components.
Multi Path	Yes	No. Passive	Less effect in short range.
Health Hazard	Safe Emits N-IR light at low power	Safe Passive	Hazardous Lasers are dangerous to the eyes.
Moving Objects	No	No	Yes A nodding mirror sweeps the beam across the field of view
Size	Small Light source and sensor can be very closely packed	Depth Resolution depends on distance between the imagers	Large Mechanical mirror and high powered laser are bulky.
Cost Potential	Cheap Based on solid state technology.	Limited 2 high resolution cameras	Expensive Laser and mechanical parts

Table 2.1: Comparison of the SwissRanger, Stereo Vision and Laser Based Range Cameras

Chapter 3

The Cartesian Robot

The Cartesian robot in IMM's ImageLab is a "caged robot" i.e. it has an arm that can move within the cage (See figure 3.1). The SwissRanger camera is mounted on this arm. The arm is moved by servo-motors one for the X-direction another for Y and the third for Z. The robot had been used before to move around in a predefined path, e.g. to simulate a moving camera for quality inspection ([15]).

The goal was to use the TOF camera to control the robot, i.e. send the robot

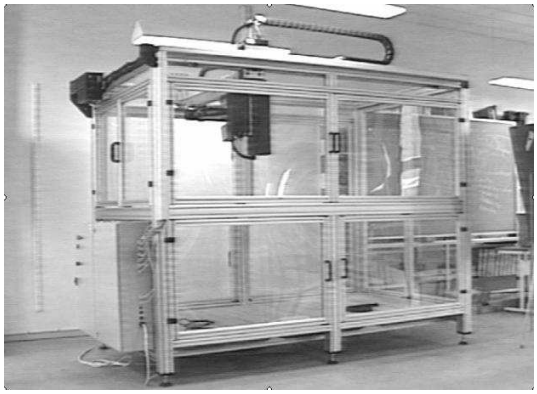

Figure 3.1: The ImageLab Cartesian Robot

instructions based on its observations on the environment.

3.1 The Robot, Dimensions and Mobility

Figure 3.2 shows the ground plan of the robots cage and the arms mobility. The motors are quite powerful and can actually make the whole cage jump, but

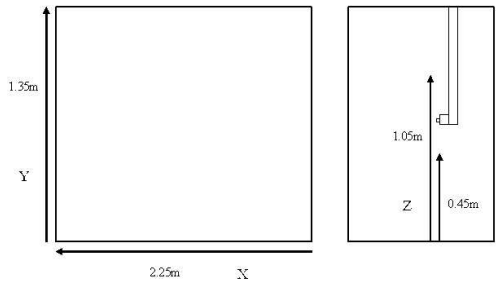

Figure 3.2: The Cartesian Robots Ground Plan and Mobility. The coordinates of the robot are marked X, Y and Z

for the purposes of this project the acceleration and velocity were chosen low as high speed was not a goal.

3.2 Control

The Robot can be controlled by a *C++* program on a PC via a digital IO board. The IO board is connected to a board of relay switches which then switch on and off the commands on the motor controllers. These commands are basic movement action commands for each motor: e.g. move slowly to right/left, move with acceleration a and velocity v in direction right/left, save current position and go to saved position.
The main modifications for the purposes of this project was a command to move the arm a certain relative distance, i.e. translate the arm from the present position by a distance (X_t, Y_t, Z_t). This is done by calculating the time the arm takes to move at the given acceleration and velocity and simply turning on the specific relays for that period of time.
Also the program structure was changed to a class structure, the console interface was changed and a work on a new *dll*-interface was started. This way

Matlab will be able to load the *dll* and interface with the Robot and control the camera-robot interaction. Details of the robot class and interface can be found in A.

3.3 Pan-Tilt Camera and Rotating Table

To add some flexibility to the SwissRangers viewing angle it was mounted onto a Sony EVI-D31 pan-tilt camera (figure 3.3). The Sony control class was integrated into the Robot console and the camera can be panned and tilted to some key positions.

Figure 3.3: The SwissRanger Mounted on The Sony EVI-D31.

A rotating table (figure 3.4) was also used to make datasets of objects from multiple angles. The rotating table is also controlled via the C++ interface or directly from *Matlab*. It sends commands to the table controller which triggers one of three functions: a 1° movement, 2° or a 10° movement.

More details about the Robots *C++* interface, member functions etc., specifications about the Sony camera and the JVL table can be found in appendix B.

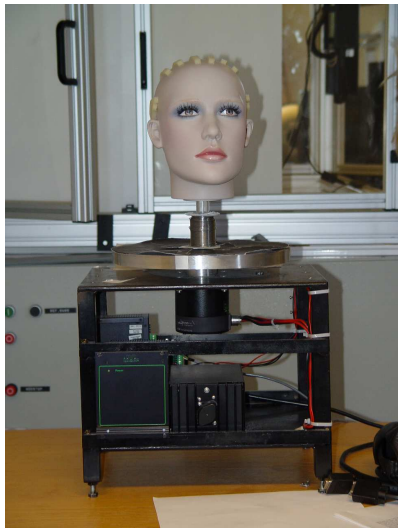

Figure 3.4: The JVL Rotating Table

Part II

Theory

Chapter 4

Robot Localization

How does a Robot know where it is?
Robot designers have been trying to solve the localization problem since the beginning of robotics.
The numerous approaches to mobile robot localization can be divided into two major classes: using a geometric model, or using a topological model([29]). The geometric approach depends on the metric representation of the robot and its environment, i.e. building a sensor based map in some world coordinate frame. The more abstract topological approach uses a graph representation that captures the connectivity of a set of features in the environment. The topological model has shown great qualities in localization and obstacle avoidance where speed is a virtue and robustness where noisy range sensors such as sonar and infrared sensors are used.
For the purposes of this project the geometric approach is better suited. The camera gives geometric measurements of the small indoor scenario that can be processed into a map of the robots surroundings.

The algorithm used for this SwissRanger robot vision system is designed to find typical indoor scenario structures, namely the floor and walls. The solution can be split into two steps: First a preprocessing step where a range image acquired from the camera is segmented into primary surfaces. Then a robust fitting and mapping step; where the segments are parametrically presented and

recognized as the room's floor and walls. These two basic steps and the full algorithm are showed in the flowchart in figure 4.1. This approach is a fast and robust, surface normal based, plane finding algorithm. In the next sections the algorithm is covered in-depth, first by introducing range image segmentation algorithms and then the steps of the algorithm are described

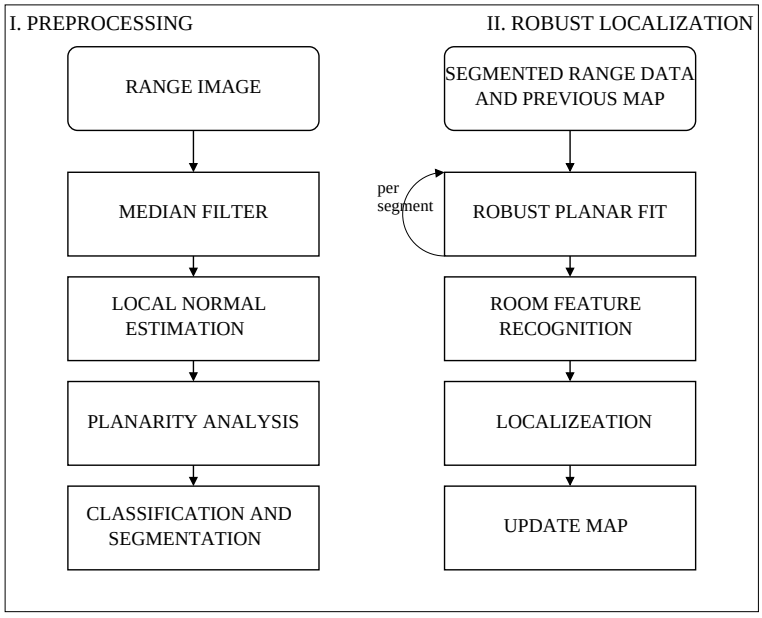

Figure 4.1: Flowchart of the Localization Algorithm

4.1 Range Image Segmentation

The divide and conquer approach is generally used in computer vision to "understand" images, i.e. they are usually segmented into regions of structures. For a rough; subject - foreground - background segmentation, range images offer a convenient and simple solution. By introducing a depth threshold, saying that everything that is farther than a value is background and closer than a value is foreground can yield all the separation that is needed[1].
A meaningful separation of all surfaces in range images to recover the true geometry of the environment is on the other hand a much more complicated issue. To

[1]This is utilized in the preprocessing of the pose estimation data in chapter 7.1

understand the 3D scene, the first step is usually partitioning it into structured parts according to eitherthe surface normals, the curvature of the surfaces, their connectivity etc. The research in this field is booming as the results are useful in so many fields, that use various types of range finders which all have their own attributes. Fields such as robotics, autonomous navigation, underwater navigation, remote sensing, surveillance technology etc.
The most widespread segmenting methods can be divided very roughly into two categories; those that use parametric model fitting and those that do not. The latter are methods that use edge-maps and region growing methods, e.g. based on distance thresholds and distance maps [18] to find depth discontinuities or surface normal inconsistencies. Lately the parametric methods have been more popular mainly because of their robustness to noise.
Computer vision systems are usually working in manmade environments which can be described by lines, planes, low-level curves and curved surfaces. These regions have simple mathematical representations, which the modeling methods find using various means. A functional parametric model algorithm can fit accurately in the presence of substantial noise or outliers and can deal with an unknown number of models in the data.
Probably the most difficult problems of the model approach is classifying the image into regions where a specific model should be fitted to the data. Lately, very advanced classifying-while-fitting algorithms have been developed ([27]) but they are all very complicated and enormously expensive computationally. Therefore simpler approaches where the regions are first classified are currently more popular, though the simultaneous approach is probably the future. This more basic approach is performed here by preprocessing the image, segmenting it into regions, according to the local surface normals, which can then be fitted individually.

4.2 Preprocessing

The preprocessing accomplishes the rough segmentation into planar areas with similar surface normal directions. The left side of figure 4.1 illustrates the preprocessing steps.

4.2.1 Obtaining the Local Surface Normals

To accurately obtain local normals in every point in a point cloud is not a simple task. especially if the surfaces are complicated and irregular. The analytical

approach, fitting continuous differentiable function to the data and calculating the derivatives analytically, is usually not an option as the grade of complexity is much to high because of noise, different surface shapes etc. A more general approach is finding the normal of the local plane i.e. obtaining the normal by fitting a plane to a small $N \times N$ neighborhood around the point. Many methods have been developed but the most general is the least square fitting method (LSQ) where the a plane is optimally fitted according to the sum of the squared error of all the points (this will be explained in detail later in this section). The main problem with LSQ is that is very vulnerable to outliers, i.e. one "bad point" can alter the whole outcome (this is illustrated in figure 4.2). Other so called robust methods have been proposed as they select which points are feasible to fit the plane to (the inliers) and omit the rest(the outliers). Such methods include: RANSAC[2], weighted least squares and least trimmed squares. All of these methods are on the other hand quite slow compared with directly using the LSQ method which is why it is used in this step.

The outlier problem causes blur in the edges if the neighborhood straddles an

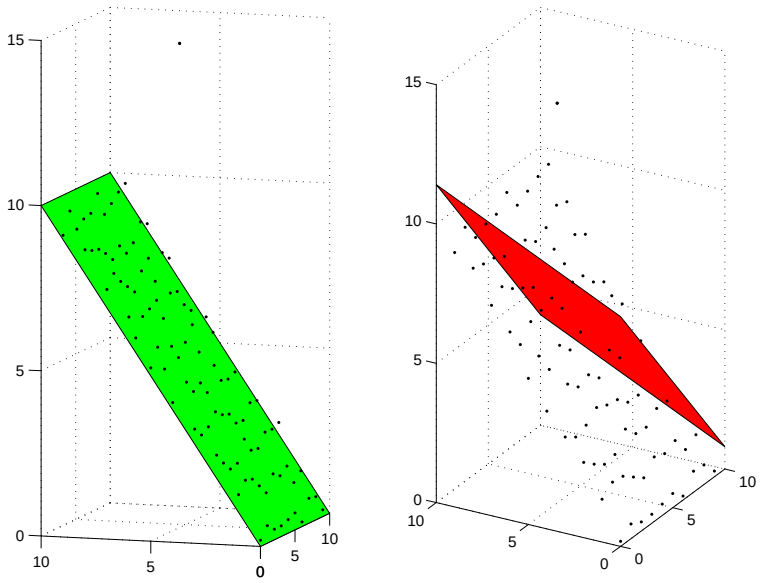

Figure 4.2: The Least Squares Outlier Problem. The left plot omits the outlier, the right one takes it into account.

edge. This can be avoided by preprocessing, constructing an edgemap first and avoiding straddling edges by simply skipping the points on the other side of the edges. Edges in range images are two kinds: step edges and roof edges (figure

[2]RANSAC will be discussed in detail later in this chapter

4.3). Step edges can be found by standard methods of thresholding the gradient as they are detected as fast spatial changes in the image i.e. spikes in the gradient image. Roof edges on the other hand are harder to detect. The distances

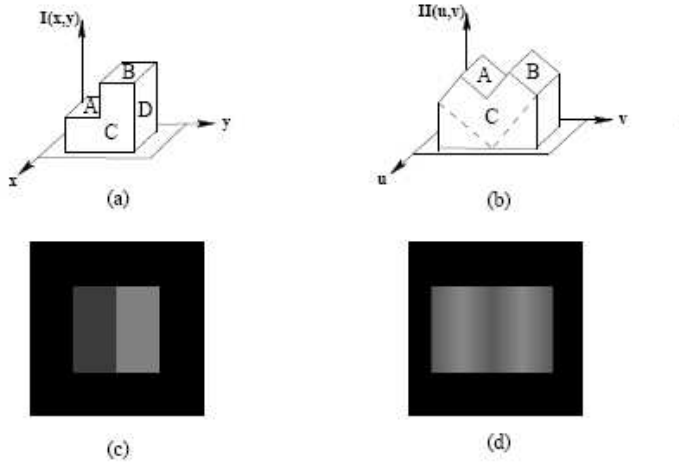

Figure 4.3: A Step Edge and a Roof Edge. An object viewed from different angles may result in different edge types.(from [34])

measured by the range finder on both sides of the edge are very close and are thus not detected in the gradient image. Many methods have been suggested in finding roof edges and were experimented in the progress of this work. Methods such as calculating the root mean square(rms) error for each local plane and refitting afterwards with smaller neighborhoods where the rms-error had past a threshold. This method is not successful in the SwissRanger images as the roof edges are very much blurred because of the multipath problem, causing smoothed roof edges in the images.
The normal finding implementation is thus; first filtering with a 3×3 median filter - as the median does not blur edges and reduces shot noise. Then the step-edges are found and used to guide the selection of neighborhood points: If a edge is in the neighborhood the points that are connected to the center point are used in the fitting. If the point is located on the edge a neighboring normal is used instead. This slowed down the process somewhat and is used more as an option. If the scene is cluttered the step edge method might give better results but if the step edges are few in the scene then their blurring isn't of much importance for the goal of this project.

As mentioned before then the least square solution to the general fitting problem

is the solution that gives the smallest sum of squared errors. Modeling the function $y(\mathbf{x})$ with a linear model:

$$y(x) = \sum_{i=1}^{d} w_i x_i = \mathbf{w}^T \mathbf{x} \tag{4.1}$$

where $\mathbf{w}$ is a weight vector to be optimized and $\mathbf{x} = (1, x_1, \ldots, x_d)^T$ thus restricting the solution to a $(d+1)$-dimensinal space. The generalized sum of squared errors for N points takes the form:

$$\begin{aligned} \varepsilon(\mathbf{w}) &= \frac{1}{2}\sum_{n=1}^{N}(y(\mathbf{x}^n;\mathbf{w}) - t^n)^2 \\ &= \frac{1}{2}\sum_{n=1}^{N}(\mathbf{w}^T\mathbf{x}^n - t^n)^2 \end{aligned} \tag{4.2}$$

Rewriting in matrix form where $\mathbf{X}^T = (x^1, x^2, x^3 \ldots, x^N)$ and $\mathbf{t} = (t^1, t^2, \ldots, t^N)^T$:

$$\varepsilon(\mathbf{w}) = \frac{1}{2}(\mathbf{w}^T\mathbf{X}\mathbf{X}\mathbf{w} + \mathbf{t}^T\mathbf{t} - 2\mathbf{w}^T\mathbf{X}^T\mathbf{t}) \tag{4.3}$$

This equation is quadratic in $\mathbf{w}$ and has a minimum where:

$$\mathbf{X}^T\mathbf{X}\mathbf{w} = \mathbf{X}^T\mathbf{t} \tag{4.4}$$

solving for $\mathbf{w}$:

$$\mathbf{w} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T t \equiv \mathbf{X}^\dagger t \tag{4.5}$$

Where $\mathbf{X}^\dagger$ is the pseudo-inverse which is the matrix where $\mathbf{X}^\dagger\mathbf{X} = I$ (in general then $\mathbf{X}\mathbf{X}\dagger \neq I$). Instead of calculating the pseudo-inverse directly the singular value decomposition (SVD) method gives a computationally attractive solution to solve this, especially when using computing languages optimized for matrix calculations such as *Matlab*. The SVD of $\mathbf{X}$ is defined as:

$$\mathbf{X} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T \tag{4.6}$$

Where $\mathbf{U}$ is a unitary matrix ,$\mathbf{\Sigma}$ is diagonal with nonnegative numbers on the diagonal and zeros off the diagonal and $\mathbf{V}$ is also a unitary matrix(i.e. the inverse is its transpose). Using the pseudo-inverse:

$$\mathbf{X}^\dagger = \mathbf{V}\mathbf{\Lambda}^{-1}\mathbf{U}^T \tag{4.7}$$

from equation 4.5 this leads to:

$$\mathbf{w} = \mathbf{X}^\dagger t = \mathbf{V}\mathbf{\Lambda}^{-1}\mathbf{U}^T \tag{4.8}$$

Hence SVD uncovers the least squares solution, and because of the nature of these 3 matrixes the invers are easily computed by transposing.

The least square plane fitting for N-points is thus solved by setting $\mathbf{X}$ as a $d+1=4$ dimensional matrix with the points:

$$\mathbf{X} = \begin{pmatrix} x_1 & y_1 & z_1 & 1 \\ x_2 & y_2 & z_2 & 1 \\ x_3 & y_3 & z_3 & 1 \\ \vdots & \vdots & \vdots & \vdots \\ x_N & y_N & z_N & 1 \end{pmatrix} \tag{4.9}$$

The optimal solution is then last column vector in $\mathbf{V}$ in the SVD of $\mathbf{X}$ giving:$\hat{\mathbf{w}} = (v_1 d, v_1 d, v_3 d, v_4 d)^T$. This gives the equation of the plane in the general form:

$$w_1\mathbf{x} + w_2\mathbf{y} + w_3\mathbf{z} + w_4 = 0 \tag{4.10}$$

and the normal is $\vec{n} = (w_1, w_2, w_3)^T$.

4.2.1.1 Planarity Analysis

The goal is to extract the planar features out of the range images. To focus the search on planar regions, it is helpful to classify the pixels as planar or not. These non-planar points can be noise, curved surfaces (objects) or caused by the multipath problem. Full scale curvature analysis that can handle statistical outliers etc. is very difficult and not the purpose of this step. Instead a simpler and efficient approach is used that numerically estimates the curvature at point p from its neighbors in a $N \times N$ neighborhood. The estimator derives the curvature from the change in the normal direction , in point p ($\vec{n}_p$) to the normal in neighbor point q ($\vec{n}_q$) [25].

$$\kappa(p,q) = \frac{\|\vec{n}_p - \vec{n}_q\|}{\|p - q\|} \tag{4.11}$$

This is the discrete approximation to the one-dimensional curvature along the curve between p and q and is accurate enough if the distance between p and q is short i.e. the Euclidean distance is close to the geodesic distance. Estimating $\kappa(p,q)$ for all q in the $N \times N$ neighborhood and taking the median gives a planarity measure pm_p:

$$pm_p = \text{median}_q\{\kappa(p,q)\} \tag{4.12}$$

This measure then states that the point is planar if it is under a threshold pm_L and non-planar if it is higher [11]. This threshold can be found to be characteristic for the data. The non-planar points are removed for the next step in the algorithm.

4.2.2 Classification of the Normals

After normalizing the surface normals to a unit length they can be described simply by the two angular components in the spherical coordinate representation; ϕ, the angle from pole-axis, and θ the angle from the positive x-axis in the xy-plane.

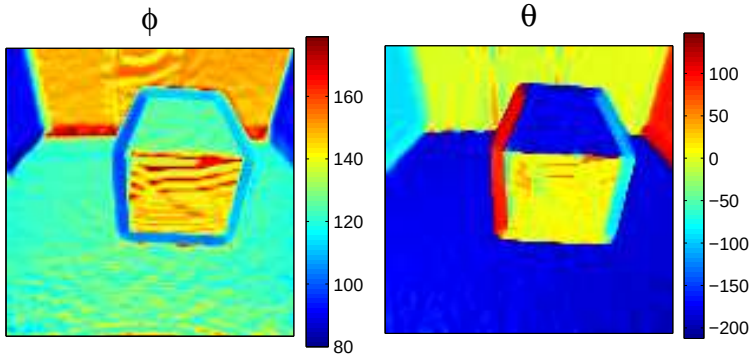

Figure 4.4: The Two Spherical Coordinate Components of the Local Surface Normals. Normals found without looking for step-edges.(Range data from USF database [17])

Figure 4.4 shows an example how the θ-component is much more discriminating factor between the surfaces than ϕ. This strongly indicates that the normal data's dimensionality can be reduced to only one component without missing to much information and thus simplifying the classification considerably. Principal Component Analysis (PCA) is used to select the optimal component for the angular data, projecting the points onto the component with the maximum variance[3]. Using the same example figure 4.5 shows the 1D angular component that contains 97.4 % of the total variance.

Taking the histogram of the projected data shows how the density takes the form of a Gaussian Mixture Model (GMM), i.e. a mixture (or sum) of M

[3]A detailed description of PCA is given in chapter 5.1.1

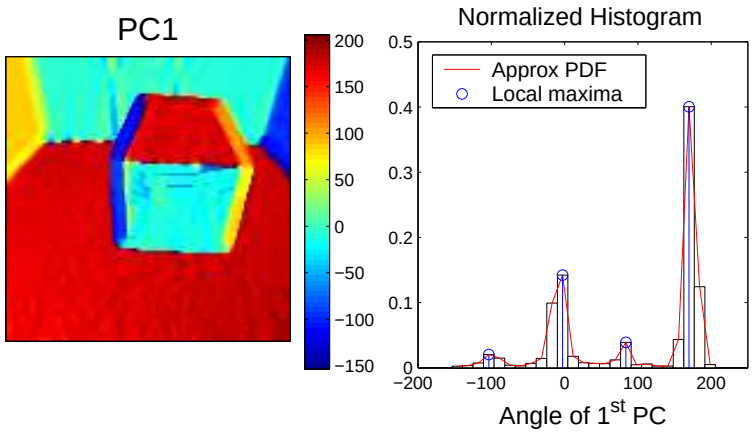

Figure 4.5: The projection of the Angular Data onto the 1^{st} Principle Component. Per Pixel and Normalized Histogram.

Gaussian curves with different parameters:

$$p(x|\mathbf{w}) = \sum_{j=1}^{M} P(C_j)p(x|j, \mathbf{w}_j) \tag{4.13}$$

where each component density i is a normal distribution with the mean and variance as feature parameters: $\mathbf{w}_i = \{\mu_i, \sigma_i^2\}$.

$$p(x|\mu_i, \sigma_i^2) = \frac{1}{\sigma_i\sqrt{2\pi}} exp\left(\frac{(x-\mu_i)^2}{2\sigma_i^2}\right) \tag{4.14}$$

$P(C_i)$ is the relative probability or priori of component density i and:

$$\sum_{j=1}^{C} P(C_j) = 1 \tag{4.15}$$

A simple probabilistic classifier called the Bayesian classifier is very effective on such data, calculating the posterior probabilities $p(w_i|x)$, i.e. the probability for x being part of class C_i. The GMM is found by estimating the μ_i, σ_i^2 and $P(C_i)$. The GMM is further discussed in the next section.
Bayes' theorem states:

$$p(\mathbf{w}_i|x) = \frac{p(x|\mathbf{w}_i)P(C_i)}{\sum_{j=1}^{M} p(x|\mathbf{w}_j)P(C_j)} \tag{4.16}$$

i.e. in general terms:

$$\text{posterior} = \frac{\text{likelihood} \times \text{prior}}{\text{normalization factor}}$$

The importance of Bayes' theorem lies in the fact that it states the posterior probability in terms of the conditional and prior probabilities that can be derived directly from the density function in equation 4.13.

Calculating the posterior probabilities for each point and choosing the class with the highest one, minimizes the probability of misclassification. This has been called the MAP decision rule or Maximum A Posteriori.

4.2.2.1 GMM Estimation

The estimation of the GMM can be done in numerous ways. In this case the easiest method is from the (smooth) normalized histogram, the local maxima are found located at the mean values μ_i with the normalized amplitude as the priori $P(C_i)$. The local maxima can be found by simply taking the difference between adjoining points on the curve thus approximating the derivative. Then by detecting the zero-crossings of the derivative the extrema are located. The variance is then set equal for each Gaussian at some high value found to be characteristic for the measurements.

The Expectation-Maximization (EM) algorithm is a more advanced method for

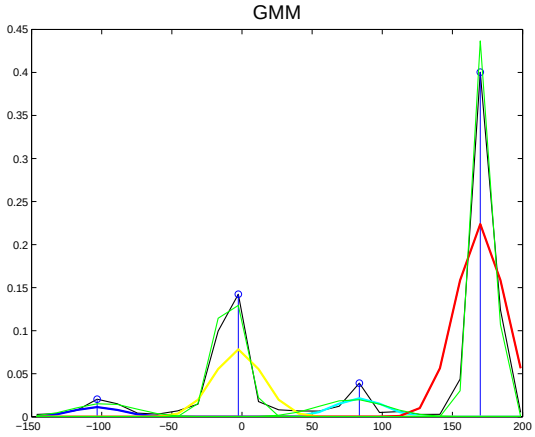

Figure 4.6: Estimation of the GMM. The black curve is the GMM, The colored are the estimations with equal variance (each color represents a plane in fig 4.5. Finally the green curve is the EM estimation with 4 iterations.

a better fitting GMM. It is an iterative maximum likelihood method that strives at increasing the likelihood of the parameters in each step [5]. Like all optimization problems this maximum likelihood problem starts with a costfunction. The costfunction for the GMM with dataset $\chi = \{x_1, x_2, \ldots, x_N\}$ in equation 4.13 is:

$$\begin{aligned} \varepsilon(\mathbf{w}) &= \sum_{n=1}^{N} -\log p(\mathbf{x}_n|\mathbf{w}) \\ &= \sum_{n=1}^{N} -\log \sum_{j=1}^{M} p(\mathbf{x}_n|\mathbf{w}) \end{aligned} \tag{4.17}$$

For the optimal parameters the derivative is found w.r.t. each of them. For μ_j it is:

$$\begin{aligned} \frac{\partial \varepsilon}{\partial \mu_j} &= -\sum_{n=1}^{N} \frac{\partial / \partial \mu_j \sum_{j'=1}^{M} p(\mathbf{x}_n|j') P(C'_j)}{p(\mathbf{x}_n||\mathbf{w})} \\ &= \sum_{n=1}^{N} P(C_j|\mathbf{x}_n) \frac{\mathbf{x}_n - \mu_j}{\sigma_j^2} \end{aligned} \tag{4.18}$$

The derivative w.r.t.the variance is:

$$\frac{\partial \varepsilon}{\partial \sigma_j} = \sum_{n=1} P(C_j|\mathbf{x}_n) \left(\frac{2}{\sigma_j} - \frac{(\mu_j - \mathbf{x}_n)^2}{\sigma^3} \right) \tag{4.19}$$

For the derivative w.r.t. the prior the constraint of equation 4.15 and the softmax[4] function with M auxiliary variables γ_j are used:

$$P(C_j) = \frac{\exp(\gamma_j)}{\sum_{j'=1}^{M} \exp(\gamma_{j'})}$$

Leading to the derivative:

$$\begin{aligned} \frac{\partial \varepsilon}{\partial \gamma_j} &= \sum_{k=1}^{M} \frac{\partial E}{\partial P(C_k)} \frac{\partial P(C_k)}{\partial \gamma} \\ \frac{\partial \varepsilon}{\partial P(C_k)} &= -\sum_{n=1}^{N} \frac{1}{p(\mathbf{x_n})} p(\mathbf{x}_n|k) = -\sum_{n=1}^{N} \frac{P(k|\mathbf{x}_n)}{P(C_k)} \\ \frac{\partial P(C_k)}{\partial \gamma} &= \delta_{k,j} P(C_k) - P(C_k) P(C_j) \end{aligned} \tag{4.20}$$

[4] or the normalized exponent

giving:

$$\frac{\partial \varepsilon}{\partial \gamma_j} = -\sum_{n=1}^{N} [P(C_j|\mathbf{x}_n) - P(C_j)] \tag{4.21}$$

Equating equations 4.18,4.19,4.21 to zero give the maximum likelihood solution:

$$\begin{aligned} \widehat{\mu_{\mathbf{j}}} &= \frac{\sum_{n=1}^{N} P(C_j|\mathbf{x}_n)\mathbf{x}_n}{\sum_{n=1}^{N} P(C_j|\mathbf{x}_n)} \\ \widehat{\sigma_j^2} &= \frac{\sum_{n=1}^{N} P(C_j|\mathbf{x}_n)(\mathbf{x}_n - \mu_j)^2)}{\sum_{n=1}^{N} P(C_j|\mathbf{x}_n)} \\ \widehat{P(C_j)} &= \frac{1}{N}\sum_{n=1}^{N} P(C_j|\mathbf{x}_n) \end{aligned} \tag{4.22}$$

For estimating this maximum likelihood derived above the EM-iteration scheme is used. The change in the costfunction when iterated is:

$$\begin{aligned} \varepsilon^{new} - \varepsilon^{old} &= -\sum_{n=1}^{N} \log \frac{p^{new}(\mathbf{x}_n)}{p^{old}(\mathbf{x}_n} \\ &= -\sum_{n=1}^{N} \log \left(\frac{\sum_{j=1}^{M} p^{new}(\mathbf{x}_n|C_j)P^{new}(C_j)}{p^{old}(\mathbf{x}_n)} \frac{P^{old}(C_j|\mathbf{x}_n)}{P^{old}(C_j|\mathbf{x}_n)} \right) \\ &\leq -\sum_{n=1}^{N}\sum_{j} P^{old}(C_j|\mathbf{x}_n) log \left(\frac{p^{new}(\mathbf{x}_n|C_j)P^{new}(C_j)}{p^{old}(\mathbf{x}_n)P^{old}(C_j|\mathbf{x}_n)} \right) \end{aligned} \tag{4.23}$$

Where the inequality is based on the Jensen inequality [5]. This is a upper bound problem that is minimized yielding the same results as equation 4.22 in iteration form. This is the M-step of the EM-algorithm:

$$\begin{aligned} \widehat{\mu_{\mathbf{j}}^{\mathbf{new}}} &= \frac{\sum_{n=1}^{N} P^{old}(C_j|\mathbf{x}_n)\mathbf{x}_n}{\sum_{n=1}^{N} P^{old}(C_j|\mathbf{x}_n)} \\ (\widehat{\sigma_j^{new}})^2 &= \frac{\sum_{n=1}^{N} P^{old}(C_j|\mathbf{x}_n)(\mathbf{x}_n - \mu_j^{new})^2)}{\sum_{n=1}^{N} P^{old}(C_j|\mathbf{x}_n)} \\ \widehat{P^{new}(C_j)} &= \frac{1}{N}\sum_{n=1}^{N} P^{old}(C_j|\mathbf{x}_n) \end{aligned} \tag{4.24}$$

[5] $log\left(\sum_j \lambda_j x_j\right) \geq \sum_j \lambda_j \log(x_j)$

The E-step of the algorithm is to re-calculate the posterior using Bayes' theorem:

$$p^{old}(C_j|\mathbf{x}_n) = \frac{P^{new}(C_j)p^{new}(\mathbf{x}_n|C_j)}{\sum_j P^{new}(C_j)p^{new}(\mathbf{x}_n|C_j)} \tag{4.25}$$

By initializing the EM-algorithm with the parameters found by the simple maxima search method a quite accurate solution is found, this is illustrated in figure 4.6 using the data example from before and 4 iterations.

4.3 Extracting and Recognizing Room Features

The most interesting features in the room are the floor and walls, i.e. planar surfaces close to the geometric extrema of the point cloud. From the results of the preprocessing the regions recognized as planar are fitted to mathematical parameterizations of planes.

4.3.1 Parametric Model Fitting

Parametric methods are used to efficiently describe manmade structures in images. Many methods have been developed through the years to fit data to mathematical structures while striving to solve the outlier problem mentioned earlier. These methods have been called robust estimators. One of the most popular of these methods is RANSAC which will be discussed in the next section.

4.3.1.1 RANSAC estimator

RANdom Sampling Consensus or RANSAC is a iterative robust estimator presented by Fischler and Bolles in 1981 [9] for model fitting in image analysis. Since then it has become a very widespread technique especially in the image analysis field. The algorithm is very simple as it chooses random points to fit a model and measures its quality. This is repeated until it has found the best model. The basic RANSAC algorithm follows:

Given:

The data $\mathbf{X}$ - a set of M datapoints

The model - a model that can be fitted to data points
n - the minimum number of data values required to fit the model.
k - the maximum number of iterations allowed in the algorithm.
t - threshold value for determining if a point fits the model.
d - number of data values required to determine that a model fits.
Algorithm:

1. Select n random points.
2. Fit them to the model.
3. Find inliers: points that fit to the model with less error than t. It's a good model if inliers are more than d.
4. Compare result with best previous. If error is better than the previous best; store this model and its error.
5. Update k and iterate. jump to step 1.

Return best model

The method only needs one user defined parameter t. d is not necessary if the all the points are evaluated.
RANSAC uses a probability based method to re-evaluate k the number of iterations. From $w = \text{inliers}/M$ the probability of a point fitting the model, then the probability of finding only bad points is $z = (1 - w^n)^k$. Solving this for k defines the maximum number iterations necessary:

$$k = \frac{\log(z)}{\log(1 - w^n)} \tag{4.26}$$

The basic RANSAC can be used for plane fitting by saying that $n = 3$ as a plane is defined by 3 points. These points, p_1, p_2 and p_3 then constitute as the current model. The inlier evaluation is done by calculating the distance $\mathbf{D}$ between each point and the plain:

$$\begin{aligned} \vec{n} &= (p_2 - p_1) \times (p_3 - p_1) \\ \mathbf{D} &= (\mathbf{X}) \cdot \vec{n} \\ \text{inliers} &= \text{points where abs}(\mathbf{D}) < t \end{aligned}$$

Where $\vec{n}$ is the normal to the plane. The threshold parameter t is found as a number charecteristic for the SwissRanger camera and its noise. This plane fitter is very robust and due to the preprocessing it finds quite fast well fitting models in relatively few iterations.

4.3.2 Recognizing the Room

A simple algorithm was made to construct a room map with planes as building blocks. The assumption is made that only three views are possible: the floor is always in the field of view and then it is possible for one, two or three walls to be in the view. The flowchart in figure 4.7 shows the decisions in recognizing the view. First the correct plane P_F must be identified as the floor. This is done by a simple grid search method:

1. Divide the range image into 4×5 squares.
2. Find which plane has the most inliers in center bottom square.
3. If there is only one: Return plane as floor.
   If there are none, goto next step.
   If they are two save as candidates P_a which is closer to the bottom, and P_b
4. Look in square above the last one
   If there is one or more and it is P_b and not P_a return P_a
   If it is a new plane goto next step.
5. Search square next to the first on the right side.
   Follow steps 2-4
6. Search square next to the first on the left side.
   Follow steps 2-4
7. Algorithm fails.

This has proved to work well given the assumption that the scene is not to cluttered and the floor is always in the view.
The left and right planes are identified in the same way just starting in the top left and right corners and then testing the next squares etc.

Choosing view model 1 is straight forward but models 2 or 3 depend on the intersection line between P_L and P_R. If the planes are close to parallel their intersection line should be far away or in infinity. If their intersection line is inside of the pointcloud view 2 is chosen. The intersection lines are found by first checking the plane normals: $\vec{n}_R \times \vec{n}_L < 10^{-10}$, which means they are close to parallel. If not, the line of intersection must be perpendicular to both $\vec{n}_R$ and $\vec{n}_L$, which means it is parallel to their cross product: $\vec{a} = \vec{n}_R \times \vec{n}_L$. To specify the line a point on it $\mathbf{x}_o$ is needed that satisfies:

$$\begin{aligned} \vec{n}_L \cdot \mathbf{x}_o &= -p_1 \\ \vec{n}_R \cdot \mathbf{x}_o &= -p_2 \end{aligned} \tag{4.27}$$

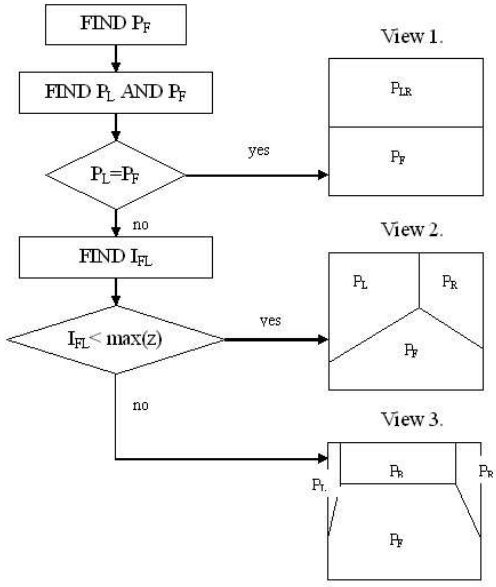

Figure 4.7: Choosing the Basic Possible Views of the Walls and Floor. P_F is the floor, P_L is the left side wall, P_B the backside wall and P_R is the right side wall

Where p_1 and p_2 are points on planes P_L and P_R

This can be solved by setting

$$\begin{aligned} \mathbf{m} &= (\vec{n}_R \vec{n}_L) \\ \mathbf{b} &= -\begin{pmatrix} p_1 \\ p_2 \end{pmatrix} \\ &\text{and} \\ \mathbf{m}\mathbf{x}_o &= \mathbf{b} \end{aligned} \tag{4.28}$$

Solving this for the point $\mathbf{x}_o$ on the line needs some linear algebraic tricks that end in the solution:

$$\mathbf{x}_o = \frac{p_1 \vec{n}_L \vec{n}_R \vec{n}_R - \vec{n}_R p_2 \vec{n}_L \vec{n}_R}{\begin{vmatrix} \vec{n}_L \vec{n}_L & \vec{n}_L \vec{n}_R \\ \vec{n}_L \vec{n}_R & \vec{n}_R \vec{n}_R \end{vmatrix}} - \frac{p_2 \vec{n}_R \vec{n}_L \vec{n}_L - \vec{n}_L p_1 \vec{n}_R \vec{n}_L}{\begin{vmatrix} \vec{n}_L \vec{n}_L & \vec{n}_L \vec{n}_R \\ \vec{n}_L \vec{n}_R & \vec{n}_R \vec{n}_R \end{vmatrix}} \tag{4.29}$$

If view 3 is identified the back plane P_B is located by searching in the top of the image and using the information of which planes have already been recognized. After the view has been determined then the intersection of the planes are all

found and the corners located by the intersection of the lines.
The Localization is then performed by triangulation: finding the cameras orthogonal distance to the planes and position relative to the corner. The distance from a point $\mathbf{x_0} =$ to a plane in generalized form: $ax + by + cz + d = 0$ is found by projecting the distance vector $\mathbf{w}$ onto the normal of the plane, $\vec{n}$. $\mathbf{w}$ is given by:

$$\mathbf{w} = -\begin{pmatrix} x - x_0 \\ y - y_0 \\ z - z_0 \end{pmatrix} \tag{4.30}$$

The distance D is then:

$$D = \frac{\|\vec{n}\mathbf{w}\|}{\|\vec{n}\|} = \frac{\|ax_0 + by_0 + cz_0 + d\|}{\sqrt{a^2 + b^2 + c^2}} \tag{4.31}$$

Finding the corner is also useful, which is possible from the intersecting lines found in 4.28. The crossing of two lines can be found from 4 points $\mathbf{x_1}, \mathbf{x_2}, \mathbf{x_3}$ and $\mathbf{x_4}$ then the two lines are:

$$\begin{aligned} \mathbf{x} &= \mathbf{x_1} + (\mathbf{x_2} - \mathbf{x_1})s \\ \mathbf{x} &= \mathbf{x_3} + (\mathbf{x_4} - \mathbf{x_3})t \end{aligned} \tag{4.32}$$

then eliminating s and t by:

$$\begin{aligned} \mathbf{a} &= \mathbf{x_2} - \mathbf{x_1} \\ \mathbf{b} &= \mathbf{x_4} - \mathbf{x_3} \\ \mathbf{c} &= \mathbf{x_3} - \mathbf{x_1} \end{aligned} \tag{4.33}$$

and then:

$$s = \frac{(\mathbf{c} \times \mathbf{b})^T(\mathbf{a} \times \mathbf{b})}{\|\mathbf{a} \times \mathbf{b}\|^2} \tag{4.34}$$

The intersection point is then by putting this into equation 4.32:

$$\mathbf{x} = \mathbf{x_1} + \mathbf{a}\frac{(\mathbf{c} \times \mathbf{b})^T(\mathbf{a} \times \mathbf{b})}{\|\mathbf{a} \times \mathbf{b}\|^2} \tag{4.35}$$

For evaluating the reconstructed room the dihedral angle is calculated to see if they are 90°. The dihedral angle between planes with normals $\vec{n}_1$ and $\vec{n}_2$ is:

$$\theta_d = \arccos(\vec{n}_1 \cdot \vec{n}_2) \tag{4.36}$$

4.4 Evaluation of the Algorithm.

The evaluation of the algorithm is done by using the algorithm to reconstruct several scenes using the SwissRanger and comparing with manual measurements. Quantitatively evaluating the efficiency of a range segmentation algorithm is a difficult task. Tailoring the algorithm to a certain dataset does not necessarily prove its robustness on a arbitrary range image - the algorithm might even be useless outside of the limited realms of the dataset. Also the manual measurements of the subjects can be a tedious task even to a few centimeters accuracy To avoid such bad practice the University of Southern Florida (USF) started a "open" project [6] [21] by setting up a database with range images with accurate manually specified "ground truths", together with evaluation and comparison tools. This way the algorithms results can be weighted and it is possible to compare the segmented results with the results from other segmenters. Various researchers all over the world have been using these databases and tools to evaluate their algorithms [8, 11, 28, 37, 3, 25]. In this way researchers not only can see how well their algorithm works, but also can compare with other segmenters by other researchers. This has resulted in a fruitful and comparative research environment. As this algorithm is aimed only at extracting planar room features only, the images from the database had to be chosen with this in mind. This limited the use to comparing the results to the ground truths without using the evaluation and comparison tools. These results could be compared with exact ground truth measurements unlike the images taken in the IMM image lab where all the manual measurements are not exact.

[6] http://marathon.csee.usf.edu/range/seg-comp/SegComp.html

Chapter 5

Pose Estimation

The pose of an object can be described by means of a rotation transformation which brings the object from a reference pose to the observed pose. In short, the pose says how the object is oriented in space.
Pose estimation is an active research field in computer vision and closely related to other very active topics such as object recognition, classification and face recognition. An enormous amount of papers and efforts have been dedicated to solving these problems and there is no ultimate solution on the horizon yet.

Various pose estimation techniques have emerged through the years, and short introduction to some of them is a good way to get feel of the problems vast area of research.
A widely used method in industry is CAD-model fitting such as has been done at *ScapeTechnologies*[1] [1]. There they use a CAD-Model to train a recognition engine which produces a database of features that are looked up in the real world image made from multiple images.
Extended Gaussian Images [20] is a method where the surface normals are mapped to sphere giving each position a unique presentation. This can be
Statistical approaches such as eigen-shapes [36] and optimal component projections have been showing very good results in face recognition and recently such

[1]Part of The Maersk Mc-Kinney Moller Institute for Production Technology at the University of Southern Denmark

research has been extended to range images [33] achieving good results with this 3D description method.
Such a statistical approach on range images is a new and exciting field. A field that the SwissRanger cameras fast retrievable range images can contribute to.
The key issue is to find an aspect that differentiates the objects orientation consistently. The method here is to train a model with a dataset of an object from many positions, reduce the dimensionality of the set to uncover the features that separate the different poses. An algorithm is made using a manifold learning technique called Local Linear Embedding (LLE) [31] which proves a powerful technique in revealing a high dimensional data true dimensionality.

5.1 Reduction of Dimensionality

One of the hottest topics in statistical machine learning is dealing with high dimensional data. A sequence of 100×100 pixel images can be seen as points in a 10000 dimension pixel space. If these images are in some way correlated it is likely that a much less dimensional feature space can be found; where the features in some way describe the sequence.

5.1.1 Principal Component Analysis

The classical approach to dimensionality reduction is Principal Component Analysis (PCA). PCA linearly maps the dataset to the maximum variance subspace by eigen-analyses of the covariance matrix, where the eigenvectors are the principal axes of the subspace, the eigenvalues give the projected variance of the data and the number of the significant eigenvalues gives the "true" dimensionality. The PCA transform can be expressed as follows [5]:

$$J = U_1^T I \tag{5.1}$$

Where J is the low-dimensional(d) data, U_1 is the projection matrix containing the principal components and I is the high-dimensional (D) input data. The principal components are solutions minimizing the sum-of-squares error of the

dimensionality reduction:

$$\begin{aligned} \varepsilon(u) &= \frac{1}{2}\sum_{i=d+1}^{D}\sum_{n=1}^{N}(u_i^T(x^n-\bar{x})^2 \\ &= \frac{1}{2}\sum_{i=d+1}^{D} u_i^T \mathbf{\Sigma} u_i \end{aligned} \tag{5.2}$$

where $\bar{x} = 1/N\sum(N=1)^N x^n$ is the mean of x and $\mathbf{\Sigma}$ the $D \times D$ covariance matrix of I.

To find U, Σ is SVD decomposed into:

$$\Sigma = U\Lambda U \tag{5.3}$$

U is a column-orthogonal matrix which contains eigenvectors corresponding to the eigenvalues in the diagonal matrix. U1 in equation 5.1 contains d columns (eigenvectors) corresponding to the d largest eigenvalues.

5.1.2 Nonlinear Manifold Learning

PCA's linear behavior is not optimal for all datasets. E.g. if the data set contains nonlinear structures they are invisible to PCA.
In the recent years attention has been growing towards techniques that assume that the data points lie on a low dimensional nonlinear manifolds in the high dimensional space. Figure 5.1 shows a "classic" manifold problem, the 3d swiss roll. Here the points lie on a 2D manifold in a 3D space, the goal is in an unsupervised manor to uncover the underlying 2D structure.

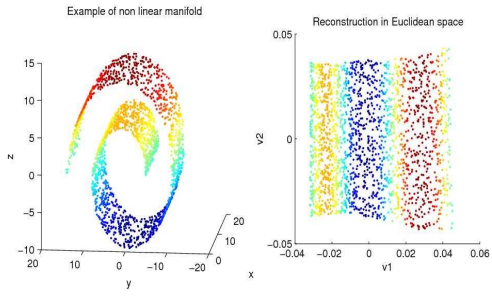

Figure 5.1: The Swiss Roll: The problem and the solution

These methods have also shown good results in diverse fields from face and 3d shape recognition ([31, 32, 35]) and robot navigation ([3]) to unrelated fields such as voice recognition([5]) and word document counts([31]) where the manifolds in the data are not obvious.

Among the most popular methods are the graph based methods Isomap ([35]) and Local Linear Embedding (LLE) ([31]). Isomap strives at finding intrinsic dimensionality while preserving geodesic distances between points. LLE embeds to the lower dimensionality while maintaining local geometry around points. LLE will be overviewed in the next section.

5.2 Local Linear Embedding

LLE is an elegant unsupervised, non-iterative method that avoids the local minima problems that plague many other methods. In figure 5.2 the algorithm is summarized into three steps for each point. For a dataset $\mathbf{X}$ with N points in D dimensions ($D \times N$), we will find an output $\mathbf{Y}$ of N, d dimensional points ($d \times N$) where $d \ll D$.

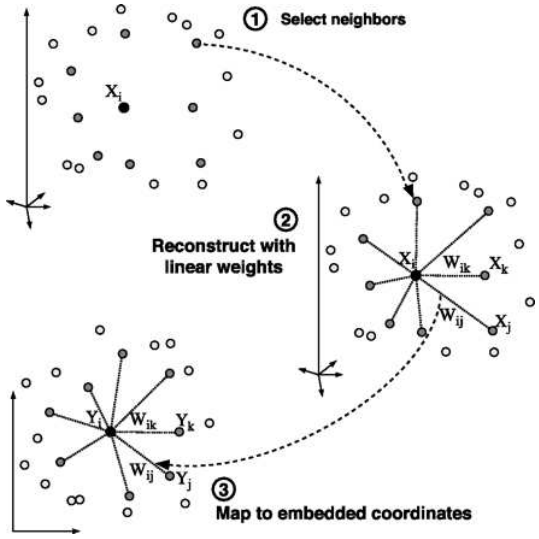

Figure 5.2: The Three Steps in the LLE Algorithm. (From [31]

The Three steps in the algorithm are as follows:
Step 1: Neighborhood search.

The K nearest neighbors are found. This can be done by various fast methods such as finding the Euclidian distances by normalized dot products (L_2 norms) constructing the $(N \times N)$ distance matrix Δ and finding the K least distances for each point.

Step 2:Constrained least squares fit.
Each point x_i is expected to lie on a close to locally linear patch of the manifold. Each sample is then approximated by weighted linear combination of its K nearest neighbors. To find the reconstruction weight matrix W the cost function is minimized w.r.t. W:

$$\varepsilon_I(W) = \sum_{i=1}^{N} \|x_i - \sum_{j=1}^{K} W_{ij} x_j\|^2 \tag{5.4}$$

this is solved under the constraints of $\sum_j W_{ij} = 1$ and $W_{ij} = 0$ if x_i and x_j are not neighbors.
The optimum weights are found by solving a least squares problem, where the numerical solution is computationally quite inexpensive.
Considering the reconstruction error in equation 5.4 for a particular data point x with K nearest neighbors η_j and reconstruction weights w_j. The error can be rewritten:

$$\epsilon = \|x - \sum_j w_j \eta_j\|^2 = \|\sum_j w_j (x - \eta_j)\|^2 = \sum_{jk} w_j w_k G_{jk} \tag{5.5}$$

Where G is the local Gram matrix:

$$G_{jk} = (x - \eta_j) \cdot (x - \eta_k) \tag{5.6}$$

In terms of the inverse Gram matrix, the optimal weights are given by:

$$w_j = \frac{\sum_k G_{jk}^{-1}}{\sum_{lm} G_{lm}^{-1}} \tag{5.7}$$

This is done efficiently without calculating the inverse of the Gram matrix by solving $\sum_k G_{jk} w_k = 1$ and then rescaling the weights so they sum to one.
These weights W are symmetric in the fashion that they are invariant to rotations, translations and rescaling of each data point and its neighbors. Hence these weights characterize the "natural" dimensions of each neighborhood and any linear decomposition can be done on the data without effecting them.

Step 3: Eigenvalue problem.
Using the weights W and the assumption that the data lies close to a nonlinear manifold $\mathbf{Y}$ of dimension $d \ll D$ there is a linear mapping (translating, rotating and scaling) that approximates a embedding from the D dimensions in each neighborhood to d dimensions of the manifold. The costfunction is minimized w.r.t $\mathbf{Y}$:

$$\varepsilon_{II}(\mathbf{Y}) = \sum_{i=1}^{N} \|y_i - \sum_{j=1}^{K} W_{ij} y_j\|^2 \tag{5.8}$$

Subject to that the embedded data has unit variance ($\sum_i Y_i Y_i^T / N = 1$).

This is solved by introducing the matrix $M = (I - W)^T \cdot (I - W)$. The costfunction of 5.8 is then rewritten in a quadratic form:

$$\varepsilon_{II}(\mathbf{Y}) = \sum_{i=1}^{N} \sum_{j=1}^{K} M_{ij}(y_i y_j) \tag{5.9}$$

M_{ij} is given by

$$M_{ij} = \delta_{ji} - W_{ij} - W_{ji} + \sum_{k=1}^{K} W_{ki} W_{kj} \tag{5.10}$$

where $\delta_j k = 1$ if $j = k$ and 0 otherwise.
M is sparse, symmetric and semipositive definite. The optimal solution to equation 5.9 are then the $d + 1$ eigenvectors of the costmatrix M corresponding to the smallest $d+1$ eigenvalues. The last eigenvector corresponding to the largest eigenvalue, is with all equal components and represents the free translation mode of eigenvalue zero.

5.2.1 LLE Extensions

Two extensions to LLE where investigated for the robot applications: classification and mapping of out-of-sample data.

5.2.1.1 Intrinsic Dimensionality and Classification

LLE is a "natural" classifier. Due to the unit variance constraint the separate classes samples are mapped separately as they lie on separate manifolds in the

data.
With C classes all samples of a certain class are mapped onto a single point in $C-1$ dimension. The choice of dimensions for the third step in LLE is then twofold; choice of the local intrinsic dimensionality d_L which is the dimension of the manifold each class lies on, and the choice of global intrinsic dimensionality which is then used in step 3:

$$d_G = C * d_L + (C-1) \tag{5.11}$$

Finding d_L can be a complicated matter as the residual variances of each component can not be measured such as is done in PCA and Isomap. On the other hand the intrinsic dimension can be shown using Isomap or by experiment in LLE to find what d_L describes the data. E.g. it has been shown in research ([31, 35]) that the dimensionality of a dataset made of images of a 3D object rotating 180° is only 1D and for the full 360° rotation 2D is needed to describe the change between the points.

5.2.1.2 "Online" Dimensional Reduction

LLE is not easily extended to out-of-sample data [2] and new images x_n can't simply be mapped to the low dimensional feature space. Calculating new LLE coordinates for each new image with a large training set is way to heavy computationally and therefore out of the question for most robot applications.
A simple method to map new data is some times called Local Linear Projecting (LLP, [13, 7, 32]). It utilizes the first two steps in LLE and omits the expensive third step. First it finds K neighbors of x_n in the training set $\mathbf{X}$ (*step 1*), then the weights are calculated as in *step 2* and these are used to make a weighted combination of the neighbors embeddings in $\mathbf{Y}$. The method thus exploits the local geometrical preservation quality of LLE. This has been proved as a quick and effective method with nice results.

5.3 Pose Estimation using LLE on Range Images

From the above it is clear that LLE is a powerful tool to uncover manifold structures in a dataset. Many researchers have used this to find structure in a set of intensity images. Range images give better descriptions of an objects 3D characteristics and it can be assumed that a model based on range images leads

to a stronger 3D description model than from normal intensity images.
The problem is then to prove this theory, to show that from a range dataset a robust model can be made to detect an objects 3D pose in space.

5.3.1 Evaluation of the Algorithm

To measure the effectiveness of the LLE pose estimator a quantative measure has to be introduced. In the given work scenario measuring an objects true orientation is problematic. Such accurate measurements call for a calibrated measuring devises and controlled environment. Instead the evaluation is done by extracting a part of the dataset and using that as a test set. These measurements are known to the degree of between which dataset points they were taken so their result position in the embedding are known.
Also for each range image dataset the intensity dataset is used as a comparison of estimators; which is a better an estimator based on intensity- or range images?

Part III

Experiments and Results

Chapter 6

Robot Localization Results

Here the results of the Robot localization experiments are presented. The purpose of these experiments were to test the range segmenting algorithm under different circumstances, measure the algorithms and cameras accuracy and overall

6.1 Data

The images were acquired by the SwissRanger in the Cartesian robot cage. The camera is set with a relatively long integration time using the auto integration time feature (between 25 and 35 msec). The view is set forward and downward so that at least the floor and one wall are in view. White cardboards and paper were used to cover some of the plexiglas and fence wire to simulate a "normal" room and also as the glass and metal have difficult reflection characteristics for the SwissRanger to handle.

6.1.1 Calibration and Accuracy

In the experiments made here some depth calibration had to be done to account for the scenarios reflection characteristics etc. By setting up the camera in front of the white cardboard and comparing the manually measured and camera results proved the depth calibration of the camera inadequate as a regularly linear error was measured[1][38]. The problem was fixed by recalculating the depth map using the linear relationship:

$$R' = 0.87 * R + 0.097 \tag{6.1}$$

where R is raw measurements from the camera in meters and R' is the corrected value. The Cartesian values where then calculated by using the same field of view calculations as are normally done on the cameras image processor.

To get an idea of the statistical spread of the data in the robot scenario. One hundred repeated measurements where taken on the cardboard surface and the average statistical behavior averaged over all of the pixels was $\sigma = 0.003m$.

6.2 Experiment 1: Step by Step

In this section a step by step segmentation using the algorithm presented in chapter 4 (figure 6.1). The first scenario is simple: only the robot cage with no obstacles (figure 6.2).

6.2.1 Preprocessing

Surface Normals

Figure 6.3 shows the smoothing effect of evaluating the normals with larger neighborhood value N. In general N=5 was used as N=7 was computationally very slow especially if the not straddling on step edge feature was used. In this part N=7 is though used.

Planarity and Classification Choosing a strict threshold for the planarity measure had positive effect on later results It is clear that the measurements of the left wall are to poor so that it can be detected as a planar surface. This is both

[1] This error is much higher than it should be and is currently being investigated by CSEM. It might be a problem regarding the specific camera used for these exercises.

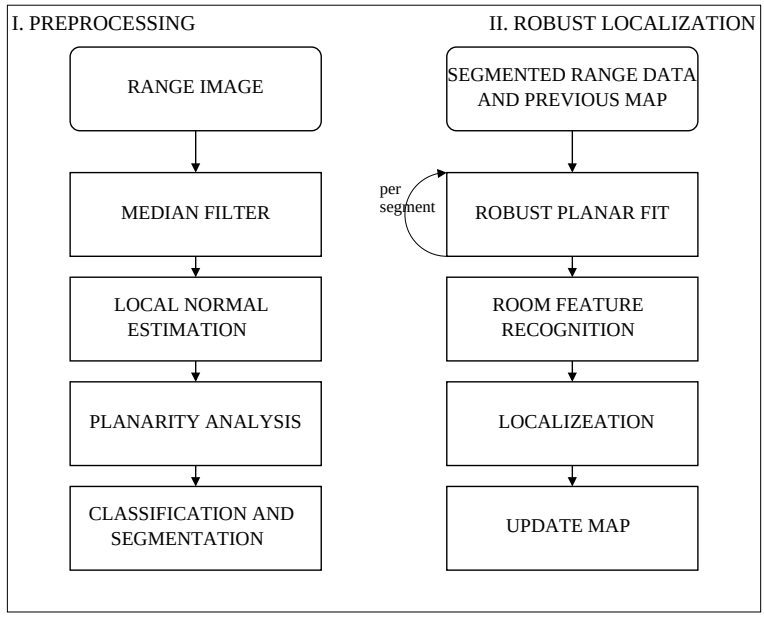

Figure 6.1: Flowchart of the Segmentation Algorithm

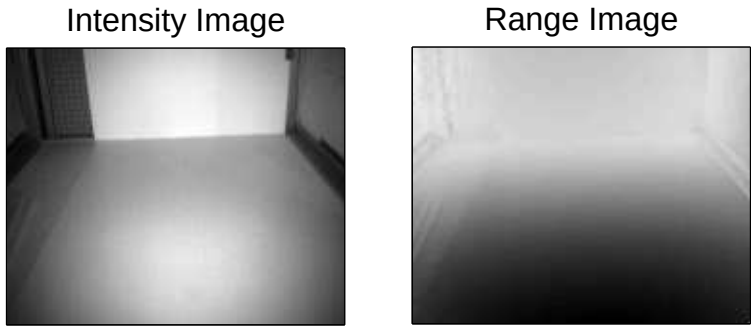

Figure 6.2: The First Scene.

due to the multipath problem and the measured intensity is simply to little. Equation 2.6 states that a good idea of the resolution can be made from the intensity values. As a comparison the intensity values in the back of the room are compare in table 6.1. The accuracy cannot be measured fully as the offset factor cannot be measured, it can just be assumed that it is equal in all pixels. The floor and back wall measurements are clearly giving much better accuracy then the darker regions that have around 4 times higher noise measures σ_R.

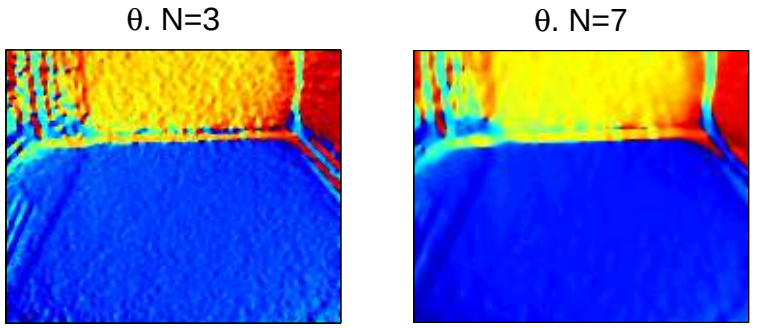

Figure 6.3: The θ Component of the Estimated Normals. Least square fitting the $N \times N$ neighborhoods.

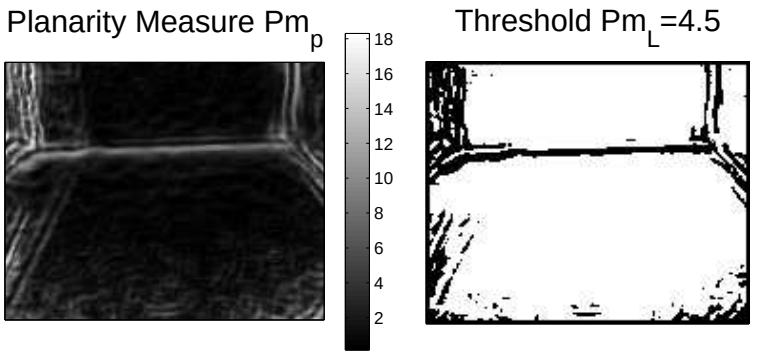

Figure 6.4: The Planarity Measure

Plane	Intensity Value	Noise σ'_R
Left wall	2400	5.5×10^4
Right wall	4900	2.7×10^4
Back wall	10000	1.3×10^4
Fence part	3300	4.0×10^4
Floor far	8100	1.6×10^4
Floor center	10000	1.3×10^4
Floor close left	3700	3.6×10^4
Floor close left	3900	3.4×10^4

Table 6.1: Typical Intensity Values in the Room (not given in any scale) and the Standard Deviation of the Noise Omitting the Offset Factor B.

The roof edges can be detected by this measure but they are not sharp as will be clearer in later steps. *Classification*
Figure 6.5 shows the classification results and the results of the GMM estimation methods are presented in table 6.2. The resulting class image are almost identical for both GMM estimation techniques. The EM is run three times and would fit even better with more iterations but this is not necessary with so few misclassifications present. Table 6.2 shows how the EM changes the GMM pa-

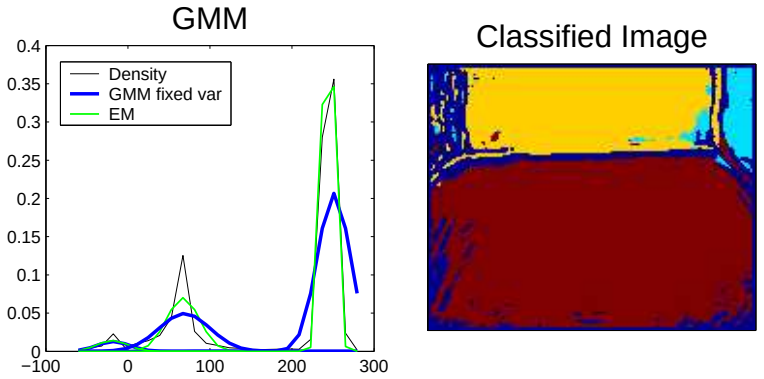

Figure 6.5: Classification into Planes. The GMM estimation and classified image. The classes are 3.

rameters from the initiation ones found with the max search. The $P(C_j)$ and μ_j only change a little while σ_j^2 changes drastically.

$P(C_j)$	μ_j	σ_j
Max Search		
0.0453	-17.8252	400
0.2487	67.0730	400
0.7061	251.0191	400
EM		
0.0463	-18.8470	338.5478
0.2733	69.0675	971.7886
0.6804	244.1862	49.1998

Table 6.2: GMM Parameter Results. The maximum search method and the EM algorithm

6.2.2 Extracting Features and Map Matching

Model Fitting
Now the planes need a mathematical representation which is found by RANSAC. RANSAC needs a threshold value for the model evaluation. This was chosen by experiment to be $t = 1 - 3cm$ as this gave quick results and good models. Figures 6.6,6.7 shows how the inliers and outliers for different values for t. The back wall is a cardboard poster and is curved in the center which explains the outlirs in that area. It is interesting to see that with t smaller, then the inliers are almost all in the zones where the accuracy according to the intensity image is best, this also supports that the t is correctly chosen.

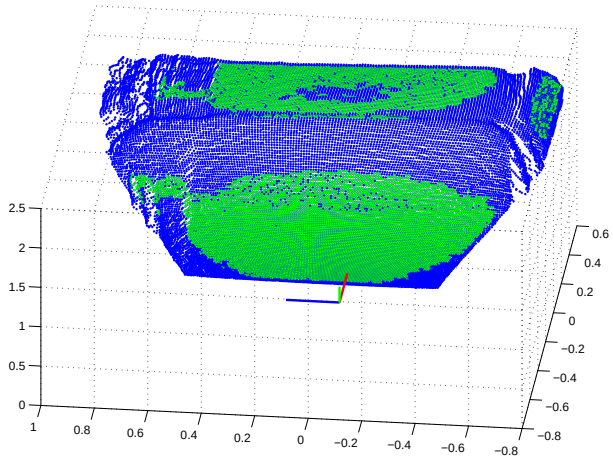

Figure 6.6: RANSAC $t = 1cm$. The inliers (green) in the 3 planes. (The axis are marked (x-blue, y-red and z-green)

From the figure it is also clear that the room is very distorted which is a problem in the discussed in the next part.
Reconstructing the Room
Reconstructing the room from three planes with the map algorithm in figure 4.7 is straight forward. The floor is recognized straight away (figure 6.8) and the intersections of the right and left plane is located within the boundaries of the image:

$$I_{LR} = \mathbf{x_o} + \vec{(a)}t = \begin{pmatrix} -0.68 \\ 0.54 \\ 2.06 \end{pmatrix} + \begin{pmatrix} -0.14 \\ 0.82 \\ -0.26 \end{pmatrix} t \tag{6.2}$$

Meaning the line passes through $\mathbf{x_o}$ which is closer than the extreme point in

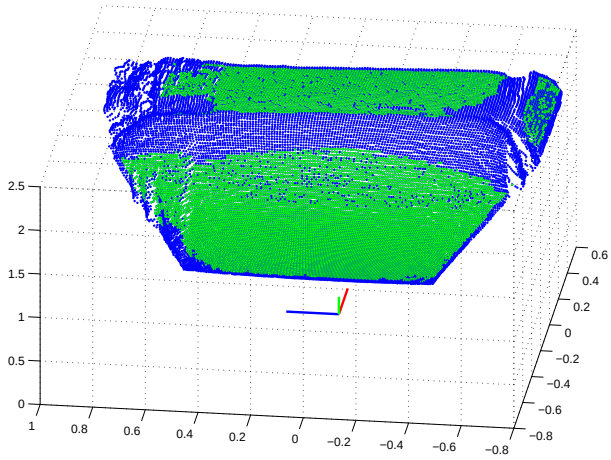

Figure 6.7: RANSAC $t = 3cm$. The inliers (green) in the 3 planes.

the z-direction: $(-0.170.102.18)^T$ and View 2 is used.

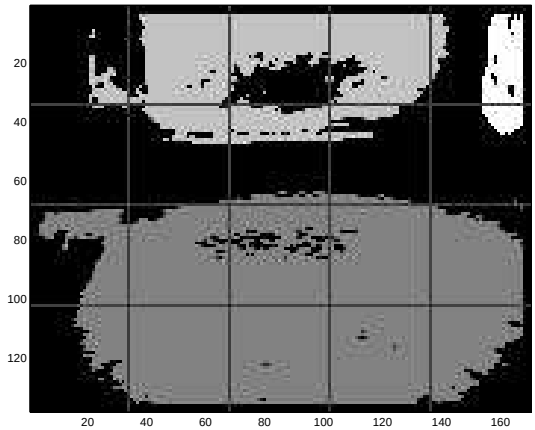

Figure 6.8: The Searchgrid for the View Recognition.

Localizeation
The planes are now used to build the room. In figure 6.10 the inliers, planes and intersection lines are plotted and the dihedral angles are given in table 6.3

Finishing the localization by finding the orthogonal distance from the camera's origin to the planes gives the ground plan position the results are in table 6.4.

Dihedral Angles		
Floor	Back Side	90.5541°
Floor	Right Side	99.41°
Back Side	Right Side	119.52°

Table 6.3: The Dihedral Angles Between the Planes.

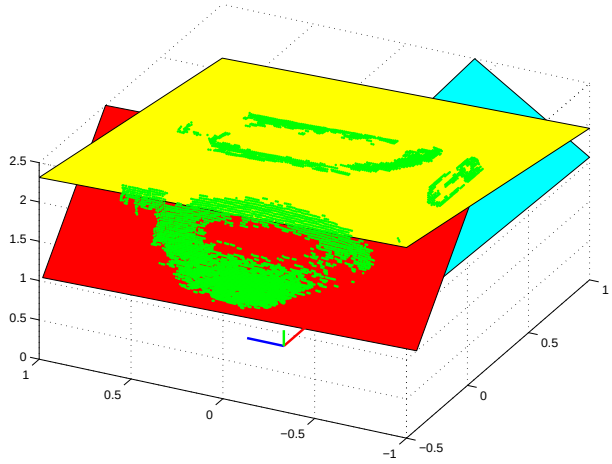

Figure 6.9: The RANSAC fitted planes

[m]	SwissRanger	Measured
Back Plane	2.09	1.99
Floor Plane	0.74	0.70
Right Plane	1.73	0.65
Corner	2.32	
Corner to Floor Point	2.18	2.10

Table 6.4: The Room Localization, The manual measurements are also subject to error.

Problems
It is clear that the rooms reconstruction is bad(tables 6.3 and 6.4 . The dihedral angles for the right side show great error and the distance measure from this wall is awful. The dihedral angle is 120° between the two walls, and the right side is 100° from the floor. The explanation lies partly in the accuracy and calibration but mainly in the multipath problem. Room corners are evidently quite difficult for the SwissRanger.

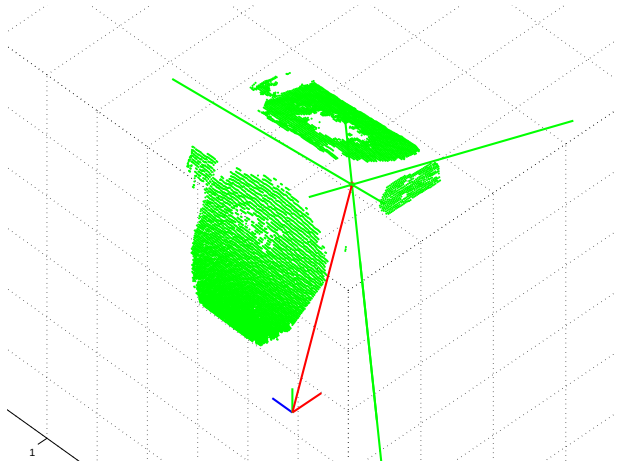

Figure 6.10: The Intersection Lines, the Corner Point and the Corner Line

Workarounds
One possible workaround to get a better localization is to triangulate the distance from the corner and taking the worst line out of the calculations. The multipath error is less innermost in the corner it self, it can thus be assumed that the corner point itself is more correctly positioned than the right wall. Knowing that the measurement from the floor and back side wasn't bad it is used to calculate the distance from the corner to the orthogonal projection of the camera onto the floor. Giving a much better Y- measurement in the robot-coordinates. The accuracy is still not good.

6.3 Scene 2: The Robot Moves

In this section the algorithm is tested after adding an object and moving the robot 11cm in X-direction and 4.5 in Y-direction in robot coordinates. The "ground truth" measurements are done manually with inaccurate tools and are subject to error.
The results are very similar to those before in the preprocessing and robust fitting (figure 6.11). The obstacle did not interfere in the classification nor plane fitting. The errors are still bad and the corner measurement methods fails to capture the exact motion. the cameras floor point should closer according to the equation$\sqrt{11^2 + 4.5^2}) = 11.88$, however this distance is only 3 cm (table 6.6. The distances from the floor and back side are though consistent with the motion made i.e. the corner has moved.

Dihedral Angles		
Floor	Back Side	89°
Floor	Right Side	97.39°
Back Side	Right Side	112.5°

Table 6.5: Scene 2: The dihedral angles between the planes.

[m]	SwissRanger	Measured	Calculated from last pos
Back Plane	1.95	1.98	1.98
Floor Plane	0.79	0.70	0.70
Right Plane	1.73	0.65	1.78
Corner Line	2.28		
Corner to floor point	2.15	1.99	2.06

Table 6.6: Scene 2: Robot has moved, X: 11 cm and Y=4.5,

6.4 USF Database Images

The images that are used are from a small Odetics scanner which measures the depth using the same time-of-flight principle as the SwissRanger, but with a pulsating laser using much more emitting power and a tilting mirror that scans the field of view in seconds. [16, 8, 17]. It is very interesting to see the results from such a laser based range finder and compare it with the SwissRanger (figure 6.12).

Figure 6.12 shows how the preprocessing steps located the planes easily and the classification worked smoothly. The robust fitting with 1 cm threshold resulted in some outliers on the floor. Localization in figure 6.13 and in table 6.8 proved excellent with a measurement error of 8 mm in the worst measurement.

Dihedral Angles	USF	Data
Floor	Back Side	92.6219
Floor	Right Side	90.47°
Back Side	Right Side	87.3009°
Back Side	Left Side	94.2549°
Floor	Left Side	92.6755°

Table 6.7: Wall Angles in USF Data

[cm]	USF	Ground Truth
Back Plane	125.8327	125
Floor Plane	52.0715	52
Right Plane	56.3332	56
Left Plane	56.3632	56

Table 6.8: Localizeation Data for the Odetics Range Image

Figure 6.11: Scene 2: The range image, classified image and inlier image with search grid

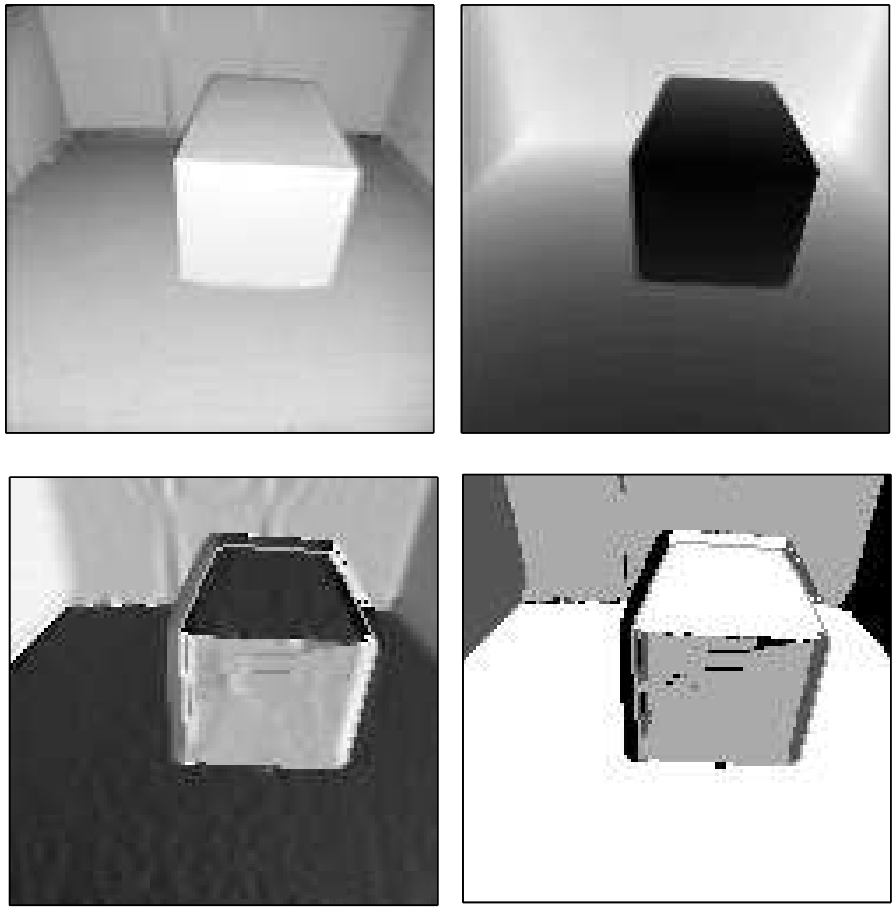

Figure 6.12: Images from the USF Database. Reflection, range, angular component and classified image. (Acquired by [19])

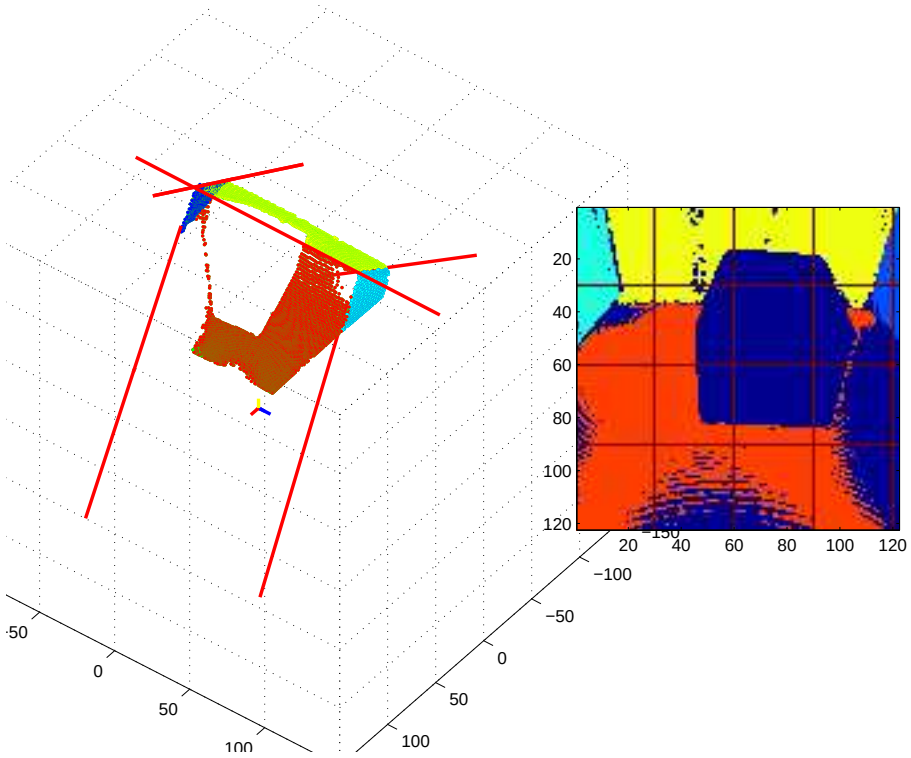

Figure 6.13: The Inliers and Intersection lines.

Chapter 7

Pose Estimation Results

In this chapter the results of the pose estimation experiments are presented. The experiments were designed to show various aspects and robustness of the LLE pose estimation method, such as classification and measurements of out-of-sample data. In all cases separate models are made from the range and intensity data, then the models are compared to see which model gives better results; the intensity image with its 2D texture information or the more 3D describing range model.

7.1 Data

For the pose estimation problem experiments were done on two datasets. The first was a cardboard box with wooden knobs attached to it (figure 7.1). This object can lie on 3 sides and is without symmetrical features. The images were acquired of the box while it was rotated 360° in each lying pose with 2° intervals (540 images in each dataset). This dataset was purposely made to make a model that could detect on which side a new image of the object was lying and detecting its orientation at the same time.

The second dataset was of a rotating mannequins head. The human face and head is very interesting 3D shape in computer vision and also it is a more challenging object than the box as it had more differences in color, reflection

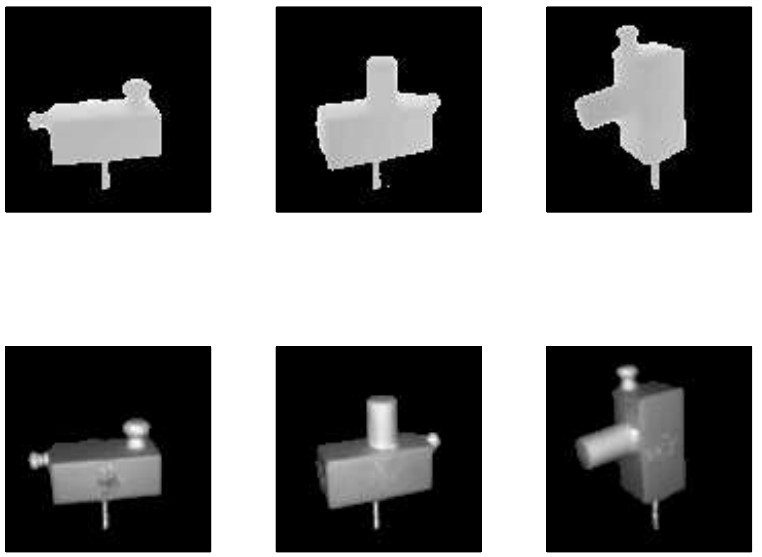

Figure 7.1: The Box Dataset. The upper row is from the range dataset and the lower is from the intensity dataset. One image from each class is shown.

shininess and texture. It too was imaged from 180 angles.

Both datasets are made of 100×100 pixel images that have been pre-processed

Figure 7.2: The Mannequin Head Dataset. Range and intensity images.

to set the focus on the subjects. This was done by subtracting the background by using a depth threshold in the range images and then applying the subtraction to the same pixels in the intensity images. Finally the datasets are vektorized to a 10000 dimensional pixel space where each image is a point:

7.2 Experiments

7.2.1 3D Object Lying on Different Sides

The two first components of both datasets in figure 7.3 shows that the classes can be separated but the method completely fails to capture the regularity of the objects change in orientation.

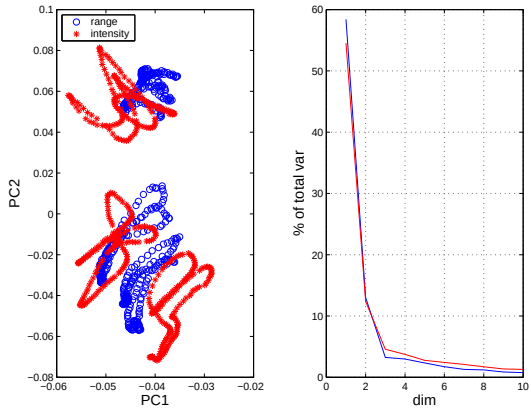

Figure 7.3: PCA of the Box data. The two first principle components cover 71% and 67% of the variance of the range- and intensity data.

The LLE embedding in 2D gives a perfect separation by mapping all the 180 data points of each class to three points (figure 7.4). The three 2D local dimensional embeddings are shown in figure 7.5. The 360° rotation of the objects are captured perfectly in all instances with circular curves.

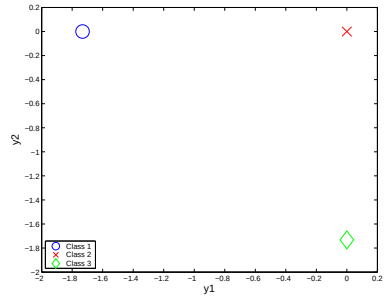

Figure 7.4: 2D LLE seperates the data perfectly into three points

7.2.1.1 Mapping Out-of-Sample Data

For online pose detection out-of-sample data is mapped onto the low dimensional embedding. This done by using the Local Linear Projecting technique described on page 47. Two separate experiments were performed to test this mapping.

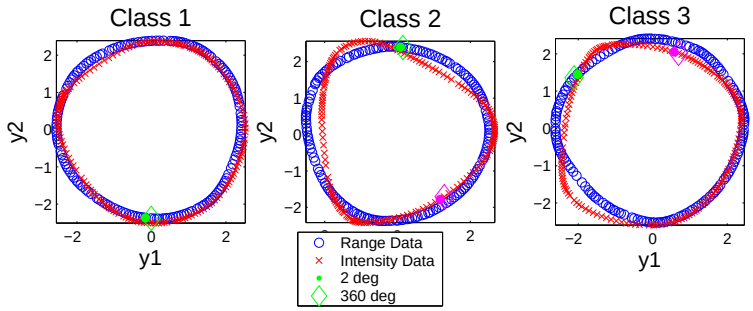

Figure 7.5: The local intrinsic dimensionality for each class. The start and end points are marked with a dots and diamonds for each curve.

Mapping a test set made of a subset of the whole data
Here a test set was made by randomly choosing 30 positions from the whole data set and using the rest as training to make the embedding. As the test points positions are known relative to the training points it can be measured if the points are mapped between the correct points.
Figure 7.5 show the local dimensions of the training data. Table 7.1 shows how all the 30 test points are correctly projected between their true neighbors.

Mapping a translated image and finding the closest match
Here a new data point was mapped to the embedding. The image was acquired after moving the camera from the position where the training data was gathered. This experiment gives an idea of the robustness of the models. On the other hand it gives results that are difficult to measure accurately and are in a way subjective. Still the results are valid as observations and test the quality of the models.

True pos.	Range	model pos.	Int	model pos.
2	1	3	1	3
20	19	21	19	21
38	37	39	37	39
56	55	57	55	57
74	73	75	73	75
92	91	93	91	93
110	109	111	109	111
128	127	129	127	129
146	145	147	145	147
164	163	165	163	165
182	181	360	181	360
200	199	201	199	201
218	217	219	217	219
236	235	237	235	237
254	253	255	253	255
272	271	273	271	273
290	289	291	289	291
308	307	309	307	309
326	325	327	325	327
344	343	345	343	345
362	361	363	361	363
380	379	381	379	381
398	397	399	397	399
416	415	417	415	417
434	433	435	433	435
452	451	453	451	453
470	469	471	469	471
488	487	489	487	489
506	505	507	505	507
524	523	525	523	525

Table 7.1: Box Data Mapping Results. The true positions of the test points and the neighboring positions in each model.

7.2.2 Rotating Head

Mapping a test set made of a subset of the whole data
In this experiment 30 points at random are chosen as test data the other 150 are used to find the embedding $\mathbf{Y}$. Figure 7.6 illustrates how the range model is much smoother while the intensity curve has overlaps. Table 7.2 shows the test points true positions and the two closest positions in each model.

The discontinuity in the intensity curve leads to an error when a test point

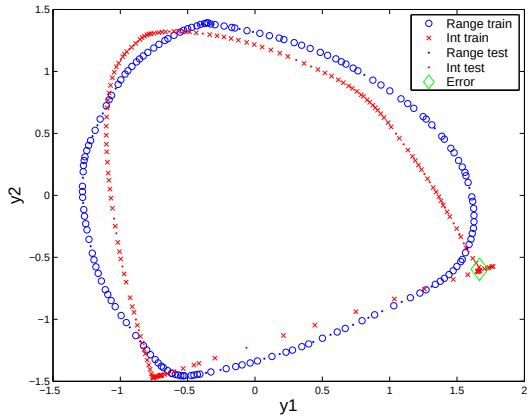

Figure 7.6: The LLE Embeddings for the Head Data and the test datas projections

is mapped close to this overlapping. The point that from position 120 is closer to points 125 and 132 on the embedding than its true neighbors 119 and 121. These false neighbors are shown in figure 7.7

Figure 7.7: Test image from position 120 and the false neighbors on the intensity embedding.

Mapping New Data from Different View Angles
To test further the robustness of the algorithm, new data points that were acquired from slightly different camera angles are mapped. The training data points and new data points are normalized to minimize the effect of the shift of the camera. Figure 7.8 shows how the range model does a good job in finding close neighbors while the intensity model has problems.

This was experimented from various views at different angles: Both models performed well until the positions came close to the intensity curves discontinuity. If the change in view was very drastic both models failed.

True pos.	Range	model pos.	Int	model pos.
2	1	3	1	3
4	3	5	3	5
8	7	9	7	9
10	9	11	9	11
30	29	31	29	31
34	33	35	33	35
36	35	37	35	37
38	37	39	37	39
46	45	47	45	47
50	49	51	49	51
56	55	57	55	57
68	67	69	67	69
70	69	71	69	71
90	89	91	89	91
92	91	93	91	93
98	97	99	97	99
100	99	101	99	101
105	104	106	104	106
107	106	108	106	108
110	109	111	109	111
112	111	113	111	113
114	113	115	113	115
116	115	117	115	117
120	119	121	125	132
130	129	131	127	128
140	139	141	139	141
150	149	151	152	155
160	159	161	159	161
175	174	176	174	176
177	176	178	176	178

Table 7.2: Head Data Mapping Results. The true positions of the test points and the neighboring positions in each model.

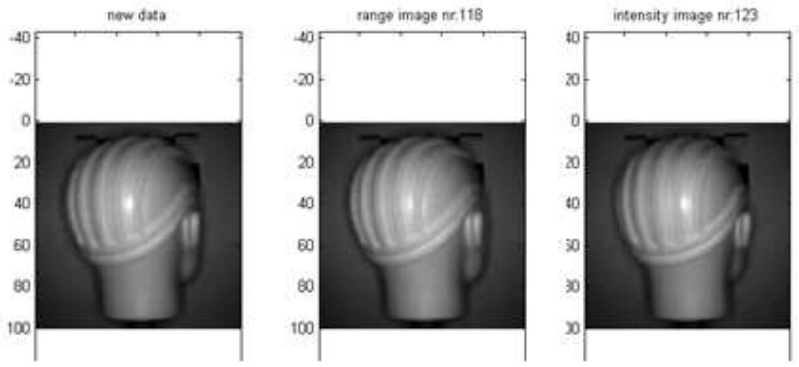

Figure 7.8: New Data acquired from Different Angle. Range model still performs correctly while the intensity model fails.(The range images intensity image is shown in this figure for better visual notice).

Part IV

Conclusion

Chapter 8

Conclusion

The SwissRanger SR-3000 range finder camera was tried in two different scenarios. It proved as a a somewhat limited localizer and 3D plane finder, but was on the other hand an excellent pose estimator.

8.1 Robot Localization

A algorithm has been developed to recognize room structures and localize a robot in an indoor environment. The algorithm successfully locates the walls and floor in the range data, giving them a mathematical parameterization that is used to calculate the robots position. The mayor problem is that the measurements from the camera are distorted and the accuracy in some regions is very poor and in others a additive error due to multipath that distorts the measurements grossly. These problems are connected to the basic design of the SwissRanger. It is a low energy active rang finder and as such is vulnerable against mainly because of:

- Accuracy: Is poor if the measuring light reflects poorly due to distance or reflection characteristics.

- Multipath: Measuring light reflects through many paths from corners adding error to the measurements.

The second factor is especially hard to quantify and has to be kept in mind in all solutions were the SwissRanger is to play part. This is not a problem with high power laser range finders as the experiments with the Odetics camera revealed; yielding results with accuracy up to a few millimeters. On the other hand the LRF have their own shortcomings especially in robotics; with their long latency and hazardous laser measurements. The results with the SwissRanger were not bad when only looking at the floor and back side wall, the relative motion between to positions were correct although the measured distance were not, which might well be a calibration issue which is under investigation by CSEM. The solution to this problem is thus to set stricter conditions on measurements, the off center pixels can not be relied on for measuring corners where the light scattering is to much. The solution is moving the camera avoiding the bad measurement conditions.

8.2 Pose Estimation

LLE proved a robust method to detect the intrinsic dimensionality of the data. It could easily find its classes and map the gradual changes between the samples to a low dimensional feature space. In all cases it outperformed the classic PCA method. The projecting of the new data onto the intrinsic dimension proved as a good pose detector and classifying tool, especially robust using the range image dataset.

The smoothness of the embedding curves gives a clear idea of the quality of the model. In all cases the range curves were smoother and especially with problematic data the range model looked much better.

Although LLE has few parameters they have to be chosen carefully. The dimension of the embedding has to be known. If it is chosen to high it starts to map noise and if it is to low it can't map the manifold properly. Other methods such as Isomap and PCA have means of determining the true dimension of the embedding. On the other hand Isomap is a poor classifier. The other parameter is K; the number of neighbors. This was not a problem in this case and didn't change much if the K was changed from 5 to 15 or more.

8.3 Future work

For a functional robot demonstration these two applications need to be connected by a robot navigation planner i.e. after the localization and room mapping, the robot should plan its path to close in to an object, which it can then estimate its pose and finally do something with it. Integrating an grasping hand or suction pick up tool for a bin picking demonstration would be an exciting use of the work that has been done.

Many improvements can be made on the algorithms, using new advanced technology that expand the usefulness of the SwissRanger even further. The most difficult problem with the localization problem was caused by the multipath error. This could be somewhat avoided if the camera would scan the room, taking multiple images that would then be matched together giving larger multipath-error free regions and relying more on the central pixels with the most accurate readings. Such matching algorithms for range images have been proposed using e.g. Iterative Closest Point (ICP [4]) based methods or orthogonal rectification of the normals [26].

The map matching algorithm here uses a very simple approach, this could be improved by indexing the data by 3D kD-trees making the wall search at the image boundaries a more optimal and robust search.
The obstacle search can be also be further improved by e.g. An Occupation grid approach would also be useful for a navigation system with obstacle avoidance. Lately many robust estimator have been suggested as improvements to the RANSAC algorithm. These methods are capable of fitting quadric and hyper quadric models to data thus modeling many types of curved surfaces (M-SAC is such a method). Other methods are better as do not rely on user set parameters in finding the inliers. Methods such as ASSC [37] is capable of automatically finding the parameters it uses.
The good results in the Pose estimation problem using the machine learning techniques indicate that there is plenty of potential in this field. As the 3D quality of the data in the pose problem was to the benefit of the 3D recognition one can imagine that this could apply in the active fields such as object tracking, face recognition etc.

Appendix A

CSEM SwissRanger Camera Specifications

In this appendix some chosen specifications of the SwissRanger SR-3000 camera are summarized. Most of these specifications and diagrams are acquired from the manual [6].

A.1 SwissRanger Dimensions, System Parameters and Performance

Parameter	Ratings	Units	Conditions
Illumination			
Number of LEDs	55	-	
Modulation Frequency	20	MHz	
Total Emitted Optical Mean Power during Integration	<1	W	@ 20 MHz Modulation @ room temperature
Peak Wavelength	850	nm	@ room temperature
Bandwidth FWHM	35	nm	@ room temperature
Emission Angle	±25°		@ room temperature
Lens			
Focal length	8	mm	
Aperture	f/1.4		
Horizontal fov	47.5°		
Lateral fov	39.6°		
Optical Filter			
Center Wave Length	870	nm	
Bandwidth FWHM	80	nm	(Full
Sensor			
Number of active Pixels	176 x 144		
Demodulation Frequency (default)	20	MHz	
Dynamic Range Modulated Light	70	dB	
Background light suppression ability	400	$W/m^2/\mu m$	corresponds to 50% of the maximal sun light
Pixel Size	40 x 40	μm^2	
Output Data			
Range Data	Spheroidal Cartesian	R [mm] x,y,z [mm]	
Intensity Data	Gray scale		

Table A.1: SwissRanger SR-3000 System Parameters.

Parameter	Ratings	Units	Conditions
Best Case Standard Deviation	< 0.3	cm	@ 20 MHz Modulation
Non-ambiguity Range	750	cm	@ 20 MHz Modulation

Table A.2: SwissRanger SR-3000 System Performance.

Distance [m]	0.3	1	2	3
Frame Rate [Hz][1]	29	20	15	12
Resolution [mm]	2.5	6	13	22

Table A.3: Depth Resolution of Central Pixel. Acquisition at room temperature; target with 90% reflectivity, 20 MHz modulation frequency (From [6])

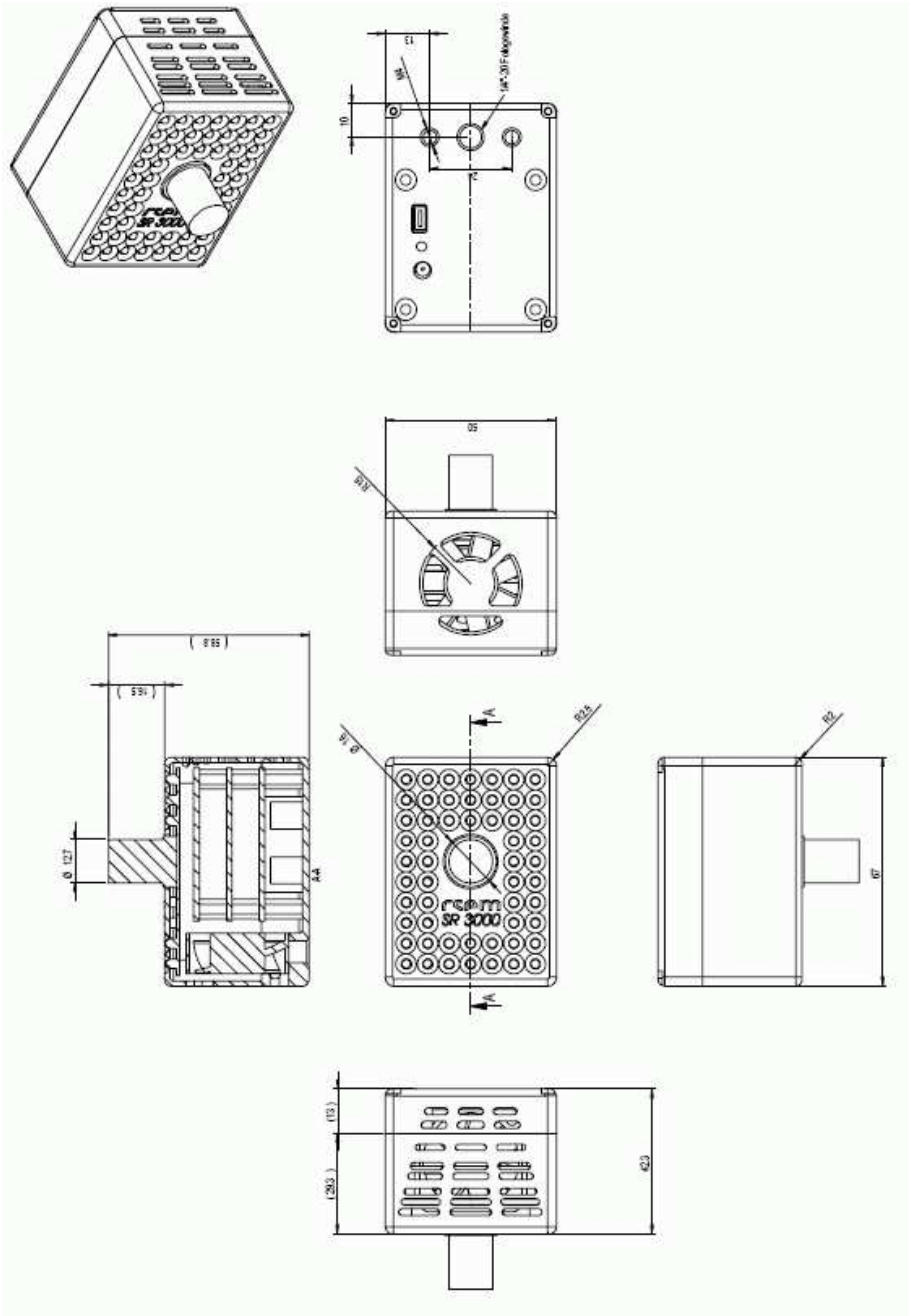

Figure A.1: Drawing of the SR-3000 Dimensions (From [6])

Appendix B

Cartesian Robot Code

B.1 Robot C++ class

The C++ programming is built on the work of Gunnar Harðarson [15, 14] in 2005. He continued the work of Jørgen Folm-Hansen's [10] in 1996. Gunnar had connected a IO controller to the Robot and could then control it from a Windows PC. It was controlled via a command prompt where a operator could send motion commands and program predefined routes.
To add to the robots functionality and to be able to interact with a camera and *Matlab*, the C++ code had to be changed to a class structure, more robot motion member functions where added, operations for the Sony EVI-D31 pan-tilt camera where integrated.
In the following the two classes functions and brief descriptions of the Matlab interface functions.

B.1.1 Robot Class

Details about the Robot and Sony VISCA Classes can be found in [30]

B.1.2 *Matlab* interface

The work on this feature has started but is yet to be completed at the time of this theses. The design is simply to make a dll-library that can be loaded by *Matlab*, which then execute the robot functions: `Move(x,y,z)` and the Pan-Tilt functions : `Pan2Pos(a)` and `Tilt2Pos()`. These functions are already functional in the console.

B.1.3 Rotating table

The imagelabs rotating table has three functions stored; one making the table turn 1°, the second turns it 2° and the third turns 10°. These programs are triggered through the controllers digital input switches which are connected to the IO board. These switches can be turned on from the C++ interface or by loading the IO boards library directly by *Matlab*. The function `SendRotTable(1)` turns on input switch 1 which activates a program stored on the rotating tables controller. See further in [30].

Bibliography

[1] I. Balslev and R. Larsen. Scape vision, a vision system for flexible binpicking in industry. *IMM Industrial Visiondays, DTU*, 2006.

[2] Y. Bengio, J. Paiement, and P. Vincent. Out-of-sample extensions for lle, isomap, mds, eigenmaps, and spectral clustering. Technical Report 1238, Université de Montréal, 2004.

[3] P. Besl and R. Jain. Segmentation through variable-order surface fitting. *IEEE Transactions on Pattern Analysis and Machine Intelligense*, 10:167–192, 1988.

[4] P. Besl and R.D. McKay. A method for registration of 3d shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligense*, 14:239–259, 1992.

[5] C. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, 1996.

[6] CSEM. *SwissRanger SR-3000 Manual*, 2005.

[7] D. de Ridder and R. Duin. Locally linear embedding for classification. *TUDelft, Pattern Recognition Group Technical Report Series*, PH-2002-01, 2002.

[8] A. Hoover et al. An experimental comparison of range image segmentation algorithms. *IEEE Transactions on Pattern Analysis and Machine Intelligense*, 18:673–689, 1996.

[9] M.A. Fishcler and R.C. Bolles. Ransac random sampling consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of ACM*, 26:381–395, 1981.

[10] Jørgen Folm-Hansen. Item-xyz, control board description, 1996.

[11] P. Gotardo, O. Bellon, K. Boyer, and L. Silva. Range image segmentation into planar and quadratic surfaces using an improved robust estimator and genetic algorithm. *IEEE Transactions on Systems, Man and Cybernetics-Part B: Cybernetics*, 34:2303–2316, 2004.

[12] O. Hall-Holt and S. Rusinkiewicz. Stripe boundary codes for real-time structured-light range scanning of moving objects, 2001.

[13] J. Ham, Y. Lin, and D. Lee. Learning nonlinear appearance manifolds for robot localization. *IEEE Pacific Rim Conference on Communications, Computers and signal Processing*, pages 2971–2976, 2005.

[14] Gunnar Harðarson. Xyz-robot, special course, 2004.

[15] Gunnar Harðarson. 3d measurement using camera. Master's thesis, IMM-DTU, 2005.

[16] A. Hoover. Perceptron images: The camera and models. Technical report, University of South Florida, 1995.

[17] A. Hoover. Range imege segmentation comparison project, 1995.

[18] A. Hoover. *The Space Envelope Representation for 3D Scenes.* PhD thesis, University of South Florida, 1998.

[19] A. Hoover, J. Jones, and O.H. Dorum. Perceptron images. Technical report, Oak Ridge National Laboratory, 1995.

[20] Berthold K. P. Horn. *Robot Vision (MIT Electrical Engineering and Computer Science).* The MIT Press, 1986.

[21] http://marathon.csee.usf.edu/range/seg comp/SegComp.html. Range image segmentation comparison project.

[22] http://www.3d shape.com/.

[23] http://www.minolta3d.com/.

[24] http://www.tyzx.com/.

[25] A.K. Jain and P.J. Flynn. On reliable curvature estimation. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 1:110–116, 1989.

[26] J.Bauer, K. Karner, A. Klaus, and R. Perko. Robust range image registration using a common plane. *Proceedings of 2004 WSCG*, 3:2155–689, 2004.

[27] B. Kamgar-Parsi and H. Wechsler. Simultaneous fitting of several planes to point sets using neural networks. *Computer Vision, Graphics, and Image Processing*, 52:341–359, 1990.

[28] K. Pulli and M. Pietikäinen. Range image segmentation based on decomposition of normals. *Proceedings of the 8th Scandinavian Conference on Image Analysis*, 2:893–899, 1993.

[29] Dandapani Radhakrishnan and Illah Nourbakhsh. Topological localization by training a vision-based transition detector. In *Proceedings of IROS 1999*, volume 1, pages 468 – 473, October 1999.

[30] Sigurjón Árni Guðmundsson. The image lab cartesian robot, sony dvi-d31 camera and rotating table. Technical report, IMM-DTU, 2006.

[31] S. Roweis and L. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290:2323–2326, 2000.

[32] S. Roweis and L. Saul. Think globally, fit locally: Unsupervised learning of low dimensional manifolds. *Journal of Machine Learning Research*, 4:119–155, 2003.

[33] A. Srivastava, X. Liu, and C. Hesher. Face recognition using optimal linear components of range images. *Elsevier Image and Vision Computing*, 24:291–299, 2005.

[34] C.J. Sze, H.M. Liao, H.L. Hung, K.C. Fan, and S.J. Lin. Multiscale edge detection in range images via normal changes. *National Science Counsel of Taiwan*, 1:1–25, 2004.

[35] J.B. Tenenbaum, V. de Silva, and J.C. Langford. A global geometric framework for nonlinear dimension reduction. *Science*, 290:2319–2323, 2000.

[36] M. Turk and A. Pentland. Eigenfaces for recognition. *Journal for Cognitive Neuroscience*, 3:71–86, 1991.

[37] H. Wang and D. Suter. A model-based range image segmentation algorithm using a novel robust estimator. *3rd International Workshop on Statistical and Computational Theories of Vision*, 2003.

[38] J.W. Weingarten, G. Gruener, and R. Siegwart. A state-of-the-art 3d sensor for robot navigation. *Proceedings of 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 3:2155–689, 2004.