Edwin T.J. van der Thiel

Location Estimation: Estimation Maximization on Omnidirectional Camera Images

Edwin Theodorus Johannes van der Thiel
email: etjthiel@science.uva.nl
Artificial Intelligence - MultiModal Intelligent Systems
Informatics Institute, Faculty of Science(FNWI)
University of Amsterdam(UvA), The Netherlands
Ter verkrijging van de graad van Master of Science

Location Estimation: Estimation Maximization on Omnidirectional Camera Images

Supervisor: Dr. Ir. Ben J.A. Kröse

August 31, 2004

Abstract

In this thesis we will first describe the general problem of robot localization and map building. Then we will describe some relevant solutions known in literature. After this we build up to our solution, starting with data known in a global reference frame, such that linear regression is sufficient. When orientation is added, an iterative process is needed. Last of all links obtained from omnidirectional camera images are used. These links are normalized, so we need to obtain the scaling factors of these links as well.

Our solution uses Expectation Maximization. This is an iterative technique based on maximum likelihood estimates which has given good results in numerous areas already. It is also used on a dataset obtained by a real robot to show its usefulness in practice.

Contents

1 Introduction 4

2 Simultaneous Localization And Mapping 7

2.1 SLAM using Global Map Building 7
2.2 SLAM using Link Estimation from Local Maps 11
2.3 Conclusion 13

3 Estimation of Positions based on Relative Measurements 14

3.1 Estimating Locations from translation data 14
3.2 Extension to a Nonlinear Measurement Equation 17
3.3 An Iterative Approach 18
3.4 Conclusion 19

4 The Omnidirectional Camera as Sensor 24

4.1 Omnidirectional Camera Images: A Description 24
4.2 Transformation of Pairwise Omnidirectional Camera Images to Links 26
4.3 Issues in Combining Camera Images 27
4.4 Conclusion 27

5 Location Estimation from Pairwise Image Links 29

5.1 The Dataset 29
5.2 Estimating the Scaling Factors 30
5.3 Removal of wrong correspondences 33
5.4 EM on Normalized Links 33
5.4.1 EM in Location Estimation incorporating Omnidirectional Camera Datasets 33
5.5 Experiments 35

6 Conclusions 39

A 40

Notations

In every method discussed the measurements were taken in discrete intervals, the interval between two adjacent sensor readings. Time i means the time the i-th sensor reading was made, assuming that when a sensor reading was made in the initial pose, its index is 0. Unless mentioned otherwise, a pose $\mathbf{x}_i$ consists of x- and y-coordinates and an orientation θ.

$\bar{A}$	The estimation of A, where A can be any quantity
C	The complete covariance matrix of the estimation of D
C_{ij}	The covariance matrix of the estimation of $\mathbf{d}_{ij}$
d	The dimensionality of the data
$\mathbf{d}_{ij}$	The link from position $\mathbf{x}_i$ to position $\mathbf{x}_j$
D	The set of all links
D_i	The set of links from pose $\mathbf{x}_i$ to (all) other poses
H	The mapping between links and poses
i, j, k, l	Indices
$\mathbf{m}_i$	The i-th element in the map
M	The complete map of the environment
M_i	The map built from the first i elements, e.g. landmarks, features, ...
N	The total amount of measurements
$P(A)$	The prior probability of A, where A can be any quantity
$P(A\|B)$	The probability of A given B, where A and B can be any quantity
r_e	The scale of a local link estimate $\bar{\mathbf{d}}_{ij}$
$R(\theta)$	The rotation matrix for a rotation of θ degrees
$T(x)$	The translation matrix corresponding to the translation vector x
$\mathbf{u}_i$	The i-th control vector
U	The set of all control vectors, derived from the odometry
U_i	The set of control vectors up to time i
W	The Mahalanobis distance
x, y	The coordinates of a point in space
x_i, y_i	The coordinates of point i in space
x_{ij}, y_{ij}	Parameters of a link $\mathbf{d}_{ij}$
...	...

$\mathbf{x}_0$	The initial pose, which is also the origin of the global reference frame
$\mathbf{x}_i$	The robot pose at time i
$\mathbf{x}_{ik}$	The k-th estimate of the robot pose at time i
X_i	The set of robot poses up to time i
X_N	The set of all poses of the robot in the global reference frame
$\mathbf{y}_i$	A feature vector determined from an image $\mathbf{z}_i$
Y	The set of feature vectors determined from images Z
Y_i	The set of feature vectors determined from images Z_i
$\mathbf{z}_i$	The observation made at time i
$\mathbf{z}_{ki}$	The observation of an element (e.g. landmark, feature) at time i
Z	The set of all observations
Z_i	The set of observations up to time i
α	The matrix containing all scaling factors d times on its diagonal in the order given by D
Γ	The complete scaling matrix for D
Γ_i	The scaling matrix for the matrix D_i
Γ_{ij}	The scaling matrix for the link $\mathbf{d}_{ij}$
γ_{ij}	The scaling factor for the translation parts of a link $\mathbf{d}_{ij}$
Θ	The set of all local orientations
θ_i	The orientation at pose i
θ_{ij}	The orientation difference between pose i and pose j
σ_{ij}	The standard deviation of a covariance matrix

Chapter 1

Introduction

A robot that wants to operate in the real world often needs information about its location. This information needs to be as accurate as possible, using minimal resources. It is not possible, however, to get the true position in the real world, because every sensor has its own inherent error. The problem is how to get the most accurate location estimation using the available sensor data.

From a robot's point of view, the sensors can be divided into three separate categories, based on the characteristics of the data they produce.

1. An interoceptive sensor measures movement of a robot by registering odometry (the movement of either the wheels or legs). This can be done either by a piece of software that intercepts the commands sent to the servo motors, or by a physical sensor that monitors the wheel movement. From this information, it is possible to construct an estimation of the complete path.

   The error inherent in this odometry data is caused by slip, due to effects such as gravel on the road or uneven terrain, which adds a vertical movement that can not be derived from observing rotation of the wheels alone. These odometry errors are cumulative, so when a robot has to operate by itself for a long time without any feedback regarding its true position, the uncertainty will grow to the point where the current location can be any location in the world.

2. To reduce the error in the odometry dataset, the robot needs to observe the outside world. For this purpose an exteroceptive sensor, mounted on the robot, is used. From the scans made by the robot, locations of objects in the world with respect to the local robot reference frame are derived. A wide range of exteroceptive sensors exist. Cameras, laser

range sensors, and radars are the most common ones. Each type of sensor measures a certain aspect of the environment and has its own limitations. For example a laser range sensor can not monitor movement of an object, but gives accurate information about the distance to an object. In contrast to the error of the interoceptive sensor, in an exteroceptive sensor all measurements are independent of one another. As a result the errors are not cumulative but constant. They only depend on the distance between the sensor and the object that is scanned.
There are several approaches which use the information of exteroceptive sensors to estimate the position.

- In appearance-based models [5], feature vectors $Y_i = \{\mathbf{y}_0 \dots \mathbf{y}_i\}$, and their corresponding positions $\{\mathbf{x}_0 \dots \mathbf{x}_i\}$ serve as a map. The feature vectors are derived from images $Z_i = \{\mathbf{z}_0 \dots \mathbf{z}_i\}$ using dimensionality reduction, such as Principal Component Analysis [4]. Furthermore the mapping M, that maps images to low-dimensional feature vectors, is stored.
  The mapping is then used to reduce any observation $\mathbf{z}_k$ made by the robot to a feature vector $\mathbf{y}_k = M \cdot \mathbf{z}_k$. A kernel density estimation is then used to compute the pose which gives the maximum likelihood for the feature vector given the map.

- It is also possible to store a complete map of the environment as a geometric model [10]. When an observation is made, all the straight edge segments are extracted. This representation of the location is then matched against all known map edges. The region of the map that gives the highest probability of being the correct region is then selected. Since the location of the local map is now fixed within the global map, the estimated location is the origin of the local map.

- Another effective approach is to store a set of landmarks which can be easily recognized [16]. The profile of the landmarks, in this case a set of edges, is stored in a database along with the absolute positions. The robot can then match his observations to the database and estimate his position by e.g. triangulation.

- An occupancy grid is a nice alternative when you wish to use multiple sensors [13]. Here each sensor determines for itself whether a location is occupied and therefore out of reach, after which it is easy to integrate the data from several sensors. In this case it is done using a simple lookup table that returns whether a grid point is occupied based on the different sensor results. The overall results can then be compared to a map, where the most probable location is chosen as the optimal estimate.

- In the case where no model of the world exists, a model can be made based on the observations. Accuracy of that model increases over time. This is because the same place is usually visited more then once, which should result in a match between the two sensor readings. Now the overall error can be reduced. This approach is known as Simultaneous Localization And Mapping (SLAM). An extensive report on SLAM is given in chapter 2.

3. The third option is to remove the exteroceptive sensor from the robot and place it at some fixed position in the real world. Errors have the same properties as with the exteroceptive sensors mounted on the robot, except they have the benefit of operating in a limited area. When the sensor detects the robot, its position can be determined either by a single sensor or by multiple sensors using e.g. triangulation.

This thesis focuses on SLAM for a mobile robot, equipped with an omnidirectional vision sensor. This sensor has the property that, when two images taken at two different poses contain enough corresponding features, the *relative* pose for one location as observed from the other location (a link) can be derived.

In chapter 2, an overview of the work related to SLAM is given. Chapter 3 will show a solution to the problem of location estimation in a simulated environment. First we will show how to estimate locations in the case that the orientation is fixed. This means that the sensors provide relative translations between the poses. The problem is then extended to a world where the orientation for each pose is variable. In this case the sensor provides both the translation and the relative orientation between the poses. Two methods are provided for this set, one that divides the data in subparts and one that uses an iterative approach.

These methods assume that the links are already given. In chapter 4 it is shown how the links can be computed in the case that an omnidirectional camera is used as an exteroceptive sensor. Following in chapter 5 is a method that uses Expectation Maximization (EM) to get the global positions using both the links from odometry and the links that were estimated using an omnidirectional camera.

Chapter 2

Simultaneous Localization And Mapping

To be truly autonomous, a vehicle has to be able to start at an unknown location in an unknown environment, and then to incrementally build a map of this environment, while simultaneously using this map to compute vehicle location [9].
There exists an extensive amount of literature on this subject, which can be divided in two categories. The techniques in the first category build an explicit, global map of the environment based on the sensor readings. This requires the recognition of landmarks from the sensor readings. Each time a new sensor reading is recieved, both the map and the robot path are updated. In the second category feature vectors derived from each sensor reading are stored along with the locations $\mathbf{x}_i$ where these sensor readings were made. When there is sufficient overlap between two feature vectors, a link is established and the geometric difference is derived. The combination of odometry data and the links will then form an overfitted network from which all poses $\mathbf{x}_i$ can be calculated.
Even though in most cases sufficient data is available to derive a global map, the data is used only to get the links between poses locally. The representation of the map is a set of local maps for each pose $\mathbf{x}_i$ in the path, where the local map is the feature vector derived from the scan $\mathbf{z}_i$.

2.1 SLAM using Global Map Building

Localization, Mapping and the Simultaneous Localization and Mapping (SLAM) Problem
Durrant-Whyte [9] describes the process of SLAM as localization of the robot

based on the landmarks, and mapping based on the poses of the robot. A landmark is an object that can be distinguished from (most) other objects in the environment.

Localization is done by fixing the map, which is a set of landmarks here, and derive the location of the robot based on the observation of the landmarks observed in that position. Mapping is done by assuming the current estimates of the robot poses are correct and deriving the map from the observations of the landmarks made by the robot. When both the map and locations are unknown, localization and mapping have to be done simultaneously.

The dataset consists of two sets, the generic observations $Z = \mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_N$ and the control vectors $U^k = \mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_N$. These are used to compute the robot poses $X = \mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_N$ and the set of all landmarks, which is the map $M = \mathbf{m}_1, \mathbf{m}_2, \ldots, \mathbf{m}_M$.

The sensors are modeled as the likelihood $P(\mathbf{z}_k|\mathbf{x}_k, M)$, meaning the probability of making the observation when the true state is $\{\mathbf{x}_k, M\}$.

The platform motion is modeled in terms of the conditional probability $P(\mathbf{x}_k|\mathbf{x}_{k-1}, \mathbf{u}_k)$, i.e. the probability that $\mathbf{x}_k$ is reached when $\mathbf{u}_k$ is the control vector starting at location $\mathbf{x}_{k-1}$.

The joint posterior probability of both the map and the locations can be estimated recursively as $P(\mathbf{x}_k, M|Z^k, U^k, \mathbf{x}_0)$, or the probability that the map and location given the complete sets of measurements and control vectors up to time k are correct, when $\mathbf{x}_0$ is the origin. The observation update step can be derived using Bayes theorem.

When the platform motion model is assumed Markovian, the probability of the location estimates that has to be maximized can be calculated recursively as

$$\begin{aligned} P(\mathbf{x}_k, M|Z_k, U_k, \mathbf{x}_0) = \quad & KP(\mathbf{z}_k|\mathbf{x}_k, M) \int P(\mathbf{x}_k|\mathbf{x}_{k-1}, \mathbf{u}_k) \\ & P(\mathbf{x}_{k-1}, M|Z_{k-1}, U_{k-1}, \mathbf{x}_0) d\mathbf{x}_{k-1} \end{aligned} \tag{2.1}$$

where K is the ratio between the error in a single control vector and a single observation, and is (approximately) constant.

Vision-based Mobile Robot Localization And Mapping using Scale-Invariant Features

In this method [14] scale invariant image features, derived from the images of a trinocular stereo camera system, are used as landmarks. The benefit of SIFT (Scale Invariant Feature Transform) features is that they are invariant to image translation, scaling, rotation and partially invariant to illumination changes and affine or 3D projection. These characteristics make them useful for robust SLAM.

The SIFT features are selected at maxima and minima of difference in a

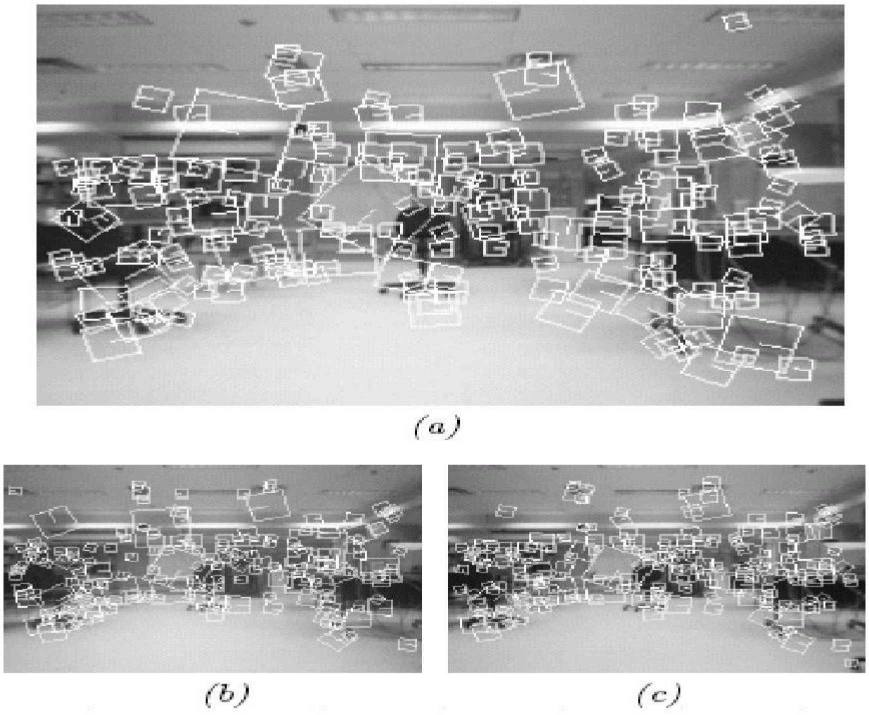


Figure 2.1: SIFT features found, with scale and orientation indicated by the size and orientation of the squares. (a) Top image. (b) Left image. (c) Right image.

gaussian function applied in scale space [17]. Since three cameras are used, the real world coordinates relative to the robot can be computed. The SIFT features can then serve as landmarks for map building and tracking.
For each stereo matched SIFT feature the coordinates $[r, c, s, o, d, x, y, z]$ are stored, where (r, c) are the image coordinates in the reference camera, (s, o, d) are scale, orientation and disparity and (x, y, z) are the coordinates in the real world relative to the reference camera.
To build the map the features are tracked over successive frames, where the odometry is used to compute a rough estimate of the next occurrence of the feature. Once SIFT features are matched, least squares is used to estimate the path, and the real world coordinates of the SIFT features are adjusted.
Each feature is stored in a database for future reference when they don't match any previous features as (x, y, z, s, o, l), where l is a count to indicate how many consecutive frames this landmark has been missed. This may happen because of occlusion. When a landmark has been missed for L consecutive frames while it should remain in the field of vision, it is removed

from the database, because it probably belongs to a dynamic object. There are four types of features to consider:

Type 1 This landmark is not expected to be within view in the next frame. It is not matched and its miss count l remains unchanged.

Type 2 This landmark is expected to be within view, but no matches can be found in the next frame. Its miss count is incremented by 1.

Type 3 This landmark is within view and a match is found. Its miss count is reset to 0.

Type 4 This is a new landmark which does not match any existing landmark in the database. It is added to the database.

Incremental Mapping of Large Cyclic Environments
Gutmann and Konolige [12] compare the local map built from the last K poses to all regions in the current global map to detect if a region has been visited before. They assume that the map M created up to pose $\mathbf{x}_{N-1}$ is correct when a new pose $\mathbf{x}_N$ is added. This map consists of a set of local scans, but the overlapping poses are deleted by this technique, so it represents one single global map.
The technique from Lu and Milios [3] described below is used to create a local map from the laser range scans of the last K poses. This local map is compared to the global map using the prior probability for each pose and the sensor response function, over all poses in the map. The sensor response function is approximated by a correlation operator between the two maps, and generates a probability distribution over the map.
For the detection of false positives, since they are extremely hazardous for the estimation process(once the map is adjusted it is extremely hard to recover), three filters are used:

High match score The unnormalized match score should be high.

Low ambiguity The peaks of clusters with high probability are compared. The ratio of the highest peak to the next highest peak should be large.

Low variance The best cluster should be sharply peaked.

When there is a match, the technique from Lu and Milios is used again on the entire set of poses to get the globally consistent pose estimation for the entire path, which is also the pose estimation for the local maps.

2.2 SLAM using Link Estimation from Local Maps

Globally Consistent Range Scan Alignment for Environment Mapping
Lu and Milios [3] use a set of links $\bar{\mathbf{d}}_{ij}$ that is obtained from odometry as well as from the matching of range scans. Each link $\bar{\mathbf{d}}_{ij}$ is a measurement of link $\mathbf{d}_{ij}$, which represents pose $\mathbf{x}_j$ in the local reference frame of $\mathbf{x}_i$.
If the links would have been in the global reference frame, the relation between all links and all poses could have been put in matrix form as $D = HX$, where $D = [\mathbf{d}_{01}\mathbf{d}_{02} \ldots \mathbf{d}_{N-1,N}]^T$ is a concatenation of all $\mathbf{d}_{ij}$ in a single vector, $X = [\mathbf{x}_0\mathbf{x}_1 \ldots \mathbf{x}_N]^T$ is a vector concatenating all $\mathbf{x}_i$, and H is the mapping from poses to links, consisting of identity matrices such that $\mathbf{d}_{ij} = \mathbf{x}_j - \mathbf{x}_i$.
The optimal estimation for the positions would have been achieved by minimizing the Mahalanobis distance W

$$W = (\bar{D} - HX)^t C^{-1} (\bar{D} - HX) \tag{2.2}$$

where C is the covariance matrix of D. This is done by solving the vector X using least squares as

$$X = (H^t C^{-1} H)^{-1} H^t C^{-1} \bar{D} \tag{2.3}$$

However, all links are in a local reference frame. The global poses are constructed from the local estimates by a 'pose compounding relation' $\mathbf{x}_j = \mathbf{x}_i \oplus \mathbf{d}_{ij}$, where the coordinates are related by

$$\begin{array}{rcl} x_j &=& x_i + x_{ij}\cos\theta_i - y_{ij}\sin\theta_i \\ y_j &=& y_i + x_{ij}\sin\theta_i + y_{ij}\cos\theta_i \\ \theta_j &=& \theta_i + \theta_{ij} \end{array} \tag{2.4}$$

Since the pose compounding relation is not linear, Lu and Milios make a linear approximation of the model to get the links in the global reference frame first.
In the case of matching range scans, when there is sufficient overlap between two range scans, the link is calculated globally for each matched feature, using the poses and the feature observation. The error in the link estimates is then minimized by least squares, resulting in a global link estimate. The odometry links are estimated as the difference between two successive poses. Now the complete set of links is known, so the Mahalanobis distance is minimized to get the best estimate for X. The map is then constructed as a set of range scans with their global poses X, which are the poses along the robot path where the scans have been made.

Sonar-Based Mapping With Mobile Robots Using EM

In this technique [6] the complete dataset is the set of range scans with the corresponding odometry data. A map is an assignment of features to each location on the robot path, so a local map is composed for each pose. Each local map is a likelihood field, that has a likelihood of occupancy for each pixel in the local map. The method uses three probabilistic models as its basis in the estimation process.

The motion model $P(\mathbf{x}'_i|\mathbf{u}_i, \mathbf{x}_{i-1})$ describes the probability that the robot's pose is $\mathbf{x}'_i$, given that $\mathbf{u}_i$ is performed at location $\mathbf{x}_{i-1}$. The perception model $P(\mathbf{z}_i|M, \mathbf{x}_i)$ models the likelihood of observing $\mathbf{z}_i$, where the map M and the pose $\mathbf{x}_i$ are known. The inverse perception model $P(M|Z, X)$ represents the likelihood of each local map in the world given the sonar scans and the pose.

Expectation Maximization (EM) is a hill-climbing procedure which alternates between an expectation step that computes the probability that the path is correct, and a maximization step that computes the best possible estimation of the path given the data.

Here the E-step computes $P(\mathbf{x}_i|M, D) = \alpha_i\beta_i$, where α_i computes the probability of the pose recursively, starting from pose $\mathbf{x}_0$, and β_i computes the probability of the pose backwards from pose $\mathbf{x}_N$. Usually the initial values are given by $\alpha_1 = P(\mathbf{x}_1|\mathbf{z}_1, M)$ and β_N is uniformly distributed.

The M-step calculates the most likely poses of the local maps. It generates a distribution μ over the poses, such that $\mathbf{x}^*_i$ is chosen as the maximum value for $P(\mathbf{x}_i|M, D)$.

When the $P(\mathbf{x}_i|M, D)$ reaches a maximum, the most likely path is found. A post-processing step calculates the global map by integrating the local maps $\mathbf{m}_i$ over their final poses $\mathbf{x}^*_i$.

Towards global consistent pose estimation from images

This method [1] has the same foundation as Lu and Milios [3], i.e. $D = HX$ is the relation between the links and the poses, and the Mahalanobis distance has to be minimized. The difference here is that no linear approximation is made. The dataset is split into a displacement vector $\bar{\mathbf{d}}^l_{d,ij}$ in the local reference frame, and an orientation difference between two poses $\bar{\theta}_{ij}$. Since omnidirectional camera images are used to compute the links, the estimated displacement vector is normalized. A reference trajectory, consisting of $\mathbf{d}^r_{d,ij}$ and θ^r_{ij}, is used to obtain the scale and global orientation of $\bar{\mathbf{d}}^l_{d,ij}$. Estimating the links from omnidirectional camera images is described extensively in chapter 4.

For the reference trajectory the odometry is reshaped using the link between begin- and end pose [11]. The scale r_e of this link is unknown, so it is initialized with an arbitrary value. First the orientations θ^r_{ij} are computed. The spatial displacement r^*_e is computed as

$$r^*_e = \operatorname*{argmin}_{r_e} \sum_{ij} \|\Delta \mathbf{d}_{r_e,ij}\|^2 \tag{2.5}$$

The lengths of the links in the reference trajectory $\mathbf{d}^r_{d,ij}$ are now used to scale the vectors $\mathbf{d}^l_{d,ij}$. The reference trajectory also contains the orientation θ^r_i of each pose, expressed in the global reference frame. Now the link in the global reference frame can be computed by derotating each vector with the estimated orientation difference θ^r_{ij}. These links are then used to minimize the Mahalanobis distance through least squares.

2.3 Conclusion

In this chapter a variety of methods were discussed. Our data consists of relative pose estimates, derived from sets of omnidirectional camera images. We chose an approach based on the globally consistent range scan alignment [3], because Lu and Milios work with a similar dataset, i.e. a set of relative pose measurements. However, our dataset represents the relative orientations along with the normalized translations, whereas Lu and Milios also have the length of the translation vector.

In the next chapter we work with simulated data. We first assume a linear measurement equation $D = HX$, to test the performance of Lu and Milios' approach. After this the orientation is varied, resulting in a nonlinear measurement equation. We try to split the dataset to solve each subpart linearly. Finally the same nonlinear measument equation is assumed, But this time an iterative approach is used to maximize the likelihood of the poses given the set of relative pose estimates. This is the basis for our solution as described in chapter 5.

Chapter 3

Estimation of Positions based on Relative Measurements

In this chapter we present three methods for location estimation. The first method provides an algorithm in the case that the poses have a fixed orientation. Now the links, or the relative pose estimates, are the same both in the global- and local reference frame.
The second method provides an extension for the case when the links are usually only known in the local reference frames. This happens when orientation is also variable. In this extension the set of links is divided in two subsets, one containing the orientations and one containing the translations.
The third method implements an iterative solution to improve the accuracy of the second method.
The datasets contain the relative link estimates $\bar{\mathbf{d}}_{ij}$, defined as an estimate of the pose $\mathbf{x}_j$ observed from pose $\mathbf{x}_i$. These links can be estimated from the odometry and/or the scans made by a mounted sensor, such as a laser range scanner. However, we use a simulated dataset here.

3.1 Estimating Locations from translation data

This first approach was designed to test the operability of Lu and Milios' method [3], when the measurement equation has the linear form $\mathbf{d}_{ij} = \mathbf{x}_j - \mathbf{x}_i$. Here $\mathbf{x}_i = [x_i, y_i]$ is pose i expressed in the global reference frame, using only position coordinates x_i and y_i. The orientation is fixed. $\mathbf{d}_{ij} = [x_{ij}, y_{ij}]$ consists of the translation needed to get from pose $\mathbf{x}_i$ to pose $\mathbf{x}_j$, so it contains only an x- and y-direction as well.
All links are concatenated into a single vector according to the notation of

Lu and Milios in section 2.2. The same goes for the poses. The measurement equation can then be put in matrix form as $D = HX$, where H is the incidence matrix, also defined by Lu and Milios. Its properties are defined in appendix A.
However, all that is available is the estimation $\bar{D}$ of the links, where $\bar{D} = HX + \Delta D$. The criterion for optimal estimation is the weighted sum of $\Delta\bar{\mathbf{d}}_{ij}$. This can be expressed as the mahalanobis distance, which is the weighted summed squared distance between the true relative positions $\mathbf{d}_{ij}$ and their measurements $\bar{\mathbf{d}}_{ij}$

$$W = (\Delta D)^t C^{-1}(\Delta D) = (\bar{D} - HX)^t C^{-1}(\bar{D} - HX) \tag{3.1}$$

Here C is the covariance of $\bar{D}$, which represent the weights. The inverse covariance matrix gives high weights to links that have low variability to express their importance.Since error in a single link does not affect any other link estimated from camera images (i.e. they are uncorrelated), it is a diagonal matrix with submatrices C_{ij}, where

$$C_{ij} = \begin{pmatrix} (\Delta x_{ij})^2 & 0 \\ 0 & (\Delta y_{ij})^2 \end{pmatrix}$$

The minimum Mahalanobis distance is determined by the least squares solution for the poses as

$$X = (H^t C^{-1} H)^{-1} H^t C^{-1} \bar{D} \tag{3.2}$$

Experiments

We want to study what effect increasing the covariance of each link has on the accuracy of the optimal estimate. For this the path is initialized as a random, universally distributed network of 40 positions $\mathbf{x}_i$, as can be seen in picture 3.1. The actual links are calculated according to $D = HX$. A dataset $\bar{D}$ of link estimates is then generated, using the actual set of links D. Each link $\bar{\mathbf{d}}_{ij}$ is normally distributed with mean $\mathbf{d}_{ij}$, and standard deviation σ_{ij} increasing from 1 to 30. It is estimated as $\bar{\mathbf{d}}_{ij} = \mathbf{d}_{ij} + \mathbf{d}_{ij} N(\mathbf{0}, C_{ij})$.
The Mahalanobis distance is used to calculate the remaining error in the pose estimates. Before the method is used, the Mahalanobis distance between the link estimates in the dataset $\bar{D}$ and the 'true' set $D = HX$ gives the initial error. After the method was used, the optimal estimate X' is used to create $D' = HX'$. The Mahalanobis distance between D and D' is then calculated to get the remaining error in the network. Figure 3.2(a) shows a graph where the Mahalanobis distance is plotted against σ_{ij}. Figure 3.2(b) gives the error

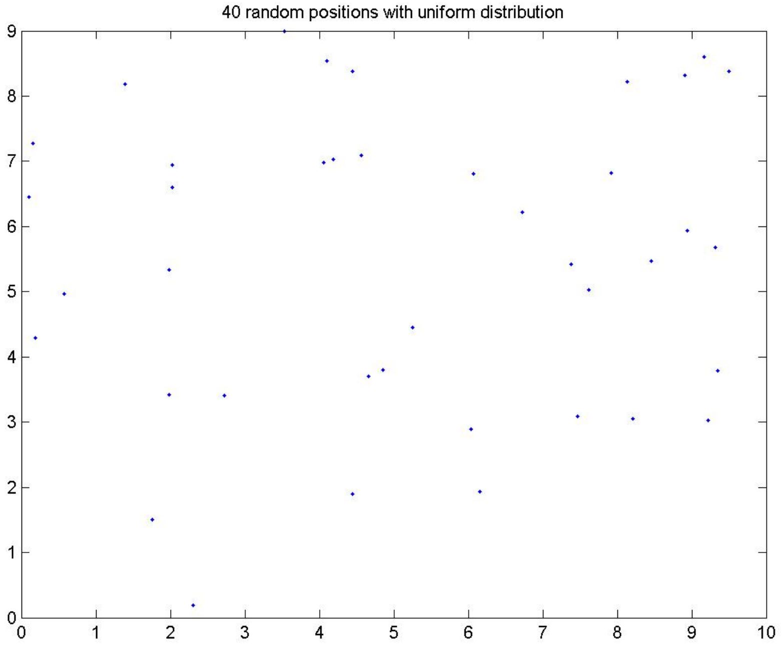


Figure 3.1: initialization of a universally distributed network of 40 positions. Displayed are the nodes of the network, which are the positions. All nodes are connected in this experiment.

in the pose estimates as $\sum_i (\mathbf{x}'_i - \mathbf{x}_i)^t (\mathbf{x}'_i - \mathbf{x}_i)$.

As can be seen in figure 3.2(a), least squares is an effective method when the links are in the global reference frame. As can be expected the pose difference increases when the amount of noise is too great. However, the Mahalanobis distance remains the same. This is because the covariance matrix is known in advance. When it is determined using the estimated links, the estimation quickly diverges to a random matrix. These runs demonstrate that least squares is sufficient when the covariance is known in advance and the links are known in the global reference frame.

3.2 Extension to a Nonlinear Measurement Equation

In this section we want to see how least squares works when the measurement equation is no longer linear, meaning that the relation between links $\bar{\mathbf{d}}_{ij}$ in the local reference frame and in the global reference frame is no longer fixed. To ensure this, the poses from the previous section will now include orientation, which is the direction in which the robot is headed in the current pose. In other words, $\mathbf{x}_i = [x_i, y_i, \theta_i]$. The local link $\mathbf{d}_{ij} = [x_{ij}, y_{ij}, \theta_{ij}]$ between two poses $\mathbf{x}_i$ and $\mathbf{x}_j$ can now be defined as the pose compounding relation from Lu and Milios [3]

$$\mathbf{x}_j = \mathbf{x}_i \oplus \mathbf{d}_{ij} \tag{3.3}$$

These vectors are related as

$$\left.\begin{array}{lcl} x_j & = & x_i + x_{ij}\cos\theta_i - y_{ij}\sin\theta_i \\ y_j & = & y_i + x_{ij}\sin\theta_i + y_{ij}\cos\theta_i \\ \theta_j & = & \theta_i + \theta_{ij} \end{array}\right\} \mathbf{x}_j = \mathbf{x}_i + R(-\theta_i)\mathbf{d}_{ij} \tag{3.4}$$

Similar to the previous section, all poses can be concatenated into a single vector. The same goes for the link estimates $\bar{\mathbf{d}}_{ij}$. We define a rotation matrix $R(-\Theta)$ as a diagonal matrix with submatrices $R_{ij} = R(\theta_i)$, which are in the same order as $\bar{\mathbf{d}}_{ij}$ in D. Now the pose compounding relation can be put in matrix form as $\bar{D} = R(-\Theta)HX + \Delta D$, which now represents the nonlinear measurement equation from appendix A.
The rotation matrix $R(-\Theta)$ contains free variables θ_i, which stand for the orientation of the local reference frame $\mathbf{x}_i$ (the direction of the robot). To remove these free variables, Lu and Milios [3] make a linear approximation of the global links by assuming that the pose errors $\Delta\mathbf{x}_k$ are small. Therefore it can also be observed that the orientations approximately form a linear subpart [1].
Our approach is to separate the orientations from the dataset, and make an estimate on the 1-dimensional dataset as described in the previous section, 3.1. This results in a set containing the optimal estimates of all θ_i. By substituting all θ_i in $R(-\Theta)$ by these estimates, the rotation matrix is now fixed. This means the measurement equation is linear, so least squares can be applied as in section 3.1.

Experiments

Once again the goal of this experiment is to determine what effect increasing the error in the dataset has on the performance of the system. The error

is calculated using the Mahalanobis distance. In the plot the difference is shown between the Euclidian distance using raw data and the Euclidian distance after using least squares. The remaining error is expected to be higher, because the orientations used to determine $R(-\Theta)$ are not the true orientations, so the rotation matrix is not completely accurate.
The dataset is generated by first generating a random network of 40 positions $\mathbf{x}_i$. The true links are calculated according to $\mathbf{d}_{ij} = R(\theta_i)(\mathbf{x}_j - \mathbf{x}_i)$. Now random covariances are generated for each link with standard deviation σ_{ij} a fixed number, $\bar{\mathbf{d}}_{ij} = \mathbf{d}_{ij} + \mathbf{d}_{ij}N(\mathbf{0}, C_{ij})$.
To get the results the method is tried on different datasets with $\sigma_{ij} = 1 \dots 25$. Shown in figure 3.3(a) is a graph where σ_{ij} is plotted against the Mahalanobis distance. The solid line is the initial error, the dashed line the error after least squares was applied.

It shows that the estimation quickly diverges. This is because we assumed that the dataset could be split into a linear- and a nonlinear part, but this is not true. The minimum mahalanobis distance of a subpart does not guarantee a minimum distance on the complete estimate. In the next section we will show an iterative approach, which analyses the dataset as a whole.

3.3 An Iterative Approach

Next we will present an iterative method in an attempt to improve the results. It requires two components as input:

- *The dataset.* This is the set of link estimates $\bar{D}$, identical to the set in the previous method.
- *The Model.* In this case the model is the path that has to be estimated, which is represented by a set of poses X^n. We will initialize it with the set of poses X^0 as they are estimated by odometry, because it is already a reasonable estimate. Since this is a hill-climbing approach, there is always the issue of whether the maximum likelihood is a local or a global likelihood. In this way, the probability that a local maximum will be reached is minimized.

Each iteration $n = \{1 \dots N\}$ we determine a set of estimates $\mathbf{x}_{ik}^n$ for each pose $\mathbf{x}_i^n$ that has to be determined in this iteration. For this the pose compounding relation from equation 3.3 is used.

$$\mathbf{x}_{ik}^n = \mathbf{x}_j^{n-1} \oplus \bar{\mathbf{d}}_{ij} \tag{3.5}$$

The new pose $\mathbf{x}_i^n$ is the mean value of all pose estimates $\mathbf{x}_{ik}^n$

$$\mathbf{x}_i^n = \frac{\sum_k \mathbf{x}_{ik}^n}{\#\ \text{estimates}} \tag{3.6}$$

Each step $n = 1 \dots N$ the estimated model X^n is derived in such a way that the likelihood $P(X^n|D, X^{n-1})$ is maximized. As soon as this likelihood no longer increases, the model is considered an optimal fit to the dataset.

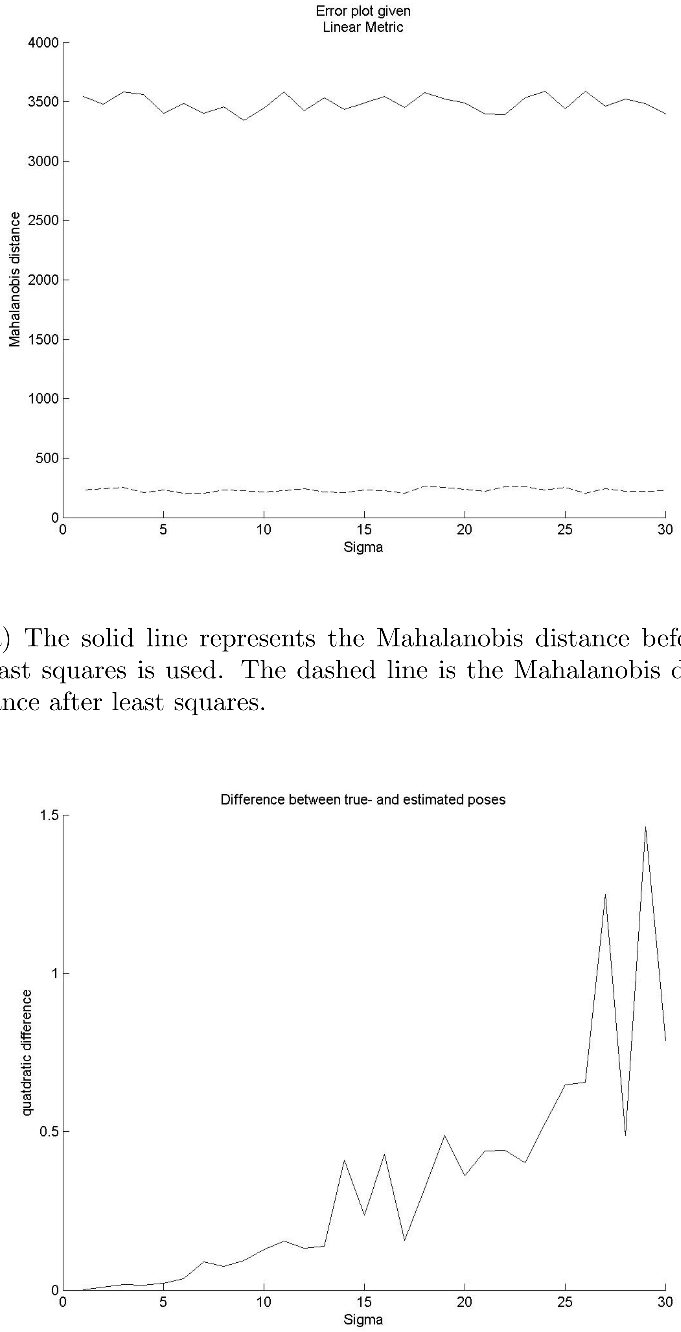
Experiments

We will analyse the influence of error on the performance of our iterative method. The Mahalanobis distance is used to express the total amount of error in the pose estimates, given the set of observed links. Two plots are shown in figure 3.4, One that shows the Mahalanobis distance and one that shows the difference between the estimated- and real poses $\sum_i (\mathbf{x}'_i - \mathbf{x}_i)^t (\mathbf{x}'_i - \mathbf{x}_i)$. It is expected to give better performance then our previous approach, since an iterative technique has no linearity constraint on the measurement equation, as opposed to least squares.
We generate the links from odometry readings and the links from sensor readings separately, but both in the same way as the previous method, i.e. the true links are calculated according to $\mathbf{d}_{ij} = R_{ij}(\mathbf{x}_j - \mathbf{x}_i)$. Then random covariances are generated for each link with standard deviation σ_{ij} a fixed number, after which both sets are calculated as $\bar{\mathbf{d}}_{ij} = \mathbf{d}_{ij} + C_{ij}\mathbf{d}_{ij}$. Odometry is used to obtain a set of links D between successive poses $\bar{\mathbf{d}}_{i,i+1}$, determined from the wheel rotations.
We increase σ to get results for increasing error once again, $\sigma = 1 \dots 30$. The results are more promising then with a least squares approach. According to figure 3.4(a) about 75% of the total error is removed, even with large error. However, figure 3.4(b) shows that some poses still contain significant error. Obviously there is much uncertainty in some of the poses in this set. Unfortunately, we did not find the exact cause for this due to a time restraint.

3.4 Conclusion

First least squares was used on a dataset containing only location information. Because of the linear properties of these links, this produced very good results.
However, when the orientation of the local reference frames is varied, it causes

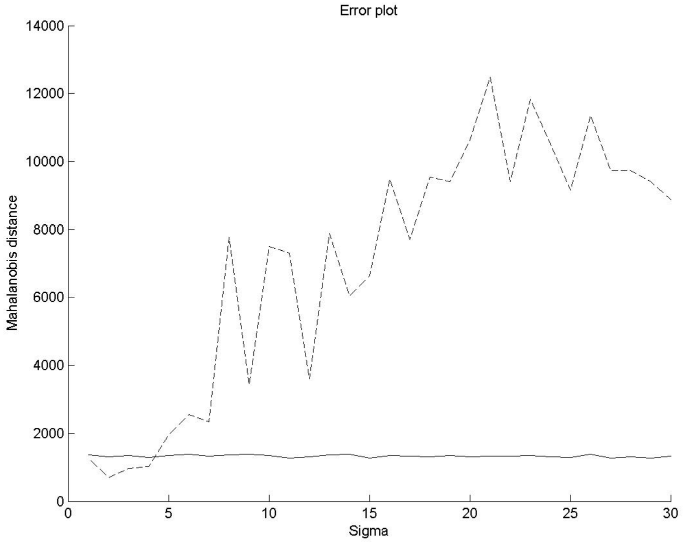
the measurement equation to become nonlinear. Even though these orientations could be estimated effectively, the result still contained substantial error. Therefore the rotation matrix used to estimate the positions was not completely accurate, and least squares was insufficient for the estimation of the complete poses when the standard deviation grew. We tried an iterative process which had to solve the problem of nonlinearity. The results were not as good as we expected, especially when we looked at the difference between the estimated- and the real poses.



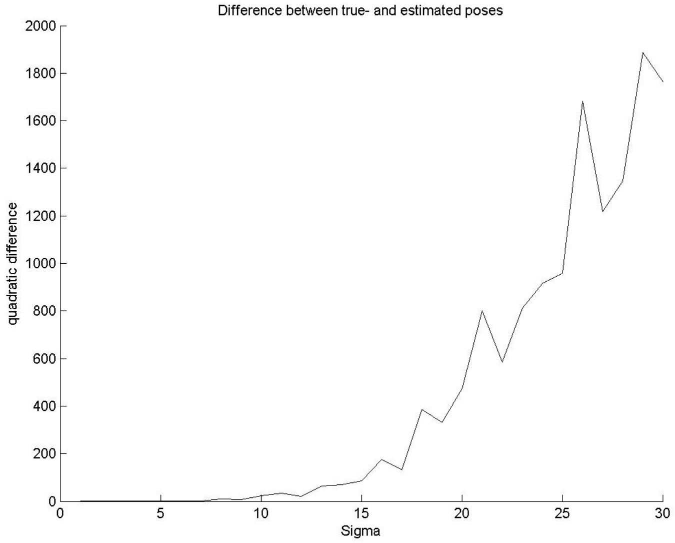
(a) The solid line represents the Mahalanobis distance before least squares is used. The dashed line is the Mahalanobis distance after least squares.

(b) The difference between the estimated coordinates and the true ones, as $\sum_i (\mathbf{x}'_i - \mathbf{x}_i)^t (\mathbf{x}'_i - \mathbf{x}_i)$.

Figure 3.2: Results from least squares, when the measurement equation is linear.

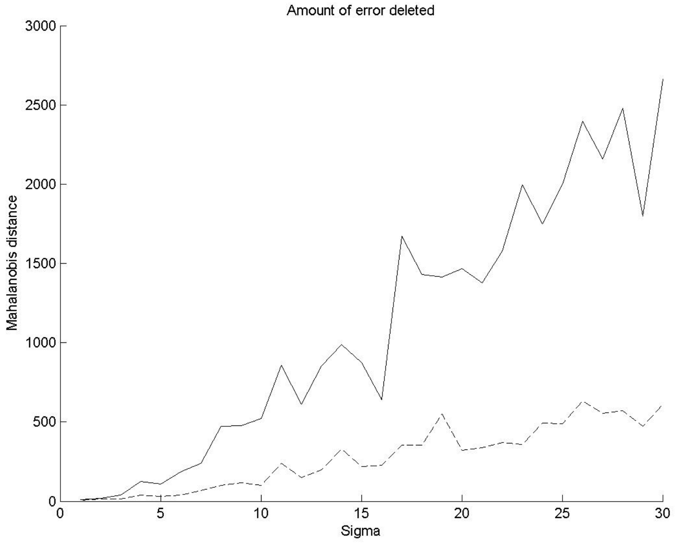


(a) The solid line represents the Mahalanobis distance before linear regression is used. The dashed line is the same Mahalanobis distance after linear regression. This result is from 30 nodes and the orientation estimation that came from the least squares on the entire dataset is used.

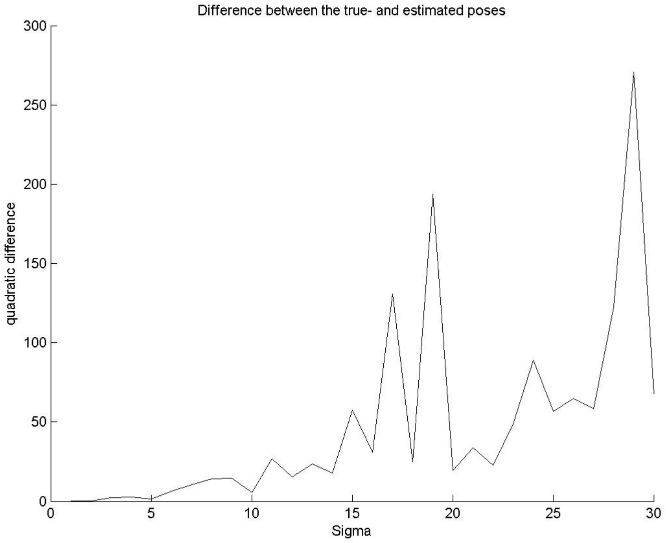


(b) The difference between the estimated coordinates and the true ones, as $\sum_i (\mathbf{x}'_i - \mathbf{x}_i)^t(\mathbf{x}'_i - \mathbf{x}_i)$.

Figure 3.3: Results from least squares when the measurement equation is nonlinear.



(a) We plotted the Mahalanobis distance against the standard deviation σ and the threshold. The solid line is the initial error, the dashed line shows the error after our method was used.



(b) This is the squared distance between the estimated- and the real poses after using our iterative method.

Figure 3.4: Results from our iterative technique.

Chapter 4

The Omnidirectional Camera as Sensor

In the previous chapter we described three algorithms that use the links between poses directly to estimate the path of the robot. The method described in the next chapter uses links estimated from omnidirectional camera images. The resulting set of links $\bar{\mathbf{d}}_{ij}$ consists of the orientation difference between two poses and a normalized vector in the direction of pose $\mathbf{x}_j$, observed from pose $\mathbf{x}_i$. The scale is not known, since a camera can not see depth, and therefore only the direction of the features that are matched are known. This dataset was used earlier [1], and was created based on a method from R. Bunschoten [7]. In all cases the same robot and camera were used. This chapter describes how the links were derived from the set of camera images [7] [1].

4.1 Omnidirectional Camera Images: A Description

The images were created using a camera that observes the world via a parabolic mirror, in such a way that all rays that pass through the focal point of the mirror are reflected into the focal point of the camera, and it is possible to observe α degrees from a horizontal line, as shown in figure 4.1. A point in $3-D$ space is given in homogeneous coordinates as $X = [x_4x_1, x_4x_2, x_4x_3, x_4]^T$. A transformation between coordinate systems is $M = TR$, where T and R are the 3x3 translation and rotation matrices between homogeneous coordinates respectively. Any point in the real world can be mapped to pixel coordinates $\mathbf{u}_C$ by mapping to the mirror frame, projecting to the plane $z = 1$ from the mirror perspective, then transform to

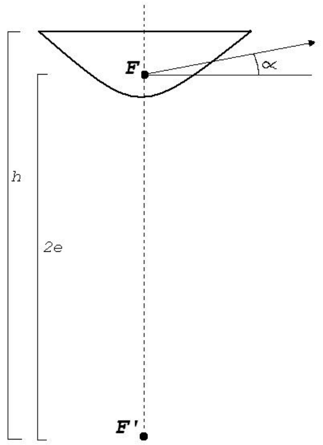


Figure 4.1: The hyperboloid and its parameters. In order to obtain a single effective viewpoint, the distance between the focal point F inside the mirror and the focal point F' of the camera should equal a constant value $2e$ (where e is the eccentricity that follows from the mirror shape parameters). The finite height h limits the maximum vertical viewing angle α that can be obtained.

the camera frame and finally to project the points to the plane $z = 1$ from the camera perspective. This just leaves a calibration to pixel coordinates to get the values, and the raw real world view is formed as in figure 4.2.

This image is then resampled to get a cylindrical view of the room by trans-

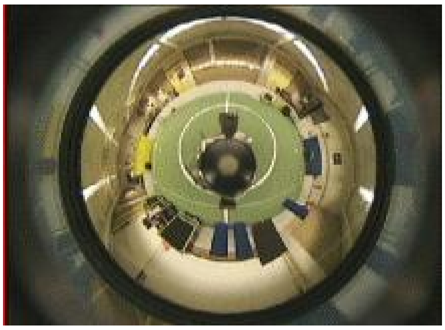


Figure 4.2: A direct perception of the environment by the omnidirectional camera. It gives a complete 360 ° view of the room.

forming the cylindrical pixels through the mirror frame back to the camera, after which another calibration takes place. To get the in-between pixel values, Bilinear interpolation is used. The result is shown in figure 4.3.

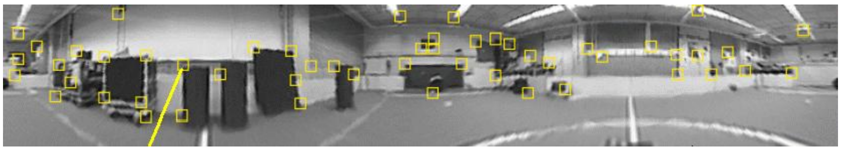


Figure 4.3: The derived 360 ° cylindrical view of the room. squares indicate features, which are derived in section 4.2

4.2 Transformation of Pairwise Omnidirectional Camera Images to Links

From a pair of cylindrical images, taken at any two poses $\mathbf{x}_i$ and $\mathbf{x}_j$, the link $\bar{\mathbf{d}}_{ij}$ has to be derived. A Kanade-Lucas-Tomasi feature tracker [2] is first used to compute the important features in both images. The selected features are the ones that are easy to track, such as corners(see figure 4.3). Assumed is that, when a feature is considered important at a certain location, it is also considered important in the vicinity of that location (i.e. corners are still the corners in the vicinity of an image[1]). These are then compared by taking the summed squared difference over pixel values in a window around the features in both images. Pairs of features that have a summed squared difference over all pixels in the window below a threshold are kept as corresponding features. When there is a sufficient match between two images, a link exists. The relation between two features in two different feature vectors $\mathbf{z}_i$ is ideally $\mathbf{z}_{kj}E\mathbf{z}_{ki} = 0$, where $E = RT$ is the essential matrix. This 3x3 matrix can be rewritten in vector notation such that $\mathbf{e}$ is a 1x9 vector that represents the essential matrix and D_k is the design vector created from a match for the k-th feature (which should be the same feature in both images).
The optimal $\mathbf{e}$ in $D\mathbf{e} = 0$ subject to $\|\mathbf{e}\| = 1$ can be found by the 8-point algorithm. Here the eigenvector of $M = D^T D$ associated with the smallest eigenvalue that can be found by a Singular Value Decomposition (SVD) of M, gives the minimum for $\mathbf{e}$.

[1] This is only partly true, because corners that are detected my also be two crossing objects with different depth.

Unfortunately the properties of the essential matrix, i.e. that it has rank 2 and has two equal eigenvalues, are not enforced by the 8-point algorithm. Now let $\bar{E} = U\Sigma V^T$ be the estimated matrix and its SVD, where $\Sigma = \text{diag}(\sigma_1, \sigma_2, \sigma_3)$. The true essential matrix is calculated as $E = U\Sigma' V^T$, where $\Sigma' = \text{diag}((\sigma_1 + \sigma_2)/2, (\sigma_1 + \sigma_2)/2, 0)$.

4.3 Issues in Combining Camera Images

First of all, there are 4 possible combinations of R and T that are compatible with an essential matrix. To get the correct combination, all 4 combinations are used to compute the depth of each feature from both images. This has to be positive. In practice, however, the correspondences are noisy and the essential matrix is not completely accurate. Because of this, the combination of R and T that yields the most positive results in $\mathbf{z}_{kj} E \mathbf{z}_{ki}$ is selected.

False correspondences between features are also a source of errors. E is estimated using an m-estimator, which is a weighted least squares solution, which evaluates the equation $\mathbf{z}_{kj} E \mathbf{z}_{ki} = 0$. Using this E the features that cause the most residue are deleted. Then E is estimated using the m-estimator once more. This results in a better fit.

As an extra precaution the rotation matrix R is compared to the rotation around the z-axis R_z, and the translation T to the translation in the horizontal plane $T_x + T_y$, because the robot always moves in a 2D plane. When there is a high residue, the robot must have moved up or down (or rotated up or down), which is not possible, so it indicates false correspondences or wrong selections.

4.4 Conclusion

A method has been defined to estimate the link between two poses from the omnidirectional camera images made in both poses. First the coordinates are transformed to a plane with coordinates on a cylinder around the omnidirectional camera, and then the most important features are extracted using existing software. Rotation and normalized translation are computed from the essential matrix, and some outliers are removed. The problems with this dataset are, however, that there are still some major outliers present and the error is no longer normally distributed. Also, the links that were estimated are normalized, so only the relative orientation between the poses are known. The disadvantage of having only normalized vectors is that, without knowing the scale of these normalized links, there is insufficient information to get globally consistent pose estimates.

Chapter 5

Location Estimation from Pairwise Image Links

Section 3.3 describes an iterative approach to location estimation given a set of links, which contain both relative orientation and translation. Since our Nomad Scout robot uses an omnidirectional camera, the these links can be determined, byt the translation is normalized. Therefore, each link represents a line on which the pose estimate lies.
In this chapter we will describe how we derived the scaling factors for the translation of the links. These scaling factors are needed to determine the location of the pose estimation on the line given by the link. Then we will show how to adapt the iterative approach to include the estimation of scaling factors. Also we will add an algorithm to effectively delete the outliers discussed in section 4.3.
We will describe the experiments performed both on an automatically generated test set and finally on a real world dataset, made by a Nomad Scout robot.

5.1 The Dataset

The odometry is represented by a set of poses X, known in the global reference frame, of which pose $\mathbf{x}_0$ is the origin. Each pose $\mathbf{x}_i$ consists of a location and an orientation of the local reference frame at that pose, i.e. $\mathbf{x}_i = [x_i, y_i, \theta_i]$.
In the previous chapter the omnidirectional camera images were transformed to a set of links $\bar{D}$. Each link $\bar{\mathbf{d}}_{ij}$ contains a normalized translation and a rotation between pose $\mathbf{x}_i$ and pose $\mathbf{x}_j$, i.e. $\bar{\mathbf{d}}_{ij} = [\bar{x}_{ij}, \bar{x}_{ij}\bar{\theta}_{ij}]$. This link is in the local reference frame of pose $\mathbf{x}_i$. This is demonstrated in figure 5.1.

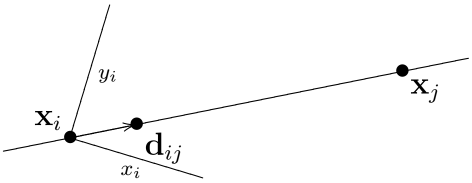


Figure 5.1: Observation of pose $\mathbf{x}_j$ from pose $\mathbf{x}_i$

Since the links are normalized, scaling factors γ_{ij} have to be determined to scale the links as

$$\Gamma_{ij}\bar{\mathbf{d}}_{ij} = \begin{pmatrix} \gamma_{ij} & 0 & 0 \\ 0 & \gamma_{ij} & 0 \\ 0 & 0 & 1 \end{pmatrix} \bar{\mathbf{d}}_{ij} \tag{5.1}$$

These scaling factors are unknown, so they have to be estimated using the available datasets X and/or $\bar{D}$.

5.2 Estimating the Scaling Factors

We wish to retrieve the scaling factor γ_{ij} for all links $\bar{\mathbf{d}}_{ij}$, so we can use our iterative approach from section 3.3 once again.
To retrieve the scaling factors for the links, we look at two possible methods. We can use only one of these methods, so we will choose one for our experiments.

Intersection This is based on vanishing points [15]. In this method, as well as with closest point estimations, we assume that all data is known in the global reference frame. Therefore we drop the orientations for now. Let us look at two lines. Their respective origins are $\{\mathbf{x}_i, \mathbf{x}_j\}$, which contain x- and y-location. Also given is a set of two links $\{\mathbf{d}_{ik}, \mathbf{d}_{jk}\}$, which give the direction of the lines. The intersection point is called $\mathbf{x}_k$. This is demonstrated in figure 5.2.

Now let γ_{ik} be the scaling factor for $\mathbf{d}_{ik}$ and γ_{jk} for $\mathbf{d}_{jk}$. We need to solve the scaling factor γ_{ik} or γ_{jk} to obtain the estimate for the location of the intersection point $\mathbf{x}_k$.
Observe that there exists a triangle with vertices $\{\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k\}$ and edges $\{\Gamma_{ik}\mathbf{d}_{ik}, \Gamma_{jk}\mathbf{d}_{jk}, \mathbf{x}_j - \mathbf{x}_i\}$. The surface of this triangle is half the surface

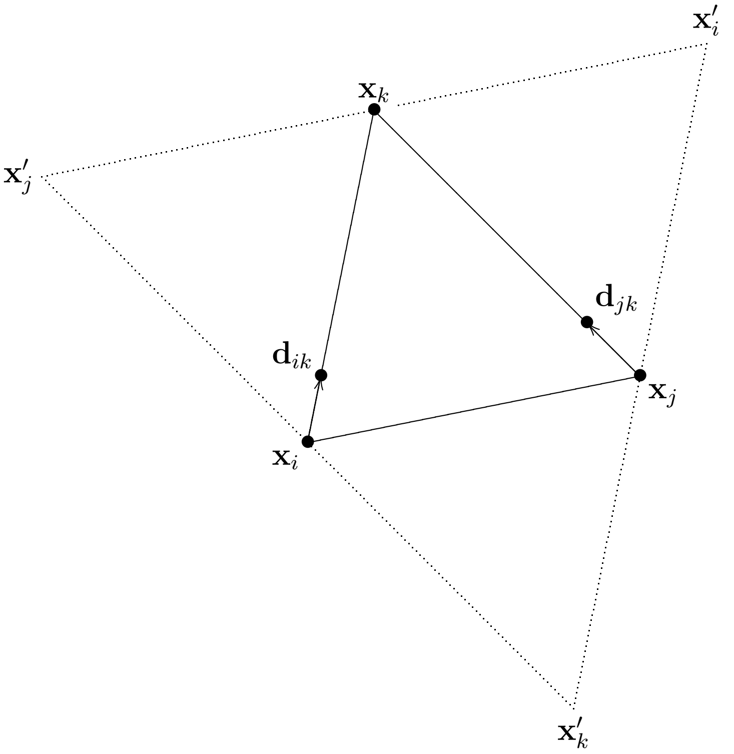


Figure 5.2: Visualization of the intersection of two lines, and the three parallellograms that can be made when both lines are known.

of the parallellogram spanned by two of its edges (figure 5.2). The surface of this parallellogram is computed by the determinant of the matrix containing these two edges.

$$\begin{aligned}
\text{surface} &= \frac{|\det\begin{pmatrix} \mathbf{x}_j - \mathbf{x}_i & \gamma_{jk}\mathbf{d}_{jk} \end{pmatrix}|}{2} \\
&= \frac{|\det\begin{pmatrix} \mathbf{x}_j - \mathbf{x}_i & \gamma_{ik}\mathbf{d}_{ik} \end{pmatrix}|}{2} \qquad (5.2) \\
&= \frac{|\det\begin{pmatrix} \gamma_{ik}\mathbf{d}_{ik} & \gamma_{jk}\mathbf{d}_{jk} \end{pmatrix}|}{2}
\end{aligned}$$

To get the scaling factors, we observe that they can be placed outside the determinant, and the division by 2 can be removed.

$$\begin{aligned}\text{surface} = \quad & \gamma_{jk}|\det\begin{pmatrix}\mathbf{x}_j - \mathbf{x}_i & \mathbf{d}_{jk}\end{pmatrix}| \\ = \quad & \gamma_{ik}|\det\begin{pmatrix}\mathbf{x}_j - \mathbf{x}_i & \mathbf{d}_{ik}\end{pmatrix}| \\ = \quad & \gamma_{ik}\gamma_{jk}|\det\begin{pmatrix}\mathbf{d}_{ik} & \mathbf{d}_{jk}\end{pmatrix}|\end{aligned} \tag{5.3}$$

The scaling factors can be placed outside of the determinant, because a parallellogram that is spanned by two vectors $\{a\mathbf{d}_{ik}, b\mathbf{d}_{jk}\}$ has surface $\frac{1}{ab}$ times the surface of the parallellogram spanned by vectors $\{\mathbf{d}_{ik}, \mathbf{d}_{jk}\}$. solving the scaling factors in equation 5.2 results in

$$\begin{aligned}\gamma_{ik} &= \frac{|\det\begin{pmatrix}\mathbf{x}_j - \mathbf{x}_i & \mathbf{d}_{jk}\end{pmatrix}|}{|\det\begin{pmatrix}\mathbf{d}_{ik} & \mathbf{d}_{jk}\end{pmatrix}|} \\ \gamma_{jk} &= \frac{|\det\begin{pmatrix}\mathbf{x}_j - \mathbf{x}_i & \mathbf{d}_{ik}\end{pmatrix}|}{|\det\begin{pmatrix}\mathbf{d}_{ik} & \mathbf{d}_{jk}\end{pmatrix}|}\end{aligned} \tag{5.4}$$

The intersection point is now calculated as

$$\mathbf{x}_k = \mathbf{x}_i + \gamma_{ik}\mathbf{d}_{ik} = \mathbf{x}_j + \gamma_{jk}\mathbf{d}_{jk} \tag{5.5}$$

Closest Point Estimate Here we look at a single line, with its origin at $\mathbf{x}_i$ and its direction given by $\mathbf{d}_{ik}$,. The closest point $\mathbf{x}_k^*$ to the point $\mathbf{x}_k$ on this line can be calculated by projecting $\mathbf{x}_k$ on the line $\mathbf{x}_i + \gamma_{ik}\mathbf{d}_{ik}$. This is done by

$$\mathbf{x}_k^* = \mathbf{x}_i + \gamma_{ik}^*\mathbf{d}_{ik} \qquad \gamma_{ik}^* = \frac{(\mathbf{x}_k - \mathbf{x}_i)\cdot\mathbf{d}_{ik}}{\mathbf{d}_{ik}\cdot\mathbf{d}_{ik}} \tag{5.6}$$

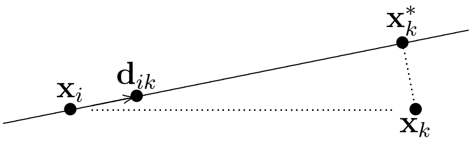


Figure 5.3: Visualization of the closest point estimate, by projecting the pose $\mathbf{x}_k$ on the line.

5.3 Removal of wrong correspondences

Another issue with estimating links from camera images is that there are wrong correspondences. Even though a lot of these were removed in the previous chapter, there are still some of these outliers left in the set of links. Assumed here is that the links $\bar{D}$ are known in the global reference frame, and the scaling factor matrix Γ has also been determined. For each pose $\mathbf{x}_i$ we can determine a subset $\bar{D}_i$, which is the subset of D that contains all the links are connected to pose $\mathbf{x}_i$. Observe here that $\mathbf{d}_{ij} = \mathbf{0} \ominus \mathbf{d}_{ji}$, where $\ominus$ is the inverse pose compounding defined by Lu and Milios [3]. Let Γ_i be the matrix containing submatrices Γ_{ij} on its diagonal, in the order given by $\bar{D}_i$. A set of pose estimates for $\mathbf{x}_i$ can now be created as $\Gamma_i \bar{D}_i$.

Assuming normal distribution of the pose estimates, we can compute the probability that each estimate is correct. This is done by first computing the covariance matrix C_i given the entire set of pose estimates $\Gamma_i \bar{D}_i$ and then determining the vector $P(\Gamma_i \bar{D}_i | N(\mathbf{x}_i, C_i))$. We remove a pose estimate (a link $\bar{\mathbf{d}}_{ij}$ and its corresponding Γ_{ij}) when this probability is below a certain threshold. The optimal threshold is determined empirically.

5.4 EM on Normalized Links

Expectation Maximization (EM) is an iterative technique to classify data into an arbitrary number of clusters. This is done in an iterative manner, where the loglikelihood of the data given the distribution of the clusters is maximized. In this context each pose in the robot path is a cluster mean. What has to be done in each iteration is to get the best estimate of each pose given the set of links and the previous estimates of the poses. The algorithm for EM is described in Statistical Pattern Recognition by A.R. Webb [4].

5.4.1 EM in Location Estimation incorporating Omnidirectional Camera Datasets

In EM, there are 4 important parts.

Initialization We need an initial estimate of the model, X^0. We will use the odometry set X for this. In this way we hope to stay away from any local maxima in the likelihood.
The dataset is our set of links $\bar{D}$. This set is fixed, the model needs to find an optimal fit to this dataset.
The covariance matrix is initialized in the first iteration, where each submatrix $C_{ij} = C_i$, as shown in section 5.3.

The likelihood function The Mahalanobis distance is a good measure of how much error is left in the estimate. Since the likelihood increases when the Mahalanobis distance decreases, it can be used effectively as a measurement of when a global (or local) maximum is reached. The Mahalanobis distance is defined as

$$W = (\Gamma \bar{D} - R(\Theta)HX)^T C^{-1} (\Gamma \bar{D} - R(\Theta)HX) \tag{5.7}$$

Here $R(\Theta)$ is the rotation matrix as defined in section 3.2. Γ is a matrix containing the scaling matrices Γ_{ij} on its diagonal, to scale $\bar{D}$. This is described in section 5.2.

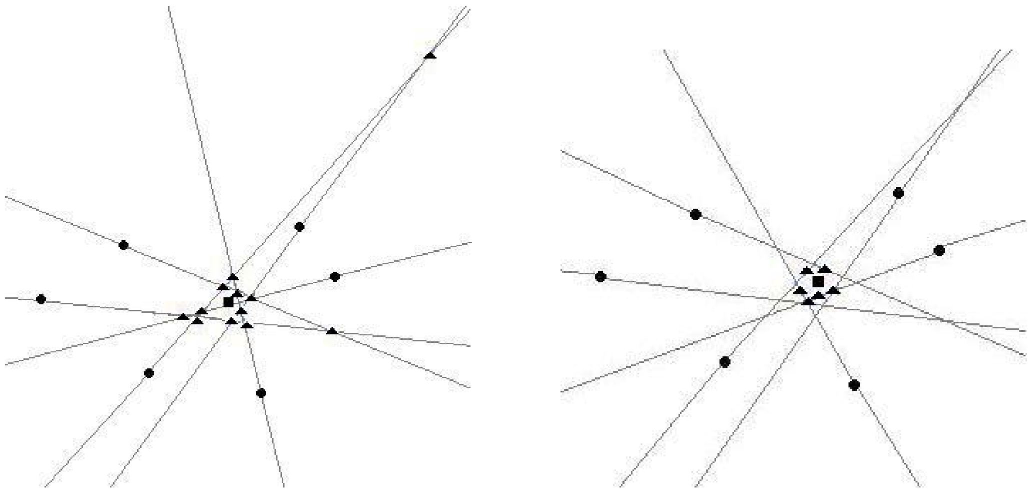
The E-step Given the current best guess of the poses, X^n, the links are transformed to the global reference frame. This is done by derotating each vector $\bar{\mathbf{d}}_{ij}$ by the current estimate of pose $\mathbf{x}_i$, which is $\mathbf{x}_i^{n-1}$. The result is a set $\bar{D}'$. When we determine the scaling factors we drop the rotations, so the links consist only of translations. This can be done, because all links are known in the global reference frame and the scaling factor for the rotations is 1.
Now the set of estimates has to be created, so the scaling factors γ_{ij} have to be determined. This has been described in section 5.2.
In the case of intersections, each possible pair of links $\{\bar{\mathbf{d}}'_{ik}, \bar{\mathbf{d}}'_{jk}\}$ and their origins, which are the poses $\{\mathbf{x}_i^{n-1}, \mathbf{x}_j^{n-1}\}$, are used to create a pose estimate $\mathbf{x}_{kl}^n$.
When the closest point estimates are made, each link $\bar{\mathbf{d}}'_{ik}$ is handled separately. It is passed along with the two poses $\mathbf{x}_i^{n-1}$ and $\mathbf{x}_k^{n-1}$ to create a new pose estimate $\mathbf{x}_{kl}^n$.
We now have a set of estimates for each pose. Assuming normal distribution for this set of estimates, the probability for each estimate can be determined so that outliers can be removed. This is described in section 5.3.

The M-step Each pose now has a set of 0 or more estimates. Given this set we determine the mean by

$$\mathbf{x}_i^{n+1} = \frac{\sum_k \mathbf{x}_{ik}^{n+1}}{\#\text{ estimates}} \tag{5.8}$$

When the number of estimates is 0, no new estimate can be made, so

$$\mathbf{x}_i^{n+1} = \mathbf{x}_i^n \tag{5.9}$$



(a) Shown in this figure is the process of getting the estimates of a pose given a set of links by intersection. All poses $\mathbf{x}_i$ are black dots. Pose $\mathbf{x}_j$ is shown as a square. Its estimates $\mathbf{x}_{jl}$ are the triangles. The links $\bar{\mathbf{d}}_{ij}$ are all lines.

(b) Shown in this figure is the process of getting the estimates of a pose given a set of links by a closest point estimate. All poses $\mathbf{x}_i^{n-1}$ are dots. Pose $\mathbf{x}_j^n$ is shown as a square. Its estimates $\mathbf{x}_{jl}^n$ are the triangles. The links $\bar{\mathbf{d}}_{ij}$ are all lines.

Figure 5.4: The results for computing pose estimates given a set of links that represent the direction in which the pose is observed.

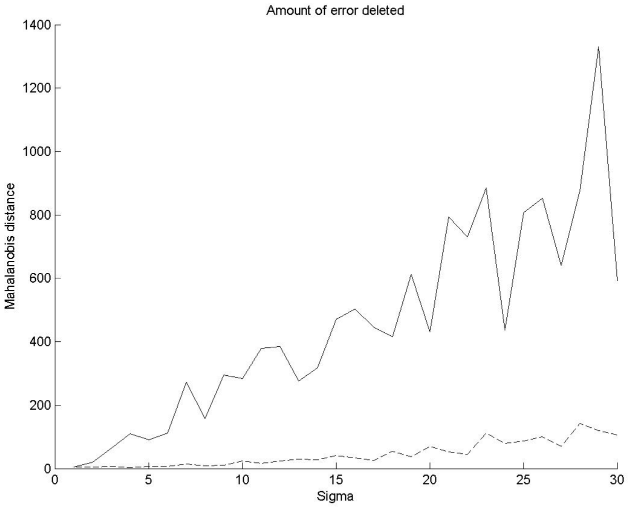
5.5 Experiments

First we did some experiments on an automatically generated test set. This test set consists of an initial representation of the path, X^0, and a set of normalized links observed from the local reference frames, $\bar{D}$. This set is generated similar to the set in section 3.3, except here the links $\bar{\mathbf{d}}_{ij}$ are normalized after the errors are added. Since this is an automatically generated test set, there are no outliers present. Therefore we set the threshold to be 0.

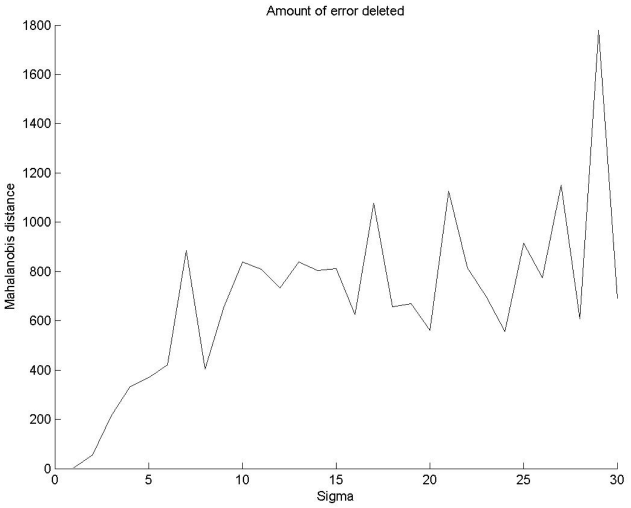
We chose to use closest point estimates for determining the scaling factors. This is because they don't have the disadvantage that the estimation diverges to an extreme value and therefore becomes useless, which does happen when intersections are used.

The ground truth for X is once again initialized randomly, with uniform distribution. It consists of 40 poses.

The odometry set X^0 is initialized by first calculating $\mathbf{d}_{i,i+1} = \mathbf{x}_j \ominus \mathbf{x}_i$ and adding noise to these links. We assume normal distribution, with mean $\mathbf{d}_{i,i+1}$ and standard deviation σ_{ij} a fixed number increasing from 1 to 30, such that$\bar{\mathbf{d}}_{i,i+1} = \mathbf{d}_{i,i+1} + \mathbf{d}_{i,i+1} N(\mathbf{0}, C_{i,i+1})$. Now $\mathbf{x}^n_{i+1} = \mathbf{x}^n_i \; oplustext\bar{b}fd_{i,i+1}$ and $\mathbf{x}_0 = \mathbf{0}$.
The set of links $\bar{D}$ is generated exactly the same way, but this time it contains $\bar{\mathbf{d}}_{ij}$ for all i and j.
Plotted in figure 5.5 are the results from our tests. Next we used a dataset created in the real world, using the Nomad scout robot. It consists of 93 poses in X^0 and 1520 links in $\bar{D}$. They are calculated using the method from chapter 4. Figure 5.6 shows the results for using EM on this dataset. Since the Mahalanobis distance is zero when all scaling factors are zero, the estimated set of poses converges to a single pose. Since we had no bias regarding the exact location of the poses, we had to delete the outliers to prevent this effect. It seems that the threshold plays an important part in the estimation. Even though our dataset contained a lot of noise/outliers, EM still resulted in a reasonable estimate of the underlying poses. The most important part now is to get a better estimate of the links from the images, so that our method can effectively decrease the error in the poses as they are obtained from odometry.

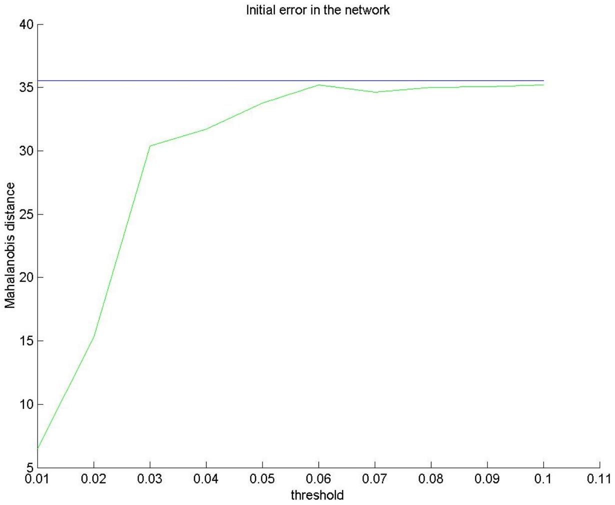


(a) Plotted here is the Mahalanobis distance before and after using our method. The solid line is before, and the dashed line after using EM.

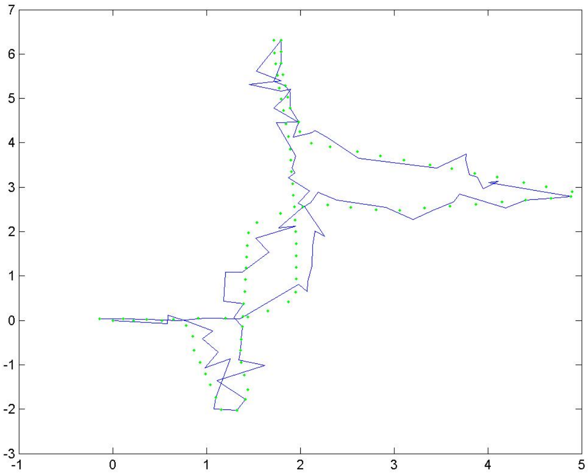


(b) Plotted here is the difference between the true- and estimated poses, as $\Sigma_i(\mathbf{x}'_i - \mathbf{i})^t(\mathbf{x}'_i - \mathbf{i})$.

Figure 5.5: Results from our iterative technique on automatically generated links.



(a) The error before and after using our method is shown for increasing thresholds. The blue line is the initial error, green shows the remaining error after using EM.



(b) This is the corresponding path at threshold 0.02. Dots show the nodes as estimated by the odometry, and the blue line is estimated by our method using the links from camera images.

Figure 5.6: These are the results from using EM when omnidirectional cameras are used.

Chapter 6

Conclusions

The goal for this research was to test whether expectation maximization is an effective method for SLAM, when omnidirectional camera images are used. These have the additional problem that no depth information can be derived, so only the orientation between poses is available. We gave an introduction in the existing methods first. From this we concluded that it would be most effective to give a method that would maximize loglikelihood by minimizing the Mahalanobis distance, much like Lu and Milios [3].
We tried to use a simple least squares method, but due to the nonlinearity of the measurement equation this seemed insufficient. Therefore an iterative process was used, which gave better results.
Since all these methods used links with both orientation and translation, we first described how our links were obtained and how they differ from the links obtained from other exteroceptive sensors, such as laser range scanners. Since we have only normalized links, we first defined how to get the scaling factors for these links, after which weadapted the iterative method from section 3.3 to match these links. Also, we deleted the outliers so that a better estimate could be made.
The results showed that, when a low threshold was used, over 80% of the error in the original set is removed. It should be noticed, though, that the minimum error occurs when all poses are on the same place. This means all lengths of the links are estimated 0, so the error also goes to 0. This is why we set the threshold to 0.02 for the optimal network.
This method is still an offline method. For future work one should look at how to do this online, using the raw camera images as input. Also the bias for estimating the scaling factors as 0 should be removed, so further research is also needed here.

Appendix A

Given a set of links $\mathbf{d}_{ij}$ that represent the pose $\mathbf{x}_j$ observed from the pose $\mathbf{x}_i$, the matrix H defines the relation between them. The complete set of links D is a vector in this case, which concatenates all links as $D = [\mathbf{d}_{12}\mathbf{d}_{13}\dots\mathbf{d}_{N-1,N}]^T$, and X is a vector that concatenates all poses. The relation is defined as $D = HX$.

The Linear Measurement Equation

When the relation between two poses is linear, the matrix H can be used to convert poses to links directly. The most generic case is when $\mathbf{d}_{ij} = A_i\mathbf{x}_i + B_j\mathbf{x}_j$. In this case, and when only two nodes are used, H looks like

$$\begin{pmatrix} A_i & B_j \end{pmatrix} \tag{A.1}$$

One of the simplest cases is when $\mathbf{d}_{ij} = \mathbf{x}_j - \mathbf{x}_i = -I\mathbf{x}_i + I\mathbf{x}_j$, where I is the identity matrix. This is the specific matrix used in section 3.1. When the set of links and the set of poses have an arbitrary size, it looks like

$$\begin{pmatrix} \mathbf{I} & \mathbf{0} & \mathbf{0} & \dots & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{I} & \mathbf{0} & \dots & \mathbf{0} & \mathbf{0} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ -\mathbf{I} & \mathbf{I} & \mathbf{0} & \dots & \mathbf{0} & \mathbf{0} \\ -\mathbf{I} & \mathbf{0} & \mathbf{I} & \dots & \mathbf{0} & \mathbf{0} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \dots & -\mathbf{I} & \mathbf{I} \end{pmatrix} \tag{A.2}$$

Any method that assumes a linear relation can be used for estimating the poses, such as for example least squares.

The Nonlinear Measurement Equation

When the relation is nonlinear, the incidence matrix can not be used to convert poses into links directly. This is because the links $D = HX$ are still in the global reference frame. However, e.g. in section 3.2, the poses also contain an orientation component. Since the links are in the local reference frames, a derotation is needed to get them in the global reference frame, from where they can be added to the poses linearly. This is formalized as $\mathbf{d}_{ij} = R(\theta_i)(\mathbf{x}_j - \mathbf{x}_i) = -R(\theta_i)\mathbf{x}_i + R(\theta_i)\mathbf{x}_j$. The incidence matrix still looks the same, but now a rotation matrix R is added to transform the links to their respective local reference frame, i.e. $D = R(\Theta)HX$. The rotation matrix R looks like

$$R = \begin{pmatrix} R(\theta_1) & \mathbf{0} & \mathbf{0} & \dots & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & R(\theta_1) & \mathbf{0} & \dots & \mathbf{0} & \mathbf{0} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \mathbf{0} & \dots & R(\theta_i) & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{0} & \dots & \mathbf{0} & R(\theta_j) & \dots & \mathbf{0} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \dots & \mathbf{0} & R(\theta_N) \end{pmatrix}$$

where

$$R(\theta_i) = \begin{pmatrix} \cos\theta_i & \sin\theta_i & 0 \\ -\sin\theta_i & \cos\theta_i & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

In other words, the rows and columns in the rotation matrix that map the global link onto the link $\bar{\mathbf{d}}_{ij}$, contain the rotation matrix $R(\theta_i)$.

Properties of the Incidence Matrix

One important issue that has to be dealt with before least squares can be used is that H has to be overfitted. This means that there have to be more difference vectors then positions. Lu and Milios [3] state that:

> If the network is fully connected and the individual error covariances are normally behaved, we believe it is possible to prove that G is invertible.

They define G as $G = H^t C^{-1} H$. Since C^{-1} is already invertible (or has rank equal to its dimension), H has to have rank equal to the dimension of the vector X. In a fully connected network this is obviously the case (with more then 3 nodes), but what is the minimum amount of links that can be used to get an estimate of the poses? We will derive this minimum by proving that (Figure A.1)

> If each point can be triangulated with at least two other points, G is invertible.

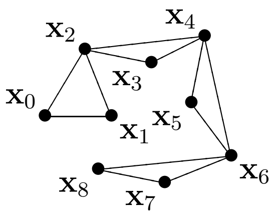


Figure A.1: The minimum amount of links needed to invert the incidence matrix.

It has to be stated here that, even though G is invertible, it is only possible to make a decent estimation if each point can be triangulated, directly or indirectly, with each other point (which is not the case in figure A.1). Otherwise the network can be separated into at least two smaller networks, say $N1$ and $N2$, and one single node S that forms the link between them. Since there is no triangulation between the three, i.e. there is no edge between a node in $N1$ and a node in $N2$, only the odometry passes S. This means that the error in S is the odometry error, so no minimization is made (the resultant error equals the prior error).

It is also necessary that one single point is excluded from the estimation process, since we need the absolute coordinates in the real world of the global reference frame. If this is not done, the coordinates of the global reference frame remain as a roaming variable in the estimation process. It is a matter of custom to exclude $\mathbf{x}_0$, and thereby let it be the origin of the global reference system. If it is not the origin, an extra operation is needed to transform each pose estimate in the network to the global reference frame.

Proof of minimal G

This proof uses induction. It is only done for the linear case, since the rotation matrix is always invertible.

If we look at H, it seems it has a clearly defined structure. To see if G is invertible, we first look at the rank of H. This is because linear algebra tells us that if a matrix has an inverse, the determinant is not 0. It is also true that $A = B \cdot C \Rightarrow |A| = |B| \cdot |C|$. So the determinant of each matrix is nonzero. In a broader context where B and C aren't square, the minimum demand is that the rank of a matrix is equal to the dimension in which it operates. Intuitively this means that a matrix that operates in a subspace of its own dimension mirrors data into its own subspace, and loses information. This results in a singular matrix.

So now the thing to prove is that H has rank equal to dimension[1]. First we delete as many rows as possible, until the network is reduced to its minimal form: a triangle strip. This does not increase the rank, so it does not influence our proof. It does prove that, if the minimal form is invertible, any form that has more links is also invertible.

Now if we notice that a submatrix H_{ij} of H, which is the link between a single $\mathbf{d}_{ij}$ and two poses $\mathbf{x}_i$ and $\mathbf{x}_j$, is nonsingular, we can reduce the problem to 1 dimensional points and say that this matrix has to have rank equal to the dimension of this matrix. Now each triangle has in each dimension the structure

$$\begin{pmatrix} -1 & 1 & 0 \\ -1 & 0 & 1 \\ 0 & -1 & 1 \end{pmatrix} \tag{A.3}$$

We assume that in computing the REF of the complete H the first column was swept by the previous triangle, so the result is

$$\begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & -1 & 1 \end{pmatrix} \tag{A.4}$$

This means that each triangle adds rank 2 per dimension of the data. This is also required. The only remaining issue is the first triangle. This one does not have a previous triangle, so the first column is not swept. Now the issue of having $\mathbf{x}_0$ as a reference point comes into play, for it can not be included in H. Observe that $\mathbf{d}_{ij} = \mathbf{x}_j - \mathbf{x}_i \Rightarrow \mathbf{x}_0 = 0 \Rightarrow \mathbf{d}_{0j} = \mathbf{x}_j$. This means that if we have a fixed reference point, G is invertible, and if the reference point is included in the estimation process, we can solve the system up to a common factor, it being the actual location of the global reference frame.

[1] assumed here is that the covariance matrix is not singular

Bibliography

[1] Stephan H.G. ten Hagen and Ben J.A. Kröse. ”Towards global consistent pose estimation from images”. In *Proceedings of the International Conference on Intelligent Robots and Systems (IROS), 2002*, paper #727.

[2] S.Birchfield. KLT: an implementation of the Kanade-Lucas-Tomasi feature tracker. Source code: *http://robotics.stanford.edu/b̃irch/klt/*, 1997.

[3] F. Lu and E. Milios. ”Globally Consistent Range Scan Alignment for Environment Mapping”. In *Autonomous Robots, 1997*, pages 333 - 349.

[4] A.R. Webb. ”Statistical Pattern Recognition second edition”. ISBN 0-470-84514-7, 2002.

[5] B.J.A. Kröse, N. Vlassis and R. Bunschoten. ”Omnidirectional Vision for Appearance-based Robot Localization”. In *the International Workshop on Sensor Based Intelligent Robots, 2000*, pages 39 - 50.

[6] Wolfram Burgard, Dieter Fox, Hauke Jans, Christian Matenar and Sebastian Thrun. ”Sonar-based Mapping With Mobile Robots Using EM”. In *Proceedings of the 16-th International Conference on Machine Learning (ICML) 1999.*

[7] Roland Bunschoten. ”Mapping and Localization from a Panoramic Vision Sensor”. Doctorate thesis, ISBN 90-9017279-3, 2003.

[8]

[9] Hugh Durrant-Whyte. ”Localisation, Mapping and the Simultaneous Localisation and Mapping (SLAM) Problem”. In *SLAM Summer School 2002.*

[10] Kristian T. Simsarian, Thomas J. Olson and N. Nandhakumar. ”View-Invariant Regions and Mobile Robot Self-Localization”. In *IEEE Transactions on Robotics and Automation, 1996*, pages 810 - 816.

[11] Stephan H.G. ten Hagen and Ben J.A. Kröse. "Trajectory reconstruction for self-localization and map building". In *Proceedings of the International Conference on Robotics and Automation (ICRA), 2002.*

[12] Jens-Steffen Gutmann nad Kurt Konolige. "Incremental Mapping of Large Cyclic Environments". In *2000 IEEE International Symposium on Computational Intelligence in Robotics and Automation (CIRA).*

[13] Jun Miura, Yoshiro Negishi and Yoshiaki Shirai. "Mobile Robot Map Generation by Integrating Omnidirectional Stereo and Laser Range Finder". In *Proceedings of the 2002 IEEE/RSJ International Conference on Intelligent Robots and Systems.*

[14] Stephen Se, David Lowe and Jim Little. "Vision-based Mobile Robot Localization And Mapping using Scale-Invariant Features". In *Proceedings of the International Conference on Robotics and Automation (ICRA), 2001*, pages 2051-2058.

[15] Michael Bosse, Richard Rikoski, John Leonard and Seth Teller. "Vanishing Points and 3D Lines from Omnidirectional Video". In *IEEE 2002 International Conference on Image Processing (ICIP), 2002*, paper #2771.

[16] Artur Arsénio and M. Isabel Ribeiro. "Absolute Localization of Mobile Robots using Natural Landmarks". In *IEEE International Conference on Electronics, Circuits and Systems (ICECS), 1998.*

[17] Tony Lindeberg. "Principles for Automatic Scale Selection". *Technical report ISRN KTH NA/P-96/18-SE. Department of Numerical Analysis and Computing Science, Royal Institute of Technology, S-100 44 Stockholm, Sweden, May 1996.*

Met dank aan...

Allereerst wil ik Ben Kröse bedanken voor zijn continue steun en begeleiding tijdens het project. Zonder die twee-wekelijkse evaluaties had dit slechts een schim geweest van wat het geworden is. Stephan ten Hagen wil ik bedanken voor het bijpraten aan het begin, wat zeker bijgedragen heeft aan een vlotte start. Ook wil ik hem bedanken voor de Nomad Scout dataset, die voor de praktijkresultaten heeft gezorgd.

Vervolgens zijn daar familie. Mijn familie wil ik bedanken voor het geduld en de steun die ze mij gaven, ondanks dat datgene wat ik deed niet altijd even goed begrepen werd. Vader, Moeder, Peter, Natasja, Marcel, Lidy, Niels, Maartje, Marcel, Ronald, Wendy, Opa en Oma en natuurlijk de rest.

Bedankt aan mijn vrienden, die zorgen voor regelmatige ontspanning buiten werktijden om. Arie, Renee, Dirk, Nancy, Rob, Dorathe, Rob, Remko, Dave, Kelly, Vincent, Sjoerd, Kobus, Bart, Bennie, Joyce, Mark, Pieter, Mark, Vera, Ruud en Rolinda. Omdat ontspanning net zo belangrijk is als inspanning, en zij daar altijd voor konden zorgen.

Mijn collega's wil ik graag bedanken: Karim Ayachi, Martijn Brekhof, Eelco Schatborn, Anouschka Aralgoe, Nadeem de Vree, Philip Prins, Arjan Scherpenisse, Frank Geerlings en Marcus Heukelom, voor de lunch en de overleggen over de verschillende projecten die we volgden. Op deze manier kom je er toch achter dat de problemen die voor je liggen niet uniek zijn, maar dat iedereen ze tegen komt.

Als laatste voor wil ik dan nog een aantal mensen bedanken voor de goede tijd tijdens (en voor/na) de studie. Jochem Liem, Jasper Uijlings, Daan Vreeswijk, Jurgen Sturm, Arjan Scherpenisse, Ork de Rooij, Gijs Kunze, Paul Ruinard, Elinor Bakker, Tonnie Wessling en Lotte Schutte.

Ben ik niemand vergeten?

P.S. Joep nog bedankt voor altijd een vrolijke thuiskomst :P.