Hough-transform based map stitching after localization uncertainty

Okke Formsma
University of Amsterdam

August 17, 2012

Abstract

Abstract

Contents

Chapter 1

Introduction

A compelling use of robotics is in the Urban Search And Rescue setting. The goal of Urban Search And Rescue robot teams are to help emergency services find victims of disasters, and perform duties which are deemed to dangerous for humans. In many disaster areas, human responders may be at risk due to fire, smoke, toxicity, radioactivity or the possiblility of a secondary disaster. A team of robots is much better suited to explore such dangerous disaster areas than humans.

Urban Search And Rescue robots face many challenges. They must navigate unknown environments with many potential hazards. The problem of mapping an unknown environment and localization within this environment is dubbed SLAM, for Simultaneous Localization And Mapping. Testing such algorithms with actual robots in real-life environments is expensive and cumbersome. Robots and sensors are expensive, as well as creating a disaster-environment.

A high fidelity simulation environment called USARSim, based on the Unreal Engine, provides an platform on which to test the performance of the robots in a disaster environment without the cost and effort associated with real robots. The RoboCup international robotics conference and competition utilizes USARSim in their Rescue Simulation League. This competition draws teams from universities all over the world to show their advances in robotic control software, focusing on issues such as human-robot interfaces, autonomous behavior and team coordination.

The univeristy of Amsterdam has joined forces with Oxford for the RoboCup, creating the AOJRF (Amsterdam-Oxford Joint Rescue Forces team [1]) team. The Simulatenous Localization And Mapping (SLAM) method employed by the AOJRF is not robust against sensor failures, which occur for example when the laser scanner is severly tilted. In this report, we propose a novel method which splits a map when the localization uncertainty exceeds a certain threshold. The different pieces of the map are then stitched together using a Hough transform based map stitching method.

This report is structured as follows. In the next chapter, an overview of the Simultaneous Localization and Mapping problem and related algorithms is provided. Chapter 3 presents the Hough-based stitching method. The experimental method is described in chapter 4, and experimental results in chapter 5. The results are discussed in chapter 6 and pointers for future work in chapter 6.3.

Chapter 2

ManifoldSLAM

One of the goals of the USAR challenge was to create a useable map of the environment.

The problem an agent faces when it needs to find it's way in an unknown environment is called SLAM, for Simulatenous Localization and Mapping. The agent has no map of the environment, and also no way of knowing it's exact location. It needs to infer the map and it's position on the map from the information it gathers from it's sensors throughout time.

Thus, SLAM is a chicken-and-egg problem: without a map it is hard to estimate the agent's position, and without the agent's position it's hard to estimate the map!

In this section we will first examine how the robot keeps track of the state of the world, and then how the SLAM problem can be solved with Extended Kalman Filters.

2.1 State

The state of the world as it is known to the agent is denoted x. The value of this set of variables may change over time, as the agent collects sensor data from the environment. The state at time t is denoted x_t.

Many variables may be included in the state variable x. For the purpose of this work, we will assume that it contains at least the agent's *pose and a* map of the environment.

In most SLAM approaches, two types of sensor data is collected: *environmental measurement data and* control data. Environmental measurement data collects information about the environment, while control data collects information about the robot within the environment. Environmental measurement data is denoted z (at a specific point in time z_t). Examples of environmental sensors available to an agent are laser scanners, sonars and video cameras. Control data and GPS signals. Control data sensors collect information intrinsic to the agent: it's velocity and position. Examples of control sensors are odometry sensors, inertia sensors and global positioning systems.

For the purpose of this paper we are mainly interested in the online SLAM problem, in contrast to the full SLAM problem. Online (or incremental) SLAM seeks to estimate the current pose x_t and map m:

$$p(x_t, m|z_{1:t}, u_{1:t}) \tag{2.1}$$

In contrast, the full SLAM problem estimates all poses x and map m:

$$p(x_{1:t}, m|z_{1:t}, u_{1:t}) \tag{2.2}$$

In practice, the difference is that the full SLAM problem reconsiders the location of all previous poses in estimating the map, while online SLAM treats these as given. The full SLAM problem is computationally much more expensive. Thus for time sensitive applications, such as real time robot control, usually only online SLAM is performed.

The robot's knowledge of it's current state is reflected in the *belief* function. The belief function gives a posterior probability over the state variable. The belief at time x_t is defined as follows:

$$bel(x_t) = p(x_t|z_{1:t}, u_{1:t}) \tag{2.3}$$

This belief is usually updated in two steps: once when the control data (u_t) is received, and again when new measurement data (z_t) is received. The belief of the robot after the control update is called the prediction, denoted $\overline{bel}(x_t)$:

$$\overline{bel}(x_t) = p(x_t|z_{1:t-1}, u_{1:t}) \tag{2.4}$$

A general algorithm for calculating beliefs is the *Bayes filter*. It first calculates the prediction $\overline{bel}(x_t)$ by integrating out the possible values of the previous belief x_{t-1} from previous belief $bel(x_{t-1})$ and the expectation of the current state given the previous state and control data $p(x_t|u_t, x_{t-1})$.

$$\overline{bel}(x_t) = \int p(x_t|u_t, x_{t-1}) bel(x_{t-1}) \mathrm{d}x_{t-1} \tag{2.5}$$

The final step is the measurement update, in which the measurement data z_t is incorporated:

$$bel(x_t) = \eta p(z_t|x_t) \overline{bel}(x_t) \tag{2.6}$$

In the previous formula, η is a normalization constant which ensures that $bel(x_t)$ is a proper probability.

2.2 Gaussian state representation

It is often convenient to represent the belief of the current state as a multivariate Gaussian. A Gaussian is defined by its mean μ and covariance Σ:

$$p(x) = \det(2\pi\Sigma)^{-\frac{1}{2}} \exp\{-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)\} \tag{2.7}$$

Gaussian state representations have the advantage of being easily understood and straightforwardly manipulated. This comes with a cost: a Gaussian is unimodal – there is only one maximum (the mean) possible. This makes Gaussian representations ill suited when there are many different hypothesis distributed across a map.

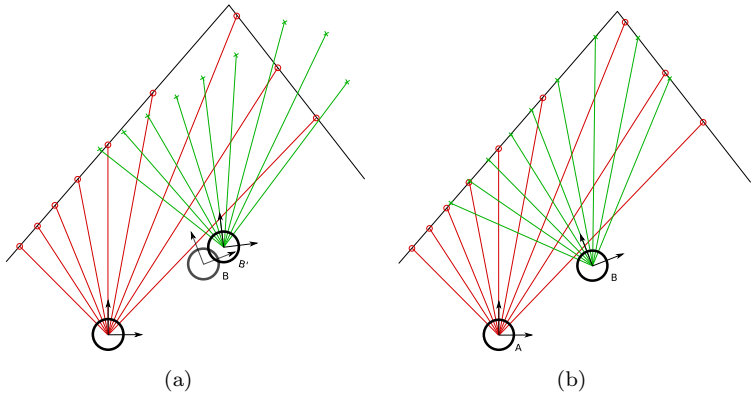
In UsarCommander, the state is represented by a 3-dimensional vector: (x, y, θ) and the corresponding 3-by-3 covariance matrix Σ. x and y are the respective coordinates of the robot in a global frame, while θ is the rotation angle of the robot with respect to the y axis.

When the state is represented by a Gaussian, the formulas in the previous section (2.1) can be relatively easily calculated.

2.3 Scan matching with the Weighted Scan Matcher

A robot has access to a variety of sensors, as discussed in section 2.1. Many sensors are not very accurate and reliable: odometry sensors yield erroneous values when the robot's wheels spin on a slippery slope, GPS sensors are unreliable indoors, and inertia sensors experience 'drift' – they need to be calibrated every once in a while. Laser range sensors provide an alternative. It measures with high precision the distance to the first object in a certain direction.

Laser range sensors are often put in a rotating device, which measures the distance to objects on a (often horizontal) plane. A field of view of 180°with a 1°resolution is common, which yields 181 distance scans. The 181 scans form a point cloud of 181 points. See image **??**.



(a) (b)

Figure 2.1: Matching two point clouds. In (a), the red and green point clouds are obviously not well aligned. In (b), they are much better aligned. Images courtesy of [**?**].

The 'scans' that the laser range sensor returns are processed through a scan-matching algorithm. These algorithms compare two scans (or a scan and a map) to see how they relate to each other. The scanmatcher needs to find out the relative change in orientation and position between the two scans.

In the context of this paper, we will focus on the Weighted Scan Matcher, but various other approaches exist. Slamet and Pfingsthorn [2] provide an excellent overview of various scanmatching algorithms such as Normal Distribution Transform (NDT), Iterative Dual Correspondence (IDC), Metric-based Iterative Closest Point (MbICP) and ICP.

The Weighted Scan Matcher (WSM) was introduced by Pfister et al. [3]. In contrast to other scan matchers, the influence of a correspondence between two measurements is weighted with the certainty of how well the two points match.

When two point clouds align perfectly, the uncertainty of these matches is zero. This will not generally be the case. Points that have less associated uncertainty will weigh more in the final scan matching result. The Weighted Scan Matcher explicitly models three error sources: the correspondence error, measurement error, and bias. See figure 2.2.

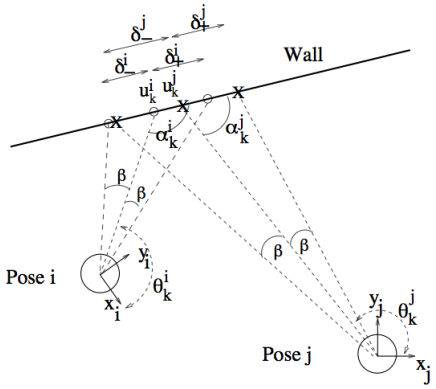


Figure 2.2: Geometry of the scan matching process. Courtesy of [3].

The correspondence error results from the limited resolution available to laser scanners. The resolution in this case is not the resolution in distance measurement, but the angular resolution of the scanner. At a distance of 5 meter and an angular resolution of 1°, the two measurements lie almost 9 centimeters apart ($500 * \tan(1°) \approx 8.7\ldots$). When the robot scans again after rotating 0.5°, all measurements will lie in between two previous measurements, and the best match will be off a few centimeters, depending on the distance from the scanner. Even worse: a thin object, such as a iron pole, can be easily missed by the scanning beams as well. All of these errors are examples of the correspondence error.

The measurement error models the error in the measurement process itself. This can because of any external factors which make the measurement erroneous.

Finally, the bias incorporates the difference between measurement methods. For example, laser scanners and sonar have different performance characteristics. The bias term compensates for this.

In general, the error between two scans a_i and b_i can be expressed as a rotation matrix R and translation t:

$$\epsilon = a_i - R \cdot b_i - t \tag{2.8}$$

With the assumption that all noise is Gaussian, measurement a_i can be expressed as the addition of the measurement error $\delta r_{a,i}$ and bias error $d_{a,i}$ to the actual distance r_i (similar for b_i):

$$a_i = r_{a,i} + \delta r_{a,i} + d_{a,i} \tag{2.9}$$

By substituting equation 2.9 into equation 2.8, the full error term is obtained:

$$\begin{aligned}\epsilon &= (r_{a,i} + \delta r_{a,i} + d_{a,i}) - R \cdot (r_{b,i} + \delta r_{b,i} + d_{b,i}) - t \\ &= \underbrace{(r_{a,i} - R \cdot r_{b,i} - t)}_{\text{correspondence error}} + \underbrace{(\delta r_{a,i} - R \cdot \delta r_{b,i})}_{\text{measurement error}} + \underbrace{(d_{a,i} - R \cdot d_{b,i})}_{\text{bias}} \end{aligned} \quad (2.10)$$

In an application where only one range measurement system is employed (such as a specific type of laser range scanner), the bias error term can be dropped. (Remember, the bias only models differences between measurement methods). In addition, the measurement error with modern laser scanners is so small that the measurements can be taken as a good approximation of the real measurement ([3]: "For practical computation, we can use [measurements] θ_k^i and l_k^i as a good estimates for the quantities [actual values] Θ_k^i and L_k^i."). Thus, when the sensor error is small enough, we can safely ignore this source of error as well.

This simplifies equation 2.10 further, so it contains only the correspondence error:

$$\epsilon = r_{a,i} - R \cdot r_{b,i} - t \quad (2.11)$$

This error term ϵ is minimized through an iterative algorithm in which the rotation and translation are estimated in turn. When the algorithm is suitably converged, it terminates. The full covariance matrix for the scan match is computed from the individual correspondence errors. We will see how this covariance matrix comes in handy for the SLAM algorithm and error estimation procedures in later chapters.

2.4 ManifoldSLAM

ManifoldSLAM was developed by Bayu Slamet and Max Pfingsthorn in 2006 [**?**] to compete in the RoboCup Urban Search and Rescue Simulation and won first place. It is based on the Manifold data structure by Howard et al. [4].

Many SLAM algorithms use a bayesian filter to refine the state by incorporate the current knowledge into a new assessment of the current state of the robot. The SLAM algorithm in use by the AOJRF team takes the map m_{t-1} and previous position x_{t-1} as given.

2.5 Uncertainty measures

There is always uncertainty associated with the results of the scan matcher. The weighted scan matcher models three sources of error for each laser scan line. The full uncertainty of the new pose is estimated by summing the information gained by each match that is scanned. More information about how this information filter works can be found in [5, p.71] and [6, p.72].

The correspondence error is a good measure to estimate the confidence of the scanmatcher because the weighted scan matcher explicitly discounts irrelevant error sources. When the correspondence error is high, the uncertainty of the scanmatcher about the current robot pose is also high.

The correspondence error can not easily be compared visually because it is represented by the covariance of a Gaussian distribution, which in this case is represented by a 3-by-3 matrix. Prior work by Visser et al. [7] used the trace of the covariance (the sum of the on-diagonal elements) to acquire a single, easily compared metric. Slamet and Pfingsthorn [6, p. 61] also mention the determinent as a good single confidence value that can be derived from this matrix. The number of matching scan lines is taken as a third measure. A single scan line in a scan is considered to match when the distance to existing datapoint on the map is no more than a certain maximum distance.

Chapter 3

Map Stitching with the Hough Transform

ManifoldSLAM only matches a recent part of the map to the current laser scan to estimate the current position of the robot. The robot is not aware of areas it has encountered before because of this. In the meantime, an erroneous location estimate may proliferate. The map stitching algorithm in this chapter can be used to find a global match between two submaps which have an unknown relative pose.

Stefano Carpin presented a novel map stitching approach based on the Hough Transform in [**?**]. It is an extention of the Hough transform which Censi et al. introduced as the *Hough spectrum* [8]. Censi et al. used the hough spectrum for a novel scan matcher.

Remember from the SLAM section (**??**) that aligning two maps takes two parameters: the rotation angle θ and translation vector $t = [x, y]^T$. The Hough spectrum is used for finding the rotation θ between a scan and reference map. The implementations of Carpin and Censi et al. diverge in their method of finding the translation vector t between two scans.

The Hough-based map stitching works with the assumption of an indoor environment. It is a global method, and does not take into consideration any prior knowledge about the relative alignment of both pieces of the map.

This chapter is setup as follows. First, we'll look at what the Hough transform and Hough spectrum are. Then, we'll take a look at how the best matching rotation $\hat{\theta}$ is found, and finally we'll examine translation t further. This chapter concludes with an elaborate example.

3.1 Hough transform

The Hough transform is in origin a line detection technique, patented by Paul Hough in 1962[9]. An improved version, which is presented here, was introduced by Duda and Hart[10]. The Hough transform is also extended to detect circles, ellipsis and parabolas, and also arbitrary 2D shapes[11].

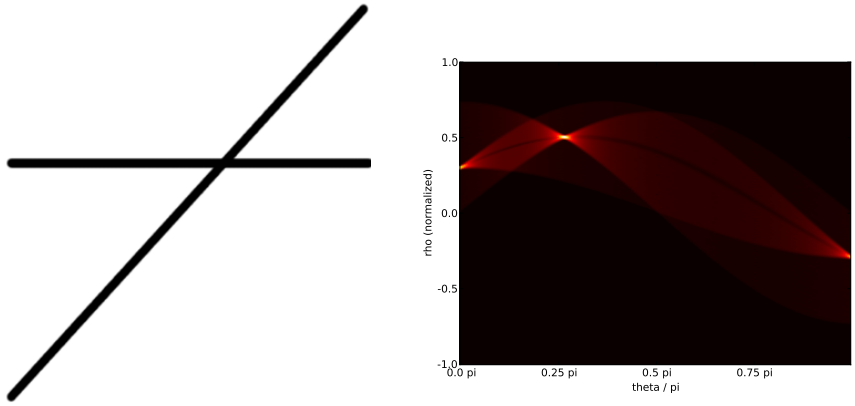
The Hough transform detects lines in the following manner. A line in Cartesian (x, y) space can be defined by it's angle (θ) and distance from the origin ρ:

$$x \cos\theta + y \sin\theta = \rho \tag{3.1}$$

Every point in (x, y) space lies on infinitely many lines satisfying equation 3.1. Every angle θ has a single accompanying distance from the origin ρ. Thus, a point in (x, y) space can be represented in the *Hough domain* (θ, ρ) as a curve. The locations where many curves intersect in the Hough domain correspond to the lines in the original image; remember that lines correspond to points in the Hough domain.

Finding the intersection of many curves in the continuous domain is computationally expensive. The discrete Hough transform mitigates this problem by discretizing the Hough domain into bins. Every point in the original image 'votes' for each bin it belongs to in (θ, ρ) representation. The strongest lines stand out as bins with a high number of votes.

There are n_θ angle-bins, and n_ρ distance bins in discrete Hough transform $\mathcal{HT}$.



(a) Two lines

(b) The Hough transform of (a)

Figure 3.1: Example of the discrete Hough transform. Notice the two maxima in (b) which correspond to the two lines in (a). There seem to be three maxima (one at 0, one at $\frac{\pi}{4}$ and one at π). However, the maxima at 0 and π are the same maximum; see section 3.4.

3.2 Hough spectrum

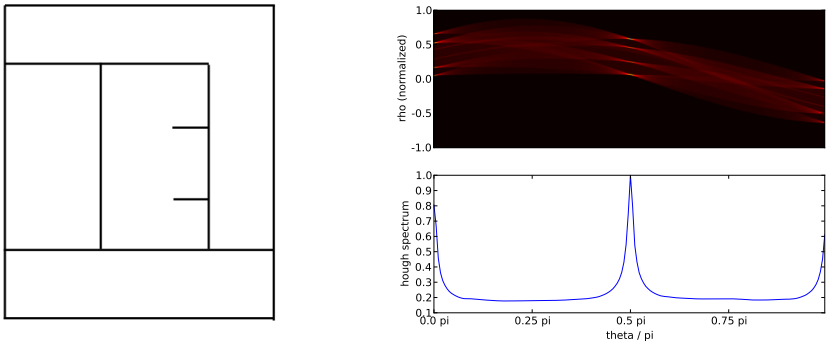
The major contribution of Censi et al. [8] is the introduction of the Hough spectrum. The Hough spectrum ($\mathcal{HS}$) is a measure for the direction of lines in an image. The discrete Hough transform finds the specific lines in the image, while the Hough spectrum finds the most pronounced *directions* of lines in the image:

$$\mathcal{HS}(k) = \eta \sum_{i=1}^{n_\rho} \mathcal{HT}(i, k)^2 \qquad 1 \leq k \leq n_\theta \tag{3.2}$$

η is a normalization value to limit $\mathcal{HS}(k)$ to the domain $[0, 1]$.

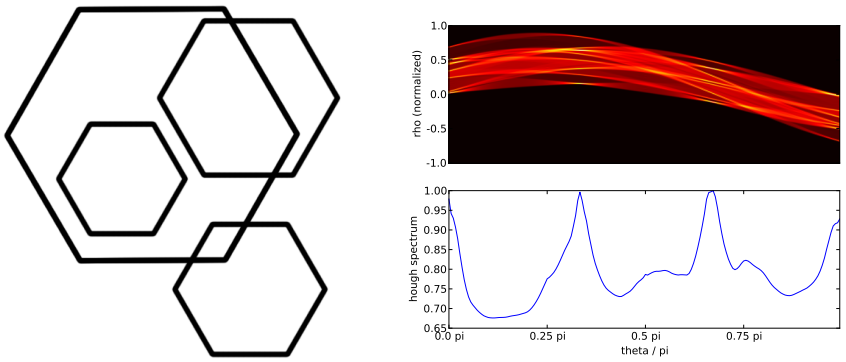
As we've seen in the previous section, each bin in the discrete Hough transform ($\mathcal{HT}(i,k)$) contains the number of pixels that lie on a line defined by (θ_i, ρ_k). The hough spectrum is a measure for how strong the lines are on a specific angle.

Thus, the Hough spectrum yields the highest value in the direction in which the most pronounced lines run. In an indoor environment, there will usually be two peaks, separated by a 90° angle, corresponding to grid-like structure most buildings are created (see figure 3.2). Incidentally, in a bee-hive, with hexagonal chambers, we would see 3 peaks each separated by 60° angles. See figure 3.3.



(a) A stylistic map of some rooms (b) The Hough transform and Hough spectrum of (a)

Figure 3.2: Example of a hough spectrum in a human-made environment.



(a) Hexagonal chambers (b) The Hough transform and Hough spectrum of (a)

Figure 3.3: Example of the Hough spectrum in a hexagonal environment. Notice the three maxima in the Hough Spectrum.

3.3 Finding rotation θ

In the previous section we've seen how to calculate the Hough spectra $\mathcal{HS}_{M_1}$ and $\mathcal{HS}_{M_2}$ for maps M_1 and M_2. We will calculate the best rotation by performing

a cross-correlation over the spectra. The cross-correlation of the two signals shows the similarity of the two spectra as one of them is shifted over the x-axis. The cross-correlation $\mathcal{CC}_{M_1,M_2}$ is calculated as follows:

$$\mathcal{CC}_{M_1,M_2}(k) = \sum_{i=1}^{n_\theta} \mathcal{HS}_{M_1}(i) \times \mathcal{HS}_{M_2}(i+k) \qquad 1 \leq k \leq n_\theta \tag{3.3}$$

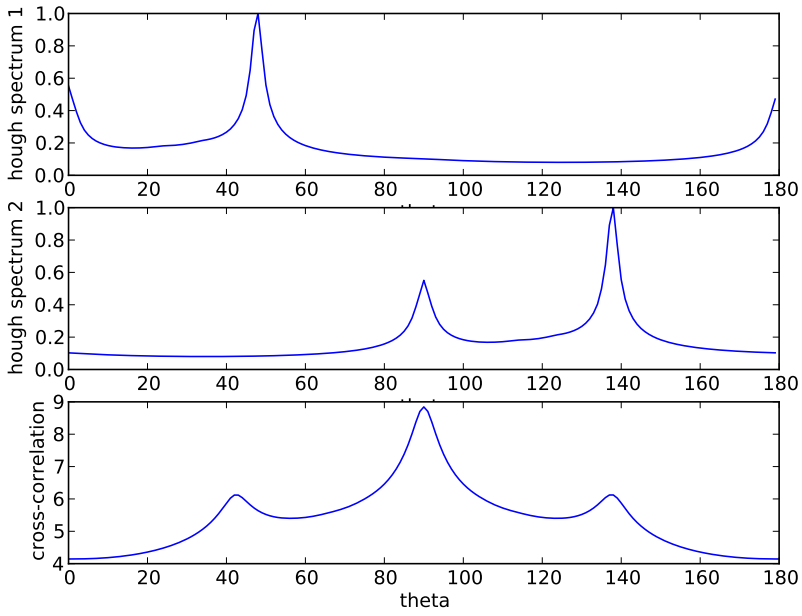


Figure 3.4: The top two plots are the hough spectra of image 3.1(a). The image for the second hough spectrum was rotated 90° counter-clockwise. The cross-correlation shows that the most probable rotation is indeed 90°, with smaller peaks for 45° and 135°.

Notice that the spectra have a periodicity of 180° (π), and have similar peaks. However, the peaks are at different angles. The cross-correlation finds the optimal rotation; see figure 3.4.

3.4 Periodicity of θ

Why are do the Hough spectra have a periodicity of π? The answer lies in the following equations:

$$x\cos\theta + y\sin\theta = \rho \tag{3.4}$$
$$x\cos(\theta+\pi) + y\sin(\theta+\pi) = -\rho \tag{3.5}$$

Thus, the following equation holds: $\mathcal{HT}(\theta_\alpha, \rho_\alpha) = \mathcal{HT}(\theta_\alpha+\pi, -\rho_\alpha)$. In turn,

the Hough spectrum of θ_α and $\theta_\alpha + \pi$ are the same, because $\mathcal{HS}$ is summed over all ρ.

Thus, the domain of θ only needs to be $[0, \pi\rangle$, whereafter it wraps back to itself. Prior work by Jankowska used the domain $[0, 2\pi\rangle$ [12], which is thus superfluous.

One must take care when using this representation to remember the possible alternative hypothesis for θ. Every candidate solution $\hat{\theta}$ found by the cross-correlation method has an additional solution $\hat{\theta} + \pi$!

3.5 Finding translation $\hat{t}$

In the previous section we found rotation candidates which align the maps according to the major wall-directions. To find the best match between two maps, we also need to find the optimal translation between them. Jankowska [12] takes the same approach as Carpin [13], which is a simplification of the method by Censi et al. [8]. We will discuss the more complex method later.

The simplified method is to calculate an X-spectrum and Y-spectrum from the maps, and again perform cross-correlation on these. The X- and Y- spectra are calculated by summing the number of occupied pixels along one dimension; see equations 3.6 and 3.7. See figure 3.5 for an example. The optimal translation is found by cross-correlating the x- and y-spectra from both maps.

$$\text{X-spectrum}(x) = \sum_{y \in Y} \text{occupancyMap}(x, y) \tag{3.6}$$

$$\text{Y-spectrum}(y) = \sum_{x \in X} \text{occupancyMap}(x, y) \tag{3.7}$$

Before the x- and y-spectra are taken of the image, care must be taken to align the image to the predominant direction ϕ. In this way, the angular features of the environment (walls) show up as spikes in the spectrum. If the image is not aligned to the predominant direction ϕ the spectrum yields mostly noise, as can be seen in figure 3.6.

Censi et al. [8] propose a least-squares error optimization to find $\hat{t}$. Instead of taking a X-spectrum and Y-spectrum, the maps are rotated around many different angles and the spectra of those angles are cross-correlated. A least squares error solution is fit on the resulting optima. The X- and Y-spectrum method is a special case of this method; with only 2 constraint sets (one for the x-direction and one for y), the method yields an exact result for the two free parameters (x, y).

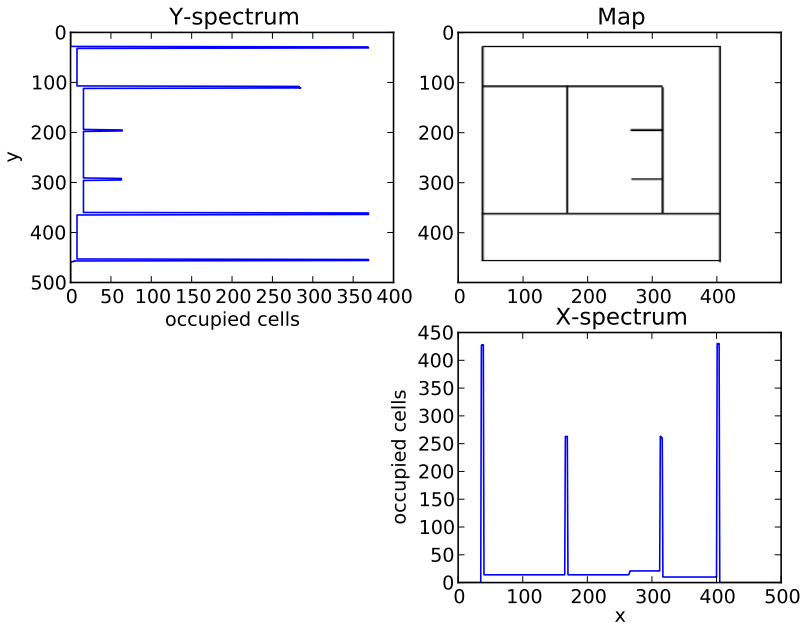


Figure 3.5: The original map (top right) with the y-spectrum (top left) and x-spectrum (bottom right).

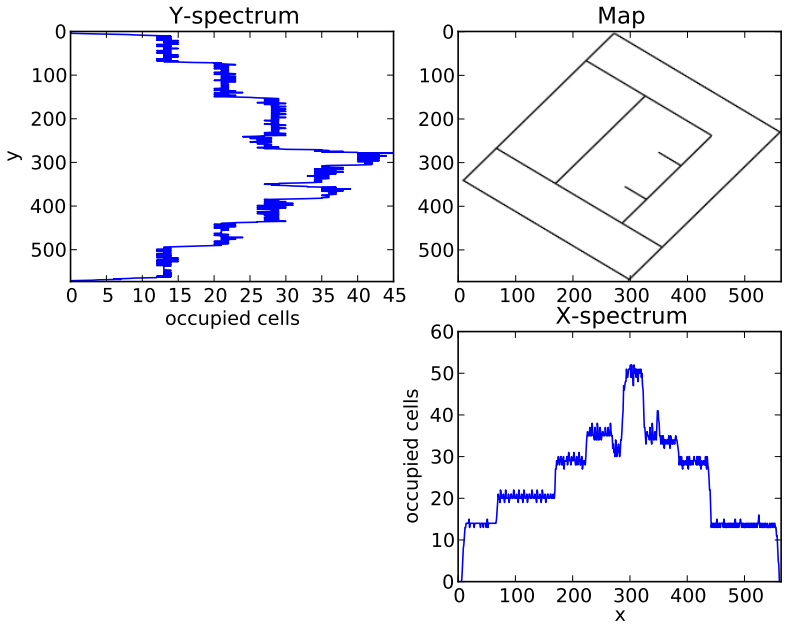
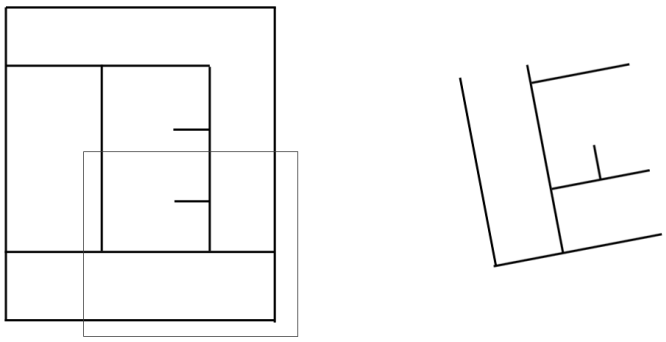


Figure 3.6: The X- and Y-spectra of a map that is not aligned to the predominant direction ϕ. Compare the spectra with those from figure 3.5.

3.6 Example

Let's look at an example. We will examine more complicated cases in chapter 5. In figure 3.7 you see again the image of the artificial map and a small piece which we will try and map onto the other.



(a) Full map (b) Cutout

Figure 3.7: The map and a cutout that we want to match onto the map

The first step is to find candidate translations θ. We do this by calculating the hough spectra and performing cross-correlation on them. See figure **??**. The highest peak in the cross-correlation lies at $\theta_{1a} = 169°$, the second highest at $\theta_{2a} = 79°$. Because of the periodicity of the Hough spectrum there are also two dual-hypothesis: $\theta_{1b} = 180° + 169° = 349°$ and $\theta_{2b} = 180° + 79° = 259°$, respectively. Note that the first hypothesis and the second hypothesis lie 90° apart, which corresponds to the two main wall-directions in the image.

As you can see in figure 3.8, the first image is not aligned with the predominant direction ϕ: the highest peak in the hough spectrum lies at 90°. Thus, we have to rotate it 90° counterclockwise before calculating the x- and y-spectra. Let's test the first hypothesis θ_{1a}. We will rotate the cutout $169° + 90°$ counterclockwise, according to the first hypothesis plus the predominant-direction correction. See figure 3.9. In figure 3.10 you see the x-, y-spectra and their cross-correlations.

The maxima in the X- and Y-spectra yield a translation estimate $\hat{t} = (-60, -89)$. In figure 3.11(a) the result of the stitch is seen. The map is printed in blue, the cutout in pink. Astute readers will have noticed that the rotation estimate is 90° off, and the second hypothesis θ_2 might yield a better result. They are right, as can be seen in figure 3.11 where the results for all θ hypothesis are shown.

In the experiments section, the optimal rotation $\hat{\theta}$ will be chosen as the candidate that is closest in degrees from the estimate the inertia sensor gives.

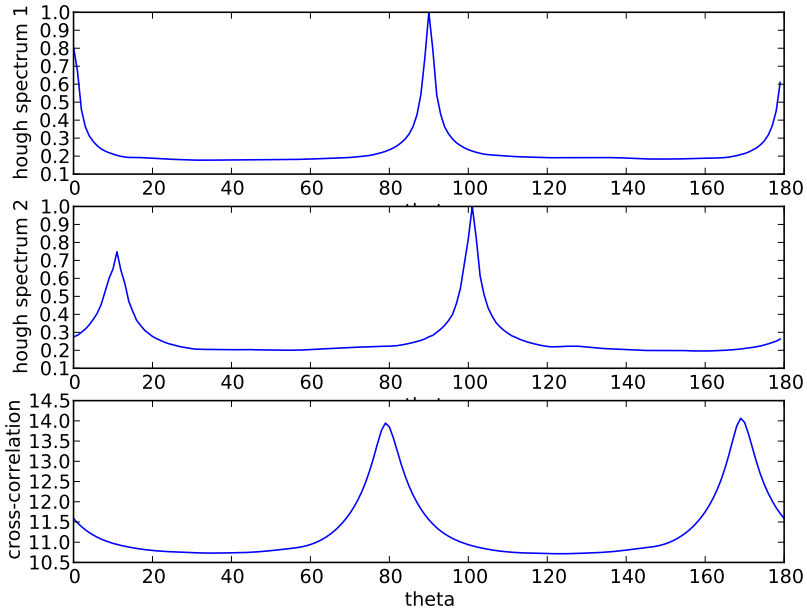


Figure 3.8: The Hough spectra of respectively figure 3.7(a) and 3.7(b) and their cross-correlation.

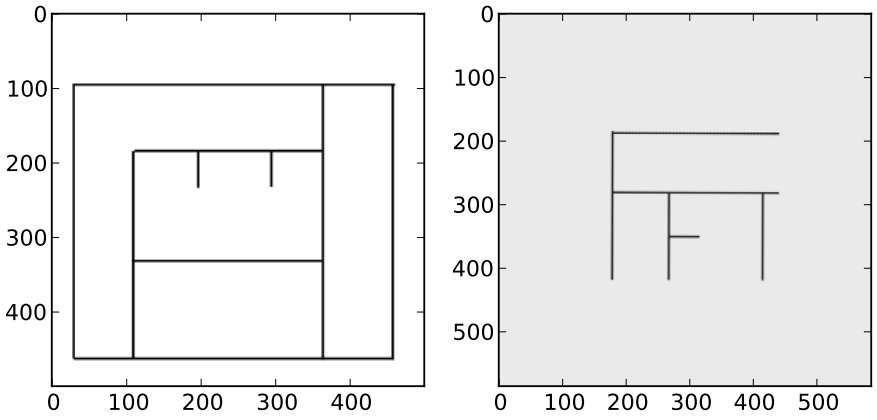


Figure 3.9: The images from figure 3.7 rotated along ϕ and $\phi + \theta_{1a}$, respectively

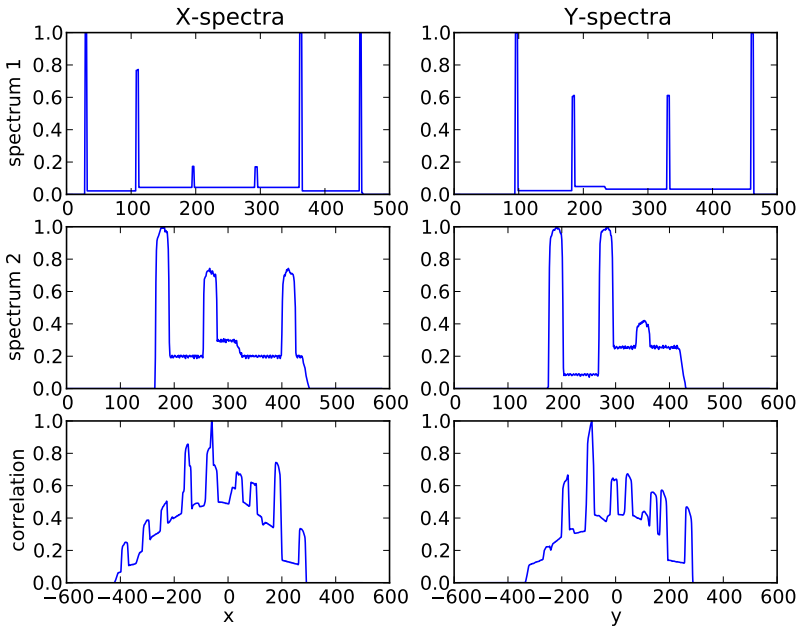
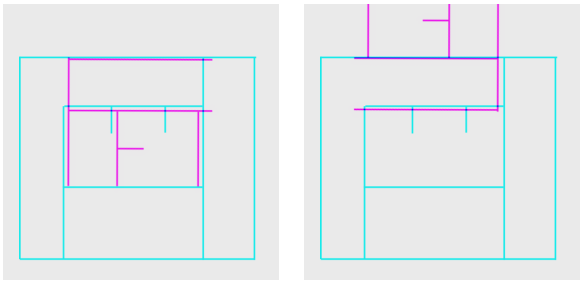
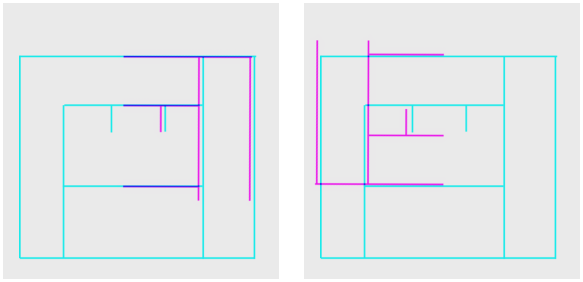


Figure 3.10: The X- and Y-spectra and correlation of the images from figure 3.9.



(a) θ_{1a} (b) θ_{1b}



(c) θ_{2a} (d) θ_{2b}

Figure 3.11: The results of map stitching according to all four θ hypothesis. Hypothesis θ_{2a} yields the correct result.

Chapter 4

Method

In this chapter the design of the experiment is outlined. The experiment consists of three phases: dataset collection, map generation, map segmentation and map stitching. Each of the phases is described in detail below.

4.1 Dataset collection

The dataset for the experiments consists of sensor data acquired by a robot in the USARSim (Urban Search And Rescue Simulation) environment [14, 15]. USARSim is a high fidelity simulation environment in which reconnaissance and rescue missions can be imitated. These environments can both be indoors and outdoors, and are typically not larger than a housing block. A team of robots is deployed in this environment with the task of localizing (human) victims. One of the goals of the USARSim project is to improve autonomous behavior, allowing one human operator to control a team of up to 16 robots simultaneously.

Much work has been done to increase the realism of the simulation. A selection of recent work: an assessment for use of USARSim in Human Robot Interaction [16], for use with walking robots [17], adding realistic laser sensor behavior in environments with smoke [18] and creating a omnidirectional camera sensor [19].

Teams from all over the world compete in the annual RoboCup Rescue Simulation competition [20]. The universities of Oxford and Amsterdam participated until 2011 with a cooperative team: the Amsterdam Oxford Joint Rescue Forces [1, 21]. The robot control software used by this team is UsarCommander (available on the AOJRF site [22]). UsarCommander (see figure **??**) contains a number of SLAM implementations, including ManifoldSLAM.

Three datasets were collected on three recent USARSim maps. The maps in which the data was gathered were selected to be diverse: an indoor map, an outdoor map, and a map with smoke (which impairs sensor readings). The robot was driven around the map. Care was taken to ensure that the robot visited the same locations a number of times to make the map-stitching method feasible.

Each dataset was recorded. The datasets are available within the code repository [23].

4.2 Map segmentation

The datasets thus recorded are processed through the UsarCommander ManifoldSlam implementation. At every timestep the scan matcher runs and saves current position and rotation according to the groundtruth, the inertia sensor and the SLAM algorithm, as well as the correspondence error matrix as discussed in section 2.3.

After creating the map, three uncertainty metrics are evaluated to decide where the map should be segmented. The three metrics are the determinant of the correspondence error, the trace of the correspondence error and the number of matching scans, as discussed in section 2.5.

The three metrics are manually examined for each map, as we found no prior research in a good metric for map segmentation. Obvious candidates are the maxima in the determinant and trace measures and minima for the number of matching scans. These values are expected to be highly correlated. The dataset is segmented at the thus acquired uncertainty threshold into consecutive parts. For example, if the map consisted of 100 timesteps ($0 \ldots 99$) and the uncertainty metrics indicate that the highest uncertainty was at timesteps 11, 25 and 70, the dataset is divided into four segments: $0 \ldots 10$, $11 \ldots 24$, $26 \ldots 69$, and $71 \ldots 99$. A submap is generated for each segment.

4.3 Map stitching

The segments are matched according to the Hough-transform based map stitching method as outlined in chapter 3. In that chapter it is explained that there may be a number of rotation estimates $\hat{\theta}$. We will assume in the experiment that the smallest absolute rotation candidate $|\hat{\theta}|$ will be the most probable. This is based on the observation that the scanmatcher usually does not yield very large rotation errors.

The relative X- and Y-positions of the maps are estimated exactly as explained in the map stitching chapter.

4.4 Evaluation

In line with the exploratory nature of this work the maps will not be evaluated using a groundtruth-metric. Instead, manual assessment of the results will yield a qualitative judgment about the quality of the resulting map. The main concern is whether the resulting map is an improvement upon the map provided by the ManifoldSLAM algorithm.

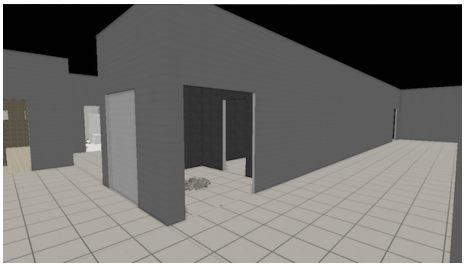
Chapter 5

Experiments

In this chapter the experimental results are presented. The experimental method is outlined in chapter 4.

5.1 Map 1: IranOpen 2012 - Pre 2

This map was used in the Iran Open 2012 competition [24]. The map features a grey hospital-like environment with grey tiled floors and walls. In figure 5.1 a screenshot of the environment is shown along with the map after the Weighted Scan Matcher was run on the simulation data.



(a) Screenshot

(b) Map created with the Weighted Scan Matcher

Figure 5.1: The ground truth, inertia sensor and slam path of the robot on a piece of map 1.

The map shows a lot of noise and errors. As can be seen in figure A.1(a) (in the appendix), the inertia sensor gives a rather good location-estimate, but the rotation estimate from position 160 onwards is off by more than 10°. The scanmatcher fails regularly because it takes the inertia sensor location estimate as begin point for its search. When the location according to SLAM and inertia sensor diverge too far, the SLAM matcher fails – it only searches a local neighborhood around the initial seed pose given by the inertia sensor. The result of this is a map with jagged lines, as can be seen in figure 5.1(b) and figure 5.1.

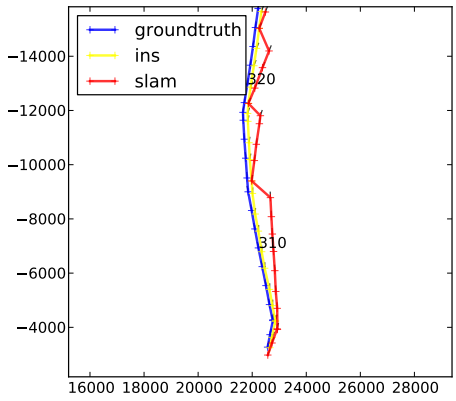
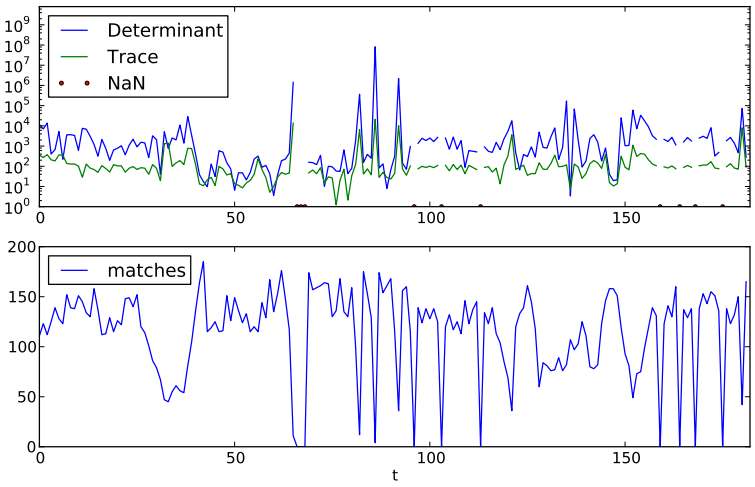


Figure 5.2: A small part of the path the robot moved in map 1. The wrong rotation estimate of the inertia sensor (yellow line) makes the slam-matcher (red line) think the robot moved in another direction than it did in reality (blue line). When the inertia sensor reading and SLAM result diverge too far, the SLAM location is reset to the inertia sensor estimate. This results in a jagged path estimate from the SLAM sensor.

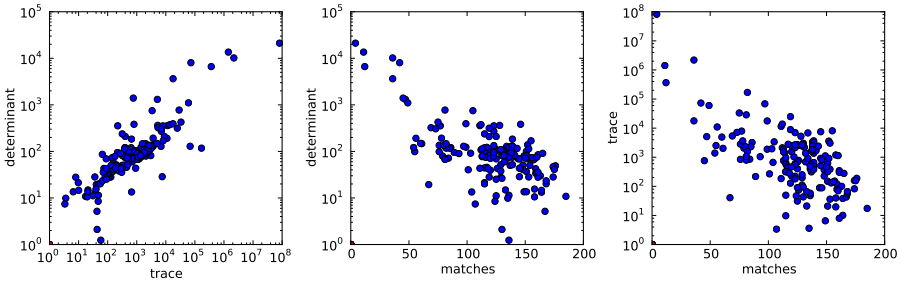
5.1.1 Segmenting the map

The confidence measures of the first map are shown in figure 5.3(a). It is immediately apparent that the extreme values of the three metrics coincide. When the scan matcher matches few scanlines, the determinant and trace values are at their maximum. When the scan matcher matches no scanlines, the determinant and trace of the covariance matrix are undefined. These show up as red dots on the x-axis. When the scan matcher matches many scanlines, its increased confidence in a correct match is reflected in a covariance matrix with small determinant and trace.

In figure 5.3(b) the values of the three confidence measures are plotted against each other to emphasize their correlation. The (Spearman) rank correlation gives an indication how well the relationship between the two variables can be described by a monotonic function. The spearman rank correlation coefficients between the confidence measures is as follows. Between trace and determinant 0.85, between number of matches and determinant -0.50, and between number of matches and trace -0.48, all with a p-value $\ll 10^{-10}$. This means that all three confidence measures are strongly correlated.



(a) Confidence measures through time



(b) Scatter plot between the three confidence measures.

Figure 5.3: Confidence measures for map 1.

When there are no matches at all, the scanmatcher has failed most spectacularly. In that case, the covariance matrix can not even be computed. In extention, the determinant or trace of the covariance matrix can not be computed either. This occurs at the following timesteps: 66 67 68 96 103 113 159 164 168 175. The greatest rift lies at $66 \leq t \leq 68$, where there were 3 consecutive timesteps that could not be matched. The submaps that are procured can be found in the appendix, figure A.2.

5.1.2 Stitching

The Hough map stitching procedure as outlined in chapter 3 between the first two sub-maps results in an optimal rotation θ_1 of 13°, with a much less pronounced secondary hypothesis θ_2 of 103°, as can be seen in figure 5.4. The X- and Y-spectra for θ_{1a} are shown in figure 5.5. The resulting map is shown in figure 5.6.

The result of this stitch is far from optimal. While the rotation angle θ_{1a} is

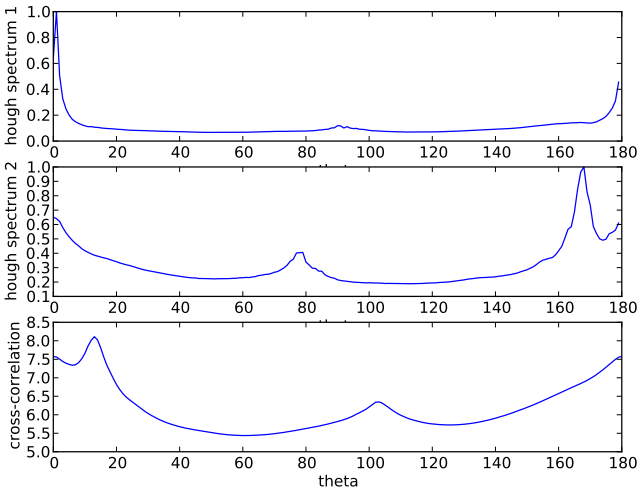


Figure 5.4: Finding optimal rotation θ through correlating Hough spectra.

optimal and the images are rotated correctly, the translation estimate t is very much off. The Hough-transform based map stitching method requires a large overlapping area between the two submaps. Because the submaps overlap very little, the stitching method fails.

In the next figure, 5.7, the final result of stitching all submaps in figure A.2 is shown. Each of the steps is separately shown in the appendix (**??**)

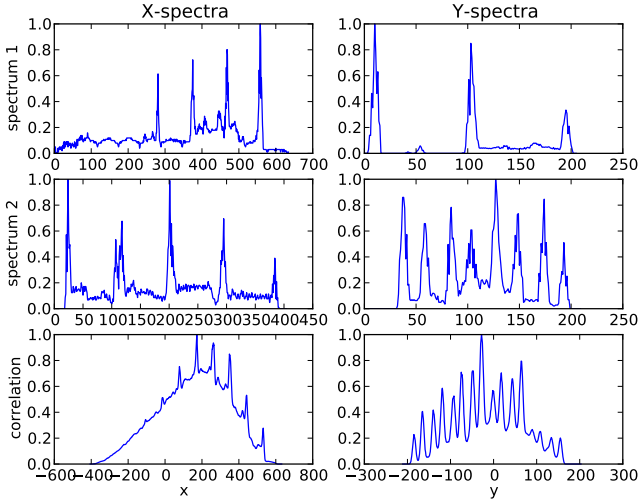


Figure 5.5: Finding optimal translation t through correlating Hough spectra.

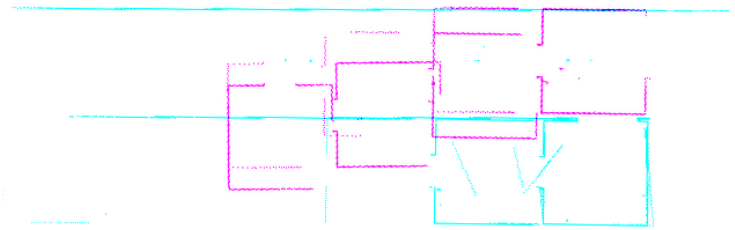


Figure 5.6: The best stitch according to θ_{1a} and optimal t.

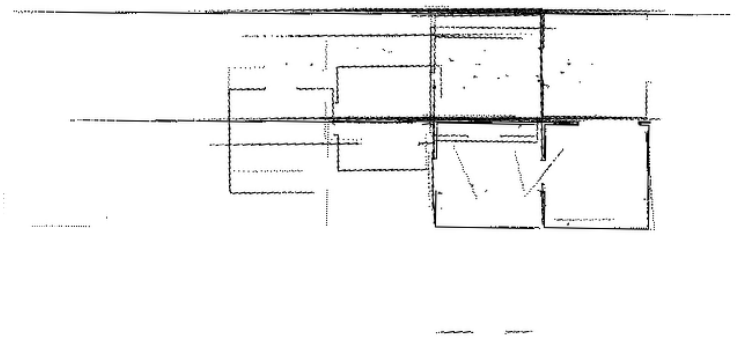


Figure 5.7: The result of stitching all partial maps (see figure A.2) according to the Hough stitching method. Compare to the original SLAM result, figure 5.1(b).

5.2 Map 2

The second map is again a map from the IranOpen 2012 championship. The map was used for the semi-final. It is rather large, and consists of many corridors and rooms filled with desks, chairs and computers. Some of the hallways are obstructed, and there is smoke everywhere. See figure 5.8.

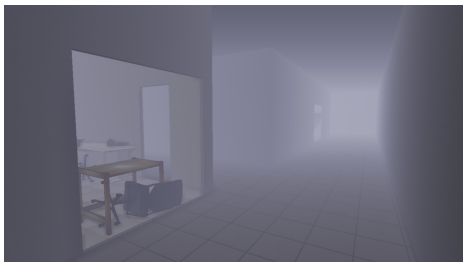


Figure 5.8: A screenshot of map 2.

The agent was driven in a large circular path ($50m$ diameter) through the building, as can be seen in figure 5.9. Again, the path given by the inertia sensor seems to lie closer to the groundtruth than the SLAM path. The SLAM path strays on many places from the path by the inertia sensor and SLAM, and is 'pulled back' to the location given by the inertia sensor when the difference becomes too large.

The error measures extracted from the Weighted Scan Matcher are shown in figure 5.10. Just as in the previous experiments, there are a number of time steps where no matching scan lines were found. At these timesteps, the covariance matrix around the location estimate could not be computed, and are respresented by a red circle at the x-axis. Instead of segmenting the map at every timestep where the number of matches was zero, a different approach is tried for this experiment. The map is segmented on at those timesteps where there are 2 or more consecutive timesteps in which there were no matching scanlines. The resulting 3 submaps are shown in the appendix, figure A.5.

The result of stitching the submaps with the Hough transform based stitching method can be inspected in figure 5.11. As can be easily visually inspected, this result is much worse than the result by the manifold SLAM.

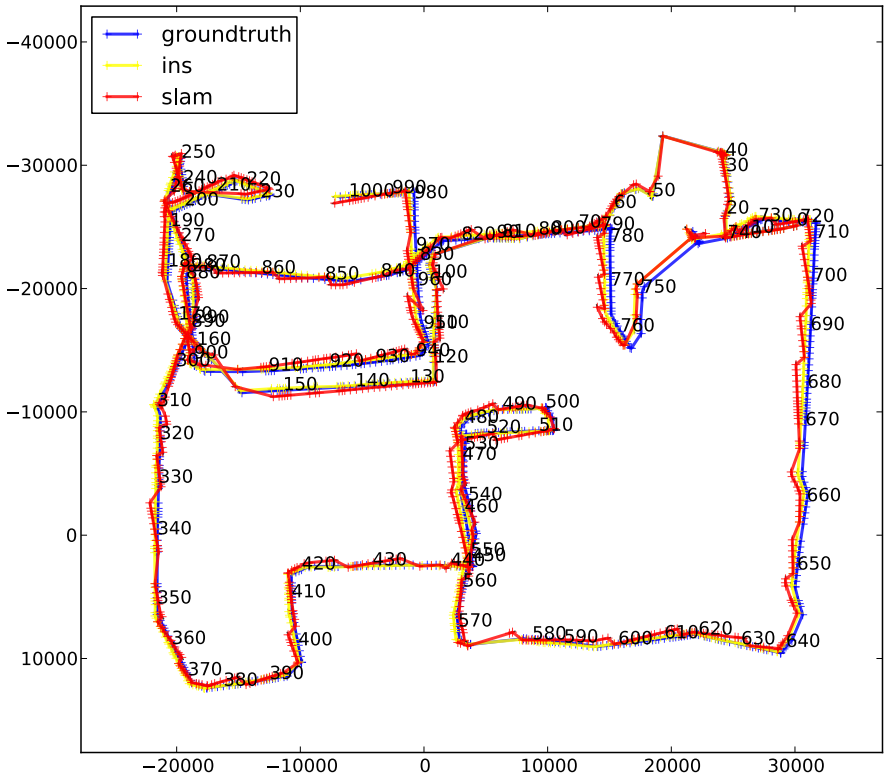


Figure 5.9: The Ground Truth data, inertia sensor data and WSM slam result for map 2.

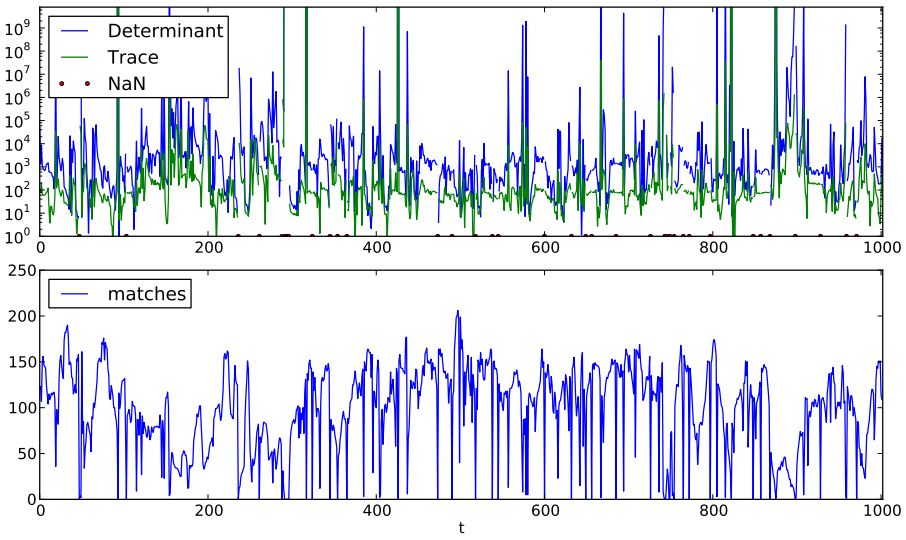


Figure 5.10: Confidence measures for map 2.

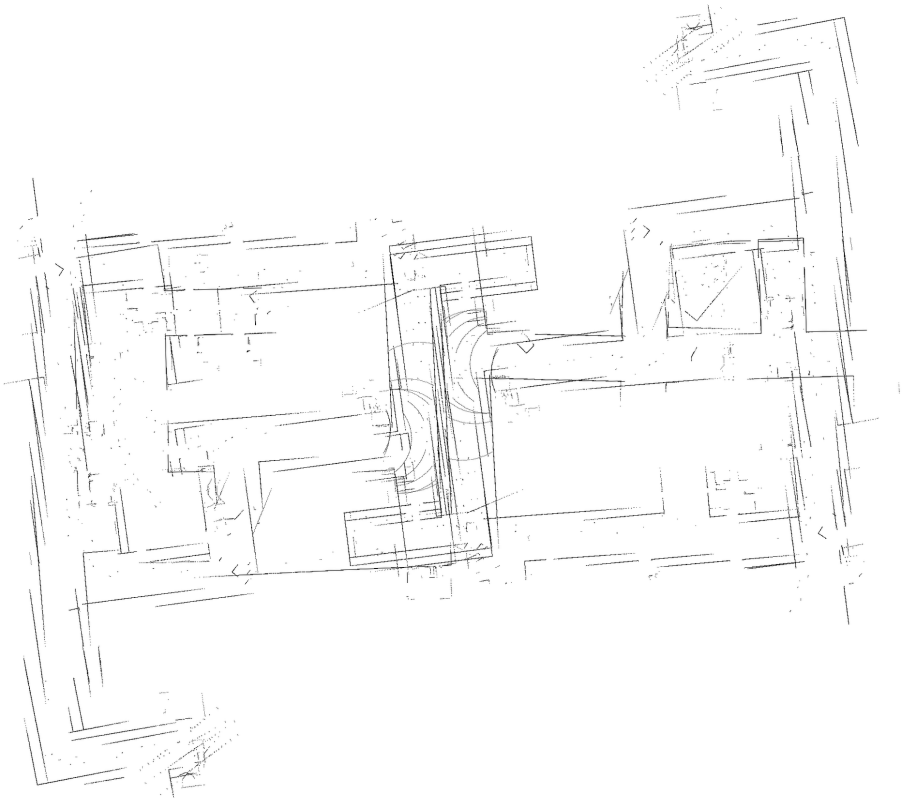


Figure 5.11: The resulting map after stitching all segments extracted from map 2.

5.3 Map 3

Due to the inferior results of the algorithm on the first two maps, the third experiment will deviate from the first two. At the first two maps, the overlap between submaps is rather limited, and not in the

Chapter 6

Discussion and future work

6.1 Discussion

The results of the map stitching method are inferior to the original maps created by ManifoldSLAM. The rotation estimates are usually correct, but the translation estimates are below par. Only for maps with very large overlapping areas, such as in the third experiment, is the result acceptable.

The quality of the rotation estimates is very high in the test environment because the test environments were all human-made. The grid structure in which the walls are placed yields very clear results in the Hough spectrum. In outdoor environments, this is probably not the case, and the results would suffer likewise.

Overall, the existing SLAM appraoches all work very well. This raises the bar for post-processing methods such as the one outlined in this work. Naïve methods which do not use all available information are at a disadvantage.

The X- and Y-spectrum method to find the best translation estimate has proven to be not good enough. Although this method is very simple and has a beautiful symmetrical with the Hough-spectrum method, too much information is lost in the transformation of the map to the spectra. Also, not all available information is being used by this method. A full global search is initiated, while the probable translation of the map is very small. With contemporary technology, the robot can not be teleported to the other side of the map, so it is unnecessary to examine that possibility. However, if the method is used for merging the information from multiple robots, a global search is preferable.

The method to cut the map in smaller pieces seems to work reasonably well - the 'number of scans matching is zero' decision value seemed to select good locations to divide the map into sections. Many of the errors in the map were caused by the rotational error in the inertia sensor. The behavior of the inertia sensor seems to quirky, which will be discussed in section **??**.

Finally, it would have been interesting to see the performance of the Hough based map stitching algorithm on an outdoor environment. However, the poor results of the (probably easier) indoor environments suggest that this method is not yet robust enough for such a challenge.

6.2 Future work

In this section a number of improvements to the Hough map stitching algorithm and the USARSim simulation environment are proposed.

6.2.1 Better translation estimate

The most room for improvement in the Hough-based map stitching method is in the translation estimate. As discussed in the previous section, the X- and Y-spectra do not suffice. Most scanmatching algorithms, as discussed in section 2.3, perform the rotation- and translation estimates separately. The Hough based map stitching method could give a good initial estimate for the rotation estimate, so that the scanmatcher only needs to find the optimal translation.

Additional improvements would be to incorporate the knowledge about the last position of the robot and the inertia sensor estimate as a basis for the translation estimate, as opposed to a global search.

Finally, in the spectrum method it is supposed that the X- and Y- spectra are independent and their maxima are independent as well. They are dependent on each other - the actual real location is a (x, y) coordinate on which x and y are dependent of each other.

6.2.2 Improved peak detection

To find the optimal rotation and translation, an argmax function is run on the spectra. While this usually gave a pretty good result, the peak surrounding the maximum was sometimes skewed left or right. To select the center of such a peak instead of the single maximum result, it might be beneficial to convolve the signal with an Gaussian filter, which smooths the peaks.

6.2.3 Inertia sensor improvements

It seems that the inertia sensor, as it is currently implemented in USARSim, returns inconsistent data. When the rotation estimate has an error, the absolute location estimate from the sensor should be consistent with this. For example, if the rotation estimate is 10° while the real rotation is 0° and the robot moves 5 meter forward, it should return a new position estimate of $(x, y, \theta) = (5\sin(10), 5\cos(10), 10)$ instead of $(5, 5, 10)$. Otherwise, the data it returns is not consistent.

Alternately, the UsarCommander software could use the scanmatcher data to calibrate the inertia sensor. If the inertia sensor registers that the robot has turned, but the laser scan data is not in agreement, the inertia sensor could be 'reset' or it's data could be filtered to fit the laser scan results.

** This is only a list of points that need to be discussed, need to be fleshed out **

It is easy to break the map where the scanmatcher fails - it fails most significantly when it can not find a single matching scan. See also part about ins sensor.

Scanmatcher works too well already - the hough stitching is an inferior solution.

Locating multiple robots might be a better use of the scanmatcher than stiching the map of one robot.

The rotation estimate works very well in an indoor environment, this could be used as input for the scanmatcher.

—-

Ins sensor: should take it's rotation drift into consideration when assessing the next XY coordinates. Otherwise, update the location and rotation by the change in the sensor instead of the full values.

6.3 Future work

** This is only a list of points that need to be discussed, need to be fleshed out **

Find a better X- Y- matching algorithm for the Hough stitching method

Use the quad-tree based scanmatcher, and maybe use this mechanism to stitch maps from multiple robots. Or at least to align them.

Fix the INS sensor, because it is not internally coherent. (Or: don't trust the rotation by the INS sensor)

Acknowledgements

I would like to thank Arnoud Visser for his supervision of the project. His tremendous help in understanding the UsarCommander software was also indispensible. In addition, I would like to thank Julian de Hoog for sending me Magdalena Jankowska's thesis on Hough transform based map stitching.

I would like to thank my friends and family for their support, and in particular Leonie, who supplied the essential final deadline.

Bibliography

[1] N. Dijkshoorn, H. Flynn, O. Formsma, S. van Noort, C. van Weelden, C. Bastiaan, N. Out, O. Zwennes *et al.*, "Amsterdam Oxford Joint Rescue Forces Team Description Paper Virtual Robot competition Rescue Simulation League RoboCup 2011", 2011.

[2] A. Ethembabaoglu, "Active target tracking using a mobile robot in the USARSim", 2007.

[3] S. Pfister, K. Kriechbaum, S. Roumeliotis and J. Burdick, "Weighted range sensor matching algorithms for mobile robot displacement estimation", in "Robotics and Automation, 2002. Proceedings. ICRA'02. IEEE International Conference on", volume 2, pp. 1667–1674, IEEE, 2002.

[4] A. Howard, G. Sukhatme and M. Mataric, "Multirobot simultaneous localization and mapping using manifold representations", *Proceedings of the IEEE*, volume 94(7):pp. 1360–1369, 2006.

[5] S. Thrun, "Probabilistic robotics", *Communications of the ACM*, volume 45(3):pp. 52–57, 2002.

[6] B. Slamet, M. Pfingsthorn, N. Vlassis, A. Visser, L. Dorst, W. Jansweijer *et al.*, "Manifoldslam: a multi-agent simultaneous localization and mapping system for the robocup rescue virtual robots competition", 2006.

[7] A. Visser, B. Slamet and M. Pfingsthorn, "Robust weighted scan matching with quadtrees", 2009.

[8] A. Censi, L. Iocchi and G. Grisetti, "Scan matching in the Hough domain", in "Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on", pp. 2739–2744, Ieee, 2005.

[9] P. Hough, "Method and means for recognizing complex patterns", December 18 1962, uS Patent 3,069,654.

[10] R. Duda and P. Hart, "Use of the Hough transformation to detect lines and curves in pictures", *Communications of the ACM*, volume 15(1):pp. 11–15, 1972.

[11] D. Ballard, "Generalizing the Hough transform to detect arbitrary shapes", *Pattern recognition*, volume 13(2):pp. 111–122, 1981.

[12] M. Jankowska, "A hough transform based approach to map stitching", 2009.

[13] S. Carpin, "Merging maps via Hough transform", in "Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on", pp. 1878–1883, IEEE, 2008.

[14] B. Balaguer, S. Balakirsky, S. Carpin, M. Lewis and C. Scrapper, "USARSim: a validated simulator for research in robotics and automation", in "Workshop on Robot Simulators: Available Software, Scientific Applications, and Future Trends at IEEE/RSJ", 2008.

[15] "Urban Search and Rescue simulation website: `http://sourceforge.net/apps/mediawiki/usarsim/index.php?title=Main_Page`", 2012.

[16] J. Wang, M. Lewis, S. Hughes, M. Koes and S. Carpin, "Validating usarsim for use in hri research", in "Proceedings of the Human Factors and Ergonomics Society Annual Meeting", volume 49, pp. 457–461, SAGE Publications, 2005.

[17] S. van Noort and A. Visser, "Validation of the dynamics of an humanoid robot in USARSim", 2012.

[18] O. Formsma, N. Dijkshoorn, S. van Noort and A. Visser, "Realistic simulation of laser range finder behavior in a smoky environment", *RoboCup 2010: Robot Soccer World Cup XIV*, pp. 336–349, 2011.

[19] T. Schmits and A. Visser, "An omnidirectional camera simulation for the USARSim world", *RoboCup 2008: Robot Soccer World Cup XII*, pp. 296–307, 2009.

[20] "RoboCup general website `http://www.robocup.org/`", 2012.

[21] A. Visser, N. Dijkshoorn, S. van Noort, O. Zwennes, M. de Waard, S. Katt and R. Rozeboom, "UvA Rescue Team Description Paper Virtual Robot competition Rescue Simulation League RoboCup 2012", 2012.

[22] "Amsterdam Oxford Joint Rescue Forces website `http://www.jointrescueforces.eu/`", 2012.

[23] "GitHub code repository `http://www.github.com/okke-formsma/slam-confidence/`", 2012.

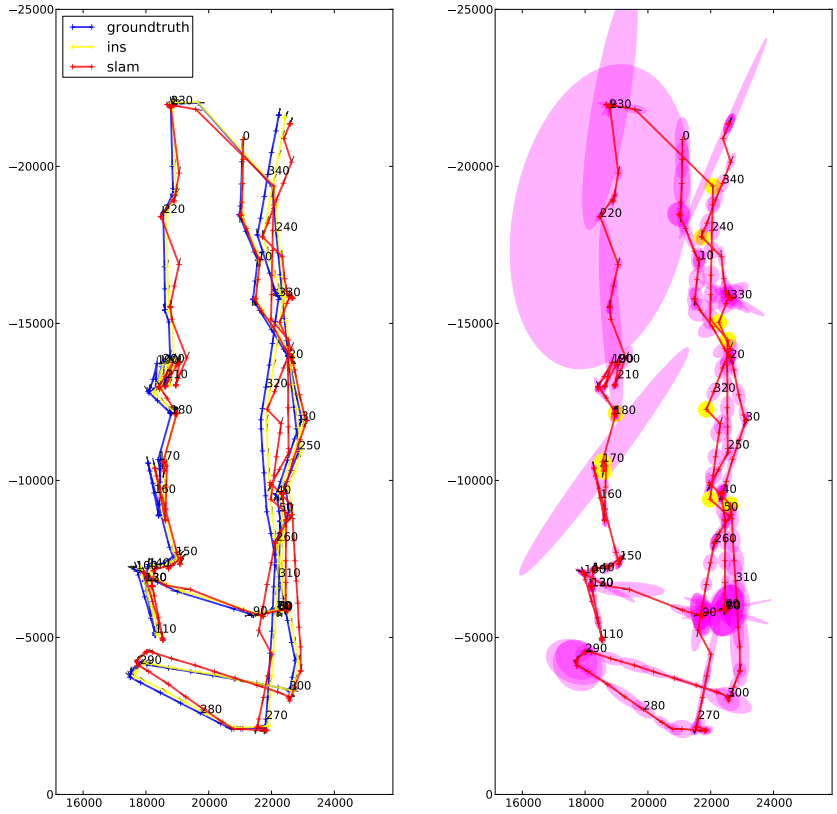
[24] "Iran Open 2012 website: `http://2012.iranopen.ir/`", 2012.

Appendix A

Additional images

This appendix contains additional images belonging to the experiments which were too unwieldy to put in the main text body.

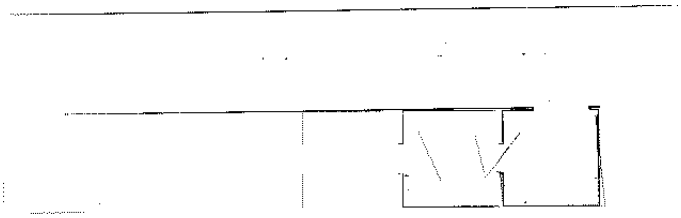
A.1 Experiment 1



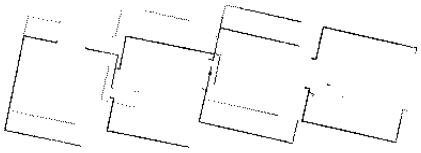
(a) The path of the robot according to the groundtruth, inertia sensor and SLAM

(b) The path of the robot according to SLAM, with associated uncertainty ellipsis. The error is plotted with yellow circles if there was infinite error (no matches at all).

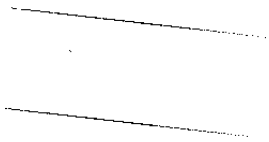
Figure A.1: Map 1 paths



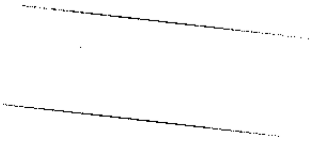
(a) Piece 1: $0 \leq t \leq 65$



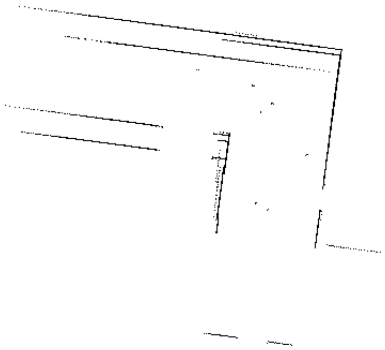
(b) Piece 2: $69 \leq t \leq 95$



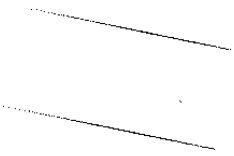
(c) Piece 3: $97 \leq t \leq 102$



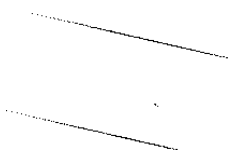
(d) Piece 4: $104 \leq t \leq 112$



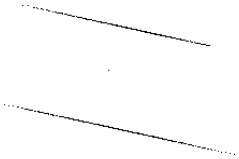
(e) Piece 5: $114 \leq t \leq 158$



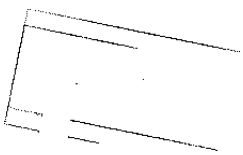
(f) Piece 6: $160 \leq t \leq 163$



(g) Piece 7: $165 \leq t \leq 167$

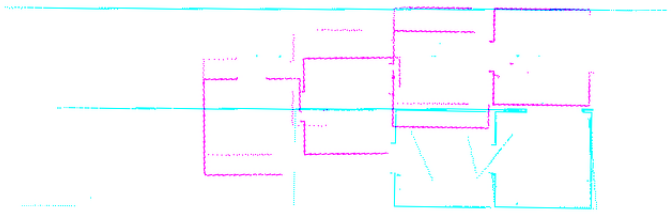


(h) Piece 8: $169 \leq t \leq 174$

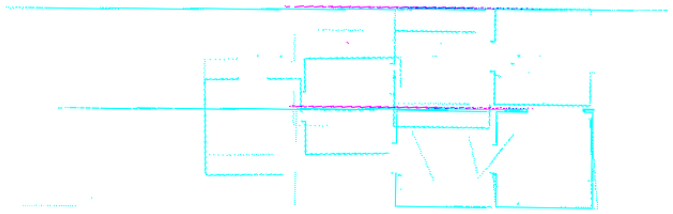


(i) Piece 9: $176 \leq t \leq 182$

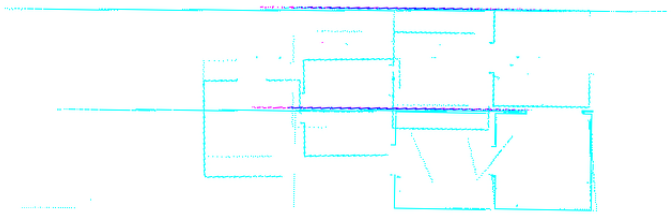
Figure A.2: Map 1 parts



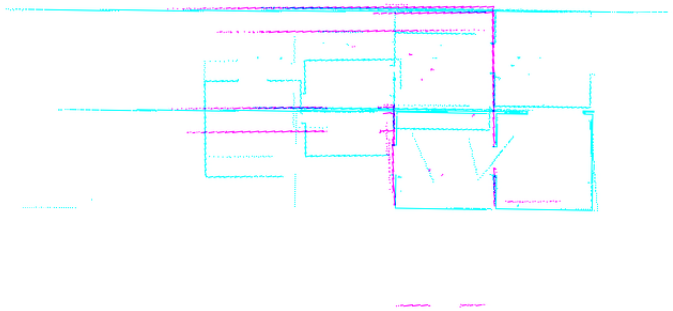
(a) step 1



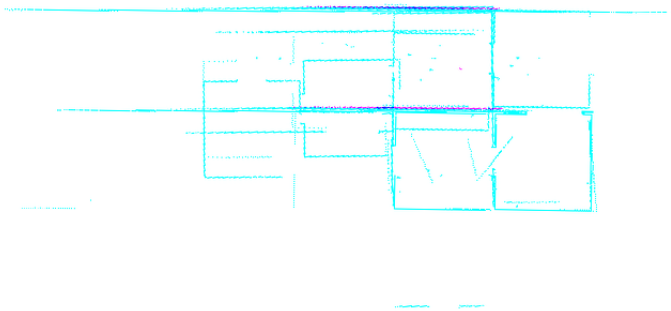
(b) step 2



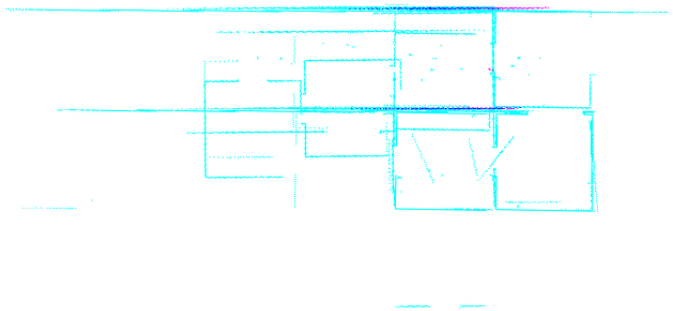
(c) step 3



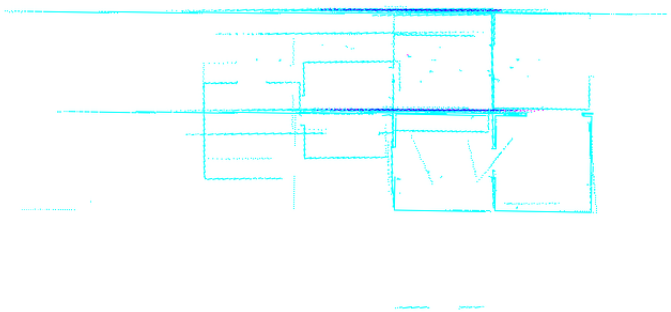
(d) step 4



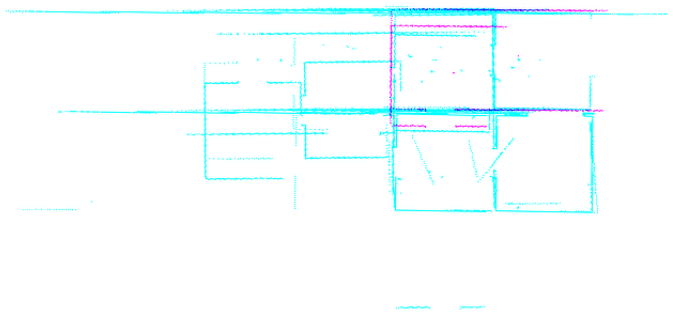
(e) step 5



(f) step 6



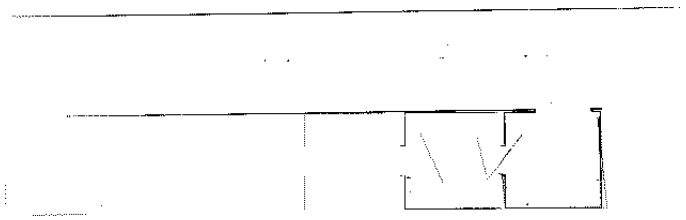
(g) step 7



(h) step 8

Figure A.3: Map 1 stitching steps

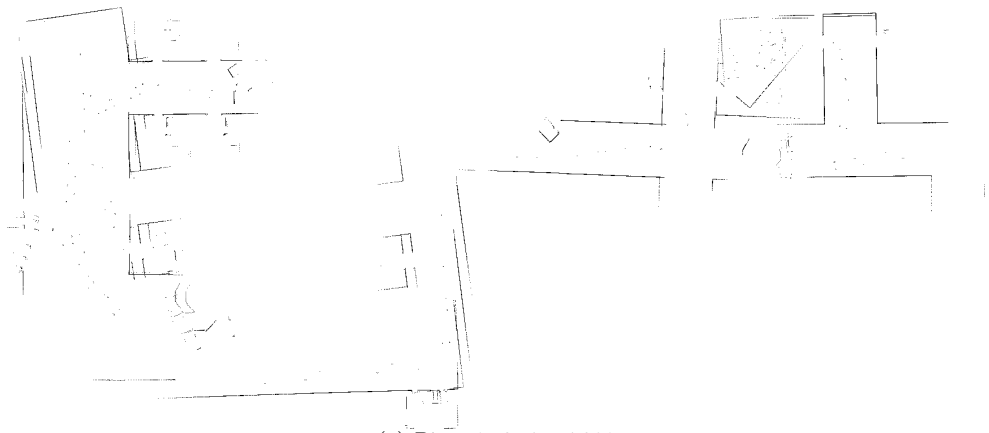
A.2 Experiment 2



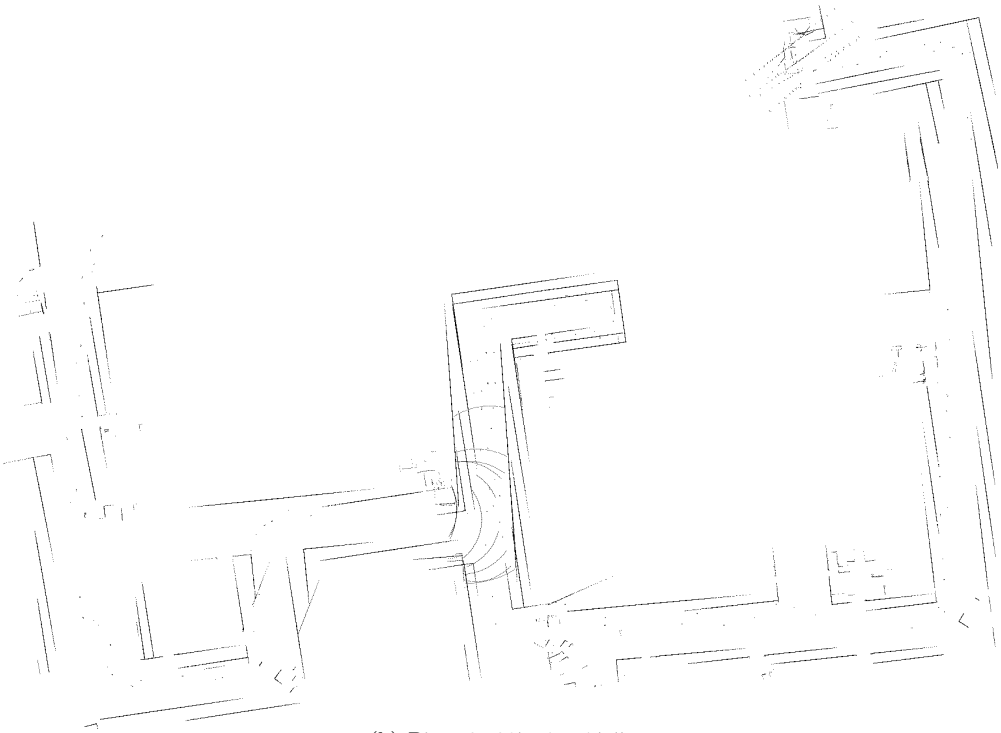
(a) Piece 1: $0 \leq t \leq 65$

Figure A.4: Map 2 parts

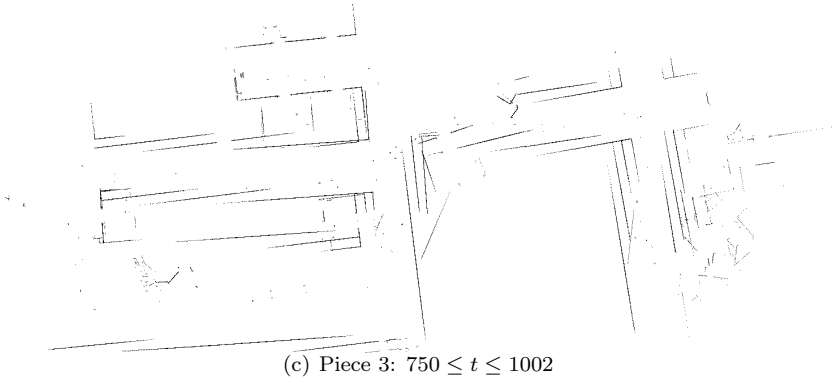
A.3 Experiment 3



(a) Piece 1: $0 \leq t \leq 291$

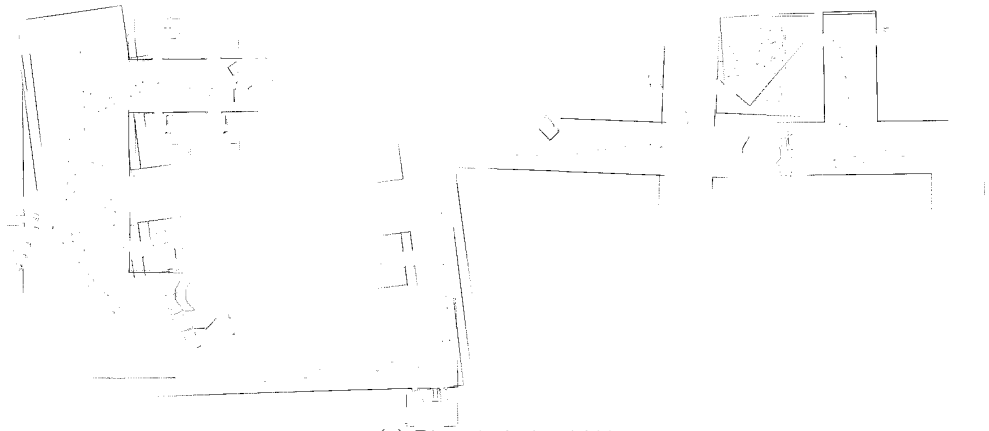


(b) Piece 2: $297 \leq t \leq 747$

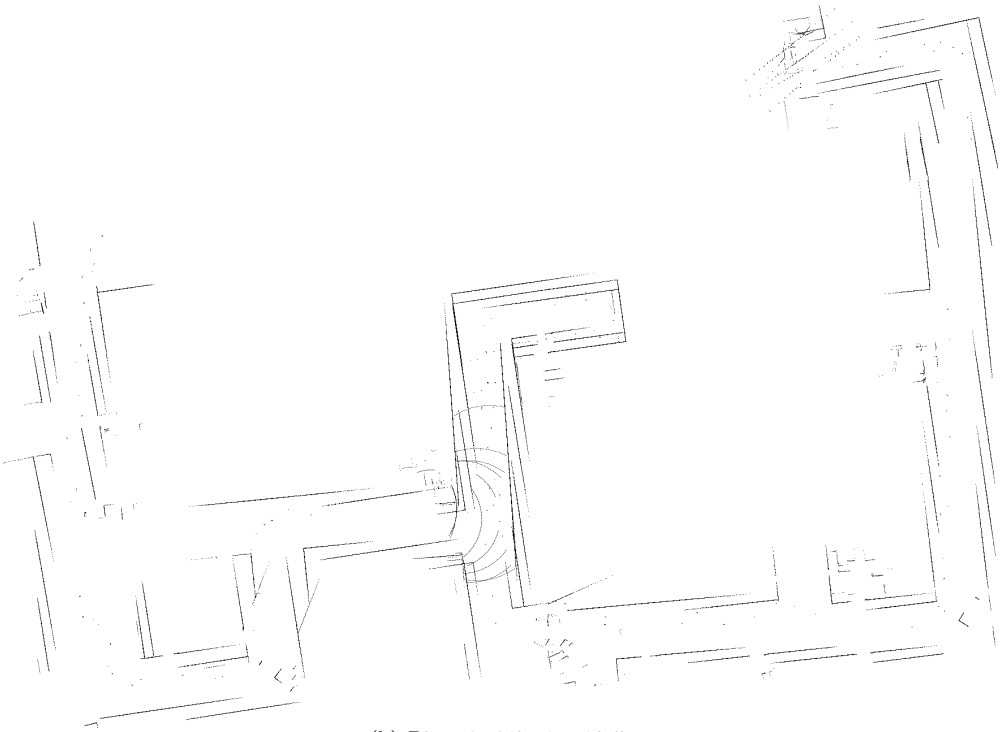


(c) Piece 3: $750 \leq t \leq 1002$

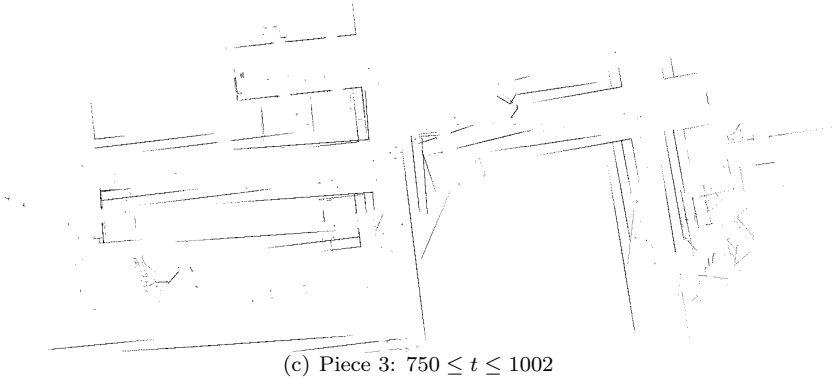
Figure A.5: Map 3 parts



(a) Piece 1: $0 \leq t \leq 291$



(b) Piece 2: $297 \leq t \leq 747$



(c) Piece 3: $750 \leq t \leq 1002$

Figure A.6: Map 3 results