Hough-transform based map stitching after localization uncertainty

Okke Formsma
University of Amsterdam

August something, 2012

Abstract

Abstract

Acknowledgements

Thanks Arnoud, etc. Thanks Julian? Thanks Magda?

Contents

Chapter 1

Introduction

The Simulatenous Localization And Mapping (SLAM) method employed by the AJORF (Amsterdam-Oxford Joint Rescue Forces team ¡¡cite team description paper¿¿ 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.

1.1 Background

1.1.1 USAR Sim

The Urban search and rescue challenge

What's wrong with SLAM

Why Map Stitching through hough transform

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.

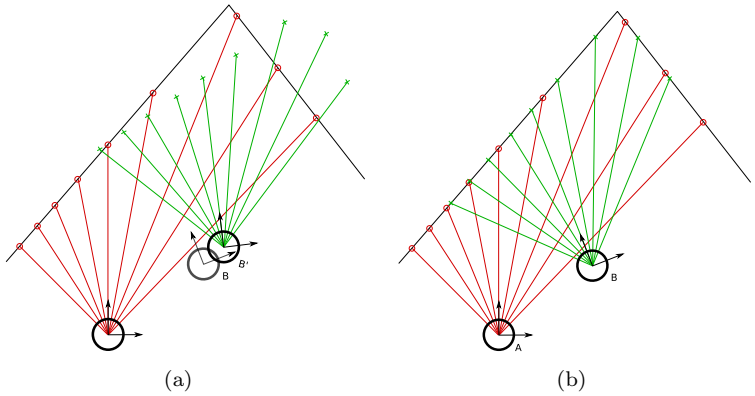
In UsarCommander, the state is represented by a 3-dimensional vector: (x, y, θ) and the corresponding 3-by-3 covariance matrix Σ. x and y are the respective coordinates of the robot in a global frame, while θ is the rotation angle of the robot with respect to the y axis.

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 [1].

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.

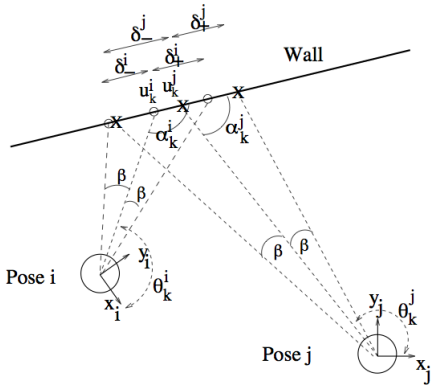


Figure 2.2: Geometry of the scan matching process. Courtesy of [3].

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 [1] 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]. The world/map representation is

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 AJORF team takes the map m_{t-1} and previous position x_{t-1} as given.

2.5 2D slam in 3D environment

Only a slice of the world is returned - so we need to make sure we are horizontal. Otherwise, the scanner returns wrong readings.

2.6 Uncertainty measure

UNCERTAINTY MEASURE BY ARNOUD: To see if the reduced correlation distance also improves the robustness of the scan matcher the uncertainty measures returned by the algorithms were analyzed. This uncertainty measure is the

full 3-by-3 covariance matrix of the Gaussian distribution over the displacement estimate and does not lend itself well for charting in its original form. Therefore we took the on-diagonal elements that describe the independent uncertainty in x and y direction and combined these into a Euclidean distance measure. The trace of the covariance matrix acquired with Q-WSM is in most cases (94average uncertainty of the incremental version.

Chapter 3

Map Stitching with the Hough Transform

Why hough based? Alternatives?

[what is map stitching, why is it necessary again?]

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* [5]. 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[6]. An improved version, which is presented here, was introduced by Duda and Hart[7]. The Hough transform is also extended to detect circles, ellipsis and parabolas, and also arbitrary 2D shapes[8].

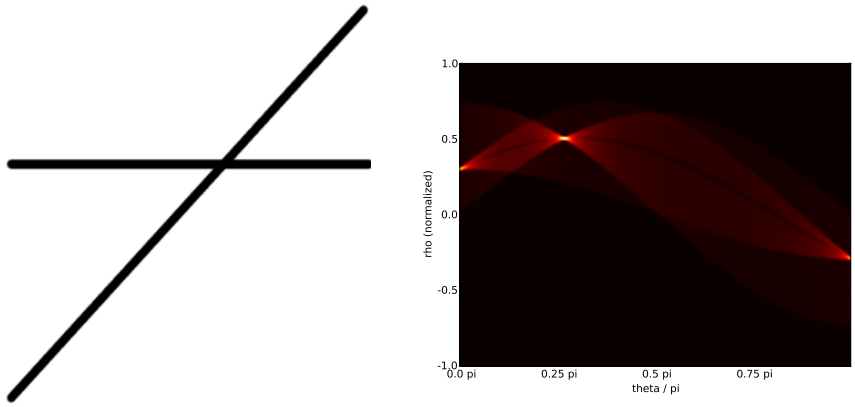
The Hough transform detects lines in the following manner. A line in Cartesian (x, y) space can be defined by it's angle (θ) and distance from the origin ρ:

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

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. [5] 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:

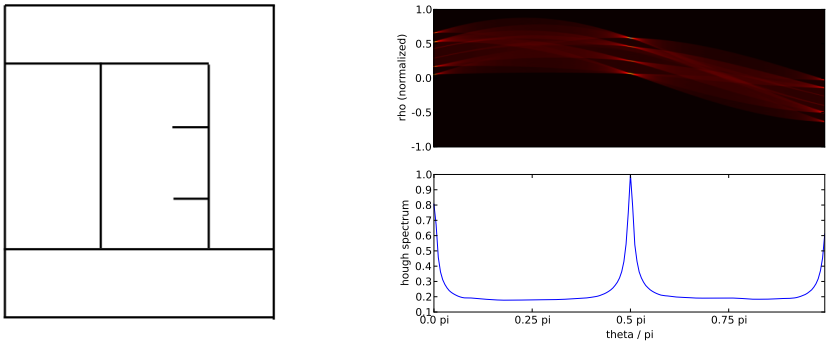
$$\mathcal{HS}(k) = \eta \sum_{i=1}^{n_\rho} \mathcal{HT}(i, k)^2 \qquad 1 \leq k \leq n_\theta \tag{3.2}$$

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

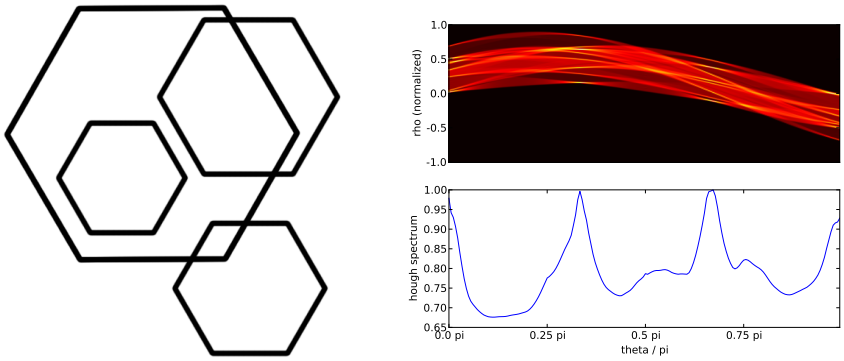
(θ_i, ρ_k). The hough spectrum is a measure for how strong the lines are on a specific angle.

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:

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

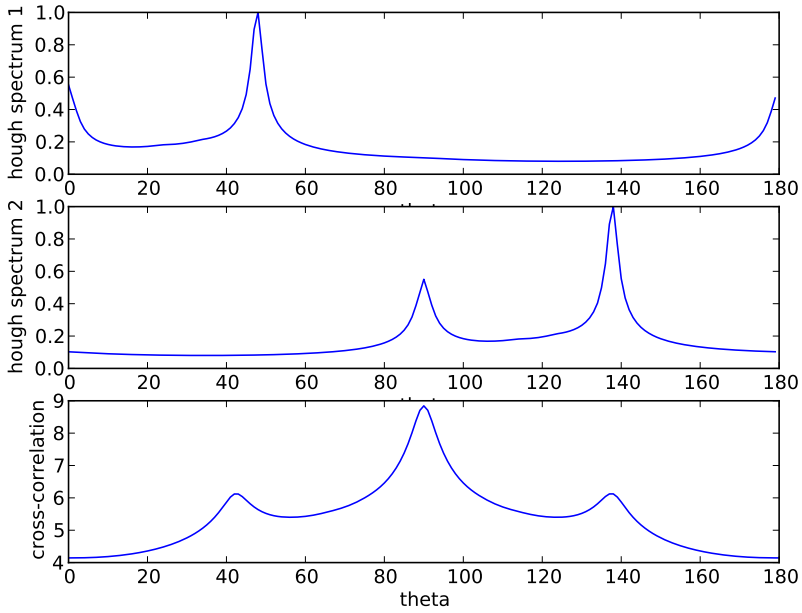


Figure 3.4: The top two plots are the hough spectra of image 3.1(a). The image for the second hough spectrum was rotated 90° counter-clockwise. The cross-correlation shows that the most probable rotation is indeed 90°, with smaller peaks for 45° and 135°.

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 $[0, 2\pi\rangle$ [9], 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 [9] takes the same approach as Carpin [10], which is a simplification of the method by Censi et al. [5]. 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. [5] 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).

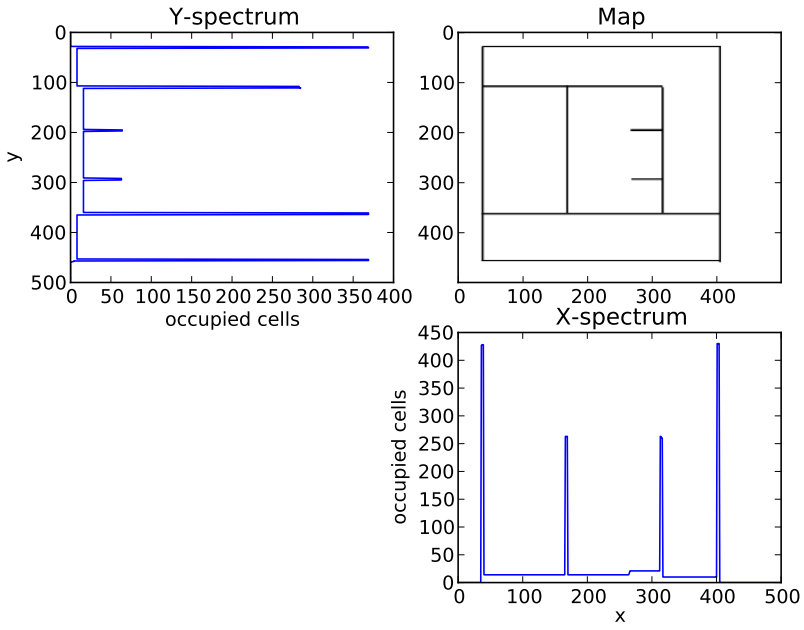


Figure 3.5: The original map (top right) with the y-spectrum (top left) and x-spectrum (bottom right).

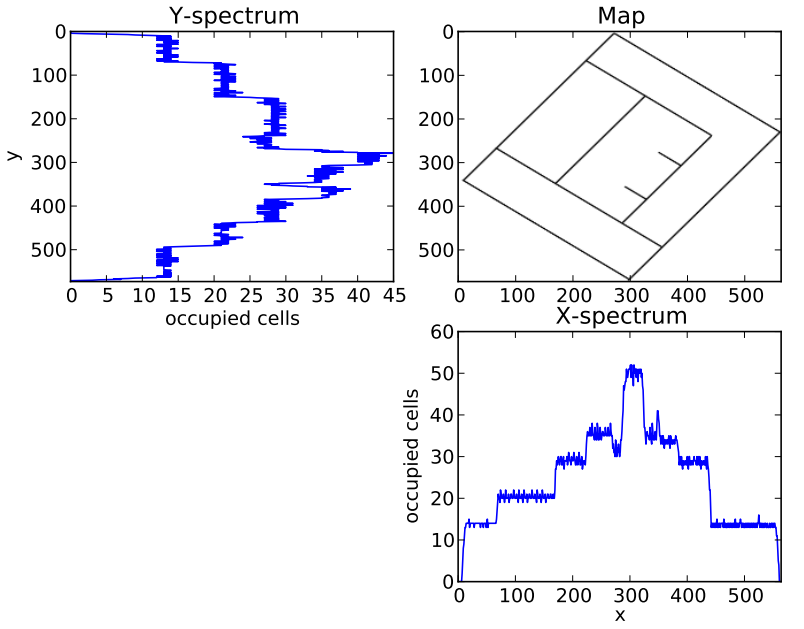
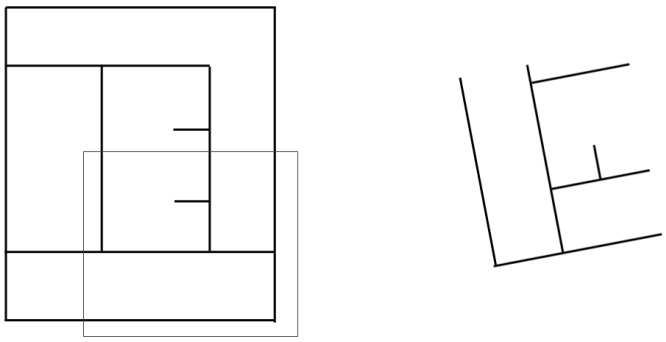


Figure 3.6: The X- and Y-spectra of a map that is not aligned to the predominant direction ϕ. Compare the spectra with those from figure 3.5.

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.

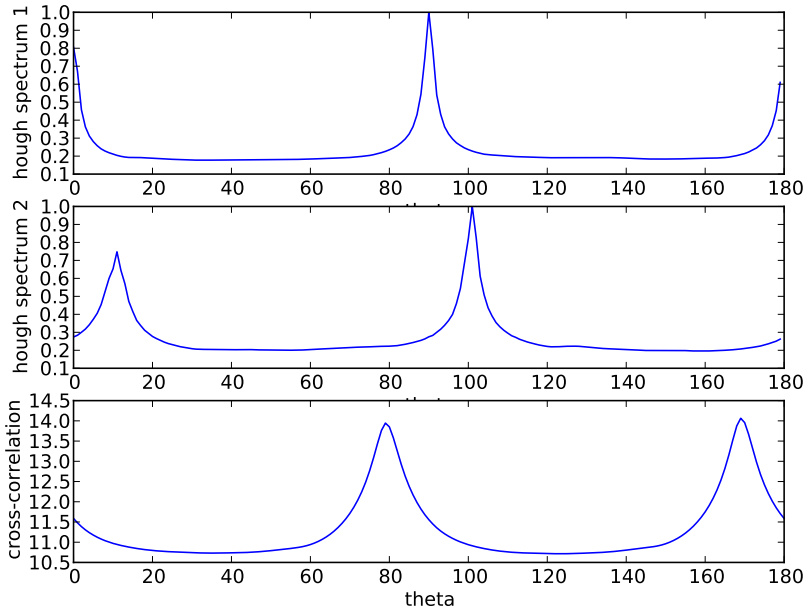


Figure 3.8: The Hough spectra of respectively figure 3.7(a) and 3.7(b) and their cross-correlation.

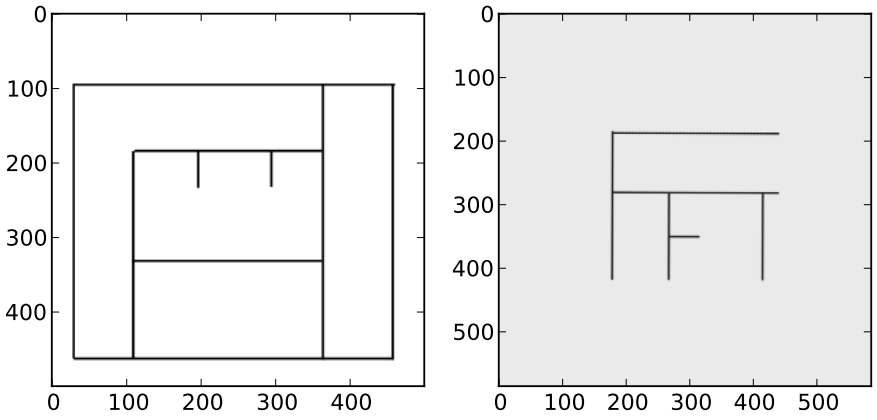


Figure 3.9: The images from figure 3.7 rotated along ϕ and $\phi + \theta_{1a}$, respectively

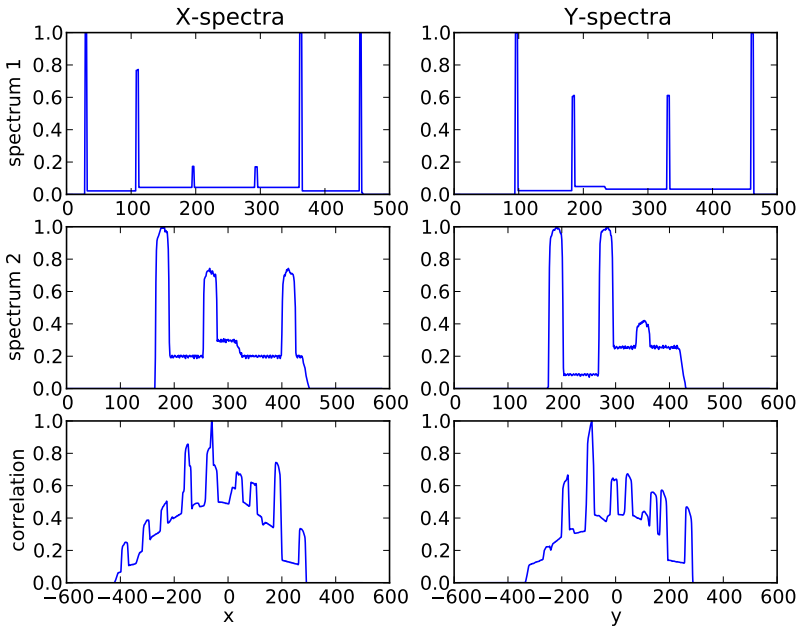
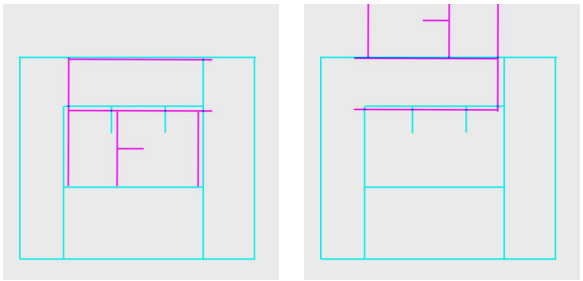
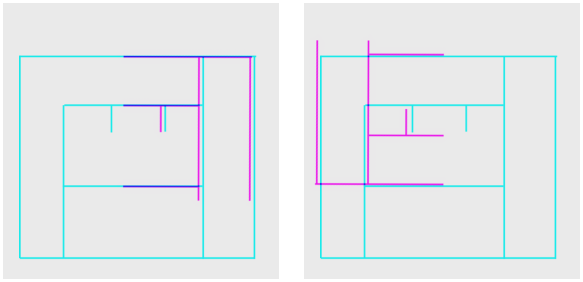


Figure 3.10: The X- and Y-spectra and correlation of the images from figure 3.9.



(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

4.1 Map Stitching

Map stitching prior work, magda etc

4.2 SLAM confidence measure

From Bayu et al: "cite bayu and max: Good examples of single confi- dence values that can be derived from a covariance matrix are the determinant det() or the trace trace(). The relations lend themselves very well for use in path-planning and search algorithms. In addition, heuristic algorithms can take advantage of the uncertainty information stored on the links. (P61)"

From SlametPfingstorn:

To asses the quality of fit of a new range scan, the same notion of being well-fitted is used as during the construction of local sub-maps (refer to Section 5.3.3). Recall that we already obtained the covariance matrix t of the Gaussian distribution that was estimated by the scan matcher. The Fisher information matrix I is computed by taking the inverse of this covariance matrix I = 1 ttt and we compute the 95% confidence interval for every pose variable as follows: = 1.96I1

1) Determinant of covariance matrix

2) Trace of covariance matrix

3) Whether there are any matches at all

4) Attack angle?

The confidence measure based on the determinant of the covariance matrix of the pose uncertainty. The hypothesis is that if this measure is larger, then the confidence is lower (the uncertainty is bigger).

The determinant of the covariance matrix is used as metric:

is the sample variance-covariance matrix for observations of a multivariate vector of p elements. The determinant of D, in this case, is sometimes called the generalized variance. (http://www.itl.nist.gov/div898/handbook/pmc/section5/pmc532.htm)

4.3 Inertial sensor data

Gives the rotations on three axis, enables you to detect

4.4 Confidence measures

(paper not found): L. Carlone, M. Kaouk Ng, J. Du, B. Bona, and M. Indri, Reverse KLD-Sampling for measuring uncertainty in Rao-Blackwellized Particle Filters SLAM, in Proc. of the 2009 IEEE/RSJ Int. Conf. on Intelligent RObots and Systems, Workshop on Performance Evaluation and Benchmarking for Next Intelligent Robots and Systems, 2009.

Carlone et al. use a particle filter based representation of their belief.

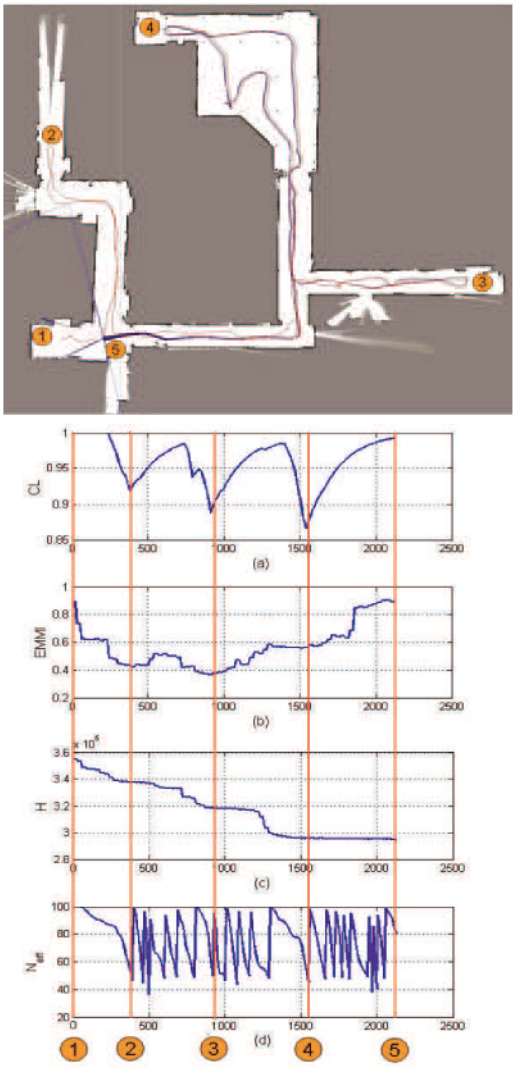


Fig. 4: Time-history of the different uncertainty estimators obtained in real experiment performed in Politecnico di Torino (100 particles): (a) CL, (b) *EMMI*, (c) Map Entropy *H* and (d) The effective sample size *Neff*.

Figure 4.1:

Chapter 5

Experiments

In this chapter the experimental results are outlined. The experimental method is outlined in chapter 4. For the experiment, a number of simulation runs are recorded in USAR Sim. The map is then

5.1 Experimental Runs

5.1.1 Map 1: IranOpen 2012 - Pre 2

This map was used in the Iran Open 2012 competition [11]. 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.

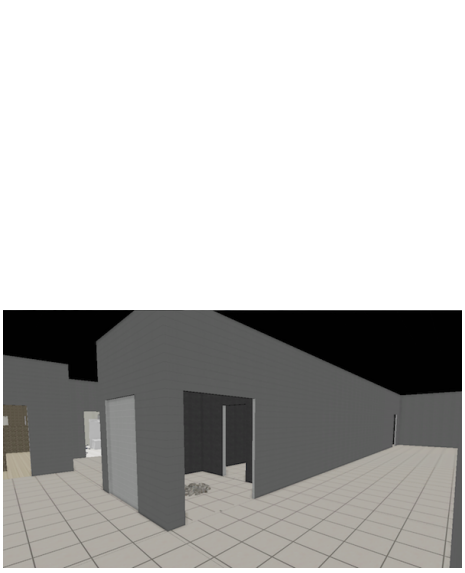
The map shows a lot of noise and errors. As can be seen in figure 7.1(a) (in the appendix), the inertia sensor gives a rather good location-estimate, but the rotation estimate from position 160 onwards is off by more than 10°. The scanmatcher fails regularly because it takes the inertia sensor location 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.

5.1.2 Map 2

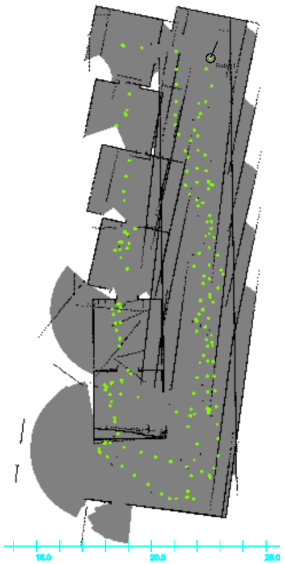
Gotta make a second run.

5.1.3 Map 3

Gotta make a third run.



(a) Screenshot



(b) Map created with the WSM

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

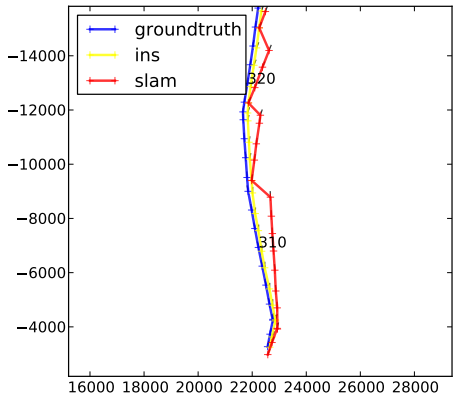


Figure 5.2: A small part of the path the robot moved in map 1. The wrong rotation estimate of the inertia sensor (yellow line) makes the slam-matcher (red line) think the robot moved in another direction than it did in reality (blue line). When the inertia sensor reading and SLAM result diverge too far,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

5.2 Confidence measures

The confidence measures of the first map are shown in figure **??**.

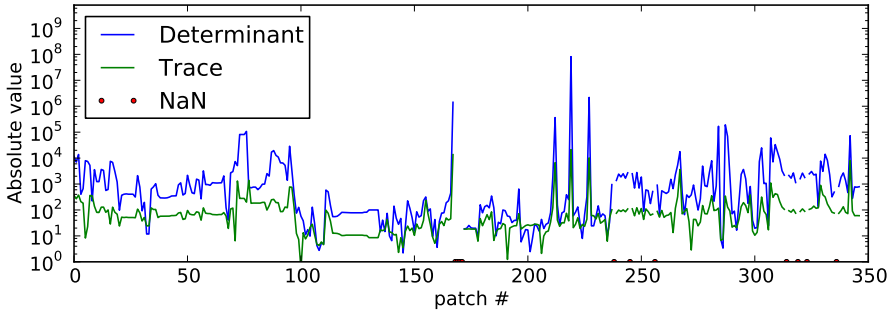


Figure 5.3: Confidence measures for map 1.

5.3 Map segments

5.4 Stitching

5.5 Results

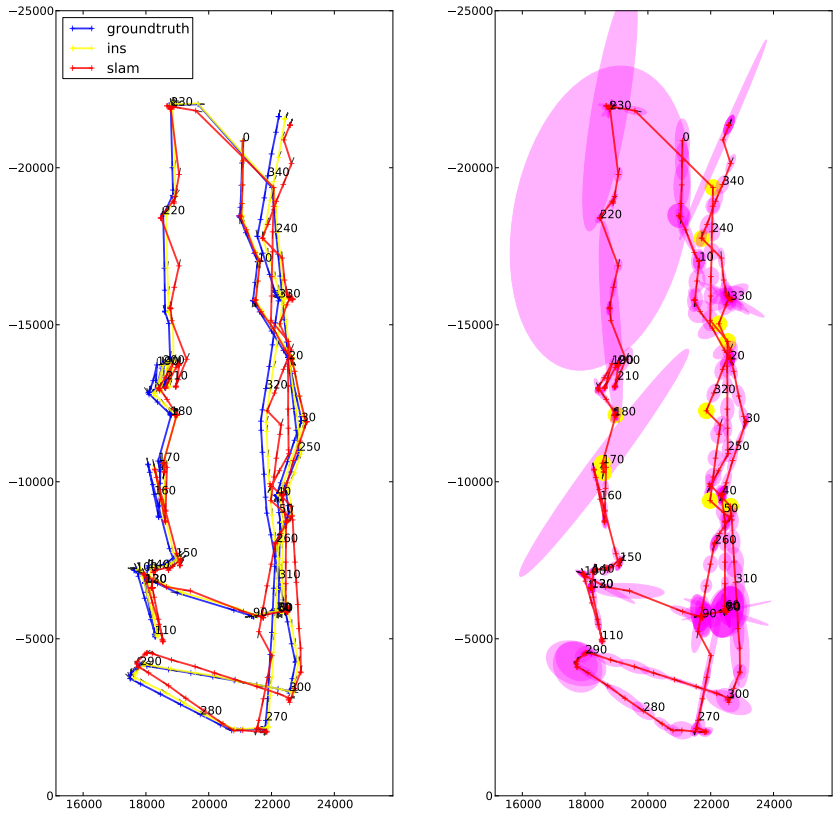
Chapter 6

Future work

Use the quad-tree based scanmatcher with a guessed initial position over the entire map.

Chapter 7

Appendix: images



(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 7.1: Map 1

Bibliography

[1] 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.

[2] A. Ethembabaoglu, *Active target tracking using a mobile robot in the USARSim*, Ph.D. thesis, Bachelors thesis, Universiteit van Amsterdam, 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] 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.

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

[7] 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.

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

[9] M. Jankowska, *A hough transform based approach to map stitching*, Ph.D. thesis, University of Oxford, 2009.

[10] 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.

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