2D Multi-Resolution Correlative Scan-Matching using a Polygon-Based Similarity Measurement

Philipp Vath **Benjamin Ummenhofer**
Department of Computer Science, University of Freiburg

Abstract

In order to reliably explore an environment and locate itself at the same time, an autonomous mobile robot has to accurately solve the Simultaneous Localization and Mapping (SLAM) task. Finding the alignment of 2D lidar scans can be regarded as a sub problem of this task. This paper presents a 2D scan matching approach solving this problem by computing an optimal rigid-body transformation between two 2D lidar scans.

Our approach exhaustively searches at multiple resolutions for the best transformation and adds a Gaussian distribution around each measurement to model its uncertainty. As a novel contribution, we enhance accuracy as we take into account the unoccupied space between the robot and the scan end points. We illustrate the quality of our implementation in accurate maps of real buildings with respect to ground-truth.

1 Introduction

There are two main goals for autonomous mobile robots navigating in unknown environments or known surroundings: On the one hand, a robot is expected to build an accurate map of the environment and to reliably update information about known surroundings respectively. On the other hand, it has to precisely determine its own position and orientation, i.e. its pose in that environment in order to navigate through it. The precision of the solution of SLAM or navigation tasks depend on the accuracy of the hardware used as well as on the software processing the measurements. 2D scan matching uses the 2D lidar range scans to correct the odometry data of a robot. The odometry is much less precise than the results of scan matching. This holds especially for static environments. Additionally, odometry measurements are susceptible to the risk of accumulating errors and they are prone to unpredictable wheel slippage. That is why 2D scan matching plays an important role in robot mapping, localization and navigation tasks.

Given two odometry poses of a robot and the two 2D lidar scans that were measured at these poses, the task of scan matching is to compute the corrected robot pose. When talking about one scan , we mean a set of range values. For instance a scan consists of range values that represent the sampled horizontal cross-section of the environment. To correct the robot pose, we want to find the optimal rigid-body transformation between a new uncorrected scan and an earlier corrected scan or a corrected map. To achieve this goal and to produce a high quality solution, it is important, define what transformations the approach takes into account. Another important point is how the algorithm rates the similarity between two aligned scans. We present a technique that considers all transformations according to a certain predefined resolution. Thus, we perform a complete search in a discretized parameter space. In order to compute the quality of a match between two scans, we introduce the following concept: We take into account the endpoints and the scanned unoccupied area provided by a new scan and compare it with *every relevant* information of earlier corrected scans accumulated in what we call a grid map. More details on how we compute the score of a match between two scans can be found in section 3.3. We present the grid map concept that we have implemented in section 3.4.

In addition, we model a measurement's uncertainty by adding a Gaussian distribution centered at its location. The sum of corrected scans allows us to compute an accurate occupancy map of the whole observed environment and to visualize a corrected robot path within it. Our work puts a focus on the enhancement of the quality of the scan matching solution: We introduce the concept of a similarity measurement based on the polygon described by the endpoints of the scan and the robot position. The polygon represents the total area resulting from the entire observation a robot gains during a single new scan. We visualize our scan polygon concept in Figure 1.In contrast, approaches that investigate only into the scan endpoints provide less quality as they omit the unoccupied space between the scan endpoints and the robot. We demonstrate this in an experiment comparing the resulting occupancy maps of the different approaches. As an additional feature, our approach is flexible, as it allows the area in which the robot is located to be theoretically unbounded. This feature is realized by our grid map concept described in 3.4.

2 Related Work

Various techniques have been proposed for matching range scans. One of these techniques is the Iterative Closest Point

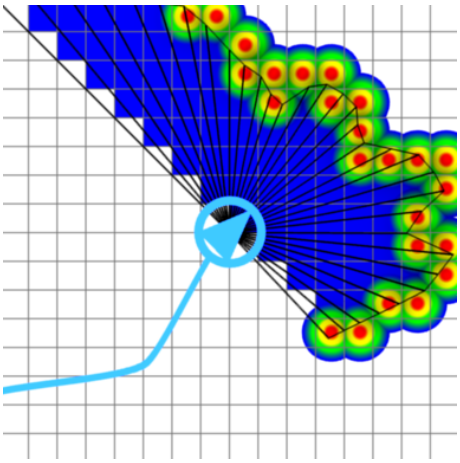


Figure 1: The **scan polygon** is shaped by the sum of the range scans *(black)*. Scan endpoints are rasterized to occupied positions *(cells around the red peaks)*. The unoccupied space between them and the robot is rasterized as "unoccupied cells" *(blue)*. The values of the Gaussian distributions around the scan endpoints are visualized ranging from *blue* to *red*. Every other position defines unknown space *(white)*.

(ICP) algorithm. The algorithm computes a rigid body transform that aligns two sets of corresponding points. For the computation of the transform itself a closed form solution can be used (see [Horn, 1987] or [Umeyama, 1991]), while the selection of corresponding points is an iterative process.

An overview of some variants is given in [Rusinkiewicz and Levoy, 2001]. Application of ICP-based algorithms is shown in [Lu and Milios, 1997] and [Besl and Mckay, 1992].

Other techniques exist that use polar coordinates [Diosi and Kleeman, 2007], or employ the Hough transform [Censi *et al.*, 2005].

In contrast to these algorithms, [Olson, 2009] and [Konolige and Chou, 1999] have presented correlative approaches. These approaches operate on a grid map. They compute a similarity measurement for all possible alignments of a scan in the grid map. To reduce the search-window of possible alignments, prior information is used. The implementation of Olson uses position estimations from the robot's odometry. Further, it uses a multi-resolution approach that aligns the scans on a coarse grid and refines the alignment on a higher resolution grid map. This makes the algorithm fast enough for matching scans in real-time.

Our work is quite similar is based on ideas of [Olson, 2009]. Like Olson we use a multi-resolution approach for better performance, but we primarily focus on accuracy and quality. Our approach uses as much information as possible for aligning scans. A grid map allows the combination of information of multiple scans. Data captured and corrected from earlier scans is kept as long as it is relevant in our grid map and is only removed at the moment when it is considered to be irrelevant. Intuitively speaking, this occurs whenever the robot is too far away from this data. Our main contribution is a score function that does not only consider the endpoints of a scan, but the covered unoccupied area as well.

3 Approach

3.1 Probabilistic formulation

To explain the probabilistically-motivated foundations of our approach, we illustrate the scan matching problem in a theoretical way. The probability of the robot pose x_i is

$$p(x_i|x_{i-1}, u, m, z). \tag{1}$$

The distribution depends on the previous pose x_{i-1}, the motion u, the map m and the observation z. The pose is a triplet (x, y, θ), where x and y are the Cartesian coordinates and θ is the heading of the robot. The motion describes how the pose of the robot changes. The map m in our implementation is represented by a grid map storing the occupancy of the environment. The observation z is a list of range readings.

We apply Bayes' rule on (1) and get

$$p(x_i|x_{i-1}, u, m, z) \sim p(z|x_i, x_{i-1}, u, m)p(x_i|x_{i-1}, u, m). \tag{2}$$

According to the graphical model in Figure 2, we can simplify the equation to

$$p(x_i|x_{i-1}, u, m, z) \sim p(z|x_i, m)\ p(x_i|x_{i-1}, u). \tag{3}$$

The main goal is to find the a-posteriori probability distribution over the robot's pose x_i, i.e. $p(x_i|x_{i-1}, u, m, z)$. To be more precise, in our context, it is sufficient and most interesting respectively, to know the maximum likelihood.

The probability distribution $p(x_i|x_{i-1}, u)$ represents the motion model and $p(z|x_i, m)$ represents the observation model. To entirely account for the above, we consider the following uncertainties:

1. *odometry uncertainty* (coarse):
   occurs when the robot drives, usually a banana shaped distribution.
2. *measurement uncertainty* (fine):
   resulting from the 2D lidar scan endpoints, probability is dominated by the distance of the endpoints to obstacles.

The observation model $p(z|x_i, m)$ can be obtained from aligning laser range scans against the map and computing a

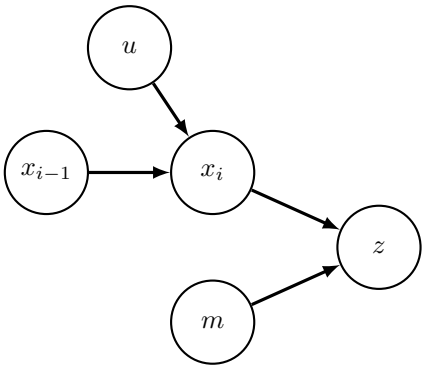


Figure 2: A graphical model showing the dependencies of the variables. The current position of the robot x_i depends on the motion u and the previous position x_{i-1}. The observation z depends on the current robot position and on the map m.

score function. The uncertainty of each endpoint in range and bearing is modelled with a symmetric Gaussian distribution. This allows us to model the uncertainty by convolving the scan endpoints with a Gauss filter.

The motion model $p(x_i|x_{i-1}, u)$ can be modeled by a multivariate Gaussian distribution.

3.2 Multi-resolution approach

This section gives an overview of our method and describes how it is accelerated to be practically applicable for real-time usage. The goal is to compute the pose of the robot x_i that maximizes $p(z|x_i, m)$. A naive approach would compute a score for every x_i inside the search window. The search window is a 3D parameter space representing the possible transformations to align the scan to the grid map.

As section 3.3 details, calculating the score is computationally expensive because it involves the following steps: At first, a scan polygon (see Figure 1) has to be rasterized. Then, each position (pixel) of its rasterized area has to be converted to an index (pixel) of the grid map. Afterwards, for each pixel of the scan polygon, the score can be computed according to section 3.3. The accuracy increases with an increasing scan polygon area, because a larger part of the environment is taken into account. Unfortunately, the runtime increases as well. Like Olson [2009], to reduce runtime, we implement a multi-resolution level correlation based method. Instead of searching for the global optimum in a high resolution parameter space, we search the optimum first on a coarse and thus smaller parameter space. Additionally, we use a faster score function that considers only the endpoints for computing the scores on the coarse parameter space. In the following, we call the high resolution parameter space fine grid and the low resolution parameter space coarse grid. Only the subspace at the optimum of the coarse grid is further investigated afterwards, on the fine grid. We use two levels: Our coarse grid has a resolution corresponding to the one of our coarse grid map: We use $\Delta x = \Delta y = 0.5$ meter per pixel and $\Delta\theta = 1$ degree for the angle step. Our fine grid corresponds to our fine grid map and thus uses a resolution of $\Delta x = 0.05$ meter per pixel and $\Delta\theta = 0.1$ degree for the angle step.

To start with, we add the first scan to the coarse and fine grid map. Then we align the following scans into the grid maps, if the robot has moved at least 0.05 meter per pixel or turned one degree. Finally, we store the corrected scan and integrate it into the coarse and fine grid map.

3.3 Compute score

In order to obtain a measure for the quality for an alignment of a transformed scan with the grid map, we evaluate a score function. The score function takes the map m, the scan z and a rigid body transformation T as parameters. The map is a binary matrix. For occupied cells the map stores a '1'. All other cells are encoded with the value '0'. The scan z is a list of endpoints of the form (x, y). The transformation T is a translation and rotation. When the transformation is applied on the scan, all endpoints are mapped to integer indices by using the nearest neighbour interpolation.

We use two different score functions in our implementation. The first score function uses solely the endpoints of the scan. We use this function for the scan matching on the coarse grid. It is defined as

$$S_1(m, T, z) = \sum_{(x,y)\in T(z)} \langle m_{x,y}, G\rangle \,. \tag{4}$$

The $\langle\cdot,\cdot\rangle$ denotes the dot product. The matrix $m_{x,y}$ is an extract of the map around $m(x, y)$, for example

$$m_{x,y} = \begin{bmatrix} m(x-1,y-1) & m(x,y-1) & m(x+1,y-1) \\ m(x-1,y) & m(x,y) & m(x+1,y) \\ m(x-1,y+1) & m(x,y+1) & m(x+1,y+1) \end{bmatrix}$$

The matrix G is a Gauss filter matrix of the same size as $m_{x,y}$:

$$G = \begin{bmatrix} 0.075 & 0.124 & 0.075 \\ 0.124 & 0.204 & 0.124 \\ 0.075 & 0.124 & 0.075 \end{bmatrix}$$

The filter matrix G represents the uncertainty of the scan endpoint position. We simply iterate over the transformed endpoints and sums up the dot products of the endpoints with the map.

Our second score function tries to consider the endpoints and the scanned area. Therefore, we construct a polygon from the scan endpoints and the robot position. The polygon covers the scanned area, this is illustrated as blue cells in Figure 1. To find these cells, we use a simple scan-line polygon filling algorithm. The contour of the polygon is computed using Bresenham's fast line algorithm. A detailed description of these algorithms can be read in [Hearn and Baker, 1996] and [Bresenham, 1965]. The score function is used on the fine grid map and is defined as

$$\begin{aligned} S_2(m, T, z) = {} & S_1(m, T, z) \\ & - \sum_{(x,y)\in T(P(z))} M(m(x, y), T(P(z))(x, y)). \end{aligned} \tag{5}$$

The list of cells belonging to the polygon is denoted as $P(z)$. The function M returns the value of $m(x, y)$ if the point $T(P(z))(x, y)$ is not a scan endpoint. If the point is an endpoint M returns zero.

The general idea in our score computation using M is that matching occupied space contributes positively to the score whereas truly mismatching unoccupied space contributes negatively to it. To be more precise, we treat the different cases as follows:

1. If and only if $T(P(z))(x, y)$ is not an endpoint and $m(x, y) = 1$, the score is decreased by $m(x, y)$, because: In that case, we assume from the current scan's observation, that the cell is unoccupied. However, the cell in the grid map is occupied.
2. If $T(P(z))(x, y)$ is not an endpoint and $m(x, y) = 0$, we do not increase the score, because: On the one hand, both the scan's cell and the grid map cell could be unoccupied. On the other hand, it could be that there is unknown space in the grid map (that overlaps with unoccupied space from the current scan). Thus, we cannot guarantee a perfect match here.

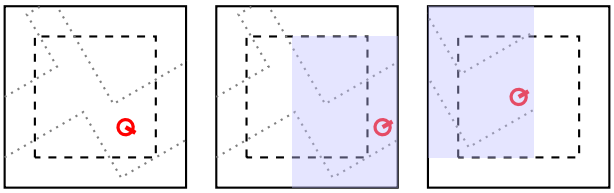


Figure 3: The dashed rectangle is the threshold area of the grid map. When the robot leaves the threshold area, the map is relocated. We create a new map centered at the robot position. The overlap between the old and the new map is copied from the old to the new map.

3. If $T(P(z))(x, y)$ is an endpoint and $m(x, y) = 0$, we also do not increase the score, because: On the one hand it could be a mismatch. On the other hand, it could be that there is unknown space in the grid map, that overlaps with occupied space from the current scan. Thus, we do not punish for a mismatch here.
4. If $T(P(z))(x, y)$ is an endpoint and $m(x, y) = 1$, the score is increased using S_1.

As a consequence, we could improve our score computation by explicitly modeling and storing unknown space, as discussed in section 6.

3.4 Grid map

The grid map is a 2D array storing occupancy information of the environment. When the map is known in advance, we can easily compute the space we need to allocate. However, when the map is not known or the environment is too big to be stored, we can store only a part of the map.

Our grid map is a square with an edge length of 64m. We maintain two grid maps at different resolutions. A coarse grid map with 50cm per pixel and a fine grid map with 5cm per pixel.

Initially, the grid maps are centered at the robot position. When the robot leaves a threshold area, we relocate the grid maps such that they are again centered at the robot position. This process is illustrated in Figure 3 The important part is that we try to keep parts of the already incorporated data. Therefore, we compute the overlap of the newly centered grid map with the grid map at its previous position. The data in the overlap is preserved and copied to the new map. This relocation is an expensive step for large grid maps.

To keep the number of required relocations low, the size of the map and the threshold area must be chosen carefully. A too small threshold area can cause frequent relocations. If the threshold area is as big as the grid map, it is possible that most of the scan endpoints lie outside the grid map. This corrupts the reliability of the scan matching process. A bigger grid map can alleviate these problems. However, a huge grid map requires a lot of memory, which leads to frequent CPU cache misses.

A threshold area with an edge length of 14m, has proven to be a good choice for our grid map size.

4 Improving Performance

Instead of evaluating the whole parameter space on the fine grid, we can use a hill climbing approach. This is especially useful as the computation of our score function considering the unoccupied space is very expensive. The goal is to provide a fast approximate solution for the observation model $p(z|x_i, m)$.

The search is started at the solution obtained from the coarse parameter grid. The algorithm then searches the local neighbourhood of the current pose in the parameter space: If we find a pose at $(x \pm \Delta x, y \pm \Delta y, \theta \pm \Delta\theta)$ such that the score function returns a better score, we further search in that direction of the parameter space, else we halve the step size Δ. We do so until we reach a satisfying step size threshold of $\Delta_{\min} = 5\text{cm}$. At that moment, we adopt the pose that corresponds to the best score found so far.

A disadvantage of this algorithm is that it is not guaranteed to find the global maximum, as it only searches for local maxima. Thus obviously, the quality of its results is far below the quality we have reached by searching in the whole parameter space as described in section 3.2. In contrast, because hill climbing has less to compute, its performance is much better. As we have concentrated our work on the enhancement of quality, we spare detailed runtime tests for hill climbing as future work.

5 Results

5.1 Quality

We have introduced two score functions in section 3.3. In our multi-resolution algorithm we use the endpoint based measure on the coarse grid map. The evaluation is fast and accurate enough for scan matching on the coarse map.

The resolution and the score function on the fine grid map determine the quality of the match. A resolution of $5cm$ yields good results on the Kaikan data set.

In order to compare the two score functions we use a set of corrected reference scans for comparison. The error for the travel distance between the reference scan and the scans obtained from our scan matcher is computed as

$$\text{err}_{\text{dist.}} = \frac{|\Delta t_{\text{ref}} - \Delta t|}{\Delta t_{\text{ref}}}. \tag{6}$$

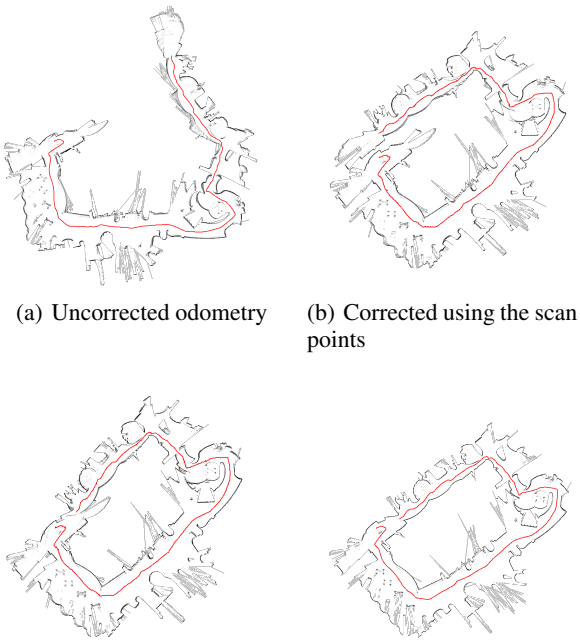
Where Δt is the Euclidean distance between two sequential robot poses. The distance of the corresponding reference scans is Δt_{ref}. To account for different travel distances, the error term is normalized using Δt_{ref}.

The error for rotation is similarly defined as

$$\text{err}_{\text{rot.}} = \frac{|\Delta\theta_{\text{ref}} - \Delta\theta|}{|\Delta\theta_{\text{ref}}|}, \tag{7}$$

where $\Delta\theta$ describes how the robot has turned between two sequential scans. Note that $\Delta\theta$ can be negative. An error for translation or rotation can only be computed when the robot has moved or turned.

We have computed these error terms for both methods on the Kaikan data set. From Table 1 and Figure 4 it can be seen that the polygon method is more accurate.



(a) Uncorrected odometry

(b) Corrected using the scan endpoints

(c) Corrected using the endpoints and unoccupied space

(d) Reference log file

Figure 4: Comparison of the different score functions on the Kaikan Museum data set. The red line shows the path of the robot. The search window for (b) and (c) was 2.5m for translation and $5°$ for rotation. The score function considering the unoccupied space yields the best results.

	Endpoints	Polygon
Mean $\text{err}_{\text{dist.}}$	0.214	0.180
Std. deviation	0.397	0.282
Mean $\text{err}_{\text{rot.}}$	0.005	0.005
Std deviation	0.004	0.004

Table 1: Comparison of the score functions used to align scans against the fine grid map. The values describe the difference in translation and rotation with respect to a corrected reference log file for the Kaikan Museum.

5.2 Performance

The time required to align a scan against the grid map is important for real time applications. The speed of the algorithm heavily depends on the score function.

We test the influence of the different methods on the processing time. As before, we stick to the endpoint based method for the coarse grid map but change the method for the fine grid map. The search window for the coarse grid map is 2.5m for translation and $5°$ for rotation. The step size is 50cm which corresponds to the resolution of the map. The step size for the rotation is one degree. On the fine grid map we have a step size of 5cm which corresponds to the map resolution. The rotation step size is $0.1°$.

	Endpoints	Polygon
Avg. time	41ms	364ms
Best time	24ms	64ms
Worst time	345ms	1048ms
Std. deviation	0.044ms	2.329ms

Table 2: Times to match a single lidar scan against the grid map on the Kaikan data set. The used search-window is 2.5m for translation and $5°$ for rotation.

The test has been performed on the Kaikan data set on a Core2 Duo at 1.8 GHz. Table 2 shows the results. The times include all stages of the algorithm. Including the relocation of the grid maps when the robot leaves the threshold area.

The polygon-based score function is about eight times slower than the method using only the endpoints of a scan. Hence, for real time applications the method considering only endpoints is more suited.

5.3 Environment

We have run our scan matcher on various data sets. Table 3 gives an overview. The data sets contain environments with wide open spaces like the Kaikan Museum and narrow corridors like the Intel Laboratory.

The search-window was 2.5m for translation and $5°$ for rotation. The resolution of the coarse/fine grid maps is $50\text{cm}/5\text{cm}$ per pixel. On the Intel data set we use a 25cm per pixel resolution on the coarse grid map. The higher resolution gives better results on that data set.

The results of the scan matcher are compared to corrected reference data sets using the terms defined in equation (6) and (7). The values are given in Table 3. They show that the scan matcher works better on data sets where the scanned area is large. On uncluttered open environments, a single scan contains a larger part of the environment. Thus, covering the environment needs less scans or to put it another way, scans of the environment contain more common data.

6 Future Work

As already mentioned, the score computation on the fine grid using the scan polygon is expensive. We have implemented a hill climbing approach to speed up our algorithm. The speed up is achieved by evaluating only a small subset of the parameter space. Hence, the algorithm cannot guarantee to find the global optimum.

Therefore, we prefer to improve the performance of the score function itself. We consider implementing the whole correlation and rasterization steps on graphics hardware. Current graphics hardware provides efficient implementations for these operations.

The grid map in our implementation does not differentiate between unoccupied space and unknown space. Hence, our score computation methods cannot account for this difference. Considering the difference between unoccupied and unknown space, would give us more possibilities for the design of score functions.

	Kaikan	Acapulco	Intel Lab	Freiburg 101	Freiburg 079
Mean distance error	0.180	0.108	0.226	0.103	0.197
Distance error σ	0.080	0.031	0.081	0.044	0.091
Mean rotation error	0.005	0.005	0.006	0.004	0.006
Rotation error σ	0.004	0.004	0.005	0.004	0.005
Average scanned area	81.04 m^2	131.14 m^2	15.06 m^2	84.71 m^2	12.88 m^2

Table 3: Accuracy of the implemented scan matcher compared to corrected reference log files. The best results are obtained on data sets where the scanned area is large.

7 Conclusion

We have presented a multi-resolution scan matching algorithm. The algorithm is fast enough for correcting odometry in real-time.

Further, two different scoring functions have been shown that can be used to determine the probability of a match. The two scoring functions have been evaluated with respect to quality and speed. Our scoring function using polygon rasterization for considering unoccupied space gives better results. The method using only the endpoints of the scans is faster and more convenient for real-time usage. However, both methods are rough approximations that cannot meet the beam based model of laser scans.

We have tested our algorithm in the context of odometry correction. In theory the implementation is suited for localization problems as well, but this has not yet been tested.

8 Acknowledgements

We like to thank the following people for providing us with their data sets: Giorgio Grisetti (Kaikan Museum), Dieter Fox (Intel lab), Cyrill Stachniss (Freiburg 101 and Freiburg 079) and Nick Roy (Acapulco Convention Center). All data sets except for the Kaikan Museum were obtained from the Robotics Data Set Repository (Radish) [Howard and Roy, 2003].

References

[Besl and Mckay, 1992] P. J. Besl and H. D. Mckay. A method for registration of 3-d shapes. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 14(2):239–256, 1992.

[Bresenham, 1965] J. E. Bresenham. Algorithm for computer control of a digital plotter. *IBM Systems Journal*, 1965.

[Censi *et al.*, 2005] Andrea Censi, Luca Iocchi, and Giorgio Grisetti. Scan matching in the hough domain. In *Robotics and Automation, 2005. ICRA '05. IEEE International Conference on*, pages 2739–2744, Barcelona, Spain, 2005.

[Diosi and Kleeman, 2007] Albert Diosi and Lindsay Kleeman. Fast laser scan matching using polar coordinates. *The International Journal of Robotics Research*, 26(10):1125–1153, October 2007.

[Hähnel *et al.*, 2003] D. Hähnel, W. Burgard, D. Fox, and S. Thrun. A highly efficient FastSLAM algorithm for generating cyclic maps of large-scale environments from raw laser range measurements. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2003.

[Hearn and Baker, 1996] Donald Hearn and M. Pauline Baker. *Computer Graphics, C Version (2nd Edition)*. Prentice Hall, 2 sub edition, May 1996.

[Horn, 1987] B. K. P. Horn. Closed-form solution of absolute orientation using unit quaternions. *Journal of the Optical Society of America A: Optics, Image Science, and Vision, Volume 4, Issue 4, April 1987, pp.629-642*, 4:629–642, April 1987.

[Howard and Roy, 2003] Andrew Howard and Nicholas Roy. The robotics data set repository (radish), 2003.

[Konolige and Chou, 1999] Kurt Konolige and Ken Chou. Markov localization using correlation. In *IJCAI '99: Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence*, pages 1154–1159, San Francisco, CA, USA, 1999. Morgan Kaufmann Publishers Inc.

[Lu and Milios, 1997] Feng Lu and Evangelos Milios. Robot pose estimation in unknown environments by matching 2d range scans. *J. Intell. Robotics Syst.*, 18(3):249–275, 1997.

[Olson, 2009] E. B. Olson. Real-time correlative scan matching. In *Robotics and Automation, 2009. ICRA '09. IEEE International Conference on*, pages 4387–4393, May 2009.

[Rusinkiewicz and Levoy, 2001] S. Rusinkiewicz and M. Levoy. Efficient variants of the icp algorithm. In *3-D Digital Imaging and Modeling, 2001. Proceedings. Third International Conference on*, pages 145–152, 2001.

[Umeyama, 1991] Shinji Umeyama. Least-squares estimation of transformation parameters between two point patterns. *IEEE Trans. Pattern Anal. Mach. Intell.*, 13(4):376–380, 1991.