The Normal Distributions Transform: A New Approach to Laser Scan Matching

Peter Biber*
University of Tübingen,WSI/GRIS

WSI-Report 2003-3
ISSN 0946-3852

*biber@gris.uni-tuebingen.de

Abstract

Matching of geometric primitives is an important problem in computer vision. Many practical applications require the solution of this problem: Object tracking, object recognition and reconstruction are well known examples. Matching of 2D range scans is another instance of this problem. Here the goal is to recover the geometric mapping, that aligns two scans optimally. Frequently, mobile robots are equipped with a laser scanner delivering such range scans and so this task is an essential part of the robotic mapping problem, that is the autonomous building of maps using only sensor data.

Since 1992, the ICP algorithm is considered widely to be the answer to this problem. Also in the field of robot mapping, a variant of this algorithm is among the most popular ones. We present a new approach to 2D range scan matching, that is significantly different from ICP. The key idea of our proposed algorithm is a new representation for a set of points: The *Normal Distributions Transform (NDT)*. This transformations models the probability of locally measuring a point at a certain position by a collection of normal distributions. Given a second set of points and a geometric mapping, a measure is defined by mapping all points according to the geometric mapping, evaluating the corresponding local normal distribution and summing the result. Optimizing the geometric parameters is then defined as maximizing this sum. We will apply the method to the problem of laser scan matching, position tracking and the simultaneous localization and mapping (SLAM) problem. First result show, that our approach is able to map middle-size indoor environments with high accuracy and in real-time.

1 Introduction

Modern mobile robots are often equipped with a laser scanner. Such laser scanners measure the distance between the robot and objects in the environment with a high angular resolution of at least one degree, covering typically 180 degrees or 360 degrees. Laser scanners are also able to measure with a high temporal resolution of at least 20 scans per second. The measured distances can be used to reconstruct the 2D positions of objects in the robot's environment, represented by a set of points. If one is able to find the optimal geometric mapping between two set of points, that were recorded by subsequent scans, the movement between the two scans is recovered. Under the assumption, that the robot is moving on a plane, this mapping is given by a 2D-translation and a 2D-rotation. Finding this optimal mapping is an instance of the geometric matching problem. For a mobile robot, the solution of this problem is essential for navigation and map building. A good solution should have two properties:

- The solution should have a high accuracy (since everybody wants of course accurate maps).
- The algorithm should be fast (to make use of the high temporal resolution).

In this report we present a method for the matching of two sets of 2D points. We will apply the method to the problem of laser scan matching, position tracking and the simultaneous localization and mapping (SLAM) problem. But we are sure, that the method can also be applied to similar problems, namely the matching of sets of 3D points and contour matching. The method we propose is significantly different from the *Iterated Closest Point (ICP)* algorithm [2], which is widely considered to be the standard algorithm in this field. We claim, that the proposed method is both accurate and fast. Experimental results indicate, that this claim is correct.

We introduce a new representation for a set of points, which we call: *The Normal Distributions Transform (NDT)*. This transformation locally models the probability of measuring a point at a certain position by a collection of normal distributions. Given a second set of points and a geometric mapping, a measure is defined by mapping all points according to the geometric mapping, evaluating the corresponding local normal distribution and summing the result. Optimizing the geometric parameters is then defined as maximizing this sum. We will apply the method to the problem of laser scan matching, position tracking and the simultaneous localization and mapping (SLAM) problem.

After specifying the scope of this work, the report starts with descriptions of previous work for scan matching. We will then present the Normal Distributions Transform and how it can be used for geometric matching. We will apply the algorithm to the problem of building maps with a mobile robot. The report concludes with the experimental results on three different data sets. Two of these sets are from environments containing cycles.

2 The scope of this work

This work is about matching two 2D range scans with respect to each other. It is also about matching several range scans with respect to each other, using only the results of the pairwise matching in a global optimization step. We will apply this algorithm to position tracking (that is estimating the ego-motion of a mobile robot) and to the simultaneous localization and mapping problem (SLAM). *Both problems are treated as pure instances of a geometric matching problem.*

We do not have a model for the motion of the robot nor do we represent the uncertainty of the robot position. At each time step, there is a unique estimate of the robot position and of the map. It is future work to combine our approach with a Bayesian framework, as it is state of the art [16].

Nevertheless, our experiments show, that we can reliably map middle size environments. Although there are no special mechanisms for closing cycles in a map, our results are accurate enough, that cycles can be closed by the global optimization procedure. For our approach to work, it is required that the environment has enough structure. Our approach will fail in long featureless corridors, where the matching problem has no unique solution. This is similar to the aperture problem in computer vision.

3 Previous work

The goal of matching two range scans is to find the relative pose between the two positions, where the scans were taken. The basis of the most successful algorithms is the establishment of correspondences between primitives of the two scans. Out of this, an error measure can be derived and minimized. Cox used points as primitives and matched them to lines, that were given in an a priori model [4]. In the Amos Project [8], these lines were extracted from the scans. Gutmann matched lines extracted from a scan to lines in a model [9]. The most general approach, matching points to points, was introduced by Lu and Milios [11]. This is essentially a variant of the ICP (*Iterated Closest Point*) algorithm ([2],[3],[20]) applied to laser scan matching. We share with Lu and Milios our mapping strategy. As in [12], we do not build an explicit map, but use a collection of selected scans with their recovered poses as an implicit map.

In all of these approaches, explicit correspondences have to be established. Our approach differs in this point, we never need to establish a correspondence between primitives. Other approaches, that avoid solving the correspondence problem are histogram based. Weiss and Puttkammer [19] used angular histograms to recover the rotation between two poses. Then x and y-histograms, that were calculated after finding the most common direction were used to recover the translation. This approach can be extended by using a second main direction [8].

Scott [13] compared three mapping algorithms using laser scans, among them the Lu and Milios algorithm.

Our work was also inspired by computer vision techniques. If the term "probability density" is replaced by "image intensity", our approach shares a similar structure to feature tracking [14] or composing panoramas [15]. These techniques use the image gradient at each relevant position to estimate the parameters. Here, derivatives of normal distributions are used. Opposed to image gradients, these can be calculated analytically.

4 The Normal Distributions Transform

This section describes the Normal Distributions Transform (NDT) of a single laser scan. This is meant to be the central contribution of the paper. The use of the NDT for position tracking and SLAM, described in the following sections, is then relatively straightforward.

The NDT models the distribution of all reconstructed 2D-points of one laser scan by a collection of local normal distributions. First, the 2D space around the robot is subdivided regularly into cells with constant size. Then for each cell, that contains at least three points, the following is done:

1. Collect all 2D-Points $\mathbf{x}_{i=1..n}$ contained in this box.
2. Calculate the mean $\mathbf{q} = \frac{1}{n}\sum_i \mathbf{x}_i$.
3. Calculate the covariance matrix $\Sigma = \frac{1}{n-1}\sum_i(\mathbf{x_i} - \mathbf{q})(\mathbf{x_i} - \mathbf{q})^t$.

The probability of measuring a sample at 2D-point $\mathbf{x}$ contained in this cell is now modeled by the normal distribution $N(q, \Sigma)$:

$$p(\mathbf{x}) = C\exp(-\frac{(\mathbf{x}-\mathbf{q})^t\Sigma^{-1}(\mathbf{x}-\mathbf{q})}{2}), \tag{1}$$

with C being a constant, that we set to one.

Similar to an occupancy grid, the NDT establishes a regular subdivision of the plane. But where the occupancy grid represents the probability of a cell being occupied, the NDT represents the probability of measuring a sample for each position within the cell. We use a cell size of 100 cm by 100 cm.

What's the use for this representation? We now have a piecewise continuous and differentiable description of the 2D plane! Before we show an example, we have to note two implementation details.

To minimize effects of discretization, we decided to use four overlapping grids. That is, one grid with side length l of a single cell is placed first, then a second one, shifted by $\frac{l}{2}$ horizontally, a third one, shifted by $\frac{l}{2}$ vertically and finally a fourth one, shifted by $\frac{l}{2}$ horizontally and vertically. Now each 2D point falls into four cells. This will not be taken into account for the rest of the paper explicitly and we will describe our algorithm, as if there were only one cell per point. So if the probability density of a point is calculated, it is done with the tacit understanding, that the densities of all four cells are evaluated and the result is summed up.

A second issue is, that for a noise free measured world line, the covariance matrix will get singular and cannot be inverted. In practice, the covariance matrix can sometimes get near singular. To prevent this effect, we check, whether the smaller eigenvalue of Σ is at least 0.001 times the larger eigenvalue. If not, it is set to this value. This also has the effect of smoothing lines a little bit.

Figure 1 contains some example transformations of single cells. Shown are the 2D points and the probability density. The visualization is created by evaluating the probability density at each point, bright areas indicate high probability densities. Figure 2 show some example transformations of whole laser scans. The next section shows, how this transformation is used to align two laser scans.

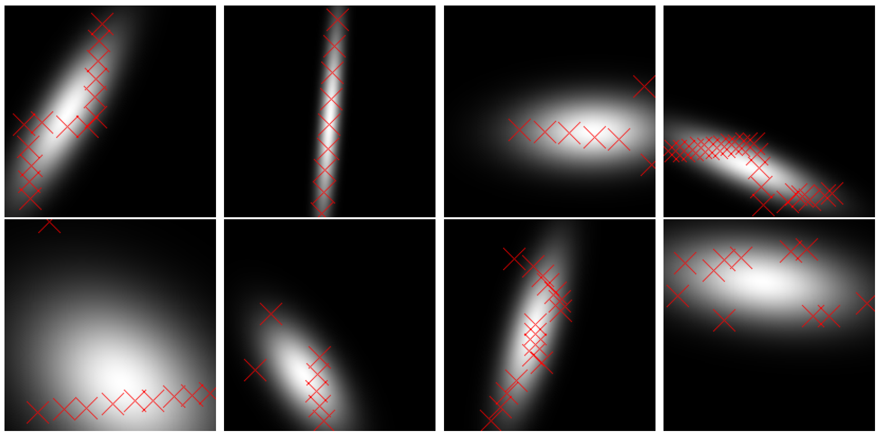
Figure 1: The NDTs of some single cells.

5 Scan alignment

The spatial mapping T between two robot coordinate frames is given by

$$T : \begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} \cos\phi & -\sin\phi \\ \sin\phi & \cos\phi \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} t_x \\ t_y \end{pmatrix}, \tag{2}$$

where $(t_x, t_y)^t$ describes the translation and ϕ the rotation between the two frames. The goal of the scan alignment is to recover these parameters using the laser scans taken at two positions. The outline of the proposed approach, given two scans (the *first* one and the *second* one), is as follows:

1. Build the Normal Distribution Transform of the first scan.
2. Initialize the estimate for the parameters (by zero or by using odometry data).

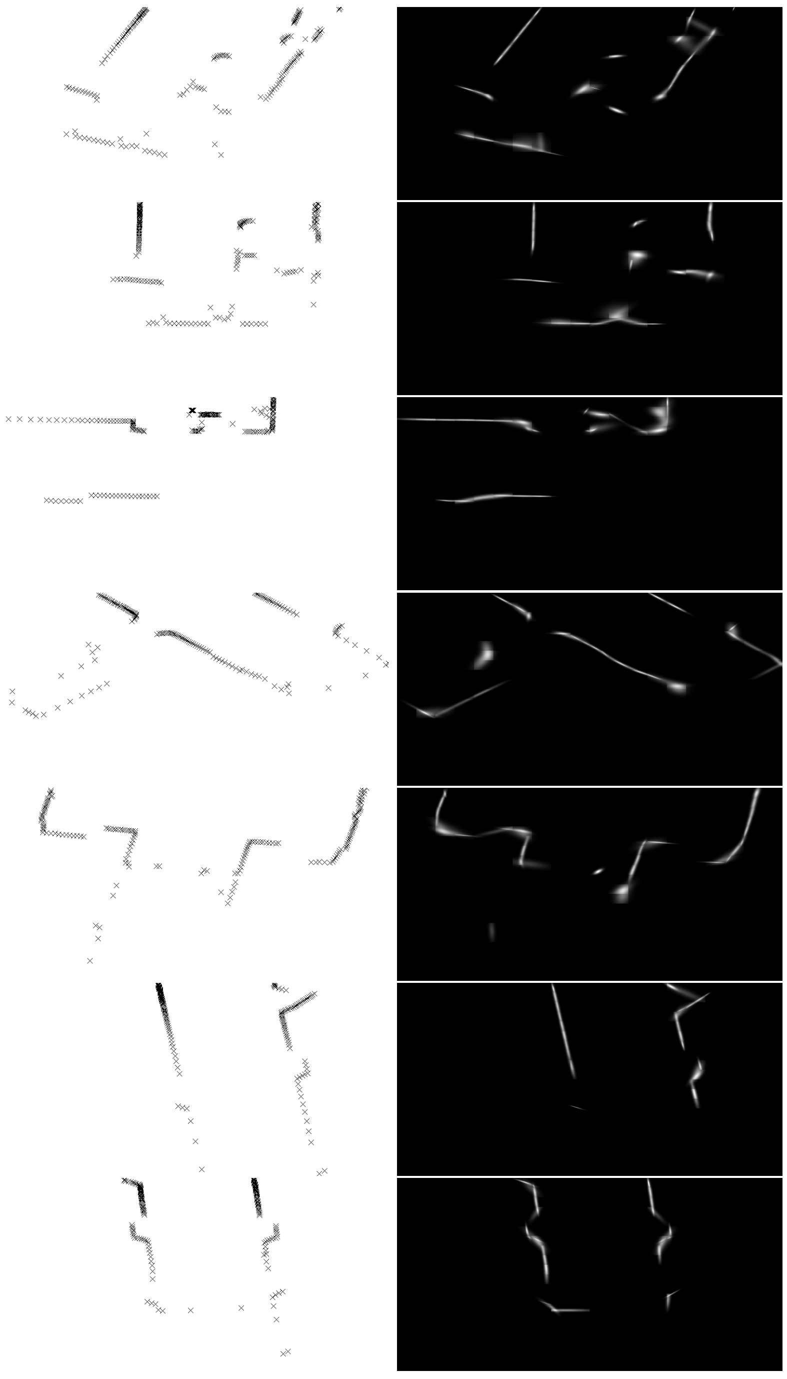

Figure 2: Some NDTs of laserscans. On the left sides are the original laser-scans, on the right side the respective transformation. Bright areas indicate high probabilities.

3. For each sample of the second scan: Map the reconstructed 2D point into the coordinate frame of the first scan according to the parameters.

4. Determine the corresponding normal distributions for each mapped point.

5. The score for the parameters is determined by evaluating the distribution for each mapped point and summing the result.

6. Calculate a new parameter estimate by trying to optimize the score. This is done by performing one step of Newton's Algorithm.

7. Goto 3 until a convergence criterion is met.

The first four steps are straightforward: Building the NDT was described in the last section. Odometry data could be used to initialize the estimate. Mapping the second scan is done using T and finding the corresponding normal distribution is a simple lookup in the grid of the NDT.

The rest is now described in detail using the following notation:

- $\mathbf{p} = (p_i)^t_{i=1..3} = (t_x, t_y, \phi)^t$: The vector of the parameters to estimate.
- $\mathbf{x_i}$: The reconstructed 2D point of laser scan sample i of the second scan in the coordinate frame of the second scan.
- $\mathbf{x_i'}$: The point $\mathbf{x_i}$ mapped into the coordinate frame of the first scan according to the parameter vector $\mathbf{p}$, that is $\mathbf{x_i'} = T(\mathbf{x_i}, \mathbf{p})$.
- $\mathbf{\Sigma_i}, \mathbf{q_i}$: The covariance matrix and the mean of the corresponding normal distribution to point $\mathbf{x_i'}$, looked up in the NDT of the first scan.

The mapping according to $\mathbf{p}$ could be considered optimal, if the sum evaluating the normal distributions of all points $\mathbf{x}_i'$ with parameters Σ_i and q_i is a maximum. We call this sum the `score` of $\mathbf{p}$. It is defined as:

$$\texttt{score}(\mathbf{p}) = \sum_i \exp\left(-\frac{1}{2}(\mathbf{x_i'} - \mathbf{q_i})^{\mathbf{t}} \mathbf{\Sigma_i^{-1}} (\mathbf{x_i'} - \mathbf{q_i})\right) \qquad (3)$$

This `score` is optimized in the next section.

6 Optimization using Newton's algorithm

Since optimization problems normally are described as minimization problems, we will adopt our notation to this convention. Thus the function to be minimized in this section is $-$`score`. Newton's algorithm iteratively finds the parameters $\mathbf{p} = (p_i)^t$, that minimize a function f. Each iteration solves the following equation:

$$\mathbf{H}\mathbf{\Delta p} = -\mathbf{g} \qquad (4)$$

Where $\mathbf{g}$ is the transposed gradient of f with entries

$$g_i = \frac{\partial f}{\partial p_i} \tag{5}$$

and $\mathbf{H}$ is the Hessian of f with entries

$$H_{ij} = \frac{\partial f}{\partial p_i \partial p_j}. \tag{6}$$

The solution of this linear system is an increment $\mathbf{\Delta p}$, that is added to the current estimate:

$$\mathbf{p} \leftarrow \mathbf{p} + \mathbf{\Delta p} \tag{7}$$

This is a step in a descent direction, provided that the Hessian is positive definite. If this is not the case, the model-trust region approach proposes to replace $\mathbf{H}$ by $\mathbf{H}' = \mathbf{H} + \lambda \mathbf{I}$, with λ chosen such that $\mathbf{H}'$ is positive definite. Details on the minimization algorithm itself can be found for example in [5].

This algorithm is now applied to the function `-score`. The gradient and the Hessian are built by collecting the partial derivatives of all summands of equation 3. For a shorter notation and to avoid confusing the parameter number i and the index of the laser scan sample i, the index i for the sample number is dropped. Additionally, we write

$$\mathbf{q} = \mathbf{x_i'} - \mathbf{q_i} \tag{8}$$

As can be verified easily, the partial derivatives of $\mathbf{q}$ with respect to $\mathbf{p}$ equal the partial derivatives of $\mathbf{x_i'}$. One summand s of the sum in equation 3 is then given by

$$s = -\exp \frac{-\mathbf{q^t \Sigma^{-1} q}}{2}. \tag{9}$$

The entries of the gradient are then (using the chain rule):

$$\begin{aligned} g_i &= \frac{\partial s}{\partial p_i} = \frac{\partial s}{\partial \mathbf{q}} \frac{\partial \mathbf{q}}{\partial p_i} \\ &= \mathbf{q^t \Sigma^{-1}} \frac{\partial \mathbf{q}}{\partial p_i} \exp \frac{-\mathbf{q^t \Sigma^{-1} q}}{2}. \end{aligned} \tag{10}$$

The partial derivatives of $\mathbf{q}$ with respect to p_i are given by the Jacobi matrix $\mathbf{J_T}$ of T (see equation 2):

$$\mathbf{J_T} = \begin{pmatrix} 1 & 0 & -x \sin \phi - y \cos \phi \\ 0 & 1 & x \cos \phi - y \sin \phi \end{pmatrix}. \tag{11}$$

The entries of the Hessian $\mathbf{H}$ are given by:

$$\begin{aligned} H_{ij} = \frac{\partial s}{\partial p_i \partial p_j} &= \exp \frac{-\mathbf{q^t \Sigma^{-1} q}}{2} \\ &((\mathbf{q^t \Sigma^{-1}} \frac{\partial \mathbf{q}}{\partial p_i})(-\mathbf{q^t \Sigma^{-1}} \frac{\partial \mathbf{q}}{\partial p_j}) + \\ \mathbf{q^t \Sigma^{-1}} &\frac{\partial^2 \mathbf{q}}{\partial p_i \partial p_j} + (\frac{\partial \mathbf{q}}{\partial p_j})^t \mathbf{\Sigma^{-1}} \frac{\partial \mathbf{q}}{\partial p_i}). \end{aligned} \tag{12}$$

The second partial derivatives of $\mathbf{q}$ are (see eq. 11):

$$\frac{\partial^2 \mathbf{q}}{\partial p_i \partial p_j} = \begin{cases} \begin{pmatrix} -x \cos\phi + y \sin\phi \\ -x \sin\phi - y \cos\phi \end{pmatrix} & i = j = 3 \\ \begin{pmatrix} 0 \\ 0 \end{pmatrix} & \text{otherwise} \end{cases} \tag{13}$$

As can be seen from these equations, the computational costs to build the gradient and the Hessian are low. There is only one call to the exponential function per point and a small number of multiplications. The trigonometric functions only depend on ϕ, the current estimate for the angle and must therefore be called only once per iteration. The next two sections will now use this algorithm for position tracking and for SLAM.

7 Position tracking

This section describes, how the scan match algorithm can be applied to tracking the current position from a given time $t = t_{start}$. The next section then extends this approach to SLAM.

The global reference coordinate frame is defined by the local robot coordinate frame at time t_{start}. The respective laser scan is called the *keyframe* in the following. Tracking is performed with respect to this keyframe. At time t_k, the algorithm performs the following steps:

1. Let δ be the initial estimate for pose difference between time t_{k-1} and t_k (for example from the odometry).
2. Calculate a position estimate of time t_k according to δ.
3. Perform the optimization algorithm using the current scan, the NDT of the keyframe and the new position estimate.
4. Check, whether the the keyframe is “near” enough to the current scan. If yes, iterate. Otherwise take the last successfully matched scan as new keyframe.

The decision, whether a scan is still near enough is based on a simple empiric criterion involving the translational and the angular distance between the keyframe and the current frame and the resulting score. Figure 3 shows an example of a keyframe, the last matched scan and the reconstructed robot trajectory.

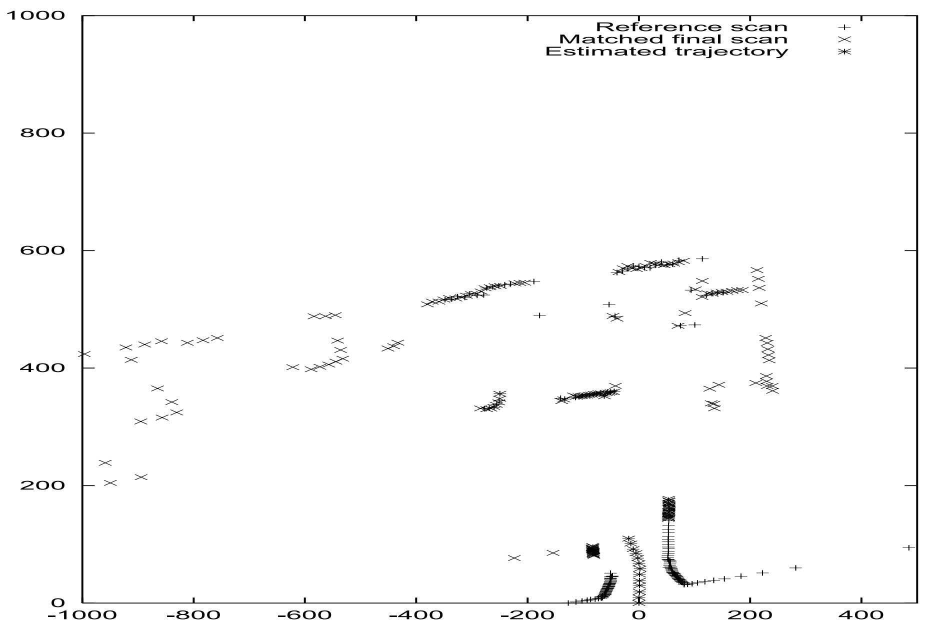

Figure 3: An example for the position tracking algorithm. Shown are the keyframe, the matched current scan and the reconstructed trajectory of the robot. Lengths are given in cm.

8 Application to SLAM

We define a map as a collection of keyframes together with their global poses. This section describes how to localize with respect to this map and how to extend and optimize this map, when the robot reaches unknown territory.

8.1 Localizing with respect to multiple scans

To each scan i in the map, an angle ϕ_i (or a rotation matrix $\mathbf{R}_i$) and a translation vector $(t_x, t_y)_i^t = \mathbf{T}_i$ is associated. These describe the pose of scan i in the global coordinate frame. The current robot pose is denoted by a rotation matrix $\mathbf{R}$ and a translation vector $\mathbf{T}$. The mapping T' from the robot coordinate frame to the coordinate frame of scan i is then given by:

$$T' : \begin{pmatrix} x' \\ y' \end{pmatrix} = \mathbf{R}_i^t(\mathbf{R} \begin{pmatrix} x \\ y \end{pmatrix} + \mathbf{T} - \mathbf{T_i}) \tag{14}$$

Only small changes are required to adapt the algorithm of section 7 to this situation. The mapping of a 2D-point of scan i is now calculated by applying T'. Further, the Jacobian $\mathbf{J}'$ and the second partial derivatives of T' get now slightly more complicated. The Jacobian of the mapping is now given by:

$$\mathbf{J_{T'}} = \mathbf{R_i}^t \mathbf{J_T} \tag{15}$$

and second partial derivatives of T' are now given by:

$$\frac{\partial^2 \mathbf{x}}{\partial p_i \partial p_j} = \begin{cases} \mathbf{R_i}^t \begin{pmatrix} -x\cos\phi + y\sin\phi \\ -x\sin\phi - y\cos\phi \end{pmatrix} & i = j = 3 \\ \begin{pmatrix} 0 \\ 0 \end{pmatrix} & \text{otherwise} \end{cases} \tag{16}$$

The gradient and the Hessian for the optimization algorithm could be built by summing over all overlapping scans. But we found an alternative, that is faster and yields equally good results: For each sample of the scan taken at the robot position, determine the scan, where the result of evaluating the probability density is maximal. Only this scan is used for this sample and for the current iteration. This way, the operations needed to build the gradient and the Hessian for the optimization algorithm are independent of the number of overlapping keyframes, except for finding the mentioned maximum.

8.2 Adding a new keyframe and optimizing the map

At each time step, the map consists of a set of keyframes with their poses in the global coordinate frame. If the overlap of the current scan with the map is too small, the map is extended by the last successfully matched scan. Then every overlapping scan is matched separately to the new keyframe, yielding the relative pose between the two scans. A graph is maintained, that holds the information of the pairwise matching result.

In this graph, every keyframe is represented by a node. A node holds the estimate for the pose of the keyframe in the global coordinate frame. An edge between two nodes indicates that the corresponding scans have been pairwise matched and holds the relative pose between the two scans.

After a new keyframe is added, the map is refined by optimizing an error function defined over the parameters of all keyframes. The results of the pairwise registration are used to define a quadratic error model for each matched pair as follows: The global parameters of two scans also define the relative pose between two scans. Let Δp be the difference between the relative pose defined by the global parameters and the relative pose defined by the result of the pairwise matching. Then we model the score of this two scans as a function of Δp, using the quadratic model

$$\texttt{score}'(\Delta p) = \texttt{score} + \frac{1}{2}(\Delta p)^t \mathbf{H}(\Delta p). \tag{17}$$

Thereby `score` is the final score, when the pairwise matching had converged and $\mathbf{H}$ is the thereby obtained Hessian. This model is derived by a Taylor expansion of `score` around $\mathbf{\Delta p} = \mathbf{0}$ up to the quadratic term. Notice, that the linear term is missing, because we expanded about an extreme point. This score is now summed over all edges and optimized.

If the number of keyframes gets large, this minimization can no longer be performed under real-time conditions (The number of free parameters is $3N-3$, where N is the number of keyframes). We therefore optimize only on a subgraph of the map. This subgraph is built by collecting all keyframes, that can be reached from the node of the new keyframe by traversing no more than three edges. We optimize the error function above now only with respect to the parameters, that belong to the keyframes contained in this subgraph. Of course, if a cycle has to be closed, the parameters of all keyframes should be optimized.

9 Experimental results

In this section, we present the results of our SLAM algorithm on three different data sets. The first set was recorded by our own robot in our building in Tübingen. The second set is taken from the CMU Mine mapping project [1], the third is recorded data of di Castello Belgioioso [10]. The maps we present in this section always consist of the set of keyframes, that was built during processing the laser scans. But first we show two examples of scan match iterations in detail.

9.1 Iterative scan matching

Figure 4 and figure 5 are showing two examples of scan match iterations in detail. In the first example, a large rotation of about -0.57rad and only a small translation of about 8 cm is successfully estimated in ten iterations. The second example shows a typical corridor scene, with almost no rotation.

With moderately fast robots and high laser scan sampling rates, initial misalignments are much smaller. On average, our algorithm normally converges in about five iterations for small misalignments (< 10 cm and $< .1$ rad).

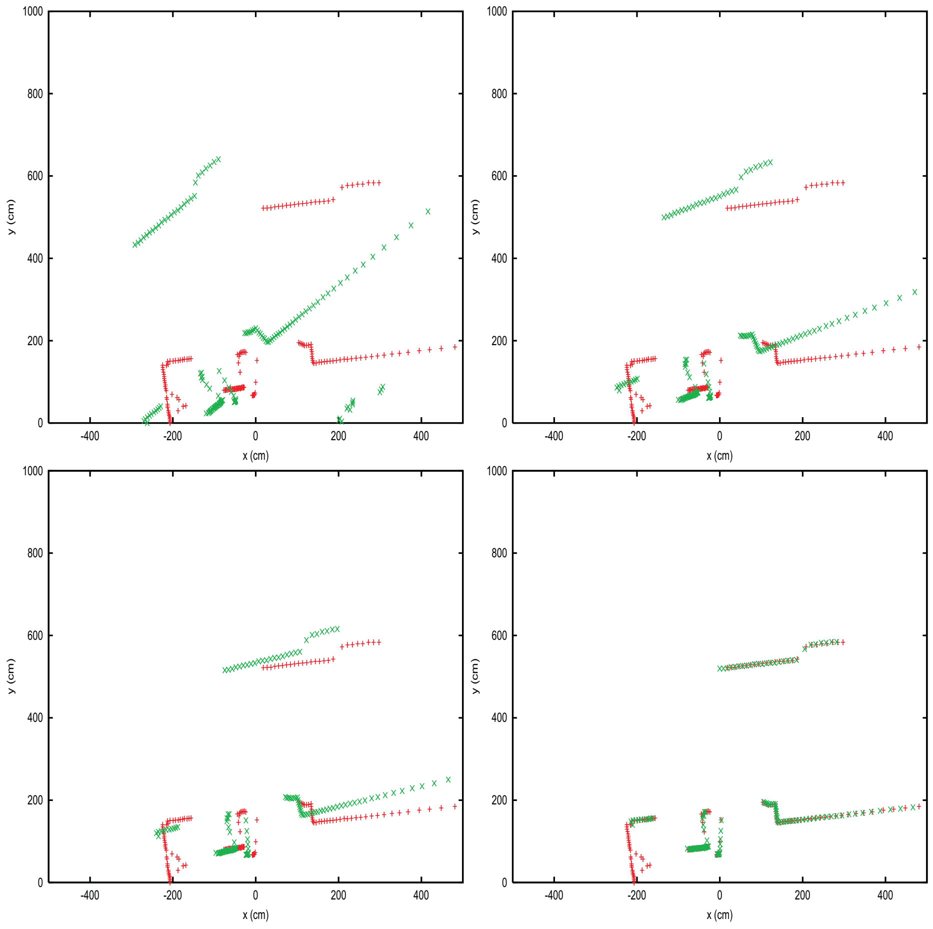

Figure 4: The scan match algorithm in action. From top left to right bottom: Initial configuration, after the first iteration, after five iterations and after ten iterations.

9.2 Experiment 1

The results of experiment 1 were performed *without using odometry*. This should demonstrate the robustness of the approach. Of course, as Thrun already noted in [17], this is only possible as long as any 2D structure is present in the world. The map presented in figure 6 was acquired by driving the robot out of its lab, up the corridor, down the corridor and then back into the lab. So the situation requires both the extension of the map and the localization with respect to

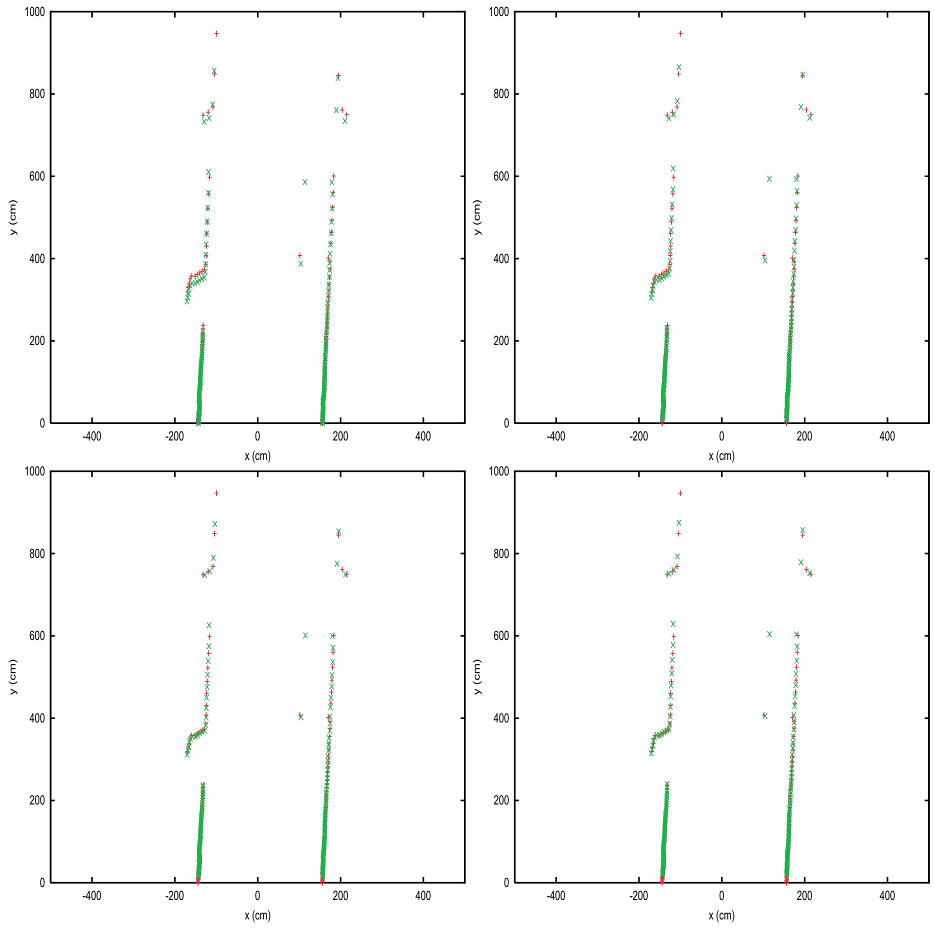

Figure 5: Another example. From top left to right bottom: Initial configuration, after the first iteration, after the second iteration and after six iterations.

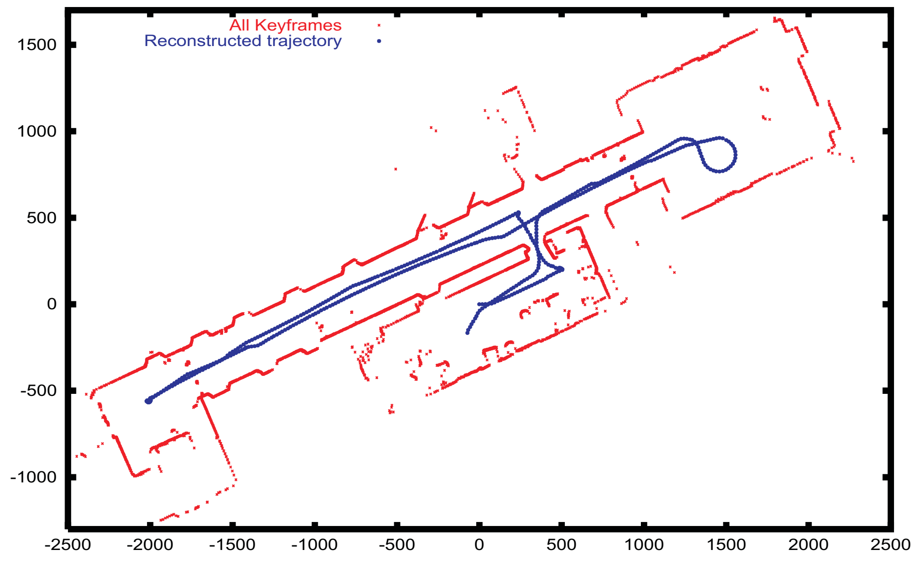

Figure 6: A map, that was composed using our scan matcher. Lengths are given in cm. Shown are the set of keyframes and the estimated trajectory.

the map. The robot collected 28430 laser scans during a travel of 20 minutes, where it traversed approximately 83 meters. The scans were taken with a SICK laser scanner covering 180 degrees with an angular resolution of one degree. To simulate a higher speed, only every fifth scan was used. The simulated speed is then around 35 cm/s and the number of scans per second is around 23.

The map was built using a combined strategy. The position tracker of section 7 was applied to every scan, whereas we initialize the parameters by extrapolating the result of the last time step linearly. Every tenth scan, the procedure of section 8 was applied.

Figure 6 shows the resulting map. Shown are the 33 keyframes, that the final map consisted of. A closer look reveals also, that our scan match algorithm is tolerant against small changes in the environment like opened or closed doors. Processing all frames offline needs 58 seconds on a 1.4 GHz machine, that's 97 scans per second.

9.3 Experiment 2

The data of this experiment is from the CMU mine mapping project ([1], available under [10]) and contains two cycles. Our algorithm is able to close such cycles only, if the scan positions are estimated well enough, such that the distance between the keyframes to be connected is smaller than the convergence radius. In this example, the estimated map is already nearly correct, when the robot finished the cycles. Figure 7 and figure 8 show the map before and after closing the first cycle. Figure 9 and figure 10 show the map before and after closing the second cycle.

It took 116 seconds to process the 6277 scans, that the dataset consisted of. That's around 54 processed scans per second.

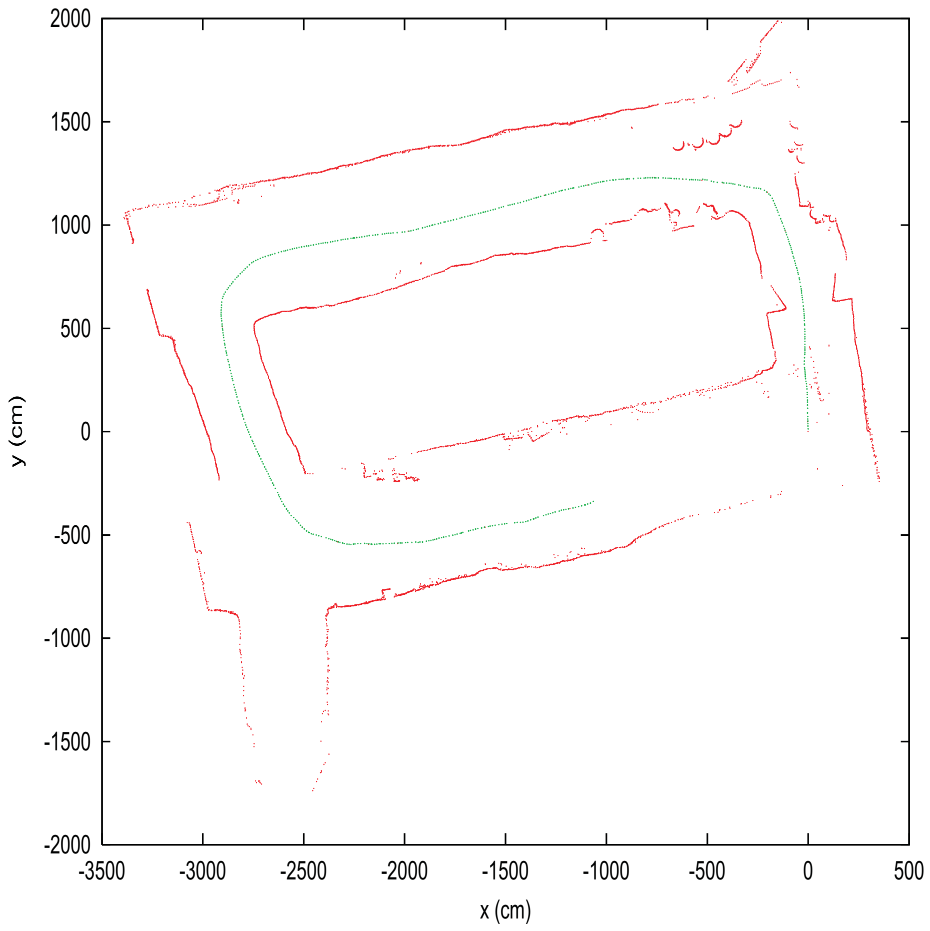

Figure 7: Map of the mine (before closing first cycle)

9.4 Experiment 3

The third data set, that was used for evaluation, is also available from [10] (the largest map of the di Castello Belgioioso set, 4047 scans). This time, our algorithm didn't work without using odometry. Reason for that is, that there are often rotations of more than 45 degrees between two subsequent scans, where our algorithm fails to find the solution. So the maps presented here were created using the odometry to get initial estimates for the movement of the robot between two scans. Figure 11 shows the map before closing the cycle. Again,

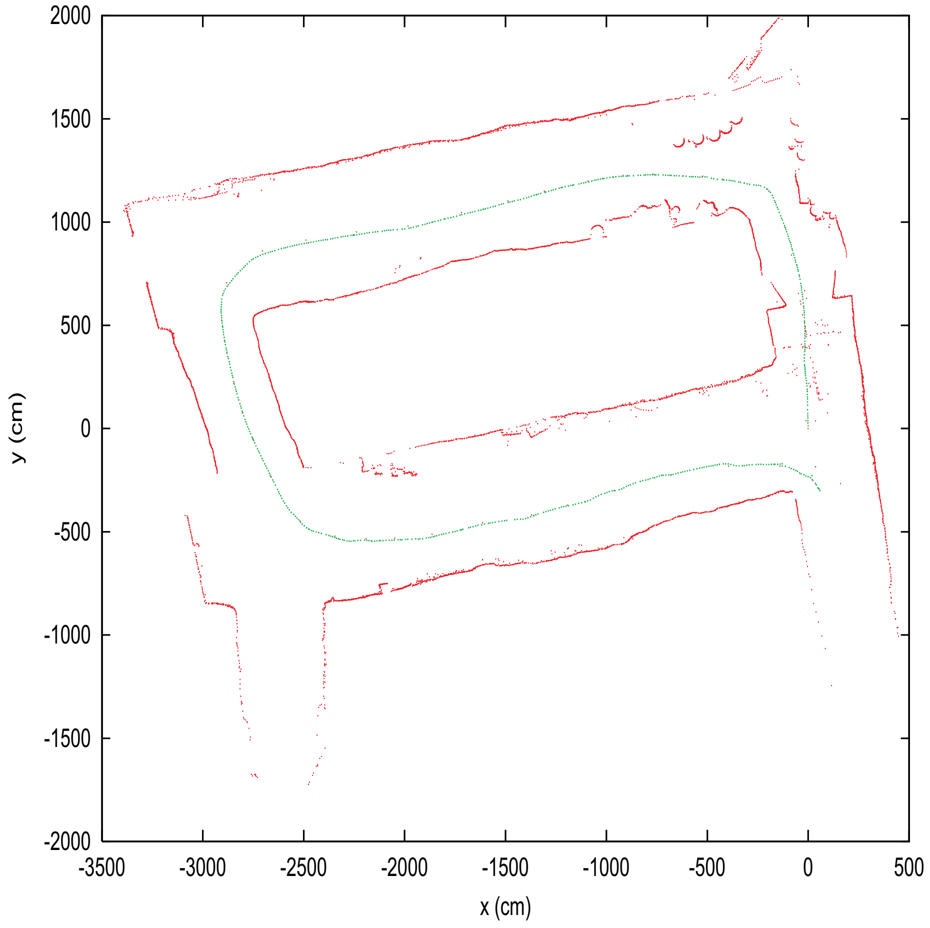

Figure 8: Map of the mine (after closing first cycle)

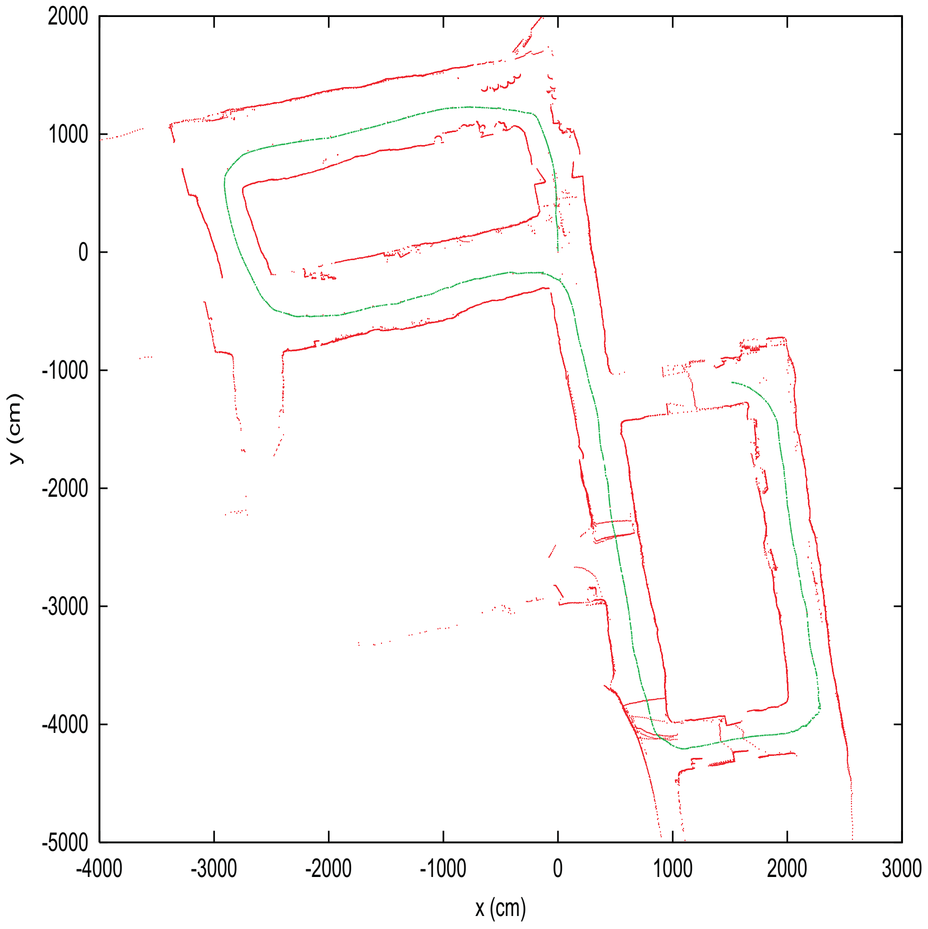

Figure 9: Map of the mine (before close second cycle)

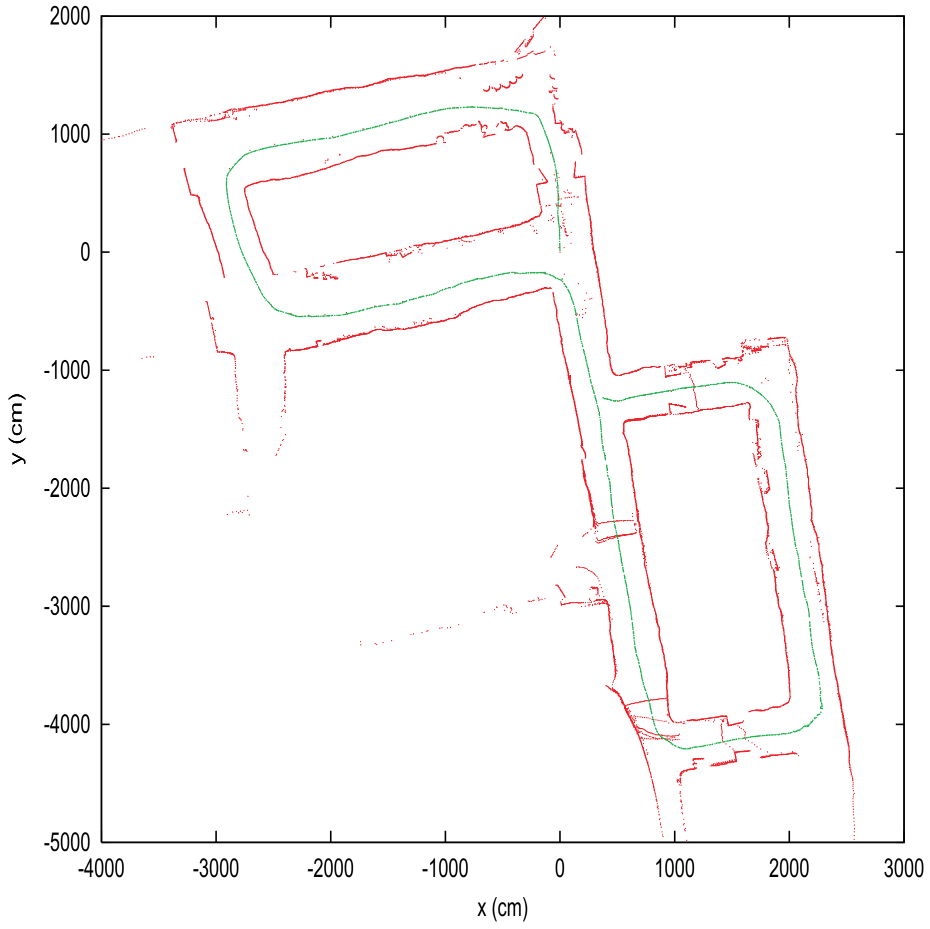

Figure 10: Map of the mine (after closing second cycle)

the estimates are good enough, that our algorithm can close the cycle (figure 12). The estimated trajectory is not updated, it can be seen, that the estimates for the scan positions have considerably changed after closing the cycle.

Around scan 3700, there is a long, relatively featureless corridor. Here, the algorithm fails. Figure 13 shows the map, shortly before the failure occurs. Possibly such situations could be handled by enforcing the distance between the scans to be in agreement with the odometry and using the scanmatcher only to estimate the angle, but that's future work.

10 Conclusion and future work

We have presented a new representation for a set of points, the Normal Distributions Transform (NDT). This transformation can be used to derive analytic expressions for matching another scan. We also showed, how our scan matcher could be incorporated for the problem of position tracking and for the SLAM problem. The major advantages of our method are:

- No explicit correspondences between points or features have to established. Since this is the most error-prone part in most approaches, we are more robust without correspondences.
- All derivatives can be calculated analytically. This is both fast and correct.
- The method is fast enough to run in real-time.

There are some open questions, which we will investigate in future work.

- What can be said about the typical convergence radius of the method, especially compared to ICP? This is also an important issue for the global localization problem.
- Is the representation by a set of keyframes really the best possibility or should the scans be integrated (at least locally) for position tracking?
- Featureless corridors pose a problem up to date, this situation must be detected and handled.
- Lastly, a probabilistic framework should be set atop our approach.

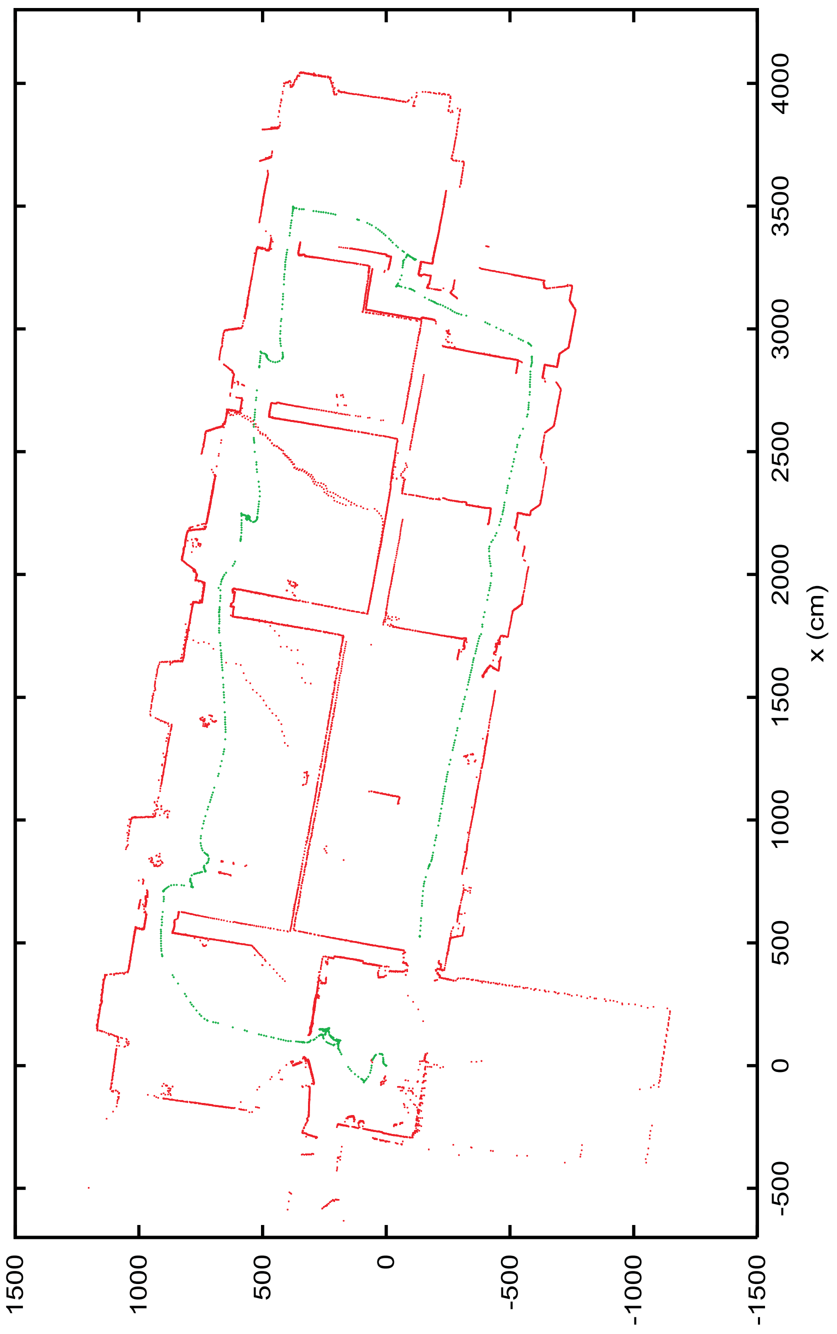

Figure 11: Map before closing cycle.

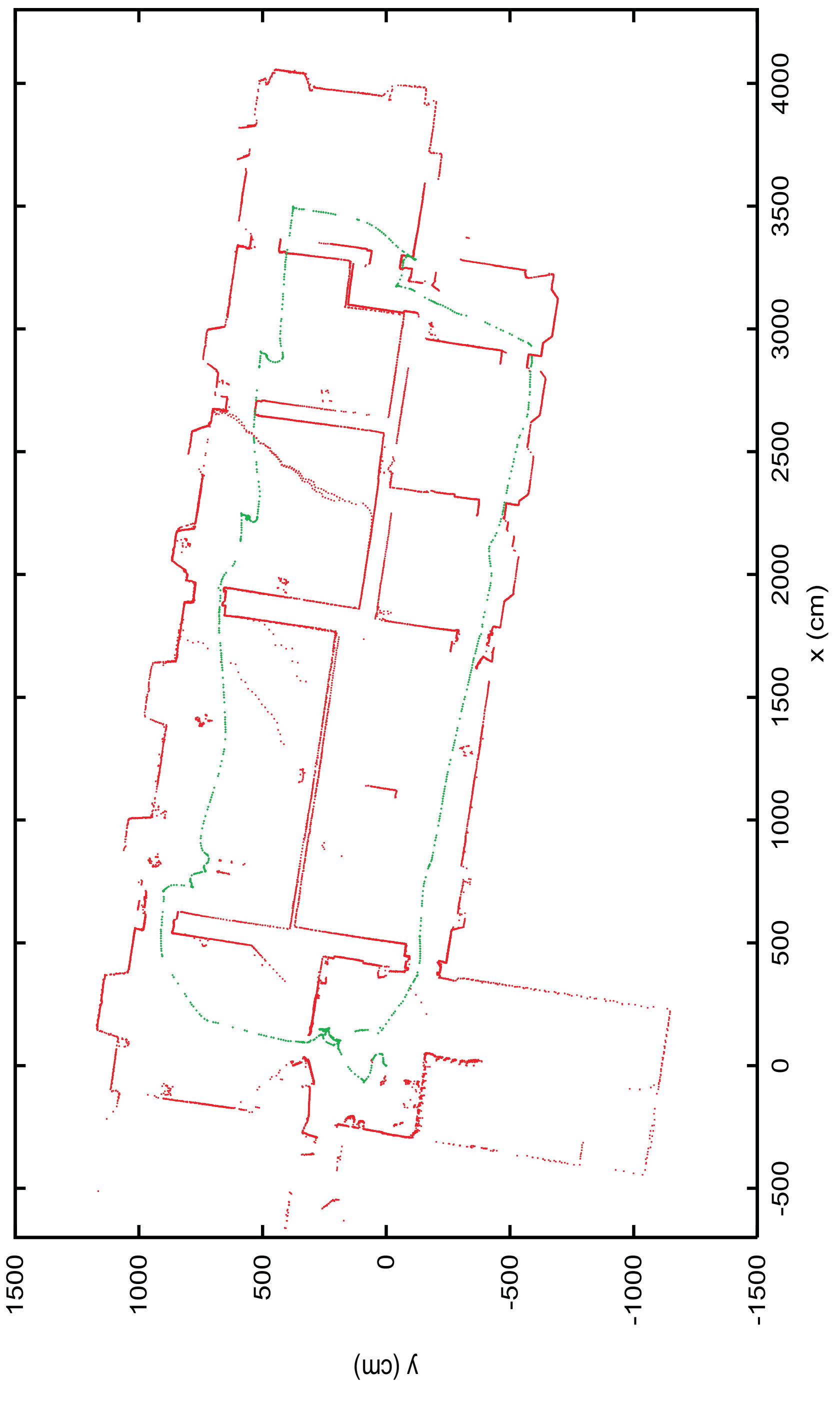

Figure 12: Map after closing cycle.

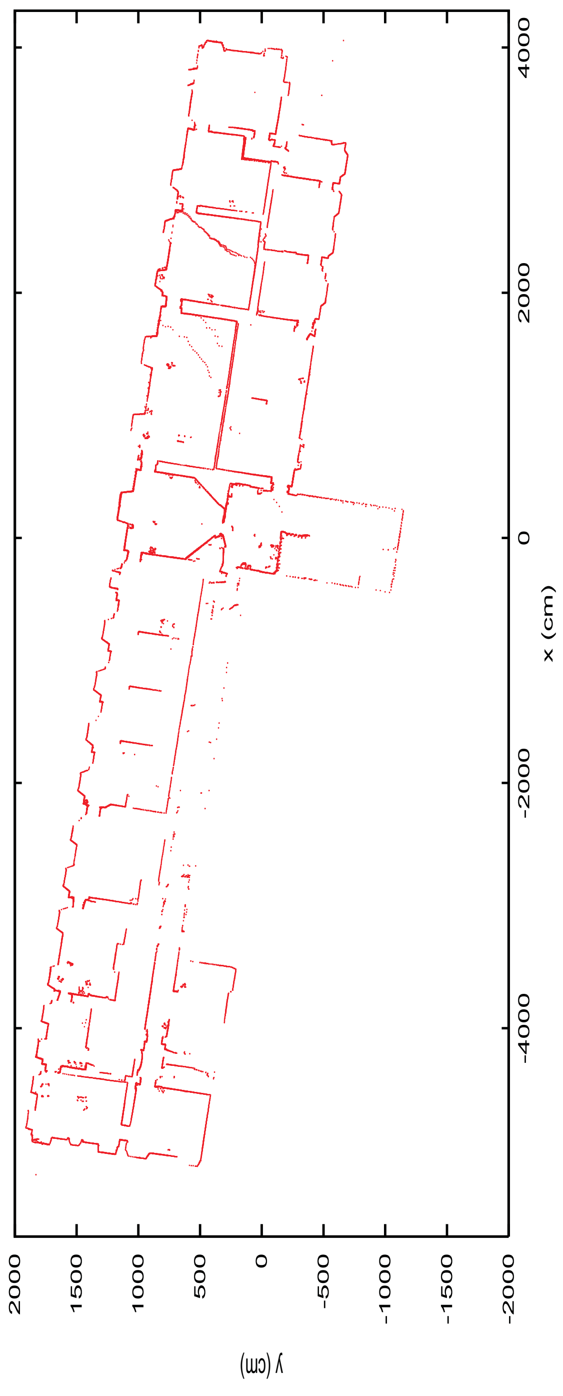

Figure 13: The map, shortly before our algorithm failed in a long corridor.

References

[1] Cmu's robotic mine mapping project. http://www-2.cs.cmu.edu/ 3D/mines/index.html.

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

[3] Y. Chen and G.G. Medioni. Object modeling by registration of multiple range images. *Image and Vision Computing*, 10(3):145–155, 1992.

[4] I.J. Cox. Blanche: An experiment in guidance and navigation of an autonomous robot vehicle. *IEEE Transactions on Robotics and Automation*, 7(2):193–204, 1991.

[5] J.E. Dennis and R. B. Schnabel. Numerical Methods for Unconstrained Optimization and Nonlinear Equations. Prentice-Hall, Englewood Cliffs, 1983.

[6] J. Gutmann, W. Burghard, D. Fox, and K. Konolige. An experimental comparison of localization methods. In *Proc. International Conference on Intelligent Robots and System*, 1998.

[7] J. Gutmann and K. Konolige. Incremental mapping of large cyclic environments. In *Proceedings of the IEEE International Symposium on Computational Intelligence in Robotics and Automation (CIRA)*, 2000.

[8] J. Gutmann and C. Schlegel. Amos: Comparison of scan matching approaches for self-localization in indoor environments. In *Proceedings of the 1st Euromicro Workshop on Advanced Mobile Robots.* IEEE Computer Society Press, 1996.

[9] J. Gutmann, T. Weigel, and B. Nebel. Fast, accurate, and robust self-localization in polygonal environments. In *Robocup workshop at IJCAI-99*, 1999.

[10] Dirk Hähnel's Homepage. www.informatik.uni-freiburg.de/ haehnel/.

[11] F. Lu and E. Milios. Robot pose estimation in unknown environments by matching 2d range scans. In *CVPR94*, pages 935–938, 1994.

[12] F. Lu and E. Milios. Globally consistent range scan alignment for environment mapping. *Autonomous Robots*, 4:333–349, 1997.

[13] A. Scott, L.E. Parker, and C. Touzet. Quantitative and qualitative comparison of three laser-range mapping algorithms using two types of laser scanner data. In *IEEE International Conference on Systems, Man and Cybernetics*, 2000.

[14] Jianbo Shi and Carlo Tomasi. Good features to track. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR'94)*, pages 593–600, Seattle, June 1994.

[15] R. Szeliski and H.Y. Shum. Creating full view panoramic image mosaics and environment map. In *Computer Graphics (SIGGRAPH 97)*, pages 251–258, 1997.

[16] S. Thrun. Robotic mapping: A survey. In G. Lakemeyer and B. Nebel, editors, *Exploring Artificial Intelligence in the New Millenium*. Morgan Kaufmann, 2002.

[17] S. Thrun, W. Burgard, and D.Fox. A real-time algorithm for mobile robot mapping with applications to multi-robot and 3d mapping. In *Proceedings of IEEE International Conference on Robotics and Automation*, 2000.

[18] Sebastian Thrun, Dieter Fox, Wolfram Burgard, and Frank Dellaert. Robust monte carlo localization for mobile robots. *Artificial Intelligence*, 128(1-2):99–141, 2001.

[19] G. Weiss and E. von Puttkamer. A map based on laserscans without geometric interpretation. In *Proceedings of Intelligent Autonomous Systems 4 (IAS-4)*, pages 403–407, 1995.

[20] Z. Zhang. Iterative point matching for registration of free-from curves and surfaces. *International Journal of Computer Vision*, 13(2):119–152, 1994.