Real-Time Self-Localization in Unknown Indoor Environments using a Panorama Laser Range Finder

Tobias Einsele
Laboratory for Process Control and Real–Time Systems
Prof. Dr.–Ing. Georg Färber
Technische Universität München, Germany
Tobias.Einsele@lpr.e-technik.tu-muenchen.de

Abstract

This paper deals with self-localization of a mobile robot on the condition that no a-priori knowledge about the environment is available. The applied method features to be accurate, robust, independent of any artificial landmarks and feasible with such a moderate computational effort that all necessary tasks can be executed in real-time on a standard PC. The perception system used is a panorama laser range finder (PLRF) which takes scans of its present environment. A modified Dynamic Programming (DP) algorithm provides pattern matching and pattern recognition on the preprocessed panorama scans and thereby renders a qualitative fusion of the sensory data. For an exact quantitative estimate of the robot's current position, a robust localization module is employed. The knowledge gained about the environment along that way is stored in a self-growing, graph based map which combines geometrical information and topological restrictions. Preliminary experiments in a common office environment proved the reliability and efficiency of the system.

1. Introduction

A mobile robot should be able to travel through its environment safely and systematically with as little human instruction and intervention as possible. This requires that it is capable to perform the high level tasks of path planning, navigation and exploration in a reliable way. The achievement of this goal basically depends on the accuracy and efficiency of the robot's self-localization system which has to solve a great deal of time-critical and challenging problems. In order to obtain an up to date position estimate during robot motion, the self-localization module has to perform real-time processing of the data delivered from internal and external sensors. Further, it needs to cope with the errors induced by imprecise or erroneous sensor readings and has to establish correspondences among the single sensory perceptions in order to fuse the gathered information. Finally the acquired knowledge is to be archived in a convenient representation. This is done to provide a working basis for the above mentioned high level tasks and to give support in treating the so-called re-entry problem. This becomes an extremely demanding challenge if no or only little a-priori knowledge about the environment is available.

On the other hand, one has to take into account that it is *not* necessary for an efficient self-localization to know *always* the precise geometrical location of the robot. E. g. when traveling along a corridor containing no 'distinctive places', the exact position of the robot within the corridor is of no relevance. The only thing that matters in this situation is that the robot knows in which direction it has to go and recognizes the place at the end of the corridor in order to re-localize exactly again [6, 7]. So one can conclude that a geometrically exact localization is only required when arriving at or moving within such a 'distinctive' place. Therefore, instead of an environmental map referring to a global coordinate system, a map handling several local coordinate systems linked to each other only by topological information is employed in this approach.

In this paper a complete self-localization system is introduced which solely utilizes a panorama laser range finder (PLRF) for obtaining information about the robot's position in the environment. A general view of the over-all structure can be gained from figure 1.

The paper is organized as follows: The next section gives a rough survey of the PLRF and the range data acquisition. Section 3 deals with the preprocessing unit which extracts line segments from the PLRF raw data. These line segments are passed to the Dynamic Programming (DP) based recognition and matching unit (section 4). The determination of the robot's exact location from the qualitative output of the matching unit is done by a localization module which is ad-

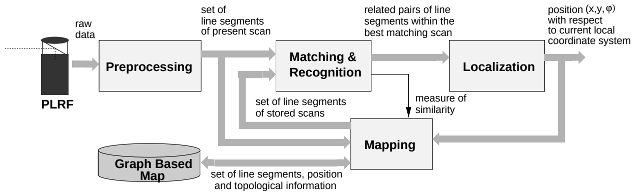

Figure 1. Structure of the self-localization system

dressed in section 5. Section 6 describes the organization and construction of the self-growing, graph based environmental map. In section 7 preliminary experimental results validate the presented approach.

2. Range data acquisition

The PLRF used here consists of an Acuity AccuRange 4000 laser range finder (LRF) together with a mirror/motor combination which generates a nearly $360°$ panorama line scan of the current environment at the height at which the sensor is mounted. The LRF does not deliver complete range measurements, but just provides raw data which is received by the AccuRange High Speed Interface board on which the information is intermediately stored and segmented into single scans. Transferring the data from the interface board to the host's memory, eliminating the dependencies on temperature, ambient light and target reflectivity as well as computing the actual range values is done on a standard i586 PC running under the Linux operating system. As memory on the interface board is limited, the data transferring task turned out to be the most time-critical and is therefore executed within a high priority interrupt service routine under control of a driver programme running in Linux kernel mode. Along this way, the system features a sampling rate of up to $50,000$ range samples per second which means that even if the scanning motor is running with its maximum rotational speed of about $3,000\ r.p.m.$ the angular resolution is still finer than $0.5°$. The range of the LRF reaches from $0.25\ m$ up to 16 m; the measurement error which was observed to be independent of the present distance is Gaussian distributed, and can therefore be expressed by its standard deviation which amounts to $\sigma = 2.5\,mm$.

3. Range data preprocessing

The task of the preprocessing unit is to extract line segments from the acquired range data. In other approaches classical clustering techniques [10], histograms [11] or the Hough transform [4] are used to solve this problem. All of these methods have the property that the range data has to be transformed into a feature space. This requires that from each range sample to e. g. its direct successor a connection line is drawn of which the Hesse normal representation with angle α and distance d is computed. The (α, d) parameter pairs are then clustered or histogrammed in the feature space. This procedure has the serious drawback that if the sampling density is high due to a small range from the sensor, the (α, d) pairs are significantly distorted although the range measurement error is quite moderate (fig. 2). Of

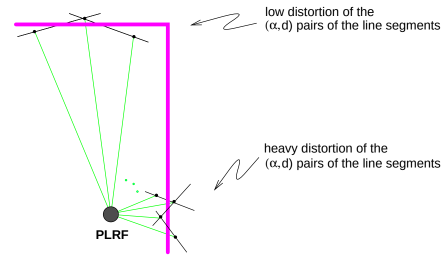

Figure 2. Parameter distortion in feature space

course, there are techniques to compensate for this handicap (averaging over several samples, range weighting etc.). On the other hand a strategy which preprocesses data in the original space itself seems to be much more straightforward.

The method presented in this paper is two-stage: First, the whole laser scan is segmented into piecewise linear sections. This is achieved by an iterative algorithm which originates from image processing and for which an effort minimized implementation exists [9]. It is therefore capable to process the rather few sampling points of a laser scan (compared to the large amount of pixels in an image) faster than e. g. a cluster algorithm – even if the simplifications induced by the successive order of range samples is considered during clustering. Another advantage is that the criterion for a valid line segment can directly be derived from geometrical considerations: Two arbitrary samples which are located not too closely to each other, are connected by a line. For all samples lying between this start and end point the rectangular distances to the line are computed. If the maximum distance exceeds a given threshold, the segment is split at that point into two subsegments. For each of these, the procedure is applied again. If the maximum distance keeps below that threshold and if the number of sampling points associated with that segment does not fall short of a specified minimum, the line segment is established (fig. 3). As can easily be seen, a high sampling density will not affect the accuracy of the segmentation algorithm.

The second step is fitting the segmented range measurements to lines. As possibly outlying samples were already removed by the above procedure, the classical least squares method is utilized which provides a robust and mathematically qualified solution for this problem.

The output of the preprocessing unit is one set of line segments per panorama scan. Each of these segments is

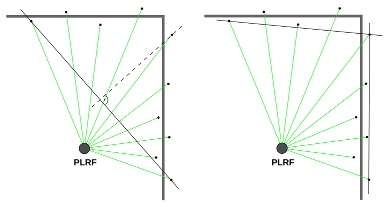

Figure 3. Line segmentation algorithm

specified by the (α, d) tuple of the Hesse normal representation, the angle φ and the angle sector $\Delta\varphi$, all with respect to the robot's coordinate system as it is illustrated in figure 4.

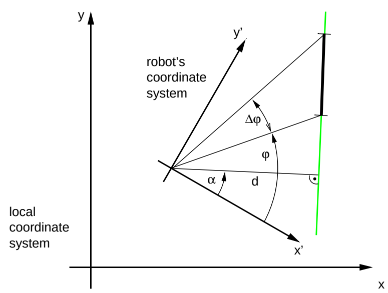

Figure 4. Representation of a line segment

4. Matching and recognition

This module performs the matching between present preprocessed scans and already stored scan data (see section 6) as well as recognition of the places in which the scans were taken. In contrast to related work in which neural network models [10], correlation of the laser scans [11] or least squares methods [2] fulfil these tasks, a Dynamic Programming (DP) [1] technique is proposed here.

When comparing matching algorithms, the computational effort is always a very decisive criterion. The DP algorithm is of order $O(n^2)$ and can therefore be put on the same level with the correlation and the least squares approaches. However, what makes DP superior to those is that it can be massively accelerated, which of course does not mean decreasing its order, but can reduce the de facto processing effort in a very significant way [8].

The DP is a classical pattern matching algorithm which establishes and evaluates correspondences between a reference and a test pattern. In this context the term *pattern* denotes the set of line segments obtained from the preprocessing unit. The actual matching of two patterns is done by finding the optimal path through a matrix of grid points which is spanned by the similarity measures between line segments of the reference set and line segments of the presently preprocessed, so-called test set. Since the sets of line segments are gained from panorama scans, the patterns to be handled by the DP are cyclic, which requires a modification of the original algorithm.

The output of the DP is a qualitative mapping between line segments of the reference scan and line segments of the test scan, as well as a measure which values the similarity between these. Therefore it is possible to offer not only a single, but several reference scans and determine the one which fits best to the present test scan. Along that way, recognition of scans can be realized.

4.1. Boundary conditions

Normally, the start point of the optimal path through the matrix of grid points is fixed to the lower left, the end point to the upper right corner. This is not allowed if patterns are cyclic because reference and test patterns may be rotated against each other, which implicates that the start and end point of the optimal path is located anywhere in the first and last column of the matrix, respectively. For the algorithm this results in admitting any grid point in the first column as a valid start and accordingly any point in the last column as a valid end point. A further consequence is that the optimal path has to perform a wrap-around. This means that it must be possible to jump from the last row back into the first row of the matrix.

Since reference and test patterns are to be matched against each other as completely as possible, a path through the matrix is only considered valid if it ends in the same row it has started (a loosening of this harsh rule is discussed in paragraph 4.3). Of course the optimal path is not guaranteed to obey these boundary conditions. In case of such a failure, the optimal path is dismissed in favour of a path which meets the requirements. However, this only happens if similarity between patterns is low. In addition, the path taken instead of the optimal one still performs a very good and reasonable assignment between the line segments, so that the practical benefit is not affected (fig. 5).

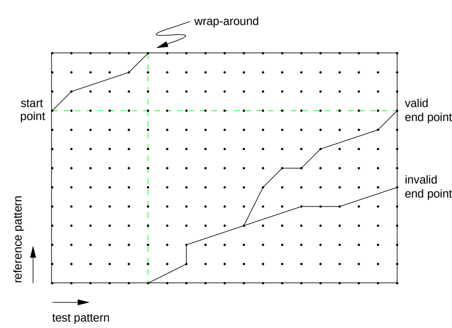

Figure 5. Paths through the matrix of grid points

4.2. Similarity measure

The similarity measure between two line segments is introduced as a cost function consisting of two additive components, which are referred to as *matching score* and *transition score*. The matching score represents the costs which are incurred by the immediate relation of two line segments. The transition score is a differential measure which estimates the costs which accrue from two successive assignments of line segments. Along this way, the history of already matched line segments can be taken into account.

The cost function s itself is heuristically determined and calculates as follows:

$$s(i,j,k,l) = s_1(i,j) + s_2(i,j,k,l)$$

with matching score:

$$\begin{aligned} s_1(i,j) = {} & c_1\,(d_j - d_i)^2 + \\ & c_2\,[((\varphi_j - \alpha_j) - (\varphi_i - \alpha_i))\,mod\,2\pi]^2 + \\ & c_2\,[((\varphi_j + \Delta\varphi_j - \alpha_j) - (\varphi_i + \Delta\varphi_i - \alpha_i))\,mod\,2\pi]^2 \end{aligned}$$

and transition score:

$$s_2(i,j,k,l) = c_3\,[((\alpha_j - \alpha_i) - (\alpha_l - \alpha_k))\,mod\,2\pi]^2$$

Indices i and k refer to line segments in the test pattern, whereas indices j and l apply to line segments in the reference pattern.

As can be seen from the above formulae, the matching score only considers the absolute differences in distances d as well as in angles φ and $(\varphi + \Delta\varphi)$. In contrast, the transition score evaluates a differential deviation in angle α. The coefficients c_1, c_2 and c_3 ensure that each addend makes an equal contribution to the over-all cost function s. This is achieved by separately histogramming the addends and then setting the values of the coefficients to the reciprocal values of the histograms' variances.

4.3. Local recombination

The local recombination of paths during the DP forward search is determined by the transitions allowed between the matrix grid points. A typical transition diagram consists of a diagonal, a horizontal and a vertical transition. This implicates that one, two or more line segments of the test pattern are allowed to be associated with one line segment of the reference pattern and vice versa. As an extension to this, it is useful that not every single line segment needs to be related to another one, which means that line segments are allowed to be skipped. This procedure has its real world analogue in the fact that it is quite possible that due to shadowing effects some line segments can only be seen in one of two laser scans, which feature quite a good similarity beside of these differences. For that reason, the transition diagram as illustrated in figure 6 is employed. As can also be seen in figure 6, there is an additional weighting factor associated with each transition. According to the number of line segments skipped, the value of the cost function s is multiplied by this coefficient in order not to disadvantage a complete assignment of the line segments. Local recombination of paths is realized then by taking the path which involves the least accumulative costs.

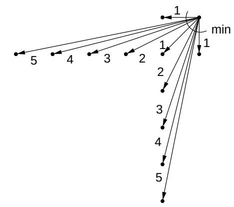

Figure 6. DP transition diagram

5. Localization

The DP algorithm gives a qualitative assignment between the line segments of the test scan and the line segments of the best matching reference scan. In the localization unit this information is used to determine the exact deviation $(\Delta x, \Delta y, \Delta\varphi)^T$ of the current position from a reference position on which a previous scan has been taken.

The procedure is as follows: Every related pair of line segments forms an equation like:

$$\cos\alpha_{h_\xi} \cdot \Delta x + \sin\alpha_{h_\xi} \cdot \Delta y = \left(d_{h_\xi} - d_{g_\xi}\right)$$

Index h_ξ refers to a line segment in the reference pattern, index g_ξ to the related line segment in the test pattern, respectively.

Since many pairs of related line segments are delivered by the DP, a linear equation system can be set up:

$$\underbrace{\begin{bmatrix} r_1\cos\alpha_{h_1} & r_1\sin\alpha_{h_1} \\ \vdots & \vdots \\ r_\xi\cos\alpha_{h_\xi} & r_\xi\sin\alpha_{h_\xi} \\ \vdots & \vdots \\ r_n\cos\alpha_{h_n} & r_n\sin\alpha_{h_n} \end{bmatrix}}_{\mathbf{A}} \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} = \underbrace{\begin{bmatrix} r_1\left(d_{h_1} - d_{g_1}\right) \\ \vdots \\ r_\xi\left(d_{h_\xi} - d_{g_\xi}\right) \\ \vdots \\ r_n\left(d_{h_n} - d_{g_n}\right) \end{bmatrix}}_{\mathbf{b}}$$

with $\sum_{\xi=1}^{n} r_\xi = 1$.

The coefficients $r_1, \ldots, r_n$ introduced in the above equation consider that the line segments consist of a different number of range samples. According to this, a matched line segment containing more sampling points will obtain a higher valued factor r_ξ than a matched segment including only a few range samples.

If the matrix $\mathbf{A}$ were square, the factors r_ξ would execute an identical operation with no effect at all. But since in most cases $\mathbf{A}$ has much more rows than columns, which means that the equation system is overdetermined, the factors fulfil their purpose of a different weighting of the single equations [5]. The so-called pseudo-inverse is involved then to solve the above equation system which results in a least squares solution for the translatory deviation vector $(\Delta x, \Delta y)^T$. The computational effort associated with this operation is very low because the pseudo-inverse is always just a (2×2)-matrix.

Furthermore, the error vector $\mathbf{w}$ is calculated:

$$\mathbf{w} = \mathbf{A} \cdot \left[\begin{array}{c} \Delta x \\ \Delta y \end{array} \right] - \mathbf{b}$$

The absolute values of the components of $\mathbf{w}$, $|w_\xi|$, represent how well the correlating equation fits into the equation system. If $|w_\xi|$ exceeds a threshold, e. g.

$$|w_\xi|^2 > \frac{1}{n} \cdot \|\mathbf{w}\|^2$$

the according equation is eliminated. Along this way, the equation system is relieved from inaccurate equations and can then be solved again with a significant gain of accuracy. This procedure is motivated by the fact that the implemented transition diagram (fig. 6) does not allow every conceivable assignment between line segments in order to save processing time. As this might lead to a path which includes some unfavourable relations, it is justified to eliminate these by omitting the according equations.

For the determination of the angle deviation $\Delta\varphi$ the average over all differences in angle α is computed:

$$\Delta\varphi = \sum_{\xi=1}^{n} r_\xi \left[(\alpha_{h_\xi} - \alpha_{g_\xi}) \, mod \, 2\pi \right]$$

It is obvious that the above method for increasing the accuracy influences the previous equation (less addends, new coefficients r_ξ), which implicates that the estimate for the angle deviation is also improved.

Since the deviation vector $(\Delta x, \Delta y, \Delta\varphi)^T$ from a reference position as well as the reference position itself with respect to the local coordinate system are known, the new position with respect to the local coordinate system is determined by a simple coordinate transformation.

6. Mapping

The information obtained about the surroundings by interpreting the PLRF data is now used to build a map of the robot's environment. The structure of the map presented in this approach is chosen in a way that the panorama scans can easily be added to the map's database when performing exploration, and can also be quickly accessed when localization or high level tasks are to be executed.

To achieve this, a graph with nodes and edges is used as the body of the map. Up to this stage of implementation a node represents a single scan which is specified by the set of line segments obtained from the preprocessing unit and, secondly, by the position on which the scan has been taken with respect to the current local coordinate system. An edge refers to a direct neighbourhood relation between two stored scans.

In order to gain and to preserve a useful database, the map needs to be updated and extended continuously. This is done by the mapping unit that shows the following behaviour which is different from what can be found in [3]: If it is possible to match a presently taken and preprocessed scan with an already stored scan, producing costs that fall below a given threshold, which means that the similarity measure exceeds a certain threshold, the robot is considered to be located inside the catchment area of an already existing node. Therefore, no new node will be established in the map. It is only checked if a new edge needs to be introduced. This would be the case if the robot had traveled into the catchment area of another node and if no edge already existed from the previous node to this node. If the costs are too high or the similarity measure is too low, respectively, a new node containing the set of line segments and the present position as well as a new edge going out from the previous node leading to the new one are established in the map.

As establishing a new node means entering unknown terrain, the plausibility of the position delivered by the localization module has to be checked. E. g. when traveling along a corridor with only two visible, parallel walls, the localization unit might give the misleading information that the robot stands still. In case of such a discrepancy which can easily be detected, the current local coordinate system is abandoned and a new local coordinate system is instantiated for the new node. The topological description how to get from a node of one local coordinate system to a node within another local coordinate system can be stored together with the according edge data. If the position estimate given by the localization unit is determined to be reliable, the current local coordinate system is kept and the position as it is calculated by the localization unit is stored together with the new node.

7. Experimental results

The self-localization system has been implemented as specified, and first experiments in a common office environment have been realized. The results presented below originate from an experiment in which the localization system acted as substitute for the robot's odometry. Since no a-priori knowledge about the environment was available, which means that in the beginning the environmental map was empty, the PLRF was in fact the only source of information. The test drive illustrated in figure 7 started in an office, led through a doorway into a corridor and ended in front of the door of another office. The total distance traveled amounts to about $8\ m$; the number of scans which were taken and processed adds up to 53. In this experiment the threshold for the acceptable matching costs was chosen that low that every acquired scan was stored as a new node in the map which resulted in a graph containing 53 nodes and 52 edges, respectively. This was done here to quantify the processing time per scan which was evaluated to be about $200\ ms$ on a i586 PC. As can be seen the scans could properly be superimposed although a doorway had to be passed, which implicated a significant change of the surroundings.

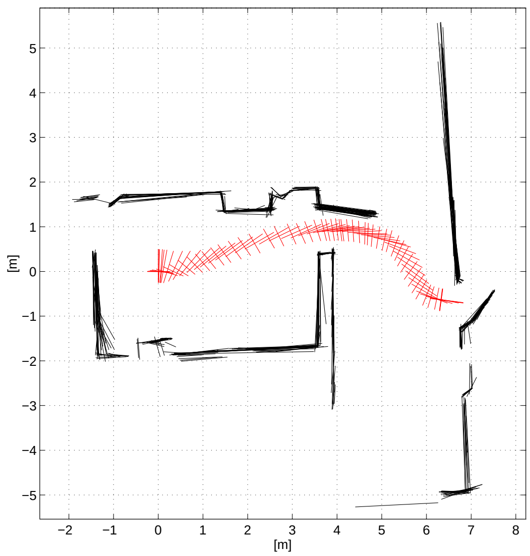

Figure 7. Self-localization experiment

8. Conclusion and future work

In this paper the problems of self-localization of a mobile robot and map building in an a-priori unknown environment are addressed. The self-localization system presented in this approach utilizes a panorama laser range finder as the environment sensor and is composed of the modules preprocessing, matching and recognition, localization and mapping. As could be shown from experiments executed in a real world office environment, the interaction of these components provides an accurate and robust estimate of the current robot's position in real-time.

Future work will focus on enhancing the map building mechanism in such way that the topological description associated with an edge will be based on a formalism which is on the one hand powerful enough to explain difficult topological relations and, on the other hand, can easily be transformed into robot's driving commands. Furthermore it will be examined if and how the catchment areas of the nodes stored in the map can be enlarged.

9. Acknowledgement

The work presented in this paper was supported by the Deutsche Forschungsgemeinschaft as a part of an interdisciplinary research project on “Information Processing in Autonomous Mobile Robots”(SFB331).

References

[1] R. E. Bellman and S. E. Dreyfus. *Applied Dynamic Programming*. Princeton University Press, 1962.

[2] J. Detlefsen and M. Rožmann. Determination of Position and Orientation of Autonomous Vehicles in Production Type Environments Using a 94 GHz Radar Sensor. In *Proc. 1994 Int. Symp. on Noise and Clutter Rejection (ISNCR)*, pages 633–638, Nov. 1994.

[3] T. Edlinger and G. Weiß. Exploration, Navigation and Self-Localization in an Autonomous Mobile Robot. In R. Dillmann, U. Rembold, and T. Lüth, editors, *11. Fachgespräch “Autonome Mobile Systeme”*, pages 142–151. Springer-Verlag, 1995.

[4] J. Forsberg, U. Larsson, Å. Wernersson, and P. Åhman. Navigation in Cluttered Rooms using a Range Measuring Laser and the Hough Transform. In *Intelligent Autonomous Systems-3 (IAS-3)*, 1993.

[5] G. H. Golub and C. F. V. Loan. *Matrix Computations*. The Johns Hopkins University Press, second edition, 1993.

[6] B. J. Kuipers and Y. Byun. A Robust, Qualitative Method for Robot Spatial Learning. In *Proc. 1988 Nat. Conf. on Artificial Intelligence*, volume 2, pages 774–779, 1988.

[7] M. J. Mataric. Navigating With a Rat Brain: A Neurobiologically-Inspired Model for Robot Spatial Representation. In *Proc. first Int. Conf. on Simulation of Adaptive Behavior, 1990 From Animals to Animats*, pages 169–175, 1990.

[8] H. Ney, D. Mergel, A. Noll, and A. Paeseler. A Data-Driven Organization of the Dynamic Programming Beam Search for Continuous Speech Recognition. In *Proc. 1987 IEEE Int. Conf. on Acoustics, Speech, and Signal Processing*, pages 833–836, 1987.

[9] T. Pavlidis and S. Horowitz. Segmentation of planar curves. *IEEE Transactions on Computers*, (C-23):860–870, 1974.

[10] S. J. Vestli, N. Tschichold-Gürman, and H. Andersson. Learning control and localisation of mobile robots. In P. Levi and T. Bräunl, editors, *12. Fachgespräch “Autonome Mobile Systeme”*, pages 202–213. Springer-Verlag, 1994.

[11] G. Weiß, C. Wetzler, and E. von Puttkamer. Keeping Track of Position and Orientation of Moving Indoor Systems by Correlation of Range-Finder Scans. In *Proc. 1994 IEEE Int. Conf. on Intelligent Robots and Systems (IROS ’94)*, pages 595–601, Sept. 1994.