Extraction of Natural Landmarks and Localization using Sonars

Olle Wijk* **and H.I. Christensen†**

S3 Automatic Control* and Autonomous Systems†
Royal Institute of Technology
SE-100 44 Stockholm
SWEDEN

Abstract. In this paper we present a new technique for on-line extraction of natural landmarks using ultra sonic sensors mounted on a mobile platform. By storing landmarks' positions in a reference map, we can easily achieve absolute localization in an indoor environment by doing matching between recently collected landmarks and the reference map. The matching procedure also takes advantage of compass readings to increase computational speed and make the performance more robust.

1 Introduction

There has been an abundance of research and development related to use of ultra-sonic sonars for local navigation in an in-door setting. A good overview can be found in [1] and [2]. It is however characteristic for much of this work that it assumes use of a priori defined map. Alternatively the methods assume that it is possible to locate linear structures [3] or regions of constant depth (RCD's) [2]. Relatively little has been performed on automatic acquisition of maps in semi-structured environments and subsequent use of such maps for navigation. Much of the work reported in this area is based on occupancy grids [4], where the map acquisition builds up a grid that subsequently can be used for matching using well defined structures [5, 6, 7] like corners, edges or walls, or through correlation based matching of the current map with the reference map [1]. For the methods based on well defined structures there is a need for post-processing of data to identify 'real' structures in the environment. In the occupancy grid based techniques there is no or little interpretation of the map and it will thus contain information about structures that are good reflectors and accidental/poor reflectors.
The approach presented in this paper is *not* grid map based. Instead we have chosen to acquire a reference map of the best sonar reflectors, termed landmarks, in the environment. The basic filtering tool we use for extracting the positions of these landmarks is *triangulation based fusion of sonar data* [8].
The reference map of landmarks is used in later missions for automatic estimation of ego position and orientation through matching against the current set of identified landmarks. The matching performed is in terms of finding the best possible affine map that transforms the current set of landmarks onto the reference map . The matching of features is similar in spirit to well known methods from image analysis, i.e., [9, 10].
The paper is outlined as follows. In section 2 the basic algorithm for natural landmark position extraction is described, whereupon section 3 explains how we can robustly match independently collected landmarks with each other and hence obtain absolute localization. Finally section 4 presents some of our experiments, which have been carried out in various room settings at our lab.

2 Landmark extraction

The on-line filtering technique presented in this section is divided into two layers. These layers are graphically illustrated in figure 1. The first layer, triangulation based fusion, provides the second layer with so called *triangulation points* denoted $(n_t, \hat{T})$. Here $\hat{T} = (\hat{x}_T, \hat{y}_T)$ is an estimate of a target position $T = (x_T, y_T)$ in the environment and n_t is the *triangulation value*, which reflects how many *triangulations* that have contributed to the target estimate $\hat{T}$. The higher the triangulation value n_t is, the better target estimate $\hat{T}$ we get. The basic triangulation principle is explained in figure 2, where we note that necessary input for the first filter

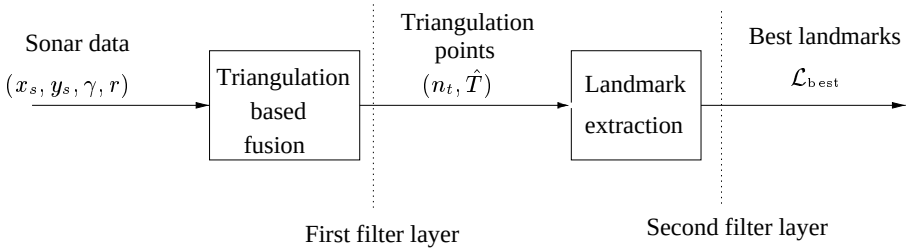

Fig. 1.: Summary of the on-line filtering process for extracting natural landmarks. Sonar data on the form (x_s, y_s, γ, r) are filtered to give triangulation points $(n_t, \hat{T})$, which in turn are filtered a second time to provide us with the "best" natural landmark positions in the environment.

layer are sonar readings on the form $R = (x_s, y_s, \gamma, r)$. Here (x_s, y_s) is the sensor position, γ the heading angle of the sensor and r the range reading. For a complete description of the first filter layer we refer the reader to [8].

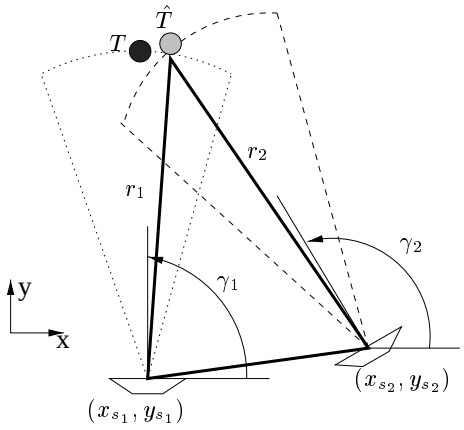

Fig. 2.: Basic triangulation principle. Given two sonar readings from the same target in the environment, a better estimate of the target position T is found by taking the intersection $\hat{T}$ of the beams corresponding to the two sonar readings. Here (x_{s_1}, y_{s_1}), (x_{s_2}, y_{s_2}) are sensor positions, r_1, r_2 range readings and γ_1, γ_2 heading directions. Usually $\hat{T} \neq T$ because of measurement errors in r_1 and r_2. In this situation the triangulation value n_t is equal to one since only "one triangulation" has contributed to the target estimate $\hat{T}$ (see marked triangle). For a better estimate of T one should fuse several triangulations [8].

The second layer of filtering is based on the "best" triangulation points from the first filtering stage, i.e. those points that belong to the following set:

$$\mathcal{D} = \left\{ (n_t, \hat{T}) \;\mid\; n_t >= n_{t_{\text{threshold}}} \right\}. \tag{1}$$

By discarding target points with a small hit-count n_t a significant number of "outliers" from the raw sonar data are removed. This is because it is unlikely that accidental reflectors will create high n_t-values. Also, with high n_t-values, we can assume the target position estimates to be quite accurate. In our implementation we use $n_{t_{\text{threshold}}} = 4$, which has turned out to be an appropriate threshold when working with a memory size of 7 sonar scans (1 scan = firing all 16 sonars on our Nomad 200 robot).
Let us now consider the second layer of filtering in detail. The first triangulation point $(n_{t_1}, \hat{T}_1) \in \mathcal{D}$ generates a landmark hypothesis $L_1 = (s_{h_1}, \hat{T}_1)$. Here s_{h_1} is a support number for the landmark hypothesis, which initially is set equal to unity. The second triangulation point $(n_{t_2}, \hat{T}_2) \in \mathcal{D}$ will result in one of the following two actions:

1. If $|\hat{T}_1 - \hat{T}_2| \leq \epsilon$ then the L_1 hypothesis is updated as $L_1 = (s_{h_1}, \hat{T}_1)$ where

$$\hat{T}_1 := \frac{1}{s_{h_1}+1}(s_{h_1}\hat{T}_1 + \hat{T}_2) \qquad \text{(recursive mean)}$$
$$s_{h_1} := s_{h_1} + 1.$$

   Here ϵ reflects our belief of position accuracy of the extracted landmarks. During experiments we found that an appropriate value of ϵ was $\epsilon = 10$ cm.

2. If $|\hat{T}_1 - \hat{T}_2| > \epsilon$ then $(n_{t_2}, \hat{T}_2)$ will trigger a competing landmark hypothesis $L_2 = (s_{h_2}, \hat{T}_2)$ with $s_{h_2} = 1$.

More generally if we have p competing landmark hypotheses $L_1, L_2, \ldots, L_p$, then the next triangulation point $(n_{t_l}, \hat{T}_l) \in \mathcal{D}$ $(l > p)$ will either update one of these hypotheses according to item 1 above, or generate a new hypothesis L_{p+1} according to item 2. In item 1 we note that the landmark position is allowed to drift by taking recursive mean. We reason this might be of help for getting a more precise position of the landmark as well as taking odometry errors into account.
Returning to the competing landmark hypotheses, a concern might be that the end-result is a large number of hypotheses, which would slow down an on-line updating algorithm. In practice this is however not a problem. When exploring a real livingroom setting (figure 4-7) with a Nomad 200 platform, equipped with 16 sonars, we found that the number of competing landmark hypotheses remained below 100.
As the on-line updating of competing landmark hypotheses evolves we keep track of the k best hypotheses with respect to their associated support number s_h. The set of these landmark positions we denote $\mathcal{L}_{\text{best}}$ and this set is later used for doing landmark matching. Experiments have shown that a reference map with twenty landmark positions ($k = 20$) is sufficient to obtain robust matching (see section 4).

3 Landmark matching

Consider a situation where a mobile platform, equipped with sonars and a compass, has collected two independent landmark sets $\mathcal{L}^{(1)}_{\text{best}} = \{L^{(1)}_1, L^{(1)}_2, \ldots L^{(1)}_{k_1}\}$ and $\mathcal{L}^{(2)}_{\text{best}} = \{L^{(2)}_1, L^{(2)}_2, \ldots L^{(2)}_{k_2}\}$ from the same room, though represented in different robot centered coordinate systems. We could imagine $\mathcal{L}^{(1)}_{\text{best}}$ being our reference map of landmarks, which is going to be matched against a recently collected set of landmarks $\mathcal{L}^{(2)}_{\text{best}}$. In figure 3 we can see a simple test example. The relation between the coordinate systems $(x^{(1)}, y^{(1)})$ and $(x^{(2)}, y^{(2)})$ will be an affine map involving a rotation point R_p, a rotation matrix $\mathbf{R}_m$ and a translation vector $\mathbf{t}$. In this section we will show how $\mathcal{L}^{(1)}_{\text{best}}$ and $\mathcal{L}^{(2)}_{\text{best}}$ can be used to estimate this affine map and hence obtain absolute localization.

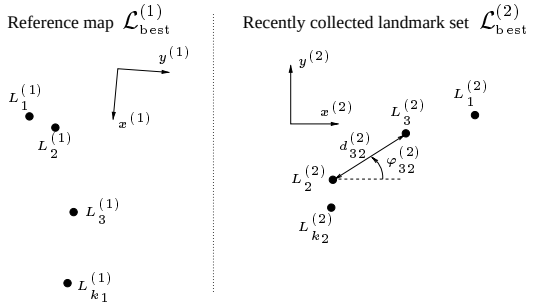

Fig. 3.: Two landmark sets from the same room have been collected in different coordinate systems. One of the sets is our reference map of landmarks while the other set has recently been collected with the mobile platform. How do we match these sets against each other and hence obtain absolute localization?

We assume that while the mobile platform is exploring the room, collecting landmarks, it is also sampling compass readings (relative to the current coordinate system). Taking the mean value of these readings gives the bearings $\bar{c}^{(1)}$ and $\bar{c}^{(2)}$ for the two runs. These bearings will later provide a start guess for the rotation matrix $\mathbf{R}_m$. The algorithm for matching the two landmark sets against each other is probably best understood by looking at some pseudo code:

/* Compute distance matrices from $\mathcal{L}^{(1)}_{\text{best}}$ and $\mathcal{L}^{(2)}_{\text{best}}$ */ (1)

$D^{(1)} = \left(d^{(1)}_{ij}\right) \qquad i,j \in [1,k_1]$

$D^{(2)} = \left(d^{(2)}_{lm}\right) \qquad l,m \in [1,k_2]$

/* Compute angle matrices from $\mathcal{L}^{(1)}_{\text{best}}$ and $\mathcal{L}^{(2)}_{\text{best}}$ */ (2)

$\phi^{(1)} = \left(\varphi^{(1)}_{ij}\right) \qquad i,j \in [1,k_1]$

$\phi^{(2)} = \left(\varphi^{(2)}_{lm}\right) \qquad l,m \in [1,k_2]$

/* Compute expected rotation angle */ (3)

$\varphi_{\text{exp\_rot}} = \bar{c}^{(2)} - \bar{c}^{(1)}$

/* Form landmark pairs in $\mathcal{L}^{(1)}_{\text{best}}$ and $\mathcal{L}^{(2)}_{\text{best}}$ respectively */ (4)

$\mathcal{P}_1 = \left\{ (L^{(1)}_i, L^{(1)}_j) \;\mid\; L^{(1)}_i, L^{(1)}_j \in \mathcal{L}^{(1)}_{\text{best}}, \quad j > i, \quad i,j \in [1,k_1] \right\}$

$\mathcal{P}_2 = \left\{ (L^{(2)}_l, L^{(2)}_m) \;\mid\; L^{(2)}_l, L^{(2)}_m \in \mathcal{L}^{(2)}_{\text{best}}, \quad l \neq m, \quad l,m \in [1,k_2] \right\}$

/* Matching loop... */ (5)

max_matches$= 0$

Loop over $\mathcal{P}_1$

  Loop over $\mathcal{P}_2$

    /* Assume that $(L^{(1)}_i, L^{(1)}_j) \leftrightarrow (L^{(2)}_l, L^{(2)}_m)$ */

    $\varphi_{\text{rot}} = \varphi^{(1)}_{ij} - \varphi^{(2)}_{lm}$

    $d = \mid d^{(1)}_{ij} - d^{(2)}_{lm} \mid$

    /* Check if the above assumption is worthwhile to consider */

    if $d < 2\epsilon$ *and angle_difference_between*$(\varphi_{rot}, \varphi_{exp\_rot}) < \alpha$ (*)

      /* Calculate affine map candidate $(R_p, \mathbf{R}_m, \mathbf{t})$ */ (6)

      $R_p = L^{(1)}_i$

      $\mathbf{R}_m = \begin{pmatrix} \cos(\varphi_{\text{rot}}) & \sin(\varphi_{\text{rot}}) \\ -\sin(\varphi_{\text{rot}}) & \cos(\varphi_{\text{rot}}) \end{pmatrix}$

      $\mathbf{t} = L^{(2)}_l - L^{(1)}_i$

      /* Compute transformed landmark set */

      $\mathcal{L}_{\text{temp}} = \left\{ L \;\mid\; L = (\, L' - \mathbf{t} - R_p)\mathbf{R}_m + R_p, \quad L' \in \mathcal{L}^{(2)}_{\text{best}} \right\}$

      /* Find number of landmarks in $\mathcal{L}^{(1)}_{\text{best}}$ which */ (7)

      /* have a neighbour in $\mathcal{L}_{\text{temp}}$ closer than 2ϵ */

      matches $=$ *nmb_of_matches_between*$(\mathcal{L}^{(1)}_{best}, \mathcal{L}_{temp})$

      /* Update best result */ (8)

      if matches$>$*max_matches*

        store$(R_p, \mathbf{R}_m, \mathbf{t})$

In words the pseudo code is explained in the following way. (1) First we form distance matrices $D^{(1)}$ and $D^{(2)}$ for the landmark sets $\mathcal{L}^{(1)}_{\text{best}}$ and $\mathcal{L}^{(2)}_{\text{best}}$ respectively. These matrices will of course be symmetric (see for example figure 3 and conclude that $d_{32} = d_{23}$). (2) Angle matrices $\phi^{(1)}$ and $\phi^{(2)}$ are then produced in a similar way, but these matrices will not be symmetric (see for example figure 3 and conclude that $\varphi_{23} = \pi + \varphi_{32}$). (3) From compass bearings $\bar{c}^{(1)}$ and $\bar{c}^{(2)}$ we can form an estimate of the rotation angle $\varphi_{\text{exp\_rot}}$ with which the two coordinate systems $(x^{(1)}, y^{(1)})$ and $(x^{(2)}, y^{(2)})$ differ. (4) Form landmark pair sets $\mathcal{P}_1$ and $\mathcal{P}_2$ from $\mathcal{L}^{(1)}_{\text{best}}$ and $\mathcal{L}^{(2)}_{\text{best}}$ respectively. (5) Matching loop. Loop over the sets $\mathcal{P}_1$ and $\mathcal{P}_2$. In the loop assumptions of type

$$(L^{(1)}_i, L^{(1)}_j) \leftrightarrow (L^{(2)}_l, L^{(2)}_m)$$

are made, meaning that pair $(L^{(1)}_i, L^{(1)}_j)$ in $\mathcal{L}^{(1)}_{\text{best}}$ corresponds to pair $(L^{(2)}_l, L^{(2)}_m)$ in $\mathcal{L}^{(2)}_{\text{best}}$. This assumption is treated as reasonable if we manage to pass the *if* sentence marked with a (*). Clearly the assumption is wrong if the distances $d^{(1)}_{ij}$ and $d^{(2)}_{lm}$ differ too much; more precisely they should not differ more than twice the uncertainty radius ϵ of a landmark. Also the calculated rotation angle φ_{rot} should not differ too much from the expected rotation angle $\varphi_{\text{exp\_rot}}$ obtained from compass readings. We use $\alpha = 45^\circ$ as we

have experienced the compass oscillating when the mobile platform is turning. (6) Calculate an affine map candidate $(R_p, \mathbf{R}_m, \mathbf{t})$ and use it for transforming the landmark set $\mathcal{L}^{(2)}_{\text{best}}$ into a temporary landmark set $\mathcal{L}_{\text{temp}}$. (7) If the affine map $(R_p, \mathbf{R}_m, \mathbf{t})$ is the one we are searching for, then the reference map $\mathcal{L}^{(1)}_{\text{best}}$ should almost overlap $\mathcal{L}_{\text{temp}}$, so we check how many landmarks in $\mathcal{L}^{(1)}_{\text{best}}$ that have a neighbour in $\mathcal{L}_{\text{temp}}$ that is closer than 2ϵ. (8) If the number of matches is greater than the maximum number of matches obtained so far, then store this transformation.

As a remark to (8) we can say that our practical implementation is slightly more advanced. We keep track of all affine maps $(R_p, \mathbf{R}_m, \mathbf{t})$ which give rise to *matches=max_matches*. The transformations are stored with respect to their rotation angle φ_{rot} and are given a support number s_m, initially set to unity. This number is increased with one unit if another transformation occurs with *matches=max_matches* and a rotation angle in the interval $[\varphi_{\text{rot}} - 1^\circ, \varphi_{\text{rot}} + 1^\circ]$. At the end we select the transformation with the highest support number s_m.
The algorithm's computational complexity is $O(k_1^2 k_2^2)$ if the (*) line is disregarded. This line however effectively sorts away a significant number of false matches (typically 98-99%) and reduces the complexity substantially. We experience a computation time of about one second when using a Ultra Sparc 1 with $k_1 = k_2 = 20$.

4 Experiments

The experiments presented in this section are taken from two different indoor settings, a livingroom (figures 4-7) and a corridor (figure 8). The livingroom is for obvious reasons more furnished than the corridor and hence contains more natural landmarks. We used a Nomad 200 platform equipped with 16 sonars to cover these indoor settings. Eleven runs were done in each setting. The 20 best landmarks' positions collected from the "zero run" was stored in a landmark reference map together with the mean value of the sampled compass readings. Before starting the "zero run" we marked out the robot position on the floor with tape. This position was the *zero position* (0,0) measured in reference map coordinates, and could later be used for checking position errors in the matching.
Between the runs 1-10 the robot was translated, rotated and reset in order to change the robot centered coordinate system. The robot was driven around in the room, and when it had collected some landmarks the technique described in section 3 was used to match these landmarks with the reference map. In most runs only parts of the room area were covered. In some of the runs people were present in the room or walking by. All this was done in order to test the robustness of the technique. After a matching had been performed we used the affine map $(R_p, \mathbf{R}_m, \mathbf{t})$ relating the reference map and the current set of landmarks to calculate the *zero position* in robot centered coordinates. The robot was then ordered to go to the *zero position* and stop there. Then we measured the distance from the robot to the *true zero position* marked on the floor and hence obtained the position error in our matching (this error of course include some odometry errors as well). Statistics from each of the runs 1-10 are provided in figures 11 and 14, and some cases are further illustrated graphically in figures 9, 10, 12, 13 and 15.
In the livingroom setting we see from the statistics in figure 11 that we have an average matching percent of 68%. The position error has a mean value of 8.7 cm and a standard deviation of 6 cm. All runs gave a correct matching, i.e. the chosen affine map $(R_p, \mathbf{R}_m, \mathbf{t})$ was the correct one. Looking more in detail at the second run which is graphically illustrated in figure 9, we see from the robot trajectory that the robot has not traveled a far distance to collect enough landmarks for a successful matching. In this run 10 landmarks were collected and 6 of them found a partner in the reference map. The third run in the livingroom, graphically presented in figure 10, show a situation where the robot has collected 20 landmarks of which 14 were successfully matched with the reference map. However, the more landmarks we allow the robot to collect, the higher risk we stand of getting some "ghost landmarks" which does not show consistency between the runs. If the number of "ghost landmarks" becomes large compared with the natural landmarks in the environment then the risk of an incorrect affine mapping $(R_p, \mathbf{R}_m, \mathbf{t})$ increases. In the livingroom there are plenty of natural landmarks, but in more smooth and symmetric environments "ghost landmarks" can cause problems.
An example of an environment which is symmetric and contains quite few natural landmarks is the corridor presented in figure 8. In this setting we carried out the same experiments as in the livingroom. From the statistics in figure 14 we conclude that the matching percent of landmarks is varying a bit more than in the livingroom case. In figures 12 and 13 we graphically present the successful matching in run 7 and 9 in the corridor. The robot has here only covered part of the corridor compared with what was covered when

collecting the reference map in the "zero run". Still the matching is successful. Concerning the doors in the corridor, they were open during the "zero run" but half open in the runs 5-7 and closed in the runs 8-10. This was done in order to check the robustness of the technique.
From the statistics in figure 14 we see that the position error is about the same as in the livingroom except for the last run where the error is as high as 50 cm. This case is graphically illustrated in figure 15 where we see that the error is elongated in the same direction as the corridor. The bad matching was caused by "ghost landmarks" along the walls. These landmarks does not show consistency between the runs and hence we would do better without them. An alternative would be to manually sort away "ghost landmarks" from the reference map, but since we want as little human interaction as possible when creating reference maps, we have chosen not to take this approach. Another alternative would be to use the fact that natural landmarks (in this case door posts and window corners) generally will have a higher associated support number s_h than "ghost landmarks" and hence some kind of weighted matching of landmarks could be of interest. This is an idea we have not investigated yet.
Summing up the experiments, they were successful in the sense that they all, except the 10:th run in the corridor, produced correct affine mappings relating the reference map and the sets of recently collected landmarks with an accuracy of about 5-15 cm. Hence absolute localization can in most cases be achieved with that accuracy.

5 Conclusions

In this paper we have presented a technique that on-line extracts natural landmarks from ultrasonic data as a mobile platform equipped with sonars is traversing the environment. It has also been explained how independently collected landmarks can be matched against each other to obtain absolute localization within an indoor setting. Typically we get a position accuracy of 5-15 cm. The technique is robust in indoor settings with lots of natural landmarks, but somewhat less robust in indoor settings that are symmetric and contain few natural landmarks.

6 Acknowledgment

The research has been carried out at the Centre for Autonomous Systems at the Royal Institute of Technology, and sponsored by the Swedish Foundation for Strategic Research.

References

1. J. Borenstein, H. R. Everett, and L. Feng. *Navigating Mobile robots.* A K Peters, Wellesley, Massachusetts, 1996.
2. J. J. Leonard and H. F. Durrant-Whyte. *Directed Sonar Sensing for Mobile Robot Navigation.* Kluwer Academic Publisher, Boston, 1992.
3. J. L. Crowley. World modeling and position estimation for a mobile robot using ultrasonic ranging. *IEEE Conference on Robotics and Automation*, 1989.
4. A. Elfes. Sonar-based real-world mapping and navigation. *IEEE Journal of Robotics and Automation*, RA-3(3):249–265, June 1987.
5. B. Schiele and J.L. Crowley. A comparison of position estimation techniques using occupancy grids. In *IEEE Conf. on Robotics and Automation*, pages 1628–1634, San Diego, CA, May 1994.
6. W.D. Rencken. Concurrent localization and map building for mobile robots using ultra-sonic sonars. In *IEEE 1993 Intl. Conf on Intelligent Robotics and Systems (IROS 93)*, pages 2192–2197, Yokohama, Japan, July 1993.
7. W.D. Rencken. Autonomous sonar navigation in indoor, unknown and unstructured environments. In *Intl. Conf on Intelligent Robots and Systems (IROS 94)*, pages 127–134, Munich, September 1994.
8. O. Wijk, P. Jensfelt, and H.I. Christensen. Triangulation based fusion of ultrasonic sensor data. In *IEEE Conf. on Robotics and Automation*, pages 3419–24, Leuven, May 1998. (In-press).
9. J. L. Crowley and P. Stelmaszyk. Measurement and integration of 3-d structures by tracking edge lines. In *Computer Vision - ECCV90*, pages 269–280. Springer-Verlag, 1990.
10. O. Faugeras. *Three-dimensional computer vision: A Geometric Viewpoint.* MIT Press, Cambridge, MA, 1993.

This article was processed using the TEX macro package with SIRS98 style

Fig. 4.: View 1 of the livingroom

Fig. 5.: View 2 of the livingroom.

Fig. 6.: View 3 of the livingroom

Fig. 7.: View 4 of the livingroom

Fig. 8.: View of the corridor. The runs were carried out between the marked lines in the figure.

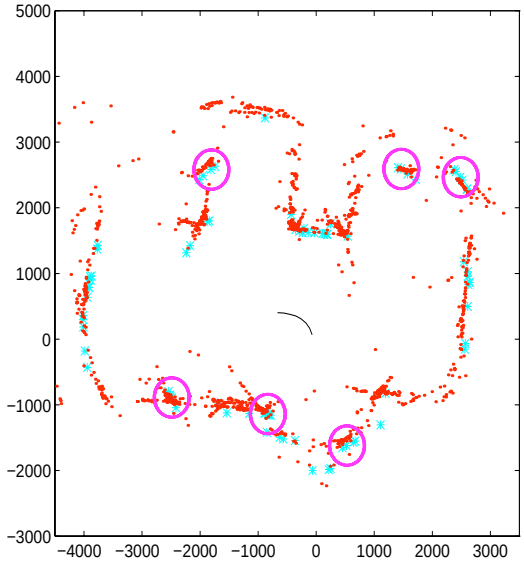

Fig. 9.: The 2nd run in the living room. The figure should be interpreted as follows. Triangulation points which were used to filter out the landmarks in the reference map (the "zero run") are drawn with dark dots. The bright stars (*) are triangulation points from the 2nd run used to filter out another set of landmarks. All triangulation points have been plotted in the same coordinate system for easy verification of the matching quality. Those landmarks that were successfully matched against each other are marked with circles. The robot trajectory in the 2nd run is shown in the figure, and it is seen that the robot has not traveled a long distance before doing this matching. The scales on the axes are given in mm.

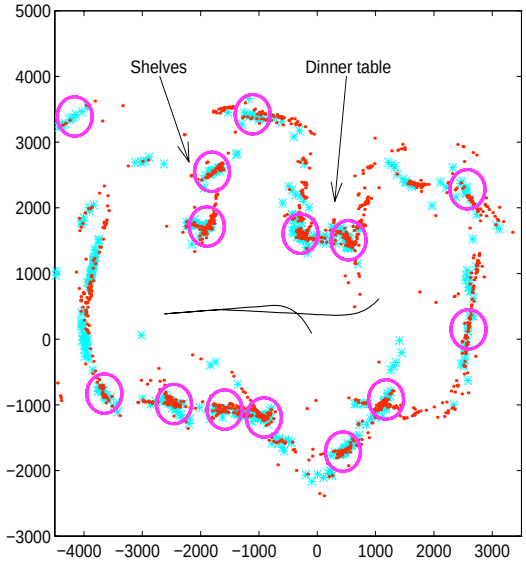

Fig. 10.: The 3rd run in the living room. The robot trajectory is here longer than in the 2nd run since we have allowed the robot to collect more landmarks. We note that a substantial part of the collected landmarks has found a partner in the reference map. For easier interpretation of the orientation of the map we have marked two objects, a dinner table and some shelves, which could be compared with the corresponding objects marked in figure 5.

Livingroom runs					
Run	k_1	k_2	# Matches	Matching percent	Position error
1	15	20	11	73.33 %	5 cm
2	10	20	6	60.00 %	15 cm
3	20	20	14	70.00 %	8 cm
4	13	20	10	76.92 %	9 cm
5	17	20	13	72.47 %	4 cm
6	14	20	10	71.43 %	17 cm
7	16	20	10	62.50 %	18 cm
8	20	20	13	65.00 %	2 cm
9	19	20	12	63.16 %	7 cm
10	20	20	13	65.00 %	2 cm
Average	16.4	20	11.2	67.98 %	8.7 cm

Fig. 11.: Statistics from the livingroom runs. Here k_1 is the number of landmarks collected from one of the runs 1-10, and k_2 the number of landmarks stored in the reference map from the "zero run". At each matching, the found affine map $(R_p, \mathbf{R}_m, \mathbf{t})$ relating the recently collected landmark set and the reference map was the one we were looking for.

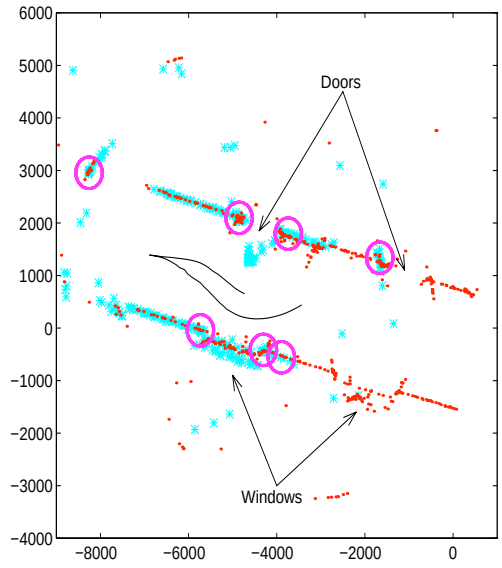

Fig. 12.: The 7th run in the corridor. The map should be interpreted in the same sense as described in figure 9. In this run the doors of the corridor were half open and only part of the corridor was covered by the robot. We see that the most important natural landmark positions, i.e. door post positions and window corners, have been matched correctly. The scales on the axes are given in mm.

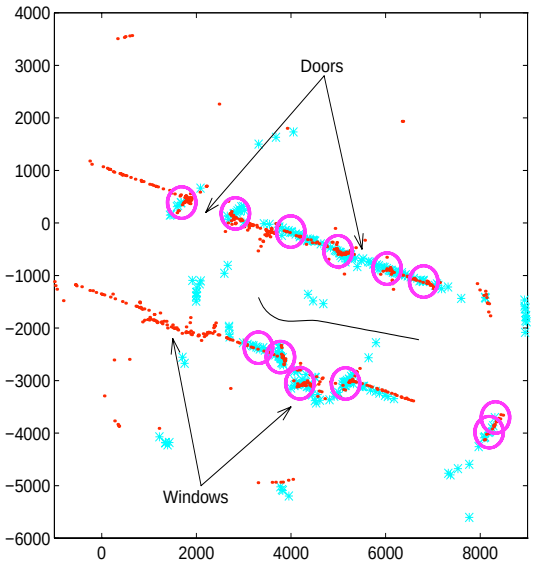

Fig. 13.: The 9th run in the corridor. In this run the doors of the corridor were closed and only part of the corridor was covered by the robot. We see that the most important natural landmark positions, i.e. door post positions and window corners, have been matched correctly. There are however also matchings between "ghost landmarks" along the walls. These landmarks occur because of the lack of natural landmarks in this environment and can cause trouble concerning localization along the direction of the corridor (see for instance data from the 10th run, which yielded a position error of 50 cm).

Corridor runs						
Run	k_1	k_2	# Matches	Matching percent	Position error	Doors were
1	15	20	12	80.00 %	5cm	wide open
2	10	20	7	70.00 %	2 cm	wide open
3	18	20	11	61.11 %	11 cm	wide open
4	14	20	11	78.57 %	4 cm	wide open
5	17	20	12	70.59 %	9 cm	half open
6	14	20	8	57.14 %	16 cm	half open
7	12	20	7	58.33 %	10 cm	half open
8	20	20	12	60.00 %	15 cm	closed
9	17	20	12	70.59 %	7 cm	closed
10	19	20	10	52.63 %	50 cm	closed
Average	15.6	20	10.2	65.90 %	12.9 cm	

Fig. 14.: Statistics from the corridor runs. Here k_1 is the number of landmarks collected from one of the runs 1-10, and k_2 the number of landmarks stored in the reference map from the "zero run". In order to test the robustness of the matching algorithm we let the doors in the corridor shift in position between the runs. The "zero run" was done with the doors in wide open position.

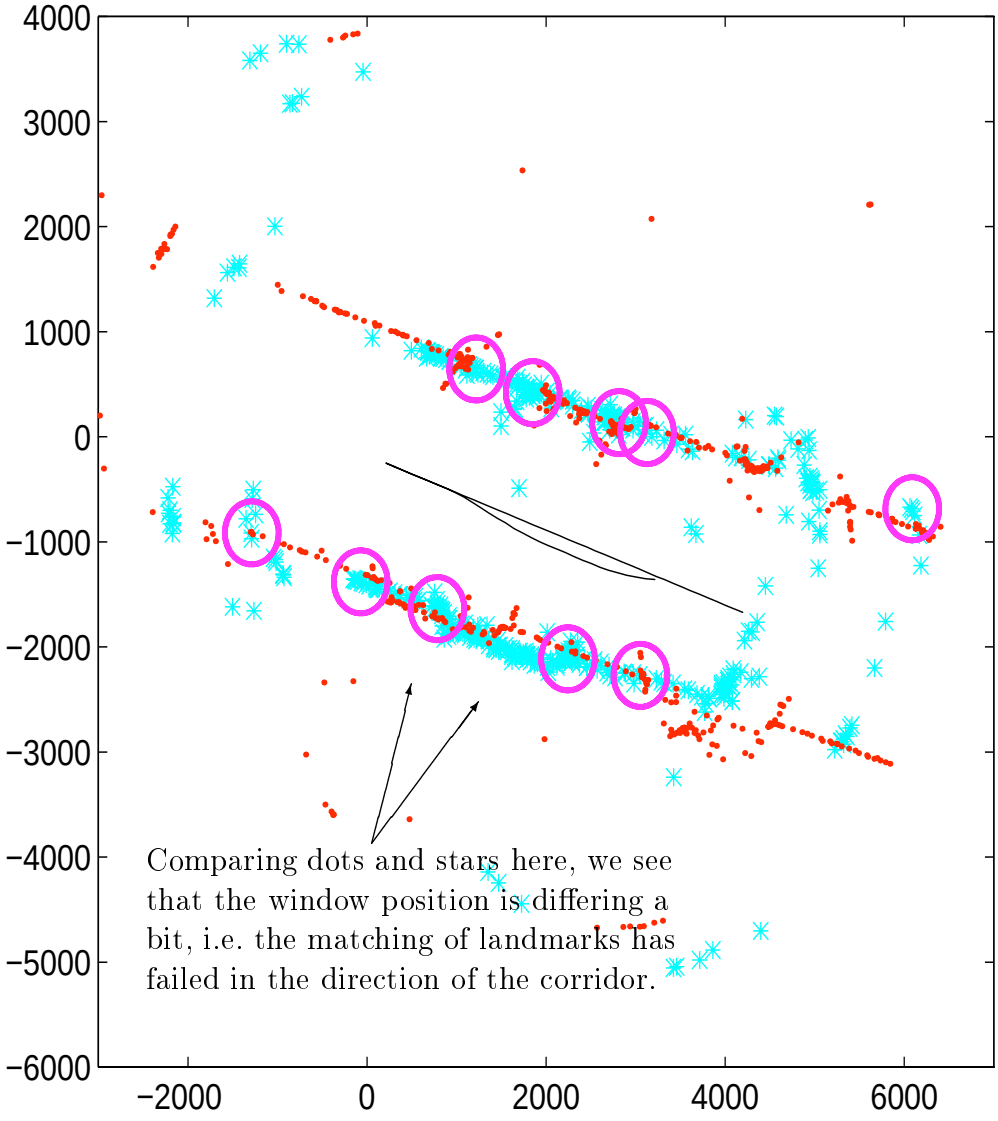

Fig. 15.: The 10th run in the corridor. The map should be interpreted in the same sense as described in figure 9. Here the matching of landmarks has failed. Comparing dark dots and bright stars in the graph it can, for instance, be seen that the window position is differing in the direction of the corridor. This case hints that symmetric environments with few natural landmarks and lots of "ghost landmarks" can cause trouble with the matching technique used in this paper. An alternative to get around this is to manually sort away "ghost landmarks" in the reference map which does not show consistency between the runs. In this case this would mean removing landmarks appearing along the corridor walls. This would decrease the risk of bad matchings. However, since we want as little human interaction as possible when creating reference maps, we have chosen not to take this approach.