A Flexible Exploration Framework for Map Building

Felix H. Wullschleger[†], Kai O. Arras[‡], Sjur J. Vestli[†]

[†]Institute of Robotics, Swiss Federal Institute of Technology (ETHZ), CH-8092 Zurich
[‡]Autonomous Systems Lab, Swiss Federal Institute of Technology (EPFL), CH-1015 Lausanne

Abstract

This paper presents a framework for exploration and incremental mapping of unknown environments. The framework allows for evaluation and comparison of different acquisition strategies. During exploration a visibility graph is constructed which holds correct topology information about the environment and provides a means for immediate planning in the partially known map. The framework has been implemented in simulation and on a real platform equipped with a 360 degree laser scanner, an algorithm for line and segment extraction and an extended Kalman filter for localization. Structured environments have been explored and mapped in a fully autonomous mode, simultaneously localizing the robot yielding results of satisfying precision. Limitations and problems of our implementation will be discussed as well.

1. Introduction

The burden to build practicable a priori maps by hand is one of the major motivations for autonomous map building. Others, not less important ones, are perceptual compatibility, i.e. acquisition and operation is done with the same sensory system, and environments dynamics, making a priori maps laborious to keep up-to-date.

For the purpose of map building three elementary questions need an answer.

(i) How is the world and the robot *represented*?
(ii) Which *formalism for integrating new information* is employed?
(iii) *Where* to acquire new information?

Several world representations are in use which are best separated by their degree of abstraction: From raw data maps with no abstraction, to metric maps with geometric features and their spatial dependencies, via topologic maps holding relationships between locally distinct locations up to semantic high-level descriptions of the world [5].

For the purpose of compact environment models and precise navigation, metric, feature-based maps are a good choice and will be employed here. This leads to the problem of representing and managing uncertain spatial relationships, constituting a globally referenced *stochastic map* [12]. The environmental entities of our map are line segments with their first order covariance estimate. Position estimation is simultaneously performed with an extended Kalman filter during exploration.

In the stochastic map, all features are referenced in a common world frame. The robot and the map entities are modeled as uncertain spatial relationships by estimating their first two moments in that frame. They are stacked into the estimated system state vector $\hat{X}$ and the estimated system covariance matrix Λ. The matrix Λ has a block-wise organization where the robot and each individual map feature has its associated covariance matrix in the diagonal. An important point is that Λ is not block-wise diagonal. The off-diagonal sub-matrices are the cross-covariances between the spatial relationships and encode their dependencies. It is important to maintain these cross-correlations as demonstrated in [7] and [3]. When inter-feature and robot-to-feature correlations are neglected, phenomena early leading to inconsistencies in the world model can be observed (see chapter 5). However, as pointed out in [7], maintaining these dependencies is not the solution to all problems when doing simultaneous localization and map building. The approach initially proposed in [12] has been shown to be sensitive to bias due to incorrect error models and inadequate error propagation [11]. Several propositions have been made to alleviate these difficulties, including suboptimal schemes [10][11], or improved techniques for error propagation in nonlinear systems [8].

The abovementioned problems and remedies are subject of on-going research and can be assigned to the second of the three elementary questions initially raised. We believe that approaching the problem from the viewpoint of the third issue, that is, by means of adequate exploration strategies, is an alternative worth to be investigated. The frequency and succession of observations and re-observations of features play a role for system convergence especially in the presence of bias. For this reason, an exploration framework is needed which allows, according to certain (possibly dynamically changing) criteria, to change, examine and compare different exploration strategies with respect to the quality of the resulting map. This paper presents first results in the attempt to push forward the limits of feature-based map building by a deliberative choice of exploration strategies.

Opposed to [13], where the frontier is the boundary between open space and uncharted territory, our approach primary focuses on the frontier between explored and unexplored area. The exploration is feature-based and, as described in [6], driven by unknown ends of features. The accuracy of the resulting map and the time taken to produce it are measures of exploration quality [9] and provide a means to compare different strategies.

For the implementation on the robot platform, some important simplifications have been done however. They are discussed in section 4.1, together with problems of the current implementation in section 5.

2. Segment-Based Maps

This work relies on the so called 2D assumption, that is, the environment can be sufficiently described in 2D since all percepted structures keep their form in the vertical dimension. The main idea where our exploration relies on, is that every horizontal slice of an obstacle has a closed contour (Figure 1). The goal is to build a closed chain of extracted features around each contour in the environment (for definition what a feature is see section 2.1). Open ends of a chain (in [6] called “non-terminal endpoints“), i.e. the end of a feature, where the successive one is still missing, indicate a frontier where new information can be gathered. The map is the collection of all chains of features.

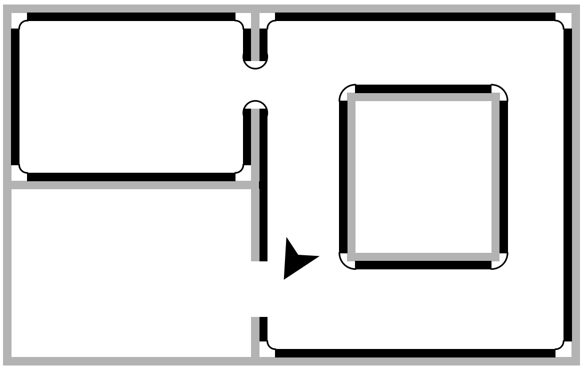

Figure 1: *Every object can be surrounded by a chain of features (black) applied to the shape. This simple scene shows two chains. One is closed, the other one is open. The robot is heading to such an open end. If all chains are closed the environment has been completely explored.*

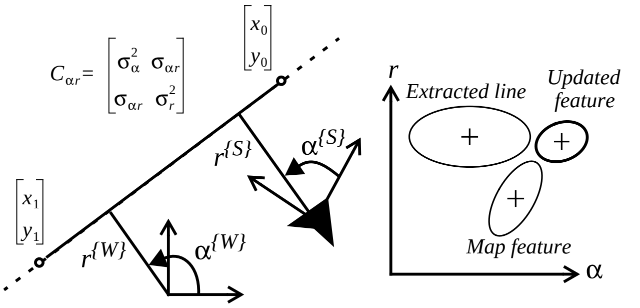
$C_{\alpha r} = \begin{bmatrix} \sigma_\alpha^2 & \sigma_{\alpha r} \\ \sigma_{\alpha r} & \sigma_r^2 \end{bmatrix}$
$\begin{bmatrix} x_0 \\ y_0 \end{bmatrix}$
$\begin{bmatrix} x_1 \\ y_1 \end{bmatrix}$
$r^{\{S\}}$
$\alpha^{\{S\}}$
$r^{\{W\}}$
$\alpha^{\{W\}}$
r
Extracted line
Updated feature
+
+
+
Map feature
α

Figure 2: *The representation of a line is given as a Gaussian distribution in model space (right) with heading* α *and distance* r *as the first moments. Ellipses are isoprobability contours showing feature position at a certain probability level. All operations like matching, fusing and update are done in this space. To get line segments the Cartesian xy-coordinates of the two endpoints are stored as well.*

2.1 Feature Representation and Handling

The line extraction method is taken from [1]. This algorithm delivers lines and segments together with their first order covariance estimate. The distinction of lines and segment is done as follows: A line is represented according to the Hessian model

$$\rho \cdot \cos(\varphi - \alpha) - r = 0 \tag{1}$$

where (ρ, φ) is the raw measurement in polar coordinates and (α, r) the model parameters. They come along with their second moments which hold the propagated uncertainty from the raw data level. Segments have a four dimensional representation. Either by the Cartesian coordinates of their endpoints or by a position, an inclination and a length [3]. In a hierarchical order, segments are below lines since they lie on a line (the supporting line) and thus have the same (α, r)-pair. In all comparisons, matchings and updates of map entities only pairs of (α, r) with their covariance matrix are used. Although this representation provides consistent treatment of all feature model parameters and thus permits ‘blind’ comparison and manipulation of features, this choice is a subpotimal one, as exposed in chapter 5.2.

The problem with segments as elements for comparison is that it is difficult to obtain physically based uncertainty models in all four dimensions. For example, the length of a segment is noisy. For consistency reasons with our statistical formalism, we would like to have an uncertainty model of that noise. When extracting segments from laser scanner data, their endpoints typically exhibit a variability which spans multiple raw points at the segment extremities. For a true uncertainty model it would be necessary to incorporate the segmentation criterion of the extraction algorithm since it classifies a raw measurement point as a model inlier or a model outlier. It is therefore not sufficient to consider the angular noise of the current endpoints for a model of segment length uncertainty since it yields too optimistic estimates. This shortcoming might not be important for map building while in the localization case the length parameter should be excluded from the robot pose updates.

When matching lines, their squared Mahalanobis distance is used. An extracted line is assigned to its best matching analogue in the map if it complies with a given level chosen from a χ_2^2 distribution. Matching segments require an additional criterion. For example, walls at the left and right of a door may have the same location in model space, but fusing them would, incorrectly, close the door. To prevent this, we compare the minimal distance between segments against an Euclidian threshold.

Feature update is done by determining the weighted average x_w, C_w of both estimated lines $\hat{x}_i = (\hat{\alpha}, \hat{r})_i$, $C_i = E[(x_i - \hat{x}_i)(x_i - \hat{x}_i)^T]$ with $i = 1, 2$ in model space according to the multivariate weighted mean

$$x_w = C_w \sum C_i^{-1} x_i \qquad (2)$$

$$C_w^{-1} = \sum C_i^{-1} \qquad (3)$$

which is a non-recursive form of the Kalman filter with a measurement model equalling identity. Since all operations on features like matching and fusion are done in (α, r)-space, a rule is required to update the segment endpoints. This is performed in the environment space where these points are orthogonally projected on the new line x_w, C_w.

2.2 Integrating New Information

Simultaneous localization and map building is performed by separating robot and map update using the relocation-fusion scheme from [11]. This strategy updates the robot position before integrating the new observation into the map (steps 2–4 below). Like this, the influence of robot position bias commonly introduced by odometry has reduced influence onto the map. The relocation-fusion scheme is suboptimal (there is a loss of information) but has been observed to increase map quality and stability, and is compatible with the simplifications of our current implementation (see chapter 4.1). A system update consists of seven steps:

1. *Extract* features from the sensory input
2. *Match* observed features to the best fitting equivalent from the map
3. *Localize* the robot with matched features
4. *Match* features with their re-prediction based on the new robot pose.
5. *Update* map features with their matched re-observation and *store* unmatched features as new ones
6. *Initialize* a new chain for each new segment
7. *Connect* open ends of chains if their Euclidian distance falls below a certain threshold.

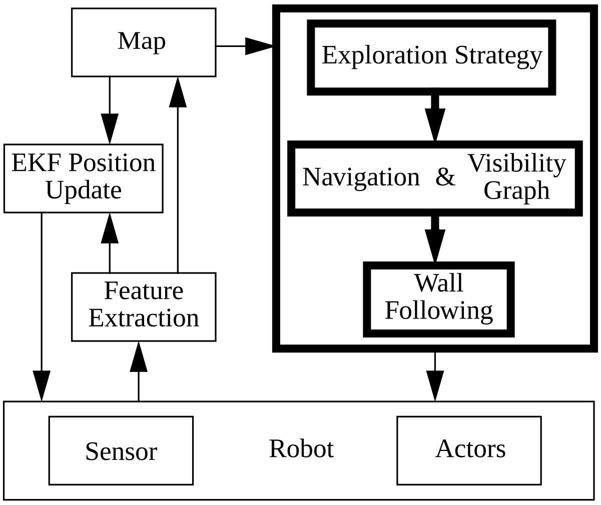

Figure 3: *Flow diagram of exploration framework, robot and mapping components. Functional blocks with thick borders constitute the exploration framework and are dicussed in the text. It transforms feature information successively into executable robot motion commands.*

3. The Exploration Framework

In order to get a scalable and transparent architecture of the framework envisaged, we take advantage of recursion. The job of map building is split into several layers. The top layer manages the entire task at a high degree of abstraction, whereas the lower layers get subgoals with increasing respect to reality. Figure 3 shows the framework including the main data flows. Each layer is discussed in the following chapters.

3.1 Wall Following Layer

The wall following component has to:

- *Follow* a given wall at defined distance. If the end of the wall is reached, the robot continues to drive around this endpoint holding the same distance until no collision-free motion is possible anymore.
- *Change* to the opposite driving direction if the follow-operation is no longer possible. Together with the above behaviour, this leads to a ping-pong movement along the wall making it irrelevant in which direction the robot started to follow the given wall. In favour of more efficient exploration an additional direction information could be passed to this layer.
- *Approach* a wall from an arbitrary direction. Used to change the actual wall to be followed.

The only input of this layer is the actual wall the robot has

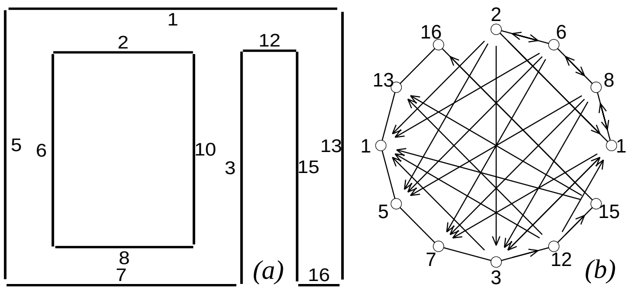

Figure 4: *A simple environment consisting of one cycle with a dead end (a) and a partial visibility graph (b). Peripheral connections between nodes denote direct neighbourhood in a chain (i.e. 2-6-8-10) whereas arrows indicate the possibility to observe a wall from the other. When the robot is following e.g. wall 8 and would like to get to wall 16, it can switch to wall 3, follow around the corner to wall 12, then wall 15 where it will finally see the goal.*

to follow. Its output are motion commands for a velocity or position controller.

3.2 Navigation Layer

The navigation component adds the capability of moving the robot from one place in the (partially known) map to another. The input of this layer is a wall representing the goal, its output is sequence of walls to be followed.

A representation for modeling the environment topology is sought which allows for global planning. For this, a visibility graph is used (figure 4). The visibility graph consists in nodes, unidirectional and bidirectional edges between the nodes. Unlike most other works where the term *visibility* refers to ‘from one place in free space to another‘, e.g. onto a Voronoi Graph, our approach is similar to [4]. The graph consists in the map features and their visibility relation. Each line in the feature based map appears as a node in the graph (numbered circles in figure 4 b). An unidirectional edge between two nodes indicates the possibility to observe a wall from an other one. Applying a conservative attitude, the conclusion that visibility is valid also in the opposite direction is not made. A bidirectional edge is registered when two walls turn out to be adjacent and with that are physically attainable in both directions. This is the case with chain 2-6-8-10 in figure 4.

The task of global path planning is then a well known search algorithm for this type of graphs. Once a node sequence is planned, it remains to pass its elements to the wall following layer. Due to the properties of the this layer, a single command is sufficient to approach, follow and drive around a wall.

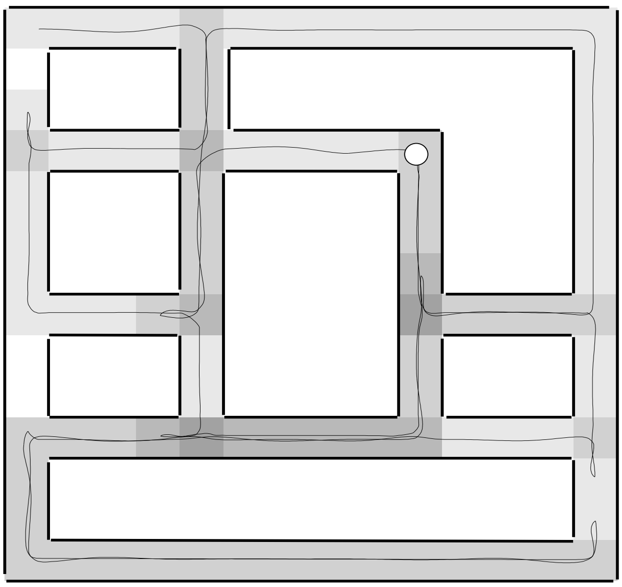

Figure 5: *The ‘next to actual position’ strategy. Shading indicates traversal frequency. The initial position is marked with a circle. During the whole exploration the robot passes there only once. The exploration behaviour is straight forward, comparable to a depth-first search.*

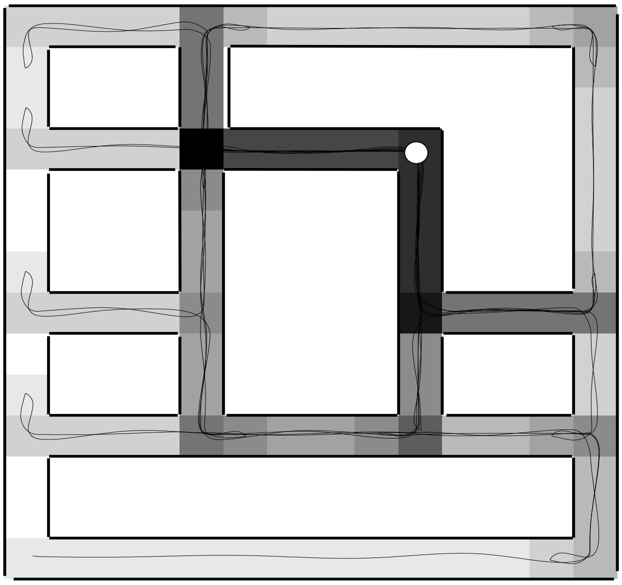

Figure 6: *The ‘next to start position’ strategy causes a path that is about three times longer than the ‘next to actual position’ path from figure 5 and results in a pendulum-like exploration behaviour around the starting point. Dark shaded cells indicate often traversed areas. The motivation of this strategy is to slow down the growth of accumulated uncertainty of features and robot at the extremities of the charted terrain by regularly passing the most certain region around the start point.*

3.3 Exploration Layer

The exploration strategy is situated in the top layer. No command input is needed for that component. Its output is the goal-wall to where the planning and navigation layer shall bring the robot.

The task of exploration is accomplished by choosing a feature that is currently the end of a chain and passing it to the navigation layer. When the robot approaches unknown territory, new information will automatically influence the map. The explorer continuously checks whether the actual feature is still interesting (i.e. that is an end of a chain) or not. The environment is completely explored, in the sense of this method, when all boundaries are closed.

There are two questions that mainly determine the behaviour of the exploration strategy: *When* to choose *which* feature for exploration? For the first question, the following rule was used:

- Abandon the actual feature if it is no longer the first or last one in the chain to be explored. This condition can be extracted directly from the map, no additional inference is needed.

For the second question, three possible strategies shall be given:

- Choose the feature with the open end that is *closest to the actual robot position* (figure 5). This leads to a depth-first search behaviour resulting in a fast exploration (further referred to as ‘next to actual position’).
- A more restrictive strategy is to decide for the *next open end in the actual chain*. This leads to a behaviour which closes a chain completely before beginning a new one. A very similar method is used in [6].
- Choose the feature with the open end *closest to the start point* (figure 6). This yields a breadth-first search behaviour generating a pendulum-like exploration around the start position. The initial position has an important property: It is the place where the robot had its lowest positional uncertainty and is therefore surrounded by the best known features in the map (further be referred to as ‘next to start position’).

The idea of the third strategy is to traverse well known area for accurately relocalizing the vehicle before starting a new attempt into unknown terrain. Further strategies where decision are made on the basis of feature or robot uncertainties is subject of current work.

3.4 Simulation

For the purpose of evaluating different exploration strategies we developed a simulation environment. A map drawn by hand can be loaded and explored. Errorless motion and perception (i.e. perfect odometry and line extraction) allow to easily develop strategies but do not provide quantitative measures of map quality. See figure 6 and figure 5 where in two simulation runs, the same environment has been explored by two strategies previously described.

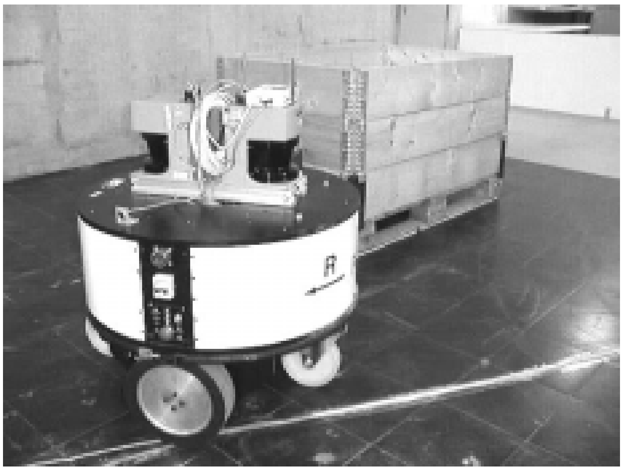

Figure 7: *The robot which was used in the experiments. It is equipped with two Sick LMS200 laserscanners covering 360°. The robot is a VME based system, currently carrying a PowerPC@100MHz. It runs the deadline-driven real-time operation system XOberon. The exploration framework is implementable such that the system could execute the task in a fully autonomous mode.*

4. Implementation

For the real-world experiments a differential drive robot depicted in figure 7 was used. It is equipped with a VME card carrying a PowerPC at 100 MHz. Two SICK LMS200 laser range finders covering together 360° with a resolution of 2° and a radial error of measurement less than ±20mm were used. All range values beyond 4 m were rejected. The entire code was written in XOberon, a deadline-driven hard real-time operating system developed at Institute of Robotics, ETH Zurich [2]. Figure 7 shows the robot in front of an obstacle placed inside free space.

4.1 Simplifications

An important simplification of the current implementation originates from the exclusive use of line segments as environment features. The environments in figure 8, figure 9 and figure 10 c are modified such as to have a degree of structuredness compatible with this constrain. Otherwise a chain of features could not have been closed. This loss of generality is neither imposed by the stochastic map which is strictly general with respect to the feature type nor by our exploration framework which principally works also in en-

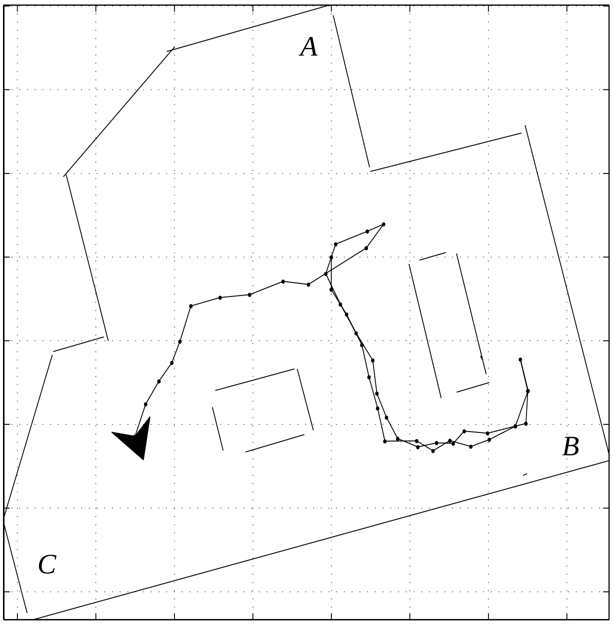

Figure 8: *Cyclic environment autonomously explored and mapped by the real robot with our simplest strategy 'next to actual position'. The distances between the corners marked with A, B and C have been measured and compared to those from the map (table 1). The grid equals one meter, like in all other maps.*

vironments with less frequent feature appearance. This simplification allows us to disregard a treatment of raw data like collision avoidance.

Although earlier work has demonstrated the need of maintaining cross-correlations in the stochastic map – [7] with simulations, and [3] with experiments on a real platform –, the current implementation neglects any cross-correlations between features and between robot and features yielding a blockwise diagonal system covariance matrix Λ. As already mentioned in chapter 2.2, concurrent localization and map building is performed by separating robot pose update and map update according to the relocation-fusion scheme from [11] (where all cross-correlations are maintained). The absence of cross-correlations lead to problems which confirm the results in [7] and [3]. See chapter 5 for their discussion.

4.2 Results

In the first experiment a cyclic 7m x 7m environment has been autonomously explored with our simplest strategy 'next to actual position' (figure 8). Map update and robot localization were done at each step on the drawn trajectory. The complete length of the exploration path is 33 meter. Three corners *A*, *B* and *C* are marked and their distances were measured by hand and taken from the resulting map. The map shows no significant mismatch when taking a look at table 1. Figure 9 shows a sequence of exploration steps of a 12m x 9m environment which has been explored with the same strategy.

	A to B	A to C	B to C
Measured by hand	6.50 m	8.35 m	7.60 m
Map	6.50 m	8.30 m	7.65 m

Table 1: *Distances of the marked corners in figure 8. Values rounded to 5 cm.*

One of the most difficult problems in map building in cyclic large environments is to correctly close loops. They represent the crucial situation of a very uncertain robot, possibly suffering from unintentionally injected position bias, which is confronted with an already visited area of relative high confidence. When integrating the local map into the global one, ambiguous matching situations are likely to occur with the danger of system state divergence due to incorrect pairings. In view of the ignorance of any cross-correlations, also regarding the simulation results in [7] and the suboptimal feature representation, we rate it as surprising that, as figure 9 shows, the robot was able to match the local map correctly after a 16 meter path around the big obstacle. We believe that the precision of the laserscanners employed and a line extraction method furnishing accurate feature estimates have the predominant contributions to the quality of these results.

5. Limitations and Problems

5.1 Map Inconsistencies and Nonconservative Estimates

The effect of 'positive feedback' between recorded features and the robot position has been observed and is illustrated in figure 10 and figure 11. With a stationary robot successively re-observing, matching and updating already mapped features, the uncertainty of robot and features converge to zero. This phenomenon is caused by the absence of cross-correlation between robot and features since then, each observation is interpreted as new and independent information on the global feature position which is not the case. As a consequence, system update yields nonconservative, 'optimistic', estimates of robot and feature uncertainties. When attempting to match extracted lines to their charted representation in the map, their squared Mahalanobis distance becomes too great and they are incorrectly recorded as new features producing thus inconsistencies in the map. Such inconsistencies are visible in the map of figure 9 (at top border, small segment in v-shaped structure) and figure 10 c.

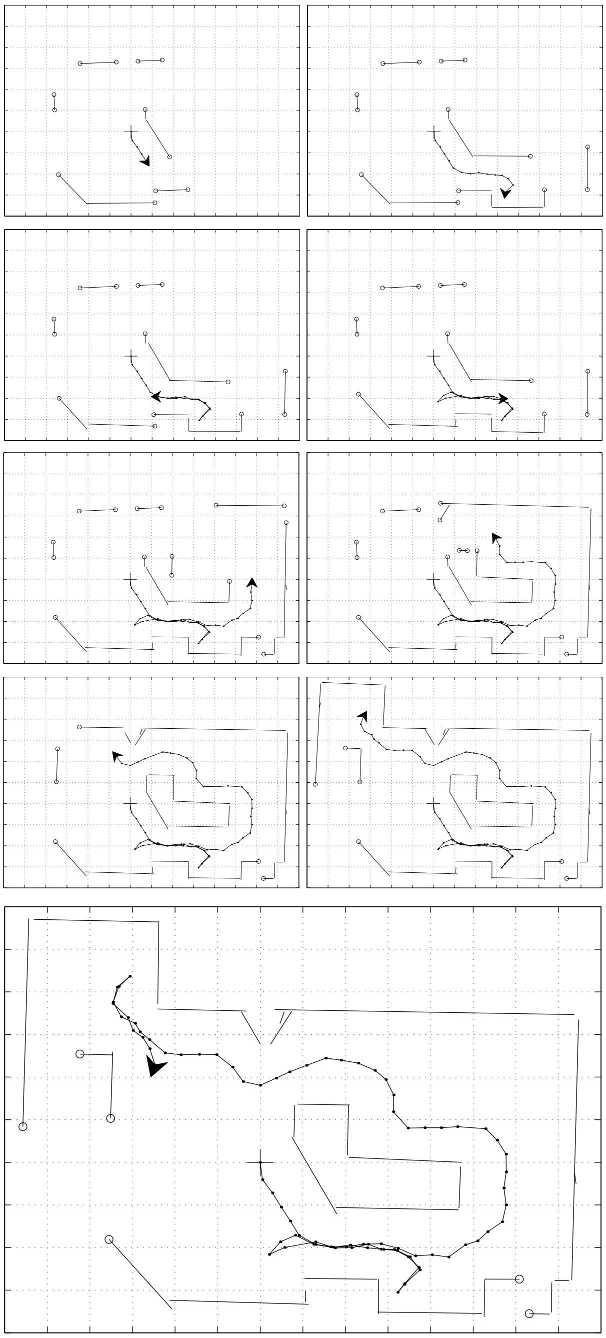

Figure 9: *This 12m x 9m environment was autonomously explored also by the ‘next to actual position’ strategy. Each open end of a chain is marked with a circle. As the sequence of pictures show, the robot was able to successfully close the chain around the big obstacle and to correctly integrate the local map into the global one. The distance error of the two most afar corners in the map, top right to bottom left, has been measured and is less than 20 cm.*

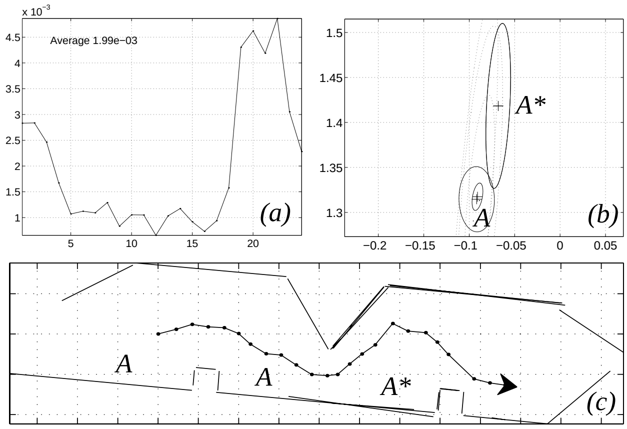

Figure 10: *As a consequence of neglecting cross-covariances, optimistic estimates of robot and feature uncertainties are produced. This can be seen e.g. in the trace of the robot pose covariance matrix (a). It is incorrect that the trace becomes inferior than its initial value. The environment (c) contains multiple entries for the same wall. The newly observed segment A* should be assigned to the supporting line A. The problem can also be visualized in the model space ((b), situation at step 9 of the exploration path in (c)). Wall A appears already as a small ellipse, too small such that A* is matched and fused correctly.*

5.2 Feature Representation

The uncertainty propagation in the line model from (1) is not invariant against frame transformations. The linearized measurement relationship for $\sigma_{rr}^{\{S\}}$ $(= \sigma_r^{2^{\{S\}}})$ with respect to $\alpha^{\{W\}}$ makes this evident. $(\hat{x}_k, \hat{y}_k, \hat{\theta}_k)$ is the current robot pose estimate at time index k, $(\alpha, r)^{\{X\}}$ the line parameters in the coordinate system $\{X\}$, where S denotes sensor frame, W world frame, and $h(.)$ the measurement model of a line also used in Kalman filter measurement prediction.

$$\sigma_{rr}^{\{S\}} \sim \frac{\partial h_r}{\partial \alpha^W} = \hat{x}_k \cdot \sin\alpha^{\{W\}} - \hat{y}_k \cdot \cos\alpha^{\{W\}} - \frac{\partial h_r}{\partial \hat{\theta}_k} \qquad (4)$$

The right hand side of the relationship (4) contains the robot pose in world coordinates thus incorrectly magnifying $\sigma_{rr}^{\{S\}}$ with the current robot position estimates $\hat{x}_k, \hat{y}_k$

6. Conclusions and Future Work

This paper presented an exploration framework for evaluation and comparison of different acquisition strategies for the purpose of autonomous incremental map making. The entire framework is implementable such that a fully autonomous system can execute the task. No off-board, off-line or post-processing procedures are required to get a practical environment model. Experiments in simulation and on a

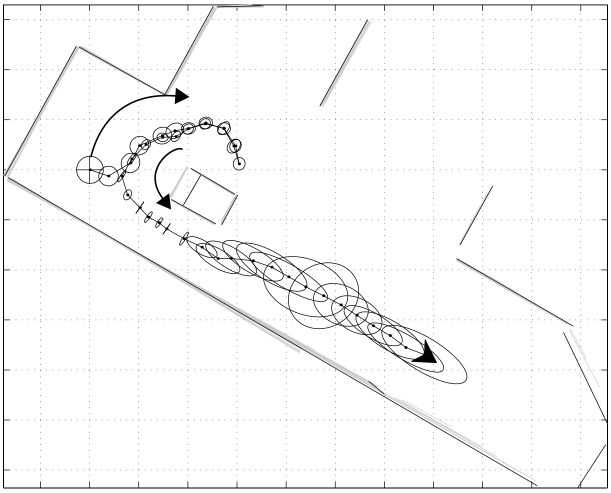

Figure 11: *In this example the robot starts at the circle with a cross and an uncertainty unequal to zero. Although it shows the desired effect of relatively decreasing uncertainty when repassing the same place, the values fall below the initial uncertainty which is unrealistic.*

real platform were carried out. Equipped with a 360° laser-scanner, an algorithm for line and segment extraction and an extended Kalman filter for localization, structured environments have been explored and mapped. During exploration a visibility graph is constructed which allows for immediate planning in the partially known map. First results were presented producing maps of encouraging quality. Limitations and problems of the current implementation were discussed.

The problems encountered and described in chapter 5 mark the most urgent points for improvements. The stochastic map shall firstly be implemented in its initial rigorous form under account of all cross-correlations [12]. Choosing a more appropriate feature representation which is compatible with the statistical formalism is the second question to be envisaged. It is further planned to develop additional exploration strategies and to examine the benefit of adaptive on-the-fly strategy changes. Finally the step to semi-structured, unmodified environments is to be made.

7. References

[1] Arras K.O., Siegwart R.Y.,"Feature Extraction and Scene Interpretation for Map-Based Navigation and Map-Building", Proc. of SPIE, Mobile Robotics XII, Vol. 3210, p.42-53, 1997.

[2] Brega R., “A Real-Time Opertaing System designed for Predictability and Run-Time Safety“, Proc. of the 4th Int. Conf. on Motion and Vibration Control (MOVIC’98), Zürich, August, 25-28, 1998, Vol. 1, pp. 379-384.

[3] Castellanos J.A., Tardos J.D., Schmidt G., “Building a Global Map of the Environment of a Mobile Robot: The Importance of Correlations“, Proc. 1997 IEEE Conf. on Robotics and Automation, Albuquerque, New Mexico, 1997.

[4] Cleary M.E., Crisman J.D., “Canonical Targets for Mobile Robot Control by Deictic Visual Servoing“, Proc. 1996 IEEE Conf. on Robotics and Automation, Minneapolis, Minnesota, 1996.

[5] Dudek G., Freedman P., Hadjres S. “Mapping in Unknown Graph-Like Worlds”, Journal of Robotic Systems 13(8), 539-559, 1996.

[6] Dudek G., Freedman P., Rekleitis I.M., “Just-in-time sensing: efficiently combining sonar and laser range data for exploring unknown worlds“, Proc.1996 IEEE Conf. Rob. Autom., Minneapolis, Minnesota, 1996.

[7] Hébert P., Betgé-Brezetz S., Chatila R., "Probabilistic Map Learning: Necessity and Difficulties," Int. Workshop on Reasoning with Uncertainty in Robotics, Amsterdam, Netherlands, 1995.

[8] Julier S.J., Uhlmann J.K., Durrant-Whyte H.F., "A New Approach for Filtering Nonlinear Systems," Proc. of the American Control Conference, Seattle, USA, 1995.

[9] Lee D., Recce M.A.A., “Quantitative Evaluation of the Exploration Strategies of a Mobile Robot”, International Journal of Robotics Research, vol.16, no.4, August 1997, p. 413-47.

[10] Leonard J.J., Durrant-Whyte H.F., Directed Sonar Sensing for Mobile Robot Navigation: Kluwer Academic Publishers, 1992.

[11] Moutarlier P., Chatila R., "Stochastic Multisensory Data Fusion For Mobile Robot Location And Environment Modelling," Proc. of the 5th International Symposium of Robotics Research, pp. 207-16, 1989.

[12] Smith R., Self M., Cheeseman P., "Estimating Uncertain Spatial Relationships in Robotics," in Autonomous Robot Vehicles, I. J. Cox and G. T. Wilfong (eds.), Springer-Verlag, 1990, pp. 167-93.

[13] Yamauchi B., Schultz A., Adams W., “Mobile Robot Exploration and Map-Building with Continuous Localization“, Proc. of the 1998 IEEE Int. Conf. on Robotics and Automation, Leuven, Belgium, May 1998.