Gold-fish SLAM: An application of SLAM to localize AGVs

Henrik Andreasson, Abdelbaki Bouguerra, Björn Åstrand and Thorsteinn Rögnvaldsson

Abstract The main focus of this paper is to present a case study of a SLAM solution for Automated Guided Vehicles (AGVs) operating in real-world industrial environments. The studied solution, called Gold-fish SLAM, was implemented to provide localization estimates in dynamic industrial environments, where there are static landmarks that are only rarely perceived by the AGVs. The main idea of Gold-fish SLAM is to consider the goods that enter and leave the environment as temporary landmarks that can be used in combination with the rarely seen static landmarks to compute online estimates of AGV poses. The solution is tested and verified in a factory of paper using an eight ton diesel-truck retrofitted with an AGV control system running at speeds up to 3 meters per second. The paper includes also a general discussion on how SLAM can be used in industrial applications with AGVs.

1 Introduction

Simultaneous localization and mapping (SLAM) have been a main research topic in mobile robotics [3]. SLAM algorithms can be run either online or offline. An online SLAM algorithm computes a robot pose estimate at runtime while at the same time computing a map of the environment. Offline algorithm on the other hand operate on previously recorded sensor data.

Despite the abundance of various SLAM approaches, the number of reported real world applications for online SLAM methods is small [5]. offline SLAM, also denoted surveying, is on the other hand used in many applications, for example, for creating a map of reflectors in a factory environment [1]. Another more recent

Henrik Andreasson and Abdelbaki Bouguerra
AASS, Dept. of Technology, Örebro University, Sweden, e-mail: firstname.lastname@oru.se
Björn Åstrand and Thorsteinn Rögnvaldsson
Dept. of Computing and Informatics, University of Halmstad, Sweden, e-mail: firstname.lastname@hh.se

example is reported in [7] where autonomous driving in a parking garage was made possible by first creating a 3D map.

In this paper we address a different setup where there exists a predefined map with static landmarks that is used at run-time in combination with an online dynamic SLAM approach to localize vehicles in real industrial environments. The main contribution lies in using dynamic features of the environment to compute reliable estimates even when the static features are not seen for longer periods of times. The dynamic features that are utilized in this work are basically goods manipulated by the vehicle. The approach has been implemented and tested on an eight tons diesel forklift providing pose estimates used by the on board AGV controller to smoothly and reliably follow predefined paths at speeds of up to 3 m/s.

1.1 Applications of online SLAM

The main purpose with online SLAM is to determine a localization estimate within a incrementally generated map. This estimate (and map) could either be provided to a human operator or to an autonomous agent. One application area which requires online SLAM is exploration of unknown environments such as in urban search and rescue or surveillance tasks [8]. For a human operator an online and up to date map is a valuable source of information to tele-operate a vehicle in a safe manner [16].

From an industrial perspective, exploration needs to be safe which means different (possibly expensive) sensors would be needed to detect different type of obstacles in the environment. The major problem for industrial vehicles is the increased cost for perception. For most AGVs the perception typically consists of a safety classified 2D laser scanner facing the direction of travel mounted at a height less than 10 cm about ground level to be able detect a person lying flat on the ground to follow the required safety regulations (EN 1525 - Safety of industrial trucks - Driverless trucks and their systems). The map will also require further manual intervention and fitted with outer properties, which typically include, predefined paths and loading / unloading areas.

2 Related Work

There is a large amount of different approaches to perform SLAM and the area is simply too wide to cover in this section [3]. The work presented here is a classical SLAM approach, where landmark observations and ego-motion estimates are combined to create a map. Landmarks are also used in extended Kalman Filter (EKF) based SLAM approaches [13]. The main drawback, however, of an EKF based approach is the computational complexity $O(n^2)$, where n is the number of landmarks, which makes the method only applicable online for smaller sized maps. Another drawback with an EKF approach is errors due to linearization.

Other landmark based methods are, for example, FastSLAM [12], a particle filter approach where each particle consists of a robot trajectory and where each landmark is treated independently. SEIF based SLAM [15] utilizes the sparseness of the information matrix instead of the full correlation matrix used in EKF. The Treemap algorithm by Frese [4], is an efficient approach $O(log(n))$, which uses a hierarchical treelike structure to subdivide the map into different regions which allows for an efficient update. Graph-SLAM approaches addresses the full-SLAM problem and are typically only used for offline mapping, however, there are some works (see, for example, Kretzschmar [6]) which enable Graph-SLAM approaches to be used over long time periods.

The work reported in Meyer et al. [10], is related to our work where they address they used a particle filter based SLAM approach that combines a predefined static map of buildings close to a parking lot and online dynamic map containing parked cars to facilitate the task of localization.

3 Gold-fish SLAM

This paper presents an approach of how SLAM can be used in a scenario where semi-dynamic landmarks (stacks of paper reels) are used in combination with static landmarks estimated (concrete pillars) to localize an eight ton diesel truck retrofitted with an AGV system. The environment used for evaluation is a warehouse for paper reels with an area of approximately 8000 m^2.

The key idea is that the vehicle position is initially known with respect to the static map and observations which can be related to the static map are used to correct the vehicle pose estimate and the position estimate of the semi-dynamic landmarks.

Beside improving odometry between areas with sparse static landmarks, SLAM is used to limit the pose uncertainty if the robot operates in the same area without seeing any static landmarks. In addition, landmarks might consists of object that the vehicle manipulates, in our specific context: paper reels, therefore it is indeed useful to estimate their position and to have them in a map.

The vehicle dot not have to keep track of all seen dynamic landmarks. It only needs to keep a limited set of them, which is reminiscent of the popular belief of a gold-fish memory.

3.1 Requirements from the AGV systems

AGVs require very good absolute localization estimates, which is used by the on board controller to follow paths. If an obstacle occurs, the AGV will stop and wait until the path is free. Therefore there is no requirement of path re-planning or obstacle avoidance behavior. Accuracy in providing localization estimate is very important for safe navigation, since industrial environments are cluttered with goods and other vehicles. It is also important for productivity because bad localization estimates can lead to unnecessary stops, for example, if a fleet of AGVs are used, other AGVs could be detected as obstacle and together cause 'dead locks'. On the other hand, on research platforms the inaccuracy of the absolute position localization estimates can be taken care of by the obstacle avoidance module, for example, vector field histogram (VFH) [17] or nearest diagram (ND) [11]. A popular localization approach used in research is Monte Carlo based localization [2], which often has a rather low update rate and where the output is typically not smooth enough to be directly feed into a control system. The output is instead used to determine if the goal state of the obstacle avoidance has been achieved where as the actual control are performed by the obstacle avoidance mechanism.

Just taking any SLAM approach and applying it directly in an industrial environment would simply not work. Firstly, the pose estimates has to be given within a specific coordinate system and bounds. This makes it clear that we also have to "anchor" the global localization into the SLAM representation and also to bound the uncertainty independently on the distance traveled.

Secondly, smooth pose estimates needs to be provided all the time, however, this is indeed related to loop closure. Unless the localization estimate are 'on the spot', jumps in the pose estimates will occur. Depending on the environment, the sensors, and the performance of the data association this jump could be large. This all comes down to the same conclusion as above; that the uncertainty in the pose estimate has to be bounded.

The uncertainty in the pose estimate is directly affected with whether the robot sees statical landmarks. In case the robot pose uncertainty grows beyond a predefined application specific threshold the vehicle will stop.

3.2 Overview and Difference Compared to online SLAM

An overview of the proposed method is depicted in Fig. 1 where input (static landmark maps, initial pose estimate and sensory data) are shown in rounded shaped boxes. The output fed into the control system is the estimated pose X_{pose} and the output of the "Safety stop" box. In the following we give more details about the functionalities of the main boxes.

- Incremental Pose Estimator: Integrates the odometry data to estimate the relative pose and its covariance.
- Prediction: Progresses the current pose estimate according to the kinematic model of the vehicle.
- Static Submap Selection: Based on the current pose estimate selects a subset of static landmarks that can be potentially observed by the system.
- Data Association: find the correspondence between landmarks (static and dynamic) and their observations.
- Init Landmarks: creates new dynamic landmarks for observations that were not matched in the data association step.

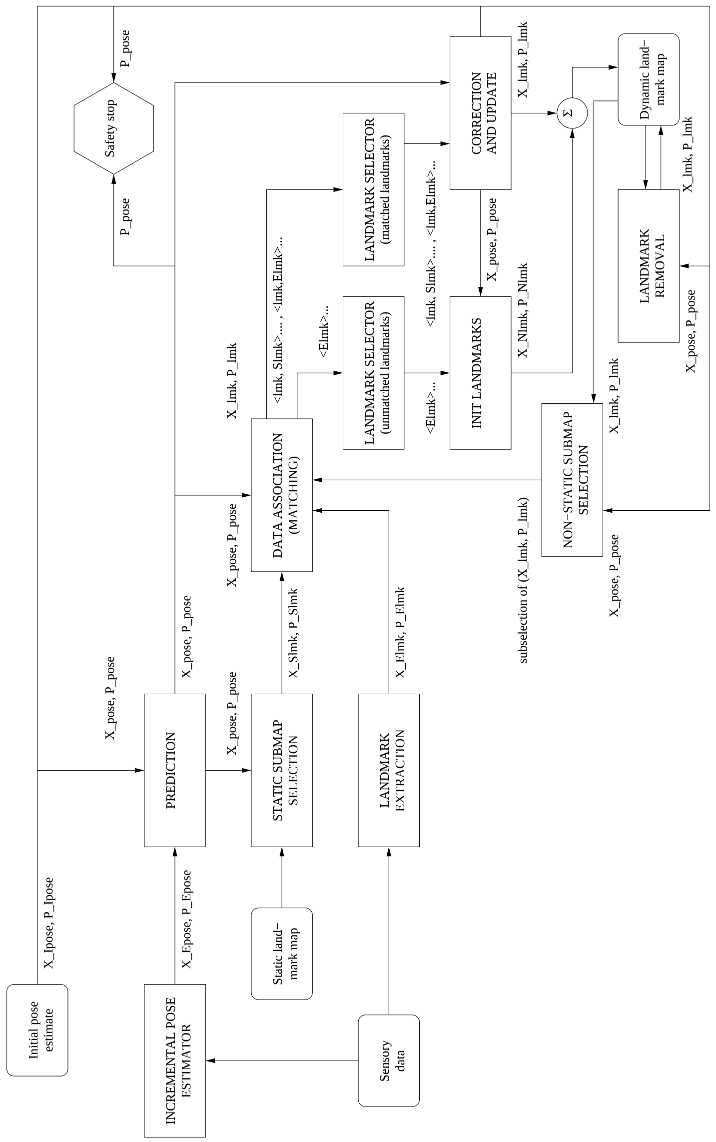

Fig. 1 Schematic overview of the system. The rounded shaped boxes indicates inputs to the system. The arrows indicates the flow of the state variables X and the corresponding parts in the correlation matrix P. The notation *pose*, *Ipose*, *Epose* corresponds to the current predicted / estimated robot pose, initial pose, incremental pose estimate (from odometry) respectively; *lmk*, *Elmk*, *Slmk*, *Nlmk* corresponds to the current predicted / estimated landmark position, the extracted landmarks, static landmarks and newly initiated landmarks.

- Correction and Update: Corrects the estimates of both the vehicle pose and landmarks positions based on the observations.
- Landmark Removals: Removes landmarks from the dynamic map based on age, uncertainty, size of the map and the max number of landmarks that can be kept.
- Non-static Submap Selection: Based on the current pose estimate selects a subset of dynamic landmarks that can be potentially observed by the system.

To avoid that the size of the static map influence the computational complexity of the system, static landmarks can be added / replaced / removed from the SLAM backend by looking at the current estimated pose. Static landmarks that are simply invisible or out of range for the sensors can safely be removed. Whether a static landmark should be added can be determined from the data-association step. The central advantage here is that the size of the sum of dynamic and static landmarks will be bounded independently on the size of the environment. One key requirement, however, is that the global pose of the vehicle is approximately known when the proposed localization system starts. In Table 1 the proposed idea is compared with an online SLAM approach to better illustrate the differences.

online SLAM vs Gold-Fish SLAM		
predict	predict based on dead reckoning	same
observe	use extracted landmarks, perform data association (either create a new landmark or assign the observation to an existing landmark)	same, but in addition perform data association to the used static landmarks (to be able to bound the error)
update	incorporate the observation, update the pose estimate and landmark estimates	same, but determine if any landmarks should be removed from the non-static map, do not update the static landmarks (position / uncertainty), and determine if any static landmarks should be added / replaced / removed in the SLAM backend based on the current pose estimate - this simply to keep the computational complexity independent on the static map size)

Table 1 Comparison of the proposed method against a standard online SLAM approach.

3.3 Landmark Extraction

Two types of landmarks are extracted from the 2d laser scanner data: paper reels and corners. Paper reels are the dynamic landmarks, while the pillars, created from a set of corners forms the static landmarks. Another major problem utilizing reflectors apart from the need to install additional infrastructure etc. in this type of environment (see Fig. 4) is they will be occluded by stacks of paper reels, therefore using

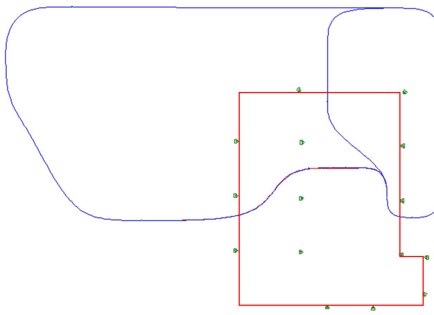

Fig. 2 The predefined paths used in the autonomous localization experiment (blue). The (green) dots represents the location of the installed reflector and the enclosed (red) area is where the reflector based localization potentially be used depending on how many reflectors are visible due to occlusions of paper reel stacks.

reflectors alone is simply not feasible, but could be used as a static landmark in the proposed method.

3.3.1 Paper Reel Extraction

Paper reels appears as sets of point of a circular arc, note that paper reels are in our settings always standing upright. To determine the position as well as the radius of each reel, a method based on Taubin's work for fitting a circle to data points [14] is used.

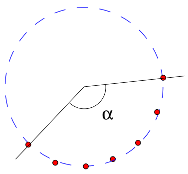

Fig. 3 The utilized arc angle α of the fitted circle, which could be interpreted how much of the actual circle was seen in the laser data. This measure is also useful to reject wrong estimates. A small value indicates that the data typically do not come from a reel (or a heavily occluded reel). A large value indicates that a corner like object are fitted instead.

To validate the extracted reels, all segment with to few points are ignored. Moreover, if the estimated reel radius is not within a predefined interval or the mean square fit (MSE) is above a threshold, the reel is rejected. To avoid to fit reels into a concave area, the estimated reel position has to be located further away than the mean position of the segment. The utilized arc angle on the circle, see Fig. 3, has to be located in a specified interval. Basically, a small value indicates that this is not likely to be a reel or that it is heavily occluded. A high value instead indicates that this is a corner like object.

An uncertainty estimate is calculated based on the distance between the laser points and the fitted circle along the radial direction.

3.3.2 Pillar Extraction

Segments that were not detected as reels are further processed to check if they form corners. We do not explicitly extract pillars as landmarks but instead uses corners directly. A corner is defined as an intersection of two walls (or lines in 2D) at an intersection of 90 degrees. The first step is to check that the size of the segment do correspond to a pillar and are within a specified interval. The segment is splitted into to two sub parts S_a and S_b at the location of the highest curvature. For both segments (S_a and S_b) a line is obtained using least square fit to each segment resulting in two lines L_a and L_b. If the MSE of the line fit is less than a predefined threshold for both segments a orthogonal check is performed to assure that the two lines are approximately 90 degrees apart $\pm$ another predefined threshold. The corner position is set to be the intersection of L_a and L_b, whereas the orientation is set using the normalized mean angle of the heading of L_a and L_b. The extracted corner consists of both a 2D position and an orientation.

The uncertainty is calculated based on the distance between the data and two orthogonal lines at $\pm$ 45 degrees relative to the orientation of the corner.

3.4 Obtaining the Static Map

The static map was created by manually driving the truck around in the ware house while collecting odometry, laser data from the safety sensor and also from the reflector based localization laser. The latter was used to more easily align the predefined map used in the AGV controller consisting of reflector poses and predefined paths. After the map was built, only the corners were saved to the static map.

4 Platform Description and Environment

4.1 The Truck Platform

The platform is based on a modified Linde H 50 D diesel forklift truck that has a load capacity of 5000 kg (see Fig. 4). The standard version of the truck was modified by shortening the mast and replacing the forks with a clamp. The truck was retrofitted with an off-the-shelf AGV control system developed by Kollmorgen. The AGV con-

Fig. 4 Left: The industrial truck used. The truck is retrofitted with an AGV controller. The bottom lasers are used as safety sensors and in this work also for landmark detection. Right: Stacked paper reels waiting to be loaded and one of the concrete pillars.

trol system comprises a set of hardware and software components (PC, IO modules, field bus controller, rotating laser ranger, etc.). The control system interfaces the actuators and sensors of the truck through the already built-in local CAN network. To detect paper reels, other landmarks and obstacles, three extra SICK laser rangers were incorporated into the truck (see Fig. 4), in the following experiments only the low mounted front and rear lasers were used.

The main functionality of the AGV controller is to navigate the truck from an initial location to a goal location. To do so, an operator defines and uploads a layout of drivable paths specified as collection of line segments and B-splines, see also Fig. 2. The controller achieves navigation tasks by following an appropriate path. The position of the truck can be tracked using a spinning laser (installed on the top of the truck) and reflective markers installed in the environment or by specifying external pose estimates, which in these experiments was provided with the SLAM based localization system. Here it was found to be very critical to give smooth and fast update rate (10Hz) to the AGV controller.

5 Experiments

The environment used in the experiments are paper reel warehouses, see Fig. 4.

Due to the lack of ground truth data, the comparison is done towards an autonomous run where the platform was controlled using localization estimates from the proposed system.

For the experimental evaluation two approaches were used. Treemap [4] and a standard EKF based method. Treemap operates on a very efficient tree-structure and have good computational properties. Treemap is a generic backend for any least square problems, however, in this approach the Treemap method is used in a similar way as a standard EKF method would be used by marginalizing out all robot poses except the last one. Input to the method are: odometry estimates with uncertainty and relative observation to landmarks with uncertainty, where as the output consists of an estimate of the robot pose including estimates of all landmarks. Due to the dependency between already added landmarks it is not straight forward to remove landmarks in Treemap therefore an EKF based approach was used in some evaluations.

The classical drawbacks of an EKF based approach are partly avoided in the proposed approach. The linearization error will not be problematic due the bounding of the error in the estimate and the complexity is somewhat covered by having a bounded map size.

5.1 Relating SLAM Estimates to a Predefined Map

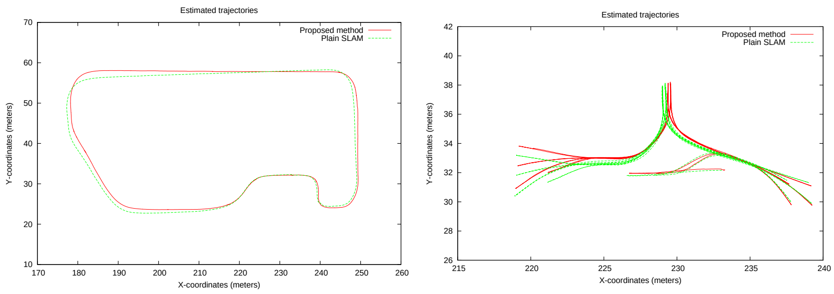

Fig. 5 Trajectories of using a ’plain’ SLAM method without any pre existing landmarks and the proposed method. Left: A complete loop. Right: Using a pick’n’place scenario.

Even though the estimates and the map are consistently built, the problem is to express the current estimates in the reference frame of the predefined map. Typically any reasonable offset in x and y directions do not cause any problem, however, only a very minor offset in orientation θ will give tremendous problems even if the distance is rather short, which can be seen in Fig. 5 where the truck was autonomously driven around in a predefined path consisting of a loop with a path length of 204 meters and the position difference is up to 2.5 meters.

In a pick’n’place scenario (see Fig. 5) paper reels were moved from a loading zone to an unloading zone using relative measurements to determine the path to drive to each reel and where each reel should be placed. Here one could argue that, given an initial estimate of the pose a plain-SLAM method would work, however, here instead the globally defined loading / unloading zones needs to be transferred into the plain-SLAM coordinate frame.

5.2 Changing the Amount of Dynamic Landmarks

The proposed method 'Gold-fish SLAM' implies that we need to constantly forget parts of the dynamic landmark map. To evaluate the impact of the number of dynamic landmarks used, one of the autonomous runs was evaluated offline. The system was rerun using recorded raw sensor data in approximately real-time for each evaluation. The results are presented in Table. 2.

Table 2 Difference in positions and heading in the proposed method while changing the amount of used dynamic landmarks

distances / max nb landmarks	0	10	20	30	40	60	80	100
avg. abs. position error	0.20	0.19	0.19	0.17	0.15	0.16	0.15	0.16
avg. abs. angular error	0.95	0.71	0.69	0.56	0.42	0.40	0.38	0.38

In the evaluated data set the number of seen static landmarks is enough to create a consisting map, however due to the smoothness criteria this approach would cause difficulties as seen in Fig. 6 where the avg. absolute position and angular error are shown using 0, 40 and 100 dynamic landmarks. From this figure it is clear that despite the rather accurate average error, there is significant difference especially in the angular estimate, which was an important control input in our platform.

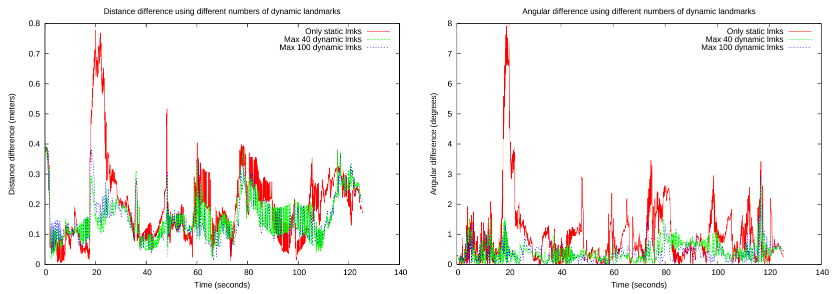

Fig. 6 Absolute difference between the estimated pose in an autonomous run with different number of dynamic landmarks. Left: distance difference. Right: angular difference.

5.3 Using Sparser Static Landmark Maps

In the previous section, the complete static landmark map was used in the evaluation. Here, the number of static landmarks used are instead decreased to check the system dependency on the amount of static landmarks. To simplify the presentation, the max number of dynamic landmark used was always 40. The accuracy of the system depends on the amount of observed static landmarks and their location rather then the number of landmarks in the static map. Therefore the evaluation also contains the ratio *r* of observation containing at least one static landmark (showing the number of static observation we have) and the number of unique static landmarks observed (indicating the map size actually used). In the evaluation 12 submaps were randomly created by subsampling the full static map. In Fig. 7, the angular and position difference are shown toghether with the number of static landmarks observed at each time for two different static sub-maps. The key difference is that in the left figure static landmarks are not seen for a longer period despite roughly the same map size.

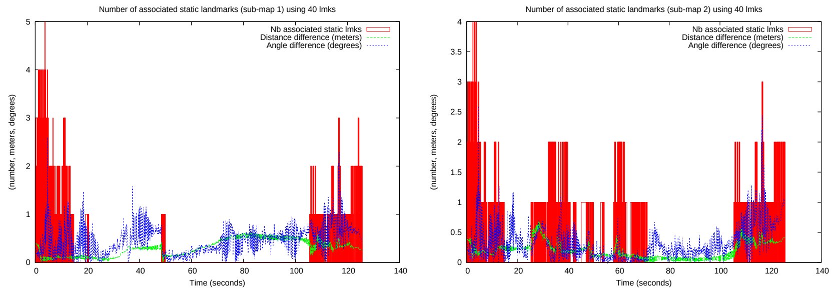

Fig. 7 Absolute difference between the estimated pose compared to an autonomous run using 40 dynamic landmarks with sub-sampled static landmarks maps. Left: Sub-map 1 with 12 used landmarks (position / angular difference : 0.3 meters / 0.46). Right: Sub-map 2 with 14 used landmarks (position / angular difference : 0.20 meters / 0.33 degrees).

Another result is depicted in Fig. 8, where the ration *r*, the angular and position difference are plotted with the number of used static landmarks. One interesting aspect is that the angular difference seems not to depend on the number of observed static landmarks. This indicates that even using few static landmarks the orientation of the vehicle can be determined, whereas to correctly determine the position is more sensitive to the density of static landmarks and preferable that the multiple landmarks are visible at the same time.

5.4 An autonomous localization loop

One critical aspect was to get the complete system to run in real-time on the platform traveling up to 3 m/s. A movie of one of the autonomous localization runs is located at the projects web page [9]. This utilized Treemap as a backend without limiting the number of dynamic landmarks.

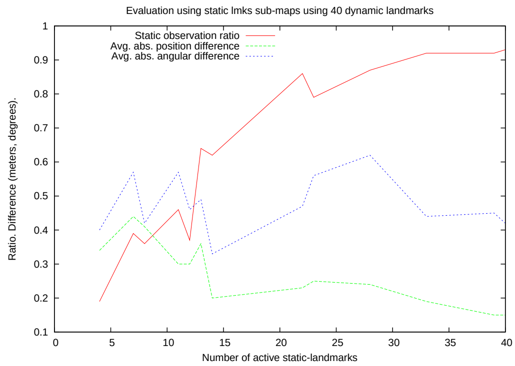

Fig. 8 Absolute difference between the estimated pose compared to an autonomous run using 40 dynamic landmarks with sub-sampled static landmarks maps. Left: Sub-map 1 with 12 used landmarks. Right: Sub-map 2 with 14 used landmarks.

6 Conclusions and Future Work

This paper presents an online SLAM approach to localize an AGV in a warehouse or factory consisting using static and dynamic landmarks. The results are so far preliminary and rather show a proof of concept. The evaluation should be extended with data covering larger areas, longer operational time and with reasonable ground truth. Future work include investigation of methods regarding selection of relevant dynamic landmarks. An interesting work would be to evaluate the approach on other types of environments.

Acknowledgment

The Authors would like to thank the teams from Kollmorgen, Linde Material Handling, and Stora Enso for their contribution. The Authors would also like to acknowledge the support of the Swedish KK foundation.

References

1. Åström, K.: Where am i and what am i seeing? algorithms for a laser guided vehicle. Master's thesis, Centre for Mathematical Sciences LTH, Lund University, Sweden (1991)
2. Dellaert, F., Fox, D., Burgard, W., Thrun, S.: Monte Carlo localization for mobile robots. In: Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA). IEEE (1999)
3. Durrant-Whyte, H., Bailey, T.: Simultaneous localisation and mapping (slam): Part i the essential algorithms. IEEE ROBOTICS AND AUTOMATION MAGAZINE **2**, 2006 (2006)

4. Frese, U.: Treemap: An $O(logn)$ Algorithm for Indoor Simultaneous Localization and Mapping. Autonomus Robots **21**(2), 103–122 (2006)
5. Frese, U., Wagner, R., Röfer, T.: A slam overview from a user's perspective. KI-Zeitschrift **24**(3) (2010). URL http://www.springerlink.com/content/p052055h1377gp85/
6. Henrik Kretzschmar, G.G., Stachniss, C.: Lifelong map learning for graph-based slam in static environments. KI-Zeitschrift **24**(3) (2010)
7. Kummerle, R., Hahnel, D., Dolgov, D., Thrun, S., Burgard, W.: Autonomous driving in a multi-level parking structure. In: Proceedings of the 2009 IEEE International Conference on Robotics and Automation (ICRA-09). Kobe, Japan (2009). To Appear
8. M. Batalin, G.S.S.: Coverage, exploration and deployment by a mobile robot and communication network **26**(2) (2004)
9. MALTA Project: http://aass.oru.se/Research/
Learning/malta/index.html
10. Meyer-Delius, D., Hess, J., Grisetti, G., Burgard, W.: Temporary maps for robust localization in semi-static environments. In: Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). Taipei, Taiwan (2010)
11. Minguez, J., Member, A., Montano, L.: Nearness diagram (nd) navigation: Collision avoidance in troublesome scenarios. IEEE Transactions on Robotics and Automation **20**, 2004 (2004)
12. Montemerlo, M., Thrun, S., Koller, D., Wegbreit, B.: Fastslam: A factored solution to the simultaneous localization and mapping problem. In: In Proceedings of the AAAI National Conference on Artificial Intelligence, pp. 593–598. AAAI (2002)
13. Smith, R., Self, M., Cheeseman, P.: Estimating uncertain spatial relationships in robotics pp. 167–193 (1990)
14. Taubin, G.: Estimation of planar curves, surfaces, and nonplanar space curves defined by implicit equations with applications to edge and range image segmentation. Pattern Analysis and Machine Intelligence, IEEE Trans. on **13**(11), 1115–1138 (1991). DOI 10.1109/34.103273
15. Thrun, S., Bugard, W., Fox, D.: Probabilistic Robotics. MIT Press, Cambridge, Massachusetts (2005)
16. Tsubouchi, T., Tanaka, A., Ishioka, A., Tomono, M., Yuta, S.: A slam based teleoperation and interface system for indoor environment reconnaissance in rescue activities. In: Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS) (2004)
17. Ulrich, I., Borenstein, J.: Vfh+: Reliable obstacle avoidance for fast mobile robots (1998)