Performance of $6D$ LuM and FFS SLAM — An Example for Comparison using Grid and Pose Based Evaluation Methods

Rolf Lakaemper, Andreas Nüchter, Nagesh Adluru, and Longin Jan Latecki

The focus of this paper is on the performance comparison of two simultaneous localization and mapping (SLAM) algorithms namely $6D$ Lu/Milios SLAM and Force Field Simulation (FFS). The two algorithms are applied to a $2D$ data set. Although the algorithms generate overall visually comparable results, they show strengths & weaknesses in different regions of the generated global maps. The question we address in this paper is, if different ways of evaluating the performance of SLAM algorithms project different strengths and how can the evaluations be useful in selecting an algorithm. We will compare the performance of the algorithms in different ways, using grid and pose based quality measures.

I. Introduction

The simultaneous localization and mapping (SLAM) problem is one of the basic problems in autonomous robot navigation. In the past many solutions of the simultaneous localization and mapping (SLAM) problem have been proposed [16]. However, it is difficult for engineers and developers to choose a suitable algorithm, due to a lack of true benchmarking experiments. In the well-known Radish (The Robotics Data Set Repository) repository [10] algorithms and results as bitmapped figures are available, but the algorithms have not been compared against each other. A valuable source for state of the art performance are competitions like RoboCup [7], Grand Challenge [4] or the European Land Robotics Trial [8]. However, the aim of such competitions is to evaluate whole systems under operational conditions, but are not well suited for the performance evaluation of vital components like perception. This paper presents two methodologies for comparing the results of state of the art SLAM algorithms, namely $6D$ LuM [3] and FFS [11].

LuM and FFS SLAM, treat the mapping problem as an optimization problem, that is a maximal likelihood map learning method. The algorithms seek to find a configuration ξ^*, i.e., scan poses that maximizes the likelihood of observations and could be written as

$$\xi^* = \underset{\xi}{\operatorname{argmax}}\, F(\xi),$$

where F is a function measuring the map quality or likelihood.

This paper is organized as follows: After an overview of related work, section III will give a brief description of the compared SLAM algorithms. Section IV presents our evaluation methodology, followed by the results. Section VI concludes.

Rolf Lakaemper, Nagesh Adluru and Longin Jan Latecki are with the Department of Computer and Information Sciences, Temple University, Philadelphia, U.S.A. lakamper@temple.edu, nagesh@temple.edu, latecki@temple.edu

Andreas Nüchter is with the Knowledge Systems Research Group of the Institute of Computer Science , University of Osnabrück, Germany. nuechter@informatik.uni-osnabrueck.de

II. Related Work

A. Robot Mapping

State of the art for metric maps are probabilistic methods, where the robot has probabilistic motion models and perception models. Through integration of these two distributions with a Bayes filter, e.g., Kalman or particle filter, it is possible to localize the robot. Mapping is often an extension to this estimation problem. Besides the robot pose, positions of landmarks are also estimated. Closed loops, i.e., revisiting a previously visited area of the environment, play a special role here: Once detected, they enable the algorithms to bound the error by deforming the mapped area to yield a topologically consistent model. For e.g. [15] addresses the issues in loop-closing problems. Several strategies exist for solving SLAM. Thrun [16] surveys existing techniques, like, maximum likelihood estimation, Expectation Maximization, Extended Kalman Filter, Sparsely Extended Information Filter SLAM. FastSLAM [18] and its improved variants like [9] use Rao-Blackwellized particle filters.

SLAM in well-defined, planar indoor environments is considered solved. However, little effort has been spent in *comparing* the performance evaluation of the SLAM algorithms. Given vast literature and various successful approaches for SLAM, such comparative studies are needed to choose appropriate SLAM algorithms for specific applications.

B. Performance Evaluation

Most research in the SLAM community aims at creating consistent maps. Recently, on the theoretical side of SLAM, Bailey et al. proves that EKF-SLAM fails in large environments [1] and FastSLAM is inconsistent as a statistical filter: it always underestimates its own error in the medium to long-term [2] that is it becomes over-confident. Besides focusing on such consistency issues, little effort has been made in *comparative* studies of SLAM algorithms.

Comparing two or more SLAM algorithms needs quantitative performance metrics like robustness, rate of convergence, quality of the results. Though the metrics used for comparison

in this paper are not completely new, the use of them in this context has not been done before, to the best of our knowledge. In this paper we mainly focus on the rate of convergence and quality of results of the two algorithms. They are measured in two different ways: occupancy grid based and pose based, as described in section IV.

III. Description of Mapping Algorithms

A. FFS

FFS [11] treats map alignment as an optimization problem. Single scans, possibly gained from different robots, are kept separately but are superimposed after translation and rotation to build a global map. The task is to find the optimal rotation and translation of each scan to minimize a cost function defined on this map. FFS is a gradient descent algorithm, motivated by the dynamics of rigid bodies in a force field. In analogy to Physics, the data points are seen as masses, data points of a single scan are rigidly connected with massless rods. The superimposition of scans defines the location of masses, which induces a force field. In each iteration, FFS transforms (rotates/translates) all single scans simultaneously in direction of the gradient defined by the force field under the constraints of rigid movement; the global map converges towards a minimum of the overlying potential function, which is the cost function. FFS is motivated by physics, but is adapted to the application of map alignment. It differs in the definition of the potential function, and in the choice of the step width of the gradient descent. The potential is defined as

$$P = \frac{1}{2} \sum_{p_i \in \mathcal{P}} \sum_{p_j \in \mathcal{P} \setminus p_i} \int_{\infty}^{r} \frac{m_1 m_2 \cos(\angle(p_i, p_j))}{\sigma_t \sqrt{2\pi}} e^{\left(-\frac{z^2}{2\sigma_t^2}\right)} dz \quad (1)$$

with $r = \sqrt{(X-x)^2 + (Y-y)^2}$,$p_i = (X, Y)$,$p_j = (x, y) \in \mathcal{P}$, $\mathcal{P}$ is the set of all transformed data points.

The potential function measures the probability of visual correspondence between all pairs of data points based on distance, direction and visual importance of data points: in (1) m_1, m_2 denote the visual importance of two data points, $\angle(p_i, p_j)$ the difference of direction of two points. Defining the visual importance of points dynamically is a simple interface to incorporate low or mid level perceptual properties (e.g. shape properties) into the of the global map into the optimization process. In contrast to algorithms like ICP, FFS does not work on optimization of nearest neighbor correspondences only, but (theoretically) takes into account all pairs of correspondences. Different techniques built into FFS drastically reduce the computational complexity.

FFS is steered by two parameters, σ_t in eq. 1 and the step width Δ_t of the gradient descent. σ steers the influence of distance between points. Initially set to a big value to accumulate information from a large neighborhood, it linearly decreases over the iterations to focus on local properties. The step width Δ_t in the FFS gradient descent is defined by an exponentially decreasing cooling process, similar to techniques like simulated annealing. Initially set to a high value it allows for significant transformations to possibly escape local minima. Decreasing the step enables local adjustment in combination with a low σ_t.

To conclude, the basic properties of FFS are

1) Data point correspondences are not made by a hard decision, but an integral between pairs of points defines the cost function instead of hard 'nearest neighbor' correspondences
2) FFS is a gradient approach, it does not commit to an optimal solution in each iteration step
3) The iteration step towards an optimal solution is steered by a 'cooling process', that allows the system to escape local minima
4) FFS transforms all scans simultaneously thus searching in $3n$ space of configurations with n scans.
5) FFS easily incorporates structural similarity modeling human perception to emphasize/strengthen the correspondences

B. $6D$ LuM

To solve SLAM, a $6D$ graph optimization algorithm for global relaxation based on the method of Lu and Milios [12] is employed, namely Lu and Milios style SLAM (LUM). Details of the $6D$ optimization, i.e., how the matrices have to be filled, can be found in [3]:

Given a network with $n+1$ nodes $X_0, ..., X_n$ representing the poses $V_0, ..., V_n$, and the directed edges $D_{i,j}$, we aim to estimate all poses optimally to build a consistent map of the environment. For simplicity, we make the approximation that the measurement equation is linear, i.e.

$$D_{i,j} = X_i - X_j.$$

An error function is formed such that minimization results in improved pose estimations:

$$\mathbf{W} = \sum_{(i,j)} (D_{i,j} - \bar{D}_{i,j})^T C_{i,j}^{-1} (D_{i,j} - \bar{D}_{i,j}). \quad (2)$$

where $\bar{D}_{i,j} = D_{i,j} + \Delta D_{i,j}$ models random Gaussian noise added to the unknown exact pose $D_{i,j}$. The covariance matrices $C_{i,j}$ describing the pose relations in the network are computed based on the paired points of the ICP algorithm. The error function Eq. (2) has a quadratic form and is therefore solved in closed form by Cholesky decomposition in the order of $\mathcal{O}(n^3)$ for n poses ($n \ll N$). The algorithm optimizes Eq. (2) gradually by iterating the following five steps [3]:

I) Compute the point correspondences (n closest points) using a distance threshold (here: 20 cm) for any link (i, j) in the given graph.
II) Calculate the measurement vector $\bar{D}_{ij}$ and its covariance C_{ij}.
III) From all $\bar{D}_{ij}$ and C_{ij} form a linear system $\mathbf{GX} = \mathbf{B}$.
IV) Solve for $\mathbf{X}$
V) Update the poses and their covariances.

For this GraphSLAM algorithm the graph is computed as follows: Given initial pose estimates, we compute the the number of closest points with a distance threshold (20 cm). If there are more than 5 point pairs, a link to the the graph is added.

To summarize, the basic properties of $6D$ LuM are

1) Data point correspondences are made by a hard decision using 'nearest neighbor' point correspondences
2) $6D$ LuM computes the minimum in every iteration
3) $6D$ LuM transforms all scans simultaneously
4) This GraphSLAM approach has been extended successfully to process $3D$ scans with representation of robot poses using 6 degrees of freedom.

In this paper, we process 2D laser range scans with the $6D$ LuM algorithm, i.e., in the range data the height coordinate is set 0. In this case the the algorithms shows the behaviour of the original Lu and Milios [12] GraphSLAM method.

IV. Evaluation

Evaluation of SLAM algorithms applied to real world data often faces the problem that ground truth information is hard to collect. For example, in settings of Search and Rescue environments, data sets are scanned, which usually have no exact underlying blue print, due to the nature of the random spatial placement of (sparse) landmarks and features. Hence map inherent qualities, like entropy of the distribution of data points, must be used to infer measures of quality that reflect their ability to map the real world. In the experiments, we will compare the performance of $6D$ LuM and FFS SLAM using a grid based and a pose based approach. Especially the grid based approach will be compared to visual inspection, which in this setting could be seen as a subjective ground truth of the performance evaluation. The reason for the choice of $6D$ LuM and FFS SLAM are the following:

- Both, $6D$ LuM and FFS SLAM, are state of the art algorithms to simultaneously process multiple scans, which is needed in settings of multi-robot mapping, which is a problem that currently has stronger focus in robot mapping.
- By visual inspection, $6D$ LuM and FFS perform, intuitively spoken, alike, although differing in details. Evaluation of the algorithms should be able to report this behavior.

It should be noted that $6D$ LuM is applied here to a $2D$ dataset, to compare it to the currently available version of the FFS algorithm, which works on $2D$ scan data only. Hence the LuM performance is only evaluated on three dimensions.

A. Occupancy Grid Based

Occupancy grids are used to represent the environment by discretizing the space into grid cells that have probabilistic occupancy values accumulated by sensor readings. They were introduced by [13] and are very popular in SLAM community. Learning occupancy grids is an essential component of the SLAM process. Once built they can be used to evaluate the likelihood of the sensor readings and also be used for guiding the exploration task as they are useful for computing the information gain of actions.

The likelihood of sensor readings is computed usually using different sensor models like beam-model, likelihood-field model or map-correlation model [17]. The information gain of actions can be computed using change in entropy of the grid. We use these basic ideas to compare the outcome of the two SLAM algorithms.

We use the beam penetration model described in [6] to compute likelihood of the sensor readings. Entropy of the grid is computed as described in [14]. Once the final map is obtained we compute the log-likelihood of *all* sensor readings with trajectory given out by the algorithm as

$$\mathcal{L}(m, \mathbf{x}_{1:n}) = \sum_{i=1}^{n} \sum_{j=1}^{K} \log(p(z_{ij})|\mathbf{x}_i, m))$$

where m is the final occupancy grid, $\mathbf{x}_{1:n}$ is the final set of poses, K is the number of sensor readings at each pose and $p(z_{ij}|\mathbf{x}_i, m)$ is computed using beam-penetration model as in [6]. The log-likelihood ranges from $-\infty$ to 0 and the higher it is the better the algorithm's output.

The entropy of the map is computed based on the common independence assumption about the grid cells. Since each grid cell in the occupancy grid is a binary random variable the entropy of $H(m)$ is computed as follows as described in [14].

$$H(m) = -\sum_{c \in m} p(c) \log p(c) + (1 - p(c)) \log(1 - p(c))$$

Since the value of $H(m)$ is not independent of the grid resolution it's important either to use same resolution or to weight the entropy of each cell with its size when comparing output from two algorithms. The lower the entropy of the map the better the outcome is. It is important to note that the entropy of the map and the likelihood-scores are not completely uncorrelated.

B. Pose Based

The occupancy grid based evaluations are very useful in the sense that they do not need "ground truths" to compare the results. But their memory requirements are proportional to the dimensionality and size of the environment. The pose based evaluations have an advantage in terms of memory requirements but require "ground truth" data to compare to. Here we present the technique that can be used to measure the quality of the output of SLAM algorithm assuming ground truth trajectory *is* available. The ground truth data can be obtained by surveying the environment as done in [5].

The SLAM algorithm gives out a final set of poses $\mathbf{x}_{1:n}$. Let the set of ground truth poses be $\mathbf{x}^G_{1:n}$. Since each pose in $2D$ mapping has three components viz. x, y, θ we compute the average error in each of the components. It is important that both the output of SLAM algorithm and the ground truth poses are in the same global frame. This could be done by rotating and translating the set of poses such that the first corresponding pose in each set is $(0, 0, 0)$. Once the poses are in same global frame the average error in each component is

computed as:

$$E(x) = \frac{1}{n}\sum_{i=1}^{n}|x_i - x_i^G|$$
$$E(y) = \frac{1}{n}\sum_{i=1}^{n}|y_i - y_i^G|$$
$$E(\theta) = \frac{1}{n}\sum_{i=1}^{n}\cos^{-1}(\cos(\theta_i - \theta_i^G))$$

$E(\theta)$ is computed as shown above so that the difference between the orientations is always between 0 and π.

V. Experiments

A. The Data, Visual Inspection

Both algorithms will be evaluated based on their performance on the NIST disaster data set with the same initial set of poses, see fig. 1. The data set consists of 60 scans and is especially complicated to map, since the single scans have minimal overlap only, and no distinct landmarks are present in the single scans. For this data set, no reliable ground truth pose data exists. This configuration was gained by random distortion of a manually gained global map.

Fig. 1. Initial configuration of the NIST data set. The data consists of 60 scans. The scale is in centimeters.

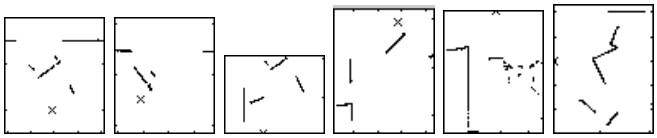


Fig. 2. 6 example scans of the NIST data set. In fig. 1, they can be located on the left side.

Six sample scans are shown in fig. 2. The final results of LuM and FFS respectively are shown in fig. 3. Visual inspection of 3 shows the following properties:

- The overall appearance of both approaches is equal.
- The mapping quality in different details is different: while FFS performs better in the left half, especially in the top left quarter, LuM shows a more visually consistent result in the right half, especially the top right corner.

To test if the evaluation does reflect these properties, we performed the following tests:

- First, entropy (and additional, likelihood-score) of the entire global maps (global evaluation) of both algorithms over all iterations are computed. This should reflect the behavior of both algorithms to converge towards optimal values, which should be in the same order of magnitude for both metrics.
- To check the evaluation of the different quality of mapping details in different areas, we split the result maps into four quarters and evaluated separately (regional evaluation).

In the LuM algorithm, 500 iterations were performed. FFS stopped automatically after 50 iterations, detecting a condition of changes in poses below a certain threshold. To compare all iteration steps, we extended the final result (iteration 50) to iterations $51 - 500$.

B. Grid Based Global Evaluation

The entropies and the likelihood-scores of the maps as the algorithms progress are shown in the fig. 4(a) and 4(b) respectively.

Please note the different scale on the iteration axis in the intervals $[1 - 50] and (50 - 500]$, in the first interval the iterations increase in units of 1, whereas in the second they increase in units of 10. This holds for all following figures.

You can see that the entropy decreases non-monotonically in case of FFS while in case of LuM it tends to be monotonically decreasing. This is based in the different nature of both algorithms: FFS is gradient based approach that has a built in "cooling strategy" for the step width to possibly escape local minima. In the beginning, FFS takes bigger steps, yielding a non monotonic behavior in its target function, which is also visible in the entropy. LuM optimizes its pose in each iteration, leading to a more smooth behavior, bearing the risk of being caught in local minima. This is also reflected in the convergence behavior in terms of speed: since LuM commits to optimal solutions earlier, it converges faster in the beginning, slowing down afterwards. FFS is slower (or more positively: more careful) in the first steps, due to the choice of step width that causes a jittering behavior. After the step width is balanced, FFS reaches its optimum very quickly. Interestingly, in both cases the near optimum value is reached after about 50 iterations.

The entropy score of both algorithms is comparable, which fulfills the expectations based on the visual inspection.

Similar behavior is observed in the likelihood scores. Hence the grid based evaluation is able to reflect the properties of both algorithms in the case of global evaluation.

C. Regional Evaluation

The maps are split into four regions, being North-West, North-East, South-West, South-East. Only the results for entropies are shown here, the likelihood scores did not lead to

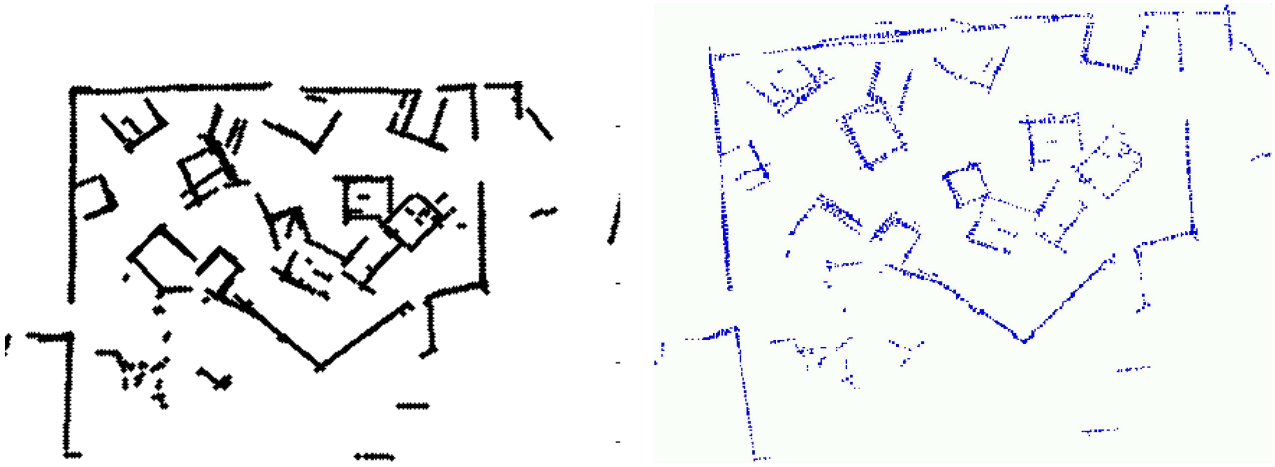
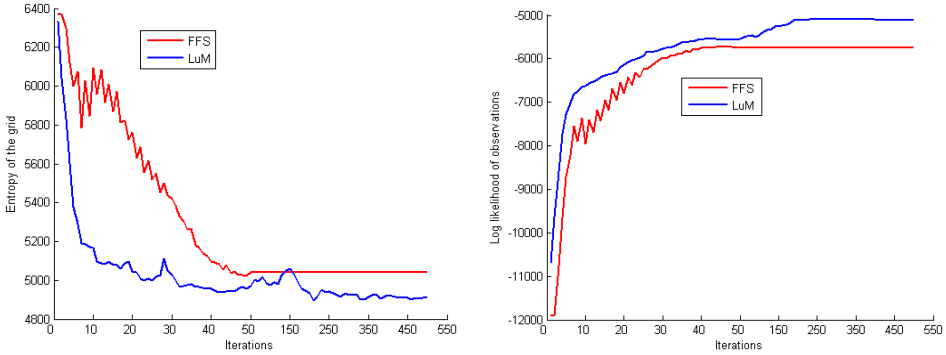


Fig. 3. Result of FFS (left) and LuM (right) on NIST data set, initialized as in 1. Evaluated by the overall visual impression, both algorithms perform comparably. Differences in details can be seen especially in the top left, where FFS performs better, and the top right, where LuM is more precise.



(a) (b)

Fig. 4. (a): The entropy of the map $H(m)$ at various stages of FFS and LuM. (b): The likelihood-score $\mathcal{L}(m, \mathbf{x}_{1:n})$ at various stages of the algorithms. Please note the different scale on the iteration axis in the intervals $[1-50] and (50-500]$.

additional further information. We expect better results for FFS in the North-West region, whereas LuM should outperform FFS in the North-East region, results for the southern regions should not vastly differ from each other.

The results are presented in fig. 5. fig. 5(a) shows the behavior for the North-West region of the map while 5(b) shows for North-East, 5(c) for South-West and 5(d) for South-East.

In accord with visual inspection, FFS is evaluated to perform better on the North-West region (fig. 5(a)) while LuM performs better in other regions. However, looking at the difference in final values, we can see that they always differ in ranges between ~ 30 and ~ 80 units: (a) $\sim 430-480$, (b) $\sim 3200-3280$, (c) $\sim 278-309$, (d) $\sim 950-1000$. Hence, although the tendency in the north regions is correct, the comparison to the southern regions, which should yield a smaller distance in values, does not clearly verify the correct estimation.

D. Global Pose Based Estimation

Pose based estimation needs a ground truth reference pose, see section IV-B. Since a ground truth for the NIST data set is not available, we just use the final set of poses of each algorithm. This necessarily leads to a graph that converges to an error of zero. Hence it does not give any information about the actual mapping quality, but it shows the behavior of the algorithms in terms of rate of convergence. Fig. 6 shows the behavior of the algorithms using error-metrics presented in section IV-B.

With respect to path to convergence, the pose based evaluation also shows the same properties of LuM and FFS as the grid based: LuM is "more monotonic", while FFS has jittering behavior. Interestingly the pose based evaluation shows FFS converging faster, which is in contrast to the result using grid based evaluation. While reasons for this different result will be topic of future discussion, it again shows that the choice of evaluation method has an influence on the property description of the algorithms.

VI. Conclusions and Future Work

This paper has presented a performance evaluation of two simultaneous localization and mapping (SLAM) algorithms namely $6D$ Lu/Milios SLAM ($6D$ LUM) and Force Field Simulation (FFS). These two algorithms have been applied to a $2D$ data set, provides by NIST. The results have been compared using two different metrics, i.e., an occupancy grid

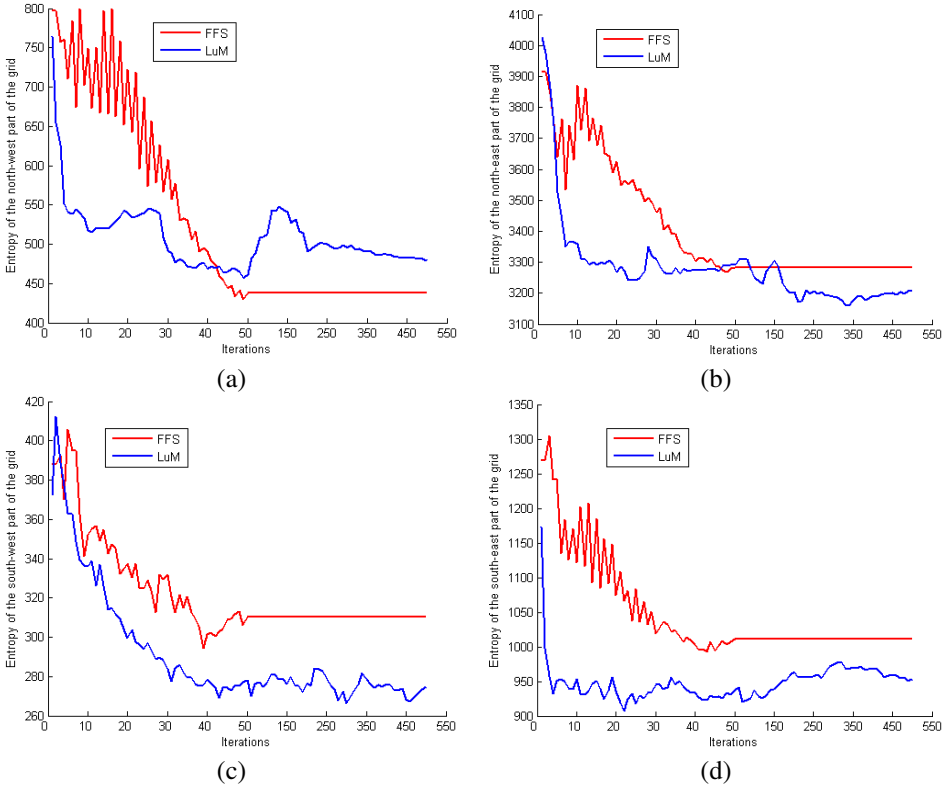


Fig. 5. (a): $H(m)$ for North-West (top-left quadrant) region of m. (b): $H(m)$ for North-East (top-right quadrant). (c): For South-West (bottom-left). (d): For South-East (bottom-right)

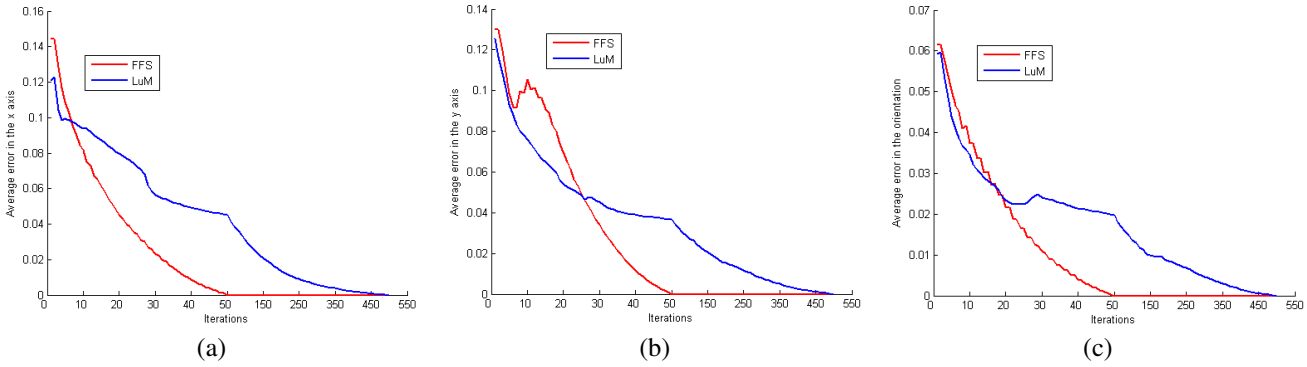


Fig. 6. (a): $E(x)$ for FFS and LuM. (b): $E(y)$. (c): $E(\theta)$ for FFS and LuM. The errors $E(x)$ and $E(y)$ are given in meters, $E(\theta)$ is given in radians.

based method and a pose based method. In addition these metrics have checked by visual inspection for plausibility. $6D$ LUM and FFS show similar performances on the data set considered in this paper.

Needless to say a lot of work remains to be done. The two algorithms have been on one data set. However, in robotic exploration task the environment is the greatest element of uncertainty. Mapping algorithms might fail in certain environments. In future work we plan to benchmark mapping algorithms using more suitable standardized tests and evaluate on automatically generated test cases. The grid and pose based evaluation methods will be used for these evaluations.

REFERENCES

[1] Tim Bailey, Juan Nieto, Jose Guivant, Michael Stevens, and Eduardo Nebot. Consistency of the EKF-SLAM Algorithm. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS '06)*, Bejing, China, 2006.

[2] Tim Bailey, Juan Nieto, and Eduardo Nebot. Consistency of the FastSLAM Algorithm. In *IEEE International Conference on Robotics and Automation (ICRA '06))*, Orlando, Florida, U.S.A., 2006.

[3] D. Borrmann, J. Elseberg, K. Lingemann, A. Nüchter, and J. Hertzberg. Globally Consistent 3D Mapping with Scan Matching. *Journal of Robotics and Autonomous Sytems*, 2007, (To appear).

[4] Defense Advanced Research Projects Agency (DARPA) Grand Challenge. `http://www.darpa.mil/grandchallenge/index.asp`, 2007.

[5] M. W. M. G. Dissanayake, P. M. Newman, H. F. Durrant-Whyte, S. Clark, and M. Csorba. A solution to the simultaneous localization and map building (SLAM) problem. *IEEE Transactions on Robotic and Automation (TRA)*, 27(3):229–241, 2001.

[6] A. Eliazar and R. Parr. DP-SLAM 2.0. In *IEEE International Conference on Robotics and Automation (ICRA '04)*, 2004.
[7] The RoboCup Federation. `http://www.robocup.org/`, 2007.
[8] FGAN. `http://www.elrob2006.org/`, 2007.
[9] G. Grisetti, C. Stachniss, and W. Burgard. Improved techniques for grid mapping with rao-blackwellized particle filters. *IEEE Transactions on Robotics (TRO)*, 23:34–46, 2007.
[10] A. Howard and N. Roy. Radish: The Robotics Data Set Repository, Standard data sets for the robotics community. `http://radish.sourceforge.net/`, 2003 – 2006.
[11] R. Lakaemper, N. Adluru, L. J. Latecki, and R. Madhavan. Multi Robot Mapping using Force Field Simulation. *Journal of Field Robotics, Special Issue on Quantitative Performance Evaluation of Robotic and Intelligent Systems. (To appear)*, 2007.
[12] F. Lu and E. Milios. Globally Consistent Range Scan Alignment for Environment Mapping. *Autonomous Robots*, 4(4):333 – 349, October 1997.
[13] H. Moravec. Sensor fusion in certainty grids for mobile robots. *AI Mag.*, 9(2):61–74, 1988.
[14] C. Stachniss, G. Grisetti, and W. Burgard. Information gain-based exploration using rao-blackwellized particle filters. In *Proceedings of Robotics: Science and Systems (RSS '05)*, pages 65–72, Cambridge, MA, USA, 2005.
[15] C. Stachniss, D. Hähnel, W. Burgard, and G. Grisetti. On actively closing loops in grid-based FastSLAM. *Advanced Robotics*, 19(10):1059–1080, 2005.
[16] S. Thrun. Robotic mapping: A survey. In G. Lakemeyer and B. Nebel, editors, *Exploring Artificial Intelligence in the New Millenium*. Morgan Kaufmann, 2002.
[17] S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics*. The MIT Press Cambridge, 2005.
[18] S. Thrun, D. Fox, and W. Burgard. A real-time algorithm for mobile robot mapping with application to multi robot and 3D mapping. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA '00)*, San Francisco, U.S.A, April 2000.