2008 IEEE/RSJ International Conference on Intelligent Robots and Systems
Acropolis Convention Center
Nice, France, Sept, 22-26, 2008

Efficiently Communicating Map Updates with the Pose Graph

Max Pfingsthorn and Andreas Birk

Abstract— **Robot mapping is a task that can benefit a lot from cooperative multi-robot systems. In multi-robot simultaneous localization and mapping (SLAM), it becomes very important how efficiently a given map can be shared among the robot team. To this end, the recently proposed pose graph map representation is used, adapted for use in a particle filter based mapping algorithm, and compared to the standard occupancy grid representation. Through analysis of corner cases as well as experiments with real robot data, the two map representations are thoroughly compared. It is shown that the pose graph representation allows for much more efficient communication of map updates than the standard occupancy grid.**

I. INTRODUCTION

Robot mapping is a task that can benefit a lot from cooperative multi-robot systems [1]. Accordingly, the currently dominant mapping paradigm of Simultaneous Localization and Mapping (SLAM) [2] is also intensively studied from the multi robot perspective [3], [4], [5], [6], [7], [8], [9], [10], [11], [12]. The core focus of this existing work is on the optimization of the map quality and the localization accuracy by taking as much as possible of each robot's data into account.

Here, it is assumed that the SLAM algorithms running on the individual robots are already performing sufficiently well. A robot r can of course nevertheless benefit from the maps on the other robots, namely if they contain parts of the environment that r has not explored yet. Such information could expedite loop closing, provide more map detail, as well as facilitate better localization. The challenge addressed here is to transmit the individual maps incrementally between all participating robots as efficiently as possible. Specifically, the case of a single robot communicating its map over the network is investigated, as this case can easily be scaled to larger teams. For the analysis presented here, it is further assumed that the individual robot knows how to incorporate the partial maps built by its peers into its own map. Several alternatives exist for this purpose in literature [13], [14], [15], [16], [17], [18].

To this end, it is shown how the recently introduced map representation of pose graphs [19], [20] can be employed for map transmission between robots and their operator for use in multi-robot mapping. Additionally, map merging is significantly easier in this representation, as mentioned in [19]. Using the pose graph representation for transmission of updates leads to a significant increase in the efficiency of the used communication bandwidth compared to using the standard occupancy grid representation.

Robotics Lab, Dept. of EECS, Jacobs University Bremen, D-28725 Bremen, Germany. http://robotics.jacobs-university.de, [m.pfingsthorn|a.birk]@jacobs-university.de

II. MAP REPRESENTATION AND TRANSMISSION

A. Representation

The way the map is stored clearly dictates the way it is exchanged between peers. An occupancy grid usually consists of a two dimensional array of cells, each denoting a square area with a fixed width and height. Each cell contains a value which denotes the likelihood that the covered area is occupied. The main advantage of this representation is that it is linear in the area explored, no matter how many sensor readings are aggregated within this region. However, one laser scan that is rendered into the occupancy grid could potentially change the value of many cells, for example if a robot enters a new room or turns a corner into a large corridor. This will result in a very large update message.

In this case, and many less severe ones, it would thus be more desirable to send only the range scan for deferred rendering on the receiver. However, this does not work across particle changes. If such a reevaluation of the particles occurs, for example due to a loop closure, the map has to be retransmitted as a whole to ensure map consistency. However, it would be better to just transmit the changed poses the previously transmitted laser scans were taken from.

This line of thought gives rise to a simplified version of the recently proposed pose graph map representation [19]. The pose graph map data structure is designed to be highly efficient both for the actual mapping process and for exchanging map updates over a network. For the experiments presented here, the pose graph was adapted for use in a particle filter based mapping algorithm, such as [21]. This simple version stores the original laser scan and a number of estimated poses, one for each particle in the filter.

It is important to notice that the two representations are equivalent as both hold the same information. While it is not possible to convert an occupancy grid into a pose graph, it is possible and often desirable to render the pose graph into an occupancy grid. Tasks such as path planning and frontier identification [22] are then much more straight forward.

The pose graph representation exactly applies the FastSLAM factorization [23]: All sensor observations are conditionally independent given the robot path. Instead of rendering an occupancy grid in each particle, we save pose estimates where separately stored laser scans were taken. Even though it is not generally guaranteed given this representation, it is possible to carry over the benefit of occupancy grids which are linear in the covered area. If a new scan is processed but does not add information to the map, it is simply not added to the pose graph.

The pose graph map representation also facilitates merging

maps of multiple robots online. As described in [19], the pose graph can hold multiple disconnected components at the same time. These components represent separate maps from different robots. Map merging is then reduced to the same mechanism as loop closure, e.g. by way of the relaxation algorithm described in [20]. Injecting nodes and edges into the pose graph which do not originate from the local mapping process but are transmitted via the network from another robot is trivial, as the needed parameters are already computed by the transmitting robot.

B. Transmission

There are various ways to transmit updates to both the occupancy grid and pose graph. Several message types are defined below in order to formalize their bandwidth requirements.

The most efficient way to update a grid map is to send partial updates, either through a list of changed grid cells with their new values, or through sending a complete subset of the map defined by a bounding box. In the following, these are named *CellList* and *BoundingBox*, respectively.

To be as efficient as possible, *BestGrid* dynamically chooses the best of *CellList* and *BoundingBox* for each update since each message excels in different scenarios. If many cells close to each other change, it might be best to use *BoundingBox*. Otherwise, *CellList* will probably perform better.

Specifically, the two message types require the following amount of memory:

1) *CellList*: Transmits $(x_c, y_c, v_c)^{N_c}$ which requires $3 \cdot N_c \cdot 4$ bytes. Let $CellList(N_c)$ denote this size.
2) *BoundingBox*: Transmits $(x_1, y_1, x_2, y_2, v_c^{(x_2-x_1)(y_2-y_1)})$ which requires $(4+N_b) \cdot 4$ bytes. Let $BoundingBox(N_b)$ denote this size.
3) *BestGrid*: Let $BestGrid(N_c, N_b) = min(CellList(N_c), BoundingBox(N_b))$.

with the number of changed cells N_c and the number of cells in a bounding box N_b. The above also assumes that 4-byte $float$s or ints are used to represent the data.

For the pose graph, the plain laser range scan is sent, along with the pose of the sensor when the scan was taken. If the current particle changes, the updated poses of all previously sent laser range scans are transmitted. Since both messages are necessary to transmit the simplified pose graph completely, they will be jointly named *PoseGraph*.

It is important to notice that the regular update message for *PoseGraph* has constant size since it depends on the physical resolution of the sensor. Again assuming 4-byte $float$s or ints, the message will consist of $(3+181) \cdot 4 = 736$ or $(3+361) \cdot 4 = 1456$ bytes.

In case the poses of previous observations are included, the message requires $(3+361+3 \cdot N_s) \cdot 4$ bytes, with N_s being the number of previous scans. Let $PoseGraph(N_s)$ denote this size.

These three parameters N_c, N_b, N_s and their implications are briefly discussed in the next section.

III. ANALYSIS

To properly compare the two map representations, an analysis of the corner cases as well as the best and worst cases is required. First, global best and worst cases are considered.

For both representations, the best case is to have no particle changes at all, that is the initial estimate is always the correct one. In this case, only incremental updates are needed for which the least amount of data has to be sent. Practically, N_c, N_b, and N_s are all minimized.

In the worst case each map update causes the best particle to change. This applies for both representations as well. A particle change would mean sending the entire map for the occupancy grid and sending the whole path along with the current laser range scan for the pose graph. This would result in the largest possible messages to be sent for every update. In terms of the previously identified parameters, all (N_c, N_b, N_s) are maximized.

However, these scenarios are not very informative for a comparison between the two. Much more interesting are the cases where one representation requires significantly less data to be sent than the other, i.e. $BestGrid(N_c, N_b) << PoseGraph(0)$ or $PoseGraph(0) << BestGrid(N_c, N_b)$.

For the sake of simplicity, we will assume a constant number of bytes per number in the following. Such an assumption is reasonable given the common 4-byte integer and floating point types.

The first case is best for the standard occupancy grid. Here, the map update only changes a single cell. This situation can occur when the map is already well known and new laser range scans only add very little new information. While a complete scan is integrated in the map, it only contributes new information to this one cell. It is important that it does add new information at least to one cell, otherwise the scan would be discarded completely. In contrast to the occupancy grid, the pose graph structure will send the entire scan. Concretely, the *CellList* message would transmit 3 numbers while the *PoseGraph* message transmits 184 or 364. In the worst case for the pose graph, it needs to transmit 361 numbers more.

The second case is best for the pose graph. In this case, we assume that the newest laser range scan reveals the maximum amount of free space, that is, it only contains maximum range readings in the direction of previously unknown space. Again, the *PoseGraph* message transmits 184 or 364 numbers, as it only needs to transmit the laser range scan. The occupancy grid, however, changes dramatically. Assuming a maximum range of $20m$ for the laser scanner, a cell size of $0.2m$, and that the scan was taken perpendicular to a major axis (i.e. it fits into a $20m \times 40m$ axis aligned box), the resulting *BoundingBox* message would transmit 20004 numbers. Here, the *BoundingBox* message needs to transmit 19640 numbers more. That is 5440% more than in the worst case for the pose graph. This estimate is rather conservative, as most laser scanners employed in the experiments below have a maximum range of $80m$ and the indoor maps use

a map cell size of $0.1m$. Also, sensor readings are never perfectly aligned with a major axis on the map.

These theoretical results show that the benefits of using the pose graph are significant in the extreme cases. While these cases rarely occur in practice, if ever, they do give a good indication of the potential savings the pose graph can produce. For instance, the bigger the influenced area of a single new scan, the better the pose graph should fare.

Given the enormous amount of extra information to be sent in the occupancy grid worst case and the moderate additional amount to be sent for the pose graph worst case, it is reasonable to assume that the break-even-point, i.e. the amount of change in the map for which both representations generate update messages with the same size, should fall well below the average case. This means that usually, the map changes much more than required by the break-even-point. Such an observation would imply that the pose graph would also outperform the occupancy grid significantly in the average case. However, it is hard, if not impossible, to determine the average case properly. Real data, which should give a better estimate of the average required bandwidth for each representation, is investigated in the next section.

IV. EXPERIMENTS AND DISCUSSION

A. Experiments

In order to arrive at a better comparison between the two map representations, we ran the open source rao-blackwellized particle filter based mapping algorithm of Grisetti et al. [21] on a variety of real robot data sets from the Robotics Data Set Repository (Radish) [24]. In particular, we used the "intel_lab" data set provided by Dieter Fox, the "fr-campus" outdoor data set provided by Cyrill Stachniss and Giorgio Grisetti, and the "albert-b-laser" as well as the "fr079" data set provided by Cyrill Stachniss.

An overview of the total cumulative number of bytes sent for each representation and message type is shown in Table I. Average message size is shown in Table II.

TABLE I
TOTAL NUMBER OF BYTES TRANSMITTED

	CellList	BoundingBox	BestGrid	PoseGraph
albert	23,985,384	110,706,576	21,637,008	3,258,300
fr-campus	9,062,788,884	12,440,115,576	7,565,584,256	102,890,628
fr079	35,324,268	136,852,392	29,492,924	5,041,692
intel	306,713,640	501,479,512	224,371,848	29,306,412

Table II contains two rows per data set. The first shows the average over all sent messages. The second shows the average size of all messages that did not occur from a particle change.

Figure 1 shows how the cumulative amount of bytes sent behaves over time. Figure 2 shows the relation between occupancy grid and pose graph updates over time, only showing *BestGrid* for clarity. The size ratio between updates for *BestGrid* and *PoseGraph* is plotted, which represents how much better or worse each individual pose graph update is compared to the corresponding occupancy grid one. A value below 1 means the occupancy grid update has a smaller size.

TABLE II
AVERAGE NUMBER OF BYTES TRANSMITTED PER UPDATE, WITH PARTICLE CHANGES AND WITHOUT

	CellList	BoundingBox	BestGrid	PoseGraph
albert	7,003	32,323	6,317	951
	3,393	29,874	3,393	732
fr-campus	608,400	835,130	507,890	6,907
	19,821	354,410	19,589	1,452
fr079	12,274	47,551	10,248	1,751
	4,465	42,725	4,465	1,452
intel	37,182	60,793	27,200	3,552
	4,305	39,545	4,305	732

For the Intel labs data set, we used a map cell size of $0.1m$, a maximum range of $80m$ for the laser scanner with 181 beams, and 20 particles. This configuration produced quite satisfactory maps, which is important to establish the relevance of our results. The computed maps will not be reproduced here as a well known and publicly available mapping algorithm and data sets were used.

Similar settings were used for the experiments on the Freiburg albert and fr079 data sets. For fr079, a cell size of $0.1m$, a maximum range of $80m$ for the laser scanner of 361 beams, and 25 particles were used. To compute results for the albert data set, a cell size of $0.1m$, a maximum range of $50m$ for the laser scanner of 181 beams, and 20 particles were used.

The configuration used for the Freiburg campus outdoors data set included a cell size of $0.2m$, a maximum range of $80m$ for the laser scanner of 181 beams, and 30 particles.

B. Discussion

Table I shows that for all data sets, the pose graph transmits the least total amount of data. Actually, the occupancy grid's most efficient messages transmit from 7 to 75 times as much data in total as the pose graph. Table II shows a comparison of the measured average message sizes. The outdoor data set displays a significantly larger average message size due to the larger map and higher number of scans in the data set. The indoor sets display a similar average message size when not taking particle changes into account. Even here, the outdoor set shows a significantly larger value. This is due to the much larger area each scan covers while outdoors. That the distances observed outside are much larger than the ones inside easily explains the relatively high bandwidth required by the occupancy grid. Additionally, the numbers reported in Table I clearly show the pose graphs superior performance in the outdoor setting as well, which was theorized already in Section III.

Table II also confirms another prediction from Section III, namely that the the break-even-point is far below the average case, at least given the examined data sets. Otherwise, the computed average messages sizes would be much closer when comparing *BestGrid* with *PoseGraph*.

Figure 1 shows the cumulative bandwidth usage over time. In the outdoor case (upper right in the figure), the enormous

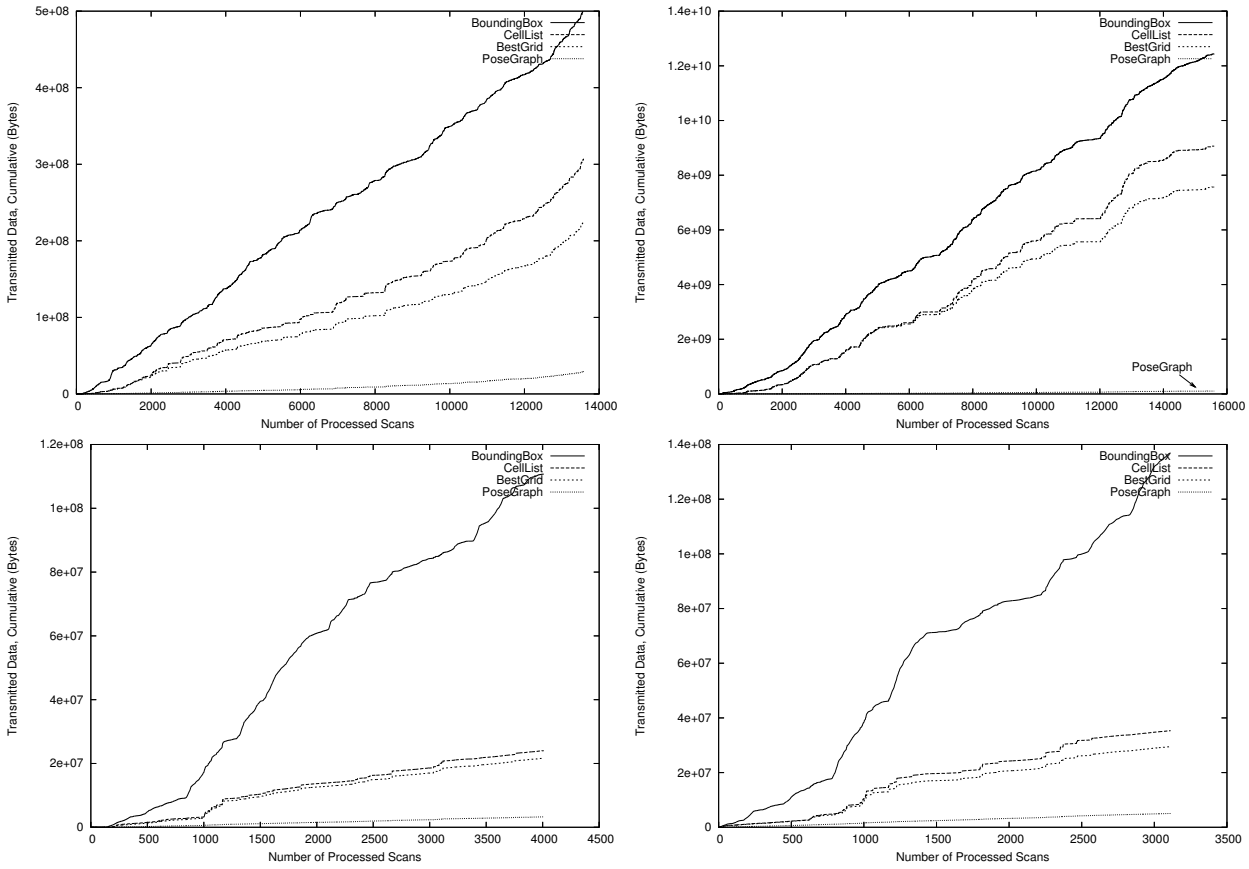

Fig. 1. Total bandwidth required for the map generated with the Intel labs (upper left), Freiburg Campus outdoor (upper right), Freiburg albert (lower left), and Freiburg fr079 (lower right) data set.

discrepancy between the occupancy grid and pose graph is again obvious. For reference, the line representing the pose graph bandwidth usage is pointed out explicitly. The clear advantage of the pose graph is also visible in Figure 2. Again, the outdoor data set shows the highest ratio between the occupancy grid and pose graph updates. On average, the pose graph update is 20.6 times smaller in this data set than the equivalent occupancy grid update. Other data sets also show the same effect, however with more moderate factors.

The indoor data sets behave in a very similar way to each other. Their data looks slightly different in Figure 1, but that is mostly because of scaling effects since the fr079 and albert sets have much less data than the Intel labs set. Within the first 4000 or so processed scans, they behave almost identical. The different configurations of the sensor and the mapping algorithm for these sets suggest that the data is representative and can be generalized to other indoor scenarios as well. That is a very important finding.

Figure 2 shows that almost every single update sent by the pose graph is smaller than for the occupancy grid. Each data set has a few outliers which happen to be in favor of the occupancy grid. As previously discussed in Section III, this is possible if only very few cells change value. In the case of the Intel labs data set, only 32 cells change value at the visible minimum. Even with the inefficient *CellList* message, that is still less data than sending a single range scan. Such small changes are however extremely rare.

V. CONCLUSION AND FUTURE WORK

A. Conclusion

The pose graph and occupancy grid representations were compared in terms of their efficiency to be transmitted via a network. After some theoretical analysis, experiments on real sensor data were performed to measure the difference between the pose graph and occupancy grid representations explicitly.

It was shown that pose graph is in fact much more efficient to share via a network than the standard occupancy grid representation while maintaining the same map. Given the extensive experiments conducted, the improved performance does not seem to be bound to a specific kind of data or environment. It excels at both indoor and outdoor scenarios.

This result is most promising for applications where extensive maps must be maintained and shared between a team of robots or their operator. As in most real scenarios

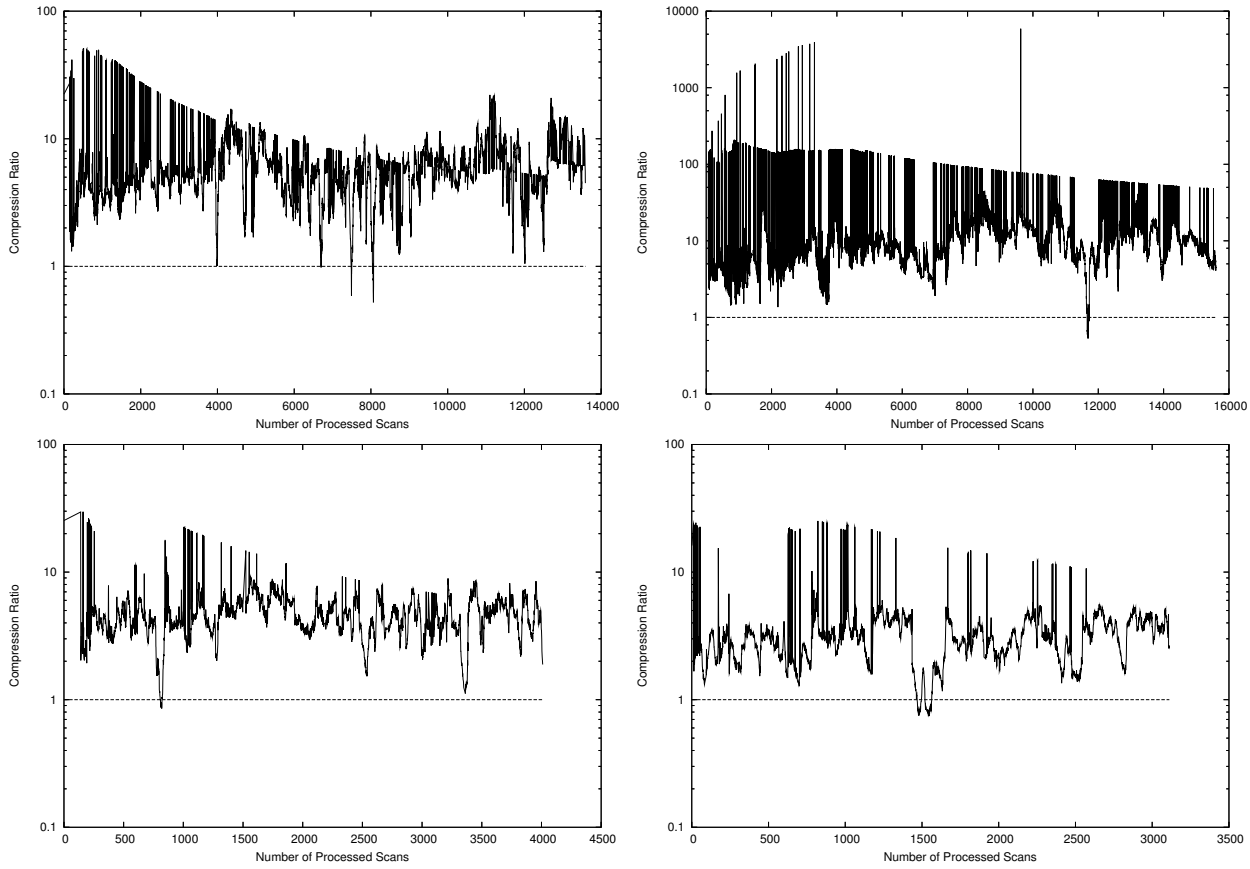

Fig. 2. Size comparison of updates for an occupancy grid vs. the pose graph for the Intel labs (upper left), Freiburg Campus outdoor (upper right), Freiburg albertb (lower left), and Freiburg FR079 (lower right) data set.

network bandwidth is rather limited, the presented results are applicable in and important to a broad variety of situations.

B. Future Work

One obvious extension to the described work is to allow for further compression of the laser range scan which is transmitted for each update. Since the nature of the data is known, namely that it is a radial scan of some environmental geometry, many possible easy to implement compression schemes come to mind. Such schemes might range from simple extrapolation and correction schemes to Huffman coding [25] of the distances measured by the scanner. The Robotics Group at Jacobs University is currently working on such a compression algorithm.

It is also conceivable to further reduce the amount of data sent for each particle change by sending only those poses that have actually changed. Most likely, poses that correspond to laser range scans taken far away from the current position of the robot will be minimally affected by the particle change. Those poses might not need to be transmitted and their omission would further streamline the pose graph efficiency.

ACKNOWLEDGMENTS

Please note the name-change of our institution. The Swiss Jacobs Foundation invests 200 Million Euro in **International University Bremen (IUB)** over a five-year period starting from 2007. To date this is the largest donation ever given in Europe by a private foundation to a science institution. In appreciation of the benefactors and to further promote the university's unique profile in higher education and research, the boards of IUB have decided to change the university's name to **Jacobs University Bremen (Jacobs)**. Hence the two different names and abbreviations for the same institution may be found in this paper, especially in the references to previously published material.

REFERENCES

[1] L. Parker, "Current state of the art in distributed autonomous mobile robots," in *Distributed Autonomous Robotic Systems 4*, L. Parker, G. Bekey, and J.Barhen, Eds. Springer, 2000, pp. 3–12.

[2] S. Thrun, "Robotic mapping: A survey," in *Exploring Artificial Intelligence in the New Millenium*, G. Lakemeyer and B. Nebel, Eds. Morgan Kaufmann, 2002.

[3] D. Fox, J. Ko, K. Konolige, B. Limketkai, D. Schulz, and B. Stewart, "Distributed multirobot exploration and mapping," *Proceedings of the IEEE*, vol. 94, no. 7, pp. 1325–1339, July 2006.

[4] S. Thrun, W. Burgard, and D. Fox, "A real-time algorithm for mobile robot mapping with applications to multi-robot and 3d mapping," in *Proceedings of the IEEE International Conference on Robotics and Automation*, 2000, pp. 321–328.

[5] J. Ko, B. Stewart, D. Fox, K. Konolige, and B. Limketkai, "A practical, decision-theoretic approach to multi-robot mapping and exploration," in *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2003.

[6] S. Williams, G. Dissanayake, and H. Durrant-Whyte, "Towards multi-vehicle simultaneous localisation and mapping," in *Proceedings of the 2002 IEEE International Conference on Robotics and Automation, ICRA*. IEEE Computer Society Press, 2002.

[7] J. Fenwick, P. Newman, and J. Leonard, "Cooperative concurrent mapping and localization," in *Proceedings of the 2002 IEEE International Conference on Robotics and Automation, ICRA*. IEEE Computer Society Press, 2002.

[8] N. Roy and G. Dudek, "Collaborative exploration and rendezvous: Algorithms, performance bounds and observations," *Autonomous Robots*, vol. 11, 2001.

[9] S. Thrun, "A probabilistic online mapping algorithm for teams of mobile robots," *International Journal of Robotics Research*, vol. 20, no. 5, pp. 335–363, 2001.

[10] W. Burgard, D. Fox, M. Moors, R. Simmons, and S. Thrun, "Collaborative multi-robot exploration," in *Proceedings of the IEEE International Conference on Robotics and Automation*. IEEE Press, 2000.

[11] R. G. Simmons, D. Apfelbaum, W. Burgard, D. Fox, M. Moors, S. Thrun, and H. L. S. Younes, "Coordination for multi-robot exploration and mapping," in *Proceedings of the Seventeenth National Conference on Artificial Intelligence*, 2000, pp. 852–858.

[12] D. Fox, W. Burgard, H. Kruppa, and S. Thrun, "A probabilistic approach to collaborative multi-robot localization," *Automous Robots, Special Issue on Heterogeneous Multi-Robot Systems*, vol. 8, no. 3, pp. 325–344, 2000.

[13] A. Birk and S. Carpin, "Merging occupancy grid maps from multiple robots," *IEEE Proceedings, special issue on Multi-Robot Systems*, vol. 94, no. 7, pp. 1384–1397, 2006.

[14] S. Carpin, A. Birk, and V. Jucikas, "On map merging," *International Journal of Robotics and Autonomous Systems*, vol. 53, pp. 1–14, 2005.

[15] S. Carpin and A. Birk, "Stochastic map merging in rescue environments," in *RoboCup 2004: Robot Soccer World Cup VIII*, ser. Lecture Notes in Artificial Intelligence (LNAI), D. Nardi, M. Riedmiller, and C. Sammut, Eds. Springer, 2005, vol. 3276, p. p.483ff.

[16] W. H. Huang and K. R. Beevers, "Topological map merging," in *Proceedings of the 7th International Symposium on Distributed Autonomous Robotic Systems (DARS)*, 2004.

[17] K. Konolige, D. Fox, B. Limketkai, J. Ko, and B. Steward, "Map merging for distributed robot navigation," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2003, pp. 212–217.

[18] G. Dedeoglu and G. Sukhatme, "Landmark-based matching algorithm for cooperative mapping by autonomous robots," in *Proceedings of the 5th International Symposium on Distributed Autonomous Robotic Systems (DARS)*, 2000.

[19] M. Pfingsthorn, B. Slamet, and A. Visser, "A scalable hybrid multi-robot slam method for highly detailed maps," in *RoboCup 2007: Proceedings of the International Symposium*, ser. LNAI. Springer, 2007.

[20] E. Olson, J. Leonard, and S. Teller, "Fast iterative alignment of pose graphs with poor initial estimates," in *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*, J. Leonard, Ed., 2006, pp. 2262–2269.

[21] G. Grisetti, C. Stachniss, and W. Burgard, "Improving grid-based slam with rao-blackwellized particle filters by adaptive proposals and selective resampling," in *Proceedings of the IEEE International Conference on Robotics and Automation, ICRA*, 2005.

[22] B. Yamauchi, "A frontier-based approach for autonomous exploration," in *IEEE International Symposium on Computational Intelligence in Robotics and Automation*, 1997, pp. 146–151.

[23] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit, "Fastslam: A factored solution to the simultaneous localization and mapping problem," in *Proceedings of the AAAI National Conference on Artificial Intelligence*. Edmonton, Canada: AAAI, 2002.

[24] A. Howard and N. Roy, "The robotics data set repository (radish)," 2003.

[25] D. Huffman, "A method for the construction of minimum-redundancy codes," *Proceedings of the IRE*, vol. 40, no. 9, pp. 1098–1101, Sept. 1952.