Towards Cooperative and Decentralized Mapping in the Jacobs Virtual Rescue Team

Max Pfingsthorn, Yashodhan Nevatia, Todor Stoyanov, Ravi Rathnam, Stefan Markov, and Andreas Birk

Jacobs University Bremen, Campus Ring 1, 28759 Bremen, Germany

Abstract. The task of mapping and exploring an unknown environment remains one of the fundamental problems of mobile robotics. It is a task that can intuitively benefit significantly from a multi-robot approach. In this paper, we describe the design of the multi-robot mapping system used in the Jacobs Virtual Rescue team. The team competed in the World Cup 2007 and won the second place. It is shown how the recently proposed pose graph map representation facilitates not only map merging but also allows transmitting map updates efficiently.

1 Introduction

The task of mapping and exploring an unknown environment remains one of the fundamental problems of mobile robotics. While many advances have been made in mechatronics, and the development of remotely controlled and autonomous single robot systems, the development of teams containing multiple autonomous robots is still an open research question.

Using cooperative robot teams for urban search and rescue (exploration and mapping) [1], space robotics [2], construction [3], or other tasks seems intuitively better than using uncoordinated teams or single robots. Using multiple robots has the obvious advantage of allowing more goals to be pursued simultaneously. Additionally, receiving data from multiple robots improves our confidence in the data, and allows the use of heterogeneous sensors (beyond the capacity of a single robot) for confirmation as well as heterogeneous manipulators.

The main challenge for a robot team is to be able to fuse their individual observations into one map. Extending robotic mapping to robot teams is the fundamental step for higher team functions, such as exploration or path planning. Distributed or cooperative mapping has been studied extensively [4–13], but often requires a significant change in the system architecture or core algorithms.

This paper presents a simple, but effective, approach to realize multi-robot cooperative mapping with a significantly reduced required bandwidth compared to the previously mentioned alternatives. The design presented here was implemented by the Jacobs University team that participated in RoboCup 2007 and won the second place in the Rescue Virtual Robots Competition.

The design was tested on simulated robots in the Urban Search And Rescue Simulation (USARSim) Environment. USARSim is a high fidelity robotics simulator [14], designed primarily to aid the development of controllers for individual robots as well as robot teams. Implemented on top of Unreal Tournament 2004, it uses the high end Unreal Engine 2 and Karma Physics Engine to accurately model the robots, the environment and interactions between the two. In addition to the simulated environment, the group also has experience with using cooperative real robot teams in USAR scenarios (figure 1).

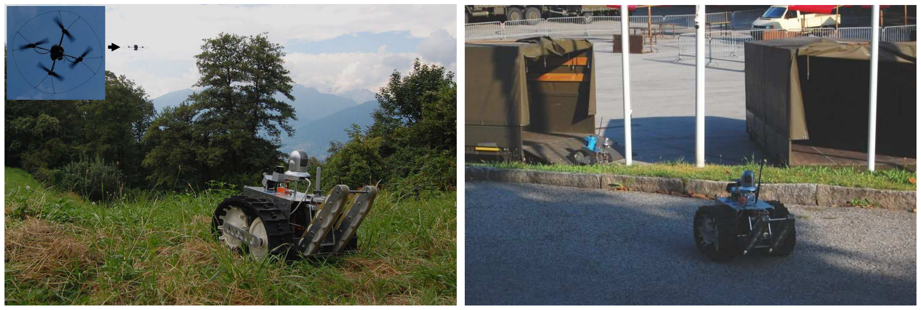
Fig. 1: Left: A Jacobs land robot cooperating with an aerial robot at the European Land Robotics Trials (ELROB) 2007 in Monte Ceneri, Switzerland. In this technology evaluation event, the aerial robot has to search for hazard areas like sites of fire in a forest, which the land robot then has to reach. Right: Two Jacobs robots support first responders at ELROB 2007 in the situation assessment after a simulated terrorist attack with NBC substances at a large public event. The robots can operate fully autonomously and perform a coordinated exploration of the area.

In the following, the proposed approach is described and evaluated in terms of generated maps and required communication bandwidth in comparison to other occupancy grid based mapping algorithms.

2 Cooperative Mapping

The presented cooperative mapping approach combines two very successful techniques, a rao-blackwellized particle filter mapping algorithm [15], and the recently proposed pose graph map representation [16, **?**].

The pose graph map consists of all laser range scans (plus additional information such as victim locations in USAR) and their relation to another. The nodes in the graph denote poses these observations were made from, and edges contain transformations between two such poses. These transformations also contain uncertainty information and might be generated by a motion model or a scan matching operation [17].

When paired with a particle filter, these transformations are not necessary anymore. Each observation simply corresponds to a specific pose in each particle. This simplified version of the pose graph map then only consists of a list of observations, e.g. laser range scans, and a number of sets of poses for these observations.

Intuitively, this map representation can be viewed as stopping a step short of actually rendering the corresponding standard occupancy grid. All information is retained, but instead of actually drawing the specific laser range scan into the occupancy grid map from the specific pose, pose and laser range scan are stored. This makes it very straight forward to render an occupancy grid when needed, e.g. for display or processing, but also facilitates efficient transmission via a network or the merging of maps from multiple robots.

In the presented approach, each robot r_i in the team runs a separate mapping process M_i. Each robot always transmits its new observations $o_{i,t}$, along with the current best pose estimate $p^*_{i,t}$, to all other team members. In each particle P_n of M_i consists of a possible pose for each observation $\{p_{i,t|n} | t = 1...T\}$. Each robot r_i also merges all received information from all other robots into its own pose graph map m_i. Therefore, each robot's map m_i is equivalent to the global map m_G.

It is important to note that the individual mapping process M_i is not required to process the input from other robots to achieve cooperative mapping. If a mapping process of some robot M_j updates its estimates for the previously made observations, e.g. in the event of a loop closure, the robot r_j would simply broadcast new pose estimates $p^*_{j,t}$. Robot r_i replaces the old estimates in its local map m_i with the newly transmitted ones in order to reflect this update.

In fact, the implementation in the RoboCup Rescue team only uses the particle filter mapping algorithm for localization. Mapping is exclusively done in the pose graph map, no fused information is used in the particle filter mapper. While this might negate a small advantage of a truly multi-robot SLAM approach, it is effective, efficient, and very easy to implement. A fused map is readily available to important other tasks that benefit tremendously from it, e.g. exploration and planning.

In some cases, it is also not necessary to process the combined pose graph in order to fuse maps from multiple robots. Often it is sufficient to know their relative starting poses. Rotating and translating the individual subsets of poses to match those starting locations and subsequently rendering the combined pose graph map already provides very satisfying results. In fact, the maps shown in section 4.1 were fused in this manner.

For cases where explicit alignment and correction of partial maps is necessary, it is possible to use algorithms presented by Olson et al [16, **?**,**?**] or Grisetti et al [**?**].

3 Incremental Map Updates

One specific important property of the presented approach is that map updates are especially efficient to share amongst team members. In this section, we compare the efficiency of the update messages for pose graphs and the standard occupancy grid representation.

There are various ways to transmit updates to both the occupancy grid and pose graph. Several message types are defined below and their sizes are briefly discussed in order to give an indication of the factors involved.

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 usually 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.

In order to compare the occupancy grid updates to pose graph updates, it is necessary to look at corresponding corner cases. We briefly describe two cases where one representation is considerably more efficient than the other.

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 small bits of 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 only 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 54.4 times more than in the worst case for the pose graph. This estimate is rather conservative, as most laser scanners have a maximum range of $80m$ and the maps shown below use a map cell size of $0.1m$. Also, sensor readings are never perfectly aligned with a major axis on the map.

These results already indicate that on average, the pose graph should be much more efficient. Section 4.2 compares the two representations given data from the USARSim simulator environment.

4 Results

4.1 Combined Maps

Figures 2 and 3 show maps generated by the presented multi-robot mapping approach.

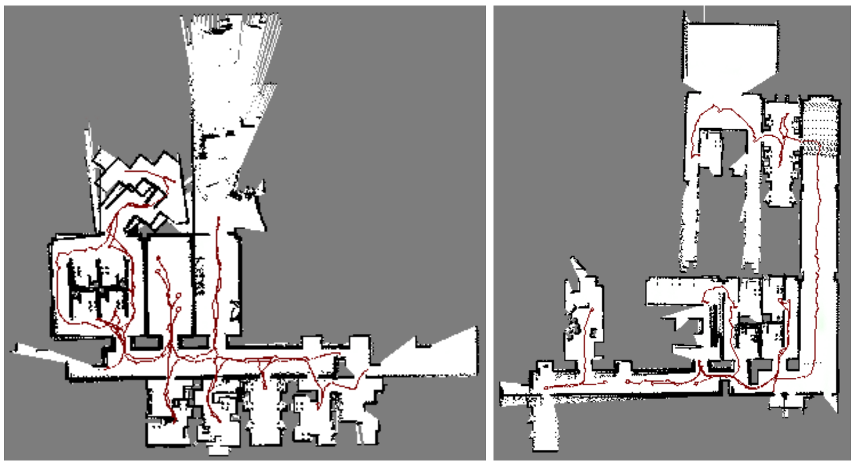

Fig. 2: Maps generated by a robot team of four robots in the RoboCup World Cup final round in 2007.

The maps shown in figure 2 shows the two delivered maps from the Rescue Virtual Robots competition final round at the 2007 World Cup in Atlanta. A team of four robots was used for both runs. Their paths are included in the maps as dark red lines.

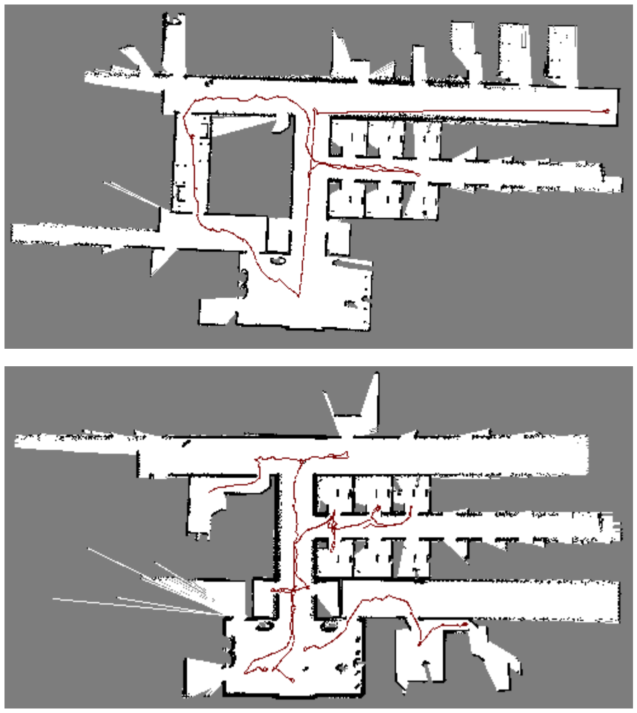

Fig. 3: Top: Map generated by single remote-controlled robot. Bottom: Map generated by an autonomous team of three robots.

Figure 3 shows two maps generated from the "compWorldDay2" environment distributed with the USARSim simulator. This simulated office building was used in the 2006 World Cup competition. As a comparison, both a single robot map as well as a multi-robot map is shown. Interestingly, the quality is very similar. There are no visible artifacts from map merging, such as slightly misaligned walls.

Up to four simulated ActivRobot P2AT platforms with SICK-LMS200 laser scanner were used in the competitions as well as for the following experiments.

4.2 Bandwidth Requirements

The theoretical discussion from section 3 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. This implies the difference between the two representations should be even more pronounced in an outdoor setting. We limit our experiments to indoor environments as it is a more common scenario.

Given the enormous difference between the two presented corner cases, 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.

Figure 4 shows experimental results from a simulated robot.

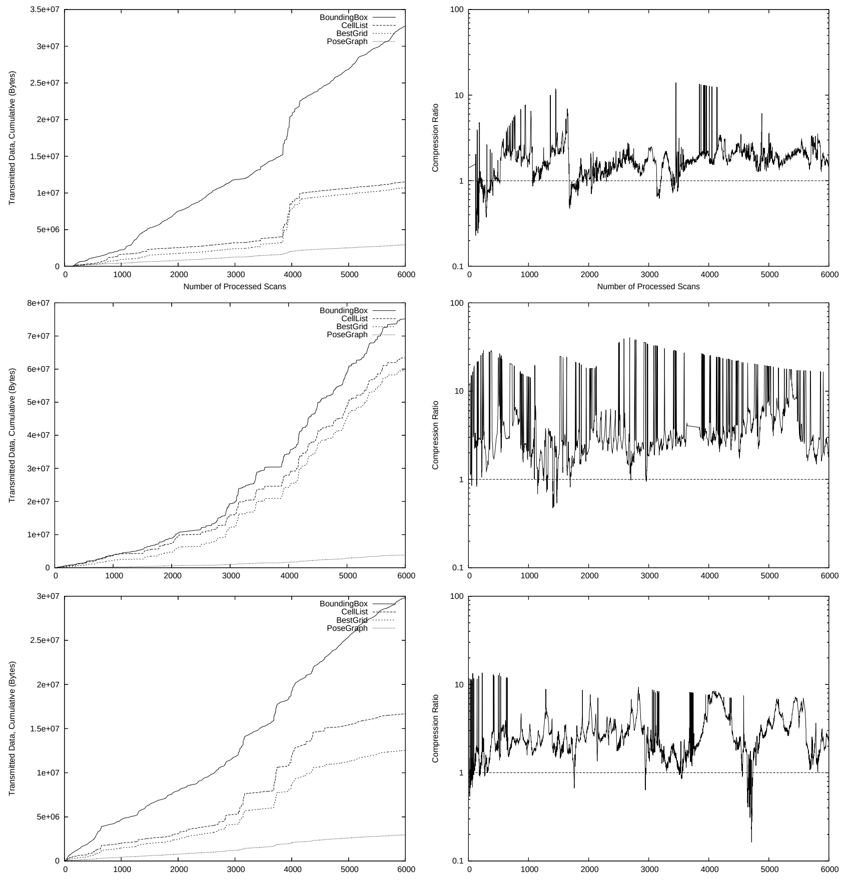

Fig. 4: Cumulative bandwidth required and comparison of individual update messages for different runs in the environment shown in figure 3.

The first row of plots in figure 4 corresponds the iterative transmission of the map shown on the top in figure 3. The other plots represent two other runs in the same environment. Maps are very similar to figure 3 and thus not reproduced.

Most importantly, the left column shows a comparison of the cumulative used bandwidth for each update message. Naturally, *BestGrid* is the most efficient for the occupancy grid. However, the bandwidth required by the pose graph consistently stays below the one for the occupancy grid. This is especially visible in the middle left plot.

The right column in figure 4 shows the size ratio of pose graph versus occupancy grid update messages. Here, only the *BestGrid* values were used for clarity. In most instances, the pose graph messages are much smaller than the occupancy grid ones. All three sets contain some outliers that are in favor of the occupancy grid. As discussed earlier, this may happen if only a few cells change value, for example when changing position slightly in an almost completely mapped room. Such small rooms are fairly common in the used environment. Also, it is visible that the average case in this environment is well above the break-even-point discussed before. The following table shows the exact amount of total cumulative bandwidth used.

	CellList	**BoundingBox**	**BestGrid**	**PoseGraph**	**Improvement**
Set 1	11,501,076	32,729,252	10,703,968	2,932,252	3.7
Set 2	63,354,144	75,161,548	59,541,032	3,823,200	15.6
Set 3	16,678,608	29,844,104	12,534,620	2,956,928	4.2

The rows in the previous table correspond to rows in figure 4. These numbers show the exact amount of bytes that needed to be transferred for each message type. In the most extreme case, the occupancy grid updates require 15.6 times more bandwidth than the pose graph ones. In the most modest one, it still requires 3.7 times more bandwidth.

While these numbers are depending heavily on the environment and the sensors used, it is reasonable to expect the pose graph to perform similarly well in other indoor environments. With different sensors, for example with a larger field of view or range, or in outdoor scenarios, the difference in bandwidth usage should be even more pronounced.

5 Conclusion and Future Work

In this paper, a simple, yet effective and efficient approach to multi-robot mapping was presented. Using the pose graph map representation, it is very straight forward to iteratively combine multiple maps from different robots. However, most importantly for many multi-robot scenarios, the pose graph is very efficient to transmit to other team members. Even in indoor environments the standard occupancy grid uses 3 to 15 times more bandwidth than the pose graph representation. This number should increase in outdoor environments.

As a logical next step, the fused pose graph map should be used in the particle filter mapping algorithm in order to reuse other robot's observations.

Currently, the presented system only uses the particle filter for localization given the robot's own sensor observations.

In order to further improve the bandwidth requirements for the pose graph representation, the group is already working on a compression method for transmitting raw laser scans.

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. Wang, J., Lewis, M.: Human control for cooperating robot teams. In: HRI '07: Proceeding of the ACM/IEEE international conference on Human-robot interaction, New York, NY, USA, ACM Press (2007) 9–16
2. Elfes, A., Dolan, J., Podnar, G., Mau, S., Bergerman, M.: Safe and efficient robotic space exploration with tele-supervised autonomous robots. In: Proceedings of the AAAI Spring Symposium. (March 2006) 104 – 113. to appear.
3. Brookshire, J., Singh, S., Simmons, R.: Preliminary results in sliding autonomy for coordinated teams. In: Proceedings of The 2004 Spring Symposium Series. (March 2004)
4. Fox, D., Ko, J., Konolige, K., Limketkai, B., Schulz, D., Stewart, B.: Distributed multirobot exploration and mapping. Proceedings of the IEEE **94**(7) (July 2006) 1325–1339
5. Thrun, S., Burgard, W., Fox, D.: 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) 321–328
6. Ko, J., Stewart, B., Fox, D., Konolige, K., Limketkai, B.: 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)
7. Williams, S., Dissanayake, G., Durrant-Whyte, H.: 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)
8. Fenwick, J., Newman, P., Leonard, J.: Cooperative concurrent mapping and localization. In: Proceedings of the 2002 IEEE International Conference on Robotics and Automation, ICRA. IEEE Computer Society Press (2002)

9. Roy, N., Dudek, G.: Collaborative exploration and rendezvous: Algorithms, performance bounds and observations. Autonomous Robots **11** (2001)
10. Thrun, S.: A probabilistic online mapping algorithm for teams of mobile robots. International Journal of Robotics Research **20**(5) (2001) 335–363
11. Burgard, W., Fox, D., Moors, M., Simmons, R., Thrun, S.: Collaborative multi-robot exploration. In: Proceedings of the IEEE International Conference on Robotics and Automation. IEEE Press (2000)
12. Simmons, R.G., Apfelbaum, D., Burgard, W., Fox, D., Moors, M., Thrun, S., Younes, H.L.S.: Coordination for multi-robot exploration and mapping. In: Proceedings of the Seventeenth National Conference on Artificial Intelligence. (2000) 852–858
13. Fox, D., Burgard, W., Kruppa, H., Thrun, S.: A probabilistic approach to collaborative multi-robot localization. Automous Robots, Special Issue on Heterogeneous Multi-Robot Systems **8**(3) (2000) 325–344
14. Carpin, S., Lewis, M., Wang, J., Balakirsky, S., Scrapper, C.: Bridging the gap between simulation and reality in urban search and rescue. In: RoboCup 2006: Robot Soccer World Cup X. LNAI, Springer (2006) 1–12
15. Grisetti, G., Stachniss, C., Burgard, W.: 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)
16. Olson, E., Leonard, J., Teller, S.: Fast iterative alignment of pose graphs with poor initial estimates. In Leonard, J., ed.: Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on. (2006) 2262–2269
17. Pfingsthorn, M., Slamet, B., Visser, A.: A scalable hybrid multi-robot slam method for highly detailed maps. In: RoboCup 2007: Proceedings of the International Symposium. LNAI, Springer (2007)