Jacobs Robotics
Jacobs University* Bremen, Germany

Max Pfingsthorn, Ravi Rathnam, and Andreas Birk

Jacobs University Bremen
Campus Ring 1
28759 Bremen, Germany
a.birk@jacobs-university.de
http://robotics.jacobs-university.de

Abstract. The 2009 Jacobs Robotics rescue virtual robots team is described in this paper. The robotics group at Jacobs University Bremen has been active in Rescue since 2001 and in Rescue Virtual Robots since the league's inception in 2005. It has competed in three RoboCup virtual rescue competitions so far.

1 Introduction

The Jacobs University robotics group has been working in the domain of rescue robots since 2001. It has also participated in the Virtual Rescue Robots league since its inception in 2005. The team has already participated in RoboCup 2005 (league demo), RoboCup 2006 in Bremen (2nd Place), RoboCup 2007 in Atlanta (2nd Place), and RoboCup 2008 in Suzhou, China (4th Place). In this paper, we present a team of heterogeneous semi-autonomous robots, which explores an unknown environment. The tasks are performed cooperatively to ensure maximum information extraction of the environment, in order to assist first responders to plan rescue operations.

The group tries to ground its research in USARSim with work on real robots. For this purpose, the group uses several rugbots - from rugged robot - as in-house designed systems for Safety, Security and Rescue Robotics [1]. The software architecture on the robots supports intelligent behaviors from semi-autonomy up to full autonomy [2–4]. This architecture includes work on autonomous multi-robot systems, e.g. for mapping of large areas with multiple robots [5, 6] and research on exploration under the constraints of wireless networking [7–9]. Despite the availability of multiple real robots in the group, the USARSim environment has been found to be a valuable tool for research and prototyping. It is an explicit goal to keep the virtual and real robots software in sync.

2 The Simulated Robot Hardware

The team consists of multiple Rugbots, a tracked robot developed at Jacobs University Bremen. The systems perform for example in Safety, Security and Rescue Robotics

* formerly International University Bremen

Fig. 1. Left: A Jacobs land robot cooperating with an arial 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 fire in a forest, which the land robot then has to reach. Right: A simulated RugBot and AirRobot in USARSim. Much like in the real scenario, the aerial robot will allow a much broader overview of the incident area.

(SSRR) field tests (figure 1) and the Rescue Robot League since 2006 [1]. The Rugbot is lightweight (about 35 kg) and has a small footprint (approximately 50cm x 50cm). It is agile and fast in unstructured environments and performs well on open terrain. This robot model has been validated extensively through experiments testing various physical aspects of its behavior in structured and unstructured terrain. Each robot has a standard payload of a laser scanner, a number of forward and backward looking sonars, a victim sensor, INS, GPS (outdoors only), and a video camera.

Additionally to the Rugbot, several other robots are used for research in heterogeneous robot teams. An AirRobot can be used to provide top down video coverage to the operator in order to coordinate the robot team. However, this only works in teleoperated scenarios due to the severe sensor load restrictions for the AirRobot. Other ground robot platforms, such as a TeleMax and a Zerg, are used to gain access to areas the Rugbots can not reach due to real mobility and artificial simulation constraints, such as USARSim's lack of a proper physics model for tracks.

3 System Architecture

The Jacobs Virtual Robot team aims to field a team of robots to perform coordinated mapping, exploration, and victim detection. However, each robot must also be able to perform its tasks individually and should not rely on other robots. For this purpose, we have designed a modular system with each module being as self sufficient as possible. Also, each module can be changed internally without affecting the other modules. The higher level intelligence is separated from the underlying sensors in order to make a seamless transition to real robots, or other robot types in the simulator.

The System Architecture can be divided into the following modules

1. Sensor and Actuator Interfaces
2. Mapping and Localization

3. Exploration
4. Navigation (including waypoint navigation)
5. Operator Interface
6. Victim Detection
7. Communication
8. Error/Crash Recovery

The following sections describe each of these in detail.

3.1 Sensor and Actuator Interfaces

A significant effort has been made to synchronize the sensor and actuator interfaces used in the robot software for the real robot and the virtual leagues. We have defined common interfaces for sensors such as a Laser Range Finder (LRF), which all classes representing specific sensor types implement. For example, a class that represents a real Hokuyo URG LRF implements the same interface as a simulated Sick LMS 200, or a real (or simulated) Sick S300, etc. Similarly, interfaces are defined for positional sensors (INS, odometry, gyro), cameras, 3D scanners (actuated LRF or range camera), etc.

Additionally, actuators, such as a differential drive, arms with several joints, pan-tilt units, have been generalized in corresponding interfaces. This allows a very flexible system of interchangeable parts and implementations. Not only is it possible to change

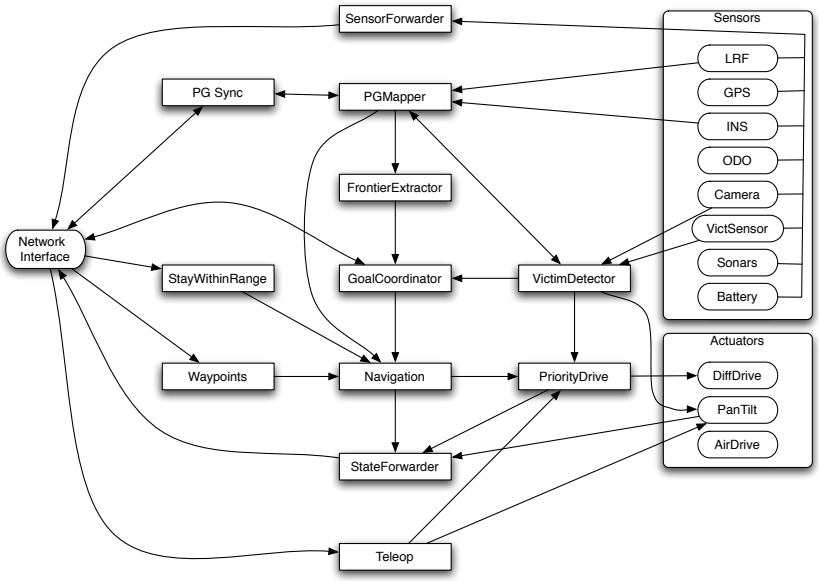

Fig. 2. The System Architecture. The arrows describe data flow. Forwarders send raw sensor or state values to the operator interface for monitoring.

all sensor and actuator instances to connect to the simulator instead of the real devices (or vice versa), but for example different LRF models with different physical properties (field of view, range, resolution) can be exchanged without needing to change the main algorithms or control software.

3.2 Mapping and Localization

Several advances are expected in the domain of mapping and localization. We will finally exchange the old mapping system based on Grisetti's open source mapping software [10] by a new maximum likelihood based system developed at Jacobs University. Last year, the system already partly moved to the new map representation called PoseGraph [11–13]. Using this new map representation made merging maps from multiple robots very simple.

This year, we extend this trend by exchanging the mapping algorithm to one specifically designed for the PoseGraph map representation. Algorithms such as Grisetti's open source iterative global relaxation implementation [14] and Olson's place recognition algorithm [15] will be used to implement such a SLAM approach quickly.

3.3 Exploration

Each robot has a shared global map which includes the sensor information gained from all robots in the team. The robots then use the combined map to perform a coordinated exploration of the environment. When a robot decides on a goal point, it informs all the other robots about the goal point. When a robot decides on a goal point, it ensures that the goal is not close to where other robots are going to. In the previous years, we have been using frontier exploration based on maximizing the area explored. However, this year, the robots explore with the aim of maximizing the area cleared using the victim sensor. By doing this, we reduce the amount of area which is explored but not cleared. Also, this reduces the pockets of uncleared space inside cleared space. This is more useful for first responders, as they do not need to go to confirmed victim free areas, as opposed to only explored area which still needs to be entered to check for victims. For example, when we compare figures 3 and 4, we can see that when we use normal frontier exploration (figure 3, the areas 1, 2 have uncleared pockets. Therefore, a rescue worker will need to enter these areas to check for victims. However, when we use cleared area based frontier exploration, we find that these areas to be cleared. A rescue worker can then ignore these areas completely, and go to other areas, thereby saving time and reducing the risk to himself. Even though the robot explores a smaller area, this exploration would be more useful to a first responder.

3.4 Navigation

The navigation module provides an interface by which other modules (Exploration, the Operator) are able to set goals and waypoints. The navigation module then computes a Harmonic Potential Function [16] which is a solution to Laplace's equation:

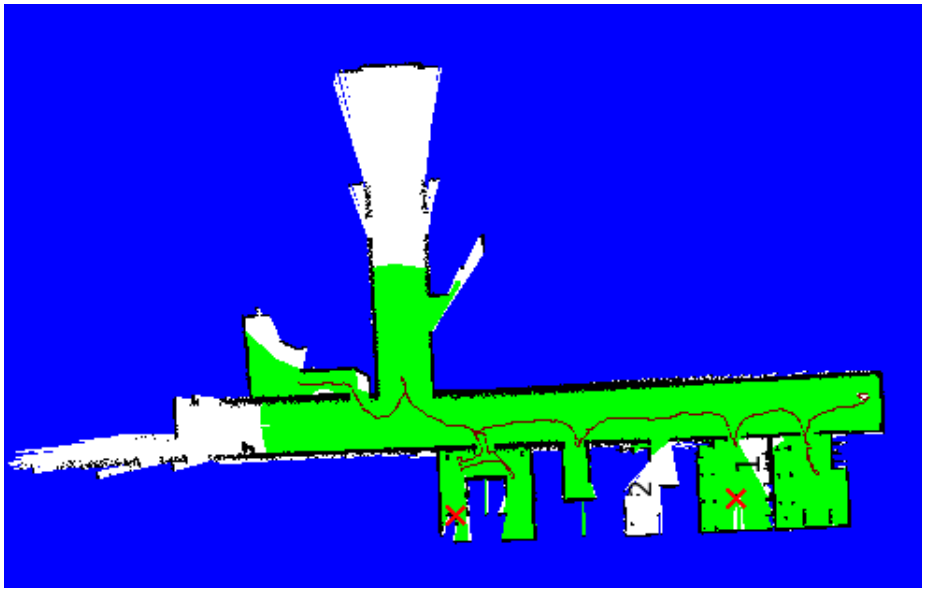

Fig. 3. A map created during using simple frontier exploration.

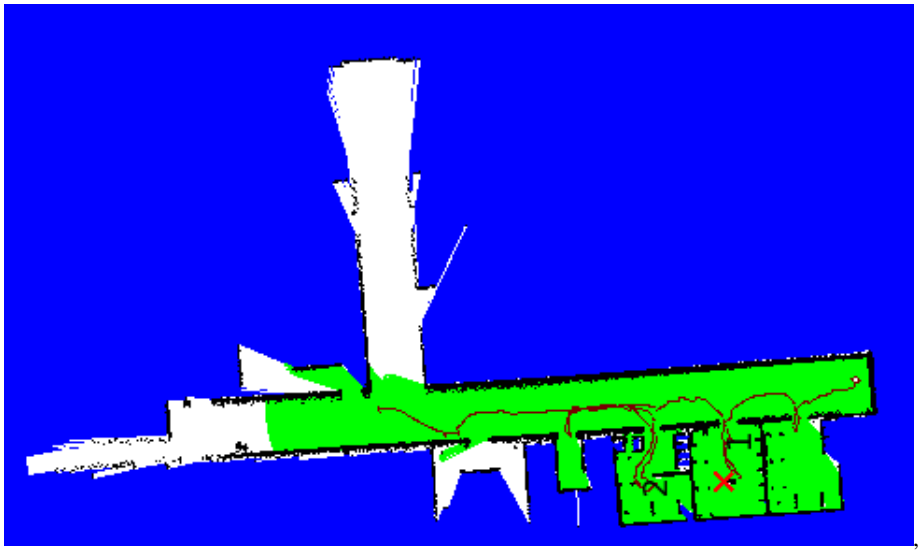

Fig. 4. A map with exploration of cleared area.

$$\nabla^2\phi = \sum_{i=1}^{n} \frac{\partial^2\phi}{\partial x_i^2} = 0$$

Using the Taylor series approximation of the function, a discrete version of the harmonic function is obtained on a grid. The harmonic function ϕ has the following properties.

- ϕ attains its global maxima and global minima on the boundary of its definition space.
- The value of ϕ at any point inside the definition space is equal to the average of the neighboring points.
- ϕ is complete- whether or not there is a solution to our problem(i.e. a path from start to goal) will be known.

Since the harmonic function does not have local minima, by following the gradient of the harmonic potential map, the navigation module guides the robot towards its goal.

To ensure quick computation, the harmonic potential map is computed in a bounding box containing the robot and a goal. If a path is not found, the boundary of the bounding box is dynamically grown. Also, the harmonic function is calculated every time the robot travels a certain distance. This ensures the navigation function is based on the newest information available to the sensors.

We are currently investigating other path planning algorithms as well. While the path planning and following based on the Harmonic Potential Field is very convenient, it takes a long time to compute long paths through the map. At the moment, we are experimenting with a hierarchical planner, using for example a fast and simple planner for the long path, breaking it into segments, and using the Harmonic Potential Field to plan and follow specific paths for each segment. Here, the PoseGraph will be of much help as it already contains a graph of traversable path segments and a simple graph search (such as A*) can be used to find a good global path.

Navigation will now also take over the obstacle avoidance logic. By very frequently updating a close neighborhood map with all available sensor data (lasers, sonars), much more informed decisions can be made in order to avoid moving obstacles, or obstacles that are not visible in the long range laser scanner. Avoidance behavior can be actively controlled to allow timely arrival at the goal point.

3.5 Operator Interface

Though the robots are able to perform their tasks autonomously, the user interface allows for high level mission control as well as low level behavior control. The Operator interface gets sensor, mapping, as well as status information from all the robots. The operator can examine the a-priori information available (any geo-referenced image or vector data), and overlay the map information to guide the robots in the environment using goal points. In addition to the map, the main display shows the path the robot is following to get to the goals, as well as any victims found in the environment. The operator can view the video stream from any of the robots to directly control the motion of the robot.

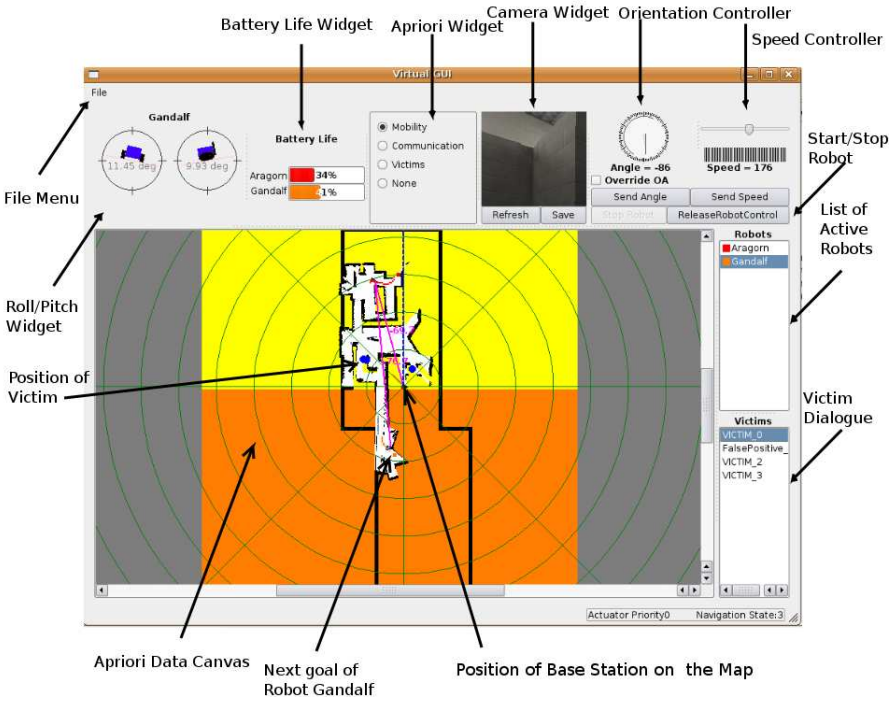

Fig. 5. Snapshot of Operator Interface

3.6 Victim Detection

The victims are mainly detected using the simulated victim sensor provided by USARSim. The victim sensor simulates image processing on a camera image. The victim detection module groups these parts by location. If there are enough parts at a particular location, it determines that there is a victim. In situations where there are not enough parts, the victim detection module drives the robot towards the potential victim until a significant number of parts can be detected. In situations when the robot approaches the object up to a specific distance, and does not find enough body parts, the robot determines this victim sensor reading to be a false positive.

Additionally, the operator can manually place victims on the map display in the GUI. Already present victim information can be examined and classified by the operator. Victim information is marked as *potential victim* if the detection algorithm is still gathering data (this victim is a high priority goal point for the navigation module) or if the operator sets it so. If a victim is a *potential victim* at the end of the run, it is reported with a lower probability. The victim is marked as *confirmed* if the detection was successful or if the operator manually confirmed the presence of a victim in the video data associated with this victim. Finally, a victim information can be marked as *false positive* by the detection or manually by the operator. In these two last cases, new victim sensor readings are added to the vicim information when it is available, but the

robot ignores this new data for navigation purposes. This means, only new victims are investigated by the robots, old, already known, victims are ignored.

3.7 Communication

The communication module is very important for the system as the robots coordinate the tasks of exploration, mapping and victim localization. It provides an interface for the transfer of information between any two robots in the network. Through a simple online graph search, the system is able to dynamically adjust the routes the data should take as robot goes out of communication range of one or more robots. Such graph searches also allow dynamic constrains on robot motion to force the robots to stay within communication range of the base station. Additionally, the communication module takes care of compressing the data in order to minimize network traffic.

3.8 Error and Crash Recovery

It is very likely that one or more of the robots or even the operator interface will crash during a mission. Usually, teams do not have much time for development during the RoboCup competitions and even less for testing their new code. Therefore, much of the code produced at the competitions will have bugs and will be prone to crashes or other errors. However, most of this new code is usually written as a response to new challenges only known after the competition started and is thus very important for the success of the team. In a real scenario, it is well conceivable that late night changes have to be made to robots and their control code on the disaster site. The risk factors of late night development are the same.

This is why we have developed a separate module explicitly designed not to prevent these bugs, but to recover from the errors or crashes they cause quickly and automatically. Since all data is replicated in the team, each robot may request old data from other team members. For example, the robot might ask its neighbor for the complete map including victim locations, etc. Other data generated at run-time can be synchronized in similar ways. This way, even if a robot crashes, it can be restarted and it can continue its operation almost seamlessly. Similarly, the GUI can re-request data from the team of robots and recreate the information lost after a crash. Currently, we are working on a way to serialize incoming data to disk on the operator machine, such that it does not have to be sent again by the robots. This would also allow us to recover from a crash of the GUI after the mission is over and the robots ran out of battery.

4 Conclusion

This year, the Jacobs Robotics team makes a few significant advances. Most prominently, a new multi-robot SLAM algorithm will be used. Another improvement is a true ad-hoc network based on online search of the connection graph between robots. Previous efforts in multi-robot coordinated exploration are continued. Finally, many under-the-hood changes significantly increase the stability and extensibility of the system as a whole.

References

1. Birk, A., Pathak, K., Schwertfeger, S., Chonnaparamutt, W.: The iub rugbot: an intelligent, rugged mobile robot for search and rescue operations. In: IEEE International Workshop on Safety, Security, and Rescue Robotics (SSRR). IEEE Press (2006)
2. Birk, A., Carpin, S.: Rescue robotics - a crucial milestone on the road to autonomous systems. Advanced Robotics Journal **20**(5) (2006) 595–695
3. Birk, A., Markov, S., Delchev, I., Pathak, K.: Autonomous rescue operations on the iub rugbot. In: IEEE International Workshop on Safety, Security, and Rescue Robotics (SSRR). IEEE Press (2006)
4. Birk, A., Kenn, H.: A control architecture for a rescue robot ensuring safe semi-autonomous operation. In Kaminka, G., U. Lima, P., Rojas, R., eds.: RoboCup-02: Robot Soccer World Cup VI. Volume 2752 of LNAI. Springer (2003) 254–262
5. Birk, A., Carpin, S.: Merging occupancy grid maps from multiple robots. IEEE Proceedings, special issue on Multi-Robot Systems **94**(7) (2006) 1384–1397
6. Carpin, S., Birk, A., Jucikas, V.: On map merging. International Journal of Robotics and Autonomous Systems **53** (2005) 1–14
7. Rooker, M.N., Birk, A.: Multi-robot exploration under the constraints of wireless networking. Control Engineering Practice **15**(4) (2007) 435–445
8. Rooker, M., Birk, A.: Communicative exploration with robot packs. In Noda, I., Jacoff, A., Bredenfeld, A., Takahashi, Y., eds.: RoboCup 2005: Robot Soccer World Cup IX. Volume 4020 of Lecture Notes in Artificial Intelligence (LNAI). Springer (2006) 267 – 278
9. Rooker, M., Birk, A.: Combining exploration and ad-hoc networking in robocup rescue. In Nardi, D., Riedmiller, M., Sammut, C., eds.: RoboCup 2004: Robot Soccer World Cup VIII. Volume 3276 of Lecture Notes in Artificial Intelligence (LNAI). Springer (2005) pp.236–246
10. 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)
11. Pfingsthorn, M., Birk, A.: Efficiently communicating map updates with the pose graph. In: Proceedings of the International Conference on Intelligent Robots and Systems (IROS). (2008)
12. Pfingsthorn, M., Nevatia, Y., Stoyanov, T., Rathnam, R., Markov, S., Birk, A.: Towards collaborative and decentralized mapping in the jacobs virtual rescue team. In Iocchi, L., Matsubara, H., Weitzenfeld, A., Zhou, C., eds.: RoboCup 2008: Robot Soccer World Cup XII. Springer Verlag, Berlin (2008) To appear.
13. 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)
14. Grisetti, G., Stachniss, C., Grzonka, S., Burgard, W.: A tree parameterization for efficiently computing maximum likelihood maps using gradient descent. In: Proceedings of Robotics: Science and Systems, Atlanta, GA, USA (June 2007)
15. Olson, E., Walter, M., Leonard, J., Teller, S.: Single cluster graph partitioning for robotics applications. In: Proceedings of Robotics Science and Systems. (2005) 265–272
16. Connolly, C.: Applications of harmonic functions to robotics. In: Intelligent Control, Proceedings of the 1992 IEEE International Symposium on. (August 1992) 498 – 502