Autonomous Mapping in the Real Robots Rescue League

Stefano Carpin, Holger Kenn, Andreas Birk

School of Engineering and Science
International University of Bremen
Germany
{s.carpin,h.kenn,a.birk}@iu-bremen.de

Abstract. We describe the mapping system implemented by the IUB team in the Robocup Real Robot Rescue league. Although very simple, this represents the first and up to now unique system which produces a human readable map that can be directly given to the rescue team to quickly locate victims. We describe the sensors suite we used, how data are fused together, and the algorithm we implemented. Examples coming both from lab experience and real competition are provided.

1 Introduction

The ultimate goal of the Robocup Rescue league is to develop autonomous systems that can be used in the time immediately following a natural disaster, like earthquakes or similar calamities ([1],[2],[3]). While the simulation league aims to develop high level coordination algorithms to optimize the interaction between different entities involved in the rescue scenario, the real robots league attacks the problem from the other side. The goal is to develop easy to deploy and to operate systems that can be used to assist human rescuers teams. In particular, such systems are supposed to be used right after the emergency starts, where human operators must be extremely cautious but also extremely fast. On the one hand, right after an earthquake, for example, buildings can be unstable, so they should be entered only when they have been assessed to be stable. On the other hand, victims should be located as soon as possible, but without exposing the rescue teams to unnecessary dangers. Robots, either autonomous or teleoperated, can then be used to quickly determine whether there are victims in the buildings or not, but keeping high safety standards. If a victim is located into a partially collapsed building, providing the rescue team a map with its location would result in a cut in the rescue operation time. This lowers the risk for the operators and enhances the likelihood that the victim will be extracted promptly. It is then evident that the ability of producing such maps is one of the fundamental abilities that rescue robots must exhibit. The challenging aspect emerges from the nature of the environment to be mapped, which is reasonably assumed to be deeply unstructured. The problem of features identification turns out to be much more difficult in such scenario, thus preventing a straightforward use of

standard techniques. An additional point which deserves attention is the *single shot* nature of the problem, and the time constraints. By single shot we mean that during the exploration, either autonomous or teleoperated, it is unlikely that the robot will visit many times the same place. Instead, it can be expected that it will run down hallways and it will enter rooms just once or very few times. This is related to the need to complete the exploration and mapping cycle quickly, in order to maximize the chances to locate possibly wounded victims. The paper is organized as follows: section 2 describes the sensors we used for building the map, while section 3 describes the mapping algorithm. Conclusions are offered in section 4. For an overall description of the robots we developed, the interested reader is referred to [4].

2 Sensors used for mapping

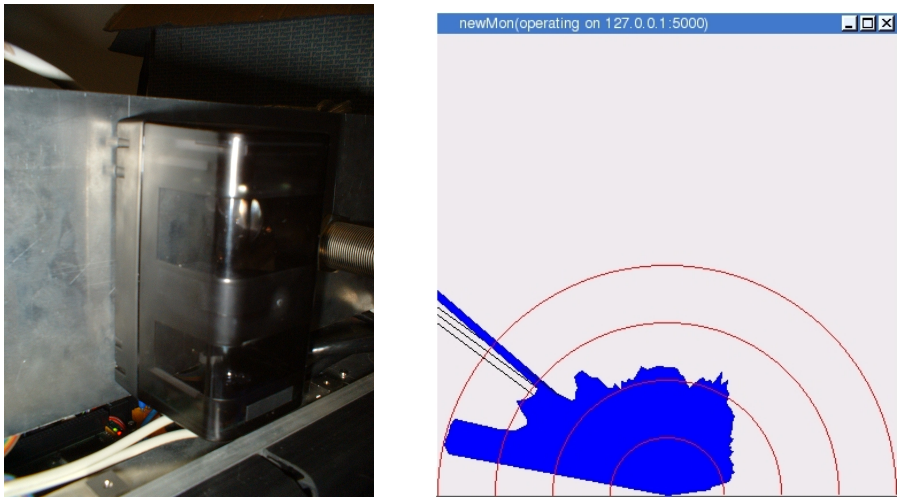
During the 2003 competition we used two different platforms, in order to be able to negotiate all the three arenas (see figure 1).

(a) (b)

Fig. 1. On the left the six wheels platform, designed to enter the yellow and orange arenas. On the right the tracked platform, designed to enter all the arenas.

The platforms however share the sensors we used for mapping purposes. Such sensors significantly drove our mapping strategy. While of course better choices could have been taken, it is worth stressing once again that the system engineering approach puts additional constraints. The main sensor we used for mapping is the Hokuyo PB911 ([5]). The sensors is a proximity range finder, based on the well established time of flight principle. The sensor covers and area 162 degrees wide with 91 samples. It returns distances in meters that can be considered reliable if below 3 meters. While compared with other commercial similar devices it could seem that the sensor performs poorly, this is not the case. In fact, rescue robots are supposed to operate in indoor cluttered environments likely to be occluded by debris and similar. Thus, the need for a long range

scanner is not so high. Additional advantages are its small size, weight and power consumption (see figure 2.a).



(a) (b)

Fig. 2. On the left, the Hokuyo PB9-11 sensor. On the right a snapshot of the data provided by the PB911 sensor. It is possible to see that the robot is facing a corner in the walls (on its left, two meters ahead), as well as spurious readings (on the right)

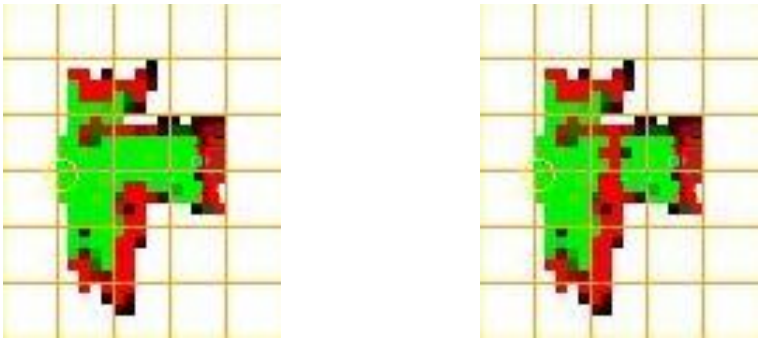
The sensor provides its output via a standard RS232 serial interface and we controlled it using the FAST-Robots software architecture recently developed at IUB ([6]). Figure 2.b illustrates how the data provided by the sensor are presented to the operator which is driving the robot.
In addition to the range scanner, the robots are equipped with traditional dead reckoning sensors, plus a magnetic compass for absolute orientation (see [4]). We also point out that for the mapping task we did not use at all the sonar sensors available on our platforms.

3 Mapping the arena

As a first step towards the implementation of a robot able to build environment maps, we implemented a very simple occupancy grid based approach. Among the many mapping approaches available ([7],[8],[9],[10] or see [11] for an excellent overview) we chose occupancy grids for the following reasons. First, the environments we wish to map are significantly unstructured. This means that features extraction could be a challenging problem. Not only this is true from

an hypothetical long term goal point of view (after an earthquake walls collapse, furniture falls, etc), but also from the point of view of the actual competition. Indeed, just the yellow arena exhibits a reasonable structure with corridors, corners and so on. On the contrary, in the orange and red arenas it is very difficult to encounter such features. Second, occupancy grids are easy to implement and can produce easy to use information without the need of an iterative procedure. The mapping procedure uses the input coming from the odometry system and from the range finder sensor. It is worth recalling that the *odometry system* does not just integrate the motors output, but also takes into consideration the input coming from the magnetic compass. This gives us the possibility to bound the orientation error which would arise from pure integration. Preliminary lab runs outlined that odometry works pretty well as long as the robot moves along a straight line, but goes significantly bad while turning. This is no surprise, as in order to turn in place our platforms rely in the skidding of the wheels or tracks. Thus, when the robot turns, odometry data coming from the encoders are corrected with the orientation coming from the compass. This matter of localization is an important aspect, as occupancy grids based approaches rely on the assumption of precise pose estimation. We are aware this simple schema for bounding such errors is doomed to accumulate large errors in a long run. On the other hand, it turned out to work acceptably during the competition. Single runs are a few minutes long and the robots do not travel for long distances.
The occupancy grid accounts for three different kinds of information, namely obstacle detected, free area detected, and no information available. This information is expressed in terms of *beliefs*. The robot starts with a completely empty grid. At every time step, the input from the range scanner is acquired. By combining this with the actual pose (x,y and orientation θ) coming from the odometry measurement subsystem, it is possible to update the beliefs of the covered grid cells. Technically, every grid cell holds an integer value, initially set to 0. This means that no information is available for that grid cell. When an obstacle is detected in the grid cell, the value is incremented. When the grid cell is determined to be free, such value is decremented. Both increments and decrements are bounded. This means that such beliefs cannot arbitrarily grow or decrease when the robot is standing at a fixed position (similar to what happens with [12]). This to take into account the possibility that some parts of the world can change, and to give a chance for updates. For example, if a door opens, after a certain number of updates such obstacle disappears from the map (see figure 3).

The promptness of such update is influenced by the increase/decrease values and by the bounds. During the competition we set the bounds to be 250 and the increments/decrements were set to 10. This means that after 50 readings a belief can be completely reversed. As data are coming from the range sensor at about 3Hz, this means that the complete update can take place in less than 20 seconds. While this could seem a big amount of time, it is worth noting that during the competition most of the robots move very slowly. A more prompt update could take place by changing the outlined values, but this comes at the



(a) (b)

Fig. 3. Figure a shows the map built by the robot while mapping a corner of the laboratory. Figure b illustrates the map produced after the door facing the robot has been closed and a few cycles elapsed. Each square has an edge of one meter.

cost of decreased map precision.
Algorithm 1 provides the algorithmic sketch of the procedure. In order to find out if a cell is free or not (lines 5 and 11), we remind that once the position and orientation are known this is a matter of simple geometric computation (see figure 4).

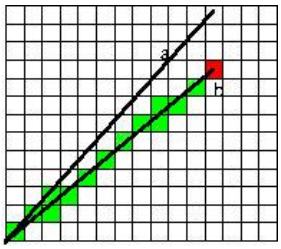


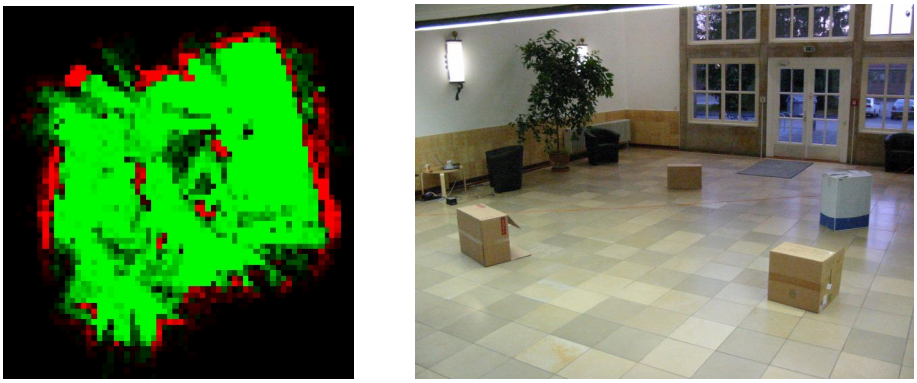
Fig. 4. Once the position and the orientation of the robot are known it is easy to determine which grid cells can be considered occupied or free. If the distance returned by the sensor is considered unreliable (i.e. greater than three meters), it is completely discarded (upper ray). If it is reliable, (lower ray), it can be used to mark free and occupied cells.

Figure 5 illustrates the mapping of the main hall of our research building. It can be appreciated how the robot mapped the shape of the room, as well as the four boxes placed around.

During the Robocup competition the grid resolution was set such that each square was 25 cm wide, while the overall map was set to be 10 by 10 meters and

Algorithm 1 Mapping procedure

```
1: Initialization: fill the grid map with 0s
2: loop
3:    Get data from scanner: vector v of MAX_READINGS distances
4:    Get x,y and θ from odometry
5:    for n ← 1 to MAX_READINGS do
6:       if v[n] < RELIABLE_DISTANCE then
7:          Let G[i][j] be the corresponding occupied grid cell (computed from x,y,θ
            and v[n])
8:          if G[i][j] < MAX then
9:             Increase G[i][j]
10:         end if
11:         for all intermediate free cells G[i][j] do
12:            if G[i][j] > MIN then
13:               Decrease G[i][j]
14:            end if
15:         end for
16:      else
17:         Discard v[n]
18:      end if
19:   end for
20: end loop
```



(a) (b)

Fig. 5. On the left, an example of map. Green points indicate detected free grid cells, while red points indicate detected obstacle cells. Black points indicate that no information is available. The intensity of colors is proportional to strength of the belief. On the right you can see a photo of the hall being mapped.

the robot was assumed to be placed initially in the middle point of such square. This because the robot was supposed to enter different arenas from different sides. The software we developed builds the map on the onboard robot PC and offloads it to the control station where the operator is teleoperating it. There the operator has a complete control of all the robot's sensors, i.e. range finders, cameras and so on. The operator can then put some notes on the map, for example to mark where victims are, or where specific environmental features are placed. This, again, to make easier and safer the work of the rescue team which will use the map to enter the explored building. In addition, the software also keeps track of the path followed by the robot while exploring the arena, thus providing also a sort topological information (see figure 6).

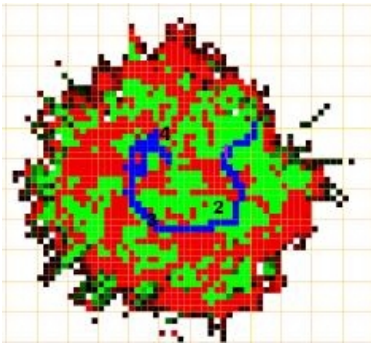


Fig. 6. One of the maps produced while mapping the yellow arena. The blue points indicate the path followed by the robot. The numbers indicate the location where victims have been located. The operator can also associate a brief textual description to each identified victim or feature, to provide additional information to the rescue team.

It can be observed that the quality of the maps obtained during the competition is below what we got while mapping the our research building. This is mainly due to the fact that the arenas are composed by movable glass like panels which often spread false reflection or turns out to be transparent to the range finder.

4 Conclusions and future work

We illustrated our approach in developing a robotic system able to autonomously build maps of the rescue arenas. Although is the robot is teleoperated, the mapping software runs on the robot itself and offloads the map to the operator base station, where the operator can put annotations on the map to make it more useful. While the approach is extremely easy in itself, it represents the first completely autonomous mapping software appeared in the competition. The representation used for the map is an occupancy grid.

In the future we will develop our research along two lines. First, we plan to implement and evaluate some of the classical algorithms developed for mapping. While we expect them to be clearly superior to our approach in terms of quality of the produced map, we are interested in seeing how they will perform under the competition constraints. Additionally, we will investigate the problem of mapping three dimensional environments. In fact, while negotiating collapsed areas in damaged buildings, one can expect the robot to climb over slopes, stairs and so on. In that case, a three dimensional map is clearly needed as the planar one could be not only imprecise, but also misleading for the rescue team. This problem calls for the fusion of different sensors as well as for more efficient mapping algorithms as the dimensionality of the environment being mapped significantly grows. In this direction we are currently evaluating the opportunity to use inertial sensors and 3D cameras for getting depth information.

References

1. Kitano, H., Tadokoro, S.: Robocup rescue. a grand challenge for multiagent and intelligent systems. AI Magazine **22** (2001) 39–52
2. Takahashi, T., Tadokoro: Working with robots in disasters. IEEE Robotics and Automation Magazine **9** (2002) 34–39
3. Osuka, K., Murphy, R., Schultz, A.: Usar competitions for physically situated robots. IEEE Robotics and Automation Magazine **9** (2002) 26–33
4. Birk, A., Carpin, S., Kenn, H.: The iub 2003 rescue robot team. (this volume)
5. Hokuyo Automation Co.: http://www.hokuyo-aut.jp/
6. Kenn, H., Carpin, S., Pfingsthorn, M., Liebald, B., Hepes, I., Ciocov, C., Birk, A.: Fast-robots: A rapid-prototyping framework for intelligent mobile robots. In: Proceedings of the Third International Conference on Artificial Intelligence and Applications. (2003) 76–81
7. McLachlan, G., Krishnan, T.: The EM Algorithm and Extensions. Wiley-Interscience (1996)
8. Kalman, R.: A new approach to linear filtering and prediction problems. Transactions of ASME. Journal of Basic Engineering **83** (1960)
9. Dissanayake, G., Newman, P., Clark, S., Durrant-Whyte, H., , Csorba., M.: A solution to the simultaneous localisation and map building (slam) problem. IEEE Transactions of Robotics and Automation **17** (2001) 229–241
10. Moravec, H.P.: Sensor fusion in certainty grids for mobile robots. AI Magazine (1988)
11. Thrun, S.: Robot mapping: a survey. Technical Report CMU-CS-02-111, Carnegie Mellon University (2002)
12. Biswas, S., Limketkai, B., Sanner, S., Thrun, S.: Towards object mapping in dynamic environments with mobile robots. In: Proceedings of the IEEE/RSJ International Conference on Intelliegent Robots and Systems. (2002) 1014–1019