SPQR-UPM Team Description Paper

Gabriele Randelli[1], Alberto Valero[2], Paloma de la Puente[2], Daniele Calisi[1], and Diego Rodríguez-Losada[2]

[1] Department of Computer and System Sciences,
Sapienza University of Rome,
Via Ariosto 25, 00185 Rome, Italy
<name.lastname>@dis.uniroma1.it
[2] División de Ingeniería de Sistemas y Automática,
Universidad Politécnica de Madrid,
Calle Jose Gutierrez Abascal 2, E-28006 Madrid, Spain

Abstract. In this paper we describe the technical characteristics of the rescue system developed by the SPQR-UPM Rescue Virtual Team for the RoboCup 2010 Virtual Robots competition. SPQR-UPM is a joint team between Sapienza University of Rome and Universidad Politécnica de Madrid. We will analyze the whole architecture, focusing on the new provided features, and we will show some preliminary results of a set of experiments, analyzing the optimal ratio operator/robot in USAR missions.

1 Introduction

In this paper, we present an overview of the robotic system adopted by the SPQR-UPM Rescue Virtual Team for the RoboCup Rescue 2010 competitions. The two involved groups are the Cognitive Cooperative Robotics Laboratory[3] at Sapienza University of Rome, and the Intelligent Control Group[4] at Universidad Politecnica de Madrid. Both groups share an interest in mobile robotics. The Italian research team focuses on autonomous exploration and navigation, and have also worked on SLAM 2D. Furthermore, it has been involved in RoboCup competitions since 1998 in different leagues (Middle-size 1998-2002, Four-legged since 2000, Real-rescue-robots 2003-2006, Virtual-rescue since 2006 and @Home in 2006). In 2007 the SPQR team got the third place in RoboCup Rescue Virtual Robots League in Atlanta (USA). The Spanish group is mostly focused on SLAM techniques, and is currently working on 3D localization and mapping.

We will show the robotic platforms with which the system has been evaluated. It will be presented the robotic framework, used for the code development and a set of techniques for mapping, localization, motion, etc. From this, we will define the taxonomy of the interaction system.

In the rest of this document, we will describe the overall system architecture, focusing on the adopted techniques for mapping, localization, motion, etc. We

[3] http://labrococo.dis.uniroma1.it/
[4] http://www.intelligentcontrol.es

will also introduce our interaction system, and our rescue interface. Finally, we will briefly analyze some applications of our system in real application fields.

2 Robotic Team and Software Architecture

2.1 Team Configuration

Our robotic team is quite heterogeneous, as it is composed of:

Simulated Rotolotto P2AT. It has the same specifications of the real Pioneer 2AT robot, and reproduces our real Rotolotto robot (see Figure 1(a)).

Simulated Nemo P3AT. Its base is the P2AT standard platform. We added an tilting device already existing on USARSim (ScannerSides) to the P2AT model. Unfortunately, this configuration has some drawbacks, which our real system does not have. In fact, it cannot provide feedback about the value of the tilt angle, and the tilt angle can be controlled only with reference to a final angle. As for the former aspect, we solved it by using small angle increments and waiting long enough to be sure that the command has been implemented. Concerning the second problem, we developed a new pan/tilt device, as will be discussed in the next section (see Figure 1(b)).

Simulated ATRV3D. Odometry is quite imprecise and the robots position cannot be accurately corrected using only odometry. Our 3D mapping algorithm is not yet sufficiently developed to localize the robot while performing a 3D scan. Due to this situation, the simulated P3AT could conduct the 3D scans only while the robot was stationary, since for building the point clouds we need to have ascertained the position of the robot. This presents no problem for evaluating the 3D mapping algorithm, but for exploration purposes this way of working is too slow. We sought the ability to achieve both 2D localization and 3D data acquisition at the same time. The solution was to use two laser scanners, one fixed and the other tilting. Since a P3AT cannot afford to carry two SICK range scanners without compromising the dynamics of the platform, we decided to mount them on an ATRV platform, which we called ATRV3D.

Moreover, as the ScannerSides device existing in USARSim did not match the characteristics of the PowerCube device, we decided to build a pan/tilt device emulating the real PowerCube070 wrist. We built a laser pan-tilt mission package which consisted of two rotational joints. These joints can be controlled independently with angle, speed and torque, and they provide feedback about their position. On one occasion, the SICK range scanner was mounted on top of such a device. With this configuration, we were able to acquire 3D data in pretty much the same way as with our real robot, while the position of the robot was corrected by the usual SLAM techniques using the 2D laser. The ATRV3D can be seen in Figure 1(c). A more detailed description of both the simulated Nemo and ATRV3D can be found in (11).

If available during the competitions, we will also deploy one unmanned aerial vehicle (UAV) with an INS sensor, GPS, and Victim Sensor.

(a) Simulated Rotolotto P2AT (b) Simulated Nemo P3AT (c) Simulated ATRV3D

Fig. 1: Team Configuration.

2.2 OpenRDK

The entire robotic software system has been developed through an open source robotic framework called OpenRDK (Open Robot Development Kit). This framework has been elaborated in the last few years by the Cognitive Cooperative Robotics Laboratory of La Sapienza and was recently adopted by the Intelligent Control Group of Madrid. It can be found on SourceForge[5]. To date, OpenRDK has been successfully applied in diverse applications with heterogeneous robots including: wheeled mobile platforms, tracked vehicles, unmanned aerial vehicles, legged robot, as well as different robotic simulators (e.g. Player/Stage, USARSim, Webots).

Some of the features of OpenRDK are:

- An agent is a list of modules that are instantiated, together with the values of their parameters and their interconnection layout. This agent is initially specified in a configuration file.
- Modules communicate using a blackboard-type object, called a "repository" (see Figure 2), in which the modules present some of their internal variables (parameters, inputs and outputs), called "properties." A module defines its properties during initialization; after that, it can access its own and other modules data, within the same agent or from other ones, by means of a global URL-like addressing scheme. Access to remote properties is transparent from a module perspective; on the other hand, it boils down to shared memory (OpenRDK provides easy built-ins for concurrency management) in the case of local properties.

Each module typically implements a task or behavior, like scan-matching, mapping, path-planning, and so forth. All these entities are implemented in the OpenRDK core and usually all a developer is requested to do is to create a new module. This is a very easy task indeed (from the point of view of the framework; he can thus concentrate on the real problem, without having to expend

[5] http://openrdk.sourceforge.net

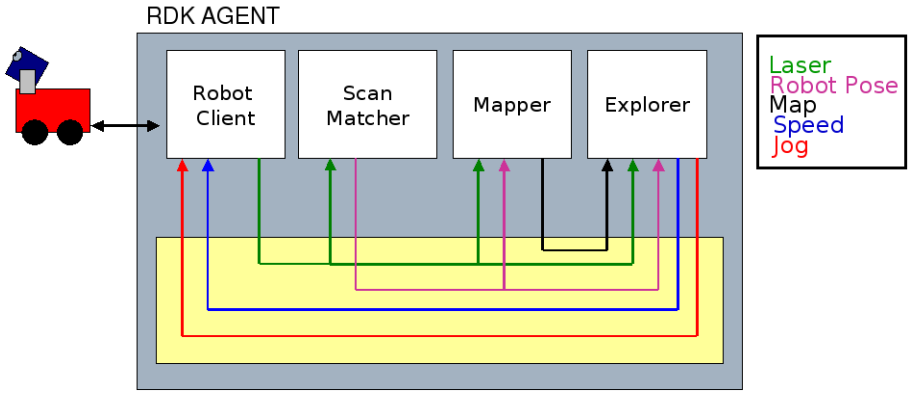


Fig. 2: RDK Agent Example

too much attention on the framework). The ensemble of all the modules required constitutes the agent.

Every agent will usually have:

- A module for communicating with the robot: simulated or real,
- a set of modules for processing the sensor data, and
- a set of modules that uses the processed data to command the robot.

Figure 2 shows an example of an agent that communicates with a robot. The process is as follows: The agent retrieves the laser data. The scan matcher module localizes the robot. The mapper module constructs a map, and then the exploration modules move the robot according to the map and to the robots position.

3 Navigation System

In this section, we will present the architecture of our robotic navigation system. This architecture has been implemented for OpenRDK. The system follows the classical layered structure, in which each layer receives an input from the previous layer, process it, and generates an output for the next layer. This structure can be seen in Figure 3. As we have already said above, a module has a certain number of inputs, which it processes to generate an output. A mapping module will have the laser readings and the robot position as input, while it will output a map. An exploration module will receive the map and robot position and will give a target point as output. For its part, path-planning will take this target goal, together with the map, and will use both to process a safe path to reach the goal.

In this structure, the HRI layer is in charge of the interaction among the operator and the robot.

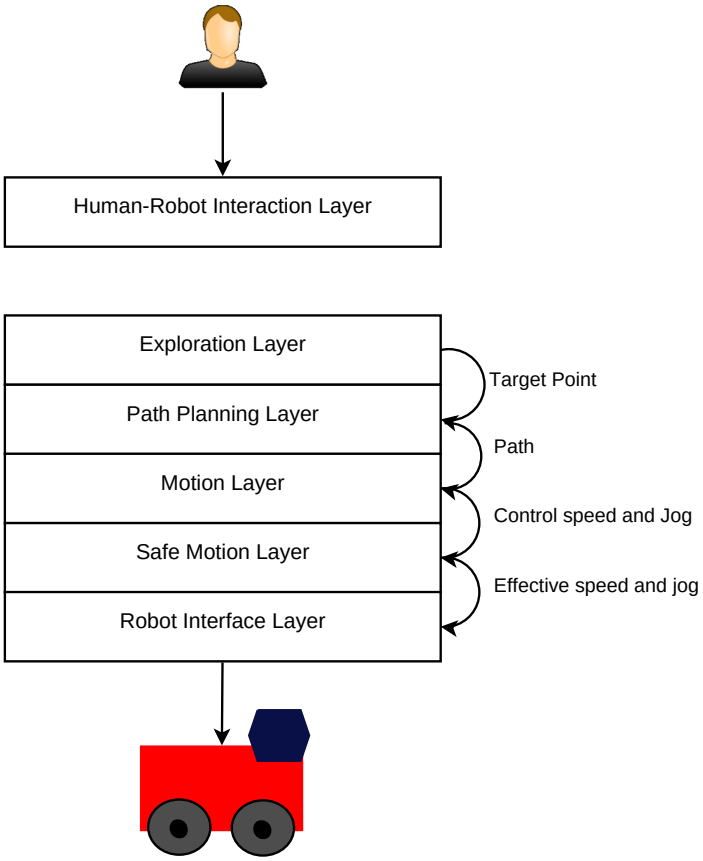


Fig. 3: System Architecture

– It receives inputs from the interface, that sends to the corresponding system layer.
– It receives the outputs of each layer: target point, path, real speed, errors; and sends them to the interface.
– It manages the autonomy adjustment.

These modules include the majority of the up-to-date techniques used nowadays in mobile robotics. Some of them have been developed as part of the group research, while others are those present in literature.

3.1 Exploration Layer

The exploration task depends on the goal of the mission. One may wish to build a map of an unknown environment, in which case one needs only to maximize

the area covered while maintaining the consistency of the map. Or one may want to deploy a team of robots in order to create an ad-hoc communications structure, or to search for victims in a disaster area. Depending on the goal, the exploration algorithms will vary accordingly.

The latest research of our groups has focused on exploration for search and rescue robot missions. "Exploration and search" is a typical task for autonomous robots acting in rescue missions; specifically, it concerns the problem of exploring the environment while searching for interesting features within it. We model this problem as a multi-objective exploration and search problem. In particular, we use a State Diagram formalism that enables the representation of decisions, loops, interruptions due to unexpected events or action failures within an otherwise coherent framework. While autonomous exploration has been widely investigated in the past, we have focused on the problem of searching for victims in the environment during the map building process (1)(2).

Our current exploration system works as an state machine. It can be seen in Figure 4.

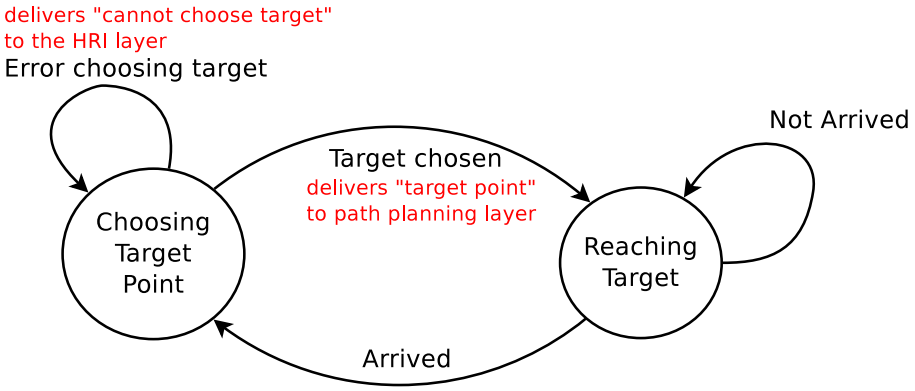


Fig. 4: Exploration Layer

3.2 Path Planning and Motion Layers

The Path Planning and Navigation Layers have been developed within the Cognitive Cooperative Robotics Laboratory of Rome. They are responsible for moving the robot to a given target point aided by a 2D grid map. They are implemented implemented using the well-known two-level decomposition technique in which a global algorithm computes a path towards the goal (path-planning layer), using a simplified model of the environment; this path is followed by a local algorithm (navigation layer), which then generates the motion commands to steer the robot to the current goal. The global algorithm computes a graph that models the connectivity of the environment, modifying this model in accord

with how the robot perceives information about the unknown environment: the path computation is thus reduced to a path search in a graph (3). The local algorithm uses a variation of the Dynamic Window Approach (4). The resulting trajectories are very smooth, thanks to the fact that the DWA generates only commands and trajectories that can be followed by the robot, with kinematic and dynamic constraints being taken into account.

The diagrams representing their working can be seen in Figures 5 and 6

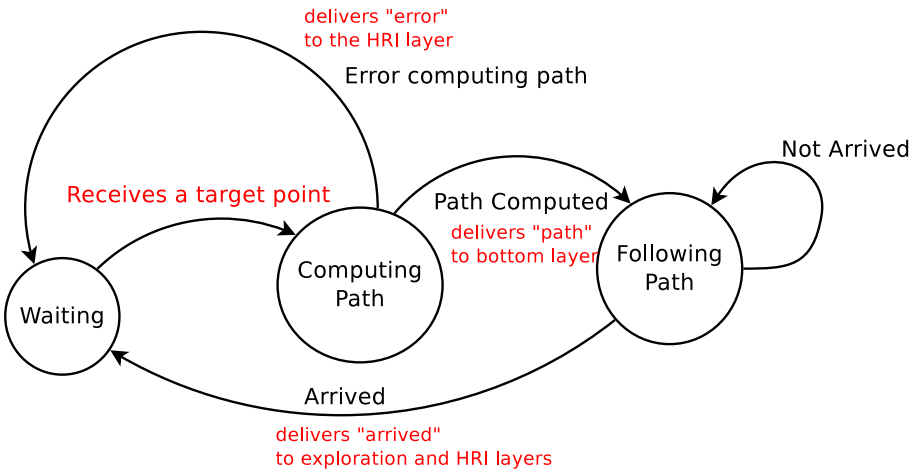


Fig. 5: Path Planning Layer

3.3 Safe Motion Layer and Robot Interface Layers

The safe motion layer is in charge of avoiding collisions. It parse the command speed and jog that receives as inputs according to the sensed obstacles, producing the final speed and jog speeds.

The robot interface layer manage the communications with the robots.

4 Localization and Mapping

All these layers work on the basis of a map where the robot is located. This involves that the system must localize the robot and build the map using the sensed data. Currently the navigation is based on a grid map. In this map it is included the 2D map; 3D information: obstacles, holes and ground detection; and semantic information.

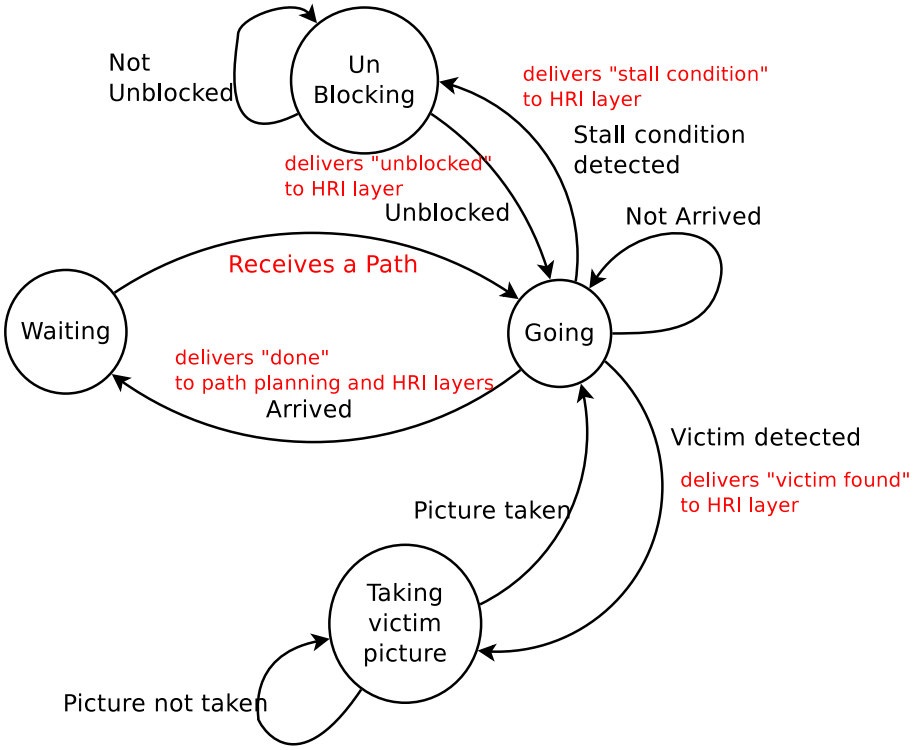


Fig. 6: Motion Layer

4.1 2D Localization and Mapping

In the course of our group research, different localization techniques have been developed. These include scan matching techniques, obtaining grid maps, and feature-based localization and mapping for obtaining maps of characteristics. All of these techniques use the data provided by the laser range scanner and the odometry function.

The most recent feature-based technique involved here was developed within the Intelligent Control Group of the Universidad Politecnica de Madrid. It provides a novel solution to the Simultaneous Localization and Map building (SLAM) problem for complex indoor environments, using a set of splines for describing the geometries detected by the laser range finder; this range finder was mounted on the mobile platform. The maps obtained are more compact that the traditional grid maps, as they contain only geometric information on the environment described with splines (8)(7). This map representation reduces considerably the amount of information exchange with the interface.

A localization and mapping algorithm based on scan matching was developed within the Cognitive Cooperative Robotics Laboratory of the Sapienza University in Rome (6)(5). The software was distributed under the name of "*GMap-*

ping" and it is among most popular algorithms used for laser based SLAM for mobile robotics. The author of GMapping has also produced an OpenRDK implementation.

4.2 3D Mapping and Ground Detection

In the Intelligent Control Group in Madrid, de la Puente is currently working on full 3D mapping and localization. To date, she has developed a feature-based 3D mapping approach with a view to obtaining compact models of semi-structured environments, such as partially destroyed buildings where mobile robots are called upon to carry out rescue activities. In order to gather the 3D data, we use a laser scanner, employing a nodding data acquisition system mounted on both real and simulated robots (10)(9). The 3D map is built up from 3D point clouds. The point clouds are extracted from the laser range scanners. Each scan has a different tilt angle covering a different 3D area. An example of the sort of 3D map constructed in this manner is shown in Figure 7.

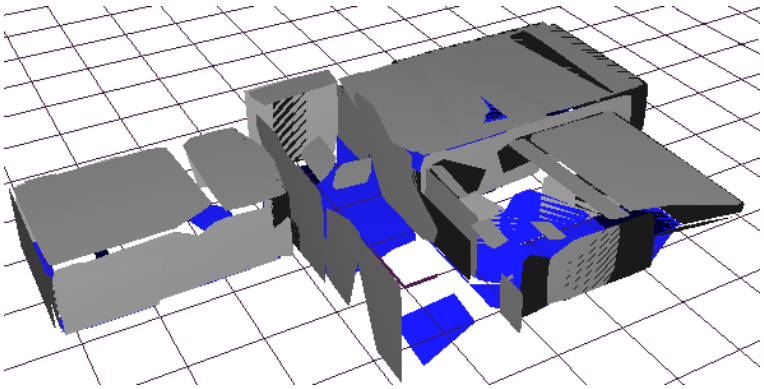


Fig. 7: 3D map obtained with the simulated Nemo Robot on USARSim

Moreover, a ground detection algorithm has been developed, in order to detect navigable areas safe for mobile robots (11). The navigable areas are displayed on a 2D grid map that can be sent to the operator interface or processed by the path planning and motion modules.

5 Communications Management

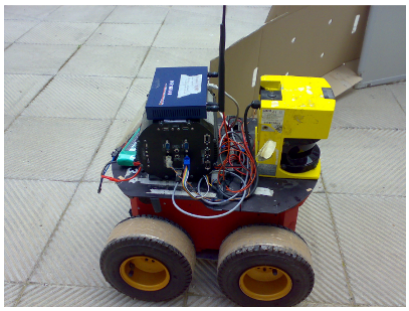
We developed a MANET (mobile ad hoc network) infrastructure to allow for robust and fault-tolerant communications among the robotic agents. A MANET

[5] http://www.openslam.org

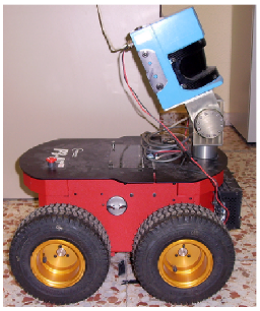
is constituted by mobile devices that communicate each other via wireless links, without relying on an underlying infrastructure. This distinguishes them from other types of wireless networks as, for example, cell networks or infrastructure-based wireless networks. To achieve communications in a MANET, each robot acts as an endpoint and as a router forwarding messages to the devices within radio range, to communicate all the information retrieved by the robots to the Base Station.

6 Practical Application to Real Rescue Robots

The whole system has also been implemented and tested on real robot units. First of all, we validated our system on a P2AT robot, Rotolotto, equipped with a SICK LMS200 laser range scanner, available at the Cooperative Cognitive Robotics Laboratory in Rome (see Figure 8(a)). Furthermore, we tested our 3D mapping system at the Intelligent Control Group at Madrid, using a Pioneer 3AT robot (Nemo) by MobileRobots.Inc. Nemo is equipped with a SICK LMS200 laser range scanner, with a servo pan-tilt PowerCube Wrist 070 unit, by Amtec Robotics (see Figure 8(b)).



(a) Real Rotolotto P2AT



(b) Real Nemo P3AT

Fig. 8: Real P2AT and P3AT robots.

Bibliography

[1] Calisi, D., Farinelli, A., Iocchi, L., Nardi, D.: Multi-objective exploration and search for autonomous rescue robots. Journal of Field Robotics, Special Issue on Quantitative Performance Evaluation of Robotic and Intelligent Systems 24, 763–777 (August - September 2007)

[2] Calisi, D., Iocchi, L., Nardi, D., Randelli, G., Ziparo, V.: Improving search and rescue using contextual information. Advanced Robotics 23(9), 1199–1216 (2009)

[3] Censi, A., Calisi, D., De Luca, A., Oriolo, G.: A bayesian framework for optimal motion planning with uncertainty. In: Proc. of the IEEE Int. Conference on Robotics and Automation (ICRA). pp. 1798–1805 (May 2008)

[4] Fox, D., Burgard, W., Thrun, S.: The dynamic window approach to collision avoidance. vol. 4 (1997)

[5] Grisetti, G., Tipaldi, G.D., Stachniss, C., Burgard, W., Nardi, D.: Fast and accurate slam with rao-blackwellized particle filters. Robotics and Autonomous Systems (2006), in press

[6] Grisetti, G., Tipaldi, G.D., Stachniss, C., Burgard, W., Nardi, D.: Speeding-up rao-blackwellized slam. pp. 442–447. Orlando, FL, USA (2006)

[7] Pedraza, L., Rodriguez-Losada, D., Matia, F., Dissanayake, G., Miro, J.: Extending the Limits of Feature-Based SLAM With B-Splines. IEEE Transactions on Robotics pp. 353–366 (April 2009)

[8] Pedraza, L., Rodríguez-Losada, D., Segundo, P.S., Matía, F.: Building maps of large environments using splines and geometric analysis. In: 2008 IEEE/RSJ International Conference on Intelligent Robots and Systems. pp. 1600–1605 (2008)

[9] de la Puente, P., Rodríguez-Losada, D., López, R., Matía, F.: Extraction of geometrical features in 3d environments for service robotic applications. In: Hybrid Artificial Intelligence Systems. pp. 441–450 (2008)

[10] de la Puente, P., Rodriguez-Losada, D., Valero, A., Matia, F.: 3D Feature Based Mapping towards Mobile Robots Enhanced Performance in Rescue Missions. In: 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems (2009)

[11] de la Puente, P., Valero, A., Rodriguez-Losada, D.: 3D Mapping: testing algorithms and discovering new ideas with USARSim. In: Robots, Games, and Research: Success stories in USARSim. IROS 2009 (2009)