FC Portugal: Search and Rescue in Urban Catastrophes

João Certo[1,2,3], Nuno Cordeiro[2,3], Luís Paulo Reis[2,3], Nuno Lau[1,4]

joao.certo@fe.up.pt, nuno_cordeiro@hotmail.com, lpreis@fe.up.pt, lau@det.ua.pt

[1] IEETA – Institute of Electronics and Telematics Engineering of Aveiro, University of Aveiro

[2] LIACC-NIAD&R– Artificial Intelligence and Computer Science Lab, University of Porto

[3] FEUP – Faculty of Engineering of the University of Porto, Rua Dr. Roberto Frias, 4200-465 Porto, Portugal

[4] DETI –Informatics, Electronics and Telecommunications Dep., University of Aveiro, Campus de Santiago, 3810-193 Aveiro, Portugal

Abstract: This paper presents an overview of the RoboCup Rescue Simulation League and the rescue team FC Portugal. RoboCup Rescue Simulation is an international joint project that promotes research on distributed artificial intelligence and intelligent robotics. It offers a comprehensive urban disaster simulator and a competitive evaluation for researchers. This paper explains the objectives of the league, the mechanics of the simulator system and the main challenges in the research conducted using the RoboCup Rescue simulator. In this ever evolving and extensive urban disaster simulator, heterogeneous teams of agents try to minimize damage to both people and urban property after the occurrence of an earthquake. The city is filled with burning buildings, civilians trapped under debris and blocked roads. Teams of simulated fire brigades, policeman and ambulances collaborate in order to face the disaster. In this context, an overview of FC Portugal's rescue team is presented, including the major coordination algorithms and implemented agent's behaviours. Some of the most interesting league problems are discussed and FC Portugal's solutions are introduced. Results achieved in international competitions are also briefly presented, with emphasis on the last European championship, held in 2006, in Eindhoven, which FC Portugal won.

Keywords: Information Systems Planning and Management. Intelligent Agents. Interworking Heterogeneous Agents. Simulation Systems.

1. Introduction

The Rescue Simulator is an environment for Information Systems (IS), Artificial Intelligence (AI), Multi-Agent Systems (MAS) and Intelligent Robotics (IR) research. The concept of Multi-Agent Systems evolved from Distributed Artificial

Intelligence (DAI), Distributed Problem Solving (DPS) and Parallel AI (PAI). As for a single intelligent agent, it can be defined as a computational entity, usually called software, that if placed in some environment, perceives it through sensors, and is capable of performing autonomous action, in order to meet its design objectives, using its actuators (Reis, 2003).

RoboCup was created as an international research and education initiative, aiming to foster research in (distributed) artificial intelligence and (intelligent) robotics research, by providing standard problems, where a wide range of technologies can be examined and integrated. With the objective of dynamizing the evolution of AI, IR and in particular MAS, the project was launched by Hiroaki Kitano, a Japanese AI researcher that became founder and president of the RoboCup Federation. It is currently divided in three major categories: soccer, rescue, and junior; each with its different leagues. Due to its prominence, soccer was the main motivator behind RoboCup. Being an extremely popular sport across most of the globe, it is able to attract people from different countries, cultures and religions into the same competition. Furthermore, it presents interesting scientific challenges, mostly because it is a team game, mingling individual efforts with collective strategy.

RoboCup Rescue Simulation is an international joint project (Kitano et al., 1999) that promotes research on distributed artificial intelligence and intelligent robotics in a socially more useful scenario than soccer. The project was started in 1999 to solve rescue problems, by integrating disaster information, prediction, planning, and training, for rescue actions. Built upon the success of RoboCup Soccer project, it aims to offer a comprehensive urban disaster simulator, forums of technical discussions and competitive evaluation for researchers and practitioners (Reinaldo et al., 2005). Through the use of an extensive, and ever evolving, urban disaster simulator, heterogeneous team agents try to minimize damage to both people and property. Burning buildings, Civilians trapped under debris, and blocked roads, are just some of the challenges simulation Rescue Teams (RTs) must overcome, coordinating as many as up to forty agents of six different types (Paquet, Bernier, & Chaib-draa, 2004).

Every year, a RoboCup international competition is organized, where, in a competitive but constructive environment, researchers from all over the world can test their agents against other RTs. By comparing approaches and exchanging ideas, progress is made at an amazing rate, in great part due to the open source nature of the project. After each competition, the source code for every team is released, so that work may be done on top of the best ideas and implementations. Following on that concept, team FC Portugal entered the Rescue project, determined to contribute to the community. The agents' base work was built on top of the code developed by SOS, a reputed RT from Iran. At the top of the RoboCup Rescue agents, strategic and coordination methodologies from the RoboCup Soccer are being used. The constant evolutions in the simulator package imply that a lot of effort is required simply to adapt rescue agents to new environment rules. FC

Portugal Rescue team is the result of a cooperation project between the Universities of Porto (LIACC/NIAD&R Lab) and Aveiro (IEETA Lab) in Portugal.

The rest of the paper is organized as follows. Section 2 presents RoboCup Rescue simulator, its main modules and the associated viewer system. Section 3 describes the process of construction of a RoboCup Rescue team and the main methodologies and algorithms needed for this task. Finally, section 4 presents some simples results and the conclusions of the paper.

2. The Simulator System

The Rescue simulator is a simplified model of a city - only data relevant to the disaster situation is reproduced, while most detail is neglected. The simulator package uses a modular approach, allowing different parts to be updated independently. Every year new features are combined with the existing ones, improving the simulation and adding complexity to the environment. The most recent large change was in the fire simulator, which was completely overhauled, requiring some changes in the agents' strategy.

The action takes place in a simulated city, where a natural disaster (earthquake) has just taken place. This city is dynamically modeled by the following equations:

$$e(t) = f(x(t),u(t),t) \quad \text{Eq. (1)}$$

$$x(t + \Delta t) = g(x(t),e(t)) \quad \text{Eq. (2)}$$

- $e(t)$ represents the effects that create change in the city, calculated by f.
- t represents the current time instant.
- f is the function describing how $x(t)$ e $u(t)$ affect the simulated world, changing its status.
- $x(t)$ is the status variable. It represents the disaster situation in instant t. Every variable such as the strength of fire or the speed of cars is saved in the form of a vector. The size of this vector proportionally increases with the size of the simulated area.
- $u(t)$ is the input vector in instant t, representing external effects like water sprayed by Fire Brigades and debris removed by Police Forces.
- Δt is the time step used to forward the simulation discretely.
- g is the function that describes the values of status $x(t)$ at the instant immediately after t, i.e. instant $t + \Delta t$ (Committee, 2000).

So, at t=0, $x(t)$ represents the initial situation. From $x(0)$ the following values can be obtained:

- Sint: total HP of all agents at start.
- Bint: total undamaged area at start.

At any time step those values can be obtained:

- P: number of living agents.
- S: remaining total HP of all agents.
- B: total undamaged area of buildings.

The simulation score V is calculated using the following equation:

$$V = \left(P + \frac{S}{Sint} \right) * \sqrt{\frac{B}{Bint}} \qquad \text{Eq. (3)}$$

From Eq. (3) it can be perceived that the evaluation rule would be: given any simulation, the higher the V value, the better the rescue operation (Akin, Skinner, Habibi, Koto, & Casio, 2004).

Note that at the beginning of the simulation (t=0):

$$V = (P + 1) \qquad \text{Eq. (4)}$$

In Eq.(4), which results from Eq.(3) at t=0, the initial score is defined as the total amount of agents plus one. As the simulation proceeds, more buildings are damaged and people hurt, causing the score to drop till its final value at t=300.

A schematic representation of the simulation system can be seen on Figure 1.

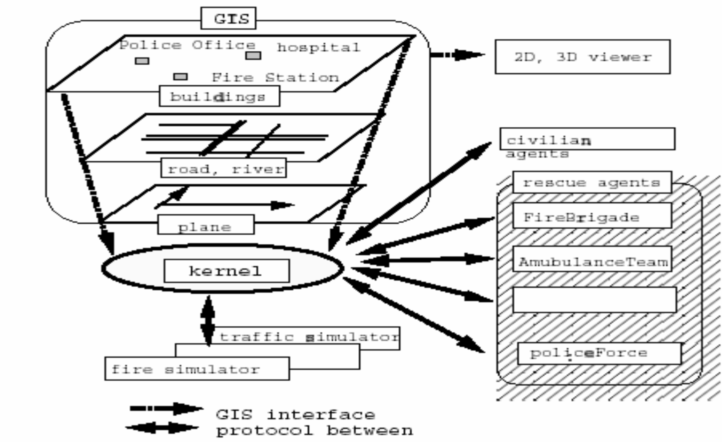

Figure 1 - Simulation System functional outline (Committee, 2000).

This structure allows a relatively autonomous development of the different simulator modules, since once the communication protocol is defined, the modules are mostly independent. The communication between modules takes place by message exchange.

The kernel is the central processing unit of the system, controlling the simulation process and facilitating information exchange between modules. It is responsible for establishing and maintaining communication with the Geographic Information System (GIS), the Simulators (Collapse, Fire, Traffic, etc.), the Viewer, and the Agents; as is depicted on Figure 2.

When the program starts, the Kernel receives from the GIS module the initial configuration of the simulated world. At every step of the simulation, the Kernel sends sensory information to all agents and receives their action commands. Information is sent and received from the modules as necessary and, for each data exchange, the command and information validity is verified (Takahashi, Takeuchi, Tetsuhiko, Tadokoro, & Noda, 2000).

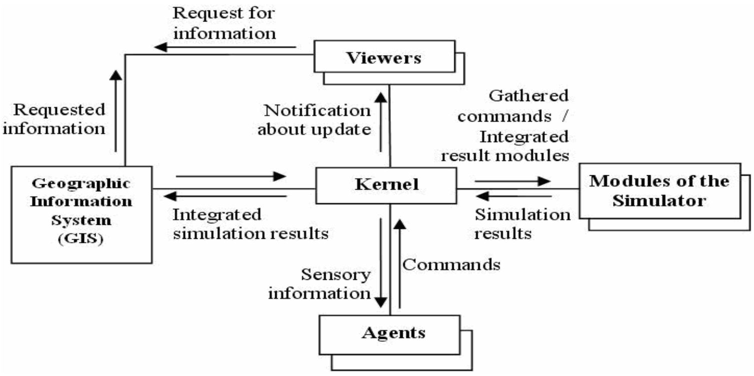

Figure 2 - RoboCup Rescue Simulation System.

The GIS module is responsible for the initial configuration of the simulated world. This is composed by the location and properties of buildings, roads, nodes, refuges, agent centers, Civilians, Ambulances, Fire Brigades, Police Forces and initial fires. It also records the simulation progress into a log file, enabling a detailed offline analysis. Additionally, this module is responsible for feeding data to the viewer.

As the project evolves, simulators are added or improved, deepening the complexity and adding realism to the simulation. The Collapse Simulator module acts on the physical state of buildings after the earthquake. On a large scale disaster like the one RoboCup Rescue aims to emulate, around 80% to 90% of households are at least partially collapsed, shortly after the calamity. Currently, this simulator is triggered only once, at the beginning of the simulation.

The Blockade Simulator module is responsible for defining the state of road obstructions. After the earthquake, a large part of the roads gets blocked, hindering traffic flow. These obstructions may have different causes such as crowds, debris from buildings and traffic accidents. Blocked roads can only be cleared by Police Force agents, this way allowing other agents to freely move through.

Every agent's movement in the world, including Civilians, is modelled by the Traffic Simulator component, which defines the pace allowed on every road section.

Width, number of agents present, and the level of "blockness", are some of the factors affecting maximum speed on a street. Usually, a road which is over 50% blocked is not traversable.

The Fire Simulator module simulates the spread of fire in the city. It is currently one of the most evolved components of the simulator package. Right after the earthquake, some buildings ignite and start radiating heat to nearby structures. This component is responsible for the physical simulation of combustion and heat spread. This is done resorting to an intelligent model in which the temperature of a building is, on the one hand increased, either by the its own combustion or by the radiation waves from neighbouring buildings, and on the other hand decreased, due to the evaporation of water, pumped by Fire Brigade agents. In the simplified combustion model used, the critical factors are temperature and fuel (buildings), with the supply of oxygen being disregarded. When a building's temperature rises above its material's flash point it bursts into flames, as seen on Figure 3. In contrast, when the temperature drops bellow this temperature, its fire is extinguished (Nüssle, Kleiner, & Brenner, 2004).

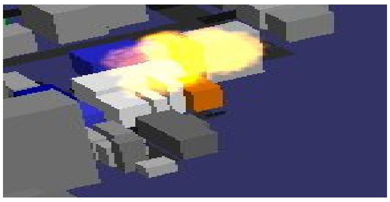

Figure 3 - An image from Freiburg's 3D viewer shows a building on fire.

The agent's status is modelled by the miscellaneous simulator. When an agent is inside a burning building or trapped under debris, its health is affected and starts decreasing. This is the module responsible for controlling the agent's properties in these situations. As a simple example, a large value for the agent's property *buriedness* describes its state as trapped under debris. When Ambulances use their rescue ability, this value is progressively reduced until the agent is free.

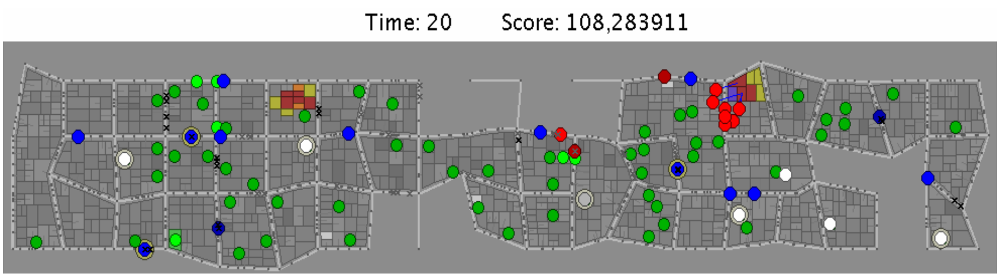

Figure 4 - Morimoto Viewer displaying simulation in RandomMedium map (RoboCup 2005)

A viewer is the graphical interface used to display the actions taking place in the simulated city. The Morimoto 2D Viewer (Figure 4) shows agents as colored circles. Ambulance Teams are white; Fire Brigades red; Police Forces blue; Civilians green and all of them get darker when hurt, turning completely black if they die. Buildings also have different colors, according to their function or status. While refuges are green and Center (agent) buildings are white, those on fire evolve from yellow, to orange, to red. Flooded and extinguished buildings have different shades of blue, while those burnt down are dark grey (almost black). Roadblocks are marked with crosses, and current time and score are displayed on top of the map.

3. Development of a Rescue Team

A RoboCup Rescue Team is composed of three kinds of field agents (Fire Brigades, Ambulance Teams and Police Forces) and their respective Center Agents. Field Agents have a limited eyesight and voice range of 10 and 30 meters respectively. The simulated map area is usually in the order of a few dozens of square kilometers. In this section, the implementation of FC Portugal's agents is discussed.

3.1. Perception, Action and Communication

In a complex domain information is everything and, the better the information is, the better the decisions can be. So, the first step in developing an agent is to properly choose what information it should possess, and how that information is stored. At each time instant after the initial setup, Rescue agents receive sensory information from the Simulator, process it, and send their commands to the simulator. Sensory information can be received in three forms:

- Visual information;
- Field hearing;
- Radio hearing.

Visual information is only sent to field agents. Each agent receives all properties of all objects within the radius of his eyesight. The same is true for field hearing, also exclusive to field agents – each agent receives all voice messages sent by an agent within voice range, with sender identification and contents.

Radio hearing is very different. Although messages are still in the sender/contents format, all team agents (field and center) receive such a message, regardless of distance. The following restrictions apply: field agents can only hear messages from their Center or from agents of the same type; center agents can only hear messages from other Centers or from their field agents of the same kind. The agents' radio communication scheme can be seen in Figure 5.

Sending information works in the opposite way (to hearing). Each field agent has two forms of sending information, voice and radio. Those forms depend on the destination environment.

There are also other restriction limiting communications in size and in number (Akin et al., 2004). Field agents can only receive 4 messages per cycle and Center agents can only receive a maximum 2*n messages, where n is the number of field agents of the Center's type. For example, if there are 10 Fire Brigades, a Fire Station can receive 20 messages per cycle, at most.

As sharing information is vital to agents' performance. FC Portugal uses a series of mechanisms to reduce the amount of information transmitted. One important step is the compression of information. All possible messages between team agents have a number, so that instead of saying "I found a Civilian" a correspondent number is sent. Following on the same line of thought, object properties are always sent in the same order, allowing for succinct information exchange. The next step regards the aggregation of information, which is done by first sending the type of message, then how many of that type (token), followed by ID and properties (as many times as the number of tokens), repeating those steps for all the message types that an agent needs to send. When necessary, information can be split into several messages.

When all agents send messages, a selection must be made on which messages to receive. The solution to this problem is based on the more relaxed number of messages a center can send and hear. So, at each cycle, field agents will ignore any message from other field agents, listening only to the center (as seen in Figure 5, field agents can only listen to their corresponding center). As for the centers, they act as repeaters, listening to information from their agents and from the other centers, and resending it. This way, any information takes at most 3 cycles to reach any other agent (agent1 -> center1 -> center2 -> agent2), so it will be available at the beginning of the fourth cycle.

Sharing all the information would lead to an exponential growth of messages sent and this growth is avoided by choosing to only send new information. FC Portugal use time stamped information. Because of this, it is easy to see if the information sensed is newer than the stored one and, if so, store and retransmit it.

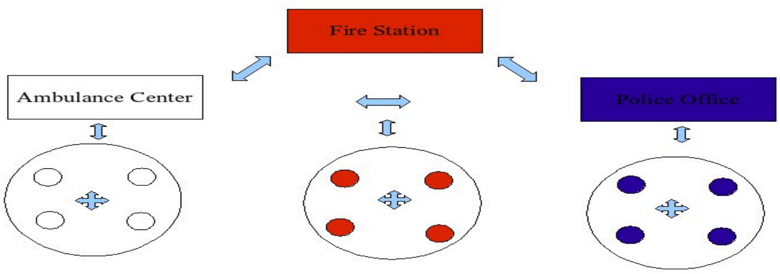

Figure 5 - Radio-communications.

3.2. Ambulance Teams and Ambulance Centers

The main function of an Ambulance Team is to unbury Civilians and take them to a refuge. The strategy for Ambulance Teams follows.

Based on the known Civilian properties, mainly buriedness and health points, Ambulances estimate the time of death and schedule the order in which Civilians should be saved. The time taken in travelling, and whether a path to the Civilian exists, are considered.

The next steps for an Ambulance are: go to the Civilian position; unbury him; load him into the Ambulance; travel to the closest refuge and unload the Civilian to safety, moving on to the next one. However, this behaviour can change due to several events, such as receiving updated or new information about a Civilian, which may change rescue order and priorities. Behavior also changes if there is a fire in a building with a Civilian. Buried field agent will always have higher priority.

In FC Portugal's implementation, Ambulance Teams can form and act as a group. This happens because the action of unburying an agent is cumulative and directly proportional to the number of Ambulance Teams. Therefore, the more Ambulance Teams, the faster the agent will be unburied. There are some exceptions to this tactic. Since the moving cost is considered when estimating the Civilian time of death, sometimes it is more efficient for Ambulances to act individually. Another exception is at the beginning of the simulation given that, due to road blocks, Ambulance Teams cannot reach the same agent, they also act individually.

As Ambulance Teams' calculations are relatively simple, there is no need for further calculations at the Ambulance Center and, as such, its main function is to act as a repeater. It should be kept in mind that Ambulance Teams take several cycles to unbury an agent, which is more than enough time to compute the next to be saved.

3.3. Fire Brigades and Fire Station

Fire Brigade agents are the most complex field agents and their function is to extinguish fires. The strategy for Fire Brigades is the following.

Depending on map size, Fire Brigades are organized into groups - usually two or three. Based on relative positions, size, and proximity to refuges, fiery regions are prioritized, and the ones with the highest priority are assigned, sequentially, to the available groups. Each group then considers its assigned fiery region, and prioritizes the burning buildings (from now on they are called targets) to extinguish, based on relative position in the fiery region, percentage of building area unburned, proximity with building with buried Civilian, amongst other factors.

The next step is to choose a suitable neighbor building that is in the water range of the target. The conditions for this are: the building cannot be on fire; it must be reachable; and as close to the target as possible. If no suitable building is found, then a road near the target is used. The disadvantage to this solution is that the Brigade occupies a lane, which in turn increases the risk of a traffic jam. After moving to the selected building, the Fire Brigade starts watering the target. Note that if for some of the above stated reasons the target changes, and if the new target is in range of the water cannon, the Fire Brigade simply starts watering the new target without requiring a move action.

After some cycles, water in tanks is depleted and, therefore, Fire Brigades go to the nearest refuge to refill their tanks. As this action takes some cycles, when refilling finishes, Fire Brigades reprioritize between the previous assigned region and the currently unassigned ones - the Fire Brigade proceedings are then repeated. Logically, if at some point there are more groups than fiery regions, the available group will be assigned to a fiery region using the remaining criteria.

Akin to Ambulance Teams, when Fire Brigades run out of fires to extinguish, the groups are scattered and the map is divided into cells, so that new fiery buildings may be found. If none is discovered, Fire Brigades start searching for new civilians. Additionally, if all buildings have been explored, Fire Brigades keep on visiting all known living, buried, Civilians in order to update the information on their properties, allowing Ambulances to better estimate the Civilian time of death.

Fire Station agent's main function is to act as a repeater; however, some of the calculations computed by Fire Brigade agents are also computed by the Fire Station. This happens both in the prioritize regions case, and in group assignment. As seen, cycle time is limited and, sometimes, field agents have to rush decisions. Fire Stations do not have this limitation, thus it compares the field agent's solution to its own and, if the Station's solution is better, it is sent to Fire Brigades.

3.4. Police Forces and Police Office

Police Force agents' function is to clear blocked roads. The first strategic decision made is to only clear road blocks at not passable roads. This means that partially obstructed roads will not be cleared, and the reason for this is that they only affect the speed of an agent by halving it. The speed reduction isn't significant, when compared to the time it would take a Police Force to move to that road and remove the partial block.

The main problem for Police Forces has to do with the order in which road blocks are removed. On FC Portugal's implementation there are several ways to do this:

- Clearing blocks until a refuge is reached;
- Clearing blocks from a specific point to a refuge;
- Clear blocks around a refuge;
- Clear a specific path;
- Clear a specific cell.

These possible options are called tasks. Each of these tasks is given a weight, which is a parameter specified in a Police Force configuration file, and can be set for specific maps. All tasks are requested by other agents, with the exception of clearing a specific cell, which is used when no other tasks are requested. When a Police Force receives a task request, it calculates how long each task is going to take, and multiplies it with the weight factor, executing the cheapest one. If the cost of doing a certain task is too high, the task is not considered.

Police force agents are always performing tasks, unlike the other agents that must move themselves to a certain position, in order to perform a certain action. This means that a Police Force agent, when moving, is always clearing impassable blocks in its way, and no detour of blocks is made.

Besides acting as a repeater, the Police Office computes currently assigned and unassigned Police Force tasks. The assigned tasks are multiplied by a reassign coefficient and the most suitable task for each Police Force is chosen, based on relative position to objectives. As one can easily perceive, the Police Office is, when present, the main responsible for Police Force strategy.

3.5. The Map Splitting Problem

Sometimes, to perform certain tasks there is a need to split the map, equally, amongst the simulated field agents of the same kind. In FC Portugal's team this was achieved by calculating the minimum and maximum map coordinates (by comparing apexes of buildings, and node positions) and, with these coordinates, a rectangular form is created and equally divided between the field agents of the desired kind. To that division we call a cell.

The former solution would be good if the road and building density were the same, and uniformly distributed along the map. Some maps are very close to this principle, but most are not. Consequently, some agents could be forced to handle more buildings and roads than others. To resolve this issue, FC Portugal's team has implemented Voronoi diagrams as a new solution to this problem.

A Voronoi diagram is special kind of decomposition of a metric space, determined by distances to a specified discrete set of objects in the space. So, Voronoi cells will be more balanced, as they are created taking into account the density of roads and buildings. Cell attribution is made by a process, starting at setup, the kernel sends all field agents' IDs to all field agents in the same order. Each agent takes up the cell corresponding to his relative position in the initial setup data.

4. Results and Conclusions

The simulator package is an incredible piece of software, but being supported by such an extremely complex environment makes the RoboCupRescue competition prone to a myriad of problems. Issues about which direction to take are constantly debated, and maintaining equilibrium is sometimes difficult. The lack of standardization is also a problem in the RoboCupRescue environment. Since every team uses its own method for code integration, and most codes have little to no modularity, a great part of the benefits of an open source project are lost. As an example, if strict communication standards were to be created, that would enable the Police Force agents from a team to cooperate with Fire Brigades from a second team and Ambulances from another, with little to no modifications of the code. Even though with some problems, the simulator's proximity to reality is notable, streets and buildings have three dimensions, different building materials are in use and some of the sub-simulators like the fire simulator have proven realistic levels.

FC Portugal's rescue simulation team successfully qualified and participated in the world cups RoboCup Osaka 2005 and RoboCup Bremen 2006 and european cup RoboLudens Eindhoven 2006. FC Portugal's rescue team won the last mentioned tournament and is currently the European champion of RoboCup Rescue.

References

Akin, H. L., Skinner, C., Habibi, J., Koto, T., & Casio, S. L. (2004, November 2005). Robocup 2004 Rescue Simulation League Rules V1.01. from http://robot.cmpe.boun.edu.tr/rescue2004/

Committee, R. (2000). *RoboCup Rescue simulator manual v0.4*: The RoboCup Rescue Technical Committee.

Kitano, H., Tadokoro, S., Noda, I., Matsubara, H., Takahashi, T., Shinjou, A., et al. (1999). *RoboCup Rescue: search and rescue in large-scale disasters as a domain for autonomous agents research*. Proc. IEEE Int. Conf. on Systems, Man, and Cybernetics, 1999. IEEE SMC '99, 1999.

Nüssle, T., Kleiner, A., & Brenner, M. (2004). *Approaching Urban Disaster Reality: The ResQ Firesimulator* (No. 200): Univ. Freiburg, Institut für Informatik.

Paquet, S., Bernier, N., & Chaib-draa, B. (2004). DAMAS-Rescue Description Paper. In D. Nardi, M. Riedmiller & C. Sammut (Eds.), *RoboCup-2004: Robot Soccer World Cup VIII* (Vol. 3276): Springer Verlag, Berlin.

Reinaldo, F., Certo, J., Cordeiro, N., Reis, L. P., Camacho, R., & Lau, N. (2005). Applying Biological Paradigms to Emerge Behaviour in RoboCup Rescue Team. In C. Bento, A. Cardoso & G. Dias (Eds.), *Progress in Artificial Intelligence: 12th Portuguese Conference on Artificial Intelligence, EPIA* (LNCS ed., Vol. 3808, pp. 422 - 434). Covilha, Portugal: Springer-Verlag.

Reis, L. P. (2003). *Coordination in Multi-Agent Systems: Applications in University Management and Robotic Soccer*. PhD. Thesis, Univ. Porto.

Takahashi, T., Takeuchi, I., Tetsuhiko, K., Tadokoro, S., & Noda, I. (2000). *RoboCup-Rescue Disaster Simulator Architecture*. Paper presented at the Robocup 2000: Robot Soccer World Cup IV.