RoboCupRescue 2013 – Rescue Simulation League
Team Description
<AUT S.O.S. (Iran)>

Yoosef Golshahi , SalimMalakouti, Seyed Mohammad Reza Modares Saryazdi,
Sina Sheikholeslami, Sina Tagva, Hesam Akbari
MortezaRezayiKhoshdarregi, , NavidHaeri,
AngehAslanian, AramikMarkari

Robotics Research Center, Department of
Computer Engineering and Information Technology,
Amirkabir University of Technology,
No. 424, Hafez Ave.,
Tehran, Iran

{yoosef.golshahi, salim.malakouti}@gmail.com,modaresi@aut.ac.ir
{ssheikholeslami, s_taghva}@live.com {hitax13, m.rezay , navid_it}@gmail.com,
angeh.a2@gmail.com,aramik@aut.ac.ir

http://sosrcrs.com

Abstract. Continuing last years' strategy, for the upcoming year are to work very high level aspects of our team agents, we think that during 7 years of S.O.S and four 1st places in different competitions (including Robocup2009), contribution in RCRS field, we have gained enough information and experience and also having a powerful Agent Base from past years allows us to use and implement and test many high level Strategies and AI methods. We designed different Police force and Search strategies from 2009 and 2010 competitions which are more accurate and we also managed to design a complete and optimize messaging system handling all possible scenarios. We believe Our Reachability recognition method is optimized and is the fastest possible. We have also some suggestions about the behavior of the simulators which are discussed with reasons. as we only describe the most important and high level changes and new aspects of our work for the upcoming year so it's highly recommended to read the previous team description papers of S.O.S. in advance.

1 Introduction

The S.O.S. basic agent and its abilities and skills have been described in previous years TDPs, so this paper is to depict the new strategies added to our plan for Robocup 2012. Following will come the description of these improvements in details. We designed new Strategies for Police force agents and Search methods of all agents each according to team and their individual needs while considering their capabilities which we believe

is more accurate and will have a better response to the strategies of our team in 2009. We also came up with more accurate and optimized (with the server) path finding and move algorithms and we reduced their time complexity considerably. We designed a new well implement and tested reachability recognition method which lets our agent to decide precisely depending on exact information about the map. We also managed to improve our message system strategy used in 2010 handling all possible message scenarios.

2 Agents

We have designed powerful basic agent with main goal of Simplifying and translating the information which server gives us in a way that agents in higher level strategies be able to process them and make better decision. The basic agent produces information needed about reachability of the map, message system, fire zones and etc. Still many algorithms and methods used in our basic agent is the same as algorithms used in previous years basic agents because we believe they were well optimized and tested. Therefor we suggest to review previous years TDPs of S.O.S team to find out these abilities in detail. Still we have used new methods in some functionalities of S.O.S basic agent that will be described in next sections.

3 Agent skills and action selection

Most of agent general skills (i.e. low level abilities) are as it was in previous years, for example the path planning strategy, we use a special version of Dijkstra single source shortest path algorithm [4], using a priority queue implemented by S.O.S. team whose time complexity is $e \log(e)$ where e is the number of edges in the city graph (i.e. roads) and due to the fact that we consider the maximal sequence of roads between two junctions which has no junction inside as a single road the complexity decreases significantly.

Action selection of every agent is through a special architecture which is described in the software architecture section.

3.1 Police Force Team Strategy in 2012

Earth quake causes destruction on roads, buildings and etc. The blockades are the most important reason for malfunction in cooperation between agents and their action commands. Police forces' main goal is to clear roads from blockades so that the other agents will be able to navigate their path through the city faster and without being stopped. Although Police forces' operation does not effect on the score directly but since the performance of the heterogeneous agents depend on Police force agents' achievement, makes these agents to play a key role in the team's work.

We can divide police forces responsibilities to:

- They have to make blocked agents unblock.
- They have to reachable each agent to other important points.
- They have to make sure that fire brigades are able to put out the fire by clearing roads around it.
- They have to make reachable agents to most part of the map.
- They have to take care of requests

- They have to search for civilians and update the information about them so ambulance agents are able to rescue them in the best order.

In addition to states defined in the last TDP, some new normal and interrupt state are defined.

An interrupt state with an effective result is ValuableClearIntterupt(VCI). The VCI eases rescue procedure, controlling fires and saves a great deal of times. For example by one clearing action a valuable shortcut is created between two parts of the map or an important civilian is opened.

The police is considered to be unable to make communications in order to select proper task in each state but communication will improve the result of this selection.

3.2 Ambulance Team Strategy in 2012

The Problem that the ambulance Team is to deal with is that, there are a number of injured civilians in the city that some of them are also buried and the ambulance agents are to rescue and carry them to the refuges.

Any team that wants to implement a good Ambulance Team agent should meet some prerequisites such as a reliable Communication system and a good updated world model, an effective searching method, an optimized path finding method and finallya reliable death time estimator.

After examining several strategies implemented in previous years, we have done some assumptions as to be a Base of a Good Ambulance team.

- Each buried agent should be rescue first.
- Each civilian should be rescued by only one agent, excluded some critical situations.
- Rescuing less emergency civilians is a waste of time that is avoided if there are targets that are more important.

Based on these assumptions the total strategy of our ambulance team agent would be to:

1. Collect an updated list of injured agents and civilians that should be rescued.
2. Determine nearest reachable refuge for each injured civilian.
3. Remove the civilians that are impossible to be rescued from the mentioned list.
4. Assign priority tothe remaining civilians in the list.
5. Simulate rescuing operation and change the priorities if needed.
6. Rescue civilians based on simulating result.

Since radical changes are made in simulators we need to change are previous tools and for doing so, we are to collect samples and use neural networks instead of our previous tools.

Our strategy is centralized, as there is no different between each centers we define a virtual center named "ACA"(AmbulanceCenterActivity). If the communication system tells us that we have a Central Middleman we use it as ACA otherwise an agent will be act as ACA and if there are very low communication or no communication at all, every Ambulance Team agent can make a greedy proper decision based on priorities.

4 Search

This year we worked on two sections of our search strategy, search states and communication information spreading strategy.

4.1 Search states

Our search strategy has four main search states that are mentioned bellow:

1. **Block Search**

   Block search is designed to guarantee that agents have visited every building once in early cycles of the simulation. Map is clustered into n (number of block searcher agents) clusters using a semi-hierarchal clustering algorithm. Each cluster itself is divided into smaller blocks. Our proposed semi-hierarchal clustering algorithm is consisted of performing K-Means algorithm recursively. We insisted on using K-Means since it can be personalized to create clusters which have a high value of consistency among buildings according to their distance. This personalization is done by defining a heuristic function used as the distance function of K-Means.

   Block searcher agents are responsible for visiting blocks frequently to find new fire zones. Although this strategy seems to be extremely simple, its simplicity results in a rapid update of block status and affects search's throughput excessively.

2. **Fire Search**

   Fire search is designed to satisfy following requirements:

   - Updating fire zones shapes and size.
   - Finding new fire zones

   Fire search is done according to our fire estimator and other parameters such as distance to fire zones and etc. After finding a new fire zones, fire searchers try to update the shape of the fire zone by visiting random, probable and far building near the new born fire zone.

3. **Civilian Search**

   This state is designed to find new civilian and update old civilians' condition. Civilian searchers choose the most probable building to search for civilian. The probability of having civilian for buildings is

computed according to parameters such type, size and location of the building. We also considered probable use of the building which is reached by considering density of buildings in its vicinity and appearance feature such as material, size and number of its floors.

4. **Combined search**

Combined search is called after agents do not have any fire or civilian search task. Agents need to search if they don't have any other task. This means that you shouldn't stop. In previous states, agents' search task is chosen to fulfill a specific goal. This is beneficial because it lets agents to have a more concentrated algorithm, optimized for the task. However, we needed a strategy to keep the agent searching till the end of the simulation. A more dynamic and flexible search strategy that switches easily between previous search tasks. Therefore, combined search is designed to satisfy following requirements:

- Agents needed a more flexible search strategy, dynamically choosing between two type of search task
- The search phase of agent should never stop
- Agents needed a strategy to re-visit all task and update buildings

Combined search is done by evaluating buildings according to numerous features and choosing the most valuables. These features are consisted of:

- Distance to agent
- Mean distance to other agents
- Buildings usage type
- Last update time
- Last searched time (Searched for civilian)
- Size of the building
- Location of the building
- Distance to agent's cluster
- Fieriness of the Building
- Min distance to fire zones
- Etc.

4.2 Communicationless information spreading

This year we designed a strategy to increase the probability that agents exchange their information. This is done by choosing a gathering point is pre-compute. The gathering point is a road chosen by 3 parameters:

- Distance to center of map's building density, which, is the location with highest density and is computed similar to center of gravity of and object
- Reachability to highways
- Gravity based score function similar to the one explained section 3.1.1 and is computed bellow:

$$Gravity\ of\ road(i) = \sum \frac{value_j}{distance_{ij}^2}$$

In the formula above, $distance_{ij}^2$ is the distance between important entity j and candidate road i. Important entities consists of refuges and agents. Value j is a static value which depends on the type of entity which can be refuge, ambulance team, fire brigade and police force. The gathering point is used to alter agents move cost function in a way that pushes agents to choose paths that are near the gathering point, therefore, increasing the probability that agents successfully exchange their information with each other. We also, designed a saying turn pattern that guarantees a high probability that agents receive messages communicated in their vicinity.

5 Agent Coordination

Depending on the strategy each agent decides in a specific situation, the decision will specify whether to work centralized or distributed, however center agents think that their platoon agents are working centralized so they provide centralized information needed by platoon agents.

As the platoon have almost the same world model their decision about this matter will be coordinated sufficiently.

6 Messaging System

As described in previous TDPs, the possible scenarios may include:

1. One or two low-bandwidth, high reliability channels and several high-bandwidth, low reliability channels.
2. A large number (10 - 20) of low-bandwidth channels.
3. One high-bandwidth, high reliability channel and a number of high-bandwidth, low reliability channels.
4. Only one channel with moderate bandwidth.
5. Only one low-bandwidth channels.
6. No radio channels at all.

Thus different strategies should be used for too specific scenarios, such as one low bandwidth channel, low reliability channels and no radio channels. Many of these strategies were previously explained, thus, we only described the following section:

6.1 Strategy for one low bandwidth channel

For low bandwidth channel, A new kind of message blocks are used and only one message block is sent per package. This package's size is only 16 bits. At first the fire message will be sent, and then only civilians' positions are reported. The message is sent as soon as the agent considers it important regarding its priority.

6.2 Strategy for low reliability channels

We believe that, receiving important messages in the low reliability channels is more efficient than getting too many messages with normal and important priority messages and since the probability of having noise in the second packet it at most $p_f * p_f$, we decided to duplicate important messages. However, duplicating messages results in wasting bandwidth. Therefore, the messages are sent normally and the important messages are only duplicated using the aforementioned method in section 6.1.

7 Software Architecture

S.O.S agents are based on SOS 2009 that described in the 3 year ago. In year 2010 we changed it duo to kernel changes and try to integrated it into a newer version. we try to find base problems and solve them. now we developed very good tools like reachablity, message system, sensible area, fire sensible area and etc. these are tools for developing rescue agent strategies. the agent strategies structure is define as follow:

7.1 Agent design and code structure

We designed a multi layered structure because we believe it is easier to optimize and debug. We also could divide the decision making process of the agent to different – higher and lower- levels. Therefore as you can see in figure we designed four layers.

7.1.1 High level decisions

This level chooses which state should be taken care of at the moment. It check the state that the map is at. It changes the priority of tasks considering the situation of the environment such as blackness and size of fire zones. This is the part that we have been trying to train it will and lets are agents to be flexible in different maps and scenarios.

7.1.2 States decision

For each situation we should have a plan. States are the activity of a situation that we have planned. This makes high level decisions to decide easily and make it easier to handle the situation without considering to other situations.

7.1.3 Low level

In this level we use methods implemented in S.O.S basic agent and low level acts such clearing a blockade. We decide in order to clear a road which blockade should be cleared first and how should it be cleared.

8. References

1. A General Computational Recognition Primed Decision Model with Multi-Agent Rescue Simulation Benchmark by Alireza Nowroozi; Mohammad Ebrahim Shiri, Assistant Professor; Angeh Aslanian; Caro Lucas, Full Professor
2. Modaresi, M et. al: S.O.S. Team Description Paper Proceeding of Robocup 2012.
3. Ansari, M. et. al: S.O.S. Team Description Paper Proceeding of Robocup2006.
4. Azizpour, H. et. al: S.O.S. Team Description Paper Proceeding ofRobocup 2007.
5. Ghaffuri, M et. al: S.O.S. Team Description Paper Proceeding of Robocup 2008.
6. Hashemi, B et. al: S.O.S. Team Description Paper Proceeding of Robocup 2009.
7. Cormen, T., Leiserson, C., Rivest.: Introduction to Algorithms MIT Press, Cam bridge(2000)
8. Horstmann, C: Object-Oriented Design and Patterns
9. R.C. DubesandA.K.Jain. Algorithms for Clustering Data. Prentice Hall, 1988.
10. Markari, A et. al: S.O.S. Team Description Paper Proceeding of Robocup 2010.
11. Robotic Rescue Simulation league Rules.