AUTRescue Team Strategy Description

Mohammad Ebrahim Shiri, Adrin Jalali, Mohammadreza Jooyandeh, Nazanin Roshandel Tavana, and Mahsa Behzadi

Math and Computer Science Department,
Amir Kabir University of Technology(Polytechnic), Tehran, Iran
{adrin_jalali,mr_jooyandeh}@aut.ac.ir
http://www.AUTRescue.tk

Abstract. In this paper we present the description of the strategy and algorithms used by AUTRescue team participating in RoboCup Rescue contest 2003. We will present the low level changes that we applied to the base code ADK(Agent Development Kit). Then some algorithmic strategies as the neural networks, Q-Learning and path planning by D^* are briefly described.

1 Introduction

Beside our interest in the programming and algorithmic aspects of the robocup rescue contest, we are specially motivated to try to improve the algorithms for this task with hope that these algorithms can be useful in the similar situations in the real world. With this aim, our work has some main parts; We changed the structure used by the kernel by adding a header, center had heard bit and constructor ID to the message protocol. The used functions are more efficient with this new structure. Note that what we mean by a change in the message protocol, is in fact the change in usage of the body of message(logical change) and not change in the physical description. The map of objects (red-black tree) is changed to another tree structure which helps us to realize faster searches for the search of an ID of a special type. section 2 is dedicated to description of these changes. In the section 3 the agent tasks in our strategy is presented. And finally in section 4 path finding algorithm of our strategies is discussed.

2 Base Code

The base code is ADK, but we have changed some parts of that in order to increase the efficiency of our algorithms. The main changes consist of two parts; In one hand we added a message object to the ADK and in the other hand we have changed the structure of the memory.

2.1 Communication Protocol

Our message protocol consists of a *header*, *center had heard bit*, *constructor ID* and the *main body* of the message. The header indicates the type of message such

as road information or clean road. The center had heard bit is set to 1 if there is a center which had heard this message otherwise it will be 0. The constructor ID is the ID of the main(initial) sender of the message. The main body of the message is the information which is sent, such as ID of the road, which should be cleaned by a police force. The constructor ID and the center had heard bit are necessary in the newest version of the kernel because when a fire brigade tells something, a police force is not able to hear the message of a fire brigade. Let us explain this by an example: suppose a message should be sent by a police force to a fire brigade; Firstly this message should be sent to the police office, then to the fire station and finally to the fire brigade Police office hears the message sent by the police force and tells it, so fire station hears it. similarly when fire station tells the message, fire brigade hears it; but here there is a conflict because the police office will hear the message for the second time. The police office need to understand that it should not send the received message again. The Center Had Heard bit helps the police office to solve this problem. If this center had heard bit is true so the center will not send the message for the second time.

2.2 Memory Structure

The data structure used in ADK is mainly a link list and a map (realized with red black tree) to the objects. This map contains all the Objects sorted by their ID.

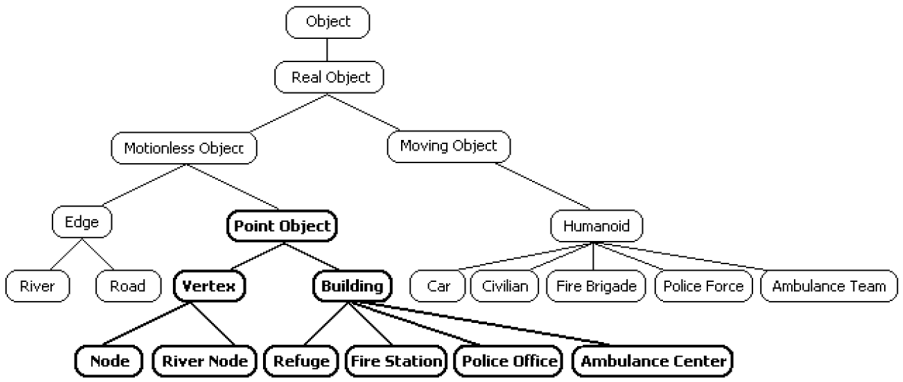

Fig. 1. The new structure for the objects.

We have changed this structure into a tree shown in figure 1.

In this new structure, if the type of object is known, searching for an ID will be faster; suppose that we want to search for a point object with a known ID. Firstly we traverse the tree till reaching the point object. Then, we call the search function for all the children of the point object (These children themselves

can be implemented by a tree structure). In the other words the search function called for the children of the point object node in the tree is not the same as the functions which are called in its child. By this way we will improve the efficiency of our searches. Remember that every leaf in this tree has a map of IDs of its own type. Another usage is when we want to retrieve all the vertices of a given type. In this case using this memory data structure enables us to avoid testing(exploring) the vertices of other types. As the trace of all the objects is needed using a linked list would also help us to have better efficiency.

3 Agents

3.1 Police Force and Police Office

In our strategy some police forces have to clean the *main road*s of the city. For this purpose, the police office detects the main roads and tells the ID of the *main road cleaner polices* to the all of the police forces. By definition the main roads are the set of adjacent roads, which have almost the same priority which is higher than all other roads. *Priority* is found by a function which has many parameters. The position of road in the city, number of shortest paths of the city graph which go through that road, distance of the road from fire centers and other topological properties of this edge in the graph of the city roads are the parameters which are considered for computation of the priority of a road. Each parameter has a coefficient and these coefficients should be obtained during the solution. For the aim we use neural nets and Q-Learning to learn these coefficients. When the first set of main roads is cleaned completely the police office will finds another main road set. Police forces who should clean the main roads are also found and chosen by the police office. A combination of a shortest path algorithm and covering algorithms of a graph is used to find the nearest polices to the main roads. Then police office sends the main road set to main roads cleaner polices so they move in direction of the main road through the ways, which are updated every cycle. In the Current implemented version number of polices, who clean the main road, is 2. Another important task for polices is to clean the blocked roads by which other agents want to pass in their path. so agents tell their ways so that polices hear that and some polices start to clean the roads needed to be cleaned. Whenever a road is cleaned, the police force who cleaned that road, tells to other agents that this road is cleaned. Also whenever a blocked road is found by other agents, they tell this information so that police forces will be informed about that and that road will be checked and cleaned sooner than other roads.

3.2 Fire Brigade and Fire Station

At the beginning of simulation, some buildings begin to burn and when time elapses, neighbors of firing buildings also begin to burn . so some regions of the disaster space can be supposed to be centers of fire. The agents set some priorities for these centers according to some parameters such as the elapsed

time from beginning of burning, number of civilians in that center, number of burning buildings, material of buildings and so on. some agents (number of them depends on the above parameters) start to go to that center to extinguish. The Order of building to be extinguished in the same center is very important. By using learning algorithms the buildings in a center will be sorted. The permutation is obtained by using many different parameters. Finally as due to current parameters, kernel does not alert all fires to all fire brigades, every agent who understands the existence of a fire, sends this information for all the agents can hear that.

3.3 Ambulance Teams and Ambulance Center

The task of ambulance teams is to rescue agents. Injured agents ask for help by saying a help message. Every agent, who hears the help message from a civilian, tells its position to others, so that the ambulance teams can estimate the place of injured civilians. When ambulance teams are informed of the place of injured civilians, they start to rescue civilians. Each civilian has a priority level for each ambulance team. so ambulances start to go to rescue the civilians having more priority. After rescuing the civilian, one ambulance team gets him/her to the refuge. Civilians who are near fire, and have more hit points, have higher priority. The distance of an ambulance from a civilian is the important parameter for the priority of that civilian for that ambulance.

4 Path Finding

In our strategy, agents use the D^* path planning algorithm based on the Focused D^* paper by A. Stentz (CMU Robotics Institute). The important point, here is the used heuristic functions: Each usage of the algorithm has its own heuristic functions. For instance when an ambulance team wants to find a path in order to reach an injured civilian, main roads of the city are safer and faster to pass, because when they start to go to pass the main road, this road is either being cleaned by police forces (main road cleaner polices) or already completely cleared. The idea of Q-learning is a good and necessary technique to use for this purpose.

References

1. Stentz A.: The Focused D^* Algorithm for Real-Time Replanning. Proc. UCAI, August 1995.
2. Russel S. and Norvnig P.: Artificial Intelligence: A Modern Approach. Prentice Hall Series in Artificial Intelligence.
3. Bowling M.: RoboCup Rescue: Agent Development Kit. CS department of CMU., Pitsburg. December 2000.
4. Menhaj M.B.:Computational Intelligence. Vol 1. AUT press.