RoboCup 2013 – Rescue Simulation League Team Description
<Apollo-Rescue (P.R. China)>

<Jie Shen, Zhiwei Liang, Zhiqian Liu, Siyu Zhang >

Nanjing University of Posts and Telecommunications, China
Apollo Robotics Innovation Lab, NUPT
jasonllinux@gmail.com
http://www.apollo-rescue.com

Abstract. This paper describes the main features of the Apollo-Rescue rescue simulation team that is going to participate in the RoboCup 2013. In the past year, a lot of work has been done in our code. In this paper, a new decision-making module will be introduced and a modified path planning algorithm for complex and dynamic environment will be covered. Moreover we have realized an effective fire control algorithm aimed at the dynamic fire spreading situation. Apollo-Rescue has gained 5th in ChinaOpen2012 and 8th in Iran-Open 2013.

1. Introduction

Apollo-Rescue, as a newcomer in RoboCup Rescue Simulation. We began to participate in RoboCup China Open Competition since 2010. After last competition in Iran-Open 2013, we made a conclusion and tried more to improve the performance and we have been working on the weaknesses of our code. Apollo-Rescue faces some major problems during 2012 competitions. In the first, we improve the decision-making procedure of agents. A high performance unified auction frame is used to solve task allocation problem for Ambulance Team. In Apollo-Rescue 2012, Fire Brigades cannot control the fire spreading effectively, so a new cluster algorithm for fire zone and convex hull method are introduced. Moreover, a more flexible path planning algorithm based on A-Star is used in our code. By means of these efforts, in 2013 competitions, we expect to become a high quality rescue team in the near future. More details will be introduced below.

2. Review of the RoboCup Rescue Simulation Platform

RoboCup Rescue Simulation System (RCRSS)[1] is a typical Large Multi-Agent System that is aim to manage the disaster when an earthquake happens.

In uncertain and dynamic environment, the rescue agents we developed should have the ability to known the whole map by partial observation and unreliable communication, to select behavior in quite large state space and to find optimal path in complex road condition, etc. We take RoboCup Rescue Simulation as a problem of multi-agent systems, and our long-term goal is to do research in decision-making in emergency environment, path planning and other challenging projects in Artificial Intelligence [2].

3. Agents

3.1 Decision-Making Module

An effective decision-making procedure is assurance for agents to complete all kinds of tasks. As ***Fig1*** shows, a new decision module is used in Apollo-Rescue.

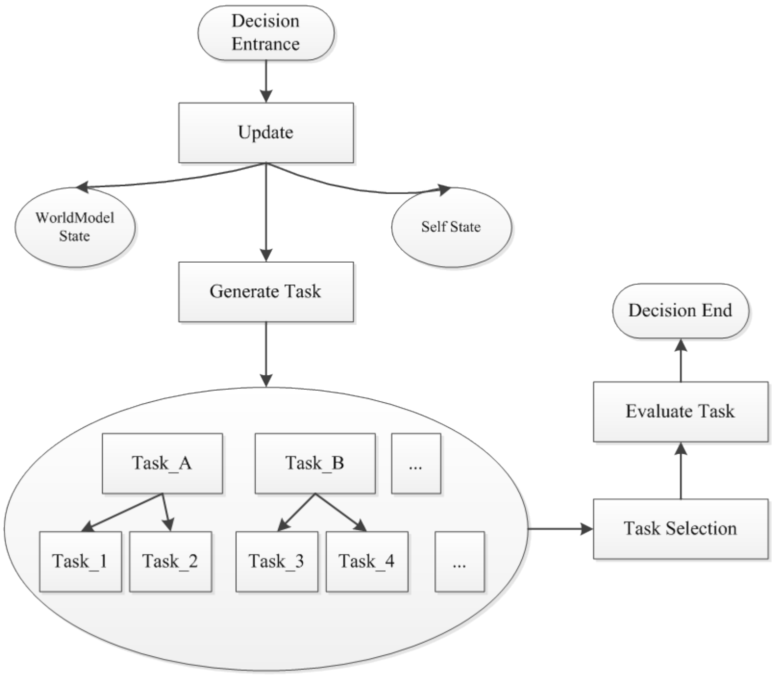

Fig.1. Decision-Making Procedure

First, agent should update state before decision. It contains two sub-procedures, i.e. update self-state and update world model. So agent can know exact information about the whole disaster city and itself.

Then, agent begins to generate task. As for task, we have almost rebuilt it. Now task consist of three parts: Task Identification, Precondition and Effect.

Task (Task-Type, state, attribute),
Precondition:
Effect:

Identification indicates the task type and some attributes of task. **Precondition** is pre-condition for the corresponding task being done. **Effect** describes how the state changed after the task is done. And it is also used to evaluate the task being taken.

In this module, we divide tasks into two classifications, i.e. coarse-grained task and fine-grained task. Fine-grained task consists of several coarse-grained tasks. The accomplishment of a coarse-grained task should be the result of a series of fine-grained tasks.

3.2 AT

Auction, as a principal means of negotiation for a multi-agent system, can be applied to the completion of task allocation for one-to-many and many-to-many [3].

For Ambulance Team, we developed a unified auction framework to solve the problem of task allocation. Auction is a fast and efficient algorithm for resources allocation. There are two roles in auction model, i.e. auctioneer and bidder. In our framework, we use distribution decision support system. Multi agents are responsible for the multi-tasks' auction job.

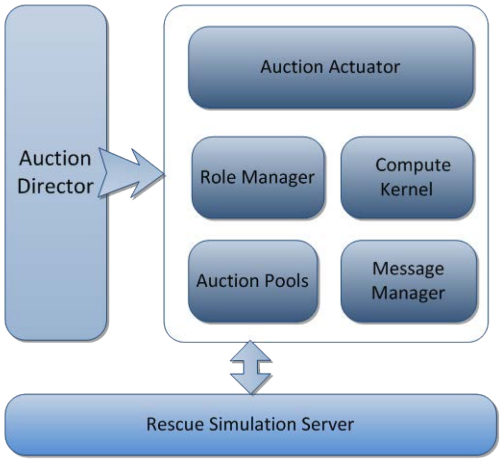

Fig.2. Unified Auction Framework

As ***Fig2*** shows, Auction Director is supervisor, and it dispatches other modules in this framework. Auction Pools stores and manages the bid that agent received and send. Role Manager is responsible for keeping agent's role in auction model, auctioneer or bidder. Compute Kernel is the important part for the whole module, agent compute the bid price by self-state and World Model state through compute kernel.

As ***Fig3*** and ***Fig4*** shows above, when agent explored a buried civilian or agent, it becomes auction agent automatically. Then auction agent evaluates the task and send message about the task. If other ambulance agents heard auction message, they become bid agents automatically. They compute bid price through compute kernel and send bid to auction agent. We set a fix time $T_{\max}$ for auction agent to wait for bid agents to send their bids. Once time out, auction agent collect all bids received and select the bid which can maximize the utility for whole system. Then auction agent will inform the bid success agents and corresponding agents will execute the allocated task.

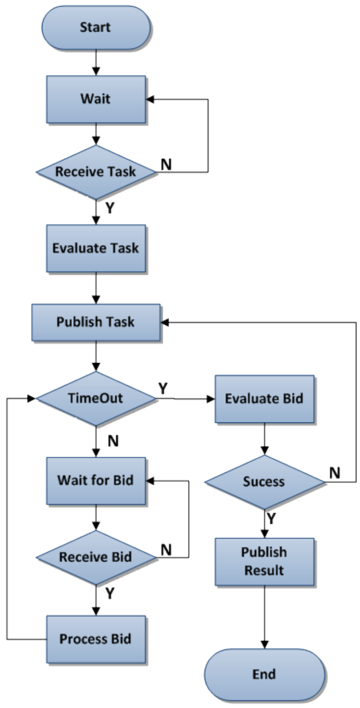

Fig.3. Procedure of Auction Agent

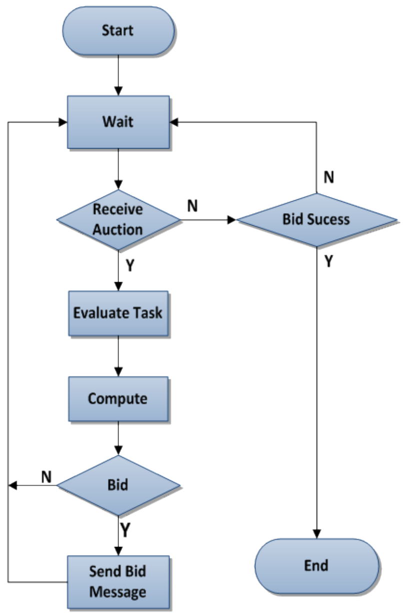

Fig.4. Procedure of Bid Agent

3.3 FB

The Fire Brigade plays a quite important role in extinguishing fire and preventing fire from spreading fast. This year we introduced the Cluster Algorithm[4] which works based on the notion of distance between two buildings as the density reachability so that buildings would be classified into different zones according to their locations in the simulative city to approach the real building grouping condition by geometry. In this way, we could implement an advanced thinking strategy on pre-computing of acting in FB agents' each decision cycle. For that the fire brigade would decide itself whether the fire should be put out or be controlled when it has a great influence on buildings in neighbor clusters, or be ignored when the fire in that zone is out of control.

If buildings are on fire, the fire spreading priority of each burning building in clustered zones will be evaluated as following:

$$p_i = \sum_{k=1}^{N}[C - Dis(k)] \cdot (M - fieriness) \tag{1}$$

$$p_j = \sum_{i=1}^{n} p_i \tag{2}$$

Where:

p_i: Priority of burning building i;

p_j: Priority of cluster zone j;

C: a constant equals the length of the longest diagonal of the map;
N: count of neighbor clusters of the cluster zone building located in;
Dis(n): distance between the building and the neighbor cluster $k, n \le 6$;

M: the max value of building fieriness, value of $M = 8$;

Then the cluster zone *j* which has the highest value of *P* will be chose as the target extinguishing zone.

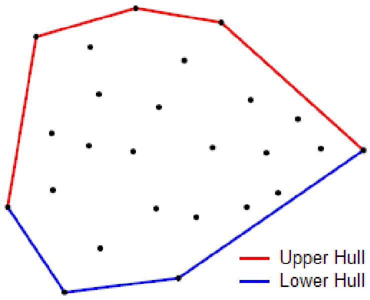

Fig.5. Convex Hull

In one certain Rescue-Simulation competition, fires always happen unexpectedly and randomly based on the sever, so there is always a situation that the amount of available fire brigades for extinguishing all the fire zones is far from adequate to meet burgeoning demand. So two methods here are required: one is used to locate the fire task zone above and another is introduced as following which focuses on specific building that is on the edge of a fire region, which could, effectively, meet progressive and dynamic needs to reach a better optimization of fire brigades' task distribution.

Therefore, another algorithm method based on the related theory of *Convex Hull* of Computational Geometry is being constructed to handle the fire-control problem. And the convex hull of set of S is the smallest convex region that contains all points in S, imagine they are nails on a board, if we stretch a rubber band around S and release it, it takes the shape of the convex hull as ***Fig 5***.

We could treat burning buildings as some points on the coordinate based on the map, the method is heading to form a finite convex set of the burning building group, then the boundary of these buildings will be exactly defined. So the buildings on the edge of an on-fire convex region can be located that Fire Brigades could move forwards to extinguish the building inferno which is prioritized on fieriness, temperature, total area and distance factor. The burning building convex hull must be updated after each time-step action and this method provides fire brigades a direct and bright way that could be more efficient to keep fire from spreading too fast and minimize the fire damage.

Here we choose Graham's Scan Algorithm[5] to compute the convex hull set, I solves this problem by maintaining a stack *Q* of candidate points. Each point of the set *S* is pushed once onto the stack, it is determined whether moving from the two previously considered points to this point is a *left turn* or a *right turn*， and the points that are not vertices of convex hull are eventually popped from the stack. When the algorithm terminates, stack Q contains exactly the vertices of the convex hull, in counter-clockwise order of their appearance on the boundary. ***Fig 6*** tells the processing details.

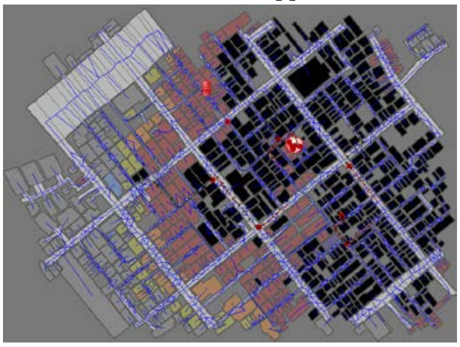

Fig.6-(a) Apollo-Rescue2012

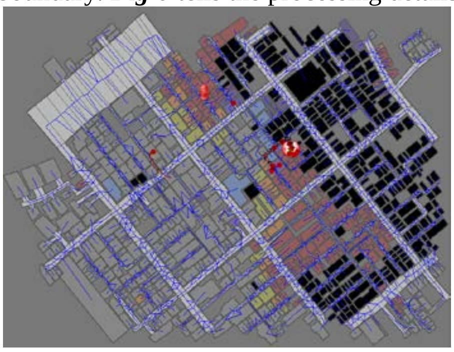

Fig.6-(b) Apollo-Rescue

The screen-cut pictures above show the final results. ***Fig 6-(a)*** shows what extinguish effect was like before the methods improvement and the promotion of fire spreading control can be founded directly in ***Fig 6-(b)*** which was captured after the application of our new Fire-Brigade methods. We chose map of Kobe here for testing, and marked effects were achieved. The burned area in the second picture is 30% less than that in the first one.

3.4 Path Planning

For solving the problem of path searching, we adopt the optimized A* algorithm. A * (A-Star) algorithm is one of the most effective methods for solving a static network shortest path. The heuristic searching makes use of the problem with heuristic information to guide searching [6], so as to reduce the range of searching and the complexity of the objective. It avoids blind searching such as breadth-first and depth-first algorithm. Considering the information of obstacles into heuristic function, we can make the choice of the path whose pass-rate is bigger and the length is shorter simultaneously.

We rebuild all roads and buildings into nodes and take the pass-rate into consideration. About the path selection ,we define a more effective valuation function : $\mathrm{f}(n)=g(n)+h(n)+p(n)$ (where $f(n)$ is valuation function which is from the initial node by node n to the target node, g (n) is the actual price which is from the initial node to node n in the state space, $h(n)$ is appraised price of the best path which is from node n to the target node, $p(n)$ is the pass-rate of current node n). When agent sees some nodes with obstacles or it is blocked by nodes with obstacles on its way to the target node, agent will immediately update the pass-rate of these nodes in its world model. At the same time, consider the pass-rate into the next step path selection

As the ***Fig 7-(a)*** shows, we get the four vertices of the current node: A, B, C, D, connecting the four vertices into quadrilateral by the blue line and calculating the width of road: $roadwidth=d_{AD}$ ($roadwidth$ is the distance of straight line from point A to point D). Get the Blockades on the node: Blockade-A, Blockade-B. Get all the sequence of vertices of Blockade-A. By calculating the distance between each vertex in the sequence and the straight line AB, we can obtain the shortest value is $d_1 = 0$, and the widest vertical distance from the obstacle to the roadside is $d_2 = p_1q_1$. Therefore, we can believe that the width of Blockade-A is $w_1 = d_2 - d_1$. Similarly, we can obtain the width of Blockade-B: $w_2 = p_2q_2$ .Thus, we can come to calculate the pass-rate of Road. Because the width of agent is given. Hence:

$$passRate = \begin{cases} 0 & if \quad w_1 + w_2 + 2 \bullet agentWidth >= roadwidth \\ 1 & if \quad w_1 = w_2 = 0 \\ 1 - \dfrac{(w_1 + w_2)}{roadwidth + \alpha} & otherwise \end{cases} \tag{3}$$

Where α is the error coefficient.

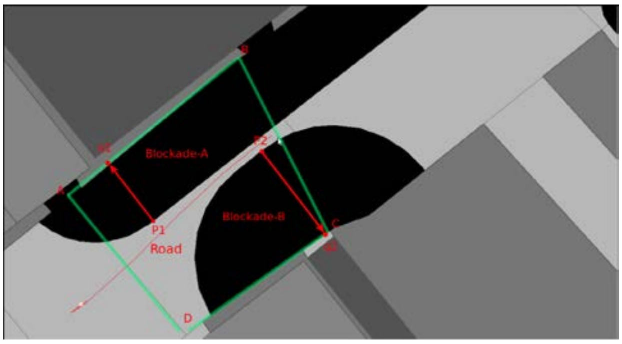

Fig.7-(a) Calculation pass-rate, the first case

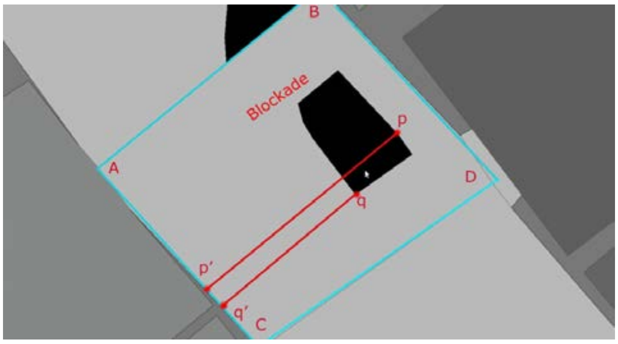

Fig.7-(b) Calculation pass-rate, the second case

Just like ***Fig 7-(b)*** shows, in this kind of circumstance, at first we find out the side which the blockade is close to.(Edge BD).At second step we calculate the maximum and minimum distance from all the vertices of the Blockade to Edge AC which is on the opposite of the Edge BD, we can obtain the shortest value of distance is $d_1 = qq'$, and the longest is $d_2 = pp'$. Hence

$$Blockadewidth = d_2 - d_1 \tag{4}$$

So we can calculate the percent of passing rate:

$$passRate = 1 - Blockadewidth / nodewidth + \alpha \tag{5}$$

Where α is the error coefficient.

4. Acknowledgements

At last, thank all of the server developers for maintaining the rescue simulation server and their kindly technical supports. Besides, we show our gratitude to ZJU-Base, SEU-Jolly, CSU_Yunlu for their kindness help. We are having our great thanks to our teacher Z.W. Liang and other members of Apollo Robotics Innovation Lab who provide us a research environment and inspire our work.

5. References

[1] RoboCup-Rescue Simulation Manual, RoboCup Rescue Technical Committee (2000).

[2] Jie Shen, Linlin Wang. Apollo-Rescue 2012 IranOpen Team Description.

[3] Craig Boutilier, Moise Goldszmidt, Bikash Sabata. Sequential Auctions for the Allocation of Resources with Complementarities. Proceeding IJCAI'99 Proceedings of the 16th international joint conference on Artifical intelligence, Vol.1, 527-534, 2011.

[4] Hua Jiang,Jing Li,Shenghe Yi et al. A new hybrid method based on partitioning-based DBSCAN and ant clustering [J]. Expert Systems with Application, 2011,38(8): 9373-9381.

[5] Thomas H.Cormen, Charles E.Leiserson, Ronald L.Rivest, Clifford Stein. Introduction to Algorithms[M],Third Edition(ISBN 978- 0- 262- 03384- 8). Massachusetts London: The Massachusetts Institute of Technology,2013.1.827~831.

[6] Taixiong Zheng, Xiangyang Zhao, Research on Optimized Multiple Robots Path Planning and Task Allocation Approach, Robotics and Biomimetics, 2006. ROBIO '06. IEEE International Conference, vol., no., 1408-1413, 17-20 Dec. 2006