Poseidon Team Description Paper
RoboCup 2016, Leipzig, Germany

Yasamin Alipour[1], Melina Farshbaf Nadi[1], Kiana Jahedi[2], Kimia Javadi[1], Pooria Kaviani[3], Seyedeh Behin Mousavi Madani[1], Rozhina Pourmoghaddam[4], Yekta Sarkamari[1], Fatemeh Zeinabadi [1]

[1] Farzanegan High School, Robotics Research Group, No. 56, Shahid Sarparast St., Taleghani Ave., Tehran, Iran
[2] Amirkabir University of Technology, No. 424, Hafez Ave.,Tehran, Iran
[3] Sharif University of Technology, Department of Computer Science, Azadi Ave., Tehran, Iran
[4] Tehran University, Enghelab Ave., Tehran, Iran

{yasamin.alipour, melinafn15, kiana.j9, kima.javadi, pooria.kaviani, behin.mousavi, rozhina.pourmoghaddam, ysarkamari, 2000negar1215}@gmail.com

Abstract. This manuscript describes the algorithms and contribution of Poseidon 2016 to acquire the optimum and most efficient solutions for rescue agent simulation problems. New ideas in World-Modeling structure and the agents' decisions were implemented. Each section explains new algorithms of this year's implementation in detail. This version of the Poseidon team is based on the sample code of the server with extensive changes in the structure of the agents and World-Model. In Poseidon 2016, computational geometry and artificial intelligence algorithms have been used for solving the problems of rescue agent simulation.[1]

Introduction

The purpose of the Rescue Simulation League is to decrease casualties and financial loss caused by natural disasters such as earthquakes, floods, etc. In order to achieve this goal, a large urban disaster is simulated which indicates each agent's actions in this situation. This simulation matches real world limits and problems as accurately as possible. Rescue simulation agents include ambulance team agents, police force agents and fire brigade agents. The principal task of the police force is to open the closed roads. The main duty of the ambulance team is to save lives. Extinguishing the fire is the main task of the fire brigade agents. In addition, all the agents are responsible for facilitating the other agents' tasks. The Poseidon team is trying to find solutions and improve available algorithms to solve the agents' problems.[1]

2. World – Modeling and Communications

2.1 Rapidly Exploring Random Tree

Agents get stuck in blockades. They remain stuck until a police force agent rescues them. This will slow down the team's overall process. In order to limit the waste of time, every time each agent gets stuck, it uses rapidly exploring random tree (RRT) algorithm to get unstuck. "Rapidly exploring random tree (RRT)" algorithm is used to efficiently search non-convex, high-dimensional spaces by randomly building a space-filling tree.[2] If an agent gets stuck, it starts to create random coordinates in the area in which it's stuck. [Fig 1]

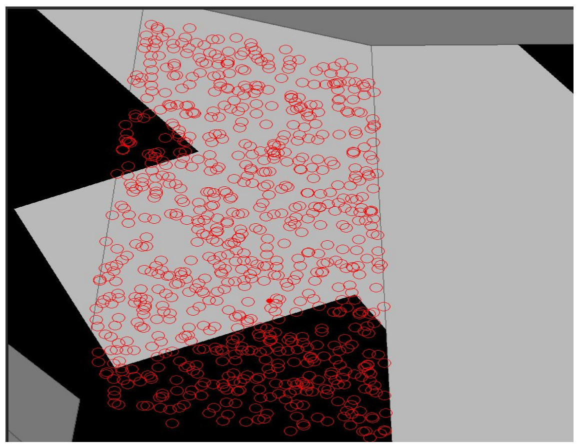

Fig 1. Random coordinates in the area of the stuck agent

Afterwards it randomly creates paths with the given coordinates. The path that doesn't have intersections with the blockades is chosen. As the paths are created randomly and aren't optimized, we will remove the coordinates that aren't essential for getting out of the blockades and have short distance from each other. [Fig 2]

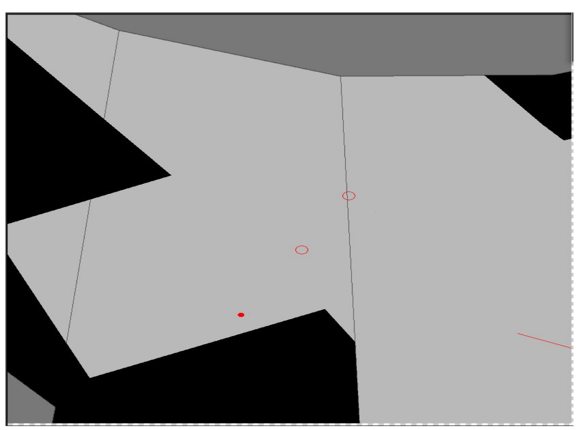

Fig 2. Selected coordinates

2.2 Send Civilians Information

Speaking in channels and sharing information is very important for synchronizing all of the agents. Therefore, we have to reduce saying unimportant information in the radar, so that we can share more essential data with other agents. One of the things that we say on the radar is civilians' IDs, positions, ***death time***s and buriednesses. So that everyone can become aware of their condition. However, the IDs are large numbers and take a lot of space of the bandwidth (up to 32 bytes) which is too massive in limited maps.

Creating fake IDs for each of the civilians decreases the length of what is sent through radar up to 5 bytes. What we need to hear to create fake IDs are civilian's position and its recognizer. A civilian's recognizer is the remainder of that civilian's ID which is divided by a specific number. This specific number differs in different maps depending on the bandwidth of the channels (The maximum is 32, and the minimum is 8). By hearing this information we can generate fake IDs for civilians and use them as real ones.

Here is an example of generating a fake ID:

2 1 0 0 - 5 6 4 3 - 0 8

- The first part is a code that shows this ID is fake. It depends on the map and the number of the included areas.
- The second part is the civilian's position. We use this feature in generating the fake ID because there might be the same civilian recognizers in the whole map and we don't want to possess the same IDs for different civilians.
- At last we must have a fake ID with same civilian recognizer as the real ID because the ambulance center's task system depends on it and we don't want to deviate the task managing system, so we add up the fake ID with a number that the final civilian's fake ID recognizer equals the civilian's real ID recognizer.

Ultimately, we add this heard civilian with its fake ID to the model and save more space for other sayings.

3. Ambulance Team Agent

The ambulance agents directly follow the main goal, which would be saving the civilians' lives. Ambulance agents find civilians, remove their buriedness, and carry them into refuges.

To follow this procedure, we need a coherent strategy that contains:

1. A systematic search to find civilians
2. A prioritized filter for recognizing more injured civilians by exact estimation of the time that civilians die

3.1 Death Time

3.1.1 Accuracy

The cycle that a civilian dies in is called ***death time***. This year, we have devised a new method for estimating ***death time*** more precisely by influencing more exact factors that define ***death time***, such as Health Point (HP) and damage. With an accurate ***death time*** estimation we will have an optimized schedule that provides better choices for saving civilians.

The problem is that we don't receive the HP accurately, so we will have an accuracy with determined span. At the time that the data (HP) changes, we know that the real data is about the border of the span. As a result, if we get a mean between last data (HP) and new data (HP), we would have more exact estimation about HP.

3.1.2 Death Time Estimating For Civilians Near Fire

The other new strategy that we follow is to avoid rescuing civilians that are near the fire by considering this factor in death time. The point is that rescuing a civilian is time-consuming, due to the limitations in removing buriedness, during this process, near areas may catch fire as well. Ambulance agents can't stay in a building which is on fire because the civilian will die soon after ignition. Moreover, the ambulances themselves will be damaged, so spending time to reach and rescue an unsuitable civilian that we predict will die before rescuing ends, is just wasting time. In the end, if the fire is put out, ambulance agents can rescue the civilian again.

For this purpose, we assume some layers in maps with specific radiuses that depend on the types of the maps. The time that it takes for fire to spread out from one side of the layer to another side, is a definite time. We calculate the distance between the target building and the nearest building which is on fire. By dividing the distance by radius of the layers we can understand how far the fire is from target and when fire will reach. This ***death time*** is compared with another ***death time*** that is calculated by damage, buriedness, etc, without influencing the effect of fire. The one which happens sooner, will be chosen as an ultimate ***death time***. [Fig 3]

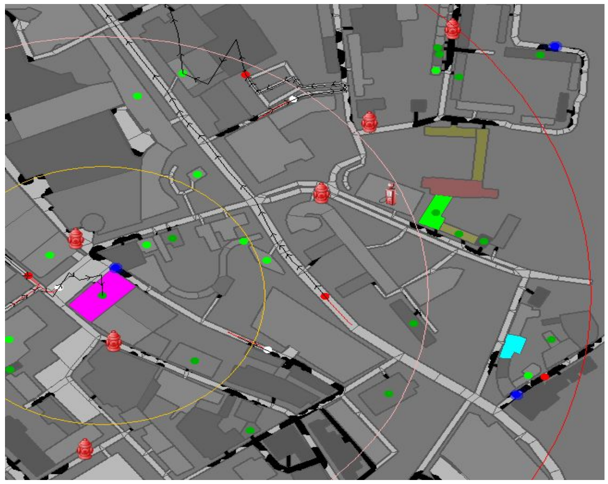

Fig 3. Layers between civilian's building (pink) and nearest building that has fire (green)

4. Police Force Agent

The main duty of police force agents is clearing the roads and opening the way for other agents. Therefore, it has significant effects on the other agents' operations.

4.1 Clear

There are two parameters in clearing that we should focus on. First, it must be performed as fast as possible in order to make the buildings in the map accessible for fire brigades, ambulance agents and civilians to reach their target. Second, clearing should be done smoothly. One of the problems of our previous clearing method was that police force agents cleared the roads, but left some blockades with sharp edges in the road. [Fig 4] The road was not filled with blockades, but because of the server's bug, the agents had trouble passing the road. To prevent this problem, police force agents try to clear just on the ***Clear Lines***.

The Algorithm of Clearing is described below:

1. Connecting the midpoints of agent's position's entrances to the entrance of agent's target (The neighbor of agent's Area which it's path contains it)
2. Selecting the nearest line to the center of agent's position's area as the ClearLine
3. Moving to the nearest point on ClearLine if the distance gets more than secureDist (The distance which is close enough to the ClearLine)
4. Defining the clear rectangle on the ClearLine
5. Detecting the blockades in agent's clear rectangle
6. Calculating the cost of clearing the blockades

7. Sending Clear if it costs more than secureCost (The cost of the maximum blockades that can be cleared each cycle)

In this method there will be less sharp edges that cause problems for other agents. [Fig 5]

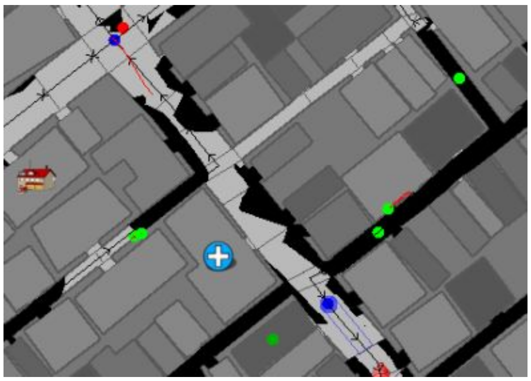

Fig 4. Previous clear

Fig 5. New clear

4.2 Task-Specific Agents

Since clearing around a fire zone and making the fire accessible for fire brigades is a critical task for police forces, one third of all the police forces are allocated to ***fire tasking*** in their defined zones. They don't receive tasks from center and their only duty is finding and clearing fire zones in their zones. If their zone doesn't have any known ignited buildings, they will search in it until they find one. In conclusion, the risk of not being informed of the fire is getting less and the act of ***fire tasking*** is done faster.

4.3 Clearing around the fire

For bringing fire under control, fire brigade agents need to have access to buildings which are on fire. so we have to open their way as fast as possible. In new method we only clear the way to the buildings which have three features. First, being among ***around buildings***[3] which are on fire. Second, having high probability of ignition. third, having high spread of fire.

We identify them by calculate average of how far fire would spread by buildings, then we choose buildings which they can spread fire equal or more than average. These are buildings with high priority and buildings around them are those which have high probability of ignition.

When these buildings are identified, last step is clearing the roads around them. With this method we save more time and fire brigade agents will have more control over the fire and can extinguish more ignited buildings.

5. Fire Brigade Agent

Fire Brigade agents' main duty is to control the fire in the critical situation to decrease the Losses of the fire.

5.2 Extinguishing unreachable buildings

If a building around the fire zone catches fire we must attempt to extinguish it. However, when a fire brigade agent doesn't have any access to that building, things will get more complicated. If we know a building is on fire but it isn't reachable, the agent will find the nearest area which the building can be extinguished from (distance from these areas is equal or less than our extinguish range). Then the agent will move to that place and start to extinguish the target for a fixed time until more buildings become reachable.

6. Search and Zoning

Having a good search algorithm results in more rescued civilians and more extinguished buildings; therefore, it is important to have a sufficient search algorithm to visit buildings faster.

6.1 K-means Clustering

K-means is one of the simplest algorithms that solve the well known clustering problem. The procedure follows a simple and easy way to classify a given data set through k clusters.

The main idea is to define k centers, one for each cluster. These centers should be chosen efficiently because different locations result in different outcomes. A better choice, in order to equalize the size of the clusters, is to place them as far from each other as possible. The next step is to associate each building with the nearest center, until no point is left without a cluster. At this point we need to re-calculate k new centroids as barycenter of the clusters resulting from the previous step. After we have these k new centroids, we have to reassign the buildings and the nearest new center. We repeat this step until the k centers don’t change their location. We are using this algorithm to divide our map into k clusters where k is the number of agents. [Fig 6] Afterwards, each cluster is assigned to an agent using the Hungarian Algorithm.