MIC

Independent

RoboCup Rescue Simulation League 2018

M I C

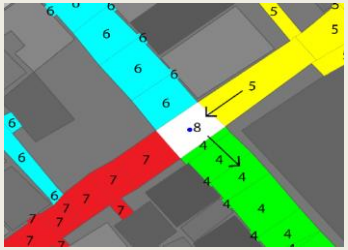
Introduction

- Agents
  - Police Force
  - Fire Brigade
  - Ambulance Team
- Clustering
  - DBSCAN
- Path Planning
  - Primary Path Planning
  - Ant Colony Optimization

Agents

Police Force

To have a better result on clearing the blockade, we used an algorithm that is similar to Ant Colony Optimization (ACO), and it works like the opposite of the ACO. The algorithm set a value to each road that the police force has been clear or already visited that road. And whenever the police visit that road again, it will add a number to the road value. The police force agent must visit and clean the roads with the fewer value. We used this algorithm, and we saw that more than ***90*** percent of the situations, police force cleared all possible roads.



An example of our clustering clearing method. As you can see, the police force agent has already visited the blue, red and yellow roads and it is going to visit and clear the green roads. In this method, the road which is a neighbour of a building, will not get any value more than its neighbour road; and the agent can pass the higher valued roads if one of its neighbours has less value. If in a road, were more than one police force agent, the extra police force agent, will search the world for fire, civilians and blockade.

Our police force agents will clear the blocked roads in three groups:

1. Fire zones

   For bringing the fire under our control, fire brigade agents need to have access to buildings which are on fire. Therefore, we should open their ways as fast as possible. Our police force agents only clear the routes to the building which are near or in the fire zones.

2. Human

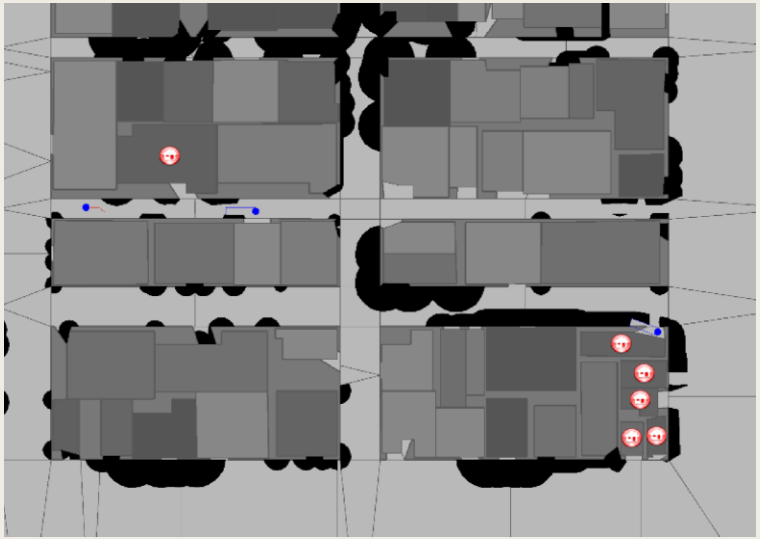
   After cleaning all possible roads, these agents search for any alive human whether if a human is stuck in a blockade or it is just behind the blockade. We use communication to be aware of the exact position of stuck agents.

Agents

Police Force

3. Refuge

We assign each refuge to one police agent. When simulation begins, police forces move immediately towards their assigned refuge and clear its entrance. Finally, As soon as all entrances are cleared, these police forces will join “Human” and “Fire Zones”agents to help them out.



There is also a probability that there would be just a building near a refuge and there are no roads nearby. In this situation, we need to get the joint edges of refuge and buildings near that refuge. We do this repeatedly until there is no nearby building left and we have reached the entrance. We also used an algorithm similar to DBSCAN. When simulation begins, agent checks if there are more than one refuges near each other. If this has happened, those refuges will be assigned to only one police force. If the map does not have any refuge, the police force will clear the way end to the hydrants.

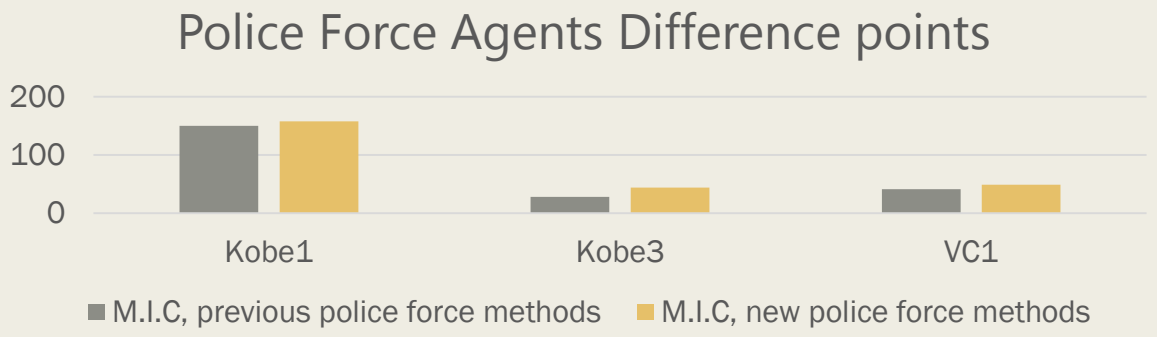
Agents

Police Force

We also developed another feature in order to minimize the wasted time on path clearing. Suppose that there are two types of roads exist on the map:

- Primary roads
- Building Entrances

We did some examinations on the elapsed time of clearing all roads in the map, and we concluded that the elapsed time remarkably decreases when police forces are not planning to clear buildings' entrances. This is because of the RS concept, where usually agent must move to the building's entrance to clear the blockade. In order to have the least wasted time in the process of clearing roads, Except for the police forces that have to clear refuges' entrances, we do not pay attention to any building and its entrance, until the ambulance agents command the police force agents to clear the blockade



Agents

Fire Brigade

The primary duty of fire brigade agents is to control the fire not to be extended and save other humans by extinguishing the fire near them. Fire search is designed to satisfy the following requirement:

1. Updating the shape of the fire
2. 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 zone, fire search tried to update the shape of the fire, and any other pieces of information of the fire zone by visiting the fire buildings near the newly born fire zone. For finding a new fire zone, we're using a clustering algorithm (DBSCAN) and a formula for searching the necessary roads, and areas.

Each of the important targets in roads has the same value as the other targets except the gas station. For example, each gas station value is equal to 10 wooden buildings. After setting the value to each of the roads, agents will start searching the roads from the highest value to the lowest value. Each of the high priority roads will be searched by 2 fire brigade agents; then they will search the other roads until they find a target.

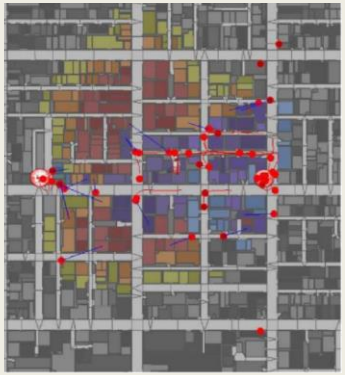
After finding fire zones and calculate the size of the fire, the most important thing to do is to, reach to that area and extinguish the on fire buildings. If the on fire building won't be reachable for the fire brigade agents (because of the roads blockade), they will command the police force to clear that blockade and if took a long time to remove the blockade, or because of the simulator problems, they stuck in there, they will try to prevent the fire transfers by filling up the nearest buildings to that fire zone.

Agents

Fire Brigade

Now we are working on a method, which will calculate and estimate the number of the needed fire brigade agents and the amount of the water that the building needs to be extinguished, using the building size, number of the building floors, material, fieryness, temperature and also agents water tank size, amount of the water that they have, distance between the each building and the agents, water pressure and etcetera. Using communication, agents will announce the building value and its information to each other, and the number of the needed agents will move to the target, and they will do their task.

There are some problems in extinguishing the fire that we try to solve. For instance, there are a lot of situations where the fire starts from a point and spreads around. Previously, we did not have any specific method to prevent the fire from spreading, but now, we developed a new algorithm to use on target allocation modules which helps us to achieve this purpose. In order to prevent the fire to be extended, we need to extinguish the fire from the outer-level of fire buildings. Regarding the fact that outer-level buildings have less fieryness value, so the first priority is the outer building because, while we are extinguishing them, we are preventing the fire transfers.



Before

After

Fire Brigades are preventing the fire from spreading by extinguishing the outer buildings. Fire brigade at first, will extinguish all yellow buildings after that the orange ones and then the red buildings. Because the high temperature of the fire, some of the fire brigade will fill up the neighbour buildings of the yellow buildings to prevent the fire transfer.

Agents

Ambulance Team

1. Ambulance Team Main Task

As soon as clustering calculations are done and finished, the same amount of agents will be assigned to each cluster. For instance, if there are n agents and k clusters on the map, each cluster will have $n\ k$ agents, but as this method might not perform as we expect. Then at the time we are writing this paper, we are actually working on a functionality which brings us a proper divider algorithm in order to assign the optimum number of agents to each cluster.
Ambulance agent needs to visit all buildings to have the best-updated world model of the civilians, to save time for the ambulance team agents, we worked on an algorithm which adds each building that the agent sees in a list of all possible entities that are in that building; and the other agents will not check those buildings again.
For no communication environments, we applied a self-organized task allocation strategy which will be a good help to the ambulance team rescuing operations. The allocation of the ambulance team agents to victims is performed based on particular specifications, like “death time,” burriedness of the civilians, and the distance between the agent and the target

2. Death Time

The cycle that a civilian dies in is called “Death Time.” We have already designed a new algorithm for our team, called “Death Time” which estimate the cycles that may cause a civilian die.

For estimating the death time, we are considering the target burriedness, health point, estimated burriedness removing, exact position URN and the distance to the refugee (if the map doesn't have any refugee, this variable will equal to 1).

Agents have the highest priority for the ambulance team, because in some situations, there is one ambulance free in the roads, and the rest of them are in a building, and they have burriedness, so the best strategy is rescuing the ambulances, fire brigades, and police forces at first, and then rescue the civilians.

Clustering

According to the basic K-Means clustering algorithm, we need to provide a variable k that contains a numeric value. This value will be the number of clusters. Being able to assign the number of clusters you need is a good feature but the necessity of having a specific value to provide, leads us to a static process of clustering, while rescue simulation is a dynamic environment. Initializing a dynamic clustering algorithm will help the entire simulation to be as real as possible and also we can gain more possible scores.

Unlike ambulance team, fire brigades need to shut off the fire with their collaboration. That means that agents should work together with a partly low distance between them. Although, the procedure of ambulance team is to spread out throughout the entire map to rescue damaged humans.

We decided to use DBSCAN as our clustering algorithm. The algorithm gets two values ε and m, where ε stands for radius and m, stands for minimum points included in radius. The overall procedure of DBSCAN is to find the neighbours of every point within ε radius, identify the core points with more than m neighbours and finally consider non-core points as noises. In this case, we choose a random building as a temporary-core point, get building-type objects in its ε range and then calculate a density rate based on the number of buildings in range. If density rate of each cluster is greater than m, that building and also nearby buildings in ε range will be included in a single cluster. Besides creating these clusters, another important point is the way we handle noise points. We have explained this in “Strategies” section.

Path Planning

Primary Path Planning

According to our examinations, A-Star acts perfectly in small graphs where we would have the shortest path in the shortest time. But as the graph becomes larger, AStar performs more like BFS algorithm and finally, as the graph becomes larger again, A-Star, BFS and Dijkstra will perform the same as each other. However, A-Star delivers much faster performance than other algorithms in any situation. In order to consider fire zones and blockades in our path calculation, we allocate more weight to some nodes based on the type of agent and its target, and automatically A-Star will not consider that node to calculate the best path.

Path Planning

Primary Path Planning

- Dijkstra

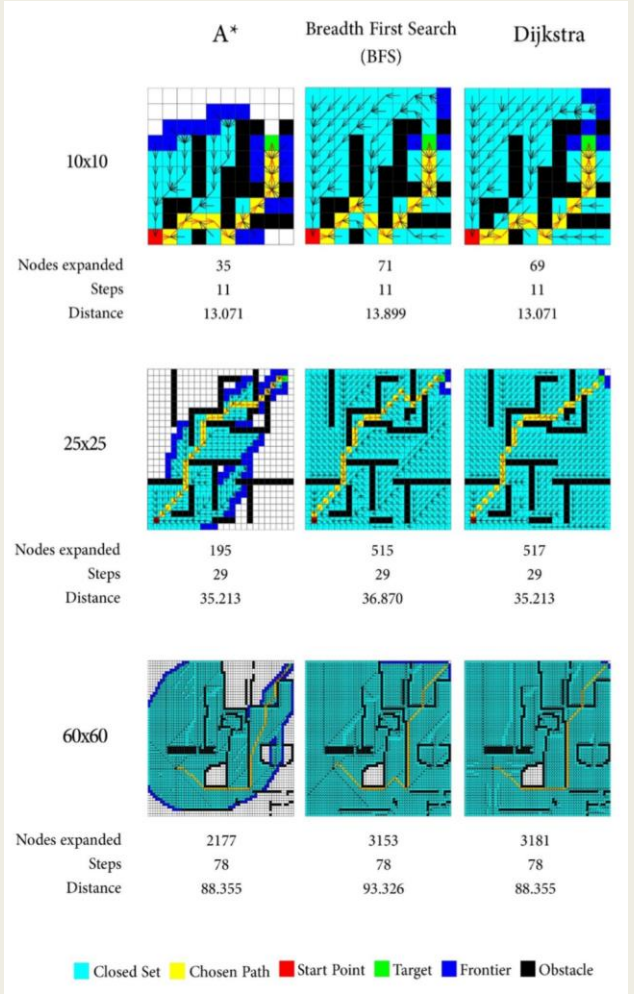
Dijkstra algorithm primary duty is to find the shortest paths between the nods in a graph. The way that the Dijkstra algorithm work is as follow:
1- At first, one of the nodes of the graph will be called as “initial node.”
2- Build a list of unvisited nods, and named them unvisited set.
3- For the current node, it will consider all of its neighbours.
4- Calculate the tentative distance between them.
5- The goal is to find the shortest way, so it will compare the distances, and it will select the smallest one.
6- After considering its all neighbours, it will mark the current node as visited and it will remove it from the unvisited set.
7- This will be done as long as checking unvisited nodes.
8- Now it will select the best path, using the smallest index.
But this algorithm has a critical problem because it uses sophisticated computing, it will take a long time to find the path, and it causes a problem in the agent's operations.

- Breadth-first search

Breadth-first search (BFS) is an algorithm for traversing or searching tree or graph data structures. This non-recursive implementation is similar to the non-recursive implementation of depth-first search, but differs from it in two ways:
1- It uses a queue instead of a stack (First In Last Out).
2- It checks whether a vertex has been discovered before enqueueing the vertex rather than delaying this check until the vertex is dequeued from the queue.
But this algorithm (BFS) has a little critical problem that will cause a delay in the agent's works because, in addition to checking the main routes, it will also review the sub-routes(paths that end in the other buildings). Because of that, we need a faster algorithm, so we decided to choose the “A*” algorithm.

- A-Star

“A-Star” pathfinding algorithm, is one of the best path planning algorithm because of its speed and accuracy to find the best path. A-Star uses the best-first search algorithm, and it finds the shortest path to the target by searching among of the possible paths.
A* is known as its fast pathfinding, so after our examinations, we decided to use the A-Star pathfinding algorithm for our path planning. A-Star is good alone, but it could be better with another algorithm that helps the agents to choose the best path to their targets. Therefore, we decided to use Ant Colony Optimization, to increase the result of our work.



Path Planning

Ant Colony Optimization

In the natural world, ants of some species and randomly, and upon finding food return to their group while laying down pheromone paths. If other ants find such a way, they are likely not to keep moving at random, but instead to follow the trail, returning and reinforcing it if they eventually see food. The overall result is that when one ant finds a right path from the colony to a food source, other ants are more likely to follow that way, and positive feedback eventually leads to all the ants following a single track. The idea of the ant colony algorithm is to mimic this behaviour with "simulated ants" walking around the graph representing the problem to solve. Ant Colony Optimization is a useful algorithm that helps our path planning system to estimate the fastest, cleanest and the best way for the agent's movement.

Thank You!

MIC
Independent

Seyedreza Vaziri Nejad
rezavnejad@gmail.com