RoboCup 2017
Rescue Simulation League Team Description
Aura (Iran)

Mobin Ghahramanpour, Arman Absalan, Alireza Kandeh

University of Tabriz, Iran
{ghahramanpour.mobin, armanaxh}@gmail.com, alirezaknd@yahoo.com

Abstract

Since we have started our collaborative work, we have been scrutinizing every paper in respect of RoboRescue competitions in the past years. Our concern is to find reliable and optimum path in possible short time because time limitation is a significant challenge. Instead of K-Means clustering algorithm, we have used DBSCAN algorithm. The algorithm helped us to make a CLEAN PATH which made path planning much easier. To find optimum path in minimum time in result graph of a given map, we have prune the path which was inevitable to pass such as leaves and 2-edges vertices, so we had small state space and could find satisfactory solution in minimum time. We also introduce a new A* Algorithm combined with the ant colony. And pass through concave shapes by converting it to convex shapes by ear clipping algorithm.

1 Introduction

When natural disaster happens, if humans would not have solutions, they might face serious problems. Over time, and history people have adopted to surviving dangerous situations. An organized complex with undisciplined approach would cause a negative effect on the process.

Clustering Algorithm checks every road and building to determine the neighbor value weight. Roads and buildings with lower neighbor value weight near each other are put in the same cluster. However, according to neighbor value weight, DBSCAN might create less or more clusters than number of specific agent number. We can handle this by setting weight limit depending on how many cluster system needs. DBSCAN might find clusters with arbitrary shapes, including CLEAN-PATH, that is, for every building there is a pass in cluster. So we never lose any chance of visiting buildings in a cluster. This dataset cannot be adequately clustered with k-means.

Path planning complexity is obvious. It is giving the chance to visit any building of map to determine civilians in them. We need shortest path available but it is not reliable unless we would know it is not blocked by blockades. Our algorithm finds a confident pass by combination of A* and Ant colony.

We outline our work in the remaining sections of this paper.

2 Modules

At first the most important action which must be taken, is to gather information of a map, such as, which buildings started to burn or which buildings have civilians in it. So we select a low-cost path in each cluster that passes through each vertex.

2.1 Clustering

We choose DBSCAN (Density Base Clustering) as our clustering algorithm since we face CLEAN PATH problem. In each cluster consisting of roads and buildings, we have CLEAN PATH, if there is a path to any of nodes in it, which path itself contains cluster nodes, CLEAN PATH problem happens when K-Means chooses a building, neighbor to a road or a building with a higher neighbor weight value (neighbor weight value is the price we take to reach the specific node). however, our aim is to choose a neighbor with lower neighbor weight value, an example is shown in Fig. 1.

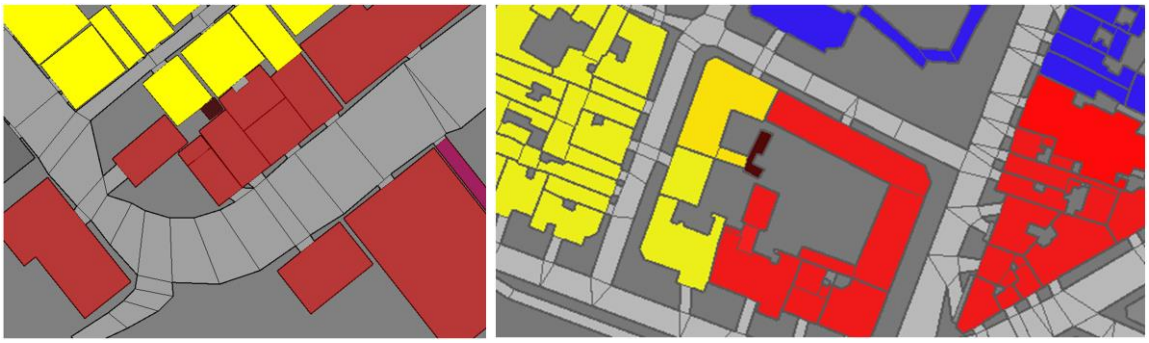


Fig. 1. A partial view of two maps clustered using k-means

How should an agent if it wants to get to the darkened building (darkened building are included in a red cluster in both maps) on the map though the path is available in another cluster (Fig. 1).
Neighbor weight value assignment (neighborhood of road to road, road to building and building to building are included) for road and building that we use in DBSCAN is as follows:

(1): Set current node a new node and set weight value to 0.

(2): For each possible choice, such as road or building which is reachable as next step (neighbor) from current node increase weight value by 1 and set it to current node and change the flag variable to visited.

(3) Repeat section (2) for not visited neighbors until we reach weight limit.

An example is shown in Fig. 2.

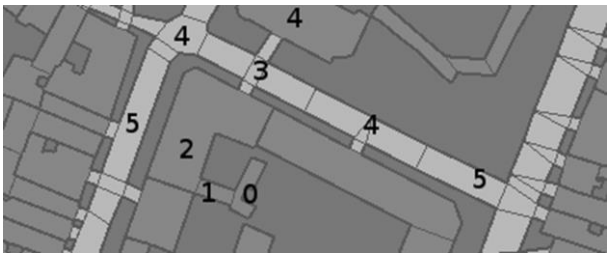


Fig. 2. A partial view of a map, neighbor weight value is shown on the map; starting node weight value is 0.

2.1.1 DBSCAN

DBSCAN [1] is a density-based algorithm which discovers clusters with arbitrary shape and with a minimal number of input parameters. The input parameters required for the algorithm is the radius of the cluster (Eps) and minimum points required inside the cluster (MinPts). In this paper we used a modified version of DBSCAN that uses cluster weight (CW, is the number that each cluster's node number cannot exceed of) and minimum points required inside the cluster (MinPts).
Assuming we have a total weight of neighbor weight value (road and building are included) of the map as TW, a total number of the specific agent set on the map as TN, thus each cluster weight (CW) is:

$$CW = TW / TN$$

Depending on CW value, DBSCAN continues adding nodes to cluster.

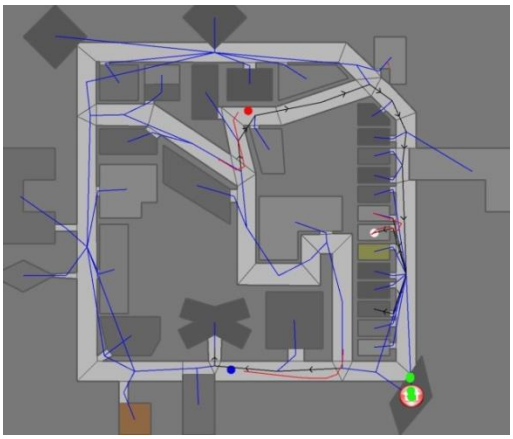
Fig. 2. Illustrates how DBSCAN can make clusters; we could have better performance on Path Planning Algorithms.

Clusters assigned to an agent by Hungarian algorithm [5].

2.2 Path Planning

2.2.1 Search Algorithm

To search on map we find a pass through each vertex with low-cost. This problem is NP-Hard. Result graph of the city map is a planar graph, to accomplish solving the problem, we make obtained graph of a cluster simpler. (Fig. 3).

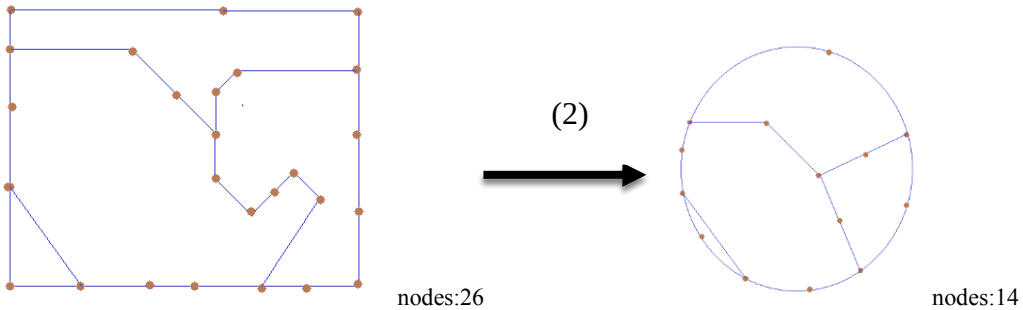


nodes:85

Fig. 3 a sample map cluster shows graph in blue

(1) Pruning leaves of the graph. Leaves are vertex that there is no longer way to a new vertex and need to trace back. After all, tracing leaves are inevitable, so we put them aside in process. (Fig. 4).

(2) After losing leaves we should prune again vertices with 2 edges that settled next to each other as they are inevitable to pass. They would be replaced by one vertex with 2 edges that connect to next vertex with more than 2 edges. (Fig. 5)



nodes:26 nodes:14

Fig 4 brown dots show vertices and blue lines show edges

A less complicated graph was obtained. (Fig. 3) Using a combination of Tabu search and Hill climbing [3] to find optimum path, might take a long time, notice that we have a time limit, so we will use best result obtained by the algorithm after a time limit we set. Some cycle after simulation starts, each agent traces its graph completely, so we have enough information of whole map.

2.2.2 Path Planning Algorithm

We introduce a new A* Algorithm combined with the ant colony [4].
The function is

$$f(x) = h(x) + g(x)$$

$g(x)$ is the cost of the path from the start node to *n*

$h(x)$ is a heuristic that estimates the cost of the cheapest path from *n* to the goal

$$h(x) = m(x) * (1 + {}^{1}/_{10} * p(x))$$

$m(x)$ calculates Manhattan heuristic function

$p(x)$ is the effect of trail pheromone in ant colony, The range is 0 to 1.

Each time an agent trace a path, $p(x)$ turns 1, each cycle it multiply by ¾, according to this information it never reach zero, we never lose each path agent passes, $p(x)$ just get decreased or increased.

Each agent has its own $p(x)$ table.through communication Agents share their table's content. In table we store path and their $p(x)$ value, thus they could know whether any other agent traced the path at any cycle or not. By adding $p(x)$ to $h(x)$, distance of a blocked path will be whether shortest or longest according to $p(x)$ value and agent calculates the $f(x)$.

Agent would calculate $h(x)$ as follows:

(1) For police force, we calculate $h(x)$ as follows:

$$h(x) = m(x) * (1 + {}^{1}/_{10} * p(x))$$

Since police force traces the whole map value of $h(x)$ will make path with trace history longest and with low probability of choice, path with no trace history or low trace history which is, had passed long cycle before will be shown shortest, and then police force can deliberately choose to clean not passed roads.

(2) For ambulance team, we calculate $h(x)$ as follows:

$$h(x) = m(x) * (1 - {}^{1}/_{10} * p(x))$$

Since ambulance teams aims to reach a target quickly in order to load trapped civilians, using this function, known roads that are with no blockade, have high probability of tracing.

(3) Fire brigade calculates $h(x)$ like ambulance team.

The shortest path is not the best path to be traced by agent, thus we choose the more confident path.

2.3 Target Allocators

2.3.1 Fire Brigade Allocation

Fire Brigade main idea is to protect civilians from fire nearby and prevent fire transfer, to more buildings. We partition buildings on fire and civilians we found on map newly, and then we select a civilian cluster with higher density near a fire cluster. Idle fire brigades will fix to a line between selected civilian cluster and fire cluster near fire cluster illustrated in Fig. 6.

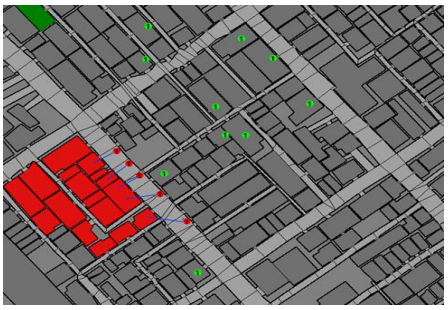


Fig. 6.

2.3.2 Police Force Allocation

Police Forces the most important task is to find connected components on a result graph of a given map. A graph is **connected** when there is a path between every pair of vertices. Minimum numbers of elements (vertex or edge) that would be removed to disconnect the remaining result graph are the nodes that police force selecting at first. (Fig. 7)

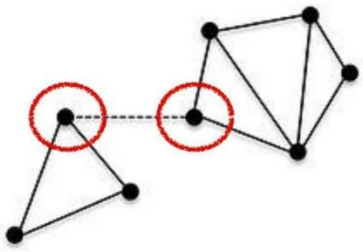


Fig. 7. This graph becomes disconnected when the dashed edge is removed. Connection components are shown as red.

Each police force searches its own cluster for connected components with blockade. Police force goes to clear blockade on those nodes at first. (Fig. 8). After this if there was any agent or civilian trapped in building closed by blockade or in a road with blockade will be rescued, and then each police in its cluster will clear roads lead to refuges in their cluster.

Police determines the shape of areas which it wants to clear. Shape may be convex or concave. This helps how we decide to clear area from entrance edge to exit edge. An agent can easily enter and exit directly from convex area but concave areas have some difficulties because direct pass line from entrance edge to exit edge through concave area may hit edge of shape and make interrupt in an agent's actions. So we should convert concave area to convex areas.

We use ear clipping [6] to convert concave to convex shapes. Through edges of convex shapes an agent can easily pass through concave shape. (Fig. 9).

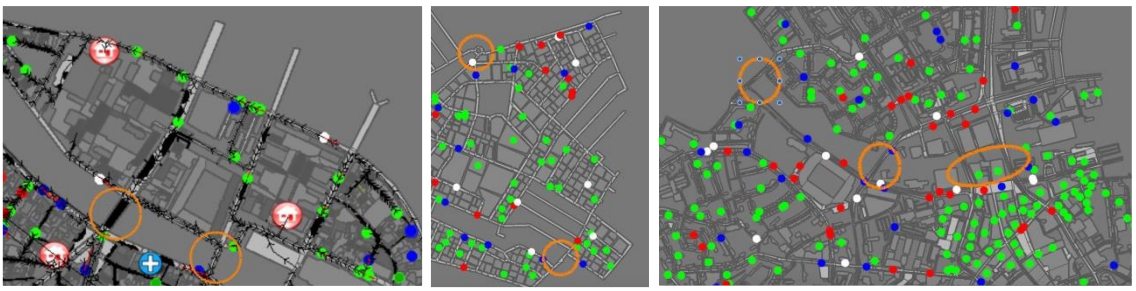


Fig. 8. A partial view a map shows a cluster with connected component that must clean first

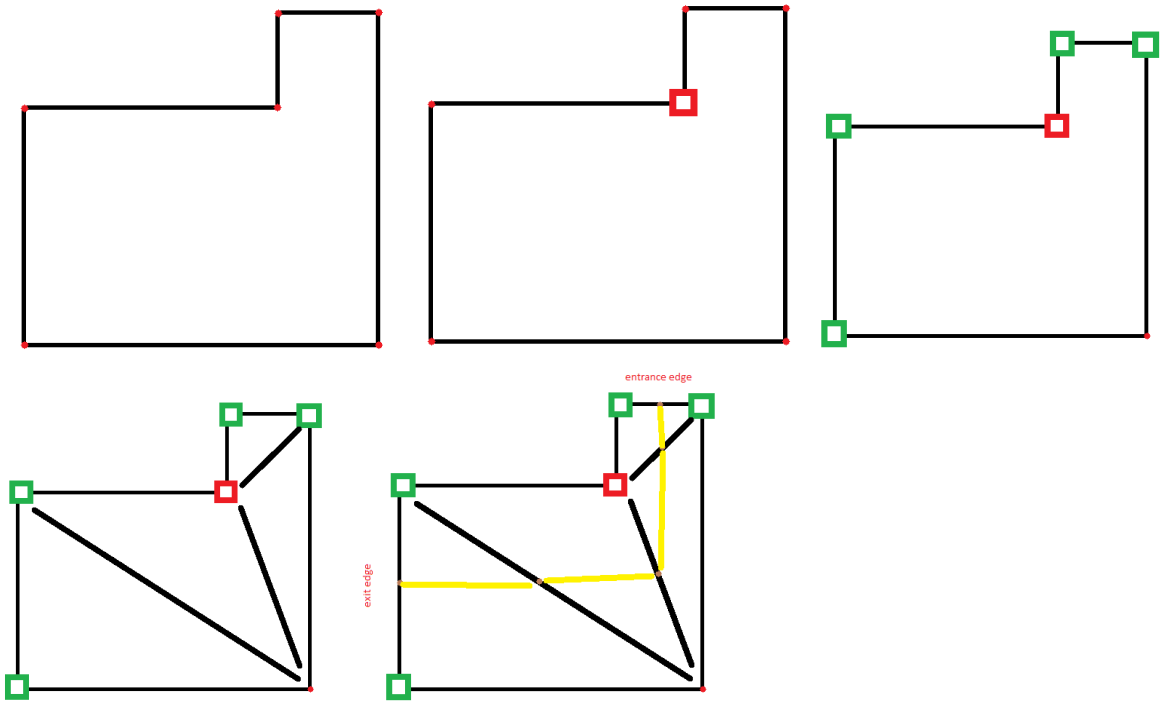


Fig. 9.

Red vertex is reflex vertex of concave area and green vertices are ear vertices. Black thick lines are edge of convex shapes created of concave area. Yellow lines are clear lines through concave area.

2.3.3 Ambulance Team Allocation

Ambulance Team uses search algorithm described before and partitions buildings on fire and list civilians near fire clusters with high rescue demand. ambulance team rescues any civilian with higher priority. Before that ambulance team will rescue any civilian at sight even if civilians were not at their cluster.

3 Conclusions

Instead of K-Means clustering algorithm, we have used DBSCAN algorithm. The algorithm helped us to make a CLEAN PATH which made path planning much easier. We also introduce a new A* Algorithm combined with the ant colony. And pass through concave shapes by converting it to convex shapes by ear clipping algorithm.

References

[1] Kamber, Han J. "Data Mining : Concepts & Techniques." 2001.

[2] M.Parimala. "A Survey on Density Based Clustering Algorithms for Mining Large." 2011.

[3] M. Yaghini, M. Momeni, M. Sarmadi. "Finding the Shortest Hamiltonian Path for Iranian Cities Using Hybrid Simulated Annealing and Ant Colony Optimization Algorithms." *International Journal of Industrial Engineering & Production Research*, 2011.

[4] Leo Willyanto Santoso, Alexander Setiawan, Andre K. Prajogo. "Performance Analysis of Dijkstra, A* and Ant Algorithm for Finding Optimal Path Case Study: Surabaya City Map." n.d.

[5] Harold W. Kuhn. 1955. The Hungarian Method for the assignment problem. Naval Research Logistics Quarterly. 83-97. Kuhn's original publication.

[6] David Eberly 2002 Triangulation by Ear Clipping