A Summary of Bachelor Test

Amin Abouee

July 11, 2019

1 Introduction

In this question, we want to find the optimal path from one location to another location. To find this path we can use a graph search algorithm, which works when the map is represented as a graph. The graph in this question, represented as a square grid graph. Each cell of this graph, contains data such as location in grid, the parent address, weight or cost for traverse in graph, and two boolean for finding the path and visited cells. In the case of algorithm for finding the shortest path between two cells or generally two nodes in a graph, there are lots of algorithms. The most three algorithms in this subject are:

1. Breadth First Search: Equally explores in all directions. This is an incredibly useful algorithm, when we just want to check the existance of a path from source to target, but it can not compute the minimum cost between these two nodes.

2. Dijkstra's Algorithm (Uniform Cost Search): Start from source and instead of growing in all direction, it favors to expand nodes with lowest cost. It expands all possible nodes and when a node visited (found), it guarantees that the estimated cost from source to desire node is minimum.

3. A* is a modification of Dijkstra's Algorithm that considers one axillary function (heuristic) to speedup the Dijkstra's algorithm. It optimized to find a single destination from a starting source. A* finds paths to one location, or the closest of several locations. By using heuristic function, it prioritizes paths that seem to be leading closer to target node.

In this solution, A* with a variety of heuristic functions were implemented to find the minimum cost between source and target. Selecting the heuristic function is completely depends to the map and the type of eligible movements that you can move on the graph (map). The heuristic function h(n) tells A* an estimate of the minimum cost from source to the target and so it is very important to choose a good heuristic function. If we assume the final cost function for each cell n in A* is $f(n) = g(n) + h(n)$. Based on efficiency of heuristic function, we can categorized the performance of A* in these three cases:

- If h(n) is always lower than the cost of moving from source to the target, then A* is guaranteed to find a shortest path. If a heuristic with lower h(n) is selected, more nodes expand, and getting it slower. Dijkstra's is A* with h(n) = 0

- If you select one heuristic function that exactly computes the cost of moving from source to the target, then A* will only traverse the best path and never expand anything else. It is getting very fast in this case but most of the time, it is very difficult to find this kind of heuristics for your problem.
- If you select the h(n) greater than the real cost of moving from source to the target, then A* does not guarantees to find a shortest path, but it is fast.

So regarding to above summary, choosing the heuristic function is very tricky and it is completely depends to our problem. For this problem, I implemented seven different methods for heuristic function to find the best one (most realistic one). Likewise, because we need to model the cost of up-hill and down-hill movements, nine different methods also implemented for this case. In the following, you can find the definition of them as well as a table for each group that compare them in the case of expansion nodes, total path, number of selected straight path and diagonal paths.

2 g(n) = Cost Function

1. Octile:
$$g(n) = \begin{cases} 1 & \text{if } move \text{ is straight} \\ \sqrt{(}2) & \text{if } move \text{ is diagonal} \end{cases} \tag{1}$$
2. Peak:
$$g(n) = max(elevation(source), elevation(target)) \tag{2}$$
3. MeanPeak:
$$g(n) = mean(elevation(source), elevation(target)) \tag{3}$$
4. L2:
$$cost = \sqrt{dx^2 + dy^2 + dz^2}$$
$$\begin{cases} cost & \text{if } move \text{ is up-hill} \\ \frac{1.2}{cost} & \text{if } move \text{ is down-hill} \end{cases}$$
5. L1:
$$cost = |dx| + |dy| + |dz|$$
$$g(n) = \begin{cases} cost & \text{if } move \text{ is up-hill} \\ \frac{2.0}{cost} & \text{if } move \text{ is down-hill} \end{cases}$$
6. L-Infinity:
$$cost = max(|dx|, |dy|, |dz|)$$
$$g(n) = \begin{cases} cost & \text{if } move \text{ is up-hill} \\ \frac{1.0}{cost} & \text{if } move \text{ is down-hill} \end{cases}$$

7. Angle:

$$cost = \angle(source, target)$$
$$g(n) = \begin{cases} cost/15 & \text{if } move \text{ is up-hill} \\ 0.0167 * cost & \text{if } move \text{ is down-hill} \end{cases}$$

8. Difficulty Level:

$$ang = \angle(source, target)$$
$$norm_2 = \sqrt{dx^2 + dy^2 + dz^2}$$
$$g(n) = \begin{cases} norm_2 * (ang/15) & \text{if } move \text{ is up-hill} \\ 0.0167 * norm_2 * ang & \text{if } move \text{ is down-hill} \end{cases}$$

9. L2 Trim:

$$ang = \angle(source, target)$$
$$norm_2 = \sqrt{dx^2 + dy^2 + dz^2}$$
$$g(n) = \begin{cases} norm_2 & \text{if } ang \text{ is less than 60} \\ \text{inf} & \text{if } move \text{ is more than 60} \end{cases}$$

In the following table, all methods examined where h(n) = 0

Cost Function g(n)				
Method	Expanded Cells	Total Path	Straight Path	Diagonal Path
Peak	3620212	6011	3546	2465
Mean-Peak	3573572	6011	3546	2465
L2	3576296	3092	2108	986
L1	4108631	3241	2304	937
Linf	3687362	3090	1548	1542
Angle	3786464	3141	1548	1593
Diff-Level	3952410	3236	1764	1472
L2-Trim	35556924	3094	2108	986

3 h(n) = Heuristic Function

1. Dijkstra:

$$h(n) = 0.0 \tag{4}$$

2. L2:

$$h(n) = \sqrt{dxy^2 + dz^2} \quad \text{where dxy is 1 or sqrt(2)}$$

3. L2-Altitude:

$$D = \text{compute an average altitude value, move from source to target}$$
$$h(n) = D * \sqrt{dxy^2 + dz^2}$$

4. L1:

$$h(n) = |dxy| + |dz|$$

5. L1-Altitude:

$$D = \text{compute an average altitude value, move from source to target}$$
$$h(n) = D * (|dxy| + |dz|)$$

6. Diagonal:

$$h(n) = max(dxy, dz) + (\sqrt{}(2.0) - 1) * min(dxy, dz)$$

7. Diagonal-Altitude:

$$D = \text{compute an average altitude value, move from source to target}$$
$$h(n) = D * (max(dxy, dz) + (\sqrt{}(2.0) - 1) * min(dxy, dz))$$

Following table, represents the evaluation of different heuristic methods for A* where g(n) for both up and down hill movements are L2

Cost Function g(n)				
Method	Expanded Cells	Total Path	Straight Path	Diagonal Path
Dijkstra	3576296	3092	2108	986
L2	2049376	3096	2110	985
L2Altitude	810639	3996	3356	640
L1	886534	3198	2322	876
L1Altitude	651415	4483	3826	657
Diagonal	1958704	3092	2102	990
DiagonalAltitude	620073	3887	3276	611