Report on Path Finding Problem

For pathfinding, A\* algorithm repeatedly examines the most promising unexplored location it has seen. When a location is explored, the algorithm is finished if that location is the goal; otherwise, it makes note of all that location’s neighbors for further exploration. A\* Search algorithms, unlike other traversal techniques, it has “brains”. What it means is that it is really a smart algorithm which separates it from the other conventional algorithms. It works on the formula “successor.f = successor.g + successor.h ” choosing the least values of f;

Where,

**g** = the movement cost to move from the starting point to a given square on the grid, following the path generated to get there.  
**h** = the estimated movement cost to move from that given square on the grid to the final destination.

This is often referred to as the heuristic, which is nothing but a kind of smart guess. We really don’t know the actual distance until we find the path, because all sorts of things can be in the way (walls, water, etc.). For solving this problem two methods were implied :

Manhattan distance :

h = abs (current\_cell.x – goal.x) + abs (current\_cell.y – goal.y)

Euclidean distance :

h = sqrt ( (current\_cell.x – goal.x)^2 + (current\_cell.y – goal.y)^2 ).

For these traversals it chose the most optimal math possible and for the both heuristics it gave the same optimal solution.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |
|  | x | x | x | x | ✓ | ✓ | ✓ |
|  | x |  |  | x | ✓ | x | ✓ |
|  | x | ✓✓  <start> | ✓ | x | ✓ | x | ✓✓  <end> |
|  | x |  | ✓ | x | ✓ | x |  |
|  | x |  | ✓ | ✓ | ✓ | x |  |
|  | x | x | x | x | x | x |  |
|  |  |  |  |  |  |  |  |