Question 6 from assignment 1:

First lets start with the results, all three test had a path of 72, A\* with regular step count had 104 spot explored. Greedy and A\* with StepTurn explored 79 spots. So why did Greedy and A\* with StepTurn perform better than basic A\* is the big question. The answer is hidden within how A\* calculate the cost function and how tie are handled.

Lets break it down.

When A\* calculates the cost function for each node it adds two things

it looks how many steps the node is from the start

and it checks the Manhattan distance to the goal

pretty simple, but these action cancel if the cost of going down a path is the same as the Manhattan distance.

|  |  |  |  |
| --- | --- | --- | --- |
| Start -> | Node -> | Node -> | Goal |
| Cost = 0, Manhattan = 3, total = 3 | Cost = 1, Manhattan = 2, total = 3 | Cost = 2, Manhattan = 1, total = 3 | Cost = 3, Manhattan = 0, total = 3 |

Because this mechanism of ties happens the order in which the nodes are placed in the frontier list is determined by how to priority queue is implemented. Because of this the explored list increases compared to Greedy algorithm. The Greedy algorithm only checks the Manhattan distance and thus does not fall into this situation. A\* with step Turn also does not fall into this situation because although it still use both the cost and Manhattan distance the cost is not mirror the Manhattan distance.