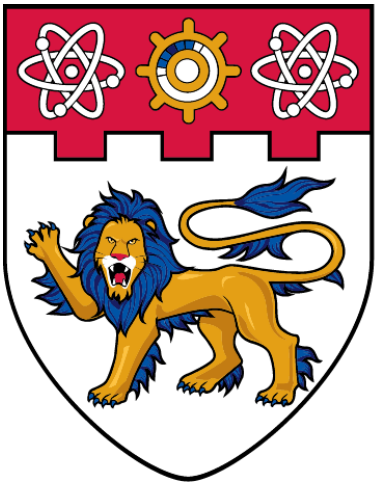


NANYANG TECHNOLOGICAL UNIVERSITY
SINGAPORE

CZ3005 Artificial Intelligence
Lab Assignment 1

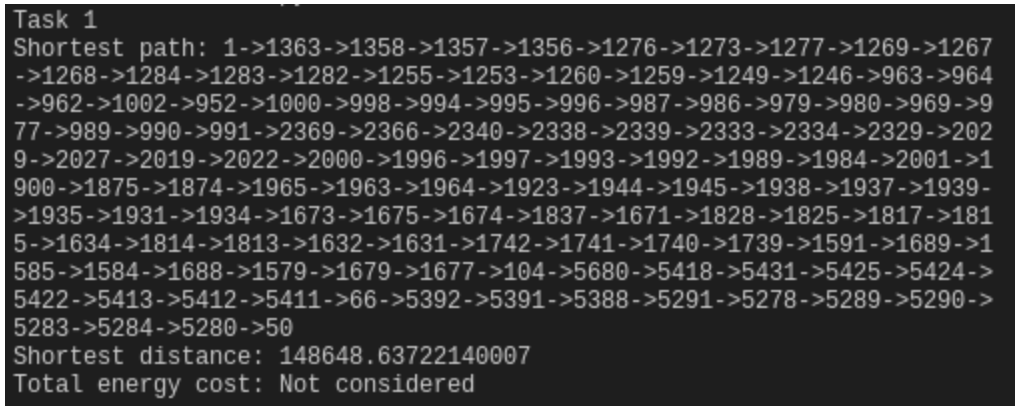
Submitted By: Goh Zongye Benjamin
Matriculation Number: U2022728L
Lab Group: SSP1

Introduction

As stated by the assignment specifications, the main application (main.py) is split into 3 parts. 1st is the relaxed NYC instance without checking for the energy consumption, 2nd is the same algorithm as the 1st but with an energy budget and the 3rd is a A* search algorithm with the energy budget.

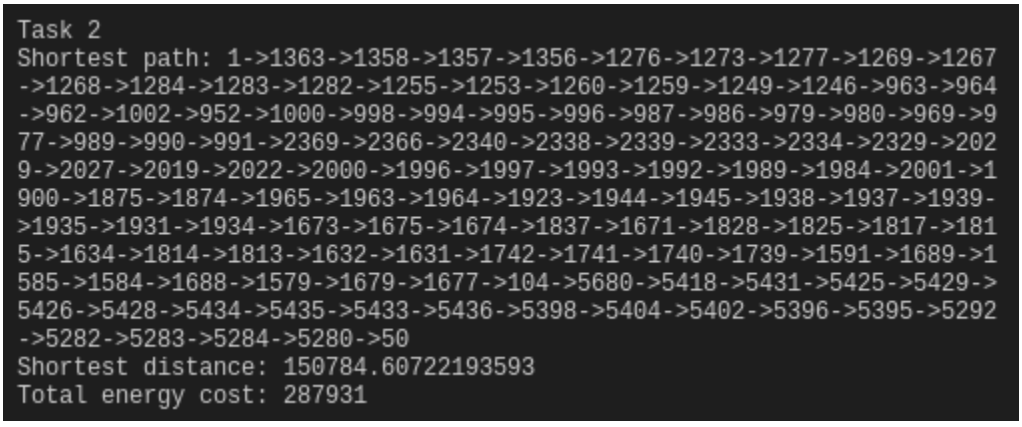
Task 1

For task 1, I have decided to use a simple UCS algorithm as it seemed to the best of the uninformed search algorithms. The UCS algorithm was modified from an existing example online to fit the assignment specifications [1].



Task 2

For task 2 the same UCS algorithm is used, with it modified for the paths to hold the energy cost and ensuring the energy budget of 287,932 is never exceeded.



As somewhat expected, the path distance is now increased to keep the shortest path within the energy budget. The initial shortest path was probably set to be larger than the energy budget to showcase the change with taking energy budget into account.

Task 3

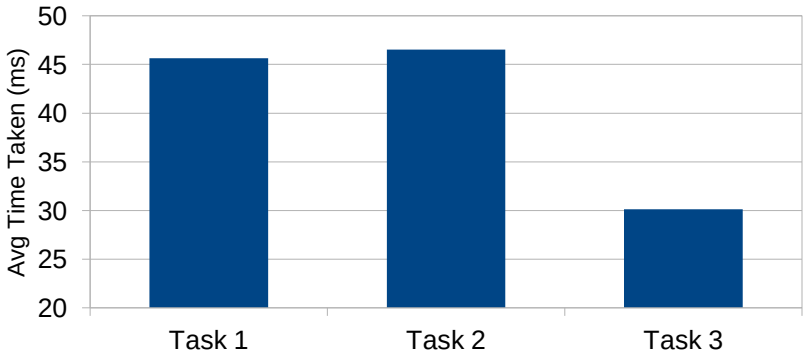
For task 3, I am required to use the A* search algorithm to find the shortest path. To do this, the UCS algorithm from task 2 is utilised as the g(n) function and writing my own heuristic function h(n) to approximate the distance from the next node to the destination.

Initially, for my h(n) it was a basic distance of the node coordinate to the destination coordinate. It can be easily found by using Pythagoras theorem with the difference in x and y positions being the width and the height of the "triangle".

```
Task 3
Shortest path: 1->1363->1358->1357->1356->1276->1273->1277->1269->1241
->1240->1235->956->953->955->957->948->949->952->1000->998->994->995->
996->987->986->979->980->969->977->989->990->991->2465->2466->2384->23
82->2385->2379->2380->2445->2444->2405->2406->2398->2395->2397->2140->
2141->2125->2126->2082->2080->2081->2073->2075->2164->2163->2162->2159
->2158->2160->71->61->60->1851->1847->1846->1824->1797->1787->1775->17
61->1760->5523->5483->5479->5470->5403->5402->5396->5395->5292->5282->
5285->5284->5280->50
Shortest distance: 162975.6345373529
Total energy cost: 272492
```

The result for the A* search implementation was that the shortest path is longer than that of the UCS algorithm in task 2. A first look might indicate the A* plainly being a worse overall algorithm to use than the UCS but if we can take a closer look at the 3 different algorithms.

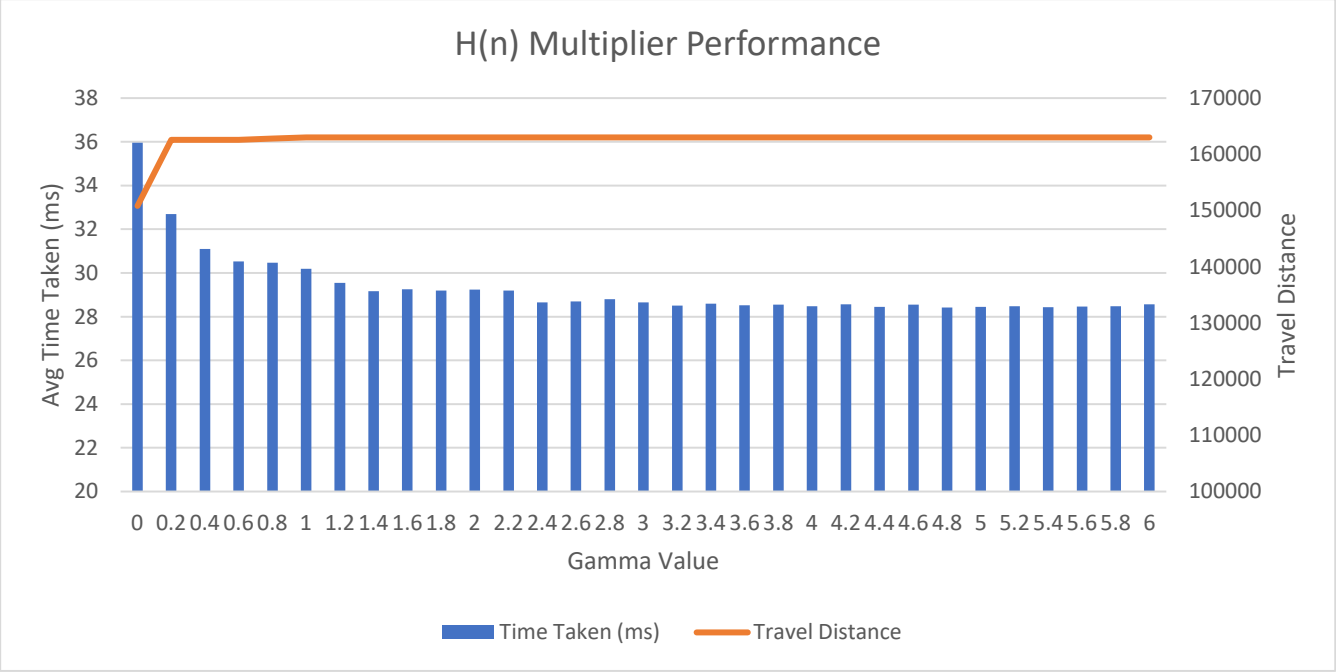
Additional Analysis



Above is the result of running each task 5000 times and getting the average time taken. If we compare time taken to find the path for task 1 and 2, it is to be expected that task 2 takes slightly more time as it has to reject some paths because of the energy budgeting.

But if we look at the time difference between task 2 and 3, the A* search takes 33% less time than the UCS. This shows that most probably, the heuristic method of just using the straight distance is fair enough approximation find the path. (An 8% increase in distance for 33% less computational time).

The A* algorithm at this point has not yet been fully explored and the heuristic function can still be tweaked further.



By adding a multiplier gamma to the heuristic function (f(n) = g(n) + y*f(n) now), we can scale how much impact it will play into the node selection order. I have tried gamma value from 0 to 6, with increments of 0.2, showing the performance difference in the chart above.

From my analysis, the travel distance peaks at 163008 with y=1 and time taken to find the path troughs at 28.66 ms with y=2.4.

From this, we can choose whether we want the actual shortest path by setting y=0 or have the fastest search time but at the cost of an inaccurate shortest path with y=2.4.

Conclusion

From finishing this lab assignment, I have learned that there is no perfect algorithm to fit every need. The optimal algorithm for a use case may not be the optimal algorithm for another. For example, if I wanted the exact shortest path distance, UCS would be my choice but for a faster path generation with a close enough approximation, A* would be better.

This is all under the heuristic function being the direct distance from node to destination. But if a better heuristic function is found, it might invalidate the UCS for shortest path to destination.

References

[1] https://gist.github.com/Kautenja/29f13b543ecd210202417dfc9e328249