Ian Quinn, Jared Tulayan, Shivam Verma
Assignment 1 - Report

Map Generation

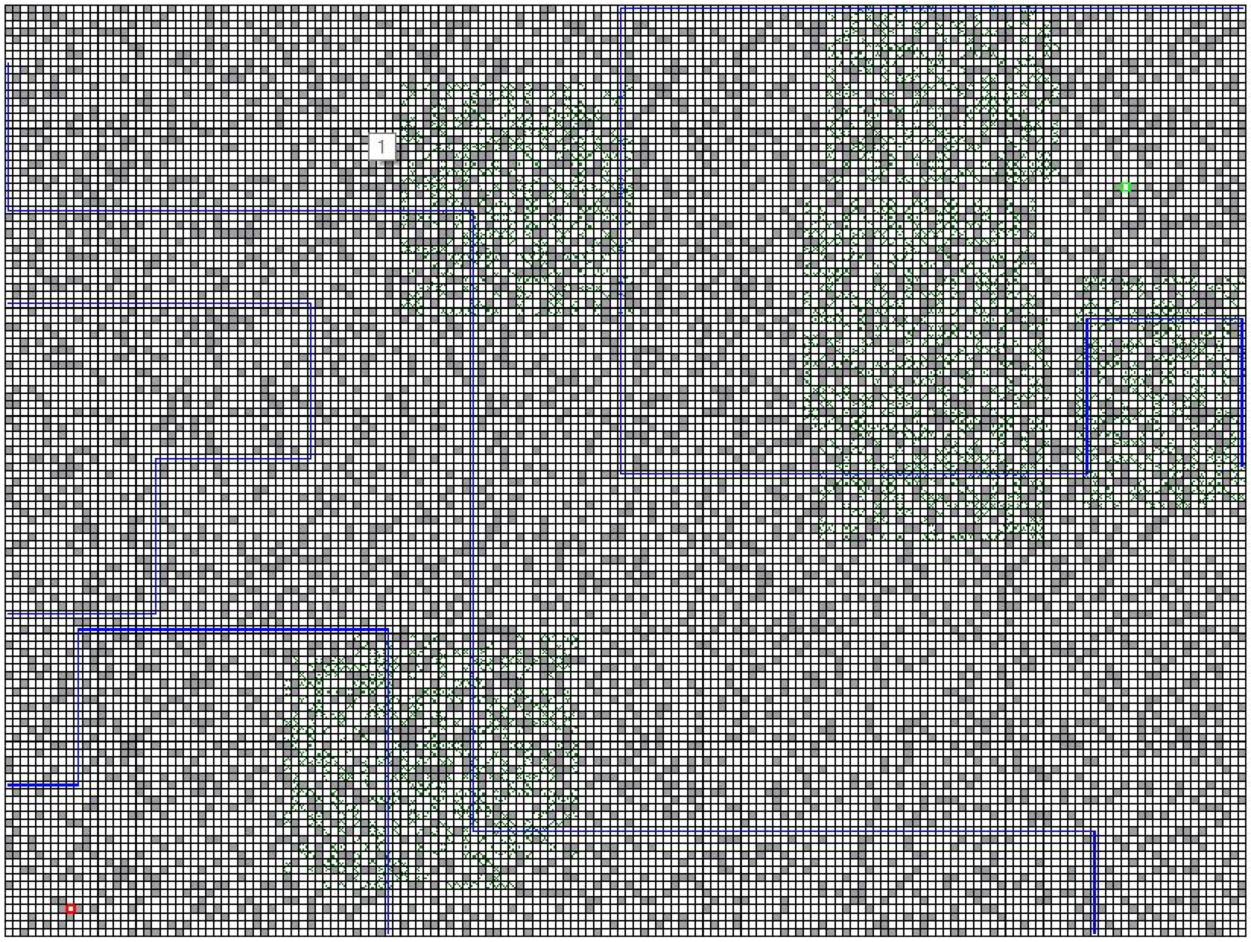


Figure 1 - A zoomed out representation of the map - Blue lines = Highways, White boxes = Unblocked, Gray Boxes = Blocked, Green Markings = Hard to Traverse, Red Square = Goal, Green Square = Start, and the 1 in the box near the top shows how when you hover the mouse on a square, it tells you what type of vertex it is.

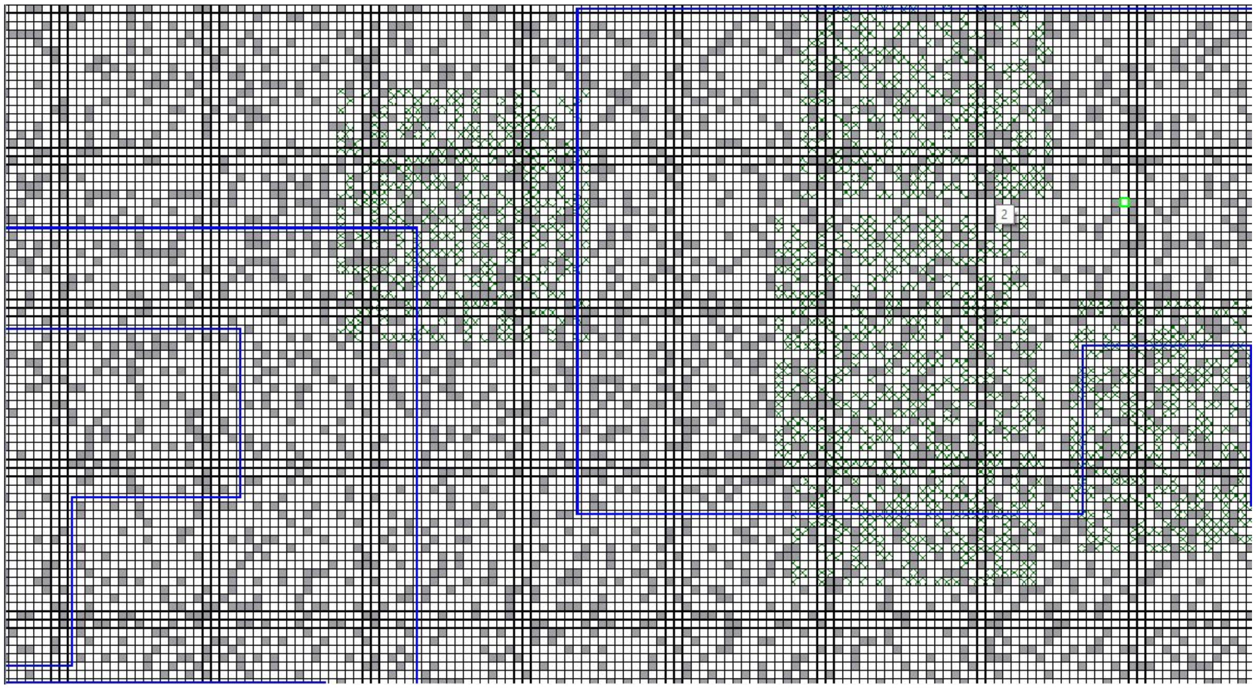
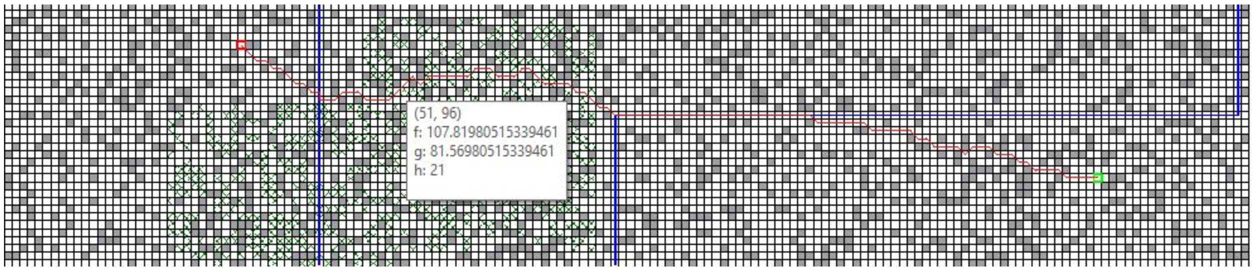


Figure 2 - Shows a zoomed in portion of the map, in this case it is the upper right side of the map in Figure 1



```
(51, 96)
f: 107.81980515339461
g: 81.56980515339461
h: 21
```

Figure 3 - Shows a zoomed in portion of the map after it has been solved using Sequential A. The red line shows the path of the agent and as shown, by placing the mouse over any given cell, it tells the f,g, and h value of the cell.*

Algorithm Optimization

We started off by writing out the algorithms in full force. Each method was its own algorithm. From there, we looked at the similarities in the three algorithms and worked to create abstract methods that could simply be called by the respective algorithm. This included created abstract methods for expanding the vertices, getting the cost of transitioning, and updating the values of the vertices. The numbers shown below highlight the change from optimizing the code in this manner in terms of line counts for each algorithm.

A* algorithm originally - 350 lines → 50 lines
Weighted A* originally - 350 lines → 50 lines
Uniform-Cost algorithm originally - 220 lines → 70 lines
Sequential A* originally - 550 lines → 90 lines

After this, we further optimized the algorithms; first by splitting them up into different files rather than having them all as different functions in the same file. Next, we combined A* and Weighted A* so, rather than two functions, they are in the same function that uses a weight as a parameter. If that weight = 0, then it means it's regular A* being used, if it is a different number, then it means Weighted A* is being used. In addition, for the moving forward of each vertex, the implementation has been placed into a for loop rather than a more manual implementation which was being used earlier.

A final optimization was done later as we realized all the algorithms: A*, Weighted A*,Sequential A*, and Uniform Search were just small variations of the same algorithm. To this end, we used Sequential A* as the base algorithm, with all of the other being called with slight variations. For example, for the uniform cost algorithm, it is simply the A* algorithm with all h values being set to 0 (It is just focused on the actual cost). Similarly, for weighted A*, the heuristic specified is our default heuristic, and we are using weightings of 1.25 and 2 to measure.

Heuristic Values
The 5 heuristic values we chose to implement were the Diagonal (Axis) Distance, Euclidean distance, Manhattan Distance, Manhattan Distance for a hexagonal grid, and a custom formula.

The diagonal, or axis distance returns the maximum of either the x distance or the y distance between the current coordinate and the goal coordinate

The Euclidean Distance finds the hypotenuse of a right triangle created by the goal coordinate and the current coordinate the agent is at

The Manhattan Distance returns the sum of the absolute values of the distances between the x/y coordinates of the current location and the goal location

The Manhattan Distance for a hexagonal grid is technically supposed to be ideal for finding distances on a grid made up of hexagonal tiles rather than square tiles as we have here, but we thought it could be interesting to see its results on a square grid.

Finally, the custom formula finds the euclidean distance from the start coordinate to the end coordinate and then subtracts from that the euclidean distance from the current coordinate to the start coordinate.

Upon conducting tests of each heuristic, we found that the Manhattan Distance was the only one out of them all which was admissible and consistent as per the triangle inequality.

Experimental Results
Discuss experimental results of 50 benchmarks with the 3 algorithms. Talk about the impact of different heuristics and discuss relative performance of each algorithm.

Test 1 - Weighted A* (w = 1.25), Base Heuristic = Euclidean (Pythagorean)

	A*	Sequential A*	Weighted A*	Uniform-Cost
Number of Expansions (Average)	3,713.76	2,798.18	963.4	14,430.08

Test 2 - Weighted A* (w = 2.50), Base Heuristic = Euclidean (Pythagorean)

	A*	Sequential A*	Weighted A*	Uniform-Cost
Number of Expansions (Average)	3,206.37	2,067.97	386.1	15,222.40

Test 3 - Weighted A* (w = 1.25), Base Heuristic = Manhattan

	A*	Sequential A*	Weighted A*	Uniform Cost
Number of Expansions (Average)	2408.4	2672.96	824.2	14543.52

Test 4 - Weighted A* (w = 2.50), Base Heuristic = Manhattan

	A*	Sequential A*	Weighted A*	Uniform-Cost
Number of Expansions (Average)	2,237.2	2,421.47	382.63	15,402.63

Test 5 - Weighted A* (w = 1.25), Base Heuristic = Diagonal (Axis)

	A*	Sequential A*	Weighted A*	Uniform-Cost
Number of Expansions (Average)	5699.24	2091.48	4373.52	14,694.56

Test 6 - Weighted A* (w = 2.50), Base Heuristic = Diagonal (Axis)

	A*	Sequential A*	Weighted A*	Uniform-Cost
Number of Expansions (Average)	5,905.27	2,270.43	414.9	15,865.5

Test 7 - Weighted A* (w = 1.25), Base Heuristic = Manhattan-Hexagonal Grid

	A*	Sequential A*	Weighted A*	Uniform-Cost
Number of Expansions (Average)	18,691.72	2,256.84	14,418.08	14,974.92

Test 8 - Weighted A* (w = 2.50), Base Heuristic = Manhattan-Hexagonal Grid

	A*	Sequential A*	Weighted A*	Uniform-Cost
Number of Expansions (Average)	13,687.3	2,245.27	47,264.27	14,493.83

Test 19 - Weighted A* (w = 1.25), Base Heuristic = Custom Heuristic

	A*	Sequential A*	Weighted A*	Uniform-Cost
Number of Expansions (Average)	12,988.04	2,799.80	38,744.24	14,892.88

Test 10 - Weighted A* (w = 2.50), Base Heuristic = Custom Heuristic

	A*	Sequential A*	Weighted A*	Uniform-Cost
Number of Expansions (Average)	15,927.27	1,891.63	109,395.87	14,446

Average Runtime - Optimal Case (w = 2.5, Heuristic = Manhattan)

	A*	Sequential A*	Weighted A*	Uniform-Cost
Average Algorithm Runtime (Seconds)	0.1977 seconds	0.3890 seconds	0.0838 seconds	1.0886 seconds

Conclusions

Upon conducting all the tests of each algorithm/heuristic, we found that using a larger weight (2.5 vs 1.25) resulted in more optimal performance for Weighted A*. In addition, we had our most optimal performance when using the one admissible/consistent heuristic - The Manhattan Distance as compared to any other heuristics. This makes sense considering that is the one heuristic that provides optimal and accurate results every time. In addition, we had two heuristics which were especially inefficient - Manhattan Distance for a Hexagonal Grid and our Custom Heuristic. We found that for these heuristics, A* and Weighted A* took significantly more expansions/moves to reach the end goal. Over all the heuristics, though, Uniform-Cost and Sequential A* had much lower variability in average results as compared to A* and Weighted A*. This is due to the fact that Uniform-Cost is not affected by a heuristic and that Sequential A* uses all the heuristics, so having a different base heuristic should not impact results.

Overall, in our optimal case of a weighting = 2.5 and base heuristic - Manhattan, Weighted A* was our most efficient algorithm followed by A* and Sequential A* as a close second and third and then Uniform-Cost Search as a distant fourth. We believe the reason for these results is because Weighted A* places increased emphasis on nodes closer to the end goal, thereby reaching that end goal much faster.