MEEN 689: Planning and Control of Autonomous Vehicles

Assignment 1

Arunachalam Venkatachalam

333000500

Assumptions

1. Assumed the given start and goal nodes are of the form (x, y) - (col, row). Interchanged the order to read it, since in python an array element is indexed as array[row][col].
2. Assumed 8 moves are possible (left, right, up, down, left-up, left-down, right-up, right-down).
3. Assumed the cost to any neighbor is 1. Although the cost to diagonal nodes should be 0.7, the cost was assumed to be 1.
4. Euclidean distance was chosen as the heuristic because it should give optimal results in comparison to Manhattan distance, given the fact that 8 moves are employed.

Run time for all resolutions

Resolution	**Start**	**Goal**	**Computation times (in seconds)**		
			Dijkstra	**A***	
2*2m	(224, 158)	(232, 1468)	0.702996	**0.478084**	
	(224, 158)	(964, 870)	0.500125	**0.461026**	
	(224, 158)	(304, 72)	Goal not reachable		
	(224, 158)	(274, 840)	**0.72151**	0.815546	**Goal not reachable**
	(436, 892)	(232, 1468)	0.221901	**0.111033**	
	(436, 892)	(964, 870)	0.29264	**0.138638**	
	(436, 892)	(304, 72)	Goal not reachable		
	(436, 892)	(274, 840)	**0.933159**	1.42381	**Goal not reachable**
5*5m	(224, 158)	(232, 1468)	0.097616	**0.066313**	
	(224, 158)	(964, 870)	**0.05889**	0.064222	
	(224, 158)	(304, 72)	Goal not reachable		
	(224, 158)	(274, 840)	**0.091575**	0.102785	**Goal not reachable**
	(436, 892)	(232, 1468)	0.024431	**0.017098**	
	(436, 892)	(964, 870)	0.04637	**0.015282**	
	(436, 892)	(304, 72)	Goal not reachable		
	(436, 892)	(274, 840)	**0.09253**	0.102056	**Goal not reachable**
10*10m	(224, 158)	(232, 1468)	**0.018954**	0.020712	
	(224, 158)	(964, 870)	0.018822	**0.018465**	
	(224, 158)	(304, 72)	Goal not reachable		
	(224, 158)	(274, 840)	0.034783	**0.027589**	**Goal not reachable**
	(436, 892)	(232, 1468)	0.013324	**0.002873**	
	(436, 892)	(964, 870)	0.013365	**0.003586**	
	(436, 892)	(304, 72)	Goal not reachable		
	(436, 892)	(274, 840)	0.030372	**0.022159**	**Goal not reachable**
1*1m	(224, 158)	(232, 1468)	3.289191	**2.140017**	
	(224, 158)	(964, 870)	2.919911	**1.874934**	
	(224, 158)	(304, 72)	Goal not reachable		
	(224, 158)	(274, 840)	**3.333529**	4.477508	**Goal not reachable**
	(436, 892)	(232, 1468)	1.273725	**0.557296**	
	(436, 892)	(964, 870)	1.435158	**0.626574**	
	(436, 892)	(304, 72)	Goal not reachable		
	(436, 892)	(274, 840)	**3.427423**	4.719319	**Goal not reachable**

Run time for all resolutions when goal is reachable, and a path exists

Resolution	Start	Goal	Computation times (in seconds)	
			Dijkstra	**A***
2*2m	(224, 158)	(232, 1468)	0.702996	0.478084
	(224, 158)	(964, 870)	0.500125	0.461026
	(436, 892)	(232, 1468)	0.221901	0.111033
	(436, 892)	(964, 870)	0.29264	0.138638
5*5m	(224, 158)	(232, 1468)	0.097616	0.066313
	(224, 158)	(964, 870)	0.05889	0.064222
	(436, 892)	(232, 1468)	0.024431	0.017098
	(436, 892)	(964, 870)	0.04637	0.015282
10*10m	(224, 158)	(232, 1468)	0.018954	0.020712
	(224, 158)	(964, 870)	0.018822	0.018465
	(436, 892)	(232, 1468)	0.013324	0.002873
	(436, 892)	(964, 870)	0.013365	0.003586
1*1m	(224, 158)	(232, 1468)	3.289191	2.140017
	(224, 158)	(964, 870)	2.919911	1.874934
	(436, 892)	(232, 1468)	1.273725	0.557296
	(436, 892)	(964, 870)	1.435158	0.626574

Run time for all resolutions when there are obstacles in-between the points and goal couldn’t be reached

Resolution	Start	Goal	Computation times (in seconds)		
			Dijkstra	**A***	
2*2m	(224, 158)	(304, 72)	Goal not reachable		
	(224, 158)	(274, 840)	**0.72151**	0.815546	**Goal not reachable**
	(436, 892)	(304, 72)	Goal not reachable		
	(436, 892)	(274, 840)	**0.933159**	1.42381	**Goal not reachable**
5*5m	(224, 158)	(304, 72)	Goal not reachable		
	(224, 158)	(274, 840)	**0.091575**	0.102785	**Goal not reachable**
	(436, 892)	(304, 72)	Goal not reachable		
	(436, 892)	(274, 840)	**0.09253**	0.102056	**Goal not reachable**
10*10m	(224, 158)	(304, 72)	Goal not reachable		
	(224, 158)	(274, 840)	0.034783	**0.027589**	**Goal not reachable**
	(436, 892)	(304, 72)	Goal not reachable		
	(436, 892)	(274, 840)	0.030372	**0.022159**	**Goal not reachable**
1*1m	(224, 158)	(304, 72)	Goal not reachable		
	(224, 158)	(274, 840)	**3.333529**	4.477508	**Goal not reachable**
	(436, 892)	(304, 72)	Goal not reachable		
	(436, 892)	(274, 840)	**3.427423**	4.719319	**Goal not reachable**

Computation time of the algorithm for the different resolutions

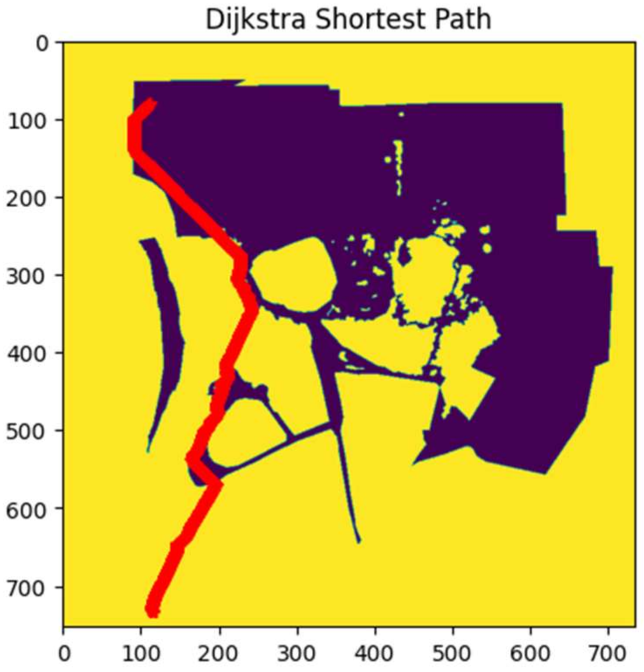
- The run time for both Dijkstra and A* reduces as the resolution decreases because the number of nodes to be considered during search is fewer.
- For some pairs, even at lower resolutions, the computation times are relatively high, indicating that the problem remains complex due to obstacles or the layout of the terrain.
- The difference in performance between the two algorithms is most pronounced at finer resolutions (1 * 1m)

Computation time and results of both the algorithms. Are either of them optimal?

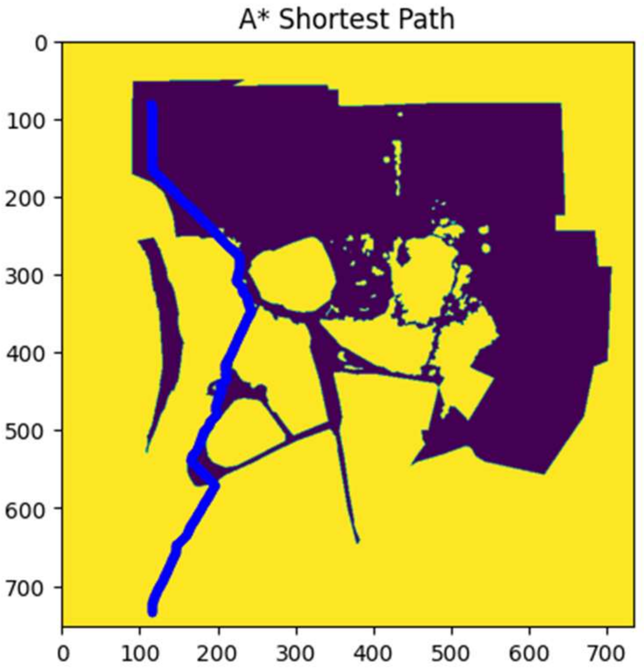
- **In terms of computation time**, A* is comparatively more optimal than Dijkstra due to the presence of heuristic which directs the search strategy towards the nodes which are more optimal, i.e., it leverages heuristics to make more informed decisions about which nodes to explore next.
    - When a path exists, the run time of A* is smaller comparative to Dijkstra because Dijkstra follows a brute-force approach, i.e., checks for all the possible neighboring nodes before choosing the shortest path whereas the search strategy of A* by definition prioritizes the nodes that are more likely to lead to the goal.
    - When the goal couldn’t be reached, both algorithms will explore the entire search space. In this case, Dijkstra's algorithm typically explores nodes uniformly, while A* may still be guided by the heuristic towards the goal initially. However, as the search progresses and it becomes clear that there's no path, A* will have spent extra time exploring nodes that Dijkstra's algorithm would not have explored as extensively.
- **In terms of path length**, both Dijkstra's algorithm and A* are guaranteed to find the shortest path.

Resolution: 2*2 m

Start: (112,79) | Goal: (116, 734)



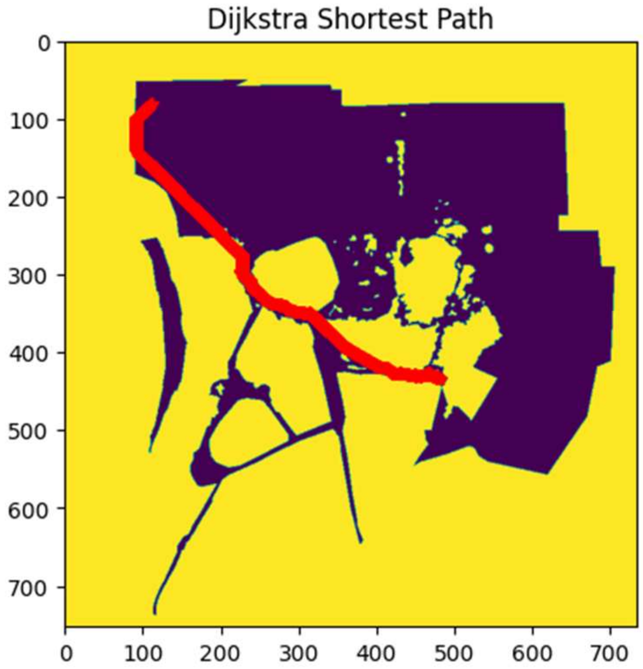
Run Time: 0.702996 seconds



Run Time: 0.478084 seconds

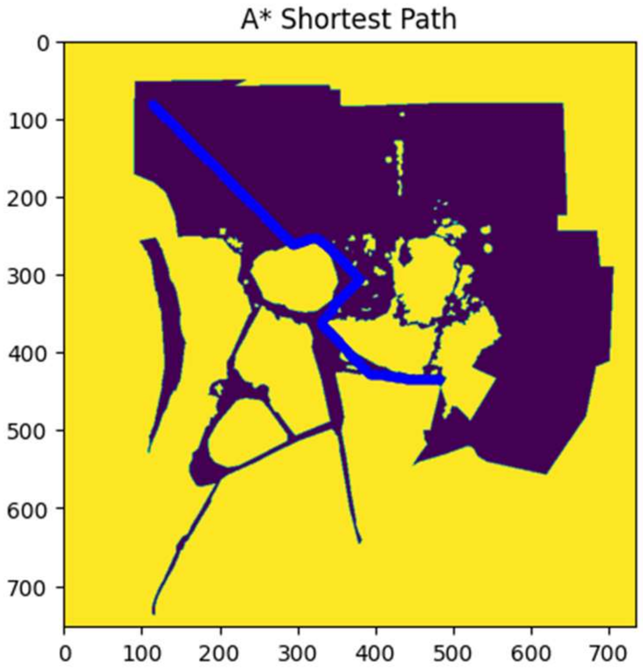
Resolution: 2*2 m

Start: (112,79) | Goal: (482, 435)



Dijkstra Shortest Path

Run Time: 0.500125 seconds

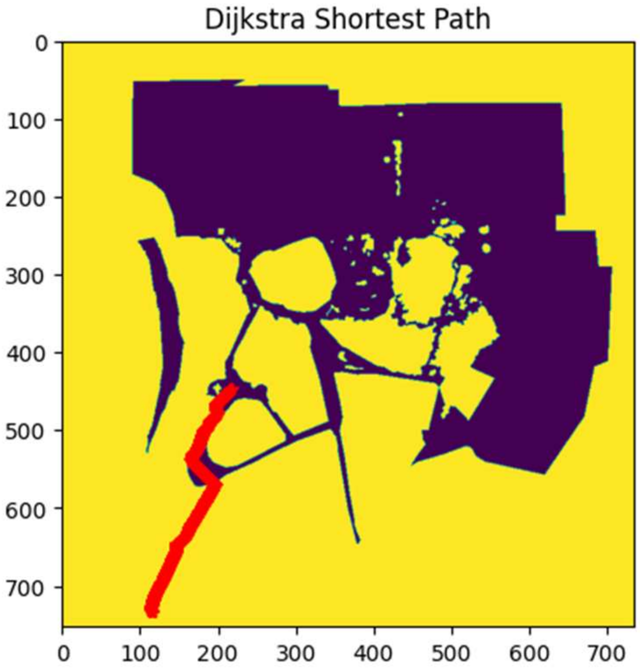


A* Shortest Path

Run Time: 0.461026 seconds

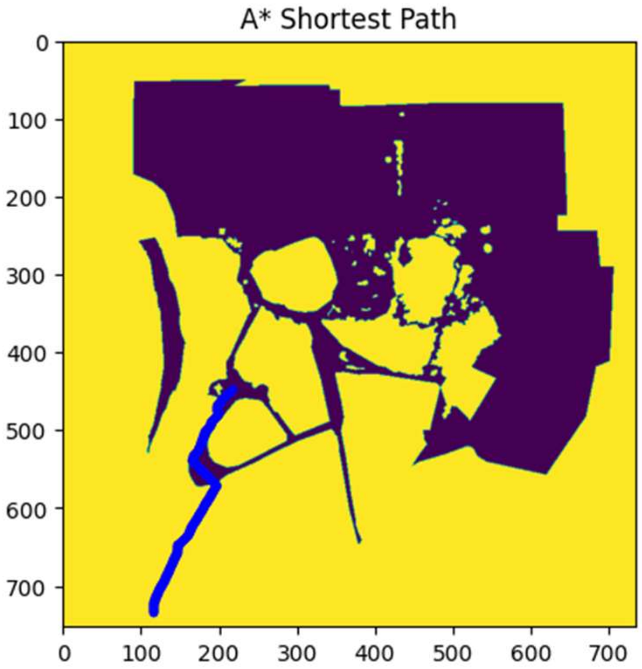
Resolution: 2*2 m

Start: (218,446) | Goal: (116, 734)



Dijkstra Shortest Path

Run Time: 0.221901 seconds

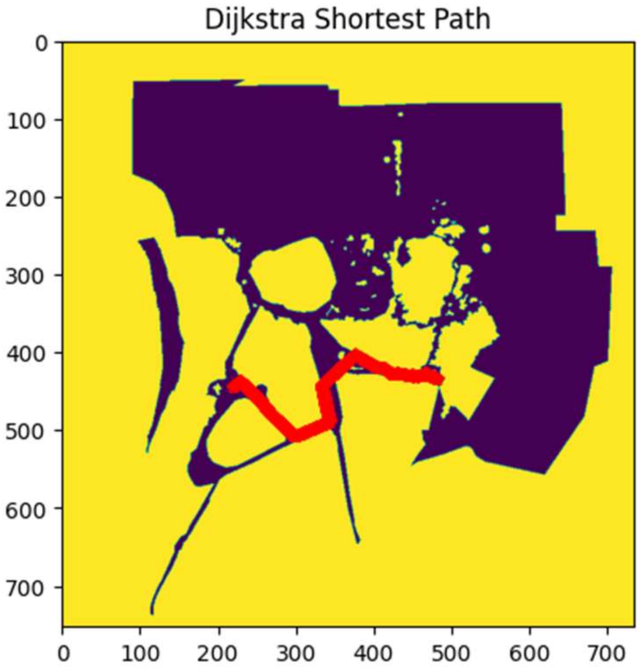


A* Shortest Path

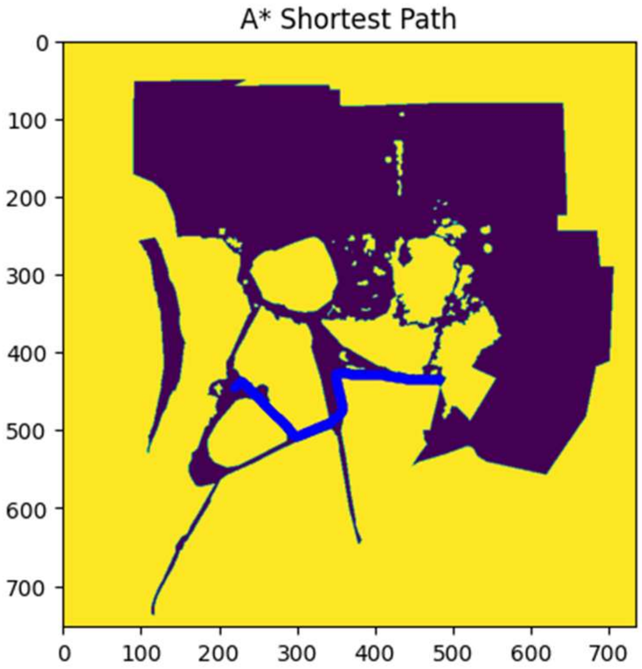
Run Time: 0.111033 seconds

Resolution: 2*2 m

Start: (218,446) | Goal: (482, 435)



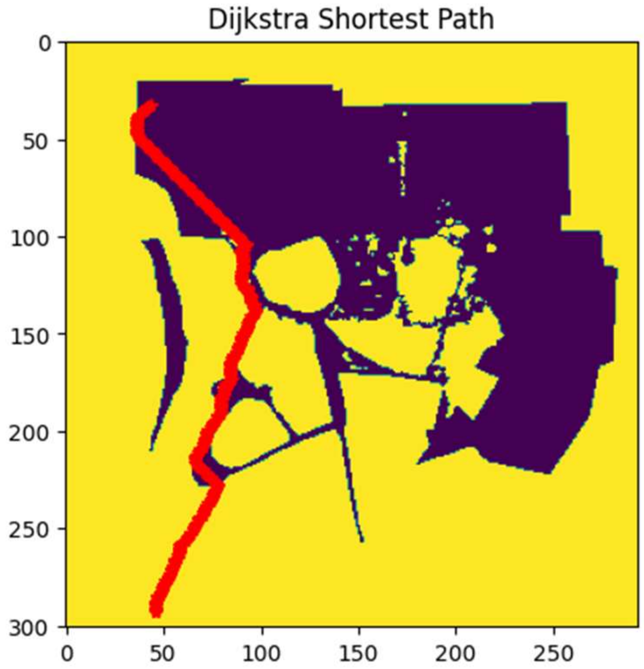
Run Time: 0.29264 seconds



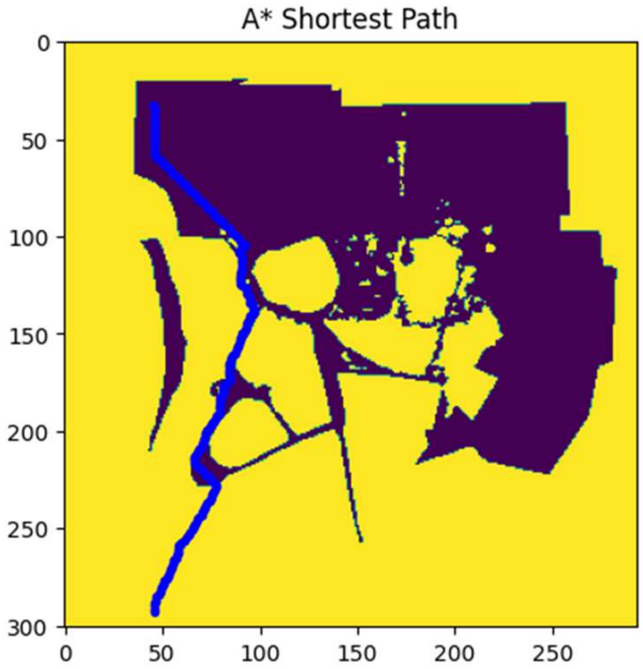
Run Time: 0.138638 seconds

Resolution: 5*5 m

Start: (44,31) | Goal: (46, 293)


Dijkstra Shortest Path

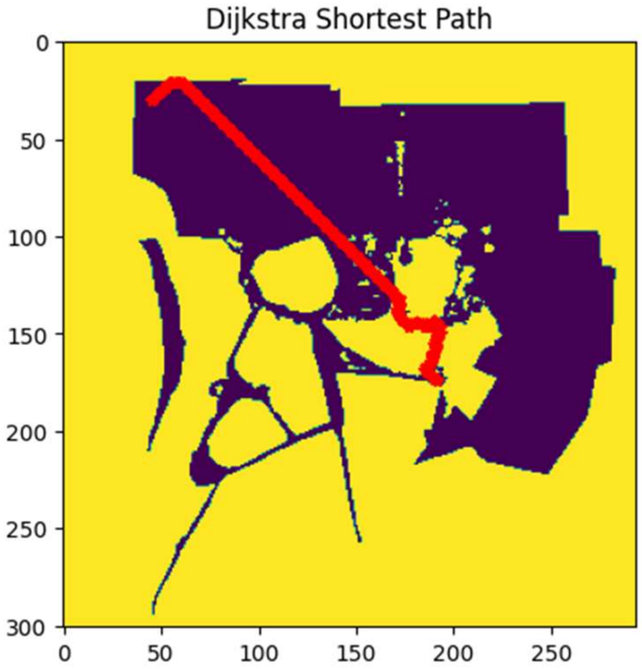
Run Time: 0.097616 seconds


A* Shortest Path

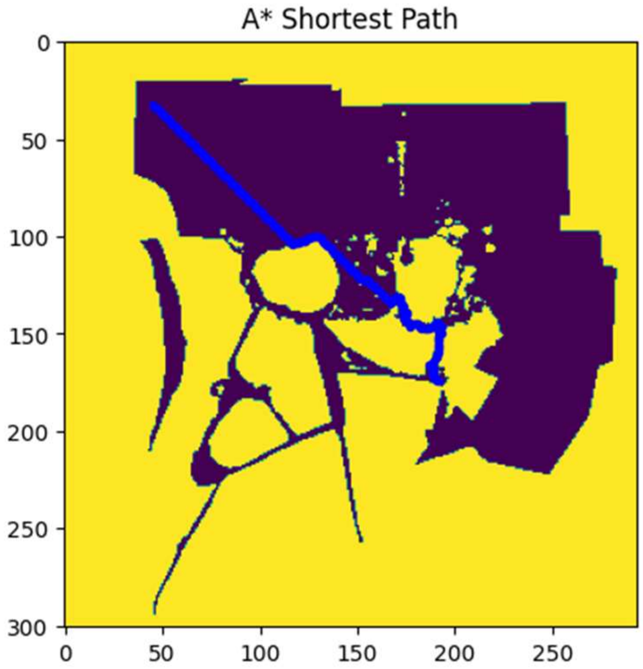
Run Time: 0.066313 seconds

Resolution: 5*5 m

Start: (44,31) | Goal: (192, 174)



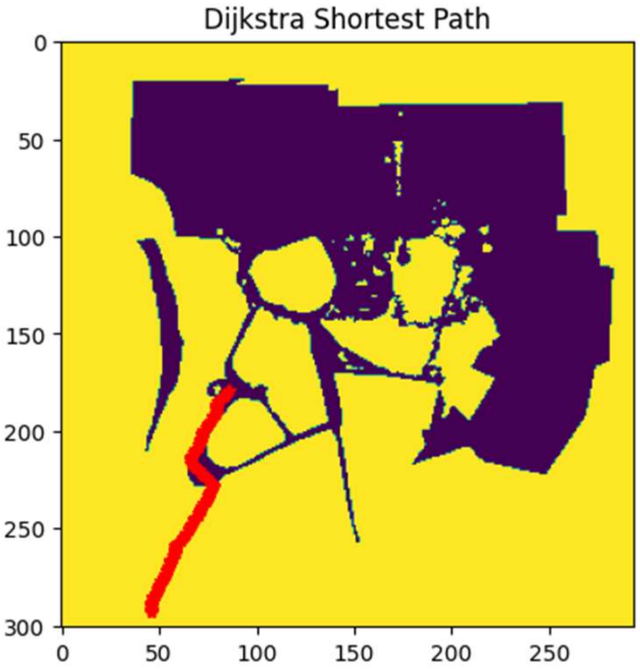
Run Time: 0.05889 seconds



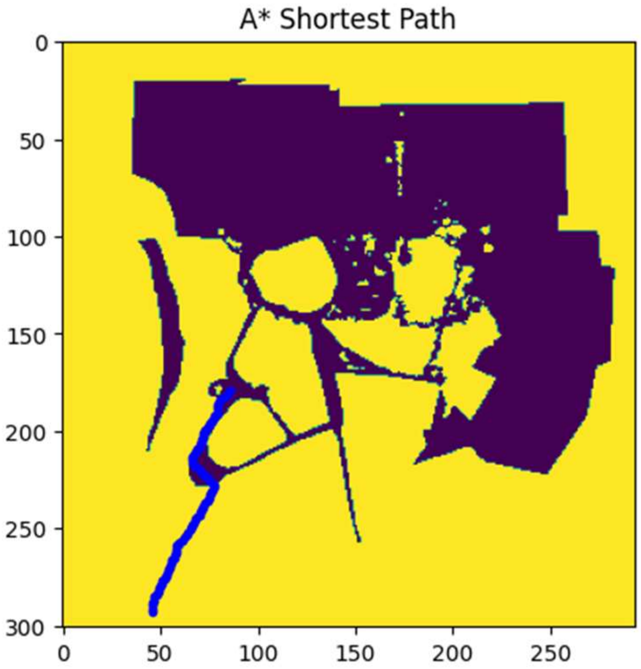
Run Time: 0.064222 seconds

Resolution: 5*5 m

Start: (87,178) | Goal: (46, 293)



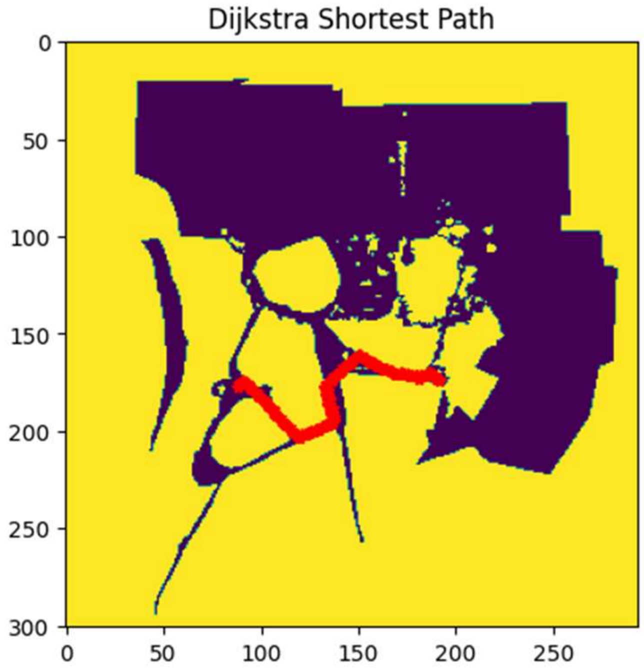
Run Time: 0.024431 seconds



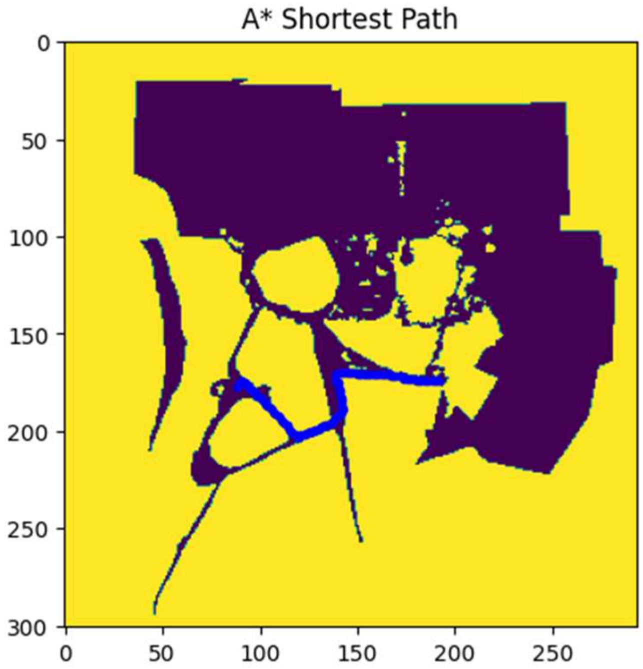
Run Time: 0.017098 seconds

Resolution: 5*5 m

Start: (87,178) | Goal: (192, 174)



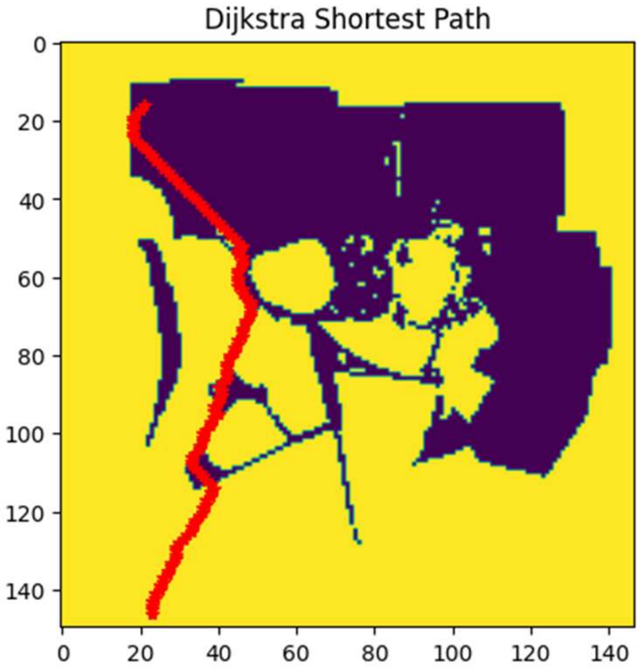
Run Time: 0.04637 seconds



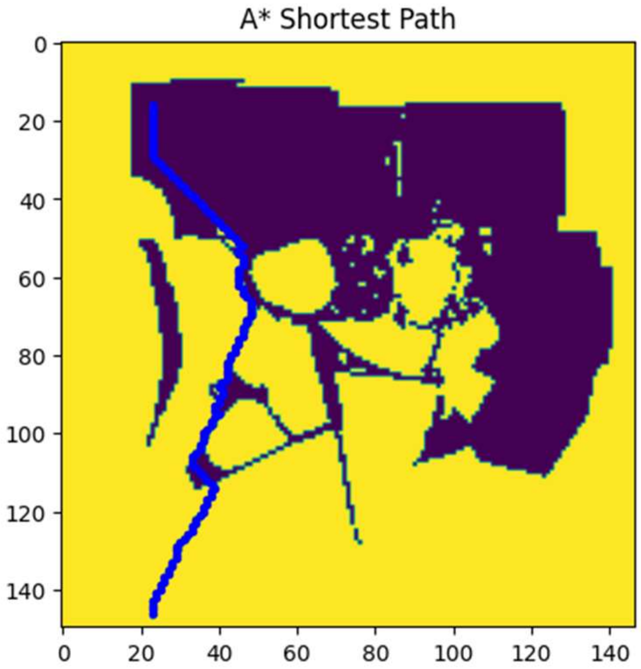
Run Time: 0.015282 seconds

Resolution: 10*10 m

Start: (22,15) | Goal: (23, 146)



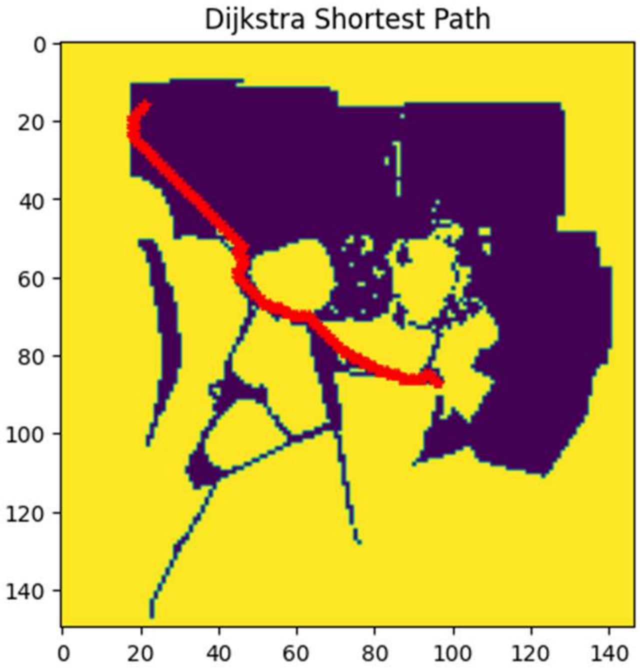
Run Time: 0.018954 seconds



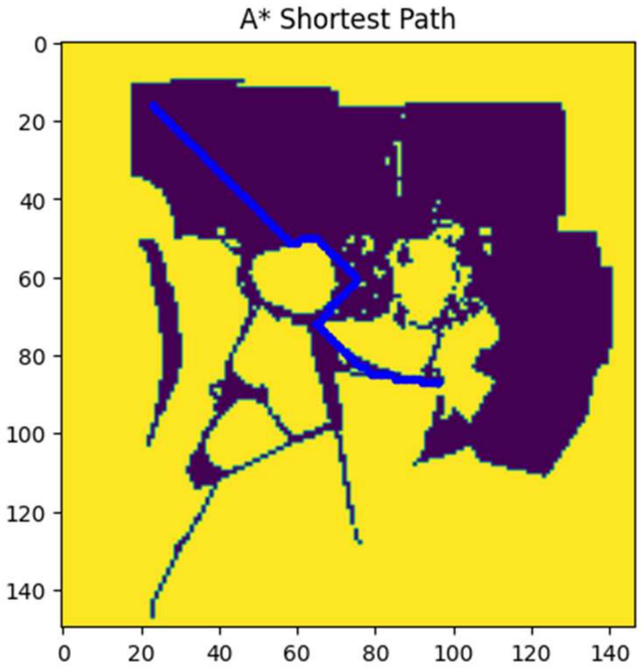
Run Time: 0.020712 seconds

Resolution: 10*10 m

Start: (22,15) | Goal: (96, 87)



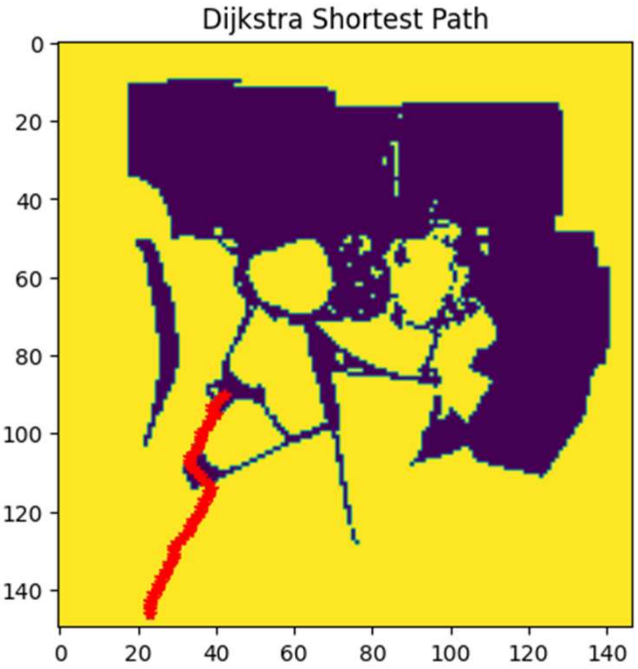
Run Time: 0.018822 seconds



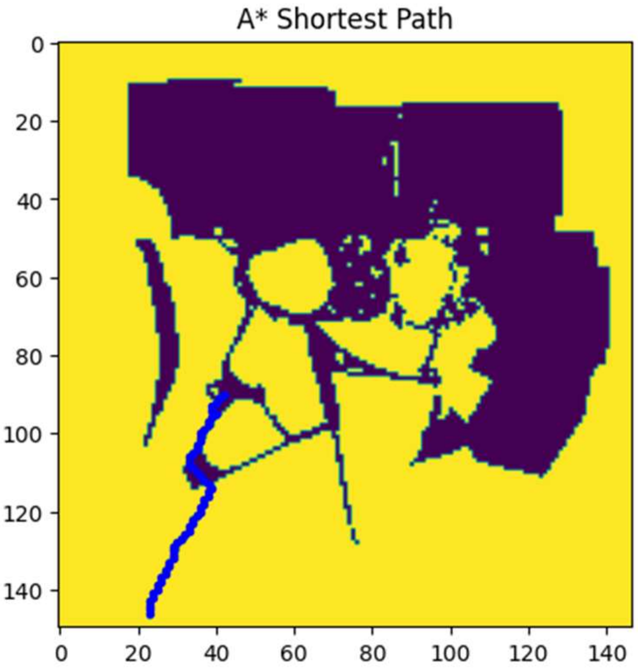
Run Time: 0.018465 seconds

Resolution: 10*10 m

Start: (43,89) | Goal: (23, 146)


Dijkstra Shortest Path

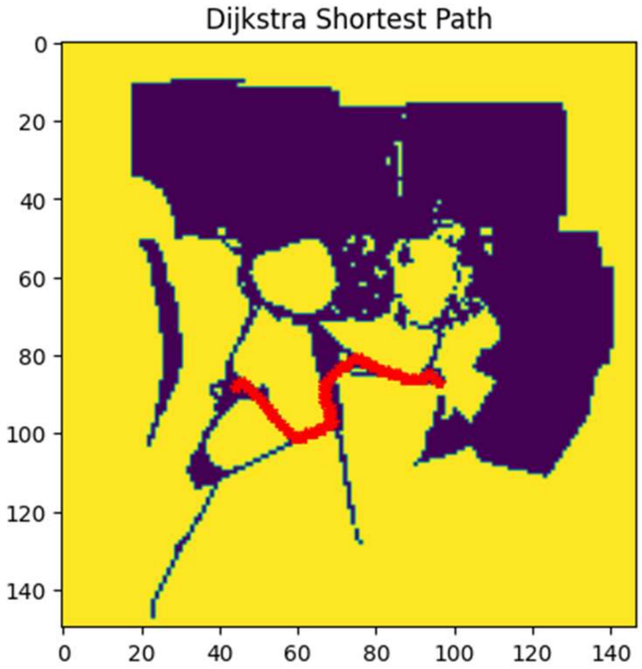
Run Time: 0.013324 seconds


A* Shortest Path

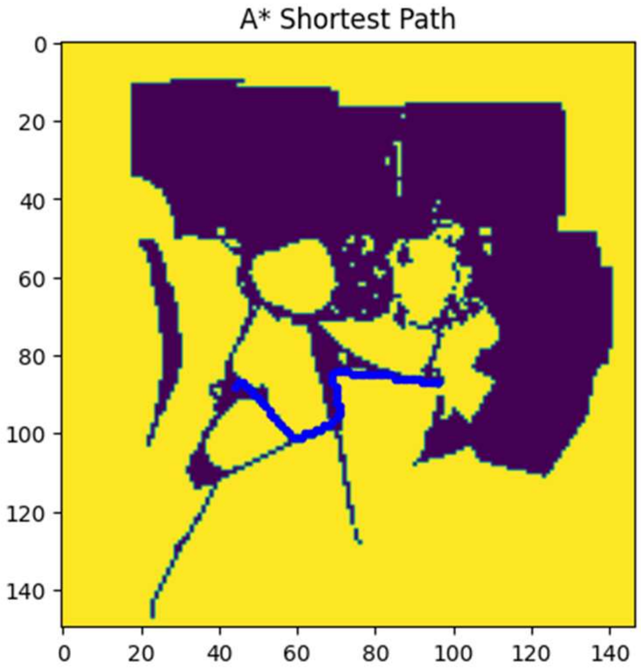
Run Time: 0.002873 seconds

Resolution: 10*10 m

Start: (43,89) | Goal: (96, 87)



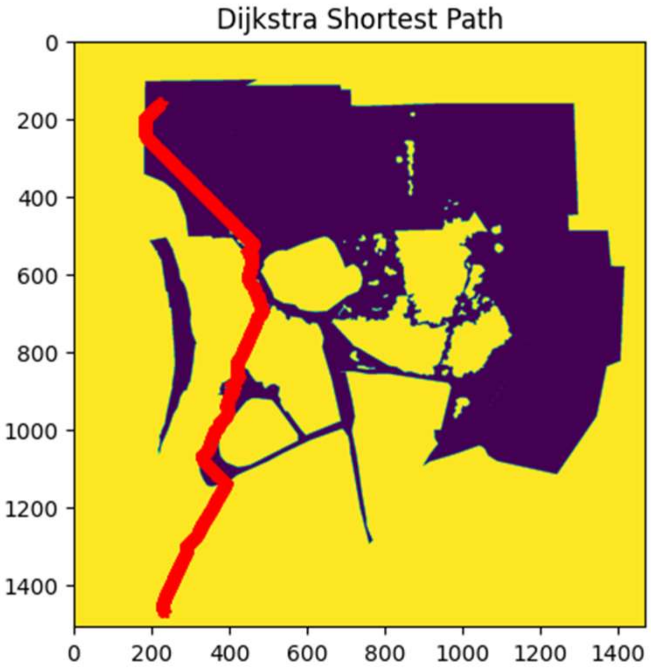
Run Time: 0.013365 seconds



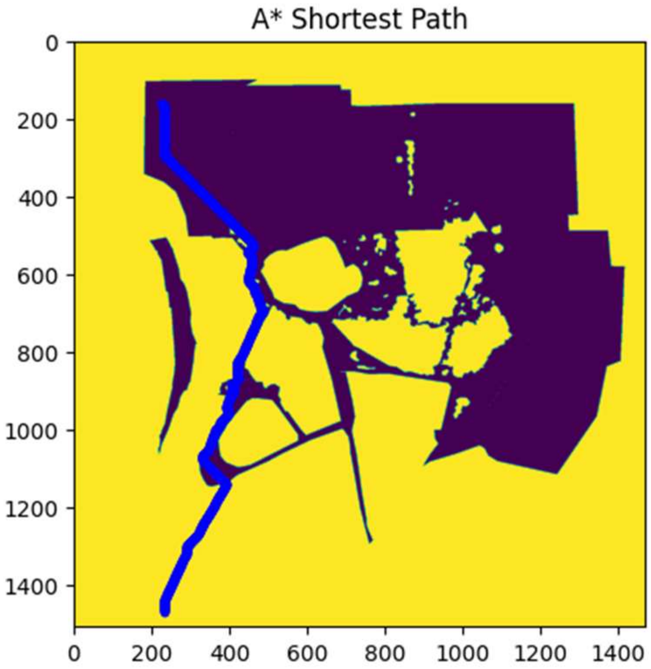
Run Time: 0.003586 seconds

Resolution: 1*1 m

Start: (224, 158) | Goal: (232, 1468)



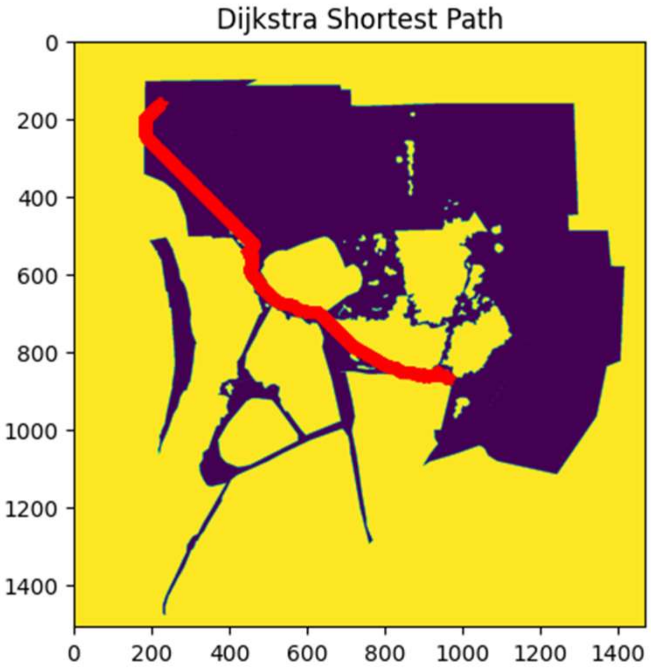
Run Time: 3.289191 seconds



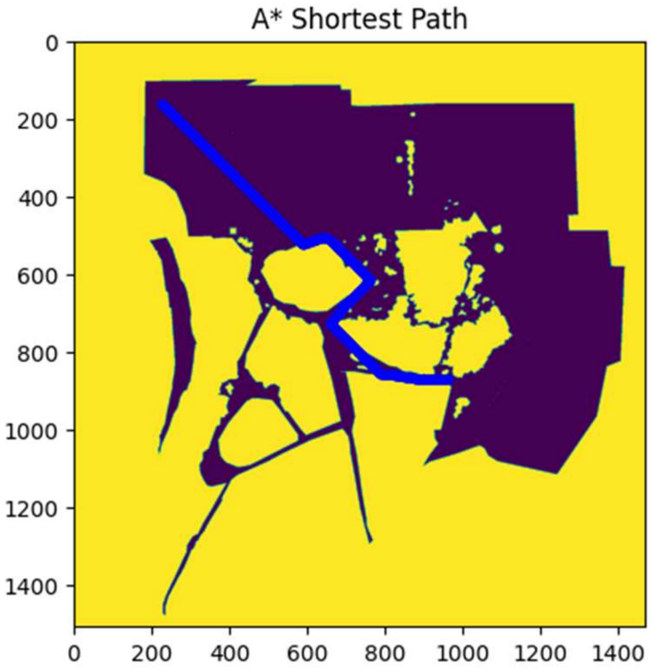
Run Time: 2.140017 seconds

Resolution: 1*1 m

Start: (224, 158) | Goal: (964, 870)



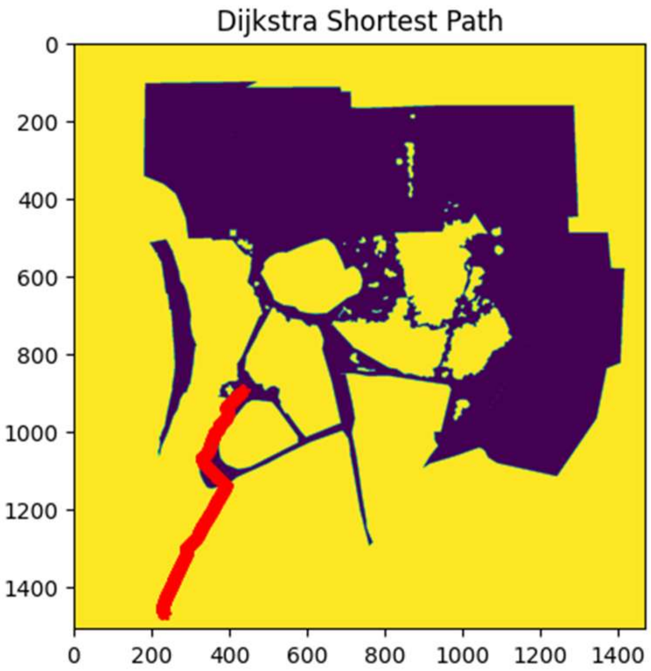
Run Time: 2.919911 seconds



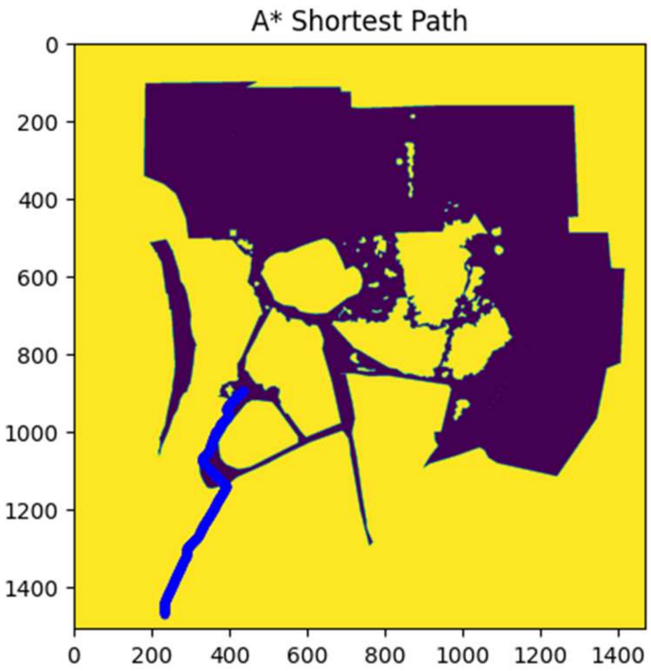
Run Time: 1.874934 seconds

Resolution: 1*1 m

Start: (436, 892) | Goal: (232, 1468)



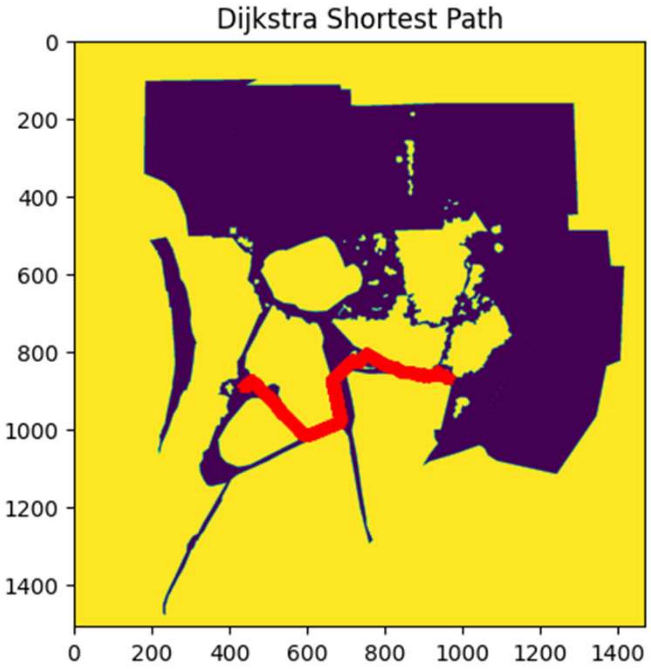
Run Time: 1.273725 seconds



Run Time: 0.557296 seconds

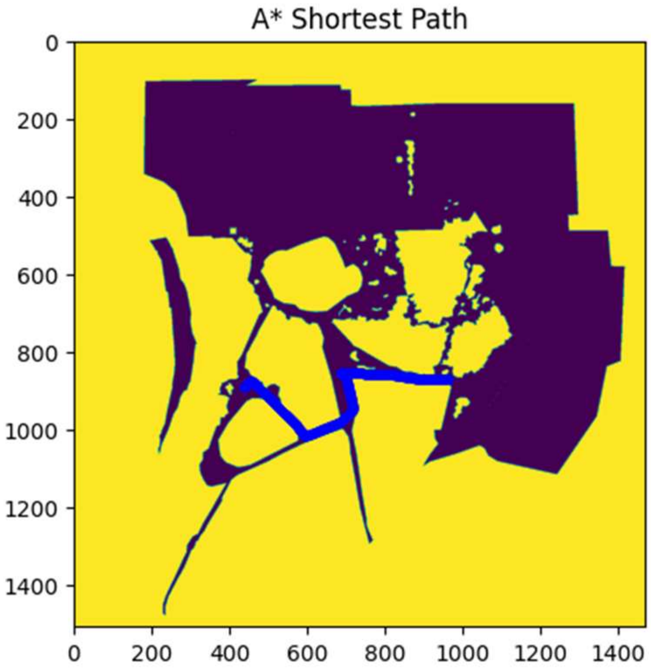
Resolution: 1*1 m

Start: (436, 892) | Goal: (964, 870)



Dijkstra Shortest Path

Run Time: 1.435158 seconds



A* Shortest Path

Run Time: 0.626574 seconds