IEEEtran Technical Reports Template Mälardalen University

Jacob Johansson*, Mudar Ibrahim†,
School of Innovation, Design and, Engineering
Mälardalens University, Västerås, Sweden
Email: *jjn20030@student.mdu.se, †mim20004@student.mdu.se,

A vs. Dijkstra's algorithm*

A* and Dijkstra's path planning algorithms are among many algorithms for finding a path to the goal node from a start node in a graph.

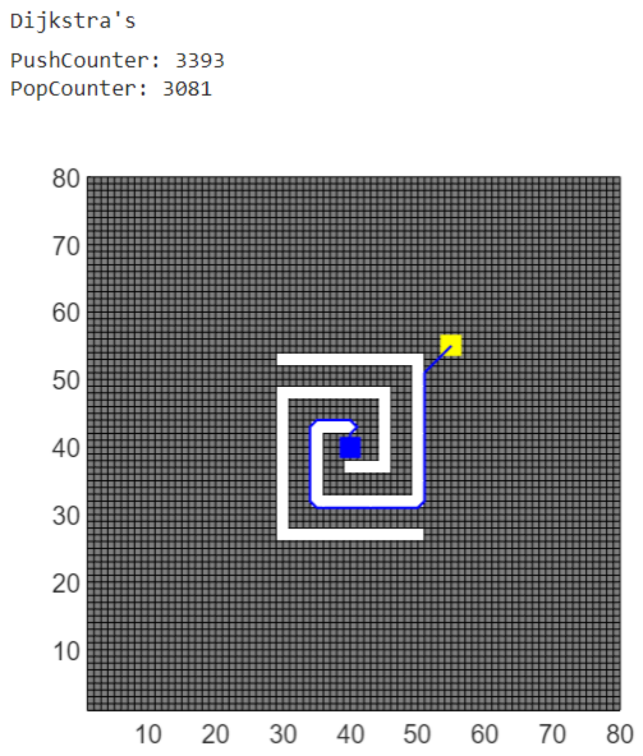
Dijkstra's: Dijkstra's algorihm find the shortest path to a goal in a graph by exploring all possible paths to the goal. It's more simple to implement than the A*, but due to exploring all possible paths the Dijkstra's consume more memory and is less efficient. Dijkstra's is most useful for applications that needs all nodes to be explored, such as a network routing.

A:* A* is an heuristic based algorithm that finds the shortest path to a goal by combining the actual distance from the start node and a heuristic estimation of the distance to the goal. It's more efficient than Dijsktra's due to the heuristic estimation reduces the number of nodes to explore. A* also guarantees finding the most optimal path. A* is useful in applications when optimization is vital and not all nodes needs to be explored. Such use case is navigation for robots.

Result

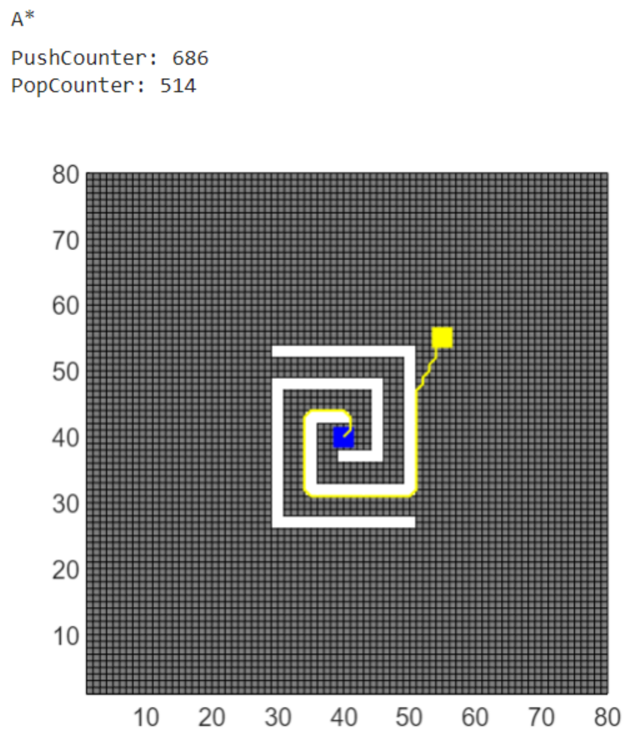
Static Environment: Figure 1(a) and 1(b) shows how well Djikstra's and A* algorithm perfom in a static, known, environment. It includes both the path taken, and the number of pushes and pops. It clearly demonstrates the efficiency of the A* algorithm compared to the Djikstra's algorithm in this use case of finding the goal node from a start node. The reason for this is the heuristic estimation A* has embedded in its algorithm.

Dynamic Environment: Figure 2(a) and 2(b) demonstrate the performance of Djikstra's and A* algorithm in a dynamic environment. It illustrates how A* is still more efficient than Djikstra's in a dynamic environment. In a dynamic environment compared to a static environment, the Djikstra's algorithm get exponentially worse than the A* algorithm.



```
Dijkstra's
PushCounter: 3393
PopCounter: 3081
```

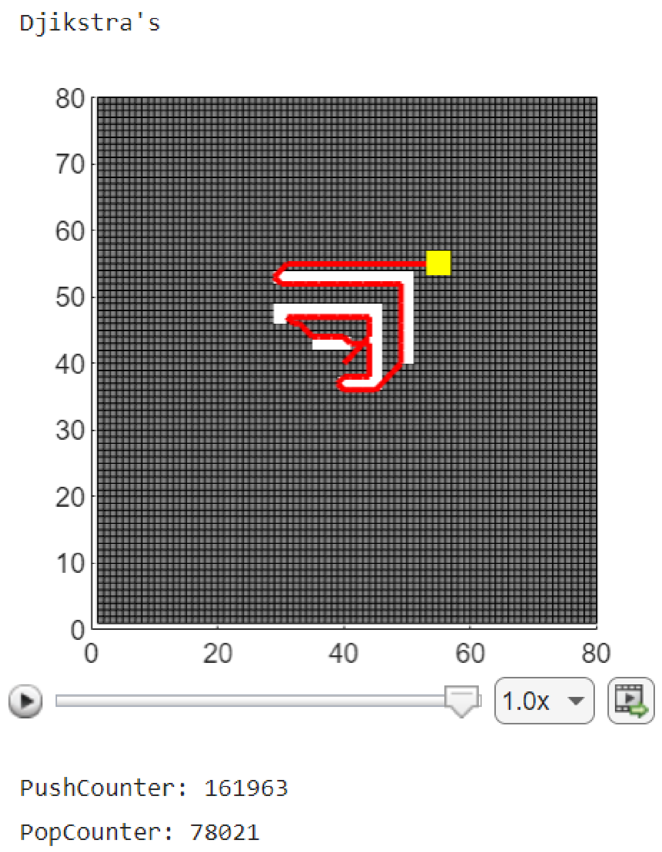
((a)) Dijkstra's algorithm in a static environment.



```
A*
PushCounter: 686
PopCounter: 514
```

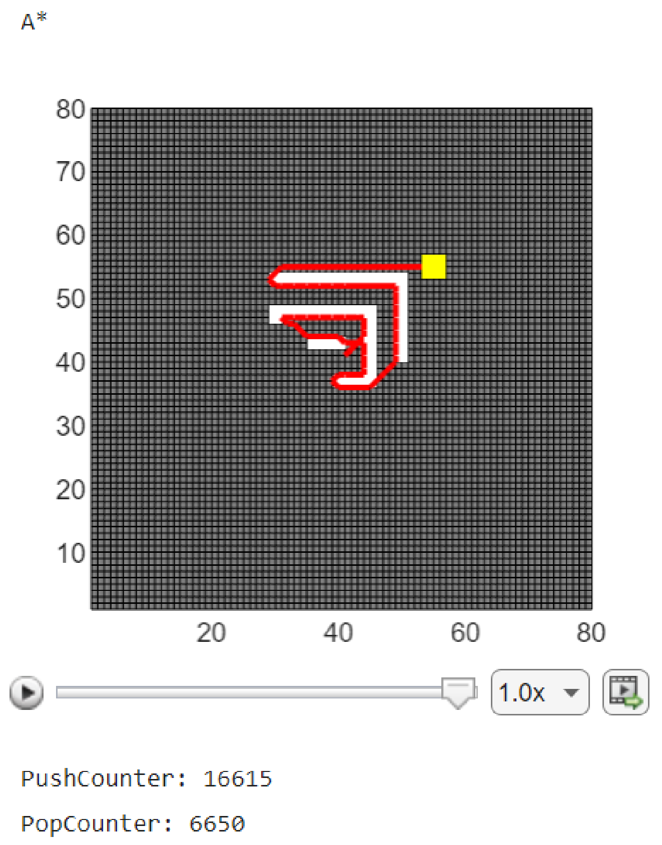
((b)) A* algorithm in a static environment.

Figure 1: Comparison of algorithms in static environments.



```
Djikstra's
PushCounter: 161963
PopCounter: 78021
```

((a)) Dijkstra's algorithm in a dynamic environment.



```
A*
PushCounter: 16615
PopCounter: 6650
```

((b)) A* algorithm in a dynamic environment.

Figure 2: Comparison of algorithms in dynamic environments.