Sumeet Shanbhag

Documentation

a) Explain in your own words, how does D* replan a path by updating the cost?

D* (Dynamic A*) is an incremental heuristic search algorithm that replans a path by updating the cost of traversing a graph when changes in the environment are detected. It is particularly useful for pathfinding in dynamic or uncertain environments, such as for mobile robots. When a change in the environment occurs, such as an obstacle appearing in the path or the cost of traversing an edge changing, D* efficiently updates the path by minimizing the recomputation required.

D* quickly adjusts the path by doing the following:

1. Detect change: It identifies the affected areas and updates the costs of moving through them.
2. Update costs: It updates the costs of the affected nodes by applying the new edge costs.
3. Use open and closed lists: D* keeps track of areas to explore (open list) and those already explored (closed list).
4. Spread cost updates: D* goes through the open list, updating costs and moving areas to the closed list. During this process, the neighbouring nodes costs may also need to be updated to maintain consistency.
5. Check affected areas: As D* propagates cost updates, it re-explores the areas affected by the change in the environment to find the new best path.
6. Termination: It stops when the open list is empty, meaning it has found the updated best path. At this point, D* has found an updated optimal path from the starting position to the goal, considering the new environment conditions.

b) Why does D* can replan faster than A* or Dijkstra?

D* can replan faster than A* or Dijkstra because it is an incremental search algorithm designed to adapt quickly to changes in the environment without having to restart the search from scratch. The main reasons why D* is faster at replanning than A* or Dijkstra are:

1. Efficient updates: When the environment changes, D* only updates the affected nodes and their associated edge costs. A* and Dijkstra would need to perform a full search again, which takes longer.
2. Reusing previous search information: D* leverages information from previous searches to focus on the areas affected by the change. This allows the algorithm to avoid reprocessing the entire search space, making it more efficient than A* or Dijkstra, which do not retain prior search information.
3. Incremental exploration: D* incrementally processes nodes in the open list (areas to explore) based on their key values (cost and heuristic), which helps it focus on the

most promising nodes. A* and Dijkstra, on the other hand, need to explore a larger number of nodes in their search.

4. Localized updates: D* concentrates on updating the costs and paths around the affected areas, reducing the computational effort. In contrast, A* and Dijkstra would need to consider the entire search space, which can be computationally expensive.

These factors contribute to D*'s faster replanning capability compared to A* or Dijkstra. In dynamic or uncertain environments, such as for mobile robots, this speed advantage allows D* to quickly adapt and find optimal paths even as conditions change, making it a more suitable choice for these situations.

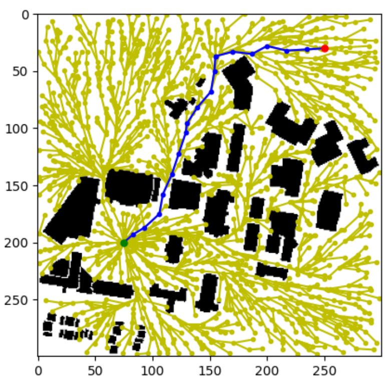
c) What is the key differences between regular RRT* and informed RRT*?

RRT* (Rapidly-exploring Random Trees Star) and Informed RRT* are both sampling-based motion planning algorithms that aim to find an optimal path from a start to a goal point in a given environment. While RRT* is an asymptotically optimal version of RRT, Informed RRT* improves upon RRT* by using additional information to guide the search process. Here are the key differences between RRT* and Informed RRT*:

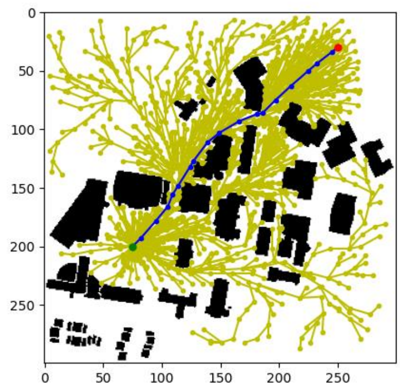
1. Heuristic information: Informed RRT* incorporates a heuristic to guide the search process, making it more informed about the goal's location. This heuristic helps reduce the search space by focusing on more promising areas of the environment. Regular RRT* does not use any heuristic and relies solely on random sampling to explore the environment.
2. Ellipsoidal sampling region: Informed RRT* introduces an ellipsoidal sampling region around the current best path, which further limits the search space. This region adapts as better paths are found, resulting in faster convergence to an optimal path. RRT* samples uniformly from the entire search space without any preference.
3. Improved convergence: Informed RRT* converges faster to a near-optimal solution compared to RRT* due to its heuristic guidance and restricted sampling region. By focusing on the more promising areas of the environment, Informed RRT* reduces the number of iterations needed to find a good path.
4. Computational efficiency: Informed RRT* is more computationally efficient than RRT* for finding near-optimal paths, as it reduces the search space and leverages heuristic information to guide its exploration.

d) By showing and comparing the results of RRT* and informed RRT*, what is the advantages of using the latter?

RRT*



Informed RRT*



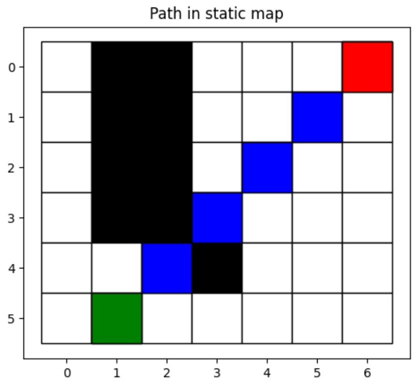

```
It took 1434 nodes to find the current path
The path length is 287.25
```

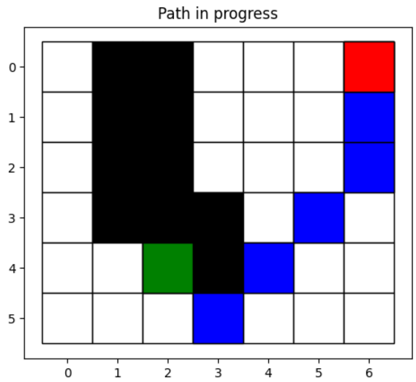
```
It took 1536 nodes to find the current path
The path length is 249.18
```

By comparing the results of RRT* and Informed RRT*, we can observe following advantages of using Informed RRT*:

1. Faster convergence to near-optimal solutions: Informed RRT* leverages heuristic information and an ellipsoidal sampling region, which guides the search towards more promising areas in the environment. As a result, Informed RRT* converges to a near-optimal path faster than RRT*.
2. Reduced search space: The ellipsoidal sampling region in Informed RRT* restricts the sampling to a smaller area around the current best path, leading to a more focused search. This reduces the overall search space, which can result in a more computationally efficient algorithm compared to RRT*.
3. Better path quality: Informed RRT* tends to produce higher-quality paths due to its goal-directed search, which is more likely to find shorter and smoother paths compared to the random exploration in RRT*. This is particularly beneficial in complex environments with many obstacles.
4. Improved efficiency: Informed RRT* can achieve better solutions in fewer iterations, leading to a more efficient algorithm. This is important in real-time or time-constrained applications, such as autonomous vehicle navigation, where quick path planning is essential.

D* Outputs


Path in static map
0
1
2
3
4
5
0
1
2
3
4
5
6


Path in progress
0
1
2
3
4
5
0
1
2
3
4
5
6