Assignment 1 , 159.740, 2020 S2

Family Name: JIANG , Given Name: TAO ,

Student ID:20004769

**ssRobot Path-Planning Algorithms for Fully Dynamic Shortest Path**

1. **User’s Guide**

Run the program: *main .\grids\grid\_Dstar\_journal.map m*

Controls:

F7: LpaStar Initial search

F6: LpaStar second search-replannning

F8:Dstar lite Initial search

F3:Dstar lite second search – replanning

F5:show details

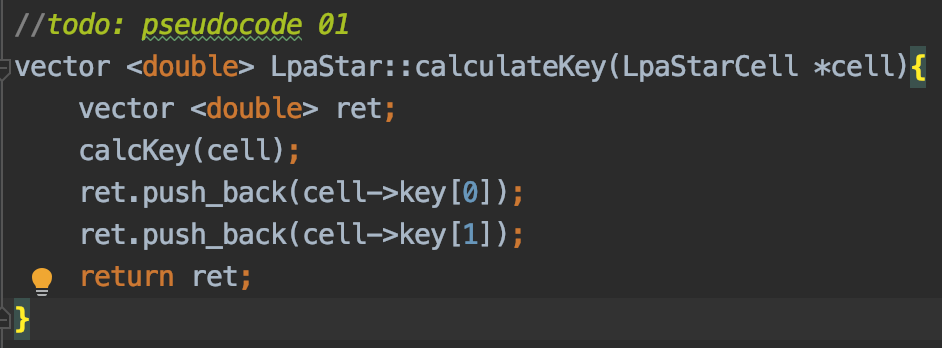
1. **Specify the data structures of algorithms**

LpaStar and Dstar lite use preiority\_queue(U) to store all nodes. Each node has attributes include g, rhs, h, key1,key2.

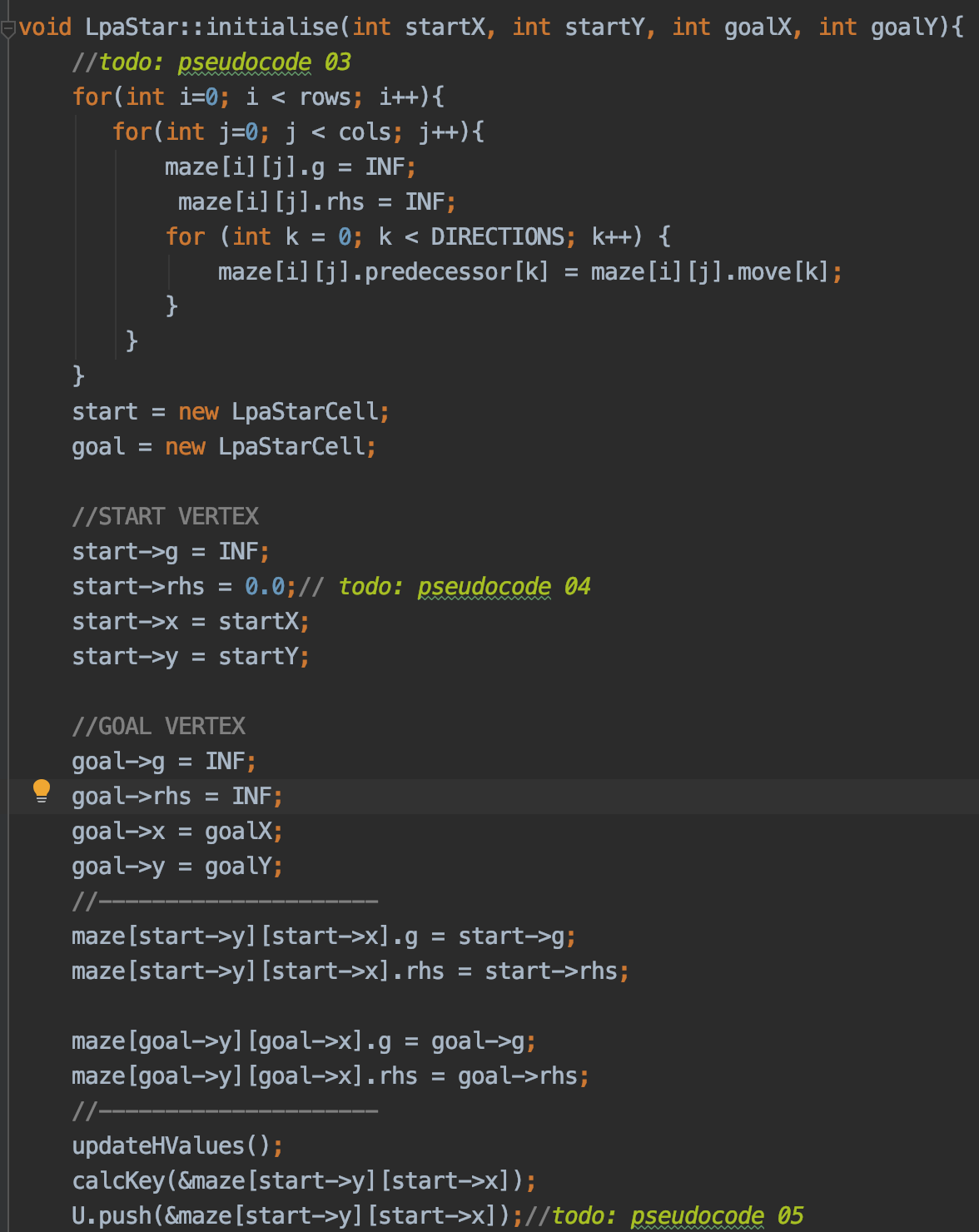
1. **skeleton C++ code for algorithms**

*3.1 LpaStar pseudocode corresponding line numbers:*

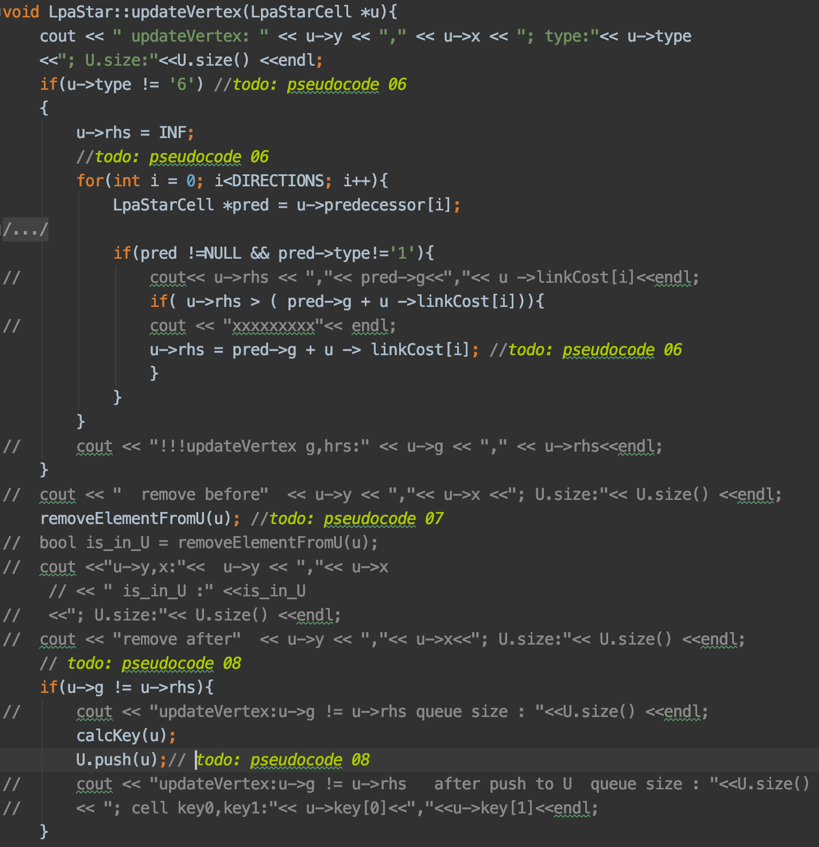
Procedure CalculateKeys():



Procedure Initialize():



Procedure UpdateVertex(U):

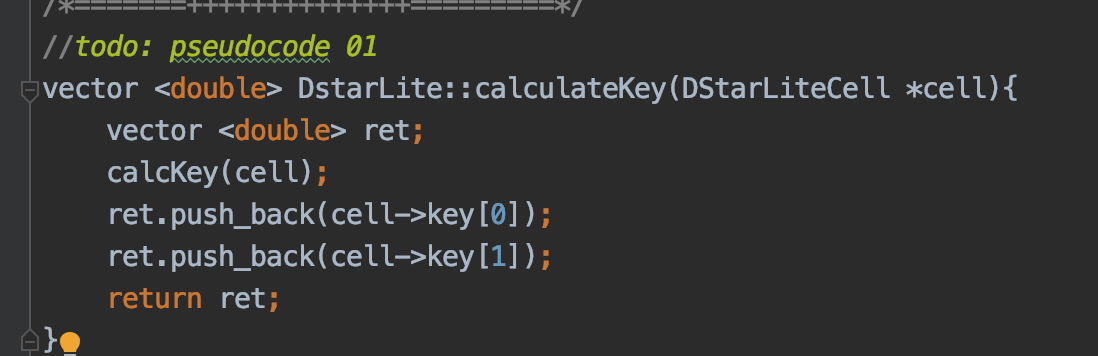


Procedure ComputeShortestPath():



*3.2 D\*Lite pseudocode corresponding line numbers:*

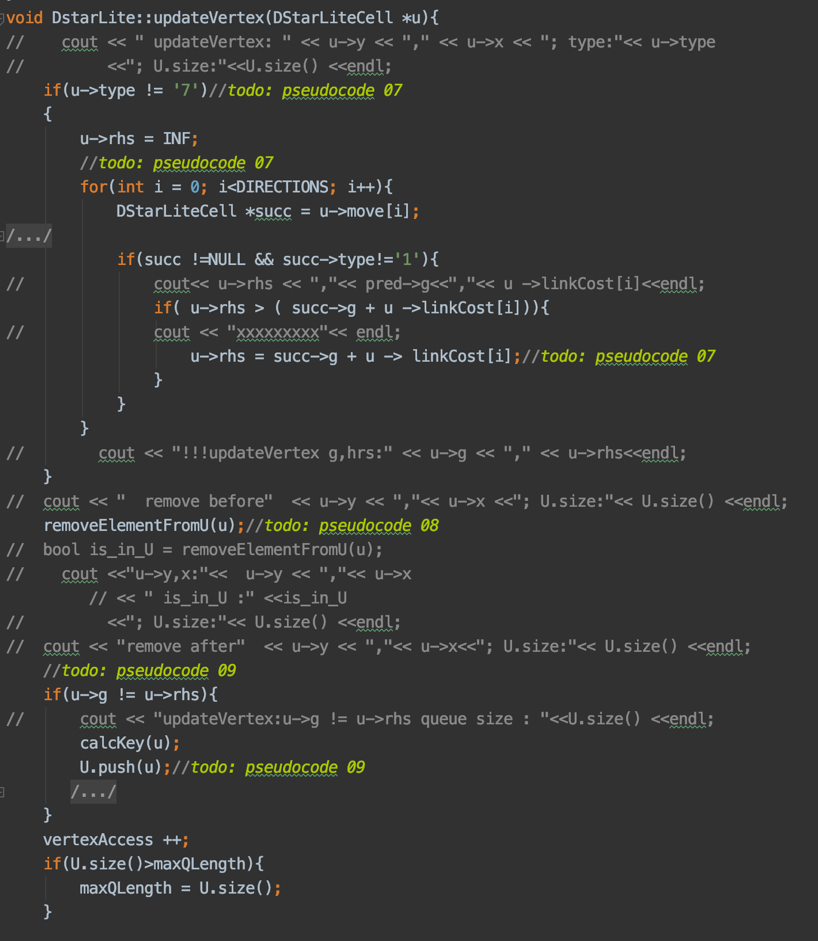
Procedure CalculateKeys():



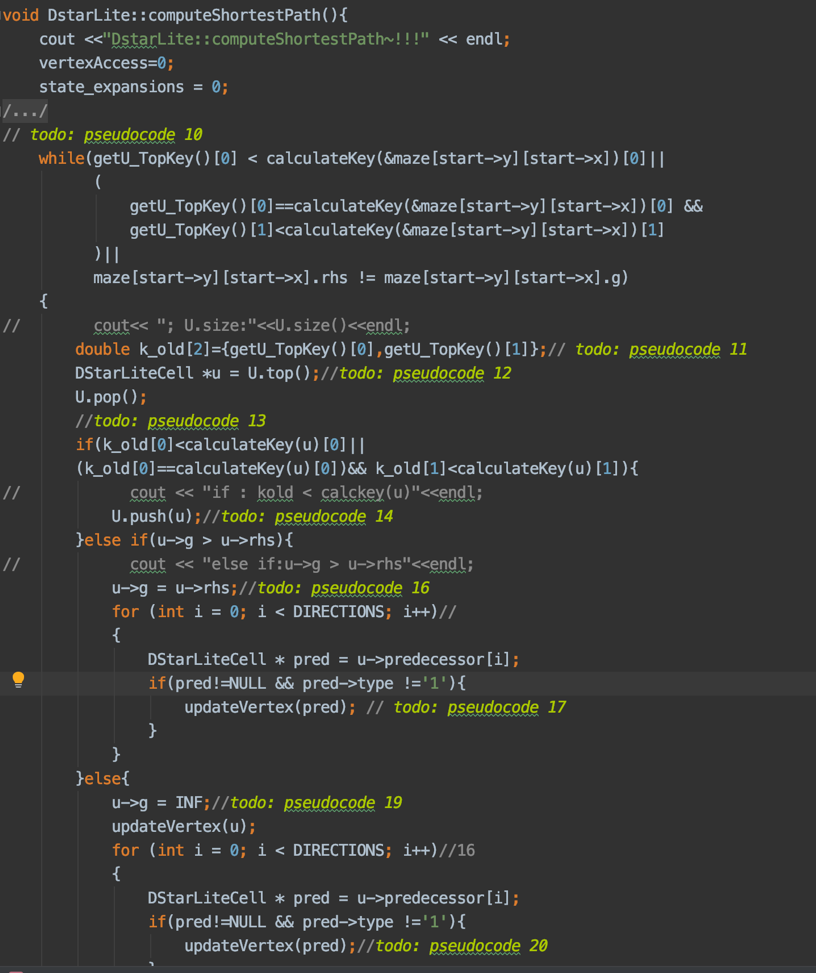
Procedure Initialize():



Procedure UpdateVertex(U):



Procedure ComputeShortestPath():



1. **D\*Lite and LPA\* journals snapshot**

Screen attached

1. **experiment analysis**

LPA \* is repeatedly planning the shortest path between the initial grid point and the target point, and the starting point is fixed at the beginning. Therefore, when the mobile robot moves, the path planned after the environmental information changes is not optimal for the mobile robot at the current moment.

D\* Lite has made improvements on this basis. It treats the current location point as a new Start grid point and repeatedly calculates the shortest path between the Goal point and the new Start point. The reverse search allows D\*lite to use the node distance information generated in the previous iteration in the dynamic obstacle graph to continuously update the closest distance from the current node to the target.

1. **experiment results**

Gridworld: *grid\_lpa\_journal.map*

Heuristic: Manhattan-8 distance

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **Algorithm** | **Max. Queue length** | **Path length** | **No. of state expansions** | **Vertex accesses** | **Actual Running Time (msec.)** |
| 1 | Lifelong Planning A\* | Initial:6, second:7 | Initial:6, second :7 | Initial:10, second :7 | Initial: 27, second :22 | Initial :94ms, second :57ms |
| 2 | D\* Lite | Initial :4, second :5 | Initial: 6, second :7 | Initial :0,  second :0 | Initial: 44, second :0 | Initial :17ms,  second :88 ms |

Gridworld: *grid\_lpa\_journal.map*

Heuristic: Euclidean distance

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **Algorithm** | **Max. Queue length** | **Path length** | **No. of state expansions** | **Vertex accesses** | **Actual Running Time (msec.)** |
| 1 | Lifelong Planning A\* | Initial :6,  second: 7 | Initial :6, second :7 | Initial :10, second :7 | Initial :27, second :22 | Initial : 74ms ,  second : 57ms |
| 2 | D\* Lite | Initial :4, second :5 | Initial :6, second :7 | Initial :0,  second :0 | Initial 44,  second :0 | Initial :25 ms,  second :128ms |

Gridworld: *grid\_Dstar\_journal.map*

Heuristic: Manhattan-8 distance

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **Algorithm** | **Max. Queue length** | **Path length** | **No. of state expansions** | **Vertex accesses** | **Actual Running Time (msec.)** |
| 1 | Lifelong Planning A\* | Initial:8, second:8 | Initial:3, second :4 | Initial:4, second :4 | Initial: 16, second :12 | Initial :55 ms, second :201 ms |
| 2 | D\* Lite | Initial :4, second :5 | Initial: 3, second :4 | Initial :0,  second :0 | Initial: 38, second :0 | Initial :11ms,  second :66ms |

Gridworld: *grid\_lpa\_journal.map*

Heuristic: Euclidean distance

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **Algorithm** | **Max. Queue length** | **Path length** | **No. of state expansions** | **Vertex accesses** | **Actual Running Time (msec.)** |
| 1 | Lifelong Planning A\* | Initial :8,  second: 8 | Initial :3, second :4 | Initial :4, second :4 | Initial :16, second :12 | Initial : 44 ms ,  second : 82 ms |
| 2 | D\* Lite | Initial :8, second :8 | Initial :3, second :4 | Initial :0,  second :0 | Initial 38,  second :0 | Initial :14 ms,  second :57 ms |

Gridworld: *grid\_lpa\_journal\_big.map*

Heuristic: Manhattan-8 distance

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **Algorithm** | **Max. Queue length** | **Path length** | **No. of state expansions** | **Vertex accesses** | **Actual Running Time (msec.)** |
| 1 | Lifelong Planning A\* | Initial:11, second:28 | Initial:13, second :14 | Initial:25, second :37 | Initial: 100, second :196 | Initial :192 ms, second :457 ms |
| 2 | D\* Lite | Initial :22, second :22 | Initial: 13, second :14 | Initial :0,  second :0 | Initial: 605, second :0 | Initial :19 ms,  second :315ms |

Gridworld: *grid\_lpa\_journal.map*

Heuristic: Euclidean distance

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **Algorithm** | **Max. Queue length** | **Path length** | **No. of state expansions** | **Vertex accesses** | **Actual Running Time (msec.)** |
| 1 | Lifelong Planning A\* | Initial :12,  second: 23 | Initial :13 second :14 | Initial :20, second :0 | Initial :73, second :170 | Initial : 169ms ,  second : 238 ms |
| 2 | D\* Lite | Initial :18, second :27 | Initial :13, second :14 | Initial :0,  second :0 | Initial 591,  second :0 | Initial :12 ms,  second :2 ms |