ELTE | FACULTY OF INFORMATICS

Path Planning Algorithm: Improved Rapidly-exploring Random Tree

Aphilak Lonklang

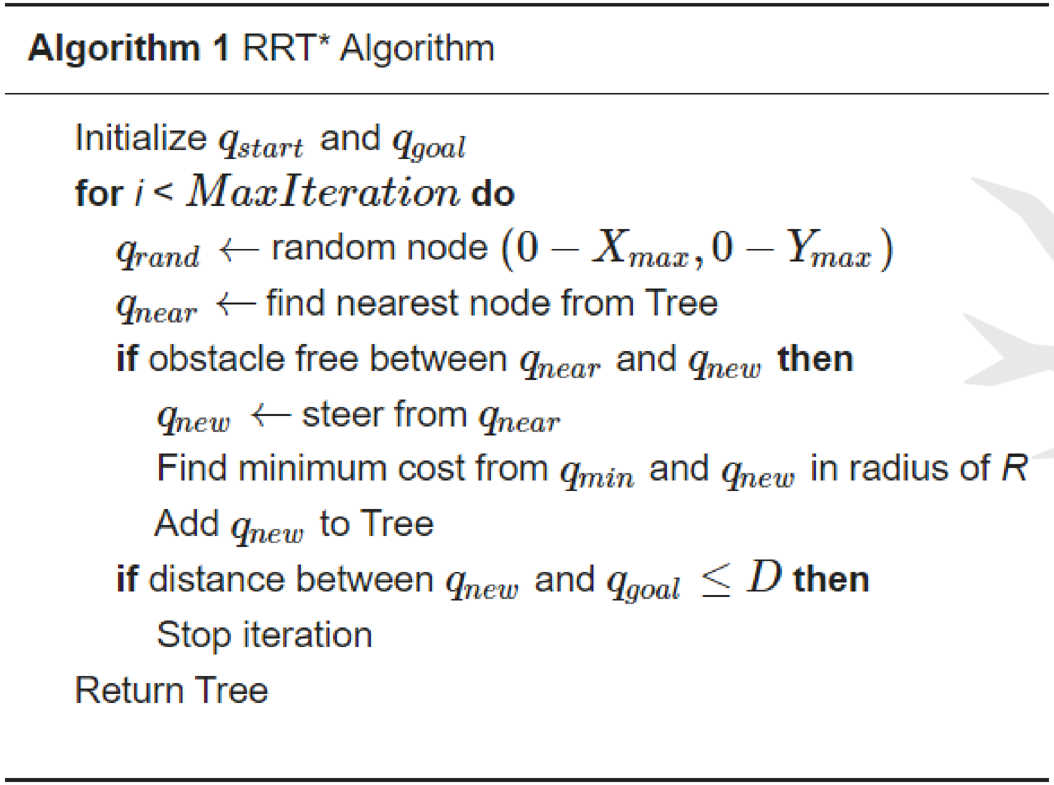
PhD Student
Department of Artificial Intelligence

Improved Rapidly Exploring Random Tree with Bacterial Mutation and Node Deletion for Offline Path Planning of Mobile Robot

Lonklang, A.; Botzheim, J. Improved Rapidly Exploring Random Tree with Bacterial Mutation and Node Deletion for Offline Path Planning of Mobile Robot. Electronics 2022, 11, 1459. https://doi.org/10.3390/electronics11091459

Rapidly Exploring Random Tree*

- RRT* is a random sampling tree structure search algorithm.



Algorithm 1 RRT* Algorithm

```
Initialize $q_{start}$ and $q_{goal}$
for $i < MaxIteration$ do
    $q_{rand} \leftarrow$ random node $(0 - X_{max}, 0 - Y_{max})$
    $q_{near} \leftarrow$ find nearest node from Tree
    if obstacle free between $q_{near}$ and $q_{new}$ then
        $q_{new} \leftarrow$ steer from $q_{near}$
        Find minimum cost from $q_{min}$ and $q_{new}$ in radius of $R$
        Add $q_{new}$ to Tree
    if distance between $q_{new}$ and $q_{goal} \leq D$ then
        Stop iteration
Return Tree
```

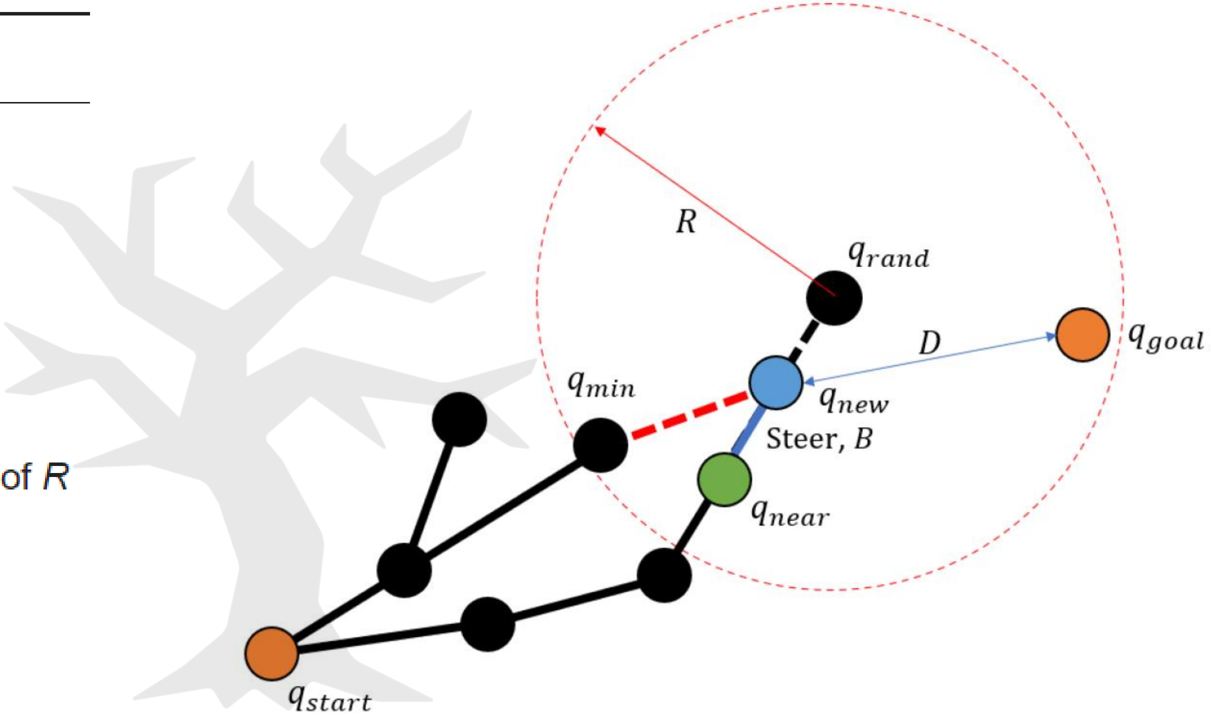
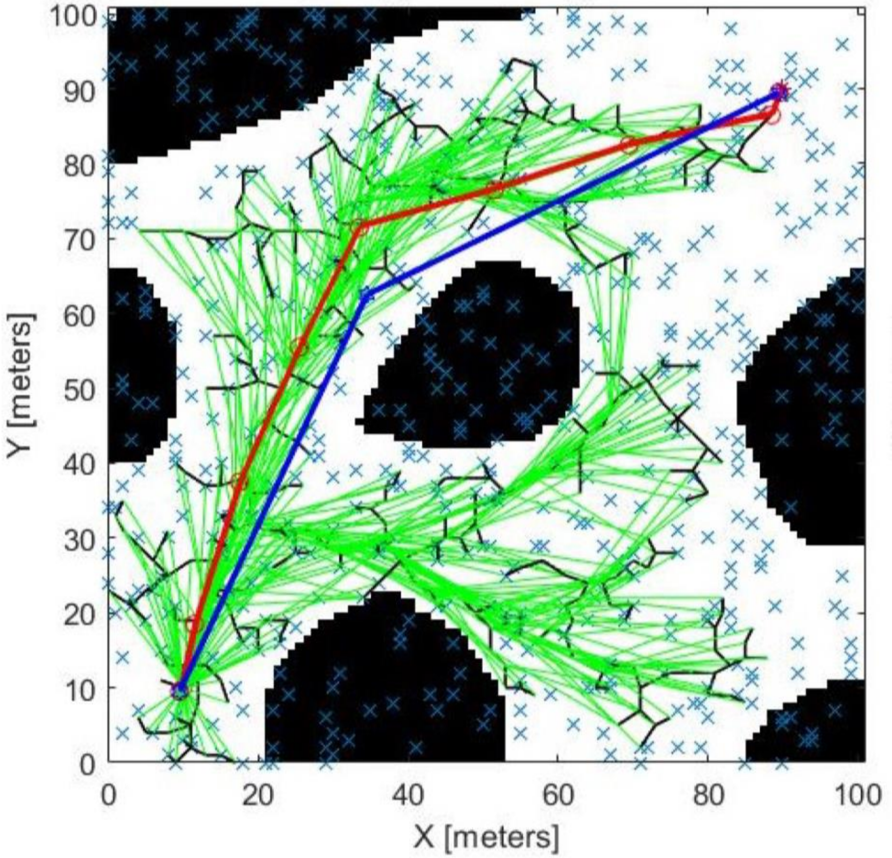


Illustration of RRT* Algorithm

Unusable Node

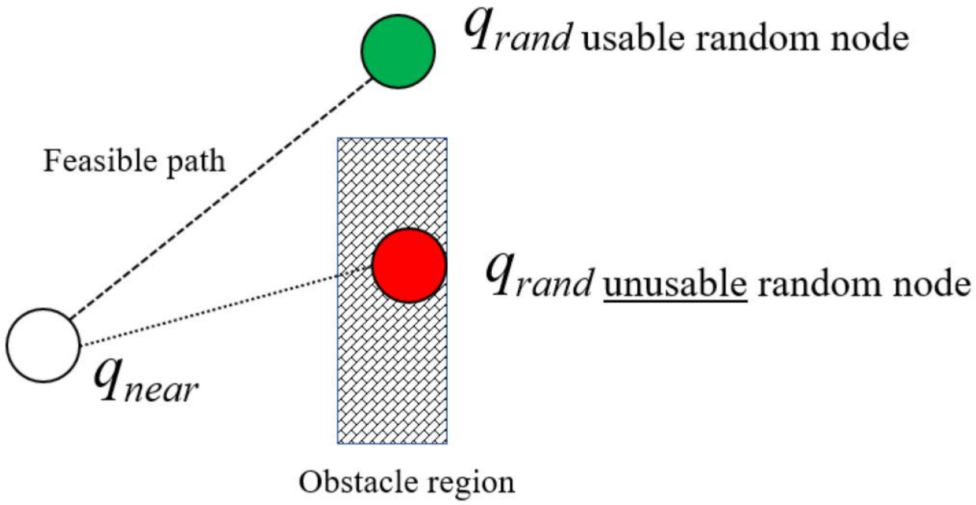
- Unusable nodes are the random nodes that fall into the obstacle regions.
- These nodes are not useful for iteration



Traditional RRT* (green)

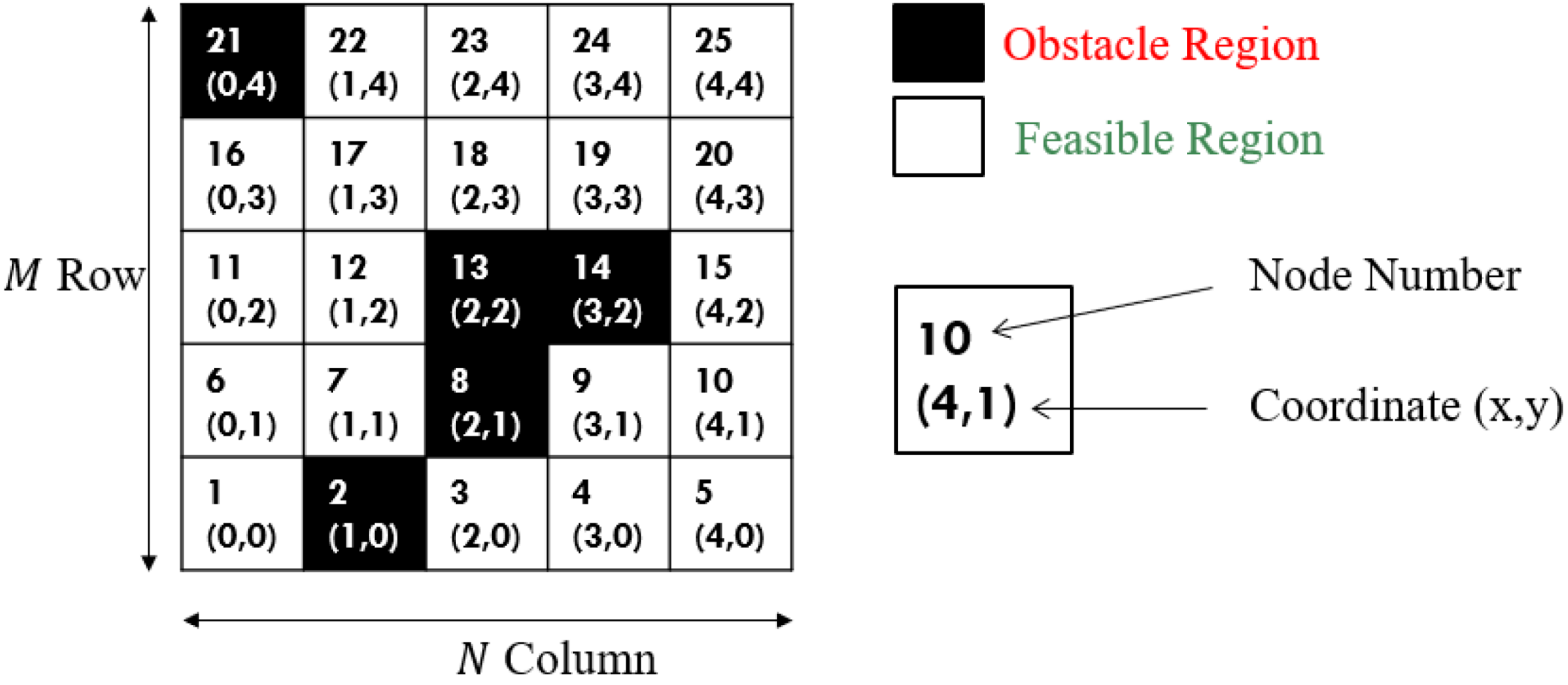
Unusable Nodes

- The number of generated unusable nodes is over 40% of RRT*.
- The main idea to improve the efficiency of the RRT* algorithm is to reduce the unusable nodes from the iteration and let the iteration flow continue with only usable nodes.


q_{rand} usable random node
Feasible path
q_{rand} unusable random node
q_{near}
Obstacle region

Unusable Node

Global Environment

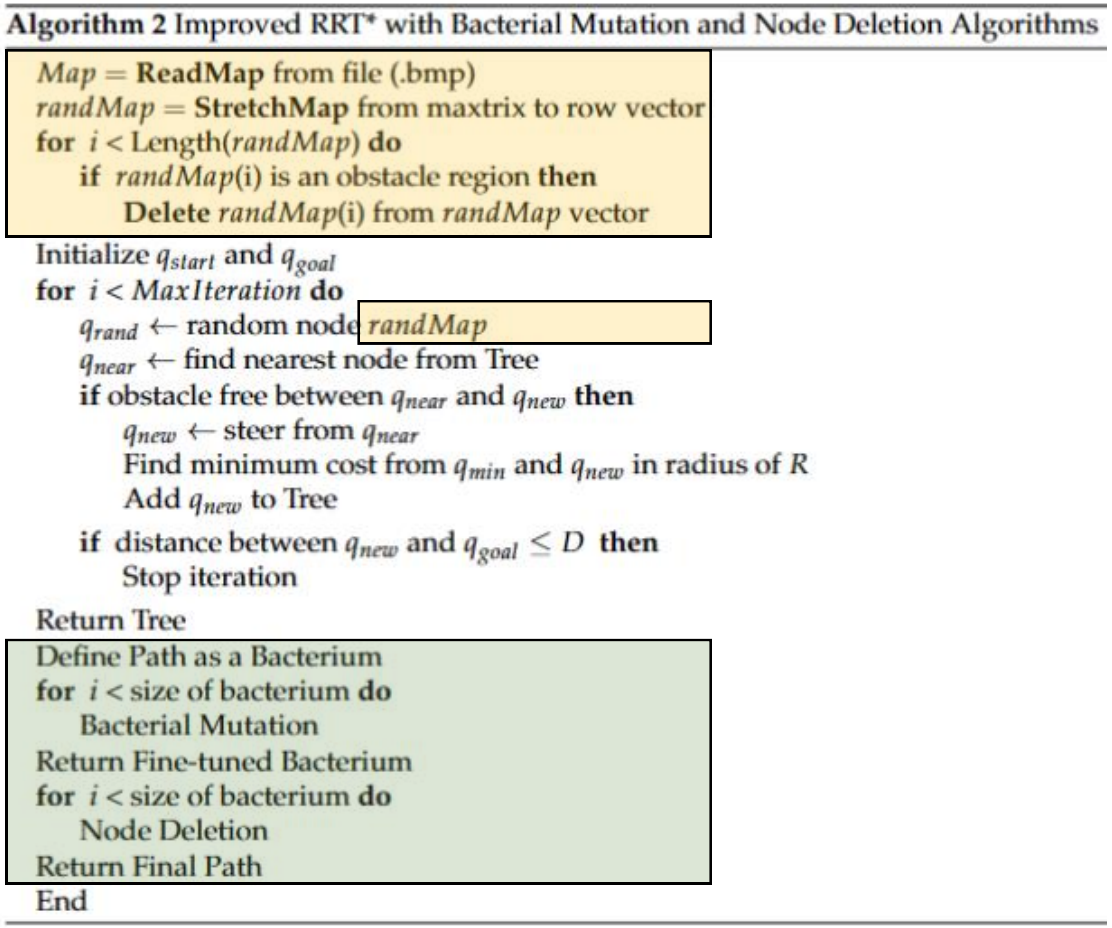


■ 21 (0,4)	22 (1,4)	23 (2,4)	24 (3,4)	25 (4,4)
16 (0,3)	17 (1,3)	18 (2,3)	19 (3,3)	20 (4,3)
11 (0,2)	12 (1,2)	■ 13 (2,2)	■ 14 (3,2)	15 (4,2)
6 (0,1)	7 (1,1)	■ 8 (2,1)	9 (3,1)	10 (4,1)
1 (0,0)	■ 2 (1,0)	3 (2,0)	4 (3,0)	5 (4,0)

■ Obstacle Region

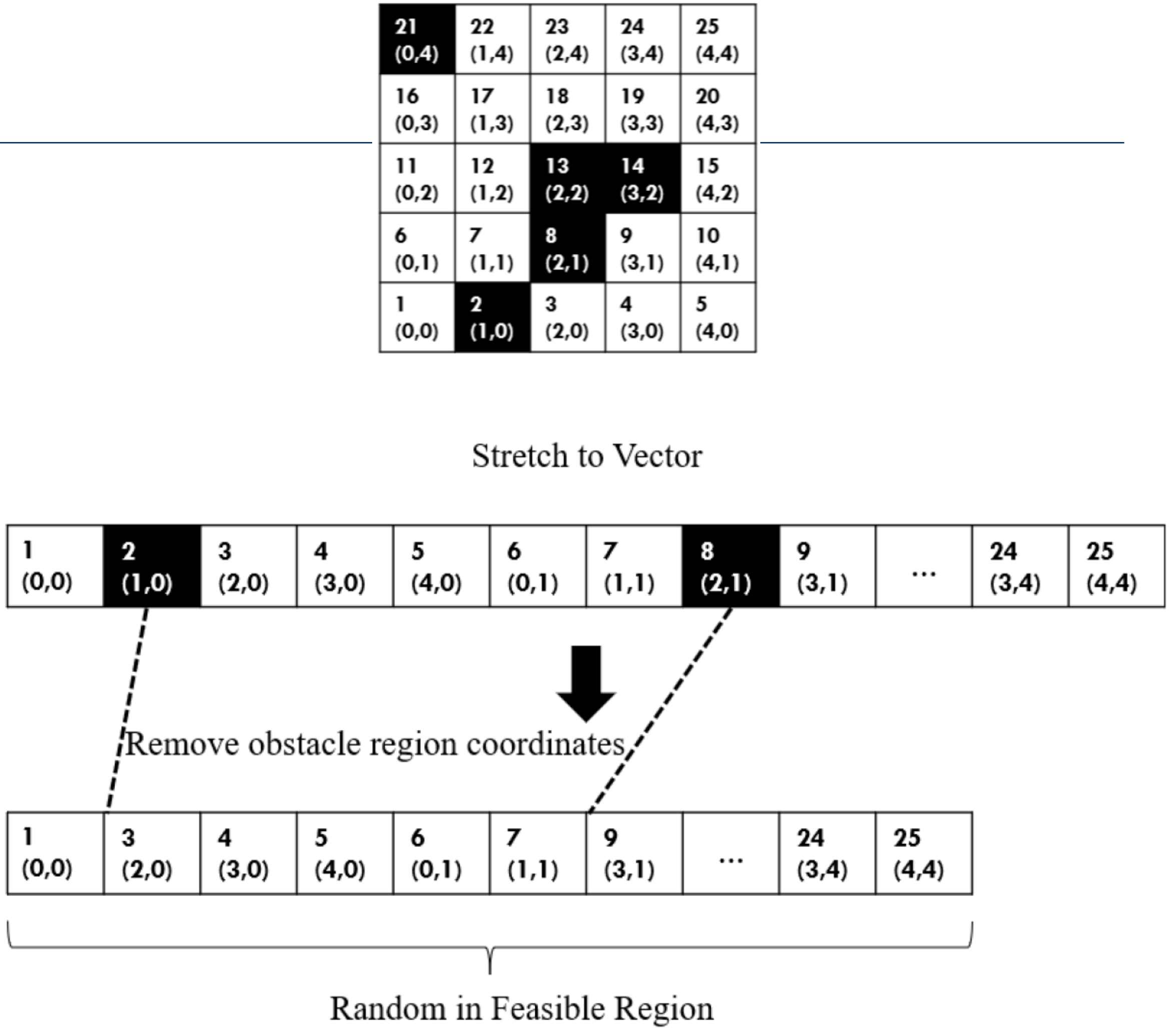
□ Feasible Region

Improved Algorithm



Algorithm 2 Improved RRT* with Bacterial Mutation and Node Deletion Algorithms

```
$Map$ = ReadMap from file (.bmp)
$randMap$ = StretchMap from maxtrix to row vector
for $i < \text{Length}(randMap)$ do
    if $randMap(i)$ is an obstacle region then
        Delete $randMap(i)$ from $randMap$ vector
Initialize $q_{start}$ and $q_{goal}$
for $i < MaxIteration$ do
    $q_{rand} \leftarrow$ random node $randMap$
    $q_{near} \leftarrow$ find nearest node from Tree
    if obstacle free between $q_{near}$ and $q_{new}$ then
        $q_{new} \leftarrow$ steer from $q_{near}$
        Find minimum cost from $q_{min}$ and $q_{new}$ in radius of $R$
        Add $q_{new}$ to Tree
    if distance between $q_{new}$ and $q_{goal} \leq D$ then
        Stop iteration
Return Tree
Define Path as a Bacterium
for $i <$ size of bacterium do
    Bacterial Mutation
Return Fine-tuned Bacterium
for $i <$ size of bacterium do
    Node Deletion
Return Final Path
End
```



21 (0,4)	22 (1,4)	23 (2,4)	24 (3,4)	25 (4,4)
16 (0,3)	17 (1,3)	18 (2,3)	19 (3,3)	20 (4,3)
11 (0,2)	12 (1,2)	13 (2,2)	14 (3,2)	15 (4,2)
6 (0,1)	7 (1,1)	8 (2,1)	9 (3,1)	10 (4,1)
1 (0,0)	2 (1,0)	3 (2,0)	4 (3,0)	5 (4,0)

Stretch to Vector

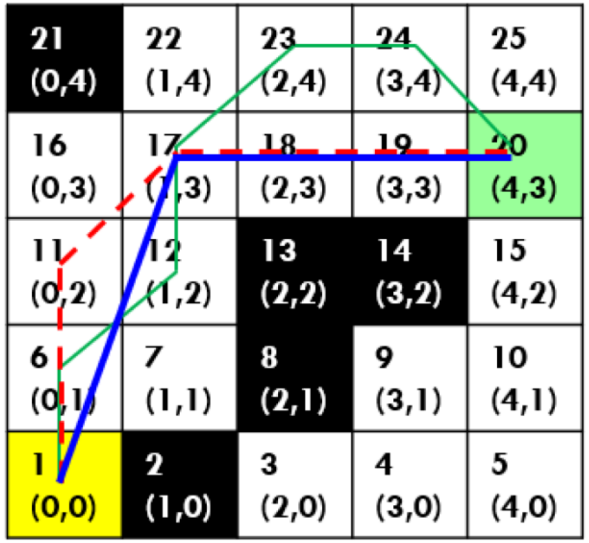
1 (0,0)	2 (1,0)	3 (2,0)	4 (3,0)	5 (4,0)	6 (0,1)	7 (1,1)	8 (2,1)	9 (3,1)	…	24 (3,4)	25 (4,4)

Remove obstacle region coordinates

1 (0,0)	3 (2,0)	4 (3,0)	5 (4,0)	6 (0,1)	7 (1,1)	9 (3,1)	…	24 (3,4)	25 (4,4)

Random in Feasible Region

Post-processing Algorithm



21 (0,4)	22 (1,4)	23 (2,4)	24 (3,4)	25 (4,4)
16 (0,3)	17 (1,3)	18 (2,3)	19 (3,3)	20 (4,3)
11 (0,2)	12 (1,2)	13 (2,2)	14 (3,2)	15 (4,2)
6 (0,1)	7 (1,1)	8 (2,1)	9 (3,1)	10 (4,1)
1 (0,0)	2 (1,0)	3 (2,0)	4 (3,0)	5 (4,0)

🟨 Starting Node

🟩 Goal Node

The solution from Improved RRT*
Define as Bacterium
(Green Path)

1 (0,0)	6 (0,1)	12 (1,2)	17 (1,3)	23 (2,4)	24 (3,4)	20 (4,3)

Bacterial Mutation

After bacterial mutation process
(Red Path)

1 (0,0)	6 (0,1)	11 (0,2)	17 (1,3)	18 (2,3)	19 (3,3)	20 (4,3)

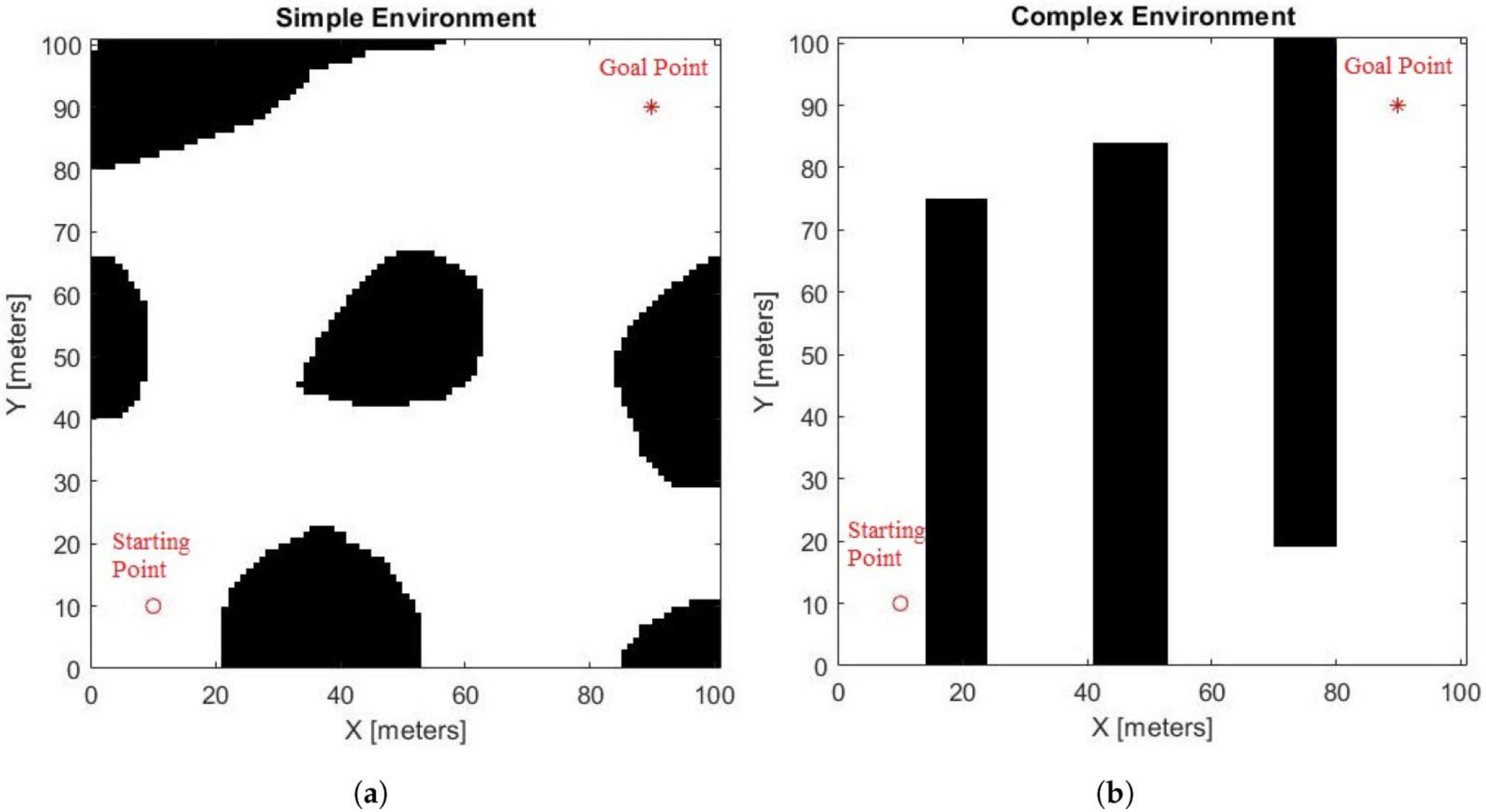
Local Search (Deletion)

Final Solution
(Blue Path)

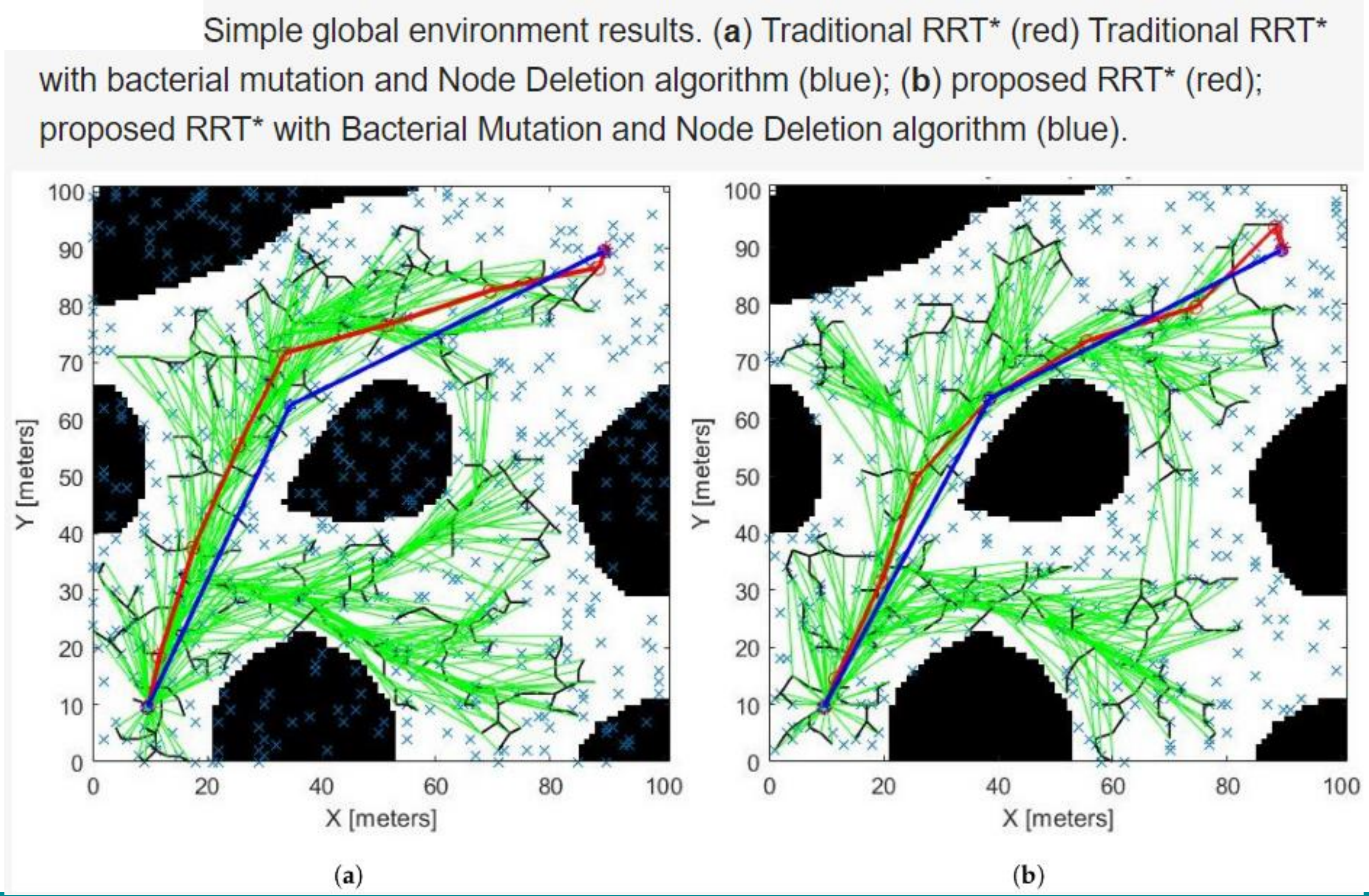
1 (0,0)	17 (1,3)	20 (4,3)

Experimental Results

- Global Environment

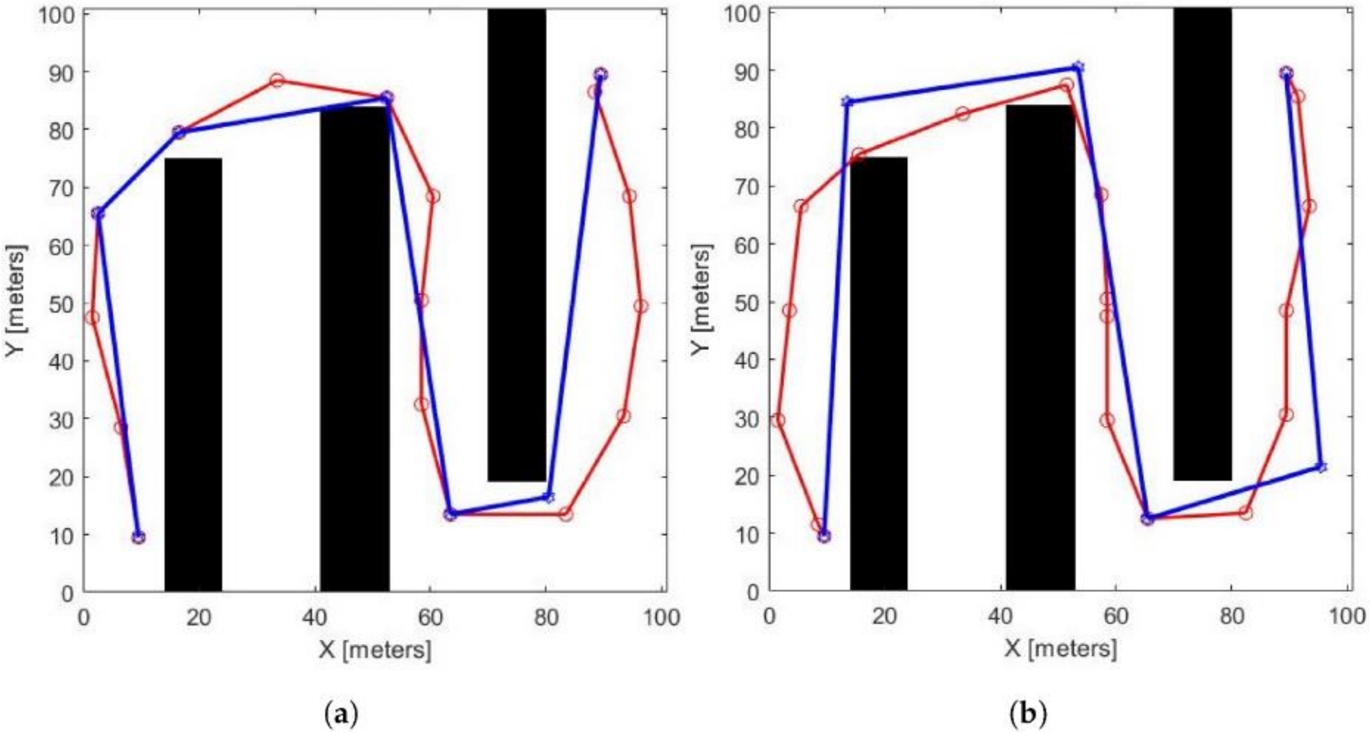


Experimental Results



Simple global environment results. (**a**) Traditional RRT* (red) Traditional RRT* with bacterial mutation and Node Deletion algorithm (blue); (**b**) proposed RRT* (red); proposed RRT* with Bacterial Mutation and Node Deletion algorithm (blue).

Experimental Results



(a)

(b)

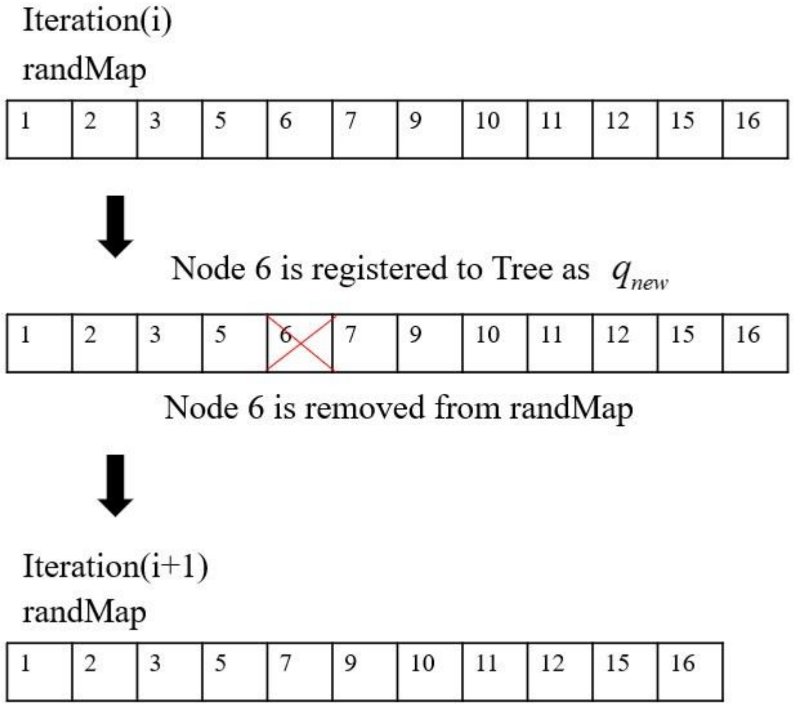
Complex global environment Results. (**a**) Traditional RRT* (red); traditional RRT* with Bacterial Mutation and Node Deletion algorithm (blue); (**b**) proposed RRT* (red); proposed RRT* with Bacterial Mutation and Node Deletion algorithm (blue).

A Rapidly-Exploring Random Tree Algorithm by Reducing Random Map Size

2023 9th International Conference on Automation, Robotics and Applications (ICARA 2023)
February 10-12, 2023 | Abu Dhabi, United Arab Emirates

Improved Algorithm

- An Improvement for Improved Algorithm for Path Planning Task of Mobile Robot
- To avoid the density of random nodes and improve the exploration of the algorithm



Iteration(i)
randMap

1	2	3	5	6	7	9	10	11	12	15	16

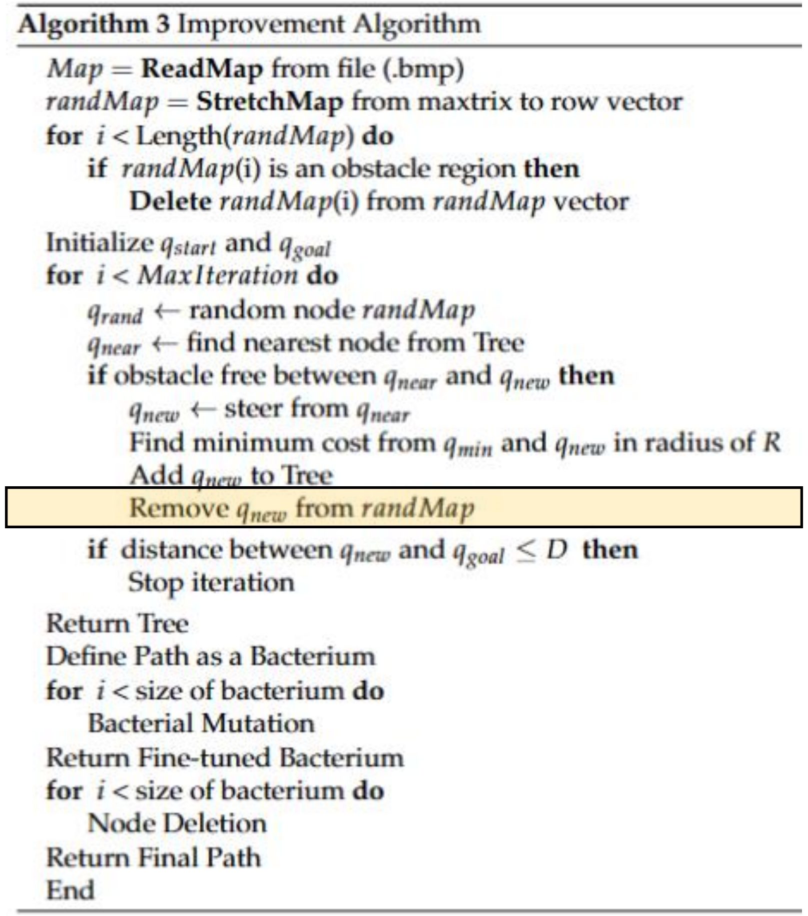
Node 6 is registered to Tree as q_{new}

1	2	3	5	$\cancel{6}$	7	9	10	11	12	15	16

Node 6 is removed from randMap

Iteration(i+1)
randMap

1	2	3	5	7	9	10	11	12	15	16



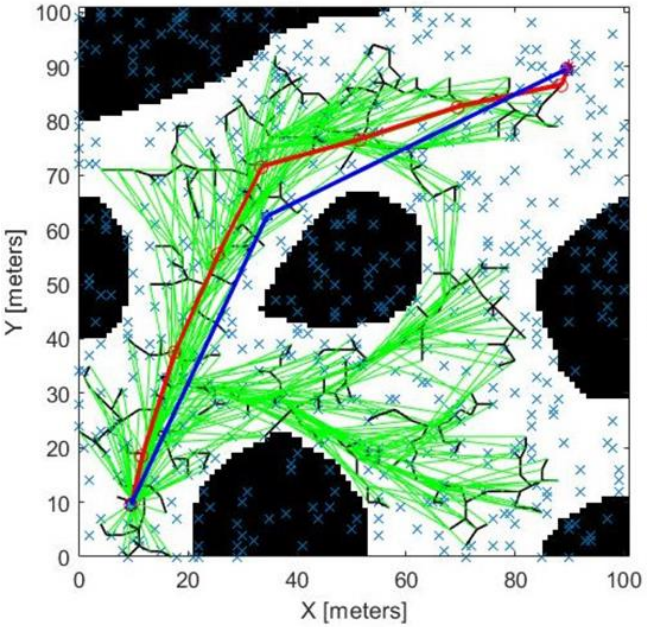
Algorithm 3 Improvement Algorithm

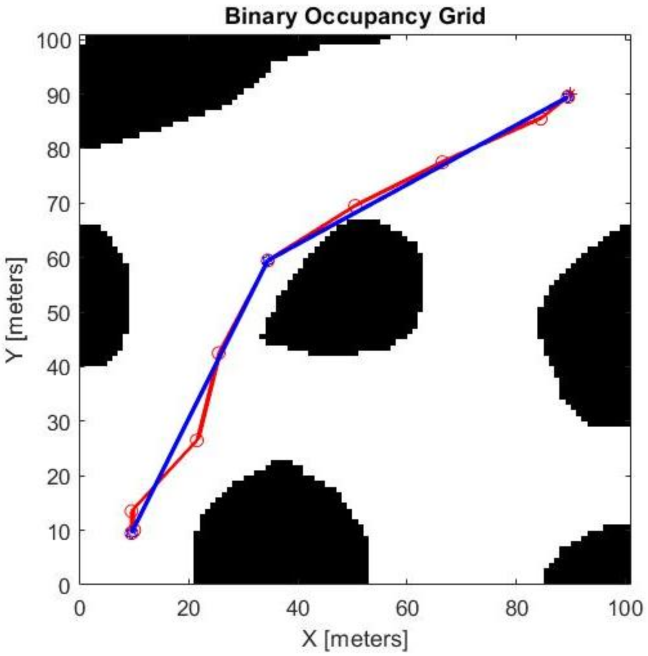
```
$Map$ = **ReadMap** from file (.bmp)
$randMap$ = **StretchMap** from maxtrix to row vector
**for** $i < \text{Length}(randMap)$ **do**
    **if** $randMap$(i) is an obstacle region **then**
        **Delete** $randMap$(i) from $randMap$ vector
Initialize $q_{start}$ and $q_{goal}$
**for** $i < MaxIteration$ **do**
    $q_{rand} \leftarrow$ random node $randMap$
    $q_{near} \leftarrow$ find nearest node from Tree
    **if** obstacle free between $q_{near}$ and $q_{new}$ **then**
        $q_{new} \leftarrow$ steer from $q_{near}$
        Find minimum cost from $q_{min}$ and $q_{new}$ in radius of $R$
        Add $q_{new}$ to Tree
        Remove $q_{new}$ from $randMap$
    **if** distance between $q_{new}$ and $q_{goal} \leq D$ **then**
        Stop iteration
Return Tree
Define Path as a Bacterium
**for** $i <$ size of bacterium **do**
    Bacterial Mutation
Return Fine-tuned Bacterium
**for** $i <$ size of bacterium **do**
    Node Deletion
Return Final Path
End
```

Experimental Results

- We combined the improvement algorithm with the Bacterial Mutation and Node Deletion algorithms

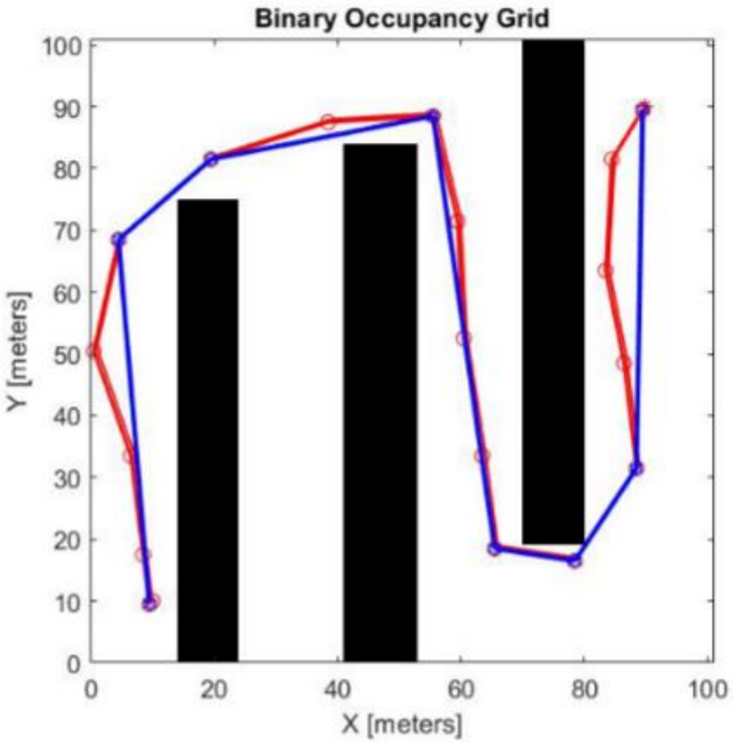
Proposed RRT* (red);
Journal Paper





Proposed RRT* (red)
Proposed RRT* with Bacterial Mutation and Node Deletion Algorithms (blue);
Conference Paper

Experimental Results of Complex Environment



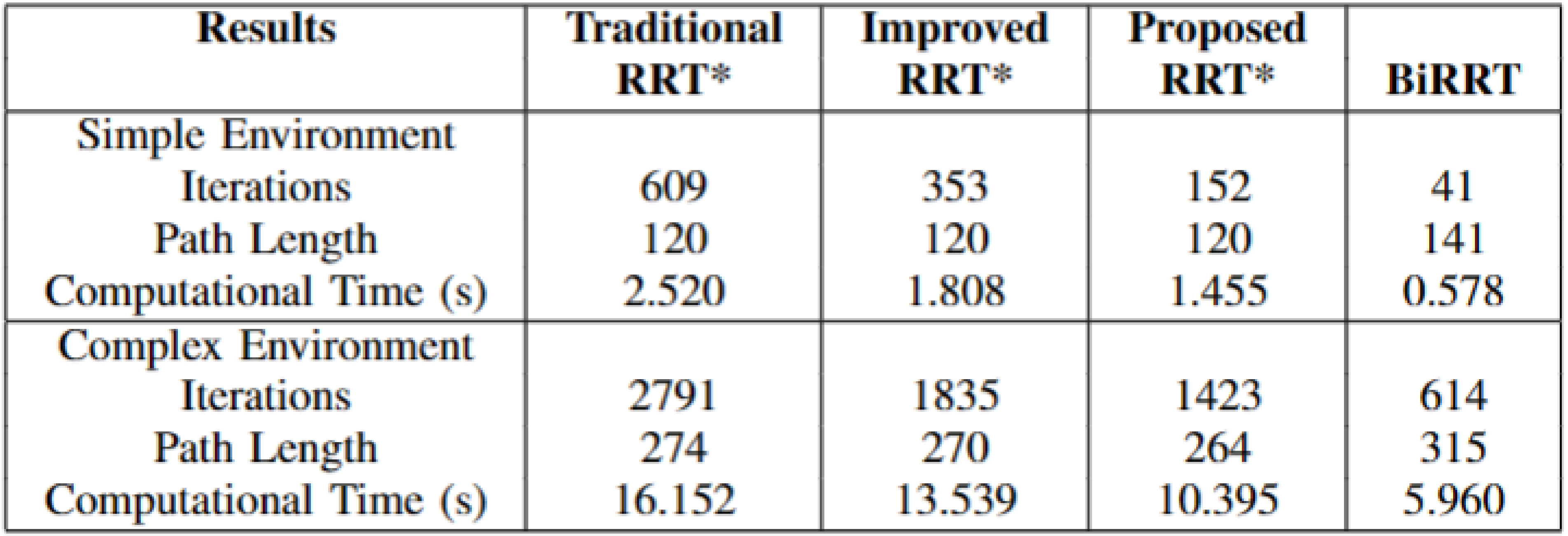
(b) Proposed RRT* with Reduced Random Map Size

Proposed RRT* (red)
Proposed RRT* with Bacterial Mutation and Node Deletion Algorithms (blue);
Conference Paper

Computational Results

Journal

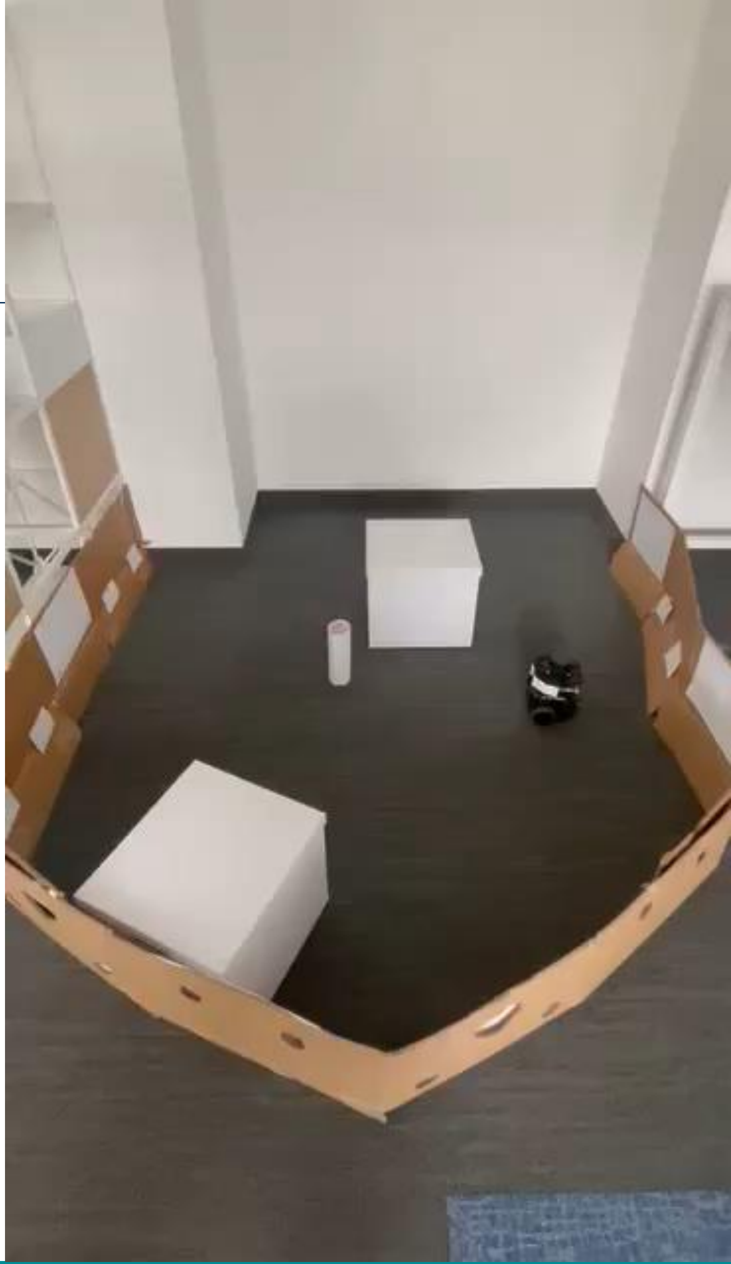
Conference



Results	Traditional RRT*	Improved RRT*	Proposed RRT*	BiRRT
Simple Environment				
Iterations	609	353	152	41
Path Length	120	120	120	141
Computational Time (s)	2.520	1.808	1.455	0.578
Complex Environment				
Iterations	2791	1835	1423	614
Path Length	274	270	264	315
Computational Time (s)	16.152	13.539	10.395	5.960

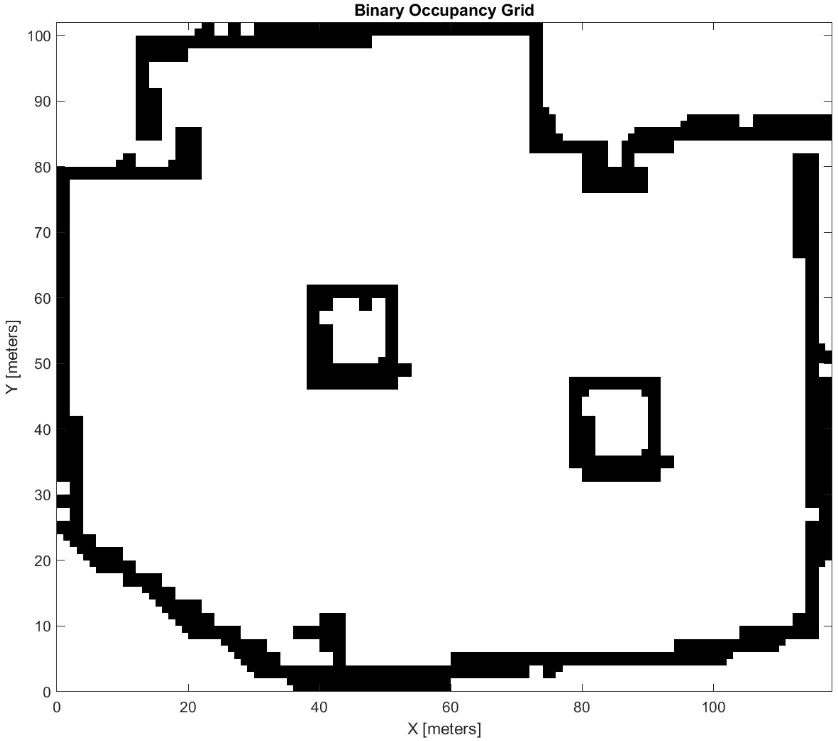
Implementing to Real Robot

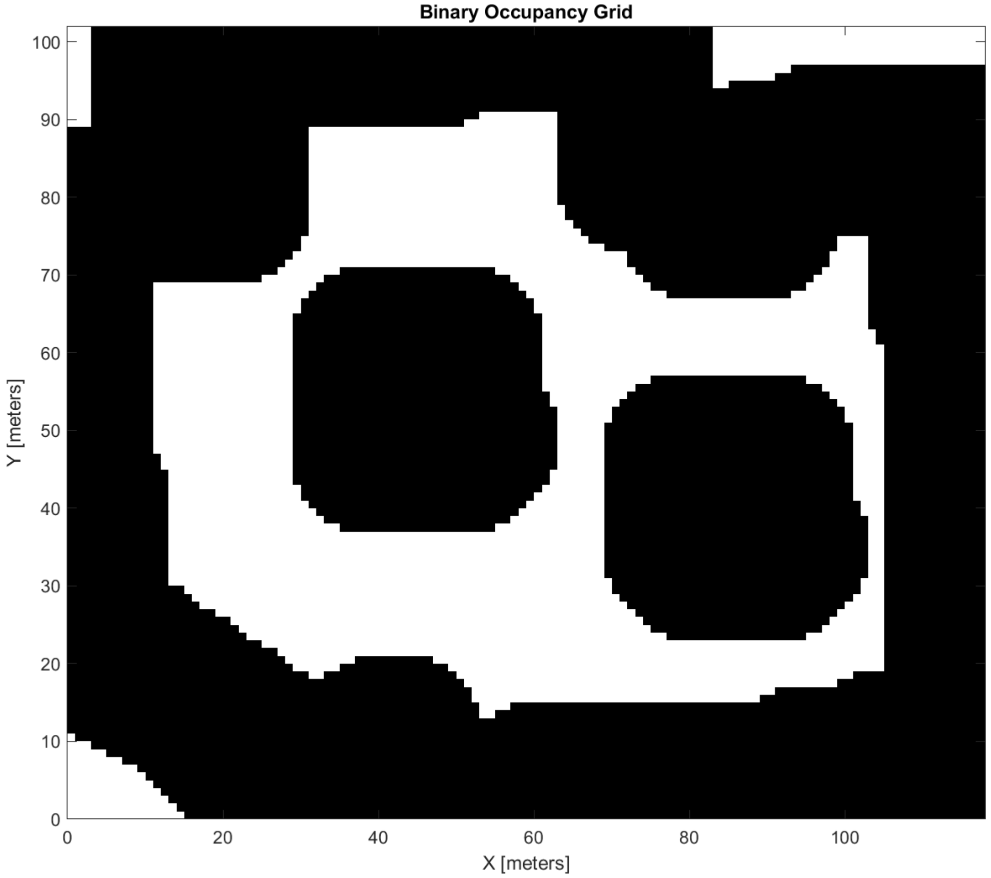
- The path planning result from a post-processing algorithm was sent to TurtleBot3-Burger by MATLAB programming via Robot Operating System.



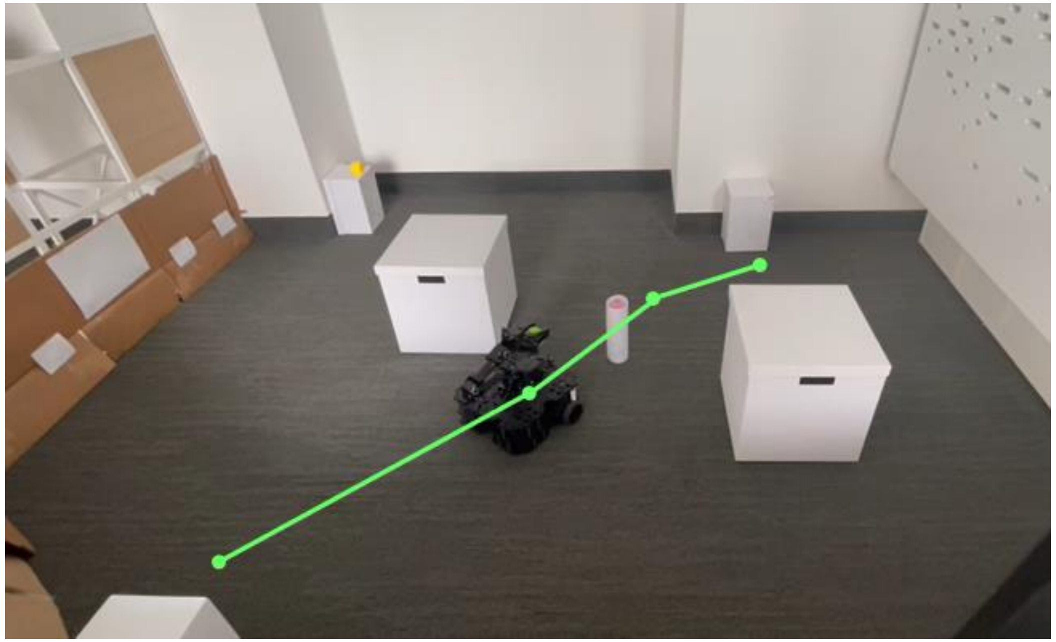
Unknown Static Obstacle Avoidance

- Map Generating from SLAM

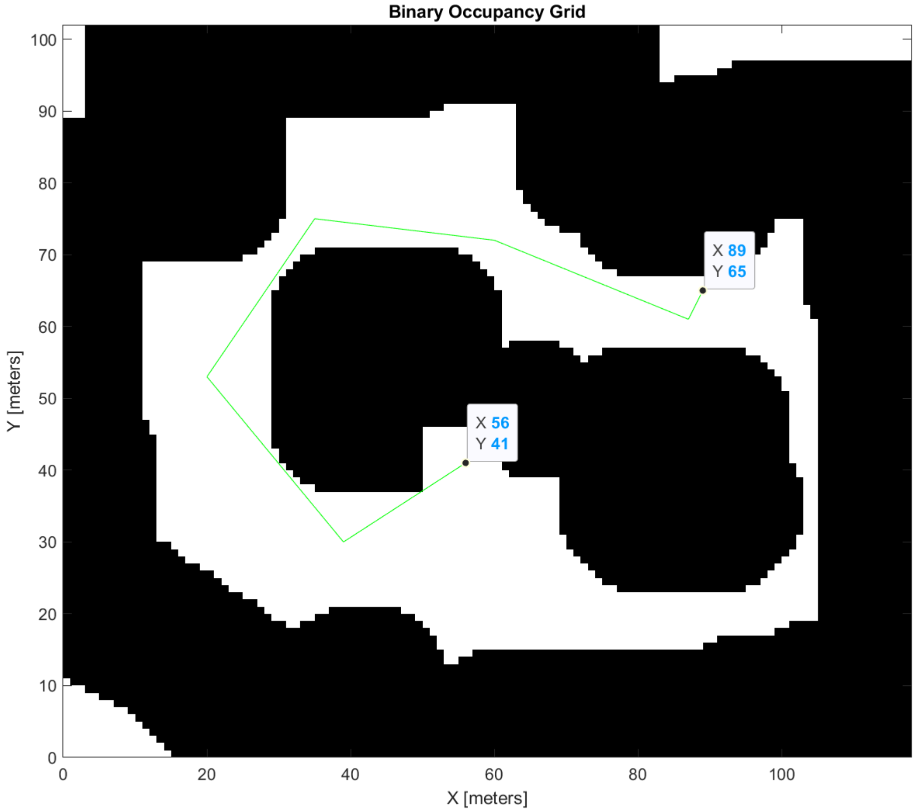




Result: Store to Station 1

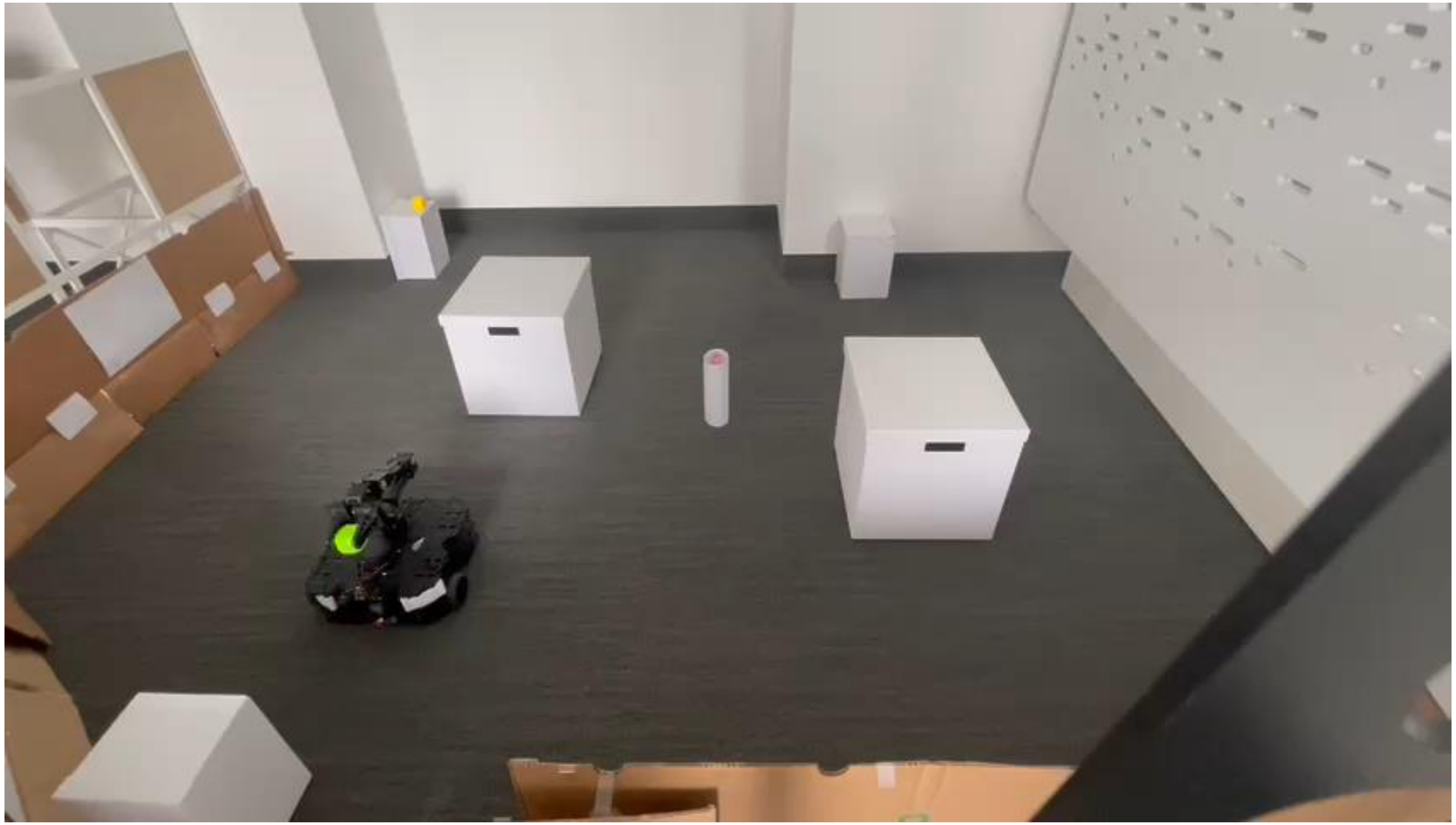


First Path Planning Result



New Path Planning

Result: Store to Station 1



Thank you for your attention