

[05-08-2022]

# [RMP final report]

[2022 Spring]

—— Lei Shi, Jiaming Zhang



**Project Title:** EN 530.663 Robot motion planning final project report  
**Project Start/End Date:** 05-01-2022 ~ 05-06-2022

## *Problem 1*

- The path planning of an 4-link planar serial manipulator, with the proximal end fixed in space.

### *1. Information on my system*

1.1. Dimension of the manipulator:  $n = 4$ ;

1.2 C-space:  $q_{init} = [0;0;0;0]$   $q_{goal} = [2;2.1;1.9;1.7]$  Workspace:  $q_{init} = [0;0]$ ;

1.3 Obstacles:  $Obs = \{[5 \ 25 \ 25 \ 5; 20 \ 20 \ 35 \ 35], [-30 \ -10 \ -10 \ -30; -25 \ -25 \ 10 \ 10], [-8 \ 30 \ 30 \ -8; -25 \ -25 \ -5 \ -5]\}$ ;

1.4 Length of each section:  $L = 8$ ;

1.5 Unit of system: T: meter(m) R: radian (rad);

1.6 Bound of world:  $X \sim [-30, 30]$   $Y \sim [-30, 40]$ ;

### *2. RRT method*

2.1 RRT settings: Step-size:  $\Delta q = 0.05$ ;

Number of nodes limit: NumNodes =  $5e5$  ;

Reaching target condition:  $\text{norm}(q_{new} - q_{goal}, 2) < 0.1$ ,  $q$  is  $4 \times 1$  vector, i.e the configuration distance;

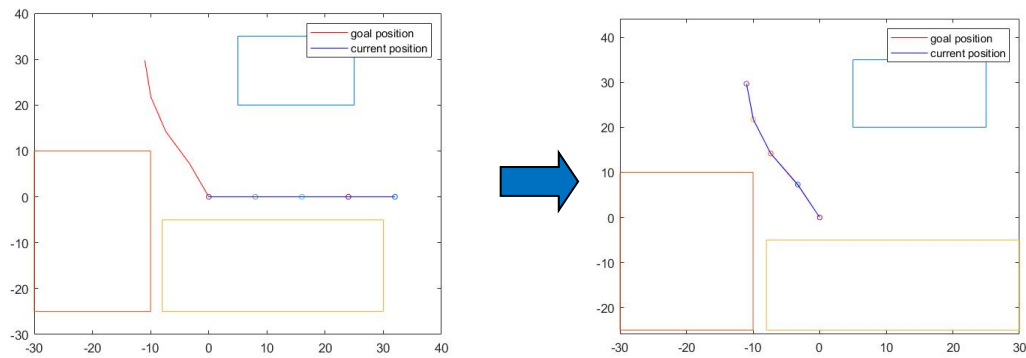
2.1. Result: Time cost: around **2.5s**;

Sample nodes number: 800~850;

Final error:

$\text{norm}(q_{last} - q_{goal}, 2)$ ,  $q$  is  $4 \times 1$  vector, i.e total configuration error is **0.0098**;

$\text{norm}(w_{end} - w_{goal}, 2)$ ,  $w$  is  $2 \times 1$  vector, i.e end joint workspace error: **0.01277**;

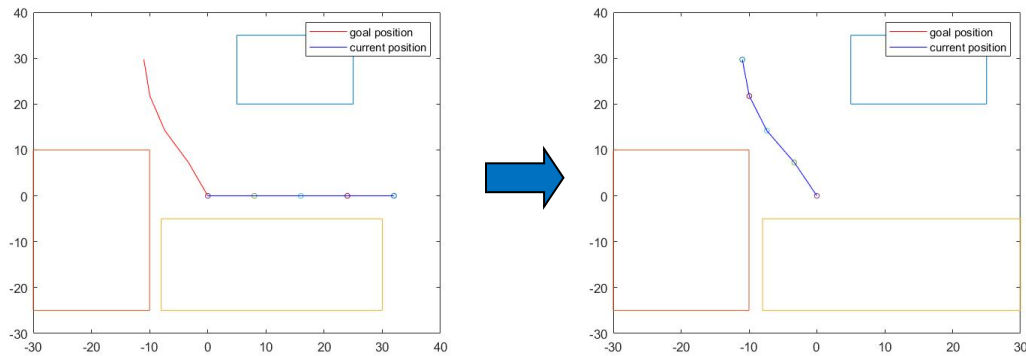


### 3. PRM method

2.1 PRM settings : Neighborhood nodes:  $K = 4$

Number of sample nodes: NumNodes = 1000;

2.2 Result: Time cost: around 4s;



### 4. Consideration

In this problem, we need to generate random nodes, find the nearest node in C-space but check collision in workspace. For RRT part, I prefer RRT\* for faster velocity: every 10 nodes, I use  $q_{goal}$  as random node.

## Problem 2

- The path planning of a holonomic planar rigid robot with artificial potential field.

### 1. Information on my system

1.1. Dimension of the robot:  $n = 3$   $(x, y, \theta)$ ;

1.2 Initial/goal point :  $q_{\text{init}} = [7, 2, 0]$   $q_{\text{goal}} = [6, 9, -0.2]$  ;

1.3 Obs #1 :  $(5, 3), (10, 3), (10, 2)$

Obs #2 :  $(0, 4), (6, 5), (0, 6)$

Obs #3 :  $(5, 7), (10, 8), (10, 5.5)$

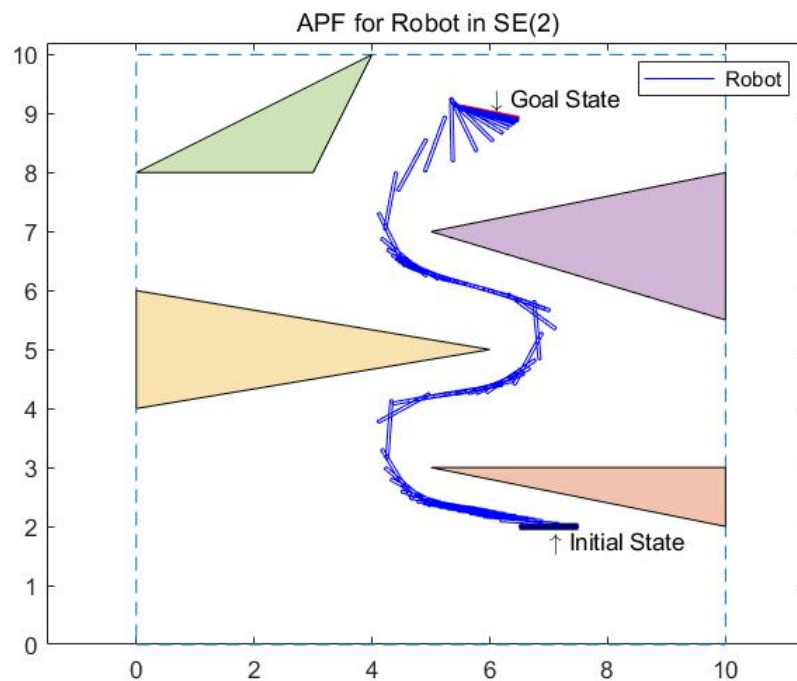
Obs #4 :  $(0, 8), (3, 8), (4, 10)$  ;

1.4 The robot is set as a Rectangular Box with Length of 1 and width of 0.05 ;

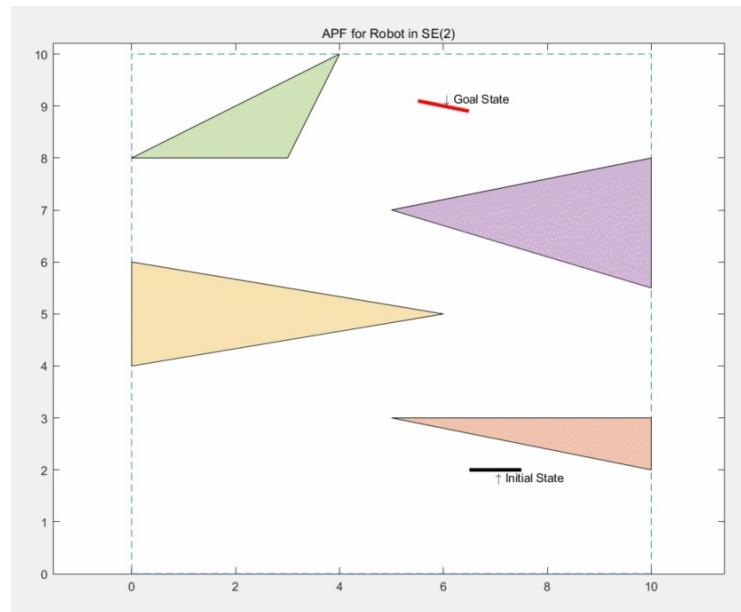
1.5 Bound of world:  $X \sim [0, 10]$   $Y \sim [0, 10]$ ;

### 2. Result

Videos:



Map:



### 3. Consideration

Based on the Non-Euclidean configuration of the robot, we added three control points on the robot. And we assume the gradient of the Artificial Potential Field can now be regarded as some imaginary force that pulling or pushing the control points.

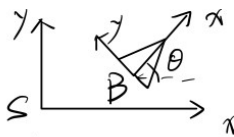
The Attractive Potential Energy is constructed by the norm of the difference of the current state vector and goal state vector.

The Repulsive Energy is constructed by the distance of the control points and the Obstacle polygons.

### Problem 3

- The path planning of a flexible needle (planar problem), where the asymmetric needle tip is modeled as a nonholonomic mobile robot (geometrically modeled as an isosceles triangle) that moves in the plane.

Analytical part:



state  $(x, y, \theta)$

$$V = r \cdot u_\phi \quad \dot{\theta} = u_w$$

$$\begin{cases} \dot{x} = r \cdot \cos \theta \cdot u_\phi \\ \dot{y} = r \cdot \sin \theta \cdot u_\phi \\ \dot{\theta} = u_w \end{cases}$$

$$\begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{pmatrix} = \begin{pmatrix} r \cos \theta & 0 \\ r \sin \theta & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} u_\phi \\ u_w \end{pmatrix}$$

$$\vec{\dot{q}} = (g^{-1} \dot{g})^v = \vec{V}^b = u_1 \vec{V}_1 + u_2 \vec{V}_2$$

$$u_1 = v \quad V_1 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \quad \text{no } y\text{-velocity}$$


$$u_2 = u_w \quad V_2 = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

#### 1. Information on my system

1.1. Dimension of the robot:  $n = 3$   $(x, y, \theta)$ ;

1.2 Initial/goal point :  $q_{\text{init}} = [0; 0; 0]$   $q_{\text{goal}} = [35; 45; 0]$  ;

1.3 Obstacles:  $\text{Obs} = \{[7 \ 15 \ 15 \ 7; 10 \ 10 \ 30 \ 30], [30 \ 40 \ 40 \ 30; 20 \ 20 \ 40 \ 40]\}$ ;

1.4 Shape of the robot: equilateral triangle with edge length 3 and radius of wheel is  $2/\pi$ . Our bodyframe origin is:  the middle point of an edge;

1.5 Unit of system: T: meter(m) R: radian (rad);

1.6 Bound of world:  $X \sim [-1.5, 50]$   $Y \sim [-1.5, 50]$ ;

## 2. RRT method

2.1 RRT settings: Step-size:  $\text{del\_t} = 0.5\text{s}$ ;

Number of nodes limit: NumNodes = 1500 ;

Reaching target condition:  $\text{norm}(\mathbf{q}_{\text{new}} - \mathbf{q}_{\text{goal}}, 2) < 0.8$ ,  $\mathbf{q}$  is  $2 \times 1$  vector, i.e the workspace distance( **our model can set final error be within 0.5, but it's too slow. So, we set error be 1.5 which leads to that it can work out within 5~30s**);

Wheel rotation rate:  $u_{\phi} = 1$ ;

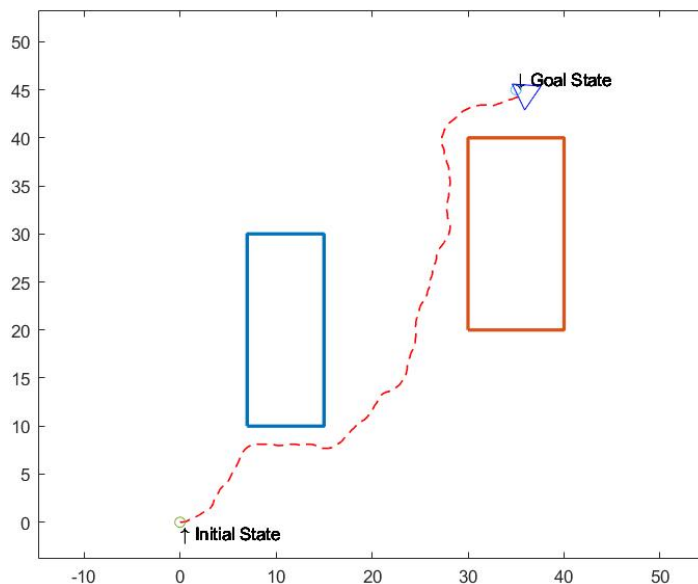
Wheel turning rate:  $u_w = \{\pi/7 \ 0 \ -\pi/7\}$ ;

2.1. Result: Time cost: around **5s**;

Sample nodes number: 1200~1400;

Final error:

norm(q\_last - q\_goal,2), q is 2\*1 vector, i.e total configuration error is **0.6848**;



## 5. Consideration

5.1 In this problem, I prefer RRT\* for better velocity: every 10 nodes, I use `q_goal` as random node;

5.2 When setting random points and seeking the nearest nodes, we only consider x-y coordinate but ignore theta coordinate;

5.3 Because of the limit of three fixed  $u_w$ , it often produces new points which are the same as existed points. So, for this situation, we need to generate new random point for different new point;

5.4 Also,  $u_w = \pi/7$  becomes the constant curvature that the needle tip follows. Then  **$u_w = -\pi/7$  corresponds to the case where you stop inserting the needle, rotate it by  $180^\circ$ , and then re-insert the needle as  $u_w = \pi/7$ , i.e flip and move.** The case  $u_w = 0$  corresponds to the case where you insert the needle while spinning it fast (100% duty cycle), which results in the straight needle insertion;

5.5 For collision check, I generate the outer circle of the positive triangle robot. Since my body frame origin is at the middle point of an edge, so I only need to ensure the distance between that origin and obstacle is more than the radius of the outer circle of the positive triangle robot, i.e.  $\sqrt{3}$ , by function ClosestPointOnEdgeToPoint;

5.4 If theta is not in  $[-\pi, \pi]$ , I convert it into the range;

5.5 When choosing  $u_w$ :

**if the angle between  $q_{near}$  and  $q_{rand}$  is within  $[-0.3, 0.3]$ ,  $u_w = 0$ ;**

**else if the angle between  $q_{rand}$  and positive x-axis is smaller than the theta of  $q_{near}$ , i.e  $q_{rand}$  is on the right part of  $q_{near}$ ,  $u_w = -\pi/7$ ;**

**else if that angle is bigger than the theta of  $q_{near}$ , i.e  $q_{rand}$  is on the left part of  $q_{near}$ ,  $u_w = \pi/7$ .**