Lab 5: Potential Field

Zheyuan Xie, Zixuan Lan

November 15, 2018

1 Introduction

In this lab we achieved the planning Lynx robot in cluttered environment by designing appropriate potential fields. We evaluate our method in simulated environment in MATLAB, explore the effect of different parameters, and compare the potential field method with the probabilistic road map method (PRM) studied in Lab 3.

2 Pre-lab Tasks

1. Write a mathematical description of a potential field controller for the Lynx robot. Explain your choices.

   Answer:
   The objective for this controller is to efficiently and accurately control the position of all joints of Lynx. Below are equations showing how my controller works. In the equations, q_f is the destination (3x1 vector), q_S (3x1 vector) is the current position of lynx robot. i indicates the iteration number. These equations apply to all joints of lynx robot. For simplicity, here I just show the equations for joint j, where j ranges from 1 to 5.

   The mathematical expression is as follows:

$$F^i = {F_{att}}^i + {F_{rep}}^i$$

   Where ${F_{att}}^i$ is the attractive force at joint j, ${F_{rep}}^i$ is the repulsive force.

$${F_{att}}^i = -\zeta^i({O(q_s)}^i - {O(q_f)}^i), \text{when } \ ||{O(q_s)}^i - {O(q_f)}^i|| \leq D_1$$

$${F_{att}}^i = -\frac{({O(q_s)}^i - {O(q_f)}^i)}{||{O(q_s)}^i - {O(q_f)}^i||}, \text{when } \ ||{O(q_s)}^i - {O(q_f)}^i|| > D_1$$

   Where D_1 is the threshold I set for switching from conic well to parabolic well potential. ζ is the attractive field strength, which is basically a scale factor. The reason I want to switch between conic well and parabolic well is that, when the robot is far away, I want the attractive force to be unit magnitude; when the robot is getting closer to the final position, I want the attractive force to be smaller and smooth. I also want to eliminate the discontinuity at final position, so a scaling factor is necessary.

$${F_{rep}}^i = \eta^i(\frac{1}{{\rho(O(q_s))}^i} - \frac{1}{\rho_0}) * \frac{1}{({\rho(O(q_s))}^i)^2} * \frac{{O(q_s)}^i - b}{||{O(q_s)}^i - b||}, \text{when } \ \rho < \rho_0$$

$${F_{rep}}^i = 0, \text{when } \ \rho > \rho_0$$

Where $\rho(O(q_s))$ is the shortest distance between O_j and obstacle, j is the joint number. ρ_0 is the threshold (distance of influence of obstacle) I set for the obstacle. O_j is the DH origin at joint j. η is the repulsive field strength, which is basically a scale factor.

The repulsive force by obstacle is 0 when far from the final position, and the repulsive force increases dramatically until it gets to infinity at obstacle boundary. Another reason for using this model is that the repulsive force at ρ_0, both equations yield, and this proves the contiguity of the function.

Therefore, I get

$$q_s{}^{i+1} = q_s{}^i + \alpha^i \frac{\tau^i}{||\tau^i||}, \text{when } ||q_s{}^i - q_f|| > e$$

$$q_s{}^{i+1} = q_s{}^i + ||q_s{}^i - q_f|| \frac{\tau^i}{||\tau^i||}, \text{when } ||q_s{}^i - q_f|| \leq e$$

$$\tau = (J_v{}^T)^i (F_{att}{}^i + F_{rep}{}^i)$$

Where τ (5x1)is the torques at joints, α is a constant step size for robot movement, e is the threshold I set for joint movement. If the required movement is larger than e, the robot would instead move by α; this is to update the orientation of movement frequently, so that the position control is more accurate. J_v is a 3x5 matrix. Note that $F_{att}{}^i$ and $F_{rep}{}^i$ is the summation of forces at joint j.

2. The performance of your potential field will change depending on what you choose for your specific parameters. For 1 parameter in your controller, predict what affect increasing/decreasing this parameter will have on your planner performance. Explain.

   Answer:

   Take the parameter η as an example. As η is the scaling factor of repulsive force, when ρ is within rho_0 , increasing η would increase the repulsive force. As the net force (F_{att} - F_{rep}) determines the orientation of the robot, increasing η might cause the robot to move for a longer distance until it gets the final position. Decreasing η would decrease the repulsive force, which means the robot would move faster toward the final position. At extreme condition, when η is very high, the robot would not be able to reach the final position, since the repulsive force is always higher than the attractive force. A direct result would be the robot oscillating at certain point. If η is too small, the attractive force is higher than repulsive force most time. The robot would move toward the final position until it hits the boundary of obstacle and stop suddenly due to the infinite repulsive force. This causes discontinuity and is bad for the motor. The robot would also oscillate at this point. Therefore, a balance between these two cases should be reached by tuning η.

3. Design a (set of) evaluation tests to check whether your prediction is correct. What environment will you use? What will you measure?

   Answer:

   The goal of the evaluation tests is to find the optimal set of parameters and also examine the influence of an individual parameter on the performance of the position control. In terms of eviroment, I would set up multiple convex obstacle in the work space. For example, 3 obstacle could be used, and two of them are close to each other and generate a narrow corridor in between. Experiments include,

   (a) test the performance of position control with 5 sets of different value of γ, from large to small value. Record the success rate of 10 different trials (final position is different per trial and 2 trials for each γ), number of iteration and time taken to get to final position.Plot the net force over time, to check the continuity of the forces exerted on each joint.

   (b) test the performance of position control with 5 sets of different value of η, from large to small value. Record the success rate of 10 different trials (2 for each η), number of iteration and time taken to get to final position. Plot the net force over time, to check the continuity of the forces exerted on each joint.

(c) test the performance of position control with 5 sets of different value of ρ, from large to small value. Record the success rate of 10 different trials (2 for each ρ), number of iteration and time taken to get to final position.Plot the net force over time, to check the continuity of the forces exerted on each joint.

(d) test the performance of position control with 5 sets of different value of $D1$, from large to small value. Record the success rate of 10 different trials (2 for each $D1$), number of iteration and time taken to get to final position.Plot the net force over time, to check the continuity of the forces exerted on each joint.

3 Method

3.1 Potential Field Planner

Our goal is to move the Lynx robot from initial configuration q_0 to final configuration q_f and avoid workspace obstacles. First define a set of control points $\mathcal{O} = \{o_1, o_2, o_3\}$, as shown in Figure 1. o_1, o_2 are center of frame 2 and 3 respectively. o_3 is the center of frame 5 at the tip of the end effector. Since the **rotation of Joint 5 has no effect on the workspace position and forces exerted on joint 2 will not produce torque on any joint**, we do not add a control point to frame 1 and frame 4. We denote by $o_i(q)$ the position of the i^{th} control point when the robot is at configuration q.

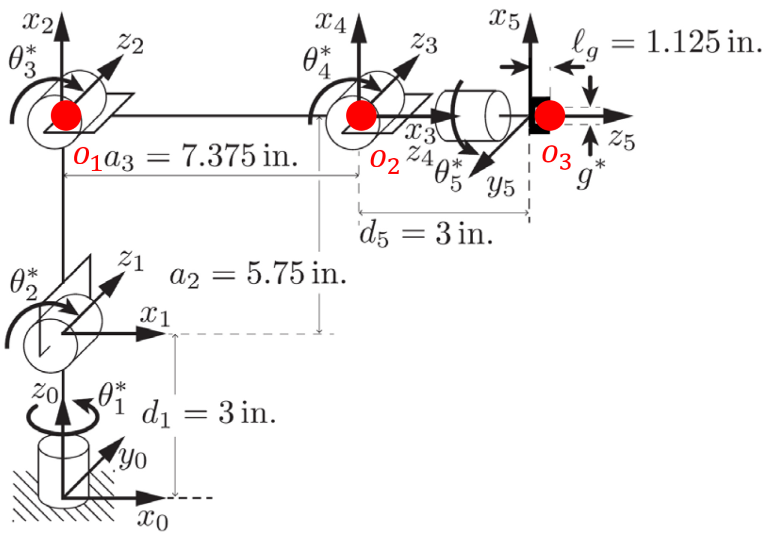


Figure 1: Control points used to define potential fields

Since we want to avoid workspace obstacles, we define the the attractive field and repulsive field both in the workspace:

- We use parabolic attractive field. The attractive field for each control point o_i is

$$U_{\text{att},i}(q) = \frac{1}{2}\zeta_i \left\| o_i(q) - o_i(q_f) \right\|^2$$

where ζ_i is the scale factor of the attractive field.

- The repulsive field for each control point o_i is

$$U_{\text{rep},i}(q) = \begin{cases} 0 & \rho_i(q) > \rho_0 \\ \frac{1}{2}\eta_i \left(\frac{1}{\rho(o_i(q))} - \frac{1}{\rho_0} \right)^2 & \rho_i(q) \le \rho_0 \end{cases}$$

where $\rho(o_i(q))$ is the shortest distance between the control point o_i and any workspace obstacles, ρ_0 is the workspace distance of influence for obstacles, and η_i is the scale factor of the repulsive field.

The robot starts from configuration $q = q_0$ and moves towards the q_f by the following steps:

1. Calculate attractive force on each origin.

$$F_{\text{att},i}(q) = -\nabla U_{\text{att},i}(q) = -\zeta_i(o_i(q) - o_i(q_f))$$

2. Calculate repulsive force on each origin.

$$F_{\text{rep},i}(q) = \begin{cases} 0 & \rho_i(q) > \rho_0 \\ \eta_i \left(\frac{1}{\rho(o_i(q))} - \frac{1}{\rho_0} \right) \frac{1}{\rho^2(o_i(q))} \cdot \frac{o_i(q)-b}{\|o_i(q)-b\|} & \rho_i(q) \le \rho_0 \end{cases}$$

   where b is the point on obstacle boundary closest to o_i.

3. Convert each workspace force to equivalent joint-space torques.

$$\tau(q) = \sum_i J_{v,i}^T(q)(F_{\text{att},i}(q) + F_{\text{rep},i}(q))$$

   Note that $J_{v,i}$ is different for every control points and have different dimensions. When summing over the torque vectors, we pad zeros to make them 5×1 vectors.

4. Update joint configuration.

$$q \leftarrow q + \alpha \frac{\tau(q)}{\|\tau(q)\|}$$

   where α is the update stepsize either can be fixed or adaptive.

5. We perform online control, step $1 \sim 4$ is executed at each time step. Otherwise iterate step $1 \sim 4$ until $\|q - q_f\| \le \epsilon$ and return a series of waypoints $< q_0, q_1, q_2, \cdots, q_f >$.

3.2 Escaping Local Minimum with Random Walk

It's not uncommon that the robot would get stuck at local minimum. To confirm if the robot is stuck at a local minimum, we check end-effector position history $< p_0, p_1, \cdots, p_{t-1}, p_t >$. We assume that if the history shows that the joints have similar position (within the threshold we set) for the last 4 iterations, the robot gets stuck at a local minimum.

$$\|p_t - p_{t-i}\| < \epsilon \quad (i = 1, 2, 3)$$

where ϵ is a empirical threshold value.

Then we take random walk to escape the local minimum. A random step in the configuration space is generated. For the Lynx robot $q_{\text{random-step}}$ has the following form (joint 5 and end effector does not affect).

$$q_{\text{random-step}} = (q_1 \pm v_1, q_2 \pm v_2, q_3 \pm v_3, q_4 \pm v_4, 0, 0)$$

v_1, v_2, v_3, v_4 are fixed small constants determined in experiment. The probability of adding $+v_i$ or $-v_i$ is 1/2. The **checkCollision.m** function created in Lab 3 is used here to make sure the new configuration after taking the random step $q_{t+1} = q_t + q_{\text{random-step}}$ won't collide with any obstacle. If the new configuration cause a collision, we sample again until there is a reasonable q_{t+1}.

3.3 Algorithm: Pseudo Code

We implemented our method in MATLAB. We define our task objective to be reaching a position, therefore the exiting condition for iteration is when the position error is smaller than a threshold. The start position and final position is convert into configuration space q with out **pose2q.m** function written in Lab 4. The orientation of the end effector is represented using Euler angles and the default orientation is z_e parallel to the ground plane.

Algorithm 1 Potential Field Planner

```
1: $q_{\text{start}} = \text{pose2q}(p_{\text{start}})$, $q_{\text{final}} = \text{pose2q}(p_{\text{final}})$
2: $q^0 \leftarrow q_{\text{start}}$, $i \leftarrow 0$
3: while $\left\|p^i - p_{\text{final}}\right\| < \epsilon$ do
4:     Calculate $F_{\text{rep}}$ and $F_{\text{att}}$
5:     Calculate $J_v$ and $\tau$
6:     $q^{i+1} \leftarrow q^i + \alpha^i \frac{\tau(q^i)}{\|\tau(q^i)\|}$
7:     if stuck in a local minimum then
8:         execute a random walk, ending at $q'$
9:         $q^{i+1} \leftarrow q'$
10:    $i \leftarrow i + 1$
11: return $< q^0, q^1, \cdots, q^i >$
```

4 Evaluation

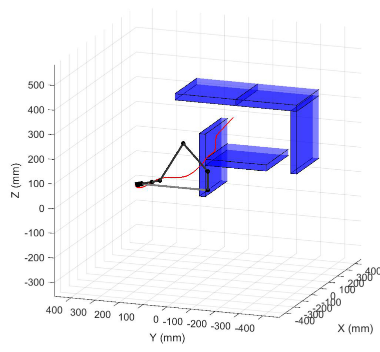
In this section we mainly use **map 4** and **example map** to explain our findings, however, our method also applies to other maps, some of the test results are displayed in Figure 2. A short video is available on YouTube: `https://youtu.be/RxHwPRxjh5o`. We will show some typical scenarios in Section 5.

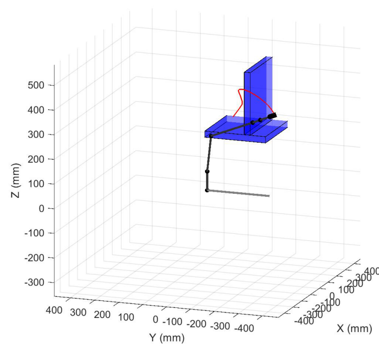
4.1 Overall Performance

We use **map 4** here to evaluate the method. The robot always starts at its zero configuration, which is $p_{start} = (292.1, 0, 222.25)$. Then, the robot tries to reach 6 different goal position using our potential field planner. The trajectories of the path is shown in Fig 3. Out of 6 trials, only 1 trial fails. The failure occurs when the robot is stuck at the local minimum, and this issue will further be discussed in the following section.

Table 1: Tests for Overall Performance

	q_{start}	q_{final}	result
trial1	[292.1,0,222.25]	[-150,200,350]	success
trial2	[292.1,0,222.25]	[-150,-200,350]	success
trial3	[292.1,0,222.25]	[100,200,300]	success
trial4	[292.1,0,222.25]	[100,-200,300];	fail
trial5	[292.1,0,222.25]	[200,100,100]	success
trial6	[292.1,0,222.25]	[200,-100,100]	success


(a) Map 1


(b) Map 2

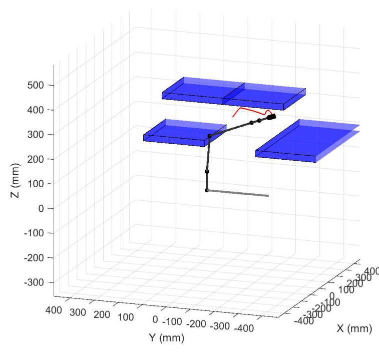
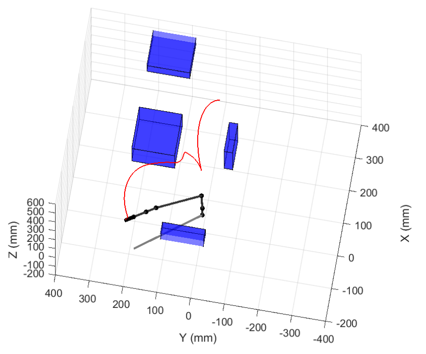
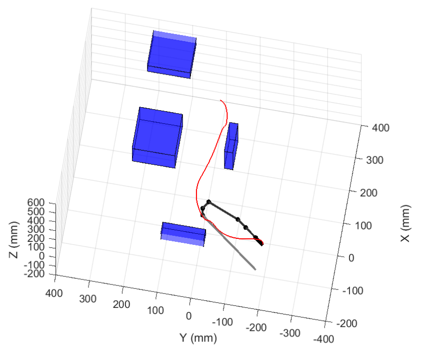

(c) Map 3

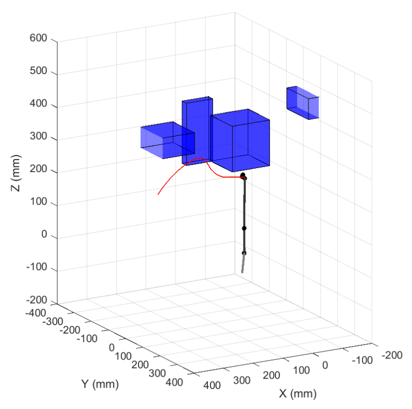
Figure 2: Performance tests in map 1, map2, and map 3.



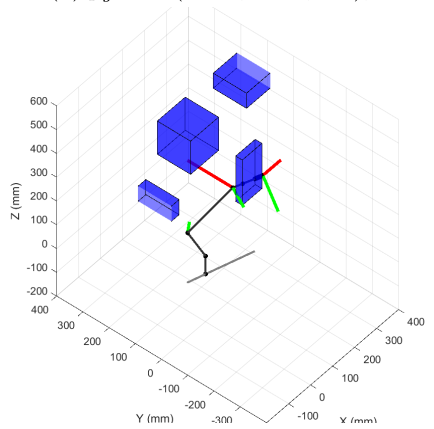
(a) $p_{goal} = (-150, 200, 350)$, success



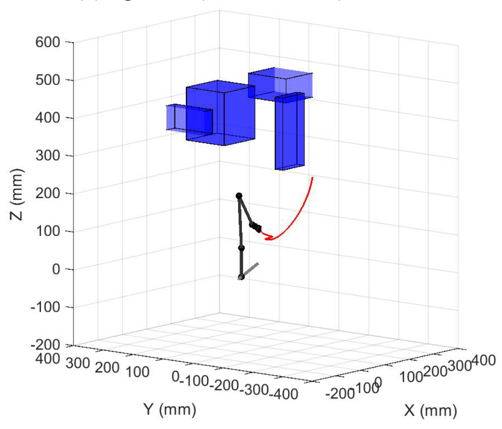
(b) $p_{goal} = (-150, -200, 350)$, success



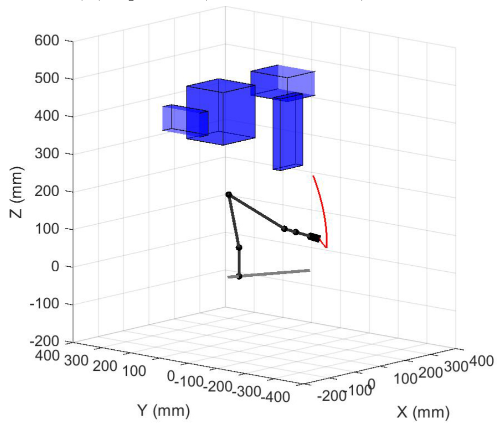
(c) $p_{goal} = (100, 200, 300)$, success



(d) $p_{goal} = (100, -200, 300)$, failed



(e) $p_{goal} = (200, 100, 100)$, success

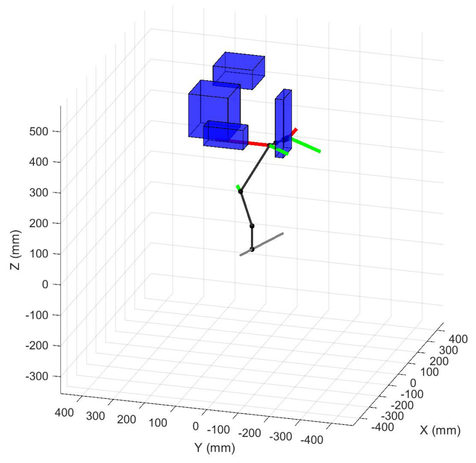


(f) $p_{goal} = (200, -100, 100)$, success

Figure 3: Reaching 6 different goal positions in map 4.

4.2 Local Minimum

In the previous section, trial 4 failed due to the robot is stuck at a local minimum. Now we enable the local minimum detection in our algorithm and take random walk as described in Section 3.2. Figure 4 shows the results after random walk. The robot could successfully go to the final position.


(a) Without random walk

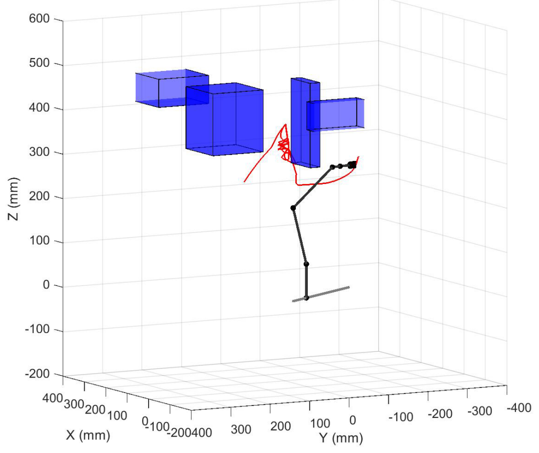

(b) With random walk

Figure 4: Escaping local minimum with random walk.

4.3 Effect of Models and Parameters

We use **example map** to investigate the effect of parameters since there's only one big obstacle in the workspace, this simple setting makes the results more intuitive.

4.3.1 Attractive Field Model

To evaluate the influence of model selection on the trajectory, we run simulation using 2 models: **conic well**, and **parabolic well**.

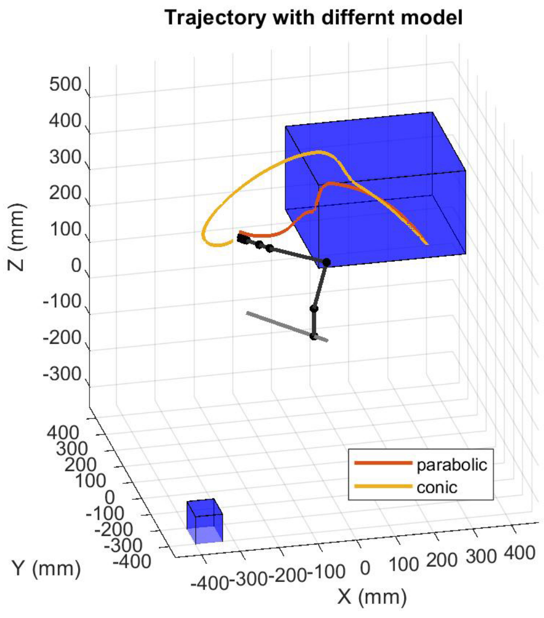


Figure 5: Parabolic well and conic well.

As shown in Fig 5, conic well and parabolic well have similar trajectory at the beginning. A possible explanation is that the starting point we choose is very close to the obstacle; at this staring point the repulsive force is more significant than attractive force and the repulsive force dominates the path at the beginning, regardless of the model used. However, the two path soon become quite different. This is because conic well attractive force doesn't have the scaling factor, so the attractive force is constant, which means the force exerted on the checkpoints are the same regardless of distance to goal. On the other hand, the parabolic well has a scaling factor, so the attractive force decreases as the robot moves toward its final position. Since the moving direction of the robot is determined by the net force on the robot, decreasing of the attractive force means the change of moving direction (assume repulsive force stays the same at the same position). Therefore, the trajectory in parabolic well model is different from that of conic well.

In our algorithm, the step size is always the same regardless of the model used. This means the choice of model only affects the direction of each step rather than the size of each step. Therefore, parabolic well is not guaranteed to have a better planning than conic well. According to the experiment we ran, the number of steps taken to get to the final position is 2363 in parabolic model, and 2197 in conic well model. Conic well model seems to work a little bit better than parabolic model in this case.

4.3.2 Effect of Parabolic Attractive Field Scaling Factor ζ

To evaluate the effect of attractive force scaling factor ζ, we run the simulation with 4 different values of ζ, 0.001, 0.1, 10, and 1000. As shown in Fig 6, when ζ increases, the trajectory gets shorter. The resulting trajectory from $\zeta = 0.001$ is very close to that of $\zeta = 0.1$, while the result from $\zeta = 10$ is very close to that of $\zeta = 1000$. This is consistent with our expectation, because when the attractive force gets larger, the robot is more likely to move along straight line toward the final position. As a result, the trajectory would be shorter when the attractive force is increased by a larger scaling factor.

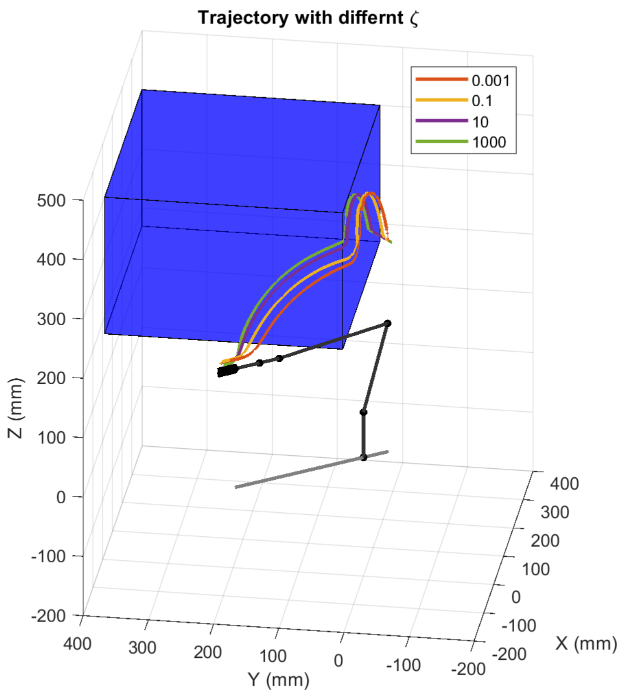


Figure 6: Effect of attractive field scaling factor.

4.3.3 Effect of Repulsive Field Scaling Factor η

To evaluate the effect of repulsive force scaling factor η, we run the simulation with 4 different values of η, 0, 10^3, 10^6, and 10^9. When η=0, there is no repulsive force and the robot would hit the obstacle. As η increases, the robot experiences larger repulsive force within obstacle influence ρ_0. As shown in Fig 7, as η increases, the trajectory gets longer. In another word, the time it takes from the starting to final position increases. This is expected because when η gets larger, the repulsive force gets larger and pushes the robot further away.

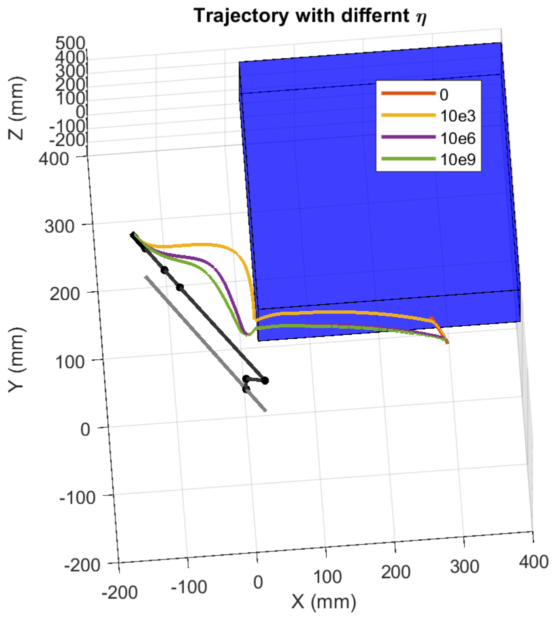


Figure 7: Effect of repulsive field scaling factor.

4.3.4 Effect of Influence Radius ρ_0

To evaluate the effect of obstacle influence radius ρ_0, we run the simulation with 4 different values of ρ_0, 5, 20, 50, and 200. As shown in Fig 8, the value of ρ_0 changes the trajectory significantly. For example, when ρ_0 is as small as 5 or 20, the robot gets to the final position by going toward negative y direction then position. In comparison, when the ρ_0 is 50 or 200, the robot is going positive y direction first and then go negative y direction. This is because when ρ_0 is large, the robot in this situation is always within the influence, experiencing repulsive force and getting pushed away from the obstacle. However, when the ρ_0 is small, the robot only experiences repulsive force when it's close to the starting point. After the robot leaves ρ_0, it is not experiencing any repulsive force and is then dragged toward negative y direction by attractive force.

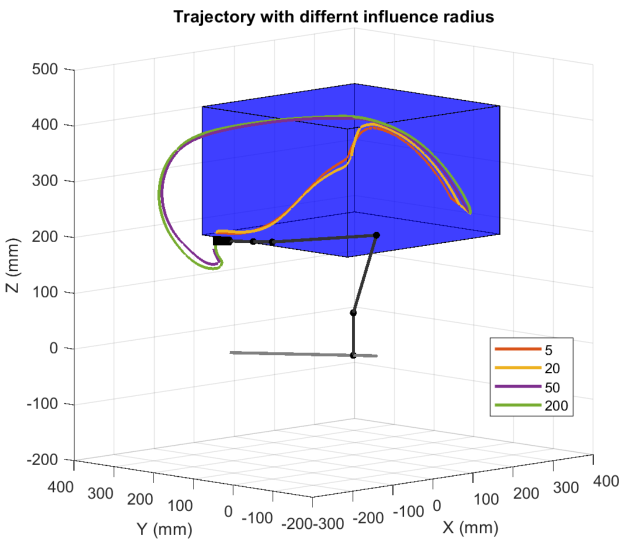


Figure 8: Effect of influence radius.

4.3.5 Effect of Step-size α

To evaluate the effect of step-size α, we run the simulation with 4 different values of α, 0.2, 0.02, 0.002, and 0.0002. The robot hit the obstacle when we take a 0.2 stepsize. However, when the stepsize is sufficiently small, its effect on trajectories are minor, as the other 3 trajectories are almost overlapping with each other as shown in Figure 9.

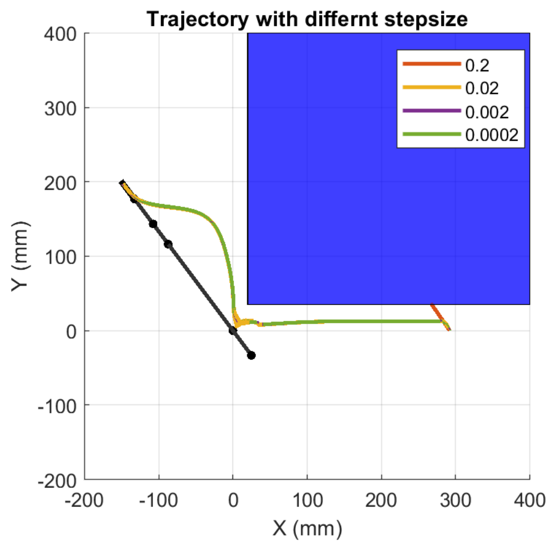


Figure 9: Effect of stepsize.

5 Analysis

5.1 Parameter Selection

In conclusion, there are 4 major parameters from our model that could affect the planner. These parameters are 1.attractive force scaling factor ζ, 2.repulsive force scaling factor η, 3.obstacle influence ρ_0, 4.step-size α. The increase of ζ would result in a shorter trajectory from starting to final position (shorter traveling time), while the increase of η would result in the opposite. The trade-off is that when ζ increases and η decreases, the robot is closer to the obstacle. This could be a problem in the physical environment, because the lynx robot has poor position accuracy due to gravity, and the position of obstacles might not be exact. Therefore, the parameters need to be tuned according to the physical set up. Besides, the change of ρ_0 would lead to totally different trajectories, depending on the start and final positions. In the absence of local minimum, the step-size has minor influence on the trajectories.

5.2 Limitations

During the experiment, while random walk method works well for most cases, some of the local minimum is vary hard to escape even if we take random walk. One of the example is illustrated in Figure 10, when the robot starts around the edge of the rectangular obstacle. This is also the failure we mentioned in Section 4.1. The robot wants to move from the left side of the vertical wall to the right side, while the wall is too high for the robot. The attractive and repulsive field is shown in Figure 11. If the robot takes a random walk, it will be dragged back to the local minimum. To solve this issue in further experiments, the scaling factor of both fields need to be carefully tuned and the step-size for random walk needs to be large enough (but not too large) to bypass the height of the wall.

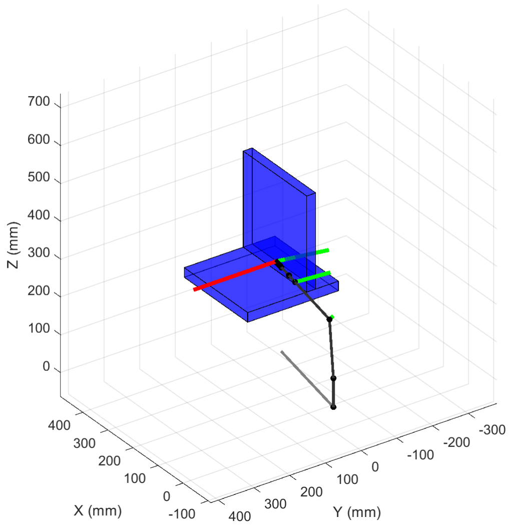
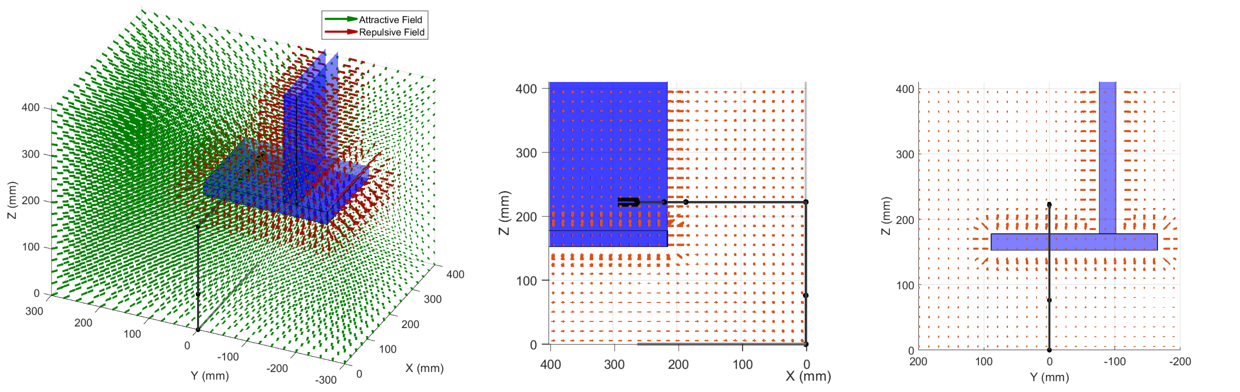


Figure 10: An Example of local minimum that is not easy to escape.



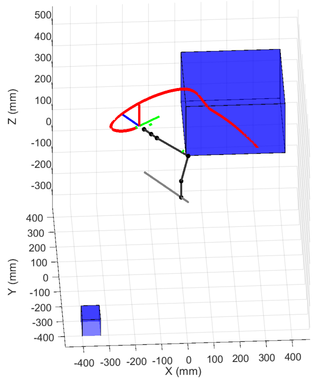
(a) Attractive Repulsive Filed (b) Potential field in x-z Plane (c) Potential field in y-z Plane

Figure 11: Potential filed of map 2 visualized.

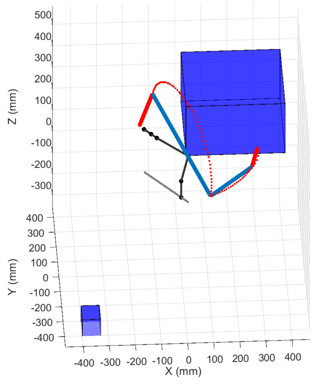
5.3 A Comparison with Probabilistic Roadmap (PRM)

We implemented Probabilistic Roadmap Method (PRM) in Lab 3. Compared to PRM, potential field method is easier to visualize, as the **potential field planner plans in workspace rather than in configuration space**. The easiness of visualization makes the debugging easier. **Besides, potential field method is more efficient in planning the path, especially when the path is short.** The major difference is that PRM requires searching in the C-space, which increases the computation. In comparison, potential field method only calculates positions that the robot swings by. Therefore, when the path is short, (start position close to final position), PRM would result in many unnecessary computation. **Another advantage of potential field method is that it enables online control**. In another word, the planner can plan the path as the robot moves, because the forces change as the robot moves, and these forces determine the next movement. However, the robot needs to have a clear roadmap in PRM before it can move. The capability of planning and moving real-time could be very important for applications such as autonomous driving vehicles.

As for advantage of PRM, **PRM doesn't have the local minimum problem**, and it doesn't have failure at local minimum as potential field method does. For example, to ensure the success of PRM at narrow corridor between obstacles, we need simply add number of sampling. However, for the potential field method, the problem becomes more complicated as we explained in section 5.2.



(a) Potential filed planner



(b) PRM planner

Figure 12: Results of two planning method.

Probabilistic road map method with 200 samples and 10 nearest neighbours connections graph took 18s to find a solution, while artificial potential field with stepsize=0.02 took less than 1s. The artificial potential field method has a significant time advantage over PRM. But we should notice that the most time-consuming step in PRM is sampling and graph building which only need to be done once. If we have prior knowledge about the environment, both algorithm can be used for online contorl.