2D Robot Path Planning

Yakup Can Karacaoğlu
Koç University Mechanichal Engineering
ykaracaoglu18@ku.edu.tr

I. Introduction

This paper will describe the methodology of potential field-based path planning of a 2D robot. The paper will include kinematic calculations of the robot, creation of configuration space and 3D potential field, and a gradient algorithm on the created potential field. The example robot has 2 links with 2 revolute joints, a limited working space and an obstacle in its working space, the details will be discussed below. There are 2 different initial configurations and 2 different final configurations for the solution, but creating a path between every solution is not possible. The local minima concept and stuck solutions will also be described.

II. Discussion

A. Forward and Inverse Kinematics

Start the discussion with the frame assignment on the given robot. In Figure 1 you can see the frame assignments on two joints. According to these assignments Θ_1 defined positive in counterclockwise and Θ_2 defined as clockwise. The kinematics will be calculated accordingly.

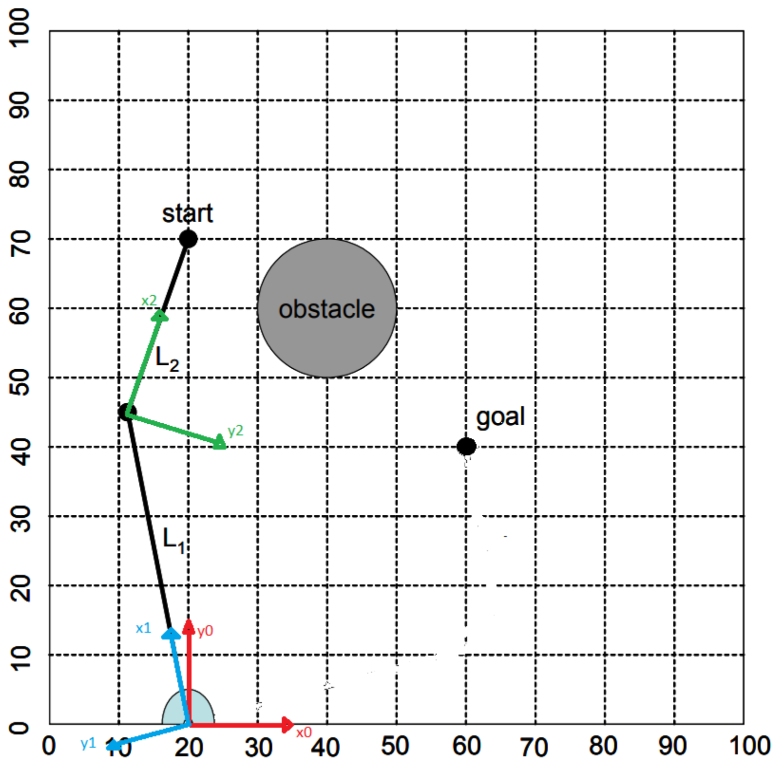

Figure 1 Frame assignments

After defining frames, the inverse kinematics should be calculated. The inverse calculation starts with the θ_2 angle. The angle represented in Figure 2 as β, and the θ_2 is represented as α. To calculate the β, cosine rule is utilized. From the Figure2, the equation 1 is obtained for r_1. Cosine of the β is calculated from the cosine rule, then the angle β is calculated by atan2 function in equation 3.

$$r_1 = L^2{}_1 + L^2{}_2 - 2L_1L_2\cos(180-\beta) = L^2{}_1 + L^2{}_2 + 2L_1L_2\cos(\beta) \qquad (1)$$

$$\cos(\beta) = \frac{r^2{}_1 - L^2{}_1 - L^2{}_2}{2L_1L_2} \qquad (2)$$

$$\beta = Atan2\left(\sqrt{1-\cos^2(\beta)}, \cos(\beta)\right) \qquad (3)$$

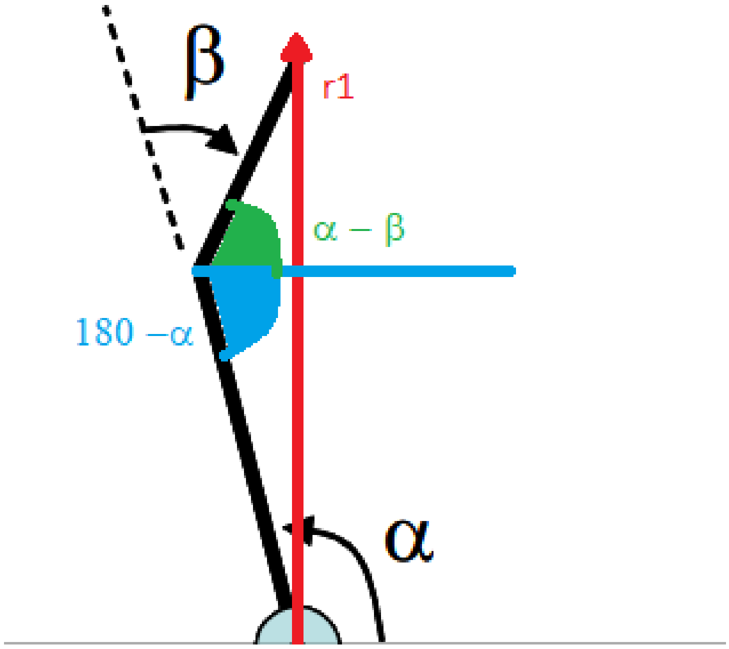

Figure 2 Inverse Kinematics for Second Angle

After β, now the angle α should be calculated. The geometry in Figure 3 is utilized for this purpose. From the figure the α is separated into two parts as α_1 and α_2. The calculation of α_1 is relatively easy, it can be read from the position of end effector directly such as in equation 4. For the calculation of α_2, the right triangle is created with red lines in Figure 3. The length of the red lines will be calculated by using the β, then they are used to calculate α_2. It can be observed in equation 5, and the result of the α in equation 6. This is the end of the inverse calculation.

$$\alpha_1 = Atan2\left(P_y, P_x - 20\right) \qquad (4)$$

$$\alpha_2 = Atan2(L_2 \sin(\beta), L_2 \cos(\beta) + L_1) \qquad (5)$$

$$\alpha = \alpha_1 + \alpha_2 = Atan2\left(P_y, P_x - 20\right) + Atan2(L_2 \sin(\beta), L_2 \cos(\beta) + L_1) \qquad (6)$$

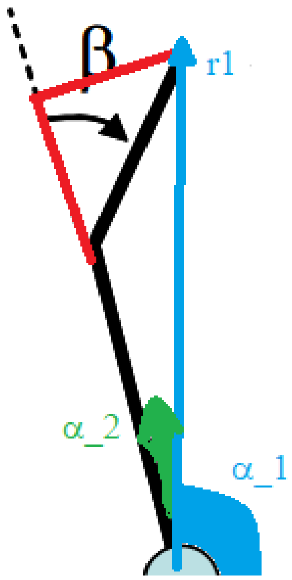

Figure 3 Inverse Kinematics for the First Angle

B. *Configuration Space*

Configuration space is calculated by using forward kinematics of the robot. The position of the joint 2 and tip point of the robot is calculated using forward kinematics, and the links will be represented as linspaces between base joint-joint 2 and joint 2- tip point. Also, the Θ_1 and Θ_2 will be discretized in enough resolution. The map in Figure 2 is discretized as Θ_2 is 2400 points and Θ_1 is 1200 points. Then a grid panel is created in the same resolution such as 2400x1200. Lastly, for every $\Theta_1 - \Theta_2$ pairs, every point on links is checked if they are colliding with obstacle or going out of the working space. If the collision is occurred, the Θ pairs in grid panel is marked with 1. The clean points marked as 0. In visualization the 1s are blue and 0s are white in Figure 4. In robotics logic, the white regions represent movement space for the robot and blue regions are the obstacles. In space there are also 4 marked points, they stand for initial and final configurations of the robot for desired movement. The blue '+'s are initial configurations, and the others are final configurations.

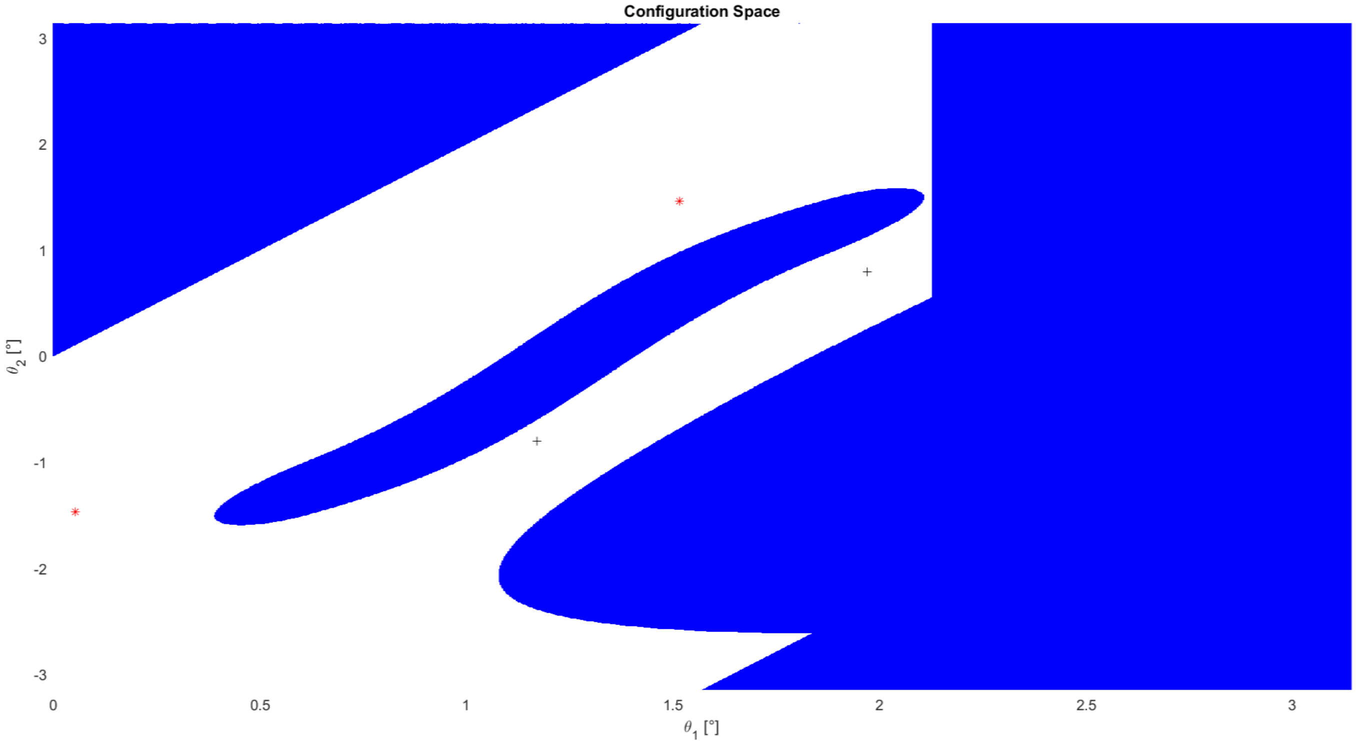

Figure 4 Configuration Space (2400x1200)

C. *Potential Field*

As introduced, the path planning algorithm represented in this paper is the potential field approach. So, a potential field should be created. Potential field has two components such as attractive potential and repulsive potential.

Attractive potential should be minimum at the goal point and should increase parallel with the distance from the goal. The formula of the potential field is equation 7, and in the code, it is calculated by checking the closest nonzero pixel. The goal point is marked as 1 and the rest of the attractive potential array is directly proportional to square of this distances. The attractive potential values are also modified with a constant coefficient $\frac{1}{2}$ and ζ.

$$U(q) = \frac{1}{2}\zeta \, ||q - q_{goal}||^2 \qquad (7)$$

$$U_{rep}(q) = \frac{1}{2}\eta\left(\frac{1}{\rho(q)} - \frac{1}{\rho_0}\right)^2 \; if\ \rho(q) \leq \rho_0 \qquad (8)$$
$$U_{rep}(q) = 0 \; if\ \rho(q) > \rho_0$$

Repulsive potential is also calculated with a similar approach, the equation is given in equation 2. In this time the repulsive effect is inversely proportional to distance from the repulsion source, and it is zero after a limit. In the code, it is mentioned that the configuration space represented by zeros, and obstacles are represented as 1s. For every zero pixel in that grid, distance from closest nonzero pixel calculated [1]. Then it is used to calculate repulsive potentials.

In the end we have two different potentials and the addition of them will give us the potential field. An important side note here is that the coefficients ζ and η should be selected to match attractive and potential values close to each other. If they are calculated so far from each other one of them could lose its effect and the algorithm will collapse. You can see the potential field in 3D grid in Figure 5.

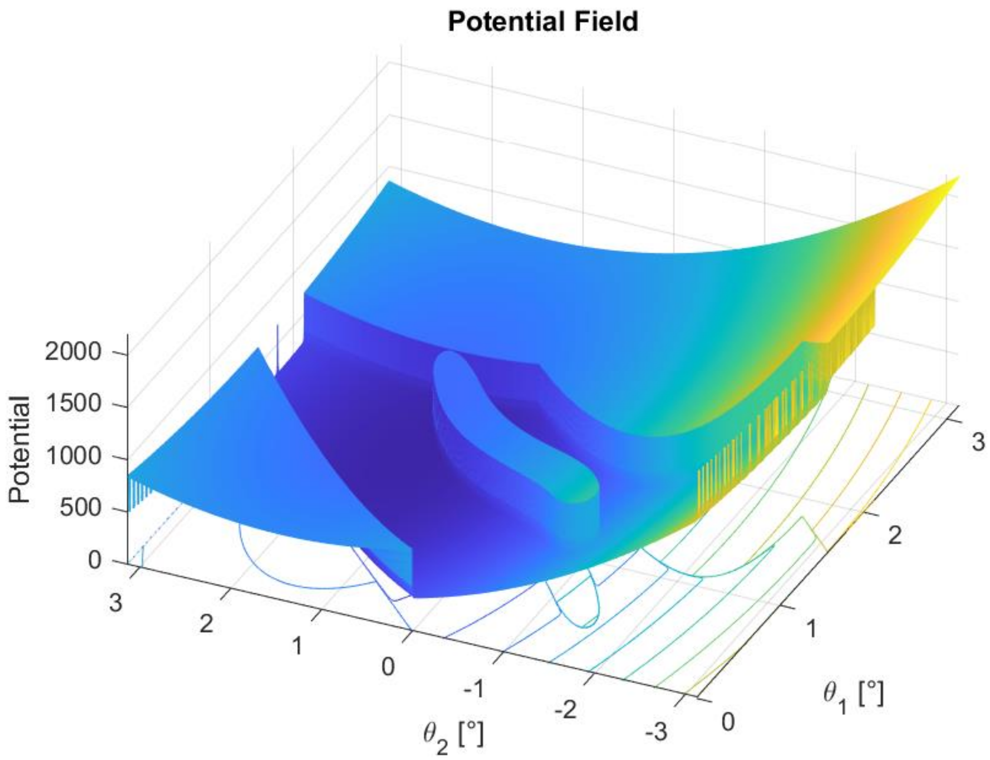

Figure 5 Potential Field

D. Graient Algorithm

The potential field observed, now the next step is moving on this field. To achieve this, gradients will be used. Every element in q1-q2 plane has gradient that changes the value of the potential. Using these gradient values, we can move towards the minimum potential. In other words, the gradient algorithms will represent the driving force of the robot. The important point is the negative of the gradient should be used to be able to move smaller potential. In the code, the gradients are calculated by using the built-in gradient function of the MATLAB, but to be able to move a grid the gradient values are normalized. In the scenario of not normalizing the gradient values, it would be hard moving an integer indexed grid. So, the gradient values are normalized, and the step size is made constant.

The algorithm works until reaches the end point with enough tolerance, or until 1000 iterations. The successful iterations in Figure 4 and Figure 5 can be observed for different configurations, but there is a problem with the gradient algorithm. The local minimum problem of the gradient algorithm can cause to cut the path before reaching the goal. As observed in Figure 6 and Figure 7 the algorithm stops before reaching the goal. It stops because of the number of iterations, but even if the algorithm continues to work it never reaches the end point. As suggested in assignment description, the solution can be reached by different base location or link lengths.

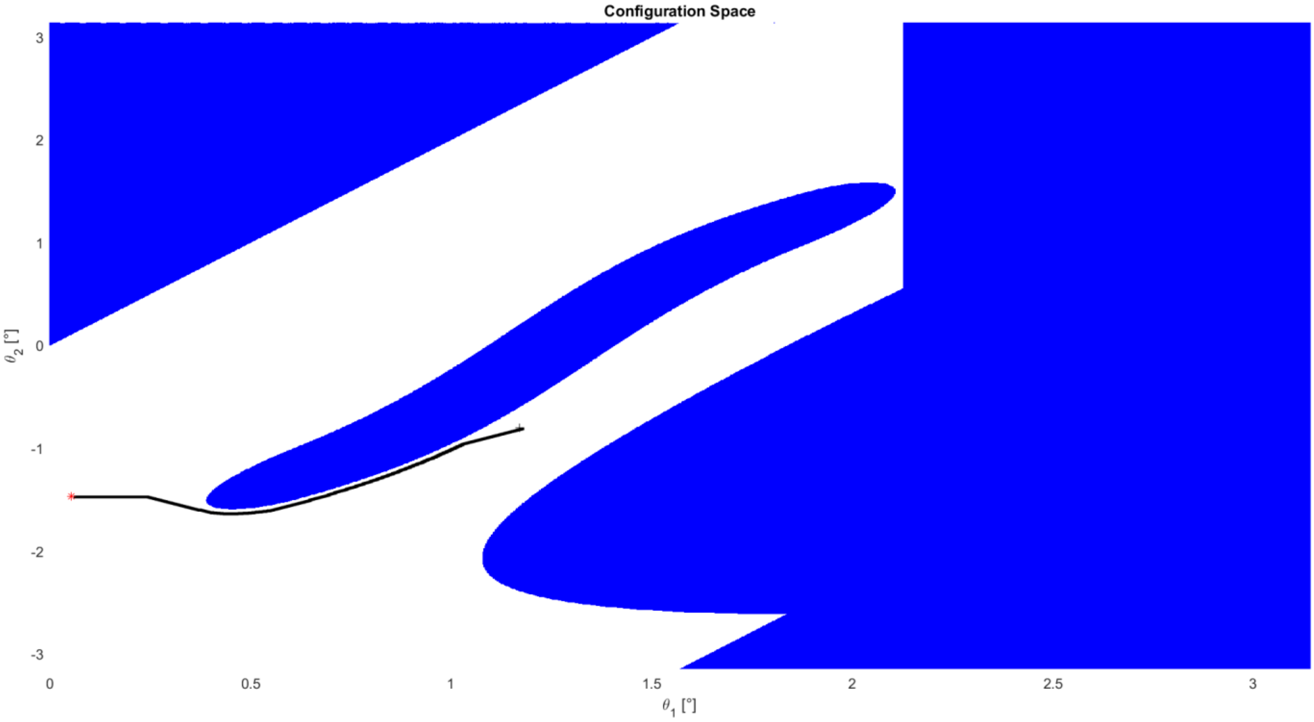

Figure 6 Initial 2 – Final 2

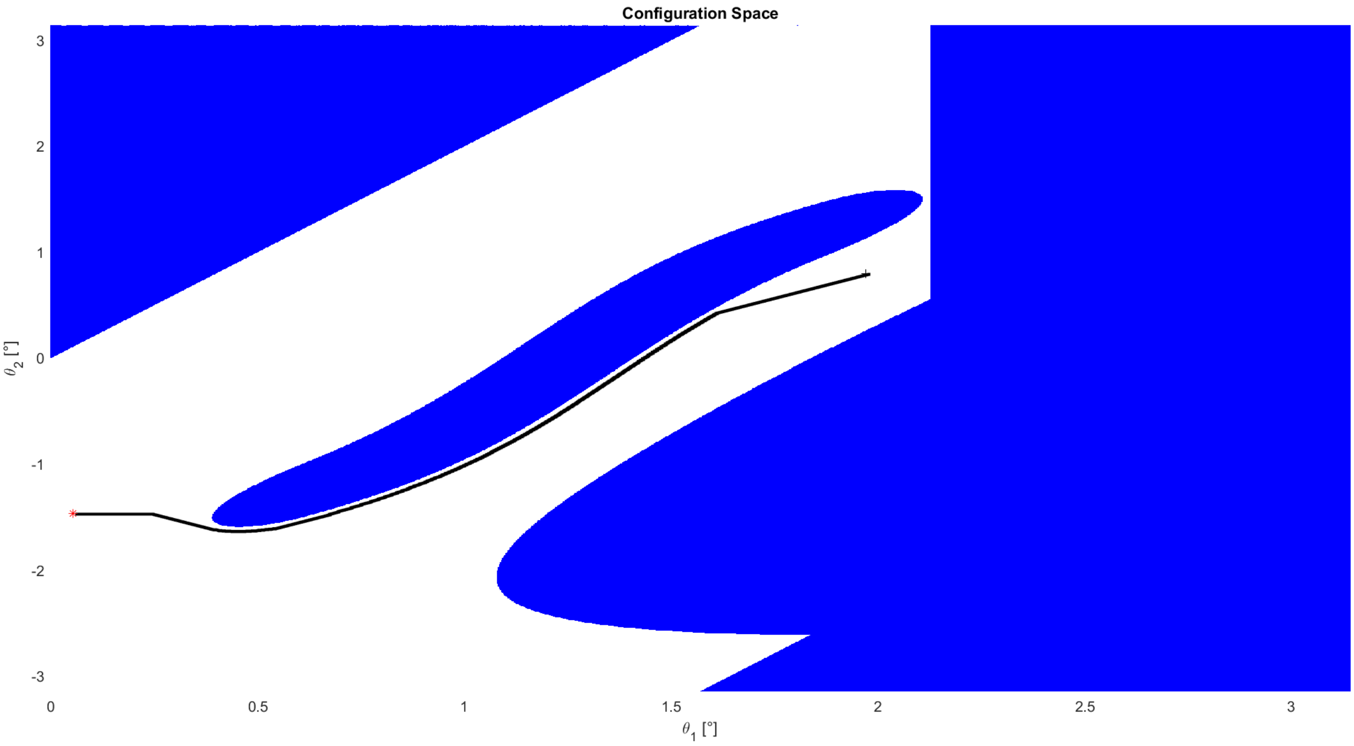

Figure 7 Initial 1 - Final 2

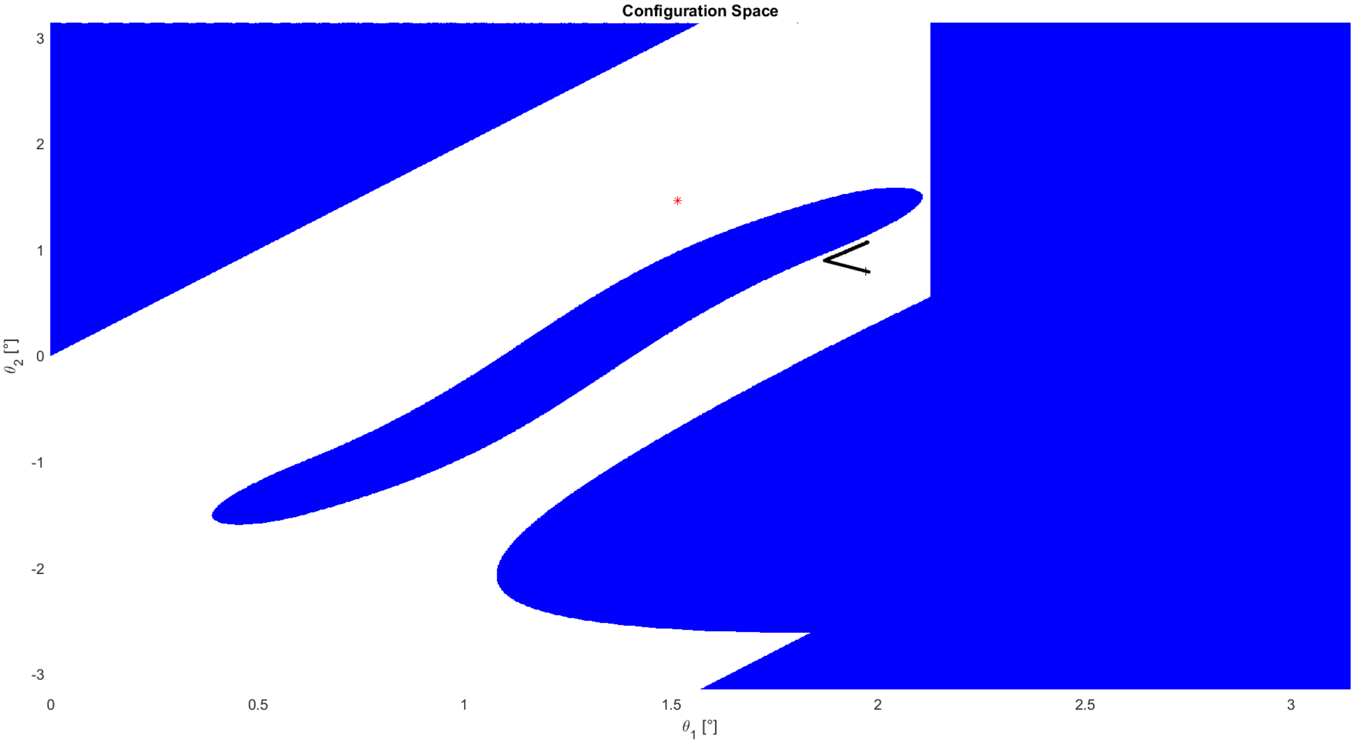

Figure 8 Initial 1 - Final 1

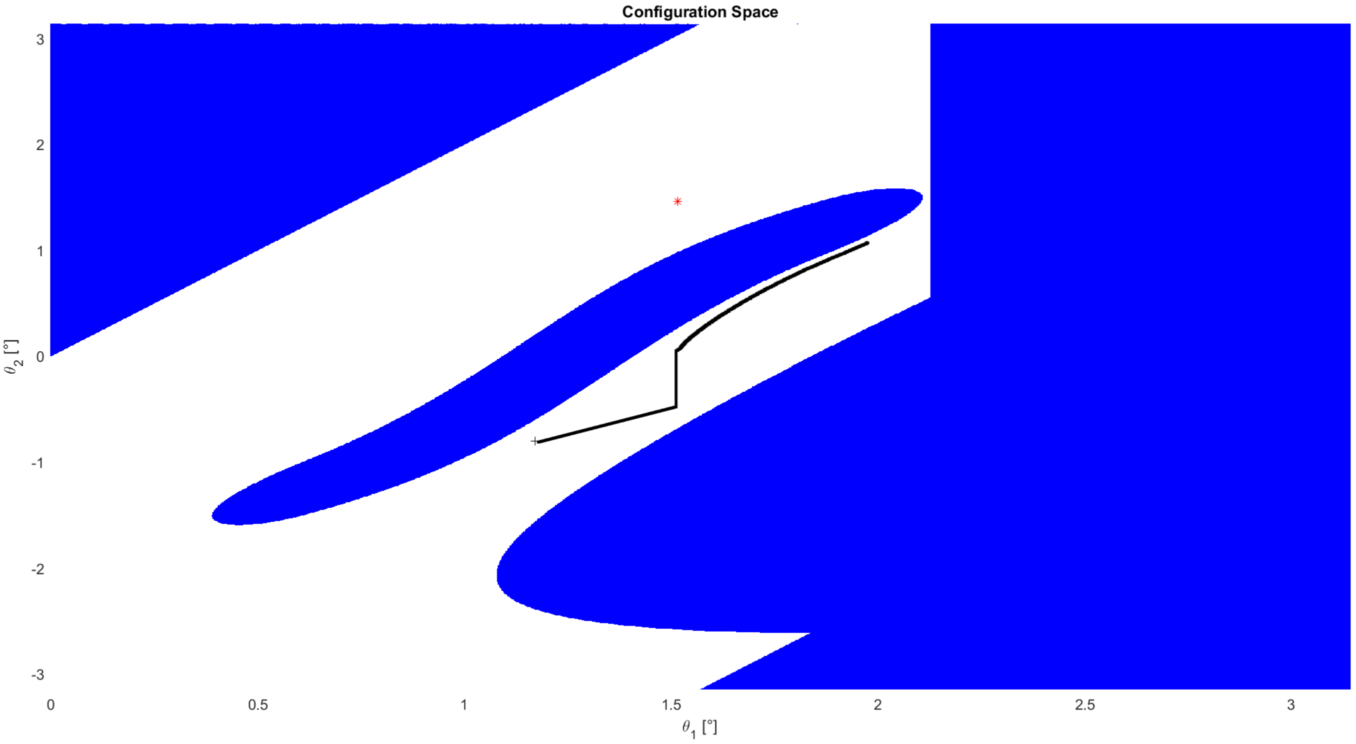

Figure 9 Initial 2 - Final 1

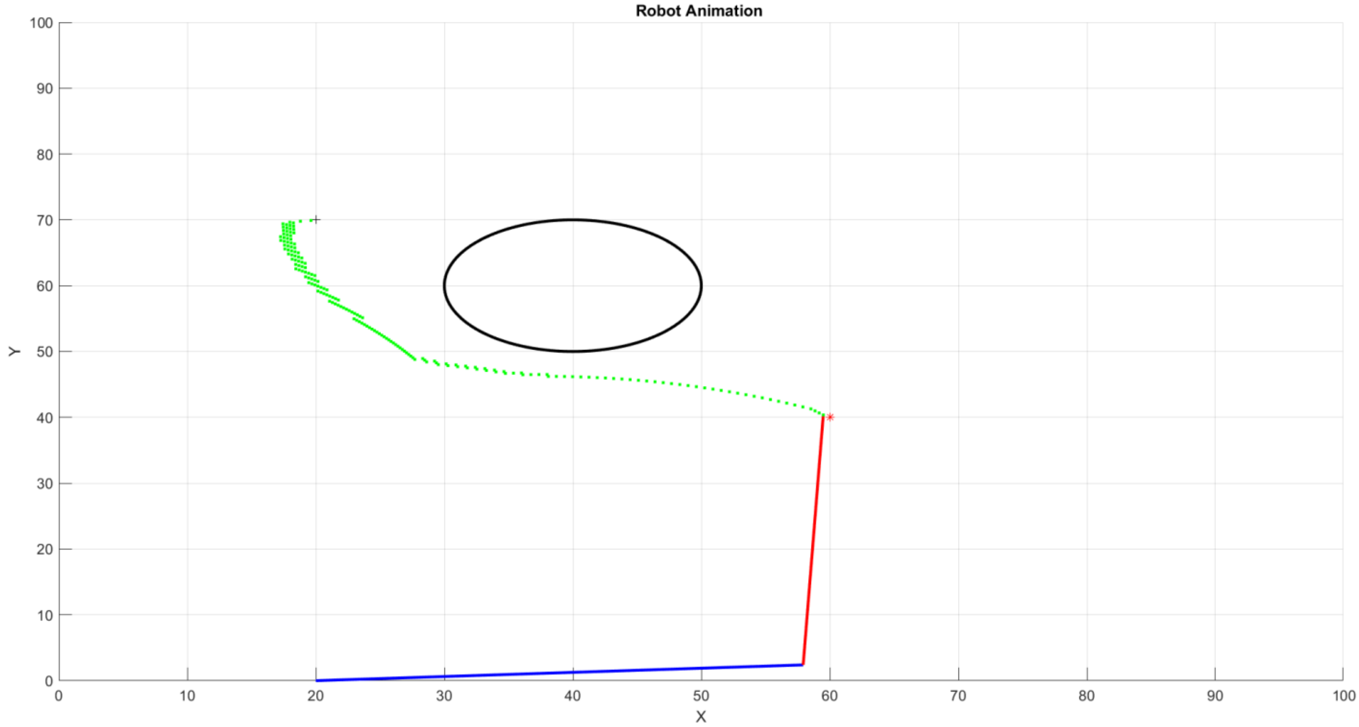

Figure 10 Path in Cartesian Initial 2- Final2

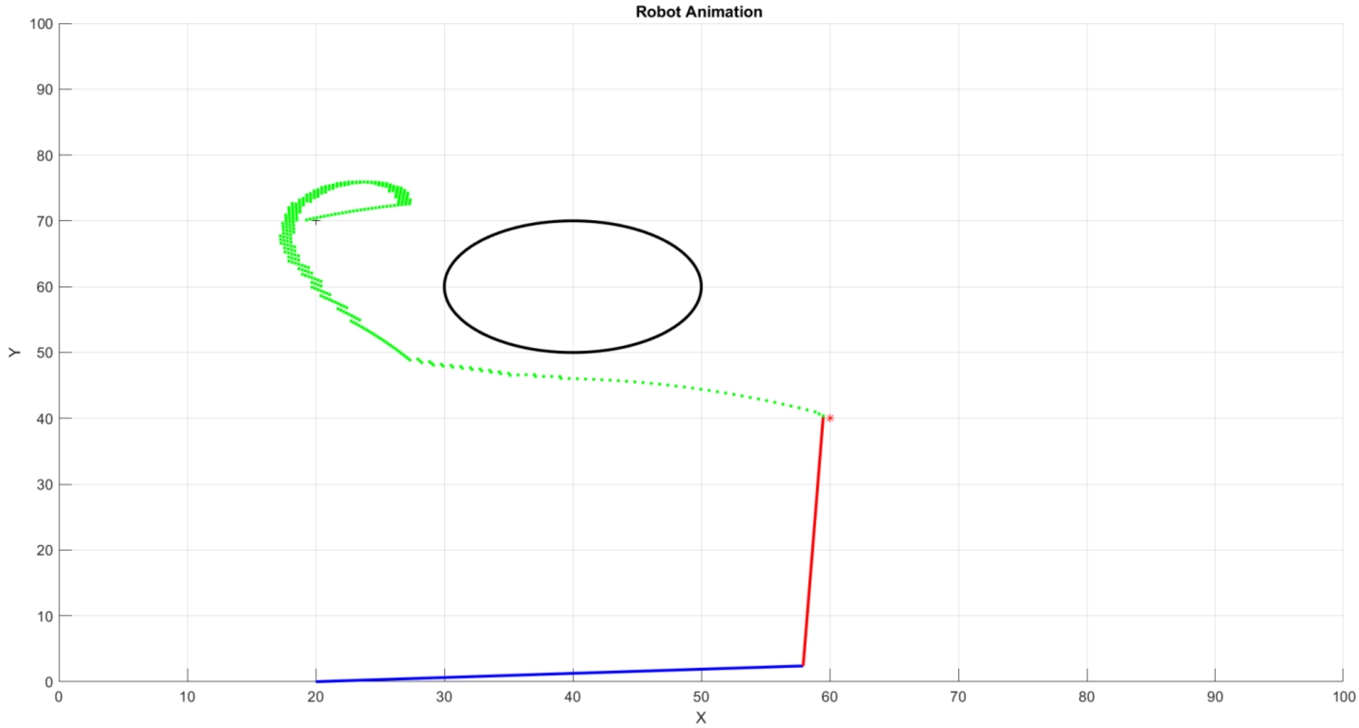

Figure 11 Path in Cartesian Initial 1 - Final 2

III. Conclusion

To sum up, the implementation of potential field approach with gradient algorithm is represented in this paper. The implementation is done for two initial and two final configurations. The algorithm worked in angle space of the robot and achieved two solutions. Also, the negative side of the gradient algorithm which is local minimum is displayed in two of the paths. The successful solutions are also represented both in cartesian and angle space. This paper showed that path planning of a 2 links planner robot can have its path by using potential field approach. Even if the local minimum concept exists, the path planning can be done by using potential field.

IV. References

[1] "BW." *Distance Transform of Binary Image - MATLAB*, www.mathworks.com/help/images/ref/bwdist.html. Accessed 19 May 2023.