Artificial Potential Fields

AI and AUTONOMOUS SYSTEMS

Luke Keating, Iñaki Román

25 November 2024

Abstract

In this laboratory, we designed and implemented autonomous navigation strategies for an integrator and a unicycle robot for navigation in a two-dimensional space using artificial potential fields. The overall objective of the laboratory is to guide the robot towards a predefined goal position while avoiding obstacles in the environment. This approach combines attractive and repulsive fields in order to find the path to the goal while avoiding the obstacle in the way. The potential field is defined mathematically, with parameters α and β that modulate the influence of the attractive and repulse forces. The trajectory of the robot is computed using the gradient of the potential field and also depends on the type of robot. The gradient ensures a smooth and short path to the goal while dynamically avoiding the obstacles. The implementation is validated through MATLAB simulations, where the vector fields and the robot behaviour are visualized and the values of α and β are tuned. Finally, a control strategy is applied to the robot so it can follow the desired optimal path given by the potential fields.

Question 1: Artificial potential fields strategy

1.1 - Introduction and navigation strategy based on artificial potential fields

Obstacle avoidance is a fundamental topic in autonomous vehicles, enabling robots to move independently within the environment while avoiding obstacles and reaching its desired goal is crucial. The achievement of this navigation capability is not trivial in autonomous navigation and requires the implementation of algorithms that can adjust dynamically the robot's path to the goal depending on the obstacles in the original trajectory. Among various strategies, artificial potential fields (APF) stand out because of their simplicity, finding the optimal route and obstacle avoidance capabilities while guiding the robot in two-dimensional space.

The APF method models the environment of the robot as a potential field of attractive and repulsive forces: where the attractive forces will guide the robot to its goal, while the repulsive forces will repel the robot from the obstacles. The overall goal of this lab is to design a control strategy where the robot can reach the goal in the most optimal way, while not crashing into an obstacle. This report explores the implementation of an APF algorithm for a robot in 2D space with an obstacle in the middle of the trajectory to the goal, examining how the potential fields can be configured and tune to optimize the robot's path.

In this exercise, the parameters of the APF will be tune to see the different configurations and balanced between the attractive and repulsive forces acting on the map, ensuring a smooth navigation. Different formulas for the attractive and repulsive potentials are evaluated, and their gradients analyse to understand the robot's behaviour on shorter or longer paths.

In the laboratory, two type of robots are explored: honolomic and non-honolomic robots, which are mainly differentiated by their freedom of movement. It is discussed how the differences on the properties of the

robots affect the navigation control strategy.

The overall goal of this computer lab is to explore and evaluate the practical implementations of artificial potential field for autonomous navigation and provide insights on how this methodology can help a smooth navigation of the environment. Also, the laboratory discusses in depth the kinematic properties of different robot models and how it affects the navigation and control strategy.

1.2 - Attractive Potentials in Artificial Potential Fields

These functions can be considered attractive, as when the position (p) is closer to the goal position (p_{goal}) the overall term (U_{att}) will get closer and closer to zero. The potential field therefore gives a clear direction for the gradient descent algorithm to follow to get to the minimum at p_{goal}.

The difference between the two functions is that one is squared whereas the other is not. The squared version will give higher values in the potential field when the position is far from the goal and smaller values when the position is very close to the goal, than the unsquared version.

These larger values pull stronger towards the goal and the reduced potential very near the goal prevents oscillation around the goal point see figure 1. For these reasons the squared version should be chosen for navigation.

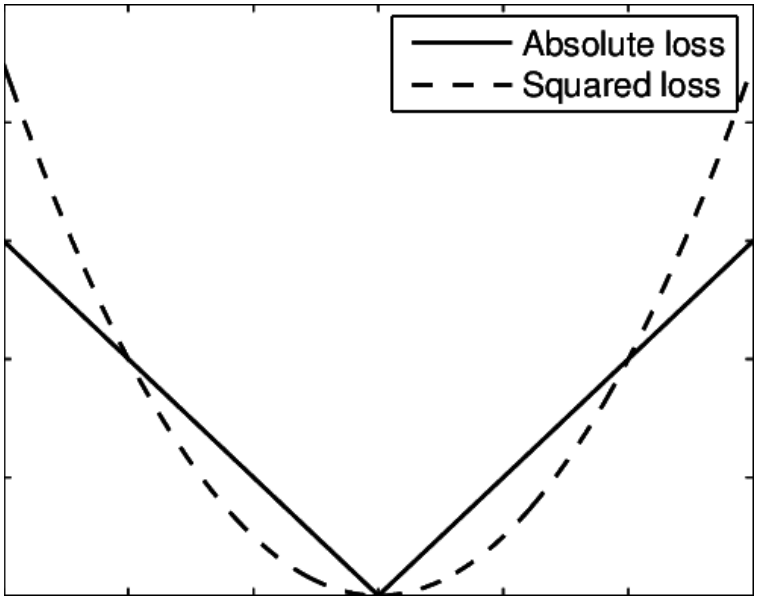

Figure 1: Absolute vs squared attractive potentials

1.3 - Repulsive potential field

The equation is suitable, as the potential will grow towards infinity as the position gets closer to the position of the obstacle. The direction of descent therefore is away from the obstacle, hence this equation is suitable for a repulsive potential for navigation and is given by 1:

$$U_{\text{rep}}(\mathbf{p}) = \frac{1}{\|\mathbf{p} - \mathbf{p}_{\text{obs}}\|} \tag{1}$$

1.4 - Potential field equation

The proposed potential field is show in equation 2, where $p = [x \quad y]^T$.

$$U(p) = \alpha U_{att}(p, p_{goal}) + \beta U_{rep}(p, p_{obs}) \tag{2}$$

Where,

$$\begin{aligned} U_{att}(p) &= \|p - p_{goal}\|^2 = (x - x_{goal})^2 + (y - y_{goal})^2 \\ U_{rep}(p) &= \frac{1}{\|p - p_{obs}\|} = \frac{1}{\sqrt{(x - x_{obs})^2 + (y - y_{obs})^2}} \end{aligned} \tag{3}$$

Therefore, using equations in 3, expand the equations and take the partial derivative with respect to x and y, the term ∇U_{att} can be found, shown in equation 4,

$$\begin{aligned} U_{att}(p) &= x^2 - 2x x_{goal} - x_{goal}^2 + (y - y_{goal})^2 \\ \frac{\partial U_{att}}{\partial x} &= 2x - 2x_{goal} = 2(x - x_{goal}) \\ \frac{\partial U_{att}}{\partial y} &= 2(y - y_{goal}) \end{aligned} \tag{4}$$

Therefore,

$$\nabla U_{att} = 2(p - p_{goal})$$

Then similarly for the repulsive potential partial derivatives are taken but using the chain rule,

$$\begin{aligned} U_{rep}(p) &= \frac{1}{\sqrt{(x - x_{obs})^2 + (y - y_{obs})^2}} = ((x - x_{obs})^2 + (y - y_{obs})^2)^{-\frac{1}{2}} \\ g(x) &= (x - x_{obs})^2 + (y - y_{obs})^2 \\ f(x) &= (g(x))^{-\frac{1}{2}} \\ \frac{\partial U_{rep}}{\partial x} &= f'(x) * g'(x) \\ &= -\frac{1}{2\left(\sqrt{(x - x_{obs})^2 + (y - y_{obs})^2}\right)^3} * 2(x - x_{goal}) \\ \frac{\partial U_{rep}}{\partial y} &= -\frac{1}{2\left(\sqrt{(x - x_{obs})^2 + (y - y_{obs})^2}\right)^3} * 2(y - y_{goal}) \end{aligned} \tag{5}$$

Therefore,

$$\nabla U_{rep} = -\frac{p - p_{obs}}{\|p - p_{obs}\|^3}$$

Therefore, the gradient of the whole potential field is shown in equation 6.

$$\begin{aligned}\nabla U(p) &= \alpha \nabla U_{att}(p, p_{goal}) + \beta \nabla U_{rep}(p, p_{obs}) \\ &= \alpha 2(p - p_{goal}) - \beta \frac{p - p_{obs}}{\|p - p_{obs}\|^3}\end{aligned} \tag{6}$$

1.5

A sketch of the proposed scenario is shown in figure 2. Where p_{obs} is the center of the circular obstacle, R_{obs} is the obstacles radius, p^*_{obs} the closest point on the circle to the robot position p and d_{obs} is the distance of p to p^*_{obs}.

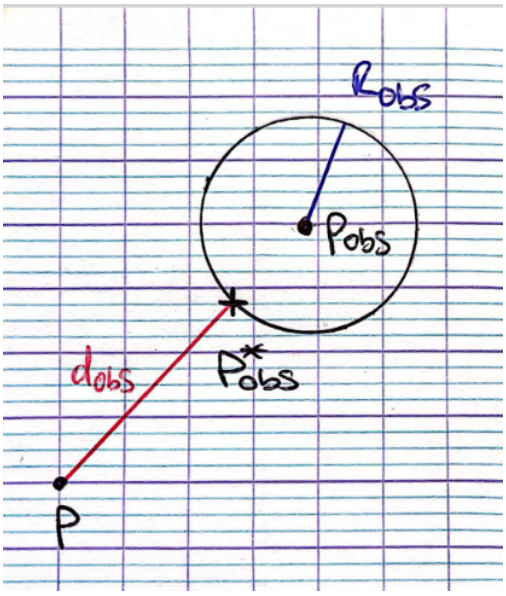

Figure 2: Drawing of robot to circular obstacle

The distance vector d_{obs} is computed as follows.

$$\begin{aligned}p^*_{obs} &= p_{obs} + R\frac{(p - p_{obs})}{\|p - p_{obs}\|} \\ d_{obs} &= p - p^*_{obs} = p - \left(p_{obs} + R\frac{(p - p_{obs})}{\|p - p_{obs}\|}\right) \\ d_{obs} &= p - p_{obs} - R\frac{(p - p_{obs})}{\|p - p_{obs}\|}\end{aligned} \tag{7}$$

1.6

Building on equation 5 derived earlier, the gradient function can then be presented as 8

$$\nabla U = \alpha 2(p - p_{goal}) - \beta \frac{d_{obs}}{\|d_{obs}\|^3} \tag{8}$$

Question 2: Autonomous navigation for a single integrator robot

2.1 - Robot integrator dynamics

After having the Potentials fields derived and implemented, we can develop a simple integrator robot that can move in any direction and control its velocity directly, the dynamics of the robot is expressed as :

$$\dot{x} = u_1 \tag{9}$$

$$\dot{y} = u_2 \tag{10}$$

where the input $U = [u_1, u_2]^T$ and $X_{state} = [x, y]^T$ is the robot's state. This robot follows a simple integrator dynamics, thus the derivative of the robot's state is directly equal to the control input $u(t)$.

2.2 - Discretization of Robot Dynamics for Artificial Potential Field Navigation

In order to analyse the artificial potential field in Matlab, we implement the navigation in discrete time. This requires updating the robot's position on each step k according to its dynamic model.
We can discretize the dynamics of the robot as:

$$\dot{X}(t) \approx \frac{X(k+1) - X(k)}{\mathrm{dt}} \tag{11}$$

Thus, the updated position at step $k + 1$ is expressed as:

$$X(k+1) = X(k) + \dot{X}(k) \cdot \mathrm{dt} \tag{12}$$

Where, $\dot{X}(k)$ is the derivative of the state, calculated using the function `robot_dynamics`. We have the control input $-\nabla U$ that represents the negative gradient of the potential field in x and y directions determines the velocity of the robot at each time step.

It can be implemented in Matlab as follows:

```
% Compute the velocity using robot_dynamics
f = robot_dynamics(X_state(:,k-1), u(:,k));


% Update the robot's state
X_state(:,k) = X_state(:,k-1) + f * dt;
```

2.3 - `draw_field.m`

The function `draw_field.m` help us visualize the artificial potential field by plotting the normalized gradient vector of the field:

- **Grid creation:** It creates the grid points within the limits specified.
- **Gradient computation:** In each point of the grid, the attractive and repulsive forces from the obstacles are calculated. This is based on α and $\$\beta$.
- **Repulsive contributions:** If a second obstacle (`p_obs2`), a repulsive force will be added to the field.
- **Normalization:** Lets us compute the gradient vectors and normalized them to ensure a consistent scaling in the field.
- **Visualization:** The normalized vector will be plotted in order to see the potential field.

This visualization corresponds directly to the artificial potential field given in the Question 1, the attractive forces point towards the goal, while the repulsive one prevents a collision.

2.4 - Design of the Control Law for the Single Integrator Robot

The control law for the robot is based on the gradient of the artificial potential field:

$$u = -\nabla U(\mathbf{p}), \tag{13}$$

where:

- $\nabla U_{\text{attr}} = 2(\mathbf{p} - \mathbf{p}_{\text{goal}})$: Attractive gradient.
- $\nabla U_{\text{rep}} = -\frac{\mathbf{p}-\mathbf{p}_{\text{obs}}}{\|\mathbf{p}-\mathbf{p}_{\text{obs}}\|^3}$: Repulsive gradient

The overall gradient is given by:

$$\nabla U = \alpha \nabla U_{\text{attr}} + \beta \nabla U_{\text{rep}}, \tag{14}$$

with $\alpha = 1$ and $\beta = 1$.

- Initial position: $\mathbf{X}_{\text{state}} = [2; 1]$.
- Goal position: $\mathbf{p}_{\text{goal}} = [6; 6]$.
- Obstacle position: $\mathbf{p}_{\text{obs}} = [4; 4]$.

The robot starts at $[2; 1]$, avoids the obstacle at $[4; 4]$, and reaches the goal at $[6; 6]$. The trajectory, along with the positions of the goal and obstacle, is plotted to illustrate the behaivour.
The robot start on the position X_{state}, avoids the obstacle in P_{obs} and reach the goal in P_{goal}. The trajectory is simulated and gives the next result:

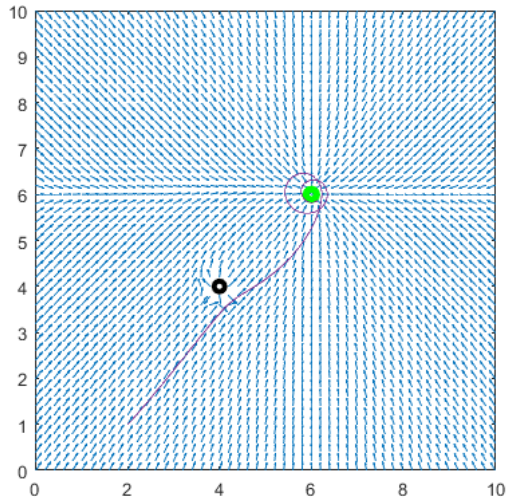

Figure 3: Integrator robot trajectory to the goal while avoiding the obstacle.

After the simulation, we can conclude that the robot reaches the goal. Nonetheless, it has some error on trajectory when it reaches the goal, we can tribute this behaviour is related to the attractive equation where the robot is closer to the goal it slows down and also the error on reaching the goal is small thus the robot takes more steps to get there.

Question 3: Autonomous Navigation for a Non-Holonomic Robot

For this next part, we will model a non-honolomic robot as a unicycle with the following eaquations of motion:

$$\dot{x} = v\cos\theta, \tag{15}$$

$$\dot{y} = v\sin\theta, \tag{16}$$

$$\dot{v} = u_1, \tag{17}$$

$$\dot{\theta} = u_2, \tag{18}$$

where:

- v: Robot's speed.
- θ: Robot's heading angle.
- u_1, u_2: Control inputs for the robot.

The navigation approach for this type of robot differs slightly from the honolomic robot, for the unicycle robot it involves computing the control inputs (u_1, u_2) given the robot's position and desired trajectory. This more complicated architecture will enable us to design control architectures for autonomous vehicles that their degrees of freedom are dependent from each other.

3.1 - State-Space Representation and Dynamics of the Unicycle Robot

The state space of the unicycle robot is defined as:

$$\mathbf{X}_{\text{state}} = \begin{bmatrix} x \\ y \\ v \\ \theta \end{bmatrix} \tag{19}$$

where:

- x, y: Position of the robot.
- v: Speed of the robot.
- θ: Heading angle of the robot.

In order to implement the dynamics of the robot shown, the function `robot_dynamics.m` is modified to compute state variables $(\dot{x}, \dot{y}, \dot{v}, \dot{\theta})$ given the control inputs u_1 and u_2. Thus, we can update the robot state in discrete time:

$$\mathbf{X}_{\text{state}}(k+1) = \mathbf{X}_{\text{state}}(k) + \dot{\mathbf{X}}_{\text{state}} \cdot \text{dt}. \tag{20}$$

Updating the robot dynamics, allow it to navigate towards the desired goal and respect the non-holonomic constrains.

3.2 - Discrete-Time Navigation for a Non-Holonomic Robot

To design the behaviour of the robot in the potential field, the robot's dynamics are discretized. Its state space is defined as:

$$\mathbf{X}_{\text{state}} = \begin{bmatrix} x \\ y \\ v \\ \theta \end{bmatrix}, \tag{21}$$

where (x, y) is the robot's position, v its velocity, and θ the heading angle. Using the unicycle dynamics the time derivative of the state is:

$$\dot{\mathbf{X}} = \begin{bmatrix} v\cos\theta \\ v\sin\theta \\ u_1 \\ u_2 \end{bmatrix}. \tag{22}$$

And, the state update is done as:

$$\mathbf{X}(k+1) = \mathbf{X}(k) + \dot{\mathbf{X}} \cdot \text{dt}, \tag{23}$$

where u_1 and u_2 are control inputs:

$$u_1 = -K_1(v - v_{\text{ref}}), \tag{24}$$

$$u_2 = -K_2 \arctan\left(\tan\left(\frac{\theta - \theta_{\text{ref}}}{2}\right)\right). \tag{25}$$

This design of the control law ensures the robot's velocity is correct and tracks the reference correctly, which is derived from the potential field:

- Initial state: $\mathbf{X}_{\text{state}} = [2; 1; 1; \pi/4]$,

- Goal position: $\mathbf{p}_{\text{goal}} = [6; 6]$,
- Obstacles: $[4; 4]$ and $[6; 4]$,
- Sampling time: dt $= 0.01$, and total iterations $T = 300$.

3.3 - Velocity and angle reference based on the potential field

In order to guide the robot towards the goal while avoiding the obstacle, the desired velocity is derived from the potential field. The total gradient of the potential is:

$$\nabla U = \alpha \nabla U_{\text{attr}} + \beta \nabla U_{\text{rep}}, \tag{26}$$

where:

- $\nabla U_{\text{attr}} = 2(\mathbf{p} - \mathbf{p}_{\text{goal}})$: Attractive gradient towards the goal.
- $\nabla U_{\text{rep}} = -\frac{\mathbf{p}-\mathbf{p}_{\text{obs}}}{\|\mathbf{p}-\mathbf{p}_{\text{obs}}\|^3}$: Repulsive gradient from the obstacle.

Desired velocity:

$$v_{\text{ref}} = \|\nabla U\|, \tag{27}$$

$$\theta_{\text{ref}} = \arctan\left(\frac{\nabla U_y}{\nabla U_x}\right), \tag{28}$$

where v_{ref} Desired velocity, and θ_{ref} desired heading angle.
The proportional control laws are given as:

$$u_1 = -K_1(v - v_{\text{ref}}), \tag{29}$$

$$u_2 = -K_2 \arctan\left(\tan\left(\frac{\theta - \theta_{\text{ref}}}{2}\right)\right), \tag{30}$$

where u_1 controls the linear velocity, and u_2 controls the orientation rate.
The robot's movement is simulated by:

- Obstacle position: $\mathbf{p}_{\text{obs}} = [4; 4]$,
- Goal position: $\mathbf{p}_{\text{goal}} = [6; 6]$,
- Initial state: $\mathbf{X}_{\text{state}} = [2; 1; 1; \pi/4]$,
- Potential field parameters: $\alpha = 1, \beta = 1$,
- Sampling time: dt $= 0.01$,
- Total iterations: $T = 300$.

The robot initial position is$[2; 1; 1; \pi/4]$, then it avoids the obstacle in $[4; 4]$, and reaches the goal at $[6; 6]$. Simulations for different initial orientations, like $\mathbf{X}_{\text{state}} = [2; 1; 1; \pi]$, show that the robot follows the trajectory correctly even when the initial conditions change.

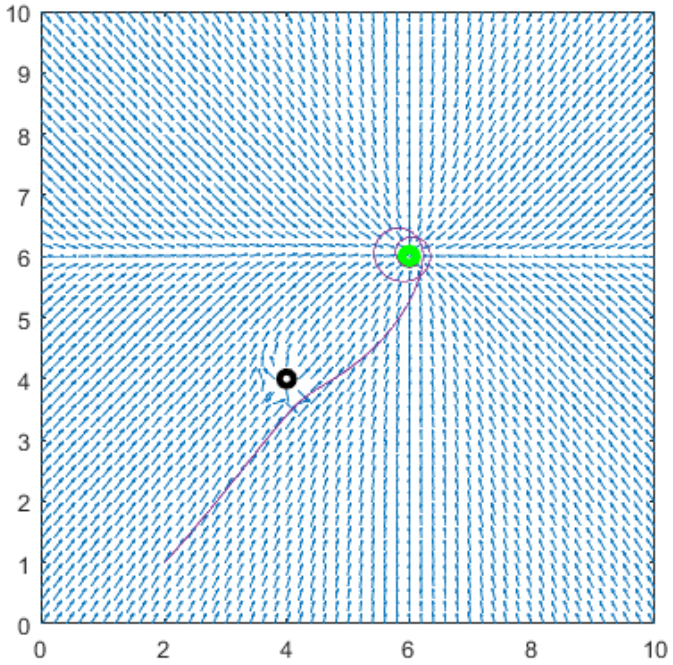

Figure 4: Non-holonomic robot with point obstacle

3.4 - Relationship of β in the repulsive and α on the attractive field

When the beta value is 20 times larger, it is too large and overpowers the alpha attractive value. See in the figure how the field converges towards a point beyond the goal, given how overpowering the repulsive force is.

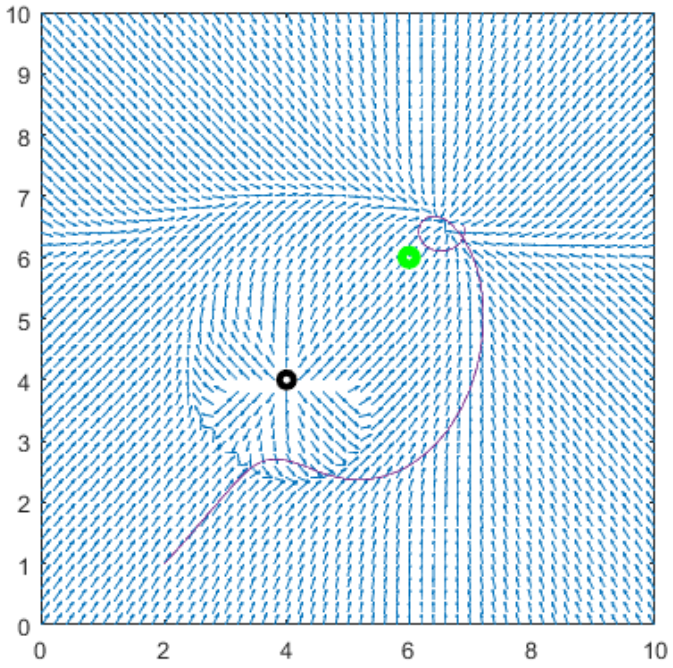

Figure 5: Non-holonomic with large repulsive beta

Questions 4: More realistic scenario with circular shaped obstacles

4.1 Computing the Distance Vector to a Circular Obstacle

In this final scenario, the obstacle is modeled as a circular object.

4.2 Avoiding Circular Obstacles with the Modified Potential Field

the gradient of the potential is modified to use the circular obstacles. Using the function `distance_obs.m` the repulsive gradient is computed:

$$\nabla U_{\text{rep}} = \frac{-\mathbf{d}_{\text{final}}}{\|\mathbf{d}_{\text{final}}\|^3}, \tag{31}$$

where $\mathbf{d}_{\text{final}}$ distance from the robot to the nearest point of collision with the obstacle.

- Obstacle center: $\mathbf{p}_{\text{obs}} = [5; 5]$,
- Obstacle radius: $R_{\text{obs}} = 0.2$,
- Goal position: $\mathbf{p}_{\text{goal}} = [6; 6]$,
- Initial state: $\mathbf{X}_{\text{state}} = [2; 1; 1; \pi/4]$.

The robot successfully avoids the obstacle and navigates towards the goal. Below is a figure showing the trajectory of the robot avoiding the obstacle and reaching the goal.

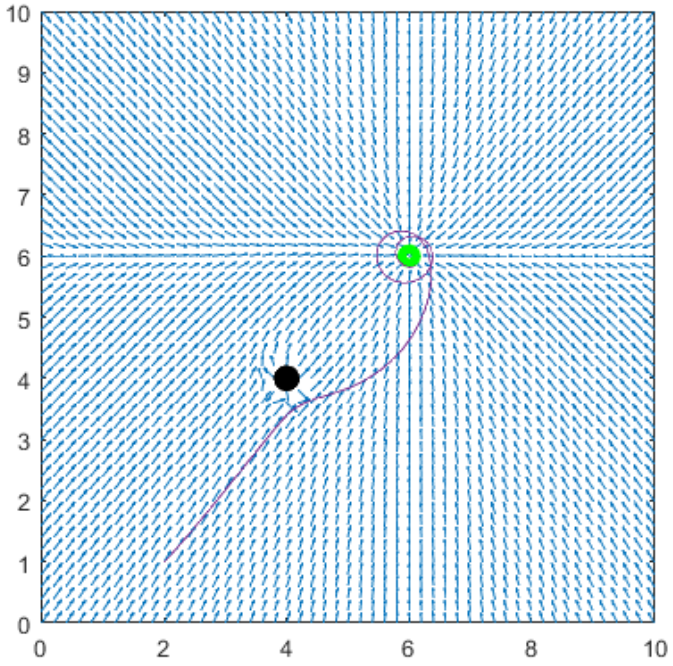

Figure 6: Circular obstacle

4.3 Navigation with Two Circular Obstacles

The gradient of the field function is modified to account for two obstacles in the map `distance_obs.m`. The total repulsive gradient is the sum of the gradients from the two obstacles, as shown below:

$$\nabla U_{\text{rep}} = \beta \cdot \left(\frac{-\mathbf{d}_{\text{final},1}}{\|\mathbf{d}_{\text{final},1}\|^3} + \frac{-\mathbf{d}_{\text{final},2}}{\|\mathbf{d}_{\text{final},2}\|^3} \right), \tag{32}$$

where $\mathbf{d}_{\text{final},1}$ and $\mathbf{d}_{\text{final},2}$ are the distance vectors to the nearest points to the obstacles.

- Obstacle 1: Center $\mathbf{p}_{\text{obs}} = [4; 4]$, Radius $R_{\text{obs}} = 0.2$,
- Obstacle 2: Center $\mathbf{p}_{\text{obs2}} = [6; 4]$, Radius $R_{\text{obs2}} = 0.2$,
- Goal position: $\mathbf{p}_{\text{goal}} = [6; 6]$,
- Initial state: $\mathbf{X}_{\text{state}} = [2; 1; 1; \pi/4]$.

In this last part, the robot avoids both obstacles and reaches the goal with no collisions. We can say for certain that the robot adapts to different scenarios where the trajectory or number of obstacles changes, and the system is robust enough to adapt to the circumstances. We can see the performance of the robot in the figure 7 below:

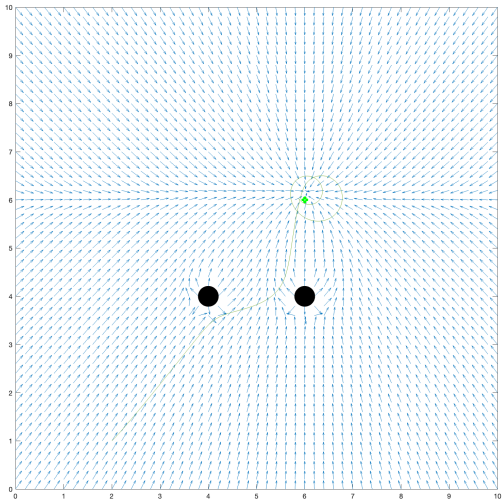

Figure 7: Two circular obstacles

Question 5: Conclusion about the use of the APF approach for autonomous navigation by analyzing its advantages and limitations

Artificial potential fields are a good way to navigate an environment for a low computation cost, making them attractive for systems with limited processing power. However, a mathematical model of the environment is needed and the equation describing the environment also needs to be differentiated which could be time consuming depending on how complicated the model. Simple shapes need to be used so complicated shapes which may be able to be approximated in a occupancy grid for example are harder to express. Despite these drawbacks, APFs remain a valuable tool for autonomous navigation.