CS – 475 Assignment 6
Path Planning using Artificial Potential Field Algorithm

Kalykakis Emmanouil
csdp1210@csd.uoc.gr

`Release date: 15/05/2023`
`Deadline: 23/05/2023`

1. Introduction

In this assignment you will implement a path planning algorithm that can be used for autonomous navigation and will guide your robot to a fixed goal position. The algorithm you are going to create is the Artificial Potential Field (APF). The idea behind APF is that you assign an imaginary ("artificial") repulsive field to every obstacle and an attractive one to the goal position. The artificial forces applied to the robot guide it to the potential global minimum, i.e. the goal point. The artificial fields can be visualized as follows:

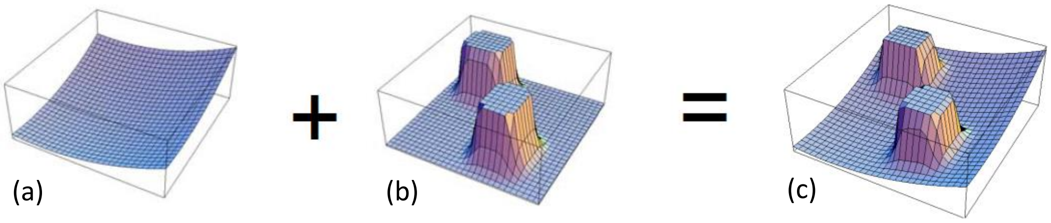

Figure 1 a) The attractive potential without obstacle b) the repulsive potential set the highest value to the obstacle c) Whole potential shows the combination of the two forces to get the final potential field result.

2. Method

There are two types of potential fields that are named attractive and repulsive. The attractive is the one placed to the goal point and practically tries to pull the robot towards its position. The mathematical formula that describes this potential is:

$$U_{att}(x,y) = \frac{1}{2}K_a r^2$$

where K_a is a positive constant that determines the strength of the attraction force and $r = \sqrt{(x - x_{goal})^2 + (y - y_{goal})^2}$ is the Euclidean distance between the robot and the goal position.

The repulsive potential is placed at each obstacle and can be described with the following formula:

$$U_{rep}(x,y) = \begin{cases} \frac{1}{2}K_r\left(\frac{1}{r} - \frac{1}{r_0}\right)^2 & r \leq r_0 \\ 0 & r > r_0 \end{cases}$$

where, K_r is the repulsive constant (how strong the repulsive field is), r_0 is the distance threshold and r is the distance between the robot and the obstacle. If this distance is less than the threshold, then the repulsive field is applied.

The total potential field is given by the sum of the attractive and repulsive potentials:

$$U_{total}(x,y) = U_{att}(x,y) + \sum_{i=1}^{\#obstacles} U_{rep,i}(x,y)$$

The gradient of the total potential function can be used to compute the force acting on the robot:

$$\vec{F} = -\nabla U_{total}(x,y)$$

The force vector $\vec{F}$ can then be used to control the motion of the robot in a desired direction. At each update step you follow the smallest value of the total potential around the current position until you reach minimum.

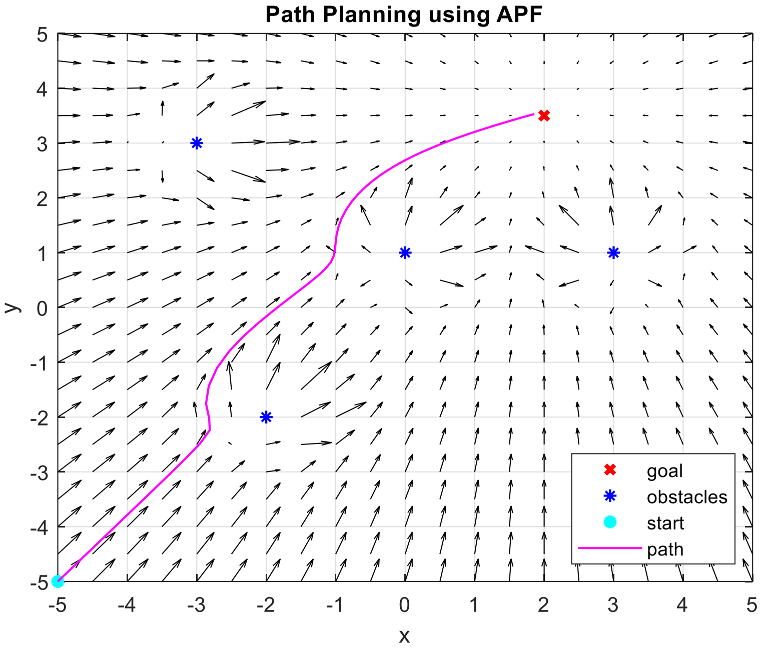

Figure 2 Total potential field and path

In an Artificial Potential Field (APF) algorithm, the position of a robot is updated based on the attractive and repulsive forces acting on it. The goal is to navigate the robot towards a target while avoiding obstacles in the environment. The formula which is used to update the position (x,y) of the robot based on the control force calculated from the artificial potential field is:

$$q = q - \alpha \cdot \nabla U_{total}(q)$$

where:

$q = (x,y)$ is the current position of the robot,

α is a positive constant (learning rate) representing the step size for the update

∇U_{total} is the gradient of the total potential energy, which is the sum of the attractive and repulsive potentials.

The gradient of each potential can be obtained by taking the partial derivative of the potential with respect to the position of the robot. Updating the position of the robot using the above formula enables the robot to move towards the goal (attractive region) while avoiding obstacles (repulsive regions) based on the control force calculated from the APF. The learning rate parameter α controls the step size of the update and affects the speed and stability of the robot's motion. It needs to be carefully tuned to achieve optimal performance. In practice, this formula is often used in an iterative manner, where the robot continuously updates its position based on the gradient until it reaches the desired goal, or a stopping condition is met.

3. Implementation

In your zip file, you are provided with a ROS package named **cs475_asgmt6**. As always, copy paste it to your ROS workspace and compile. In case you need more information, check out the previous assignments. This package contains a launch file ("burger.launch"), that opens the gazebo simulation with the turtlebot and also creates a topic "/obstacles" where the position of every obstacle is published.

For this assignment, you are not provided with any source code, and you need to create a ROS node from scratch named "apf.py". The final goal of the assignment is to visualize the path that the robot should follow in order to navigate to the goal point in **Rviz**. The actual navigation part of the robot is BONUS 10%. The initial robot position is $(-2,0)$ and the goal position should be set at $(2,0)$.

To visualize the path in RViz you can use the nav_msgs/Path message. You must publish the aforementioned message at a new topic and open RViz: **rosrun rviz rviz** then click add and by topic and choose the topic you've created. You should see a line that creates the desired path. Optionally, you can create a grid map and calculate the total potential value at each point. You can use the visualization_msgs/Marker message for visualizing the potential field.

Finally, if you want to navigate the robot (bonus part) to follow the created path, you will need to publish at the /**cmd_vel** topic after you've transformed the desired position and orientation to linear/angular velocity and time duration of the command.

4. Submission

Send your node (apf.py) attached to email: **csdp1210@csd.uoc.gr** with subject "**[CS-475] Assignment 6 submission**". Don't forget to mention your name and registration number. The deadline is due **23/05/2023 23:59.**