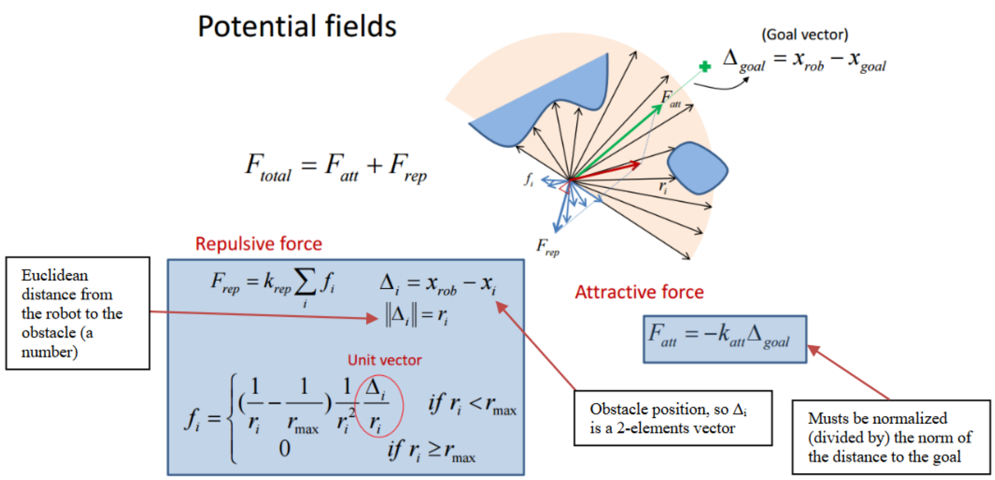
Robotics: Potential Fields

Salvador Carrillo Fuentes

January 2019

1 Introduction

In this exercise we are going to complete a code implementing a reactive navigation algorithm based on Potential Fields. This code generates a random map with a given number of obstacles, and request s the user to set the robot's start and goal positions by clicking with the mouse on that map. Then you just have to enj oy watching how the robot moves!



2 Implementation

- Implement the computation of the Repulsive Force `FRep`, which is the sum of the repulsive forces yielded by each obstacle close to the object. Recall that forces are 2-elements vectors. Plot an asterisk over the obstacles that has influence on this force, and store the handler of that plot in `hInfluentialObstacles`.

  If we discover an obstacle (or several) close (indicated by the radius of influence) to the robot position, first, we plot an asterisk over the obstacle/s. After this, we obtain the distance to that obstacle/s and compute the increment in the position of the obstacle/s to reach the robot position.

  Finally, we compute f_i using

  $$f_i = (\frac{1}{r_i} - \frac{1}{r_{max}})\frac{1}{r_i^2}\frac{\Delta_i}{r_i}$$

  If no obstacle is close to the robot, simply set `FRep = [0 ; 0];`

```
hInfluentialObstacles = plot(
                              Map(1, iInfluencial),
                              Map(2, iInfluencial), '*'
                          );
distance = Distance(iInfluencial);
difference = -Dp(:, iInfluencial);
f_i = (1 ./ distance - 1/RadiusOfInfluence) .* 1/
    RadiusOfInfluence^2 .* difference ./ distance;

FRep = KObstacles * sum(f_i,2);
```

- Compute the Attractive Force `FAtt`. Normalize the resultant Force by $\| \Delta_{goal} \|$

  Applying

  $$F_{att} = -k_{att}\Delta_{goal}$$

  and normalizing:

  `FAtt=(-KGoal*(xRobot-xGoal)) / norm(xRobot-xGoal);`

- Compute the Total Force `FTotal`.

Using

$$F_{total} = F_{att} + F_{rep}$$

`FTotal = FAtt + FRep;`

3 Understanding what's going on

- Explain the meaning of each element appearing in the plot during the simulation of the Potential Fields reactive navigation.

  La flecha azul es el robot, la estrella verde el objetivo y cada circunferencia roja es un obstáculo.

- Run the program setting different start and goal positions. Now change the values of the goal and obstacle gains (`KGoal` and `KObstacles`). How does this affect the paths followed by the robot?

  Cuanto menor es el valor de `KObstacles` más cerca pasa el robot de los obstáculos (menos repulsión). Cuanto mayor es el valor de `KGoal` mayor atracción ejerce el objetivo, llegando incluso a hacer que nunca se alcance debido a que la fuerza va oscilando.

- Play with different numbers of obstacles and discuss the obtained results.

  El robot tarda más al ir calculando las fuerzas contrarias, pero avanza seguro hacia el objetivo, el cual acaba alcanzando.

- Illustrate a navigation where the robot doesn't reach the goal position in the specified number of steps. Why did that happen? Could the robot have reached the goal with more iterations of the algorithm? Hint: take a look at the `FTotal` variable.

  Cuanto mayor es el valor de `KGoal` mayor atracción ejerce el objetivo, llegando incluso a hacer que nunca se alcance debido a que la fuerza va oscilando. Más iteraciones no servirían de nada.