SC627 - Assignment 2

Veejay Karthik J - 203230010

February 2022

1 Artificial Potential Fields

The attractive(U_{att}) and repulsive(U_{rep}) potentials are as follows,

$$U_{att}(q) = \begin{cases} \frac{1}{2}\chi d^2(q, q_{goal}) & d_{goal} \leq d^*_{goal} \\ d^*_{goal}\chi d(q, q_{goal}) & d_{goal} > d^*_{goal} \end{cases} \tag{1}$$

$$U_{rep}(q) = \begin{cases} \frac{1}{2}\eta\left(\frac{1}{d_i(q)} - \frac{1}{Q^*_i}\right)^2 & d_i(q) \leq Q^*_i \\ 0 & d_i(q) > Q^*_i \end{cases} \tag{2}$$

The given parameters for simulation are as follows,

$$\chi = 0.8 \quad ; \quad d^*_{goal} = 2 \quad ; \quad \eta = 0.8 \quad ; \quad Q^*_i = 2$$

The gradient of the attractive and repulsive potential functions are computed using the Symbolic Math toolbox of MATLAB. The following function is defined to determine the closest distance to the polygonal obstacle from the robot's position.

1.1 closestPolygonEdgeComputer()

This function computes the closest point on the obstacle with respect to the robot's current position. Since the each polygonal obstacle is characterized by a set of line segments with the vertices given by a set of point P_i, the set of points within each line segment(P_λ) can be parameterized as follows,

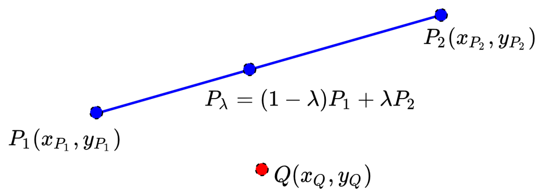

Figure 1: Parameterization of points in a line segment

$$P_\lambda = (1 - \lambda)P_1 + \lambda P_2 \quad ; \quad \lambda \in [0, 1]$$

Using this parametric representation of the line segment, the closest point on the same with respect to any arbitrary outside point can be computed as the solution to the following constrained optimization problem. (Here, $d(.,.)$ represents Euclidean distance function)

$$\begin{aligned} \min_{\lambda} \quad & d(Q, P_\lambda) \\ & 0 \leq \lambda \leq 1 \end{aligned} \tag{3}$$

This optimization problem can be solved as an unconstrained problem by employing the first-order condition for optimality and imposing bounds on the obtained value of the optimal λ^* as follows. Lets optimal value of λ obtained as a solution to the above optimization problem by solving it in the unconstrained paradigm be represented as λ^*_u.

$$\left(\frac{d}{d\lambda}(d(Q, P_\lambda))\right)_{\lambda=\lambda^*_u} = 0 \tag{4}$$

The solution to equation (4) is analytically obtained as follows,

$$\lambda_u^* = \frac{(x_Q - x_{P_1})(x_{P_2} - x_{P_1}) + (y_Q - y_{P_1})(y_{P_2} - y_{P_1})}{(x_{P_2} - x_{P_1})^2 + (y_{P_2} - y_{P_1})^2}$$

To optimal value of λ^* which is the solution to the actual optimization problem is computed from λ_u^* as follows,

$$\lambda^* = \begin{cases} 0 & \lambda_u^* < 0 \\ \lambda_u^* & 0 \leq \lambda_u^* \leq 1 \\ 1 & \lambda_u^* > 1 \end{cases}$$

Using this optimal value of λ^*, the closest line segment of the polygon P can be obtained by computing the optimal distances to each line segment of the polygon and determining that line segment which is at the least distance from an exterior point Q.

Suppose if the line segment under consideration is P_iP_{i+1}, based on the value of λ^* obtained, conclusions on the location of the closest point can be made.

1. Suppose if $0 < \lambda^* < 1$, it implies that the closest point lies in the interior of the line segment P_iP_{i+1}.
2. The cases where $\lambda^* = 0$ or $\lambda^* = 1$ corresponds to the case where the closest point on the line segment is its vertices P_i and and P_{i+1} respectively. In this case, the unit tangent vector is computed as the tangent to the circular region as shown in the previous subsection.

1.2 Artificial Potential Field - Motion Planning

1. Based on the distance of the robot's current position from the goal point, the attractive potential function is constructed as in (1). This attractive potential is given by U_{att}.
2. The closest distance to each obstacle is computed for all the polygonal obstacles by the function - closestPolygonEdgeComputer(), and the repulsive potential is cumulatively constructed according to (2). This repulsive potential is given by U_{rep}.
3. From the previous two steps, the total potential is defined by,

   $$U = U_{att} + U_{rep}$$

   To carry out the motion planning, the input ($\vec{v}$) to the motion planner is computed as follows,

   $$\vec{v} = -\vec{\nabla} U_{att} - \vec{\nabla} U_{rep}$$

4. Using the value of the computed input(v) along with the value of the step size (α) obtained from the input text file, the motion planning is recursively until the robot reaches the final goal point as follows,

   $$(x_{n+1}, y_{n+1}) = (x_n, y_n) + \alpha\vec{v}$$

1.3 Data Retrieval

The function **DataRetrievalText()** serves two purposes.

Firstly, it retrieves the necessary information for the working of the bug algorithm from an input text file. This information consists of the Start point, Goal point, step size and the set of vertices of the obstacles in the environment.

Further, it then passes this set of vertices into a function **VerticesSorter()** which sorts and rearranges in a cyclic order.

1.3.1 VerticesSorter()

To sort the vertices in a cyclic order, the following algorithm is employed. Given a set of $\{(x_i, y_i)\}$ representing the vertices of an obstacle, the centroid of this set is computed as (x_{cen}, y_{cen}). The co-ordinates of the vertices are then recomputed as $\{(x_i, y_i)\} - (x_{cen}, y_{cen})$ (Transformed Vertices).

Now, to arrange them in a cyclic order, the angle subtended by the transformed vertices with respect to the positive direction of $x-$ axis are sorted using the bubble sort technique in this function.

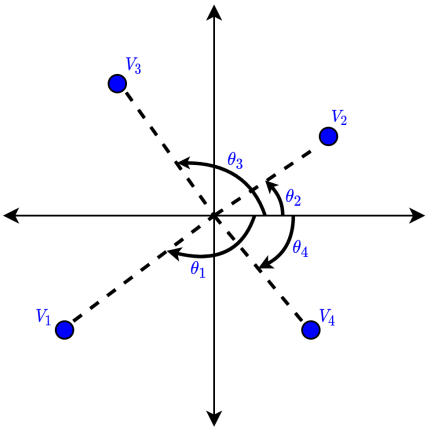

Figure 2: Transformed Vertices and their corresponding angles which are arranged in a cyclic order. Suppose if the set of vertices in the input text file are in a sequence given by $\{V_1, V_2, V_3, V_4\}$, this functions returns them in a cyclic sequence given by $\{V_1, V_4, V_2, V_3\}$ as shown in this figure.

1.4 Euclidean_distance()

This function computes and returns the Euclidean Distance between the points passed through it as its arguments.

2 Simulation Results

The simulation results of the artificial potential field based motion planner for the given inputs from the text file is as follows,

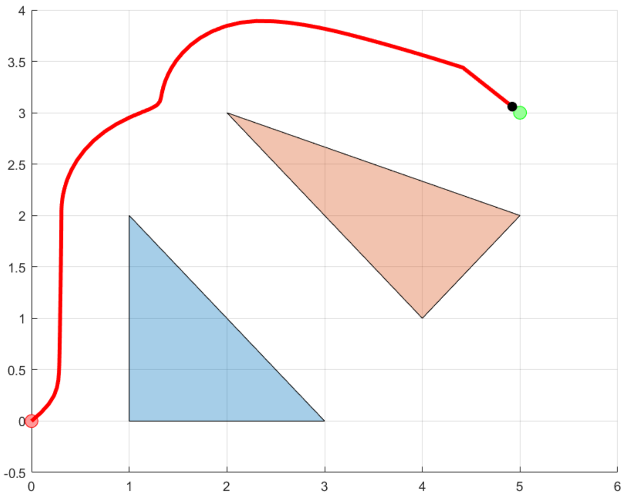

Figure 3: Simulation Outputs