Implementation of Pure Pursuit and Vector Field Histogram

Under the guidance

Of

Prof. Debasattam Pal

Report Submitted

By

BHAGAT RAM LABANA (22M1083)

SHUBHAM DEY (22M1084)

Title: Pure Pursuit Algorithm

Introduction:

- Purpose of the Experiment:
  The main purpose of the experiment was to make students to know about advanced autonomous robot systems which follow the principle of path or track following and obstacle avoidance by pure pursuit algorithm and vector field histogram method respectively.
- Background information about the topic:
  The pure pursuit algorithm is a control approach used to track a desired path by the autonomous robot or vehicles. The vehicle's current speed helps in identifying the look-ahead points on the path and calculates the required angular velocity and several other factors required to direct the vehicle toward the goal point.
  The pure pursuit algorithm is basically a method for geometrically calculating the arc required for getting the robot or vehicle on the path. Proper look ahead distance has to be selected among the different waypoints.
  The vector field histogram method helps in obstacle avoidance and is an advanced method that is devoid of the limitations which were present in the previous obstacle avoidance method such as the virtual force field method, edge detection, certainty grid, and potential field method.
- Methodology: We have used the software MATLAB 2020a to simulate the pure pursuit algorithm. From the same software, we have used the Simulink model for Simulation purposes. First, we used an inbuilt pure pursuit block in Simulink to implement a pure pursuit algorithm for the given robot system. Afterward, we go through the simulation onramp So that we can learn the various tools and functions necessary for the experiment. After we implanted in-built VFH to avoid obstacles in addition to the pure pursuit block. We created a new function block that takes waypoints, and position as inputs and provides the linear velocity, angular velocity, and that output. We ran a Matlab simulation code after discussing it with our fellow batchmates. And came up with an algorithm for pure pursuit the code and the algorithm for the same have been mentioned below.

- Pure Pursuit Algorithm
  The pure pursuit approach is a method of geometrically determining the curvature that will drive the vehicle to a chosen path point, termed the goal point. We designed an algorithm to implement the pure pursuit algorithm. We have given the Waypoints and Position as the input to the algorithm. Waypoints are the point through which the robot is desired to pass. The Input Position is reading the current position of the robot where

the current position includes X, and Y coordinates and Angle(theta) with respect to the X axis in the robot frame. And the outputs taken are the Linear velocity(v), Angular velocity(w), and Target Direction(phi). For the Algorithm we have implemented we are keeping the linear velocity fixed and varying the angular velocity according to the curvature we are finding using the pure pursuit algorithm.

Here in the given below diagrams has shown the position of the robot and the goal points. Geometry for the algorithm is given in the given figure. l is the lookahead distance from the location of the robot to the goal point. r is the curvature that has been calculated from the given figure geometry. If we set the fixed lookahead distance l and find the x coordinates of the goal point in the robot frame, we can find the curvature of the path from the robot position to the goal point. We can set some fix velocity v and by finding r from the given algorithm we can decide the desired angular velocity w.

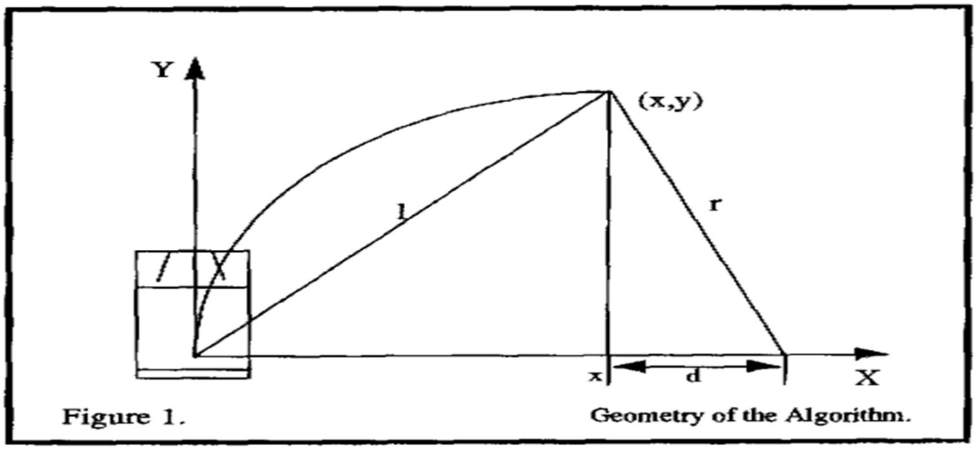

Figure 1. Geometry of the Algorithm.

$$x^2 + y^2 = l^2$$

$$x + d = r$$

$$(r - x)^2 + y^2 = r^2$$

$$r = \frac{l^2}{2x}$$

Target direction is our important goal to implement this algorithm. To find the target direction we need to find the angle(phi) of the line joining the current position of the robot to the goal point to the x-axis in the robot frame. Target Direction can be formulated as

$$target_{Dir} = theta - phi$$

Where phi is given by

$$phi = \tan^{-1} \frac{Xg}{Yg}$$

Where Xg and Yg are the goals with respect to the robot frame. As we are getting the current position point in a global frame. Our goal point also was in the global frame. We use the rotation matrix to convert the global frame goal point into the robot frame so that we can calculate the curvature of the path from the pure pursuit algorithm. The Rotation matrix is defined as,

$$R = \begin{bmatrix} \cos(theta) & \sin(theta) \\ -\sin(theta) & \cos(theta) \end{bmatrix}$$

$$\begin{bmatrix} X \\ Y \end{bmatrix} = R * \begin{bmatrix} \frac{Xg}{Yg} \end{bmatrix}$$

This X value is used to find the curvature of the path.

MATLAB Code:

```
function[V,W,Target_Dir]=Pure_Pursuit(Waypoints,pose)
V=0.5;                                             %Assigning fixed LinearVelocity to the robot
W=0;                                                       %Assigning `initial AngularVelocity
Theta=0;                                                         % Initializing the theta value
index=1;                      % Initializing a variable index to store the closest point index on the path
phi=0;                                                                   %Initializing phi value
X=0;                                                               % Goal point X coordinate
Y=0;                                                               % Goal point Y coordinate
k=0;

L = 1.5                        % Lookahead distance
Xt = pose(1)                   %Current position Xt coordinate
Yt = pose(2)                   %Current position Yt coordinate
theta = pose(3)                %Current angle w.r.t. Goal point
R_matrix = [cos(theta) sin(theta); -sin(theta) cos(theta)]     %Rotation matrix for global to robot frame
conversion
Dist_array = []
for j = index:5
     for i = index:5
         C1 = Waypoints(i,1) - Xt;
         C2 = Waypoints(i,2) - Yt ;
         d = sqrt(C1^2 + C2^2);
         d_New_array = [Dist_array d] ;
     end
    [M,k] = min(Dist_array) % minimum distance
     if M > L
         X = Waypoints(k,1)
         Y = Waypoints(k,2)
     else
         X = Waypoints(k+1,1)
         Y = Waypoints(k+1,2)
      end
      index = k
```

```
end

 Global_Frame = [X ; Y]                              % Goal point in global frame
  Robot_Frame = R_matrix*Global_Frame    % Goal point in Robot frame
  X_1  = Robot_Frame(1,1);                     %Extracting x-axis goal point coordinate in a global frame
  X_p = X- Xt;                          %  Current to goal position x-axis Coordinate difference
  Y_p = Y- Yt;                          %  Current to goal position y-axis Coordinate difference
  r = (L^2)/(2*X_1);                % Radius of curvature to find the path
  W = V/r ;                              %Angular Velocity decided by v and the radius of curvature

  Phi = atan(X_p/Y_p);               % Phi is initialized before
  Target_Dir = theta - phi ;          % Target Direction calculated to head the vehicle towards the goal
points
```

Vector Field Histogram

The vector field histogram algorithm works by computing steering directions for a robot or autonomous vehicle based on sensor readings that are obstacle free. The sensor readings are used to plot polar density histograms to calculate obstacle location and proximity. These histograms are converted to binary histograms to validate the steering directions of the robot. The VFH algorithm takes robot size and turning radius as input and provides the steering direction of the robot as output to avoid obstacles in the path and reach the goal point.

Algorithm or steps for VFH:

- Convert the active window environment into a grid or map into a histogram and the obstacles provide repulsive forces and the goal point provides the attractive forces.
- The net magnitude and directions of movement of the robot are computed depending on the values and readings taken by the sensor.
- Depending upon the magnitude of the repulsive forces the obstacles are avoided.
- Finally with the help of the above algorithm along with the pure pursuit algorithm, the goal point is reached.

Simulation and the Modeling:- We have used the software MATLAB 2020a to simulate the pure pursuit algorithm. From the same software, we have used the Simulink model for Simulation purposes. Inbuilt Pure Pursuit and Inbuilt Vector field histogram block we have used to follow the path avoiding the obstacles. Pure pursuit block already has been described in methodology section. And In VFH block Ranges , Angles and TargetDir as the input and Steering angle as the output.

References: -

1)Implementation of the Pure Pursuit Path tracking Algorithm
https://www.ri.cmu.edu/pub_files/pub3/coulter_r_craig_1992_1/coulter_r_craig_1992_1.pdf
2) J. Borenstein and Y. Koren, "The vector field histogram-fast obstacle avoidance for mobile robots," in IEEE Transactions on Robotics and Automation, vol. 7, no. 3, pp. 278-288, June 1991, Doi: 10.1109/70.88137.