EE-562 : Robot Motion Planning
Problem Set # 2

Spring 2013-14

Due Date : Feb 5, 2014.
Total Points : 100.

Tangent Bug Algorithm

Implement Tangent Bug algorithm in ROS keeping in view the following details.

Robot: The robot is a point robot.

Environment: The environment is MxN units, where the (0,0) start at the top left corner. X-axis is along vertical line while Y-axis is along horizontal.

Obstacles: The obstacles are of the following type:

- **Rectangular** parameterized by four corners (x1,y1), (x2,y2), (x3,y3) and (x4,y4).
- **Triangular** parameterized by three corners (x1,y1), (x2,y2) and (x3,y3).
- **Circular** parameterized by center (x,y) and radius r.
- **Boundary** The boundary of the environment is also an obstacle so the Robot can not escape it.

Robot Position and Goal: The current position of the robot is (Xc,Yc) and the goal is (Xg,Yg).

Sensor: The range of the sensor is R and its resolution is rx. Resolution is the difference in degrees between two consective scans.

Movement: The Robot covers dx distance in one step.

ROS Nodes: The following three nodes are must. You can add further nodes as required.

- **Motion Planning Node:** This is the main node that plan the actual motion. It also publishes the robot current position.
- **Sensor Node:** This node takes the robot current position and sensor range R and scan the surrounding area for obstacles.
- **Obstacle Collision Check Node:** This node has all the obstacles defined in it. It should provide a service that will find if a position is in obstacle or not.

Deliverables

1. Send all the code of the whole package that should execute without errors.
2. Run your system for the following parameters and send us the output path. The system is also shown in figure 1 below.
   (a) Map dimensions are 10x12 i.e. 10 units in x-axis and 12 units in y-axis.
   (b) Robot starting potion is (5,2).
   (c) Goal is at (6,10).
   (d) Step resolution is 0.1 i.e. a robot can move 0.1 unit in a single step.
   (e) Sensor view angle is 360 degree and range R is 0.4 unit i.e it can sense upto 0.4 unit from its current position in any direction as shown by yellow circle in figure 1.
   (f) Sensor resolution rx is 5 degrees.
   (g) Obstacles are as following.
      i. Boundary: Rectangle with corners (0,0), (0,12), (1,12) and (1,0)
      ii. Boundary: Rectangle with corners (1,0), (1,1), (10,1) and (10,0)
      iii. Boundary: Rectangle with corners (10,1), (9,1), (9,12) and (10,12)
      iv. Boundary: Rectangle with corners (9,11), (1,11), (1,12) and (9,12)
      v. WO1: Circle at (5,3.5) with radius r = 0.5
      vi. WO2: Rectangle with corners (3.5,5), (3.5,6), (6.5,6) and (6.5,5)
      vii. WO3: Triangle corners (5.5,8.5), (2.5,8.5), and (2.5,6.5)
      viii. WO4: Triangle with corners (5,8.5), (7.5,8.5) and (7.5,6.5)

Good Practices and Hints

1. Stores the constants like map dimensions M and N, sensor range R etc in parameter server
2. For collision with obstacle use the set of points in the plane that satisfy $ax + by + c \leq 0$
3. ROS is not real time in itself so you have to synchronize the data by using the topics publish rate.

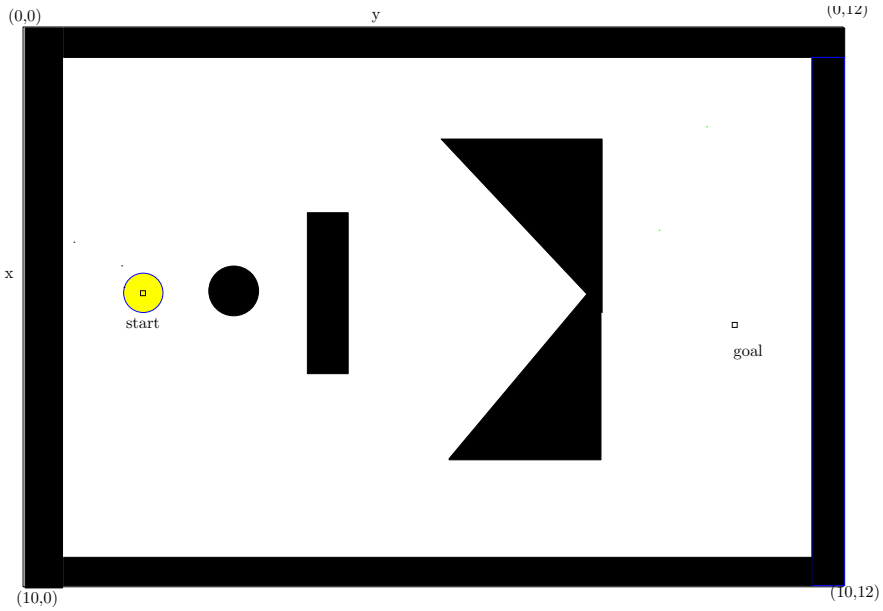

Figure 1: Scenario for Tangent Bug Algorithm