MTRX5700

Experimental Robotics

Assignment 3

Author(s):
Kausthub Krishnamurthy
James Ferris
Sachith Gunawardhana

SID:
312086040
311220045
440623630

Due: May 6, 2015

1 Question 1

Test Code Listing

See Appendix A [9.1]

2 Question 2

2.a Obtaining Obstacle Location from Laser Data

First we start by taking the data output from question one, which consisted of the robot $x - y$ coordinates in the world coordinate system, as well as the velocity and turn rate for that particular timestamp, for each timestamp that occurs in the Velocity observation data, the Compass data and the GPS data.
Similar to the way question one works, we combine this data with the Laser observation data by comparing timestamps, using the Prediction Stage equation to estimate the robots $x - y$ coordinates at the time that the laser data was generated. Combining the robots $x - y$ world coordinates with the relative position of the obstacles obtained from the Laser data.
For initial tests to attempt to identify obstacles, any range reading from the Laser data that was less than eight (the maximum range of the sensor) was considered an obstacle. It was intended to apply filters to this data once the Occupancy Grid had been generated. Until then, raw data would be used.

2.b Generate Occupancy Grid

To generate the Occupance Grid, we took the Data set of the X-Y coordinates of all 'Obstacles' detected, and determined the difference between the minimum and maximum x and y values deteced. Given a user defined size for the occupancy grid, we could then determine the $x - y$ range that corresponded to a grid location. Then for every Obstacle $x - y$ coordinate we could determine the grid location it corresponded to, incrementing the grid value to increase the weighting, which would indicate the likelihood of an obstacle being in that region. Some experimenting with the grid size using the data for the Robot path generated in question one indicated that a grid size of $200x200$ would be best, as this resulted in a grid map that, while not extremely sharp, was also not extremely blurred. Ideally, the grid would be vague enough to generalise a position for the obstacles, but not so clear that it simply resulted in a plot of every possible obstacle coordinates. See the figures below:

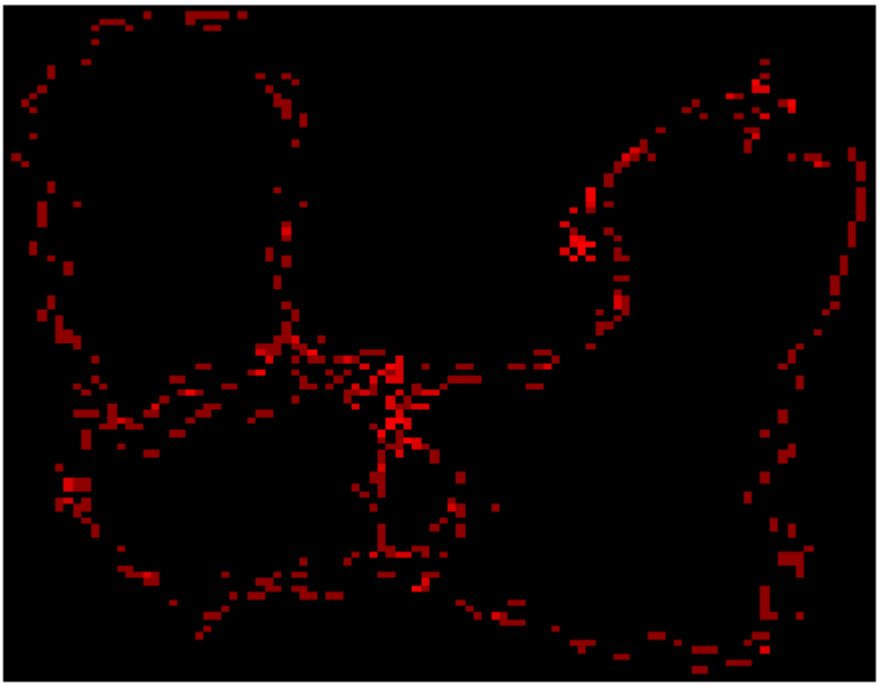

Figure 1: Grid size = 100×100

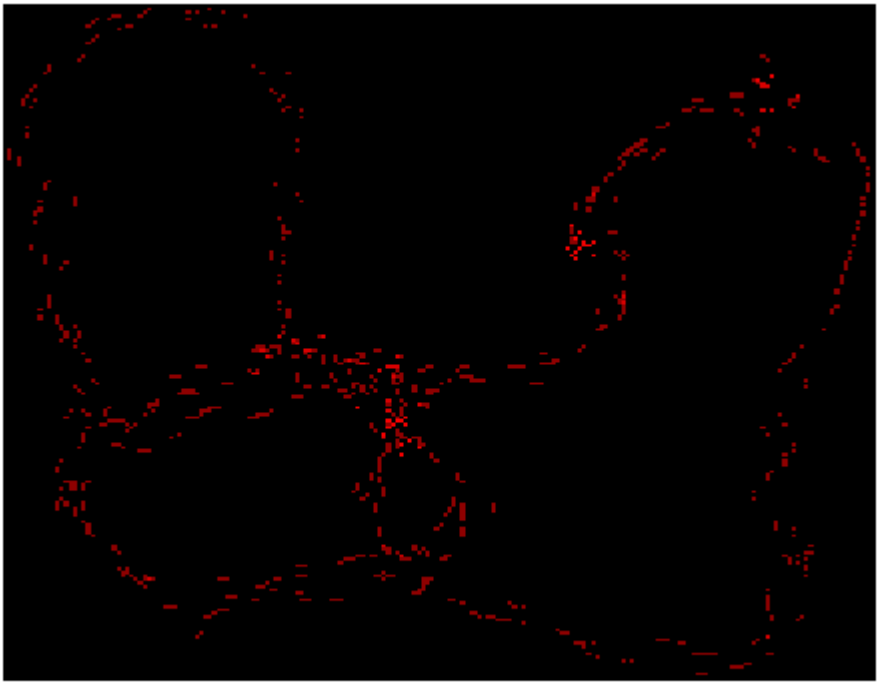

Figure 2: Grid size = 200×200

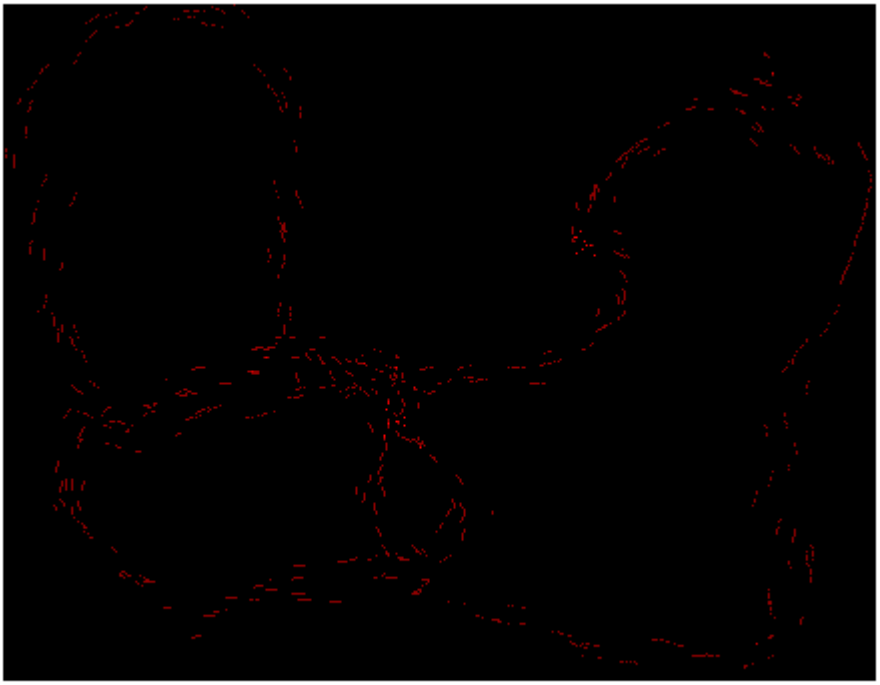

Figure 3: Grid size = 500×500

2.c Results

3 Question 3

Code Listing

See Appendix A [3] for all code used.

4 Appendix A

4.1 Question 1

4.2 Question 2

4.2.i obtainObstacles

```
clear
clc
close all

DEGREES = 180/pi;
RADIANS = pi/180;


positionData = load('q1output1.txt');

laserObs = load('laserObs.txt');

%Get output data
time1 = positionData(:,1);
Xpos = positionData(:,2);
Ypos = positionData(:,3);
heading = positionData(:,4);
velocity = positionData(:,5);
turnRate = positionData(:,6);

diary './q2Output4'

%Get laser data
time2 = laserObs(:,1) + (laserObs(:,2)*10^-6) - 1115116000;%get in microseconds


%Extracting range & intensity data from LaserObs

f1=1;
range = zeros(length(laserObs), (size(laserObs,2)-2)/2);
% intensity = zeros(length(laserObs), (size(laserObs,2)-2)/2);

%Extracting range & intensity data from LaserObs
for i=1:length(laserObs)
   for f2=3:2:size(laserObs,2)
%        if(laserObs(i,f2) < 8)
            range(i,f1)=laserObs(i,f2);
        %     intensity(i,f1)=laserObs(i,f2+1);
            f1=f1+1;
%        end
   end
   f1=1;
end

lasersX = 0;
lasersY = 0;

% alphaP = 0.5;
% alphaTH = 0.5;

lastTime = 0;
deltaT = 0;

latestVel = 0;
latestTurnRate = 0;

ourX = 0;
ourY = 0;
ourHeading = 0;

indLengths = [length(time1), length(time2)];
maxIters = max(indLengths);

interval = 20;

```

```
%iters [velInd, posInd, compInd, lasInd];
iters = [2, 2];
runFlags = [0, 0];
loopFlag = 1;
loopCount = 2;
%%loop starts



while(loopFlag == 1)
    loopCount = loopCount + 1;
    time = [time1(iters(1)), time2(iters(2))];
    nextT = min(time);


    for i = 1:2
        if time(i) == nextT
            runFlags(i) = 1;
        else
            runFlags(i) = 0;
        end
    end

    %if Positiondata
    if(runFlags(1) == 1)
        deltaT = time1(iters(1)) - lastTime;
        latestVel = velocity(iters(1));
        latestTurnRate = turnRate(iters(1));
        ourX = Xpos(iters(1));
        ourY = Ypos(iters(1));
        ourHeading = heading(iters(1));

        lastTime = time1(iters(1));%
        runFlags(1) = 0;
        if iters(1) >= length(time1)-interval
            time1(iters(1)) = 1.496*10^8;
        else
            iters(1) = iters(1) + interval;
        end
    end

% % if laser
     if(runFlags(2) == 1)
        deltaT = time2(iters(2)) - lastTime;
        pr = predictionStage(ourX, ourY, ourHeading, deltaT, latestTurnRate, latestVel);
        ourX = pr(1);
        ourY = pr(2);
        ourHeading = pr(3);
        lastTime = time2(iters(2));%
        runFlags(2) = 0;


        % Get X,Y coordinates for laser ob data

        for i = 1:size(range,2)
            if (range(iters(2),i) < 8.0 && range(iters(2),i) > 0.0001)
                lasersX = ourX + range(iters(2),i)*cos(((i-1)*0.5)*RADIANS+ourHeading);
                lasersY = ourY + range(iters(2),i)*sin(((i-1)*0.5)*RADIANS+ourHeading);
                diary ON
                fprintf('%d\t%d\n', lasersX, lasersY);
                diary OFF
            end
        end

        %increment
        if iters(2) >= length(time2)-interval
            time2(iters(2)) = 1.496*10^8;
        else
            iters(2) = iters(2) + interval;
        end

```

```
137
138       end
139
140  %check loop
141      if(time1(iters(1)) == 1.496*10^8)
142          if(time2(iters(2)) == 1.496*10^8)
143                          loopFlag = 0;
144          end
145      end
146
147  %plot stuff
148  % hold on
149  % title('Obstacles');
150  % xlabel('x-axis');
151  % ylabel('y-axis');
152  % legend('')
153  % drawnow
154  end
```

4.2.ii obtainObstacles$_v$2

```
1   clear
2   clc
3   close all
4
5   DEGREES = 180/pi;
6   RADIANS = pi/180;
7
8
9   positionData = load('q1output1.txt');
10
11  laserObs = load('laserObs.txt');
12
13  %Get output data
14  time1 = positionData(:,1);
15  Xpos = positionData(:,2);
16  Ypos = positionData(:,3);
17  heading = positionData(:,4);
18  velocity = positionData(:,5);
19  turnRate = positionData(:,6);
20
21  % diary './q2_2Output1'
22
23  %Get laser data
24  time2 = laserObs(:,1) + (laserObs(:,2)*10^-6) - 1115116000;%get in microseconds
25
26  lasersX = 0;
27  lasersY = 0;
28
29  % alphaP = 0.5;
30  % alphaTH = 0.5;
31
32  lastTime = 0;
33  deltaT = 0;
34
35  latestVel = 0;
36  latestTurnRate = 0;
37
38  ourX = 0;
39  ourY = 0;
40  ourHeading = 0;
41
42  indLengths = [length(time1), length(time2)];
43  maxIters = max(indLengths);
44
45  interval = 20;
46
47  %iters [velInd, posInd, compInd, lasInd];
```

```
iters = [2, 2];
runFlags = [0, 0];
loopFlag = 1;
loopCount = 2;
%%loop starts

xPos = zeros(1);
yPos = zeros(1);

while(loopFlag == 1)
    loopCount = loopCount + 1;
    time = [time1(iters(1)), time2(iters(2))];
    nextT = min(time);


    for i = 1:2
        if time(i) == nextT
            runFlags(i) = 1;
        else
            runFlags(i) = 0;
        end
    end

    %if Positiondata
    if(runFlags(1) == 1)
        deltaT = time1(iters(1)) - lastTime;
        latestVel = velocity(iters(1));
        latestTurnRate = turnRate(iters(1));
        ourX = Xpos(iters(1));
        ourY = Ypos(iters(1));
        ourHeading = heading(iters(1));

        lastTime = time1(iters(1));%
        runFlags(1) = 0;
        if iters(1) >= length(time1)-interval
            time1(iters(1)) = 1.496*10^8;
        else
            iters(1) = iters(1) + interval;
        end
    end

% % if laser
     if(runFlags(2) == 1)
        deltaT = time2(iters(2)) - lastTime;
        pr = predictionStage(ourX, ourY, ourHeading, deltaT, latestTurnRate, latestVel);
        ourX = pr(1);
        ourY = pr(2);
        ourHeading = pr(3);
        lastTime = time2(iters(2));%
        runFlags(2) = 0;


         xpoint = zeros(1);
         ypoint = zeros(1);
         for j = 4:2:size(laserObs,2)
             range = laserObs(iters(2),j-1);
             bearing = ((j)/2 - 90)*pi/180;
             if (range < 8.0)
                 xpoint = [xpoint  ourX+range*cos(bearing + ourHeading)];
                 ypoint = [ypoint  ourY+range*sin(bearing + ourHeading)];
             end

         end

         xPos = [xPos xpoint];
         yPos = [yPos ypoint];

        plot(xpoint(:), ypoint(:), '.');

        %increment
        if iters(2) >= length(time2)-interval
```

```
            time2(iters(2)) = 1.496*10^8;
        else
            iters(2) = iters(2) + interval;
        end


     end

%check loop
    if(time1(iters(1)) == 1.496*10^8)
        if(time2(iters(2)) == 1.496*10^8)
                    loopFlag = 0;
        end
    end

%plot stuff
% hold on
% title('Obstacles');
% xlabel('x-axis');
% ylabel('y-axis');
% legend('')
drawnow
% pause
end

xpoint(1) = [];
ypoint(1) = [];

xPos(1) = [];
yPos(1) = [];

xMin = min(xPos);
xMax = max(xPos);

yMin = min(yPos);
yMax = max(yPos);


gridSize = 100;

grid = zeros(gridSize);


yDiff = (yMax - yMin)/(gridSize - 2);
xDiff = (xMax - xMin)/(gridSize - 2);

for i = 1:length(xPos)
    tmpX = xPos(i);
    tmpY = yPos(i);
    j = 1;
    while (tmpX > xMin)
        tmpX = tmpX - xDiff;
        j = j + 1;
    end
    k = 1;
    while (tmpY > yMin)
        tmpY = tmpY - yDiff;
        k = k + 1;
    end

    grid(j,k) = grid(j,k) + 1;
end

HeatMap(grid);
```

4.2.iii generateOccupancyGrid

```
clear
```

```
close all
clc

% positionData = load('q1output1.txt');
positionData = load('q2Output4.txt');


% xPos = positionData(:,2);
% yPos = positionData(:,3);

xPos = positionData(:,1);
yPos = positionData(:,2);

xMin = min(xPos);
xMax = max(xPos);

yMin = min(yPos);
yMax = max(yPos);


gridSize = 100;

grid = zeros(gridSize);


yDiff = (yMax - yMin)/(gridSize - 2);
xDiff = (xMax - xMin)/(gridSize - 2);

for i = 1:length(xPos)
    tmpX = xPos(i);
    tmpY = yPos(i);
    j = 1;
    while (tmpX > xMin)
        tmpX = tmpX - xDiff;
        j = j + 1;
    end
    k = 1;
    while (tmpY > yMin)
        tmpY = tmpY - yDiff;
        k = k + 1;
    end

    grid(j,k) = grid(j,k) + 1;
end

% imagesc(grid);
HeatMap(grid);
```

4.3 Question 3

Code Listing

See Appendix A [9.1]