ISTANBUL TECHNICAL UNIVERSITY

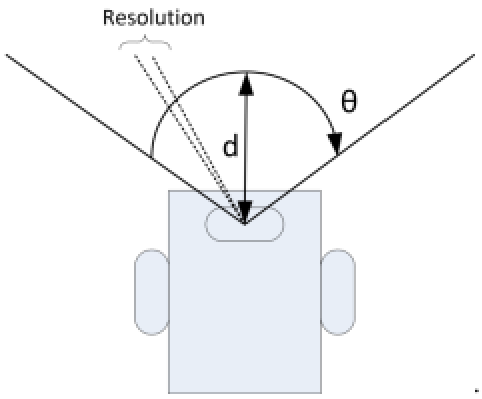
Probabilistic Methods in Robotics

HOMEWORK 5

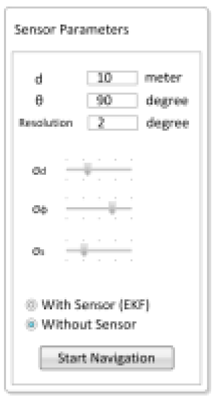
HAKAN AŞIK

518182008

Question: Consider the vehicle modelled by odometry method and the rectangular trajectory given in HW3. Assume a Laser Range Scanner is mounted on the mobile robot and it's able to measure range, bearing angle and appearance information with predefined resolutions as in the following figure:

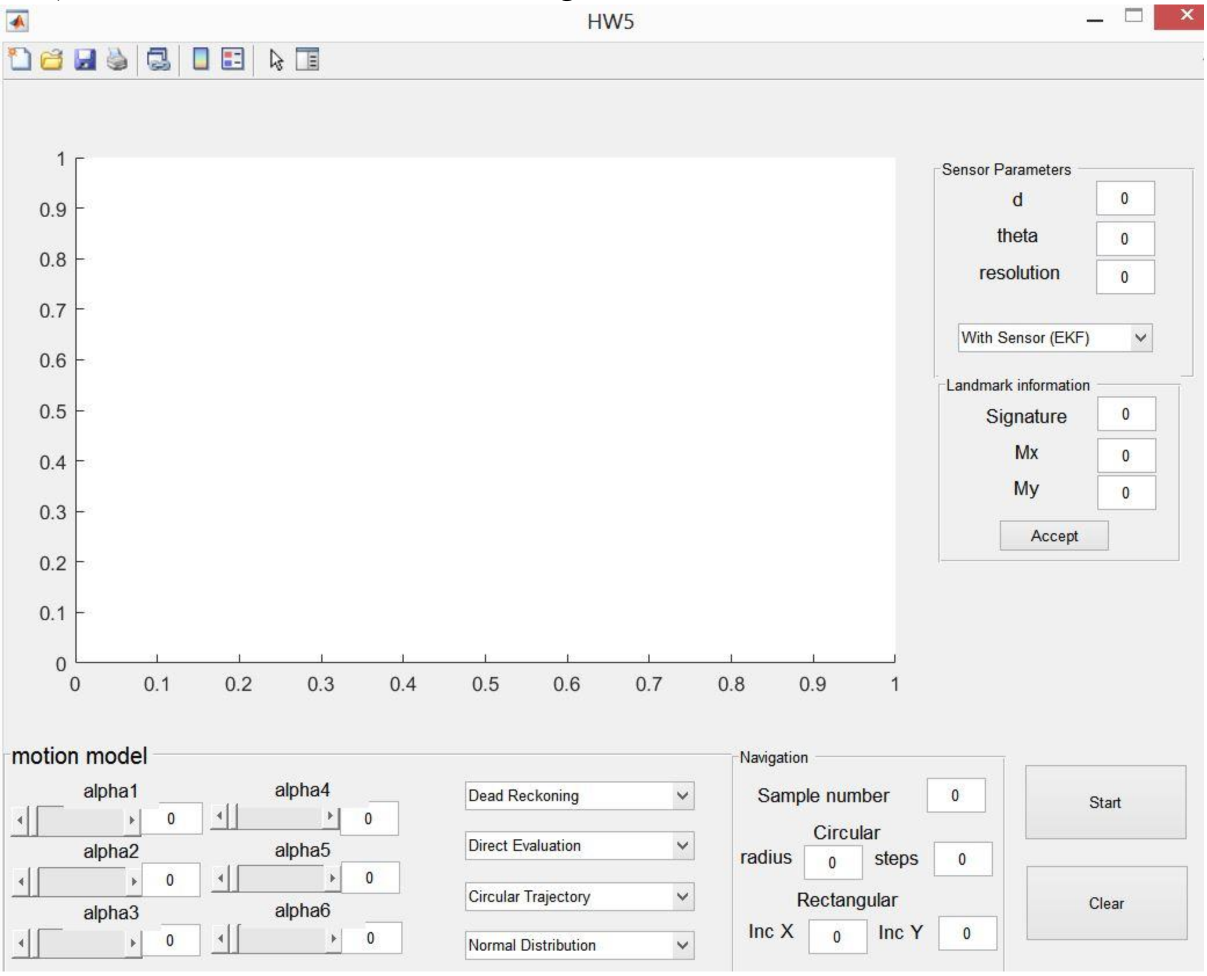


a) Add a sensor parameter window to your interface program to change sensor characteristics as below:

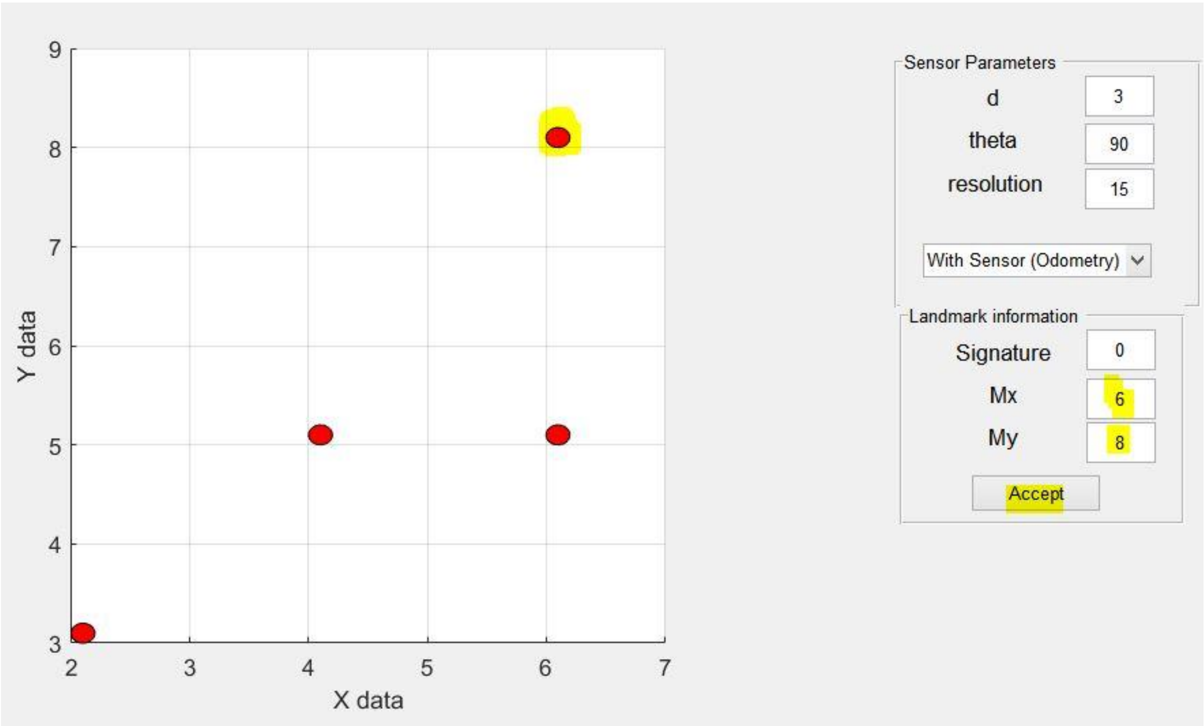


b) Add a land mark locator window to locate landmarks arbitrarily and start program to see how landmarks are detected by Laser Range Scanner sensor. And compare sensed landmark location with their correct locations

a, b-) GUI is modified to include Laser Range scanner and landmark inclusion:



Landmarks can be implemented into the graphics area by means of Landmark information section as shown in below Figure



Matlab file is **sensor_odometry.m**

```
%% Sensor points
step=step_number; % There are 12 steps but we define turning situations so
we use 14 step as input.
state=[d, theta];       % (Range, max. scan degree)
%state=[2,pi/2]
mu=[0;0;0];
for step = 1:14
       %% Trajectory
       if (step > 0 && step < 6)
           mu = [(step-1)*dx; 0; 0];  % [x,y,theta] They are the robot
ideal motions
           u = [dx; 2*pi];             % [v w]  Dead reckoning inputs
       elseif (step == 6)
           mu = [4*dx; (step - 6) * dx; pi/2];
           u = [0; pi / 2];
       elseif (step > 6 && step < 11)
           mu = [4*dx; (step - 6) * dy; pi/2];
           u = [dy; 2 * pi];
       elseif (step == 11)
           mu = [4*dx; 4*dy; pi];
           u = [0; pi / 2];
       elseif (step > 11 && step < 15)
           mu = [4*dx - ((step - 11) * dx); 4*dy; pi];
           u = [dx; 2 * pi];
       end
       % Sensor Area
       line([mu(1) mu(1)+state(1)*cos((mu(3) + (state(2) / 2))) ], [mu(2)
mu(2)+state(1)*sin((mu(3) + (state(2) / 2)))])
       line([mu(1) mu(1)+state(1)*cos((mu(3) - (state(2) / 2))) ], [mu(2)
mu(2)+state(1)*sin((mu(3) - (state(2) / 2)))])
       for i = (mu(3) - (state(2) / 2)):0.05:(mu(3) + (state(2) / 2))
           hold on
           scatter(mu(1)+state(1)*cos(i), mu(2)+state(1)*sin(i),.5)
       end
```