Probabilistic Robotics
2D Localization Lab

Emre Ozan Alkan
{emreozanalkan@gmail.com}
MSCV-5

April 3, 2014

1 Introduction

Localization is main and hard to achieve task in probabilistic robotics. Localization is the problem of guessing robot's location according to external reference frame. Specific localization problems like global localization is localizing robot by only given environment map.

1.1 Environment

In this lab, we tried to localize robot by probabilistic method with its own uncertainty. We've given 10 by 10 grid as circular map with values 'r' and 'g' representing red and green respectively. Our robot has sensor which detecting the color of the cell of the grid. It can move up, down, right and left in the map. We also suppose to have measurement data called z:

$$z = [g, g, g, r, g, r, r, r, g, r]$$

Also we suppose to have movement data along with z called u:

$$u = [[01]; [10]; [01]; [10]; [10]; [01]; [01]; [10]; [10]; [01]];$$

Where u values are:

Robot Direction	**Representation**
Left	[0 -1]
Right	[0 1]
Up	[-1 0]
Down	[1 0]
Wait	[0 0]

Since in real world, there is no certainty, we also need to keep track of our uncertainty. Thus, we have $pHit$ representing correct sensor read. Whereas we also have $pMiss$ representing probability of wrong sensor reading. Their values are 0.7 and 0.3, respectively.

On the other hand, robot movement is also not perfect in real world, hence, we should keep track of its uncertainty too. So we keep $pMove, pUndershootandpOvershoot$ with respective 0.8, 0.1 and 0.1 probabilities, representing probabilities of correct movement, less movement or more movement.

2 Implementation

We implemented 2 functions in Matlab. Sense function - takes prior probability and measurement and find posterior probability, Move function - taking prior, motion and uncertainty as input and find prediction to update measurements. Here are implementations:

Listing 1: Main.m

```

close all;
clear all;
clc;

worldWidth = 10;
worldHeight = 10;

world = [ 'r', 'g', 'g', 'r', 'r', 'r', 'g', 'g', 'r', 'r';
          'r', 'r', 'g', 'r', 'r', 'r', 'r', 'g', 'r', 'r';
          'r', 'r', 'g', 'g', 'r', 'r', 'r', 'g', 'g', 'r';
          'r', 'r', 'r', 'r', 'r', 'r', 'r', 'r', 'r', 'r';
          'g', 'g', 'g', 'g', 'g', 'g', 'g', 'g', 'g', 'g';
          'r', 'g', 'g', 'r', 'r', 'r', 'g', 'g', 'r', 'r';
          'r', 'r', 'g', 'r', 'r', 'r', 'r', 'g', 'r', 'r';
          'r', 'r', 'g', 'g', 'r', 'r', 'r', 'g', 'g', 'r';
          'r', 'r', 'r', 'r', 'r', 'r', 'r', 'r', 'r', 'r';
          'g', 'g', 'g', 'g', 'g', 'g', 'g', 'g', 'g', 'g'];

worldProbabilty = ones(worldWidth, worldHeight) ./ (worldWidth * worldHeight);
imagesc(worldProbabilty); title('Prior Probability of World');

z = ['g', 'g', 'g', 'r', 'g', 'r', 'r', 'r', 'g', 'r'];

pHit = 0.7;
pMiss = 0.3;

pMove = 0.8;
pUndershoot = 0.1;
pOvershoot = 0.1;

u = [[0 1];
     [1 0];
     [0 1];
     [1 0];
     [1 0];
     [0 1];
     [0 1];
     [1 0];
     [1 0];
     [0 1]];


pathCount = numel(z);

for ii = 1 : pathCount

    worldProbabilty = sense(worldProbabilty, z(ii), pHit, pMiss, world);

    worldProbabilty = move(worldProbabilty, u(ii, :), pMove, pOvershoot, pUndershoot);

end

pause(2);

imagesc(worldProbabilty); title('Posterior Probability of World');
```

Listing 2: sense.m

```matlab

function [ q ] = sense(p, z, pHit, pMiss, world)

[m, n] = size(p);

q = ones(m, n);

for ii = 1 : m

    for jj = 1 : n

        if strcmp(world(ii, jj), z)
            q(ii, jj) = pHit * p(ii, jj);
        else
            q(ii, jj) = pMiss * p(ii, jj);
        end

    end

end

q = q / sum(q(:));

end
```

Listing 3: move.m

```matlab

function [ q ] = move(p, u, pCorrect, pOvershooting, pUndershooting)

qCorrect = pCorrect * circshift(p, u);

qOvershooting = pOvershooting * circshift(p, u + 1);

qUndershooting = pUndershooting * circshift(p, u - 1);

q = qCorrect + qOvershooting + qUndershooting;

end
```

3 Results

Since world is 10 by 10 grid, contains 100 elements, all of its grid elements were having 0.01 probability initially. After robot starts to move, with sensing and moving, it tries to locate its belief, uncertainty, into more compound in other word less uncertain condition. After running the code, here is the results of the 10 by 10 map, indicating 4 different peaks that our robot more likely to be. Since this map is recurrent, of 4 individual square, robot is more likely to be in one of the peak of this 4 individual repetitive patterns.

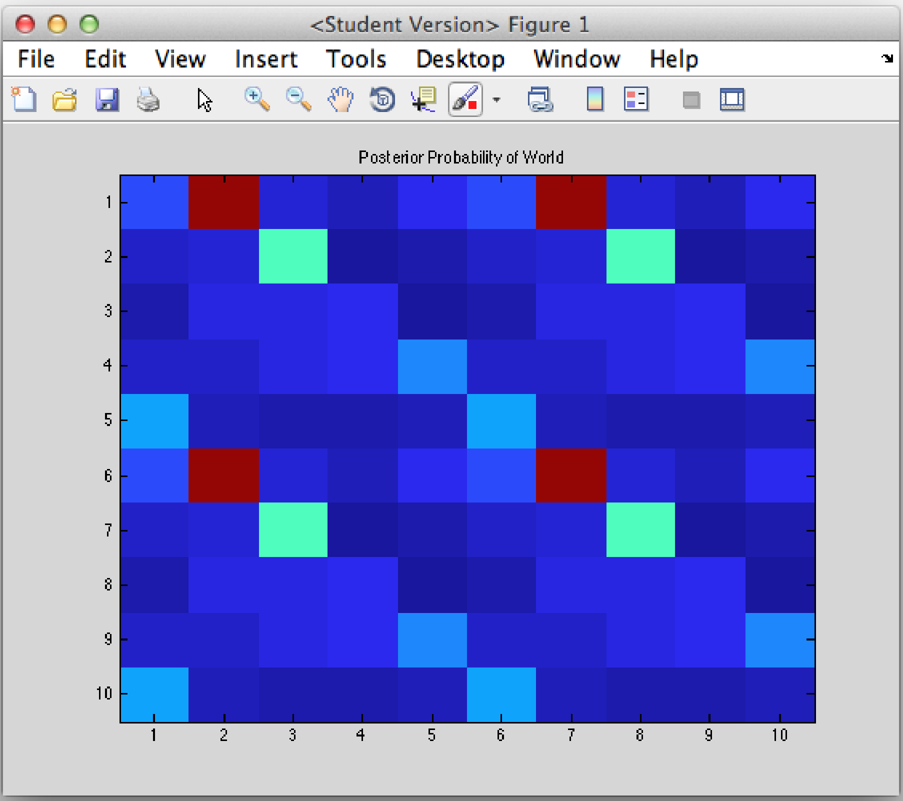

Figure 1: Posterior Probability of World

4 References

1. http://www.probabilistic-robotics.org/