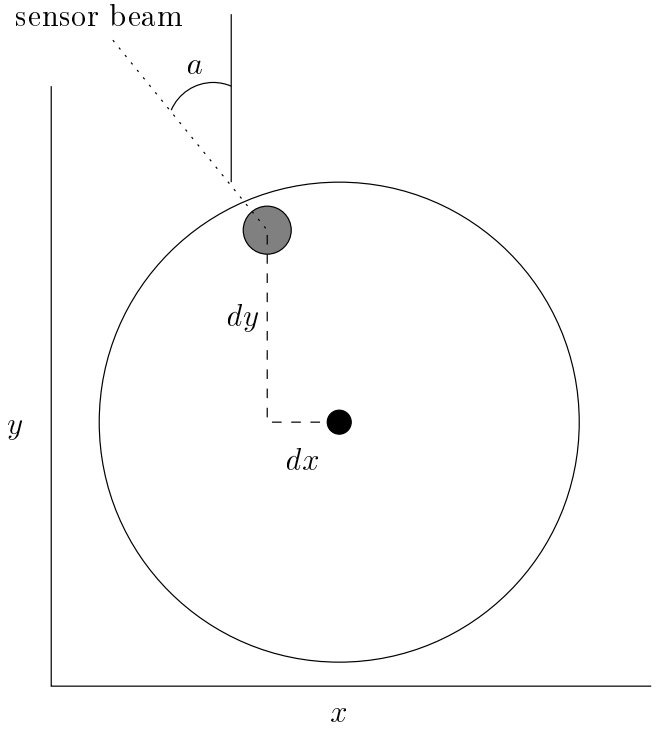
Probabilistic Robotics

Programming Assignment 1

January 29, 2016

The goal of this assignment is to write a range sensor model. The sensor model calculates $p(s|z,m)$ where s is the robot state or pose expressed as the tuple $\langle x, y, \theta \rangle$, z is an observation derived from sensor data, and m is an occupancy grid map. In this case, z is a range reading from a sensor which is mounted somewhere on the robot. For localization to work properly, it is necessary to account for the position of the sensor with respect to the center of the robot, as well as the angle of the sensor beam with respect to the front of the robot. The following figure shows the situation:



In this figure, the large circle represents the robot and the smaller grey circle represents a generic distance sensor unit. dx and dy give the location of the sensor beam receiver in robot-centric coordinates, and α gives the angle of the beam with respect to the front of the robot. An alpha value of zero means that the sensor beam projects directly ahead of the robot, while a value of ninety means that the beam is directed to the left, orthogonal to the front of the robot.

You are to write a function to calculate $p(s|z,m)$, while taking all of this data into account. The function prototype is:

```
float sensormodel(MapStruct *map,
                  int res,
                  int row,int col,
                  int theta,
                  int dx, int dy,
                  int a,int r);
```

where `r` is the range reading along the sensor beam. Note that s is actually the tuple $\langle$`row`, `col`, `theta`$\rangle$ and z is the tuple $\langle$`dx`, `dy`, `a`, `r`$\rangle$. The map is a 2D array of unsigned 8-bit integers, and a number, `res`, indicating the size of each cell. If `res` is 100, then each cell in the map is 100 mm by 100 mm. A cell value of 0 indicates that a cell is occupied with probability 1, while a value of 255 indicates that the cell is occupied with probability 0.

I have written a program to load everything and test your function. You just have to edit `sensormodel.cc` and write the function. `MapStruct` is defined in `map.h`. You may need to write some helper functions. You will definitely need a function to simulate the sensor beam. Its prototype will look something like this:

```
int simulate_beam(MapStruct *map,int res,int x,int y,double angle);
```

where `x` and `y` are the position of the sensor in map coordinates, and angle is the direction of the beam. It should return a result in millimeters.