CMSC828F Project 2:

Localization procedure with Particle Filters.

1.

The first part is a simulation. Implement the particle filter for estimating the state (position (x,y) and orientation (θ)) as described in Chapter 9 of Principles of Robot Motion (hand-out). You are given a map. The robot has 5 sensors that estimate depth. One is mounted in front (i.e. pointing forward), and two on each side, with one perpendicular to the wheels and the other at an angle of 25 degrees to the perpendicular one.

The particle filter is updated at discrete time intervals. The robot moves with the strategy described below. Use Algorithm 17 and 18 to obtain the state of particles and their weighting factor.

The motion prediction is computed by first integrating the motion between two particle updates, and then randomizing it as described in equations (9.18) – (9.21).

As sensor model p(y|x) use a simple Gaussian distribution. The output is 2 graphical displays

a.) the map with the path of the robot overlaid

b.) the map with the (current) estimated location (x,y) of the particles.

Motion of the robot: The motion is a linear combination of a high-level planned motion plus an obstacle avoidance mechanism.

That is,

motion = α * planned motion + (1-α) * obstacle avoidance motion.

How to compute the planned motion: You are given a planned path. Draw a particle from the current set of particles with a distribution as coded by their weights and whose depth values are acceptable. That is, the weights are normalized to 1. Simply choose a random number r in [0..1], and find the particle p_n with weight w_n such that $\sum_{i=1}^{n} w_i = r$. If 3 out of the 5 expected depth values for this particle are in agreement with the depth from the IR sensors, keep the particle. Otherwise draw again till you find a particle that is in agreement. Then find the planned motion for this particle (from the planned path).

For obstacle avoidance compute, as in Project 1, the motion to avoid collision, and keep a certain distance from the wall..

Experiment with the weights α. In the beginning there is large uncertainty about the location of the robot. Thus, the randomly chosen particle carries little information. As the filter progresses, the particles will converge to a cloud and the chosen particle becomes more important. Increase α, as the variance of the x- and y- positions decreases.

2.

Both in simulation and on the ER1 implement the particle filter for localization employing the IR sensors and Vision using a simple correlation procedure.

You will be given a map of the Vision lab (in the 4^{th} floor of A.V. Williams. Bldg). The sensor measurements are the five IR measurements plus two images. The IR measurements only work within a range of about 0.1m - 1m. For larger depth values they should be ignored. Two cameras will be mounted, one on the left and one on the right.

Image measurements: We code the probability of seeing the correct amount of door:
We create a 1-D image of what the camera should be seeing from the given position and orientation. This 1-D image is a binary image, containing 1 for every column, that should be a door (brown), and 0 for all other columns.
The 1-D image from the camera is obtained by first turning the actual image into a binary one, brown and not brown. Then for every column we count the brown pixels, we normalize the count, and create the 1-d image, which codes for every x-coordinate the number of brown pixels.
The image sensor function $p(y|x)$ is then obtained as normalized cross-correlation.

The complete sensor function should be a linear combination of the two images plus the product of the IR sensor data measurement (only those which are useful). As IR sensor model use the Gaussian uncertainty plus an exponential function for an unexpected obstacle (pg 324 of hand-out).

Motion of the robot:
The motion is, as before, a linear combination:

$$\text{motion} = \alpha * \text{planned motion} + (1-\alpha) * \text{obstacle avoidance motion.}$$

To obtain the planned motion, keep drawing particles till you find a particle, such that one of the expected images are close to the observed images (deviation is above some threshold).

The obstacle avoidance should keep a certain distance (e.g 0.5 m) from the wall. Doors are 0.1m farther away than walls. If, a door is detected in the images, increase the distance.