Albert-Ludwigs-Universität Freiburg
Lecture: Introduction to Mobile Robotics
Summer term 2013

Institut für Informatik

Prof. Dr. Wolfram Burgard
PD Dr. Cyrill Stachniss
Juniorprof. Dr. Maren Bennewitz
Dr. Gian Diego Tipaldi
Dr. Luciano Spinello

Sheet 7

Topic: Velocity Motion Model, Particle Filter

Submission deadline: June 17, 2013
Submit to: `mobilerobotics@informatik.uni-freiburg.de`

Exercise 1: Velocity-Based Motion Model

Remark: This exercise is to be solved without Octave.

Consider a robot which moves on a circular trajectory with noise-free constant velocities (v, w) (this situation is shown on page 30 of the *Probabilistic Motion Models* slides). The current pose of the robot is (x, y, θ).

(a) Derive the following expression for the center of the circle, (x_c, y_c):

$$\begin{pmatrix} x_c \\ y_c \end{pmatrix} = \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} -\frac{v}{w} sin\theta \\ \frac{v}{w} cos\theta \end{pmatrix}$$

(b) Now consider the situation where we are given a start pose (x, y, θ) and an end pose (x', y', θ'), connected by a circular movement. Prove that the center of the circle can be expressed as

$$\begin{pmatrix} x_c \\ y_c \end{pmatrix} = \frac{1}{2} \begin{pmatrix} x + x' \\ y + y' \end{pmatrix} + \mu \begin{pmatrix} y - y' \\ x' - x \end{pmatrix} \tag{1}$$

with some $\mu \in \mathbb{R}$

Hint: The circle lies on a ray that lies on the half-way point between (x, y) and (x', y') and is orthogonal to the line between these coordinates. Use the parametric equation for a line to represent this ray.

(c) Show that the value of μ is given by

$$\mu = \frac{1}{2} \frac{(x - x')cos\theta + (y - y')sin\theta}{(y - y')cos\theta - (x - x')sin\theta}.$$

Hint: μ can be calculated by using the fact that the line described by equation (1) and the line from (x_c, y_c) to (x, y) intersect at (x_c, y_c).

Exercise 2: Particle Filter Update

Complete the function file `measurement_model.m`, which you can download along with this sheet. This function should implement the update step of a particle filter, using a *range-only* sensor.

It takes as input a set l of landmarks, a set z of *independent* landmark observations and a set x of particles.

- l: A struct array representing a landmark map of the environment, where each landmark $l(i)$ has an id $l(i).id$ and a position $l(i).x, l(i).y$.
- z: A struct array containing a number of landmark observations, where each observation $z(i)$ has an id $z(i).id$ and a range $z(i).range$.
- x: A matrix of size $N \times 3$, where N is the number of particles, $x(:,1)$ represents the x-coordinate and $x(:,2)$ the y-coordinate of each particle. The orientation $x(:,3)$ is not used in this exercise, but will be of importance on the next exercise sheet where a complete particle filter is implemented.

It should return a vector of weights that has the same size as the number of particles. See slide 15 of the particle filter lecture for the definition of the weight w. Instead of computing a probability, it is sufficient to compute the likelihood $p(z|x, l)$. The measurement standard deviation is $\sigma_r = 0.2$. Try to avoid loops by using matrix operations where possible. You can test your implementation by executing the script `test_measurement_model.m`.

Hint: The template for `measurement_model.m` already shows how to get a landmark with a certain id.