Albert-Ludwigs-Universität Freiburg
Lecture: Introduction to Mobile Robotics
Summer term 2008

Institut für Informatik

Prof. Dr. W. Burgard
Dipl.-Inf. B. Frank
Dipl.-Inf. D. Joho
MSc ACS H. Strasdat

Sheet 4

Topic: Particle Filter

Submission deadline: Tue 3.6.2008, 11:00 a.m. (before class)

Exercise 1:

(a) Which of the following functions $g(x)$ are valid proposal functions for arbitrary distributions $p(x)$ on the interval $x \in [-2, 2]$? Give a reason for your decision.

(i) $g(x) = 5x^2 + 1$

(ii) $g(x) = 5x^2$

(iii) $g(x) = 2$

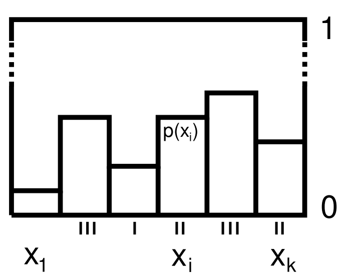
(iv) $g(x) = \sin(2x) + 1$

(v) $g(x) = \sin(2x) + 3$

(b) Explain in short the purpose of a proposal function. What is the benefit of using a proposal function compared to the usage of rejection sampling?

Exercise 2:

Consider rejection sampling for a discrete probability distribution p: We are given k states $x_1, \dots, x_k$ with associated probabilities $p(x_1), \dots, p(x_k)$.



We will use N samples. Let $c(x_i) \in \{0, \dots, N\}$ be the number of *accepted* (!) samples for state x_i. Prove that the expected probability mass

$$\tilde{p}(x_i) = \frac{E(c(x_i))}{\sum_{j=1}^{k} E(c(x_j))}$$

assigned to state x_i by rejection sampling equals the true probability $p(x_i)$:

$$\forall i \in \{1, \dots, k\} : \frac{E(c(x_i))}{\sum_{j=1}^{k} E(c(x_j))} = p(x_i).$$

Exercise 3:

Programming task: particle filtering A simulated robot can be moved through a 2D environment (using the keys “a”, “q”, “w”, “e” and “d”). There are three landmarks in the environment. If a landmark is visible, the robot can measure the range ρ and bearing ϕ to it. Track the pose (x, y, θ) of the robot using a particle filter: Complete the stubs in the class ParticleFilter. Thus, implement the particle sampling, the calculation of the importance weights, the normalization and the re-sampling. Use the following motion and sensor model:

Motion model: The forward translation δ_{trans} and the rotation δ_{rot1} are given. Use the odometry model with $\alpha = (0.05, 0.1, 0.1, 0.05)$. Assume that $\delta_{rot2} = 0$ all the time.

Sensor model: Use a Gaussian sensor model with $\sigma_\rho^2 = 1$ and $\sigma_\phi^2 = 0.25$. Remark: It is not necessary but you might want to use the classes CarmenMatrix2D and CarmenPoint2D.

Attention: After angle-operations, normalize the result between $-\pi$ and π. Therefore, use the static method *normalizeAngle* of the CarmenPoint class.