25. Robotics

25.1 [mcl-biasdness-exercise]Monte Carlo localization is *biased* for any finite sample size—i.e., the expected value of the location computed by the algorithm differs from the true expected value—because of the way particle filtering works. In this question, you are asked to quantify this bias.

To simplify, consider a world with four possible robot locations: $X = \{x_1, x_2, x_3, x_4\}$. Initially, we draw $N \geq 1$ samples uniformly from among those locations. As usual, it is perfectly acceptable if more than one sample is generated for any of the locations X. Let Z be a Boolean sensor variable characterized by the following conditional probabilities:

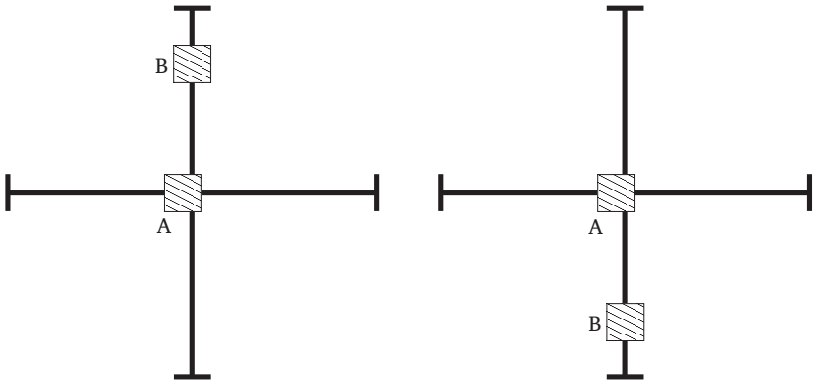
$$\begin{aligned} P(z \mid x_1) &= 0.8 & P(\neg z \mid x_1) &= 0.2 \\ P(z \mid x_2) &= 0.4 & P(\neg z \mid x_2) &= 0.6 \\ P(z \mid x_3) &= 0.1 & P(\neg z \mid x_3) &= 0.9 \\ P(z \mid x_4) &= 0.1 & P(\neg z \mid x_4) &= 0.9 \ . \end{aligned}$$

MCL uses these probabilities to generate particle weights, which are subsequently normalized and used in the resampling process. For simplicity, let us assume we generate only one new sample in the resampling process, regardless of N. This sample might correspond to any of the four locations in X. Thus, the sampling process defines a probability distribution over X.

1. What is the resulting probability distribution over X for this new sample? Answer this question separately for $N = 1, \ldots, 10$, and for $N = \infty$.
2. The difference between two probability distributions P and Q can be measured by the KL divergence, which is defined as
$$KL(P, Q) = \sum_i P(x_i) \log \frac{P(x_i)}{Q(x_i)} \ .$$
What are the KL divergences between the distributions in (a) and the true posterior?
3. What modification of the problem formulation (not the algorithm!) would guarantee that the specific estimator above is unbiased even for finite values of N? Provide at least two such modifications (each of which should be sufficient).

25.2 [mcl-implement-exercise]Implement Monte Carlo localization for a simulated robot with range sensors. A grid map and range data are available from the code repository at aima.cs.berkeley.edu (http://aima.cs.berkeley.edu). You should demonstrate successful global localization of the robot.

Figure [figRobot2] A Robot manipulator in two of its possible configurations.



Starting configuration <–0.5, 7> Ending configuration <–0.5, –7>

25.3 [AB-manipulator-ex] Consider a robot with two simple manipulators, as shown in figure figRobot2. Manipulator A is a square block of side 2 which can slide back and on a rod that runs along the x-axis from x=−10 to x=10. Manipulator B is a square block of side 2 which can slide back and on a rod that runs along the y-axis from y=-10 to y=10. The rods lie outside the plane of manipulation, so the rods do not interfere with the movement of the blocks. A configuration is then a pair $\langle x, y \rangle$ where x is the x-coordinate of the center of manipulator A and where y is the y-coordinate of the center of manipulator B. Draw the configuration space for this robot, indicating the permitted and excluded zones.

25.4 Suppose that you are working with the robot in Exercise AB-manipulator-ex and you are given the problem of finding a path from the starting configuration of figure figRobot2 to the ending configuration. Consider a potential function

$$D(A, Goal)^2 + D(B, Goal)^2 + \frac{1}{D(A, B)^2}$$

where $D(A, B)$ is the distance between the closest points of A and B.

1. Show that hill climbing in this potential field will get stuck in a local minimum.
2. Describe a potential field where hill climbing will solve this particular problem. You need not work out the exact numerical coefficients needed, just the general form of the solution. (Hint: Add a term that “rewards" the hill climber for moving A out of B’s way, even in a case like this where this does not reduce the distance from A to B in the above sense.)

25.5 [inverse-kinematics-exercise]Consider the robot arm shown in Figure FigArm1. Assume that the robot’s base element is 60cm long and that its upper arm and forearm are each 40cm long. As argued on page inverse-kinematics-not-unique, the inverse kinematics of a robot is often not unique. State an explicit closed-form solution of the inverse kinematics for this arm. Under what exact conditions is the solution unique?

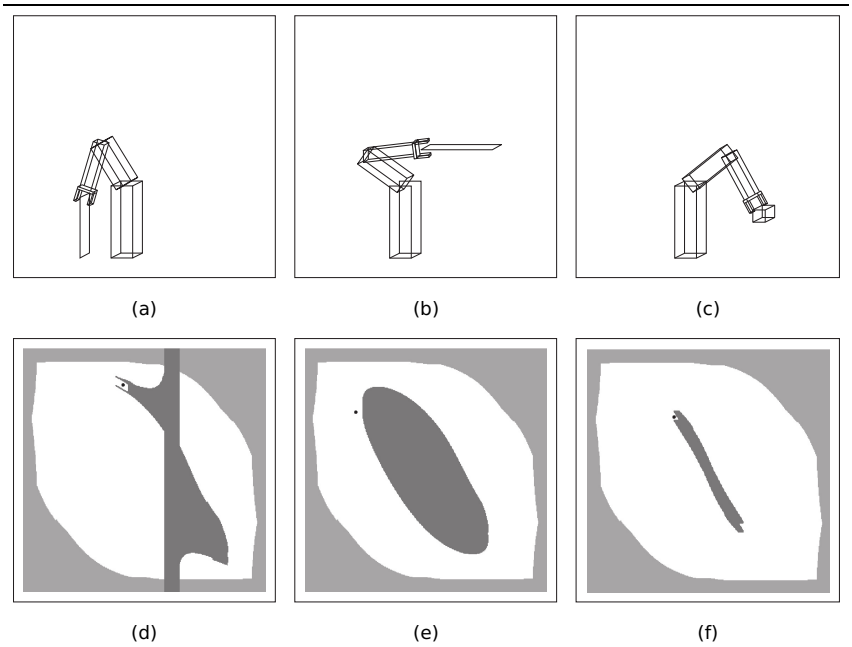
25.6 [inverse-kinematics-exercise]Consider the robot arm shown in Figure FigArm1. Assume that the robot's base element is 70cm long and that its upper arm and forearm are each 50cm long. As argued on page inverse-kinematics-not-unique, the inverse kinematics of a robot is often not unique. State an explicit closed-form solution of the inverse kinematics for this arm. Under what exact conditions is the solution unique?

25.7 [voronoi-exercise]Implement an algorithm for calculating the Voronoi diagram of an arbitrary 2D environment, described by an $n \times n$ Boolean array. Illustrate your algorithm by plotting the Voronoi diagram for 10 interesting maps. What is the complexity of your algorithm?

25.8 [confspace-exercise]This exercise explores the relationship between workspace and configuration space using the examples shown in Figure FigEx2.

1. Consider the robot configurations shown in Figure FigEx2(a) through (c), ignoring the obstacle shown in each of the diagrams. Draw the corresponding arm configurations in configuration space. (*Hint:* Each arm configuration maps to a single point in configuration space, as illustrated in Figure FigArm1(b).)
2. Draw the configuration space for each of the workspace diagrams in Figure FigEx2(a)–(c). (*Hint:* The configuration spaces share with the one shown in Figure FigEx2(a) the region that corresponds to self-collision, but differences arise from the lack of enclosing obstacles and the different locations of the obstacles in these individual figures.)
3. For each of the black dots in Figure FigEx2(e)–(f), draw the corresponding configurations of the robot arm in workspace. Please ignore the shaded regions in this exercise.
4. The configuration spaces shown in Figure FigEx2(e)–(f) have all been generated by a single workspace obstacle (dark shading), plus the constraints arising from the self-collision constraint (light shading). Draw, for each diagram, the workspace obstacle that corresponds to the darkly shaded area.
5. Figure FigEx2(d) illustrates that a single planar obstacle can decompose the workspace into two disconnected regions. What is the maximum number of disconnected regions that can be created by inserting a planar obstacle into an obstacle-free, connected workspace, for a 2DOF robot? Give an example, and argue why no larger number of disconnected regions can be created. How about a non-planar obstacle?

Figure [FigEx2] Diagrams for Exercise [confspace-exercise](#/).



25.9 Consider a mobile robot moving on a horizontal surface. Suppose that the robot can execute two kinds of motions:

- Rolling forward a specified distance.
- Rotating in place through a specified angle.

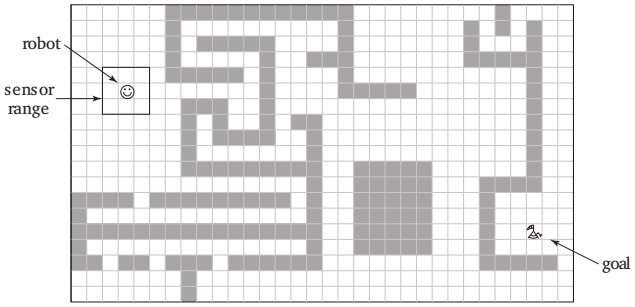
The state of such a robot can be characterized in terms of three parameters $\langle x, y, \phi$, the x-coordinate and y-coordinate of the robot (more precisely, of its center of rotation) and the robot's orientation expressed as the angle from the positive x direction. The action "$Roll(D)$" has the effect of changing state $\langle x, y, \phi$ to $\langle x + D\cos(\phi), y + D\sin(\phi), \phi\rangle$, and the action $Rotate(\theta)$ has the effect of changing state $\langle x, y, \phi\rangle$ to $\langle x, y, \phi + \theta\rangle$.

1. Suppose that the robot is initially at $\langle 0, 0, 0\rangle$ and then executes the actions $Rotate(60^\circ)$, $Roll(1)$, $Rotate(25^\circ)$, $Roll(2)$. What is the final state of the robot?
2. Now suppose that the robot has imperfect control of its own rotation, and that, if it attempts to rotate by θ, it may actually rotate by any angle between $\theta - 10^\circ$ and $\theta + 10^\circ$. In that case, if the robot attempts to carry out the sequence of actions in (A), there is a range of possible ending states. What are the minimal and maximal values of the x-coordinate, the y-coordinate and the orientation in the final state?

3. Let us modify the model in (B) to a probabilistic model in which, when the robot attempts to rotate by θ, its actual angle of rotation follows a Gaussian distribution with mean θ and standard deviation 10°. Suppose that the robot executes the actions $Rotate(90^\circ)$, $Roll(1)$. Give a simple argument that (a) the expected value of the location at the end is not equal to the result of rotating exactly 90° and then rolling forward 1 unit, and (b) that the distribution of locations at the end does not follow a Gaussian. (Do not attempt to calculate the true mean or the true distribution.)

   The point of this exercise is that rotational uncertainty quickly gives rise to a lot of positional uncertainty and that dealing with rotational uncertainty is painful, whether uncertainty is treated in terms of hard intervals or probabilistically, due to the fact that the relation between orientation and position is both non-linear and non-monotonic.

Figure [FigEx3] Simplified robot in a maze. See Exercise [robot-exploration-exercise](#/)



25.10 [robot-exploration-exercise]Consider the simplified robot shown in Figure FigEx3. Suppose the robot's Cartesian coordinates are known at all times, as are those of its goal location. However, the locations of the obstacles are unknown. The robot can sense obstacles in its immediate proximity, as illustrated in this figure. For simplicity, let us assume the robot's motion is noise-free, and the state space is discrete. Figure FigEx3 is only one example; in this exercise you are required to address all possible grid worlds with a valid path from the start to the goal location.

1. Design a deliberate controller that guarantees that the robot always reaches its goal location if at all possible. The deliberate controller can memorize measurements in the form of a map that is being acquired as the robot moves. Between individual moves, it may spend arbitrary time deliberating.
2. Now design a *reactive* controller for the same task. This controller may not memorize past sensor measurements. (It may not build a map!) Instead, it has to make all decisions based on the current measurement, which includes knowledge of its own location and that of the goal. The time to make a decision must be independent of the environment size or the number of past time steps. What is the maximum number of steps that it may take for your robot to arrive at the goal?
3. How will your controllers from (a) and (b) perform if any of the following six conditions apply: continuous state space, noise in perception, noise in motion, noise in both perception and motion, unknown location of the goal (the goal can be detected only when within sensor range), or moving obstacles. For each condition and each controller, give an example of a situation where the robot fails (or explain why it cannot fail).

25.11 [subsumption-exercise]In Figure Fig5(b) on page Fig5, we encountered an augmented finite state machine for the control of a single leg of a hexapod robot. In this exercise, the aim is to design an AFSM that, when combined with six copies of the individual leg controllers, results in efficient, stable locomotion. For this purpose, you have to augment the individual leg controller to pass messages to your new AFSM and to wait until other messages arrive. Argue why your controller is efficient, in that it does not unnecessarily waste energy (e.g., by sliding legs), and in that it propels the robot at reasonably high speeds. Prove that your controller satisfies the dynamic stability condition given on page polygon-stability-condition-page.

25.12 [human-robot-exercise](This exercise was first devised by Michael Genesereth and Nils Nilsson. It works for first graders through graduate students.) Humans are so adept at basic household tasks that they often forget how complex these tasks are. In this exercise you will discover the complexity and recapitulate the last 30 years of developments in robotics. Consider the task of building an arch out of three blocks. Simulate a robot with four humans as follows:

Brain. The Brain direct the hands in the execution of a plan to achieve the goal. The Brain receives input from the Eyes, but *cannot see the scene directly*. The brain is the only one who knows what the goal is.

Eyes. The Eyes report a brief description of the scene to the Brain: "There is a red box standing on top of a green box, which is on its side" Eyes can also answer questions from the Brain such as, "Is there a gap between the Left Hand and the red box?" If you have a video camera, point it at the scene and allow the eyes to look at the viewfinder of the video camera, but not directly at the scene.

Left hand and **right hand.** One person plays each Hand. The two Hands stand next to each other, each wearing an oven mitt on one hand, Hands execute only simple commands from the Brain—for example, "Left Hand, move two inches forward." They cannot execute commands other than motions; for example, they cannot be commanded to "Pick up the box." The Hands must be *blindfolded*. The only sensory capability they have is the ability to tell when their path is blocked by an immovable obstacle such as a table or the other Hand. In such cases, they can beep to inform the Brain of the difficulty.