Assignment 3: Probabilistic Reasoning

Part 1: Probabilistic Modeling

In this part, we will understand and apply the concepts of probability, independence, and dependence of events using some real-world examples.

Events in the real world often exhibit randomness or uncertainty. In daily life, people implicitly estimate the likelihood of events, such as predicting the weather or anticipating traffic conditions. In AI and machine learning, probabilistic modeling is a crucial tool for handling such uncertainties and making informed decisions based on available data.

Identify two events in your daily life that are random and give an estimation of their probabilities:

[Student fill]

Certain events are considered **independent**, meaning the occurrence of one event does not influence the probability of another. For example, when you roll a dice twice, the outcome of the first roll does not affect the outcome of the second. The probability of rolling a 6 in the second round remains the same, regardless of what you rolled in the first round.

On the other hand, some events are considered **dependent**. In these cases, the probability of one event is influenced by the occurrence of another event. For example, imagine you are planning a picnic. The likelihood of having a picnic outdoors is dependent on the weather. If it's raining, the probability of proceeding with the picnic decreases significantly. In this case, the event "having a picnic" is dependent on the event "weather being good." The occurrence of rain directly impacts the decision to have the picnic.

Let P1 represent the probability of an event A, P2 represent the probability of an event B, and P0 as the probability of both events A and B occurring together. We say that event A is independent of event B if P0 = P1 * P2. If this condition does not hold, then event A is dependent on event B.

Based on the events you identified in the previous questions, are these events independent or dependent? If they are independent, identify two events that are dependent on each other. Conversely, if they are dependent, identify two events that are independent of each other.

[Student fill]

Can you estimate the probability of one event occurring, given that the other event has already happened?

[Student fill]

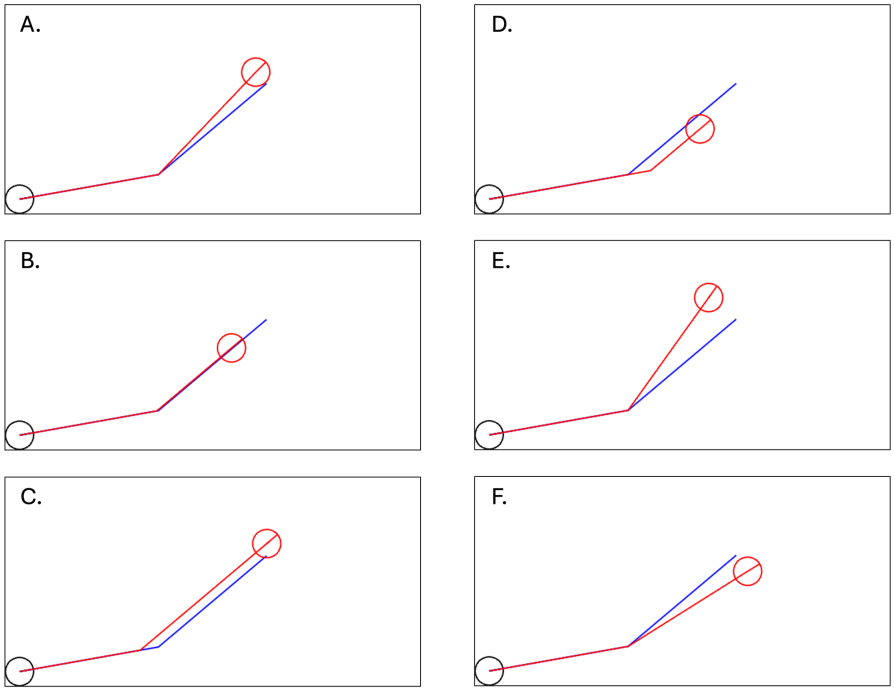
Part 2: Mobile Robot Localization

One intriguing problem in probabilistic reasoning for AI is mobile robot localization. Autonomous mobile robots need an understanding of *where* they are in the environment they are operating in, to perform tasks accurately. The problem of figuring out where the robot is in the environment, is called **mobile robot localization**.

While this may seem like a straightforward task, it is actually quite complex due to the inaccuracies of the sensors the robot uses to perceive the world and the errors in executing movement commands.

Consider a real-world analogy: you need to walk from your bedroom to the front door of your home. You start by facing the bedroom door and plan to take four steps forward, turn left by 30 degrees, and then walk another four steps. This might seem simple, but now imagine doing it in complete darkness. Despite knowing the exact steps, you are likely to bump into something. This is because movements like "walk forward" and "turn around" are not perfectly predictable—they involve some randomness. The distance you move and the angle you turn are not exact.

The figures below illustrate this concept. The blue lines represent the paths when movements are precise, with no randomness involved. In contrast, the red lines show the paths when either "walk forward" or "turn around" include some randomness, though not both at the same time. This randomness causes deviations from the endpoints of the blue lines, demonstrating how small uncertainties can lead to significant changes in the robot's trajectory.



Which figures represent the movement when only “walk forward” has randomness?

[Student answer]

Which figures represent the movement when only “turn around” has randomness?

[Student answer]

From the example above, we’ve seen how actual movement can deviate from a perfect path when randomness is involved. This is where “localization” becomes crucial. Since the robot cannot precisely predict its location due to this randomness, it needs to estimate its actual position using sensors and make adjustments accordingly. Now, let’s explore how sensors assist in estimating a robot's location.

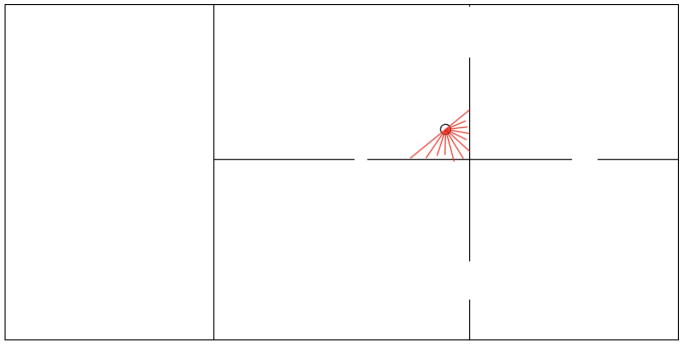
Returning to the walking in darkness analogy, instead of just walking blindly, you would likely reach out with your hands to detect any obstacles in your path. For example, if you don’t reach the door after taking four steps due to random deviations, you would realize this by not feeling the door when you reach out.

A commonly used sensor, LiDAR, operates similarly to “reaching out with your hands.” It emits laser beams and measures the distance by detecting the reflection from obstacles. These laser beams act like “arms,” allowing the robot to detect obstacles by measuring the time it takes for the lasers to bounce back.

In this LiDAR simulator, you can set the robot’s position and heading and visualize the laser beams as red lines radiating from the robot’s center.

Can you adjust the position and heading so that the robot can identify a corner of the room? Take a screenshot of the robot and its LiDAR sensor readings.

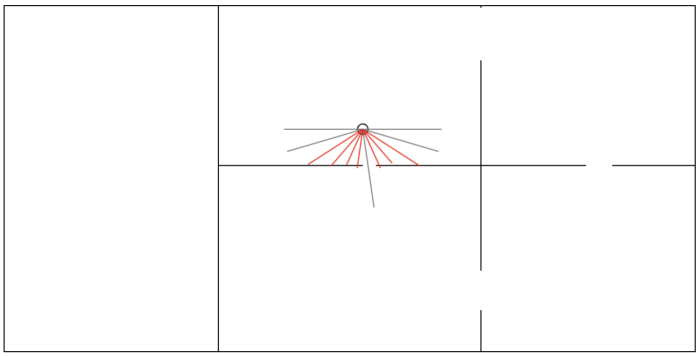
[Student upload a picture:



]

Can you set the position and the heading so that the robot can identify one of the doors of the room? Take a screenshot of the robot and its LiDAR sensor.

[Student upload a picture:



]

Just like humans cannot accurately measure the distance of the obstacles by hands (e.g., the obstacles may be too far to reach), robot sensors also have inaccuracies and limitations. For instance, LiDAR sensors have a maximum range for the laser beams, potential errors in measuring the travel distance, and a limited number of beams they can emit.

Using the scenario from the previous question where the robot observes a door, can you adjust the sensor parameters on the right side of the interface without moving the robot itself, so that the robot no longer detects the door? Once you've done this, take a screenshot of the robot and its LiDAR sensor readings:

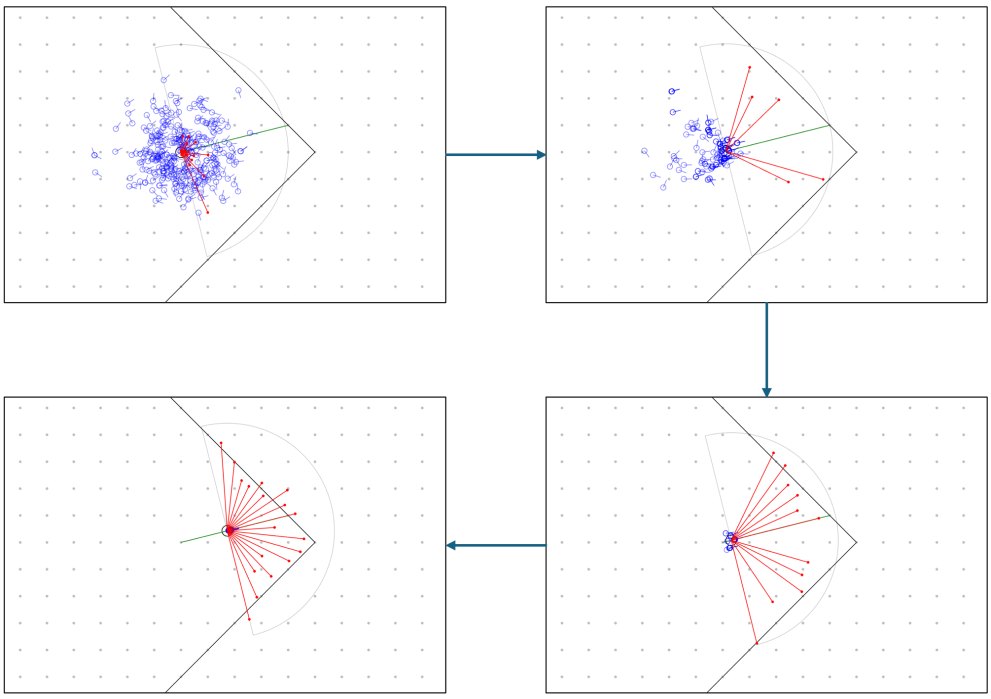
[Student upload a picture]

Part 3: Particle Filter Localization

So far, we have explored how randomness in movement necessitates localization and how sensory information is essential for this process. But how does a robot actually perform localization? While humans can easily interpret tactile information from their hands, robots must rely on algorithms to interpret their sensory data and estimate their current location.

There are numerous algorithms designed to solve the problem of mobile robot localization, one of which is known as Particle Filter Localization, or Monte Carlo Localization (MCL). Particle filters are a class of algorithms that estimate a probability distribution by maintaining multiple hypotheses about the actual state, known as particles. For mobile robot localization, the goal is to determine the robot's location based on sensor readings, possibly combined with the commands sent to the robot and a map of the environment.

As illustrated in the figure below, a robot is commanded to move in a straight line toward a corner. The particle filter begins by sampling random particles represented as blue circles as initial guesses of the current location of the robots.



As the robot approaches the corner, it begins to receive sensor readings and updates its estimation of its current location. Over time, the blue circles, representing the possible locations

of the robot, become more concentrated around its true position. In the final figure, the robot has a high degree of certainty about its location, and the blue circles almost perfectly overlap with the robot's true position.

For a hands-on experience, you can explore an interactive demo here, where you can resample a new set of particles and observe how these particles converge toward the true location of the robot as the robot gathers sensor readings from the wall.

Then, let's explore how movement randomness and sensor capabilities affect mobile robot localization through the following example.

In this demo, the robot is commanded to follow the green path on a building floor, while using a particle filter to estimate its current location. There are four sliders on the right that control key parameters: the magnitude of error from rotation, the magnitude of error from translation, the sensor radius, and sensor noise. However, the current settings are suboptimal, making it difficult for the robot to maintain an accurate estimate of its location.

Can you adjust the parameters so that the robot can accurately estimate its position? Once you've optimized the settings, upload three screenshots of the robot at different locations, showing the blue circles (representing the estimated locations) closely centered around the robot with small variance:

[Student upload three image]

In this demo, the robot is commanded to follow a zig-zag path in a corridor. Again, the current parameters of the particle filter are suboptimal. Can you adjust only one of the parameters to improve the robot's localization accuracy? Once you've made the adjustment, upload a screenshot of the robot at the end of the path, showing the improved estimation:

[Student upload an image]