Autonomous Mobile Robotics

MECH650-EECE698

Particle Filter Exercise

Nicolas Abboud

October 13, 2025

1 Introduction and Problem Setup

In this exercise, you will work through **one complete iteration** of the particle filter algorithm by hand. This will help you understand the intricacies of:

- Particle representation of belief distributions
- Sampling from motion models (prediction step)
- Importance weighting using measurement likelihoods
- Resampling to focus computational resources
- Non-parametric state estimation

1.1 Robot and Environment Description

Consider a mobile robot operating in a **2D environment** with the following properties:

- **State Space**: Robot position (x, y) where $x, y \in [0, 4]$ meters (continuous space)
- **Landmark**: A unique landmark located at position $(L_x, L_y) = (2.5, 2.5)$ meters
- **Sensor**: Range sensor that measures distance r to the landmark
- **Motion**: Robot moves with control input $\mathbf{u}_t = (\Delta x, \Delta y)$

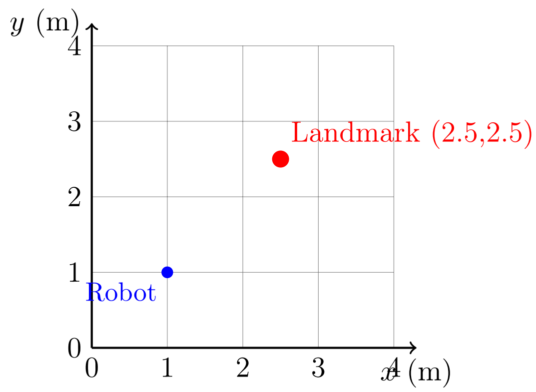


Figure 1: 2D environment with landmark at (2.5, 2.5) meters.

2 Mathematical Framework

2.1 Particle Filter Algorithm

The particle filter represents the belief $\text{bel}(x_t)$ using a set of M particles:

$$\mathcal{X}_t = \{x_t^{[1]}, x_t^{[2]}, \ldots, x_t^{[M]}\} \text{ with weights } \{w_t^{[1]}, w_t^{[2]}, \ldots, w_t^{[M]}\}$$

where each particle $x_t^{[m]} = (x^{[m]}, y^{[m]})$ represents a hypothesis about the robot's position.

2.2 Algorithm Steps (from Probabilistic Robotics textbook)

Algorithm Particle_filter(X_{t-1}, u_t, z_t):

1. $\bar{\mathcal{X}}_t = \mathcal{X}_t = \emptyset$
2. **for** $m = 1$ to M **do**
3. **Sample** $x_t^{[m]} \sim p(x_t|u_t, x_{t-1}^{[m]})$ *(Prediction)*
4. $w_t^{[m]} = p(z_t|x_t^{[m]})$ *(Importance weighting)*
5. $\bar{\mathcal{X}}_t = \bar{\mathcal{X}}_t + \langle x_t^{[m]}, w_t^{[m]} \rangle$
6. **endfor**
7. **for** $m = 1$ to M **do**
8. **draw** i with probability $\propto w_t^{[i]}$ *(Resampling)*
9. **add** $x_t^{[i]}$ to $\mathcal{X}_t$
10. **endfor**
11. **return** $\mathcal{X}_t$

Understanding the algorithm structure:

- $\bar{\mathcal{X}}_t$ = **temporary particle set** with importance weights (lines 3-6: after prediction and measurement weighting)
- $\mathcal{X}_t$ = **final resampled particle set** with uniform weights (lines 8-10: after resampling)
- The weighted particles in $\bar{\mathcal{X}}_t$ represent the belief incorporating the measurement
- The resampled particles in $\mathcal{X}_t$ represent the same belief but with uniform weights and particles concentrated in high-probability regions
- Both sets approximate the posterior belief $\text{bel}(x_t) = \eta \cdot p(z_t|x_t)\overline{\text{bel}}(x_t)$, just in different representations

3 Models and Parameters

3.1 Motion Model

The robot attempts to move according to control input $\mathbf{u}_t = (\Delta x, \Delta y)$, but the motion is noisy. We model this as:

$$\mathbf{x}_t^{[m]} = \mathbf{x}_{t-1}^{[m]} + \mathbf{u}_t + \boldsymbol{\epsilon}^{[m]}$$

where $\boldsymbol{\epsilon}^{[m]} = (\epsilon_x^{[m]}, \epsilon_y^{[m]})$ is a motion noise term specific to each particle.

Applied component-wise:

$$x_t^{[m]} = x_{t-1}^{[m]} + u_{t,x} + \epsilon_x^{[m]}$$
$$y_t^{[m]} = y_{t-1}^{[m]} + u_{t,y} + \epsilon_y^{[m]}$$

For this exercise, to make hand calculations tractable, each particle is assigned a **predetermined noise value** (simulating random sampling from the motion noise distribution). The specific noise values for each particle will be provided in the problem.

3.2 Sensor Model

The range sensor measures distance to the landmark with Gaussian noise:

$$r_{\text{measured}} = r_{\text{true}} + \nu$$

where $r_{\text{true}} = \sqrt{(x - L_x)^2 + (y - L_y)^2}$ and $\nu \sim \mathcal{N}(0, \sigma_z^2)$ with $\sigma_z = 0.5$ meters.

The likelihood function is:

$$p(z_t | x_t^{[m]}) = \frac{1}{\sqrt{2\pi\sigma_z^2}} \exp\left(-\frac{(z_t - r^{[m]})^2}{2\sigma_z^2}\right)$$

For simplicity, we'll calculate **unnormalized likelihoods** proportional to this.

Notation for weights:

- $\tilde{w}_t^{[m]}$ = **unnormalized** importance weight (proportional to likelihood)
- $w_t^{[m]}$ = **normalized** weight where $\sum_{m=1}^{M} w_t^{[m]} = 1$
- η = normalization constant: $\eta = 1/\sum_{i=1}^{M} \tilde{w}_t^{[i]}$

The relationship is: $w_t^{[m]} = \eta \cdot \tilde{w}_t^{[m]} = \frac{\tilde{w}_t^{[m]}}{\sum_{i=1}^{M} \tilde{w}_t^{[i]}}$

3.3 Task Overview

Task	Description	Status	Focus
1	Initialize particles	Given	Setup
2	Predict with motion model	Given	Setup
3a	Calculate distances & errors	Given	Setup
3b	Calculate unnormalized weights	**TODO**	**Core PF**
4	**Weight normalization**	**TODO**	**Core PF**
5	**Systematic resampling**	**TODO**	**Core PF**
6	State estimation	**TODO**	Application
7	Analysis & understanding	**TODO**	Reflection

Tasks 4 & 5 represent the key innovations that distinguish particle filters from simple Monte Carlo sampling.

4 Task 1: Initial Particle Set (Given)

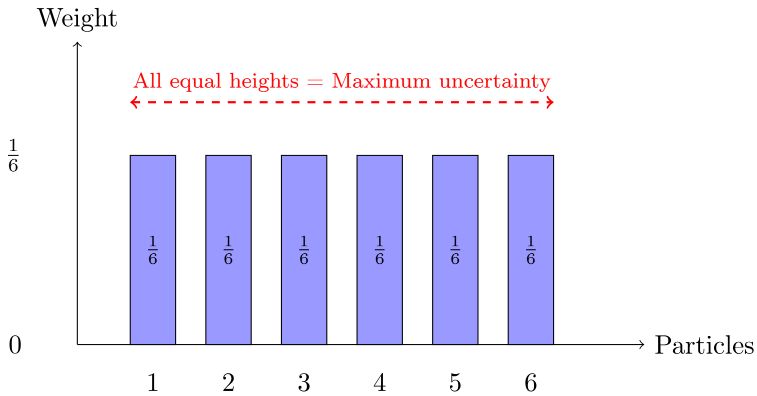
At time $t = 0$, the robot has significant uncertainty about its position. We initialize $M = 6$ particles uniformly distributed across the state space.

Initial particle set (provided):

Particle	$x_0^{[m]}$ **(m)**	$y_0^{[m]}$ **(m)**	$w_0^{[m]}$
$m = 1$	0.5	0.5	$1/6 = 0.1667$
$m = 2$	1.5	1.0	$1/6 = 0.1667$
$m = 3$	2.0	2.0	$1/6 = 0.1667$
$m = 4$	3.5	1.5	$1/6 = 0.1667$
$m = 5$	1.0	3.0	$1/6 = 0.1667$
$m = 6$	3.0	0.5	$1/6 = 0.1667$

Note: All particles have uniform weights $w_0^{[m]} = 1/6$ because we have no prior information about the robot's position. This represents maximum uncertainty. The weights sum to 1: $\sum_{m=1}^{6} w_0^{[m]} = 1$ ✓

4.1 Visualization: Uniform Weight Distribution



Equal weights mean all particle hypotheses are equally likely before any measurements.

5 Task 2: Prediction Step (Given)

The robot executes control command $\mathbf{u}_1 = (+1.0, +1.0)$ meters (move northeast by 1 meter in each direction).

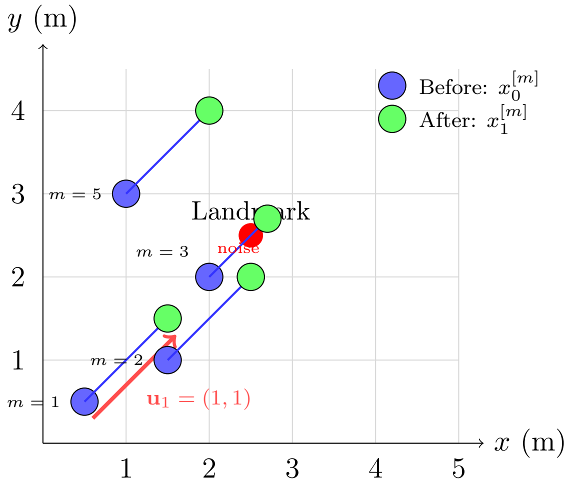
Each particle is affected by motion noise. For this exercise, each particle has been assigned a specific noise realization $\boldsymbol{\epsilon}^{[m]}$ (simulating sampling from the motion noise distribution).

Predicted particle positions using $x_1^{[m]} = x_0^{[m]} + u_{1,x} + \epsilon_x^{[m]}$ and $y_1^{[m]} = y_0^{[m]} + u_{1,y} + \epsilon_y^{[m]}$:

Particle	**Motion noise** $\boldsymbol{\epsilon}^{[m]}$	**Calculation**	$x_1^{[m]}$ **(m)**	$y_1^{[m]}$ **(m)**
$m = 1$	$(0.0, 0.0)$	$0.5 + 1.0 + 0.0$	1.5	1.5
$m = 2$	$(0.0, 0.0)$	$1.5 + 1.0 + 0.0$	2.5	2.0
$m = 3$	$(-0.3, -0.3)$	$2.0 + 1.0 - 0.3$	2.7	2.7
$m = 4$	$(+0.3, +0.2)$	$3.5 + 1.0 + 0.3$	4.8	2.7
$m = 5$	$(0.0, 0.0)$	$1.0 + 1.0 + 0.0$	2.0	4.0
$m = 6$	$(-0.2, -0.3)$	$3.0 + 1.0 - 0.2$	3.8	1.2

Important note: After the prediction step, the weights remain unchanged at $w_1^{[m]} = 1/6$ for all particles. Weights are updated only during the measurement update step based on how well each particle explains the sensor observation.

5.1 Visualization: Particle Motion in 2D Space



Each particle moves by control $\mathbf{u}_1 = (1, 1)$ *plus individual noise* $\boldsymbol{\epsilon}^{[m]}$. *Blue circles show initial positions, green circles show predicted positions after motion.*

6 Task 3: Calculate Distances and Errors (Given)

The robot's range sensor measures distance to the landmark: $z_1 = 0.5$ meters.

For each particle, we need to:

1. Calculate true distance from particle position to landmark $(2.5, 2.5)$
2. Calculate measurement error: $e^{[m]} = z_1 - r^{[m]}$
3. Calculate unnormalized weight (likelihood): $w_1^{[m]} \propto \exp\left(-\frac{(e^{[m]})^2}{2\sigma_z^2}\right)$ with $\sigma_z = 0.5$ m

6.1 Step 3.1: True Distances (provided)

Using $r^{[m]} = \sqrt{(x_1^{[m]} - 2.5)^2 + (y_1^{[m]} - 2.5)^2}$:

Particle	$x_1^{[m]}$	$y_1^{[m]}$	**Calculation**	$r^{[m]}$ **(m)**
$m = 1$	1.5	1.5	$\sqrt{(1.5 - 2.5)^2 + (1.5 - 2.5)^2} = \sqrt{2.0}$	1.414
$m = 2$	2.5	2.0	$\sqrt{(2.5 - 2.5)^2 + (2.0 - 2.5)^2} = \sqrt{0.25}$	0.500
$m = 3$	2.7	2.7	$\sqrt{(2.7 - 2.5)^2 + (2.7 - 2.5)^2} = \sqrt{0.08}$	0.283
$m = 4$	4.8	2.7	$\sqrt{(4.8 - 2.5)^2 + (2.7 - 2.5)^2} = \sqrt{5.33}$	2.309
$m = 5$	2.0	4.0	$\sqrt{(2.0 - 2.5)^2 + (4.0 - 2.5)^2} = \sqrt{2.5}$	1.581
$m = 6$	3.8	1.2	$\sqrt{(3.8 - 2.5)^2 + (1.2 - 2.5)^2} = \sqrt{3.38}$	1.838

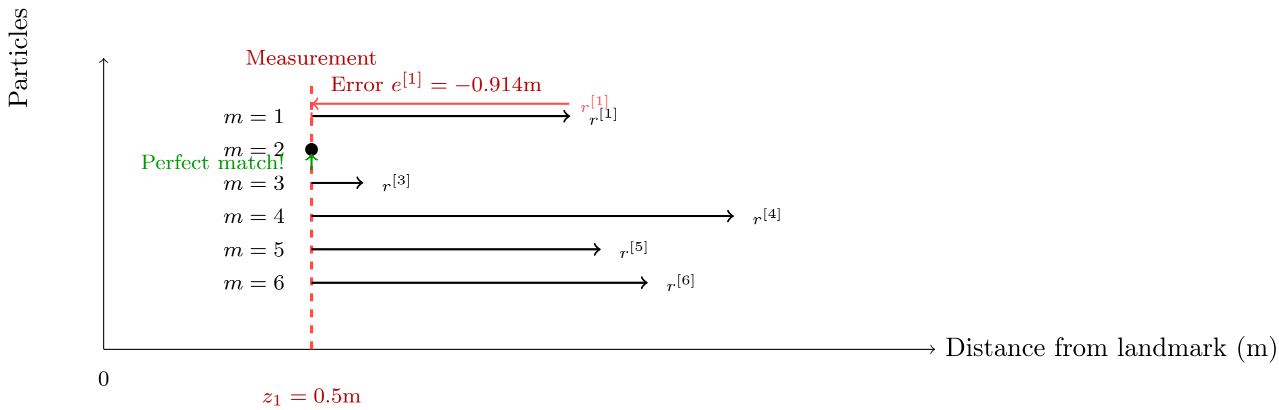
6.2 Step 3.2: Measurement Errors (provided)

Using $e^{[m]} = z_1 - r^{[m]}$ with measurement $z_1 = 0.5$ m:

Particle	$r^{[m]}$ **(m)**	**Error:** $e^{[m]} = 0.5 - r^{[m]}$ **(m)**	**Comment**
$m = 1$	1.414	-0.914	Large error
$m = 2$	0.500	**0.000**	**Perfect match!**
$m = 3$	0.283	$+0.217$	Small error
$m = 4$	2.309	-1.809	Very large error
$m = 5$	1.581	-1.081	Large error
$m = 6$	1.838	-1.338	Large error

Key observation: Particle 2 has **zero error** because its distance to the landmark (0.5 m) exactly matches the measurement (0.5 m). This particle will receive the highest weight!

6.3 Visualization: Measurement Errors and Particle Distances



Horizontal arrows show true distances $r^{[m]}$ from each particle to the landmark. Red dashed line shows measurement $z_1 = 0.5$ m. Particle 2 (green) aligns perfectly. Error shown for particle_1: measured distance (0.5m) minus actual distance (1.414m) = -0.914m.

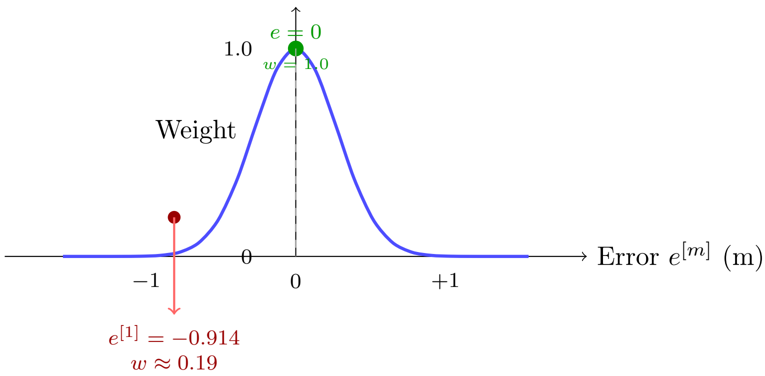
6.4 Step 3.3: Calculate Unnormalized Weights (TODO)

This calculation demonstrates the importance weighting step, the first key innovation of particle filters. In a programming implementation, this would be computed automatically. For this exercise, work through the calculations to understand how measurement likelihood becomes particle weight.

Using the Gaussian likelihood with $\sigma_z = 0.5$ m:

$$\tilde{w}_1^{[m]} \propto \exp\left(-\frac{(e^{[m]})^2}{2(0.5)^2}\right) = \exp\left(-\frac{(e^{[m]})^2}{0.5}\right)$$

Understanding the likelihood function: This Gaussian function gives higher weights to particles with smaller errors. The graph below shows how weight decreases as error increases:



The Gaussian likelihood function penalizes large errors exponentially. Smaller errors lead to higher weights. Example: particle 1 with error $e^{[1]} = -0.914m$ *receives weight* ≈ 0.19*, much smaller than the maximum weight of 1.0 at zero error.*

Calculate the unnormalized weights for all 6 particles:

Particle	$e^{[m]}$	$(e^{[m]})^2/0.5$	$\tilde{w}_1^{[m]} = \exp(-\cdot)$
$m = 1$	____________	____________	____________
$m = 2$	____________	____________	____________
$m = 3$	____________	____________	____________
$m = 4$	____________	____________	____________
$m = 5$	____________	____________	____________
$m = 6$	____________	____________	____________

Example for particle 1 (assuming $r^{[1]} \approx 1.414$ m):

$$e^{[1]} = 0.5 - 1.414 = -0.914 \text{ m}$$
$$\frac{(e^{[1]})^2}{0.5} = \frac{(-0.914)^2}{0.5} = \frac{0.8354}{0.5} = 1.6708$$
$$\tilde{w}_1^{[1]} = \exp(-1.6708) \approx 0.188$$

Question 3.1: Looking at the errors above, which particle will have the highest unnormalized weight and why?

Answer: ____

TODO: Tasks 4-7 Focus on the Core Particle Filter Innovation

The previous tasks showed you the setup: particles initialized uniformly, moved by the motion model, and their distances to landmarks measured. Now you will work through the ***key innovation of particle filters****: importance weighting based on measurement likelihood and resampling to focus computational resources on promising hypotheses.*

---

7 Task 4: Weight Normalization (TODO)

The unnormalized weights $\tilde{w}_1^{[m]}$ must be normalized to sum to 1:

$$w_1^{[m]} = \frac{\tilde{w}_1^{[m]}}{\sum_{i=1}^{6} \tilde{w}_1^{[i]}}$$

Step 4.1: Calculate the sum of unnormalized weights:

$$\sum_{m=1}^{6} \tilde{w}_1^{[m]} = \tilde{w}_1^{[1]} + \tilde{w}_1^{[2]} + \cdots + \tilde{w}_1^{[6]} = \underline{\hspace{4cm}}$$

Step 4.2: Calculate normalized weights:

Particle	$\tilde{w}_1^{[m]}$	**Calculation**	$w_1^{[m]}$
$m = 1$	________	$\tilde{w}_1^{[1]} / \sum$	________
$m = 2$	________	$\tilde{w}_1^{[2]} / \sum$	________
$m = 3$	________	$\tilde{w}_1^{[3]} / \sum$	________
$m = 4$	________	$\tilde{w}_1^{[4]} / \sum$	________
$m = 5$	________	$\tilde{w}_1^{[5]} / \sum$	________
$m = 6$	________	$\tilde{w}_1^{[6]} / \sum$	________
		Verification Sum:	________

Question 4.1: Verify that $\sum_{m=1}^{6} w_1^{[m]} = 1$. If not, check your calculations.

8 Task 5: Systematic Resampling (TODO)

Resampling focuses particles on high-probability regions. We use **systematic resampling**:

8.1 Intuition: "Survival of the Fittest"

Resampling implements a Darwinian principle that is the second key innovation of particle filters:

- **High-weight particles** (good hypotheses consistent with measurements) are likely to be **duplicated multiple times**
- **Low-weight particles** (poor hypotheses inconsistent with measurements) are likely to be **eliminated**
- This refocuses computational resources on **high-probability regions** of state space
- Without resampling, many particles would waste computation in low-probability regions

After resampling, the particle distribution approximates the **posterior belief** $\text{bel}(x_t)$ (incorporating the measurement) rather than just the prediction $\overline{\text{bel}}(x_t)$. The particles are distributed according to the posterior—those in high-probability regions appear more frequently.

8.2 Algorithm: Systematic Resampling

Input: Particle set $\{x_1^{[m]}, w_1^{[m]}\}_{m=1}^{6}$

Output: Resampled particle set $\{x_1'^{[m]}\}_{m=1}^{6}$ (with uniform weights)

Steps:

1. Construct cumulative distribution: $c[0] = 0$, $c[m] = c[m-1] + w_1^{[m]}$ for $m = 1$ to 6
2. Draw random start: $r \sim \text{Uniform}(0, 1/M) = \text{Uniform}(0, 1/6)$
3. For $m = 1$ to 6:
   - Calculate $U = r + (m-1)/6$
   - Find index i where $c[i-1] < U \leq c[i]$
   - Add particle $x_1^{[i]}$ to resampled set

Algorithmic implementation (for programming):

```
i = 1
for m = 1 to M:
    U = r + (m-1)/M
    while U > c[i]:
        i = i + 1
    Add particle x^[i] to resampled set
```

For hand calculation, simply find which cumulative range contains each U value as described above.

8.3 Step 5.1: Construct Cumulative Distribution

Index	$w_1^{[m]}$	Cumulative: $c[m] = c[m-1] + w_1^{[m]}$	Range
$c[0]$	—	0.0	—
$c[1]$	________	________	$(0.0, c[1]]$
$c[2]$	________	________	$(c[1], c[2]]$
$c[3]$	________	________	$(c[2], c[3]]$
$c[4]$	________	________	$(c[3], c[4]]$
$c[5]$	________	________	$(c[4], c[5]]$
$c[6]$	________	________	$(c[5], c[6]]$

Verification: $c[6]$ should equal _____.

8.4 Step 5.2: Draw Random Start

For this exercise, let's use: $r = 0.08$ (randomly drawn from $[0, 1/6]$)

Question 5.1: What is the range for r? $r \in [0,$ ____________________$]$

8.5 Step 5.3: Systematic Sampling

For each new particle $m' = 1$ to 6, calculate U and find which original particle to copy:

New Particle	$U = r + (m' - 1)/6$	Falls in range	Copy particle
$m' = 1$	$0.08 + 0/6 = 0.08$	________	$x_1^{[\quad]}$
$m' = 2$	$0.08 + 1/6 =$ ________________	________	$x_1^{[\quad]}$
$m' = 3$	$0.08 + 2/6 =$ ________________	________	$x_1^{[\quad]}$
$m' = 4$	$0.08 + 3/6 =$ ________________	________	$x_1^{[\quad]}$
$m' = 5$	$0.08 + 4/6 =$ ________________	________	$x_1^{[\quad]}$
$m' = 6$	$0.08 + 5/6 =$ ________________	________	$x_1^{[\quad]}$

Example for $m' = 1$:

- $U = 0.08$
- Looking at cumulative distribution, find range containing $U = 0.08$
- If $c[0] < 0.08 \leq c[1]$, then copy particle 1

8.6 Step 5.4: Resampled Particle Set

After resampling, list the new particle set with uniform weights:

New Index	Position $x_1'^{[m]}$	Copied from	New weight $w_1'^{[m]}$
$m' = 1$	________	particle _____	1/6
$m' = 2$	________	particle _____	1/6
$m' = 3$	________	particle _____	1/6
$m' = 4$	________	particle _____	1/6
$m' = 5$	________	particle _____	1/6
$m' = 6$	________	particle _____	1/6

Question 5.2: Why are all resampled particles assigned uniform weights $w_1'^{[m]} = 1/6$?

____

Question 5.3: Which original particle(s) likely appear multiple times in the resampled set?

____

9 Task 6: State Estimation (TODO)

After resampling, we can estimate the robot's position using the **weighted mean**:

$$\hat{x}_1 = \sum_{m=1}^{6} w_1'^{[m]} \cdot x_1'^{[m]} = \frac{1}{6} \sum_{m=1}^{6} x_1'^{[m]}$$

$$\hat{y}_1 = \sum_{m=1}^{6} w_1'^{[m]} \cdot y_1'^{[m]} = \frac{1}{6} \sum_{m=1}^{6} y_1'^{[m]}$$

Since all weights are uniform after resampling, this simplifies to the arithmetic mean.
Calculate the position estimate:

$$\begin{aligned}
\hat{x}_1 &= \frac{1}{6}(x_1'^{[1]} + x_1'^{[2]} + x_1'^{[3]} + x_1'^{[4]} + x_1'^{[5]} + x_1'^{[6]}) \\
&= \frac{1}{6}(\underline{\hspace{3cm}} + \underline{\hspace{3cm}} + \underline{\hspace{3cm}} + \underline{\hspace{3cm}} \\
&= \underline{\hspace{3cm}} \text{ meters}
\end{aligned}$$

$$\begin{aligned}
\hat{y}_1 &= \frac{1}{6}(\underline{\hspace{3cm}} + \underline{\hspace{3cm}} + \underline{\hspace{3cm}} + \underline{\hspace{3cm}} \\
&= \underline{\hspace{3cm}} \text{ meters}
\end{aligned}$$

Final position estimate: $\hat{\mathbf{x}}_1 = (\underline{\hspace{3cm}}, \underline{\hspace{3cm}})$ meters

10 Task 7: Analysis and Conceptual Understanding (TODO)

10.1 Question 7.1: Prediction vs. Update

How did the particle distribution change after:

- **Prediction step**: ______________________________
- **Measurement update step**: ______________________________
- **Resampling step**: ______________________________

10.2 Question 7.2: Particle Filter vs. Kalman Filter

Compare the particle filter with a Kalman filter:

Property	**Particle Filter**	**Kalman Filter**
Representation of belief	______________	Gaussian (mean, covariance)
Can handle multimodal distributions?	______________	______________
Can handle nonlinear models?	______________	______________
Computational complexity	______________	______________

10.3 Question 7.3: Importance of Resampling

What would happen if we never performed resampling?

Context: Without resampling, we would maintain weights multiplicatively over time:

$$w_t^{[m]} = p(z_t | x_t^{[m]}) \cdot w_{t-1}^{[m]}$$

After several time steps, weights would become highly skewed. Consider:

- What happens to particles in low-probability regions over time?
- How many particles would have significant weight after 10 time steps?
- Would this be computationally efficient?
- What is the concept of "particle degeneracy"?

Answer: ____

Hint: This phenomenon is called particle depletion or degeneracy—a few particles dominate all the weight while most particles contribute negligibly to the belief approximation.

10.4 Question 7.4: Effect of Number of Particles

How would the filter's performance change if we used:

- $M = 2$ **particles**: ____________________________________
- $M = 1000$ **particles**: ____________________________________

10.5 Question 7.5: Real-World Applications

Name two real-world robotics applications where particle filters are particularly advantageous:

1. ____________________________________
2. ____________________________________

11 Summary

In this exercise, you have worked through one complete iteration of the particle filter algorithm:

1. **Initialized** particles with uniform weights
2. **Predicted** particle positions using the motion model
3. **Calculated importance weights** using measurement likelihood
4. **Normalized weights** to form a proper probability distribution
5. **Resampled** particles using systematic resampling
6. **Estimated** robot position from the particle set

This non-parametric approach allows the particle filter to represent arbitrary distributions and handle nonlinear models, making it a powerful tool for mobile robot localization (Monte Carlo Localization) and SLAM (FastSLAM).