# Autonomous Robotics

## Monte Carlo Localization

Josiah Hanna
University of Wisconsin — Madison

Some figures taken from Probabilistic Robotics and Drew Bagnell's course notes.

# Programming Assignment #2

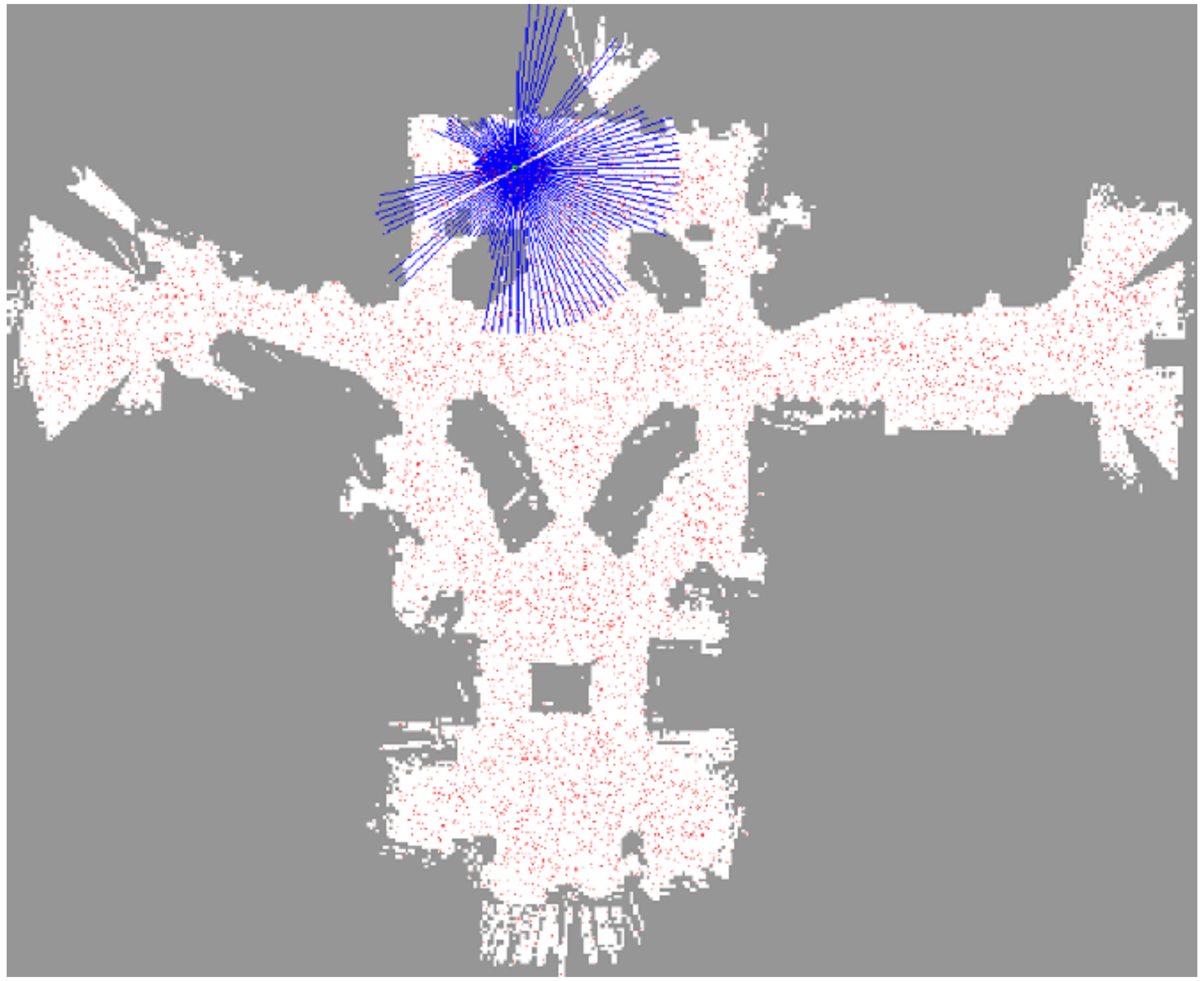
- Questions?
  - Go over action-less filters
- Comments?

# Learning Outcomes

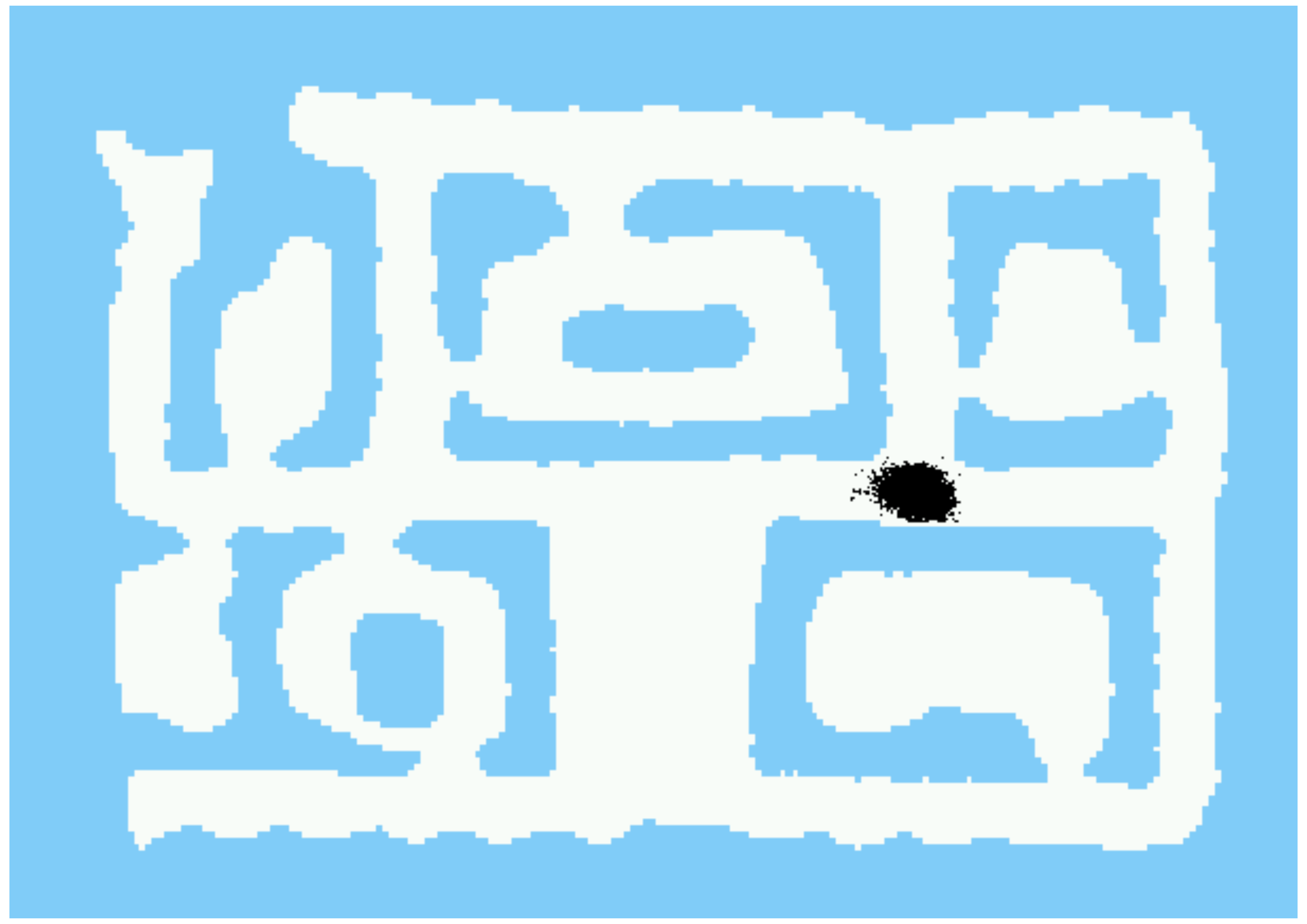
After today's lecture, you will:

- Review the particle filter
- Be able to describe and formalize the robot localization problem.

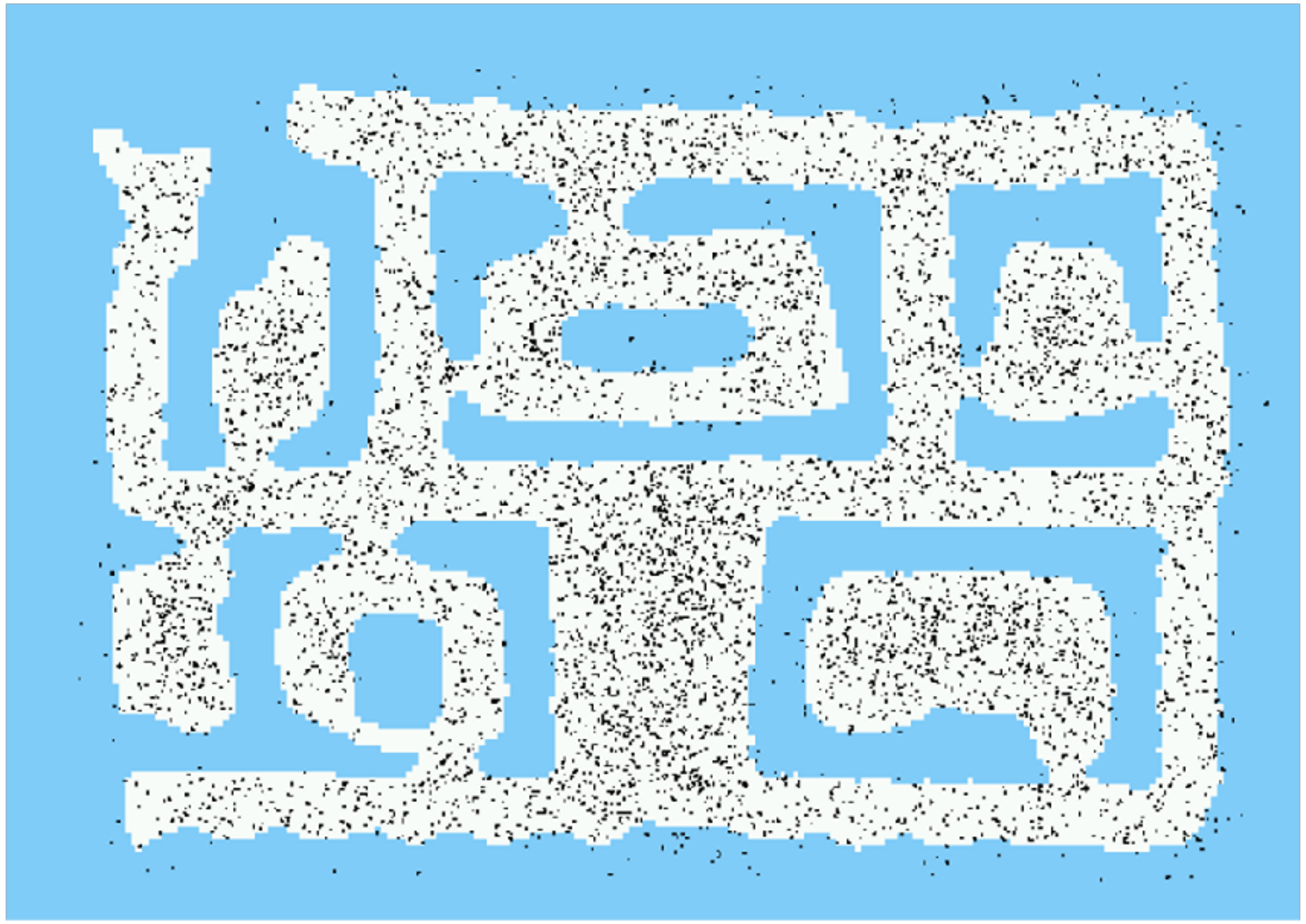
# Particle Filter Applications



**After Incorporating 65 Ultrasound Scans**

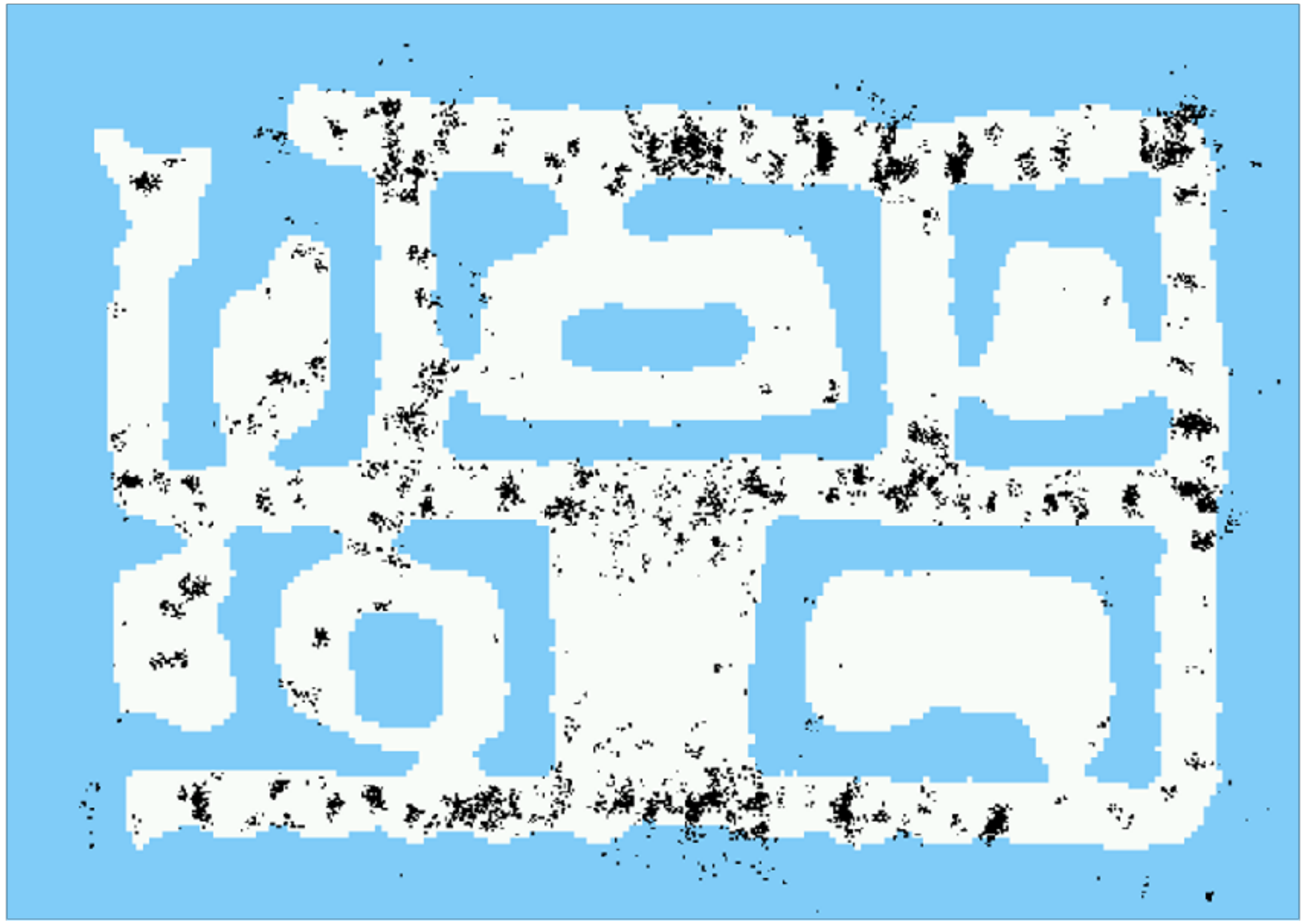


# Particle Filter Applications



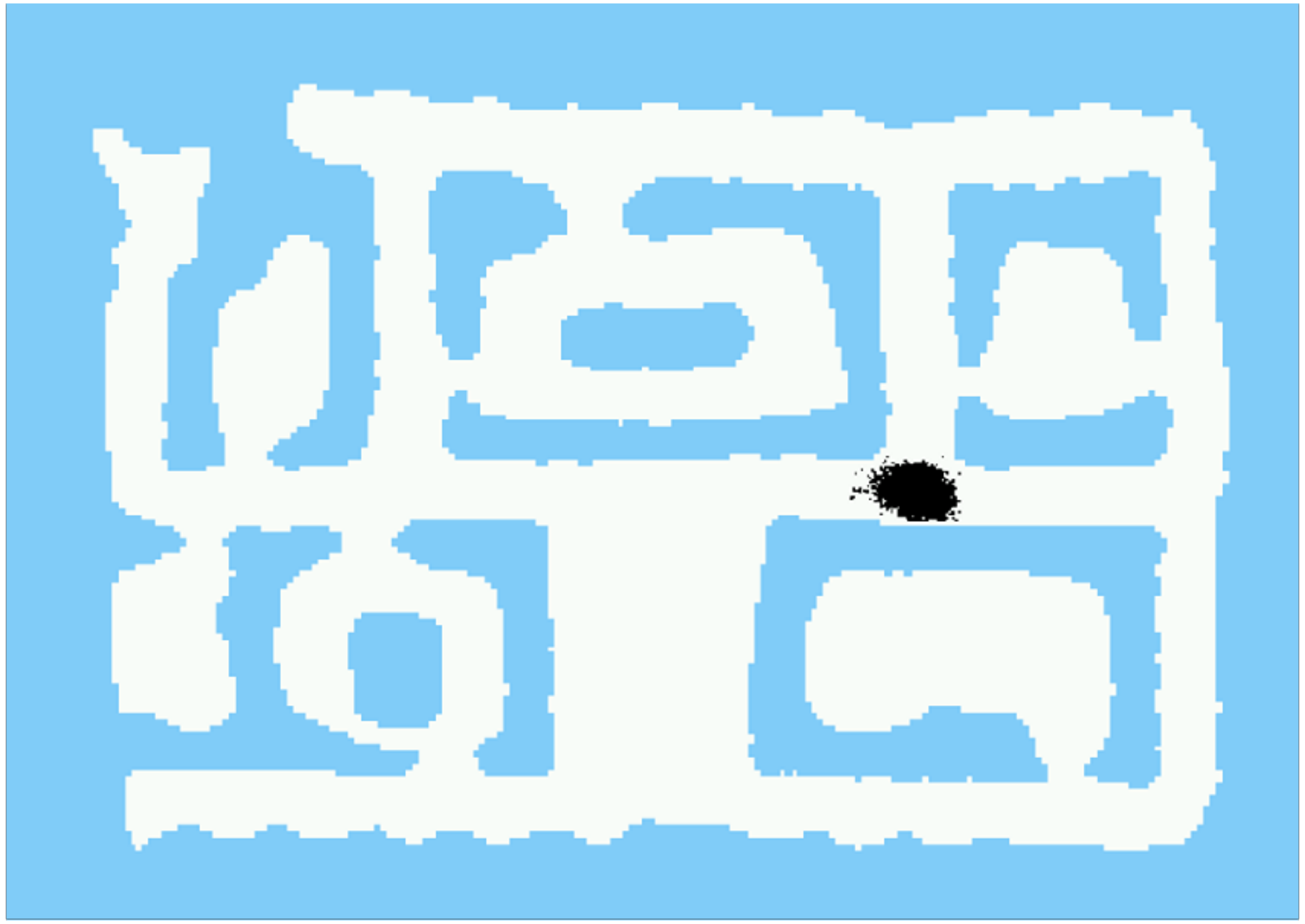
**Initial particles**

# Particle Filter Applications



**10 ultrasound scans**

# Particle Filter Applications



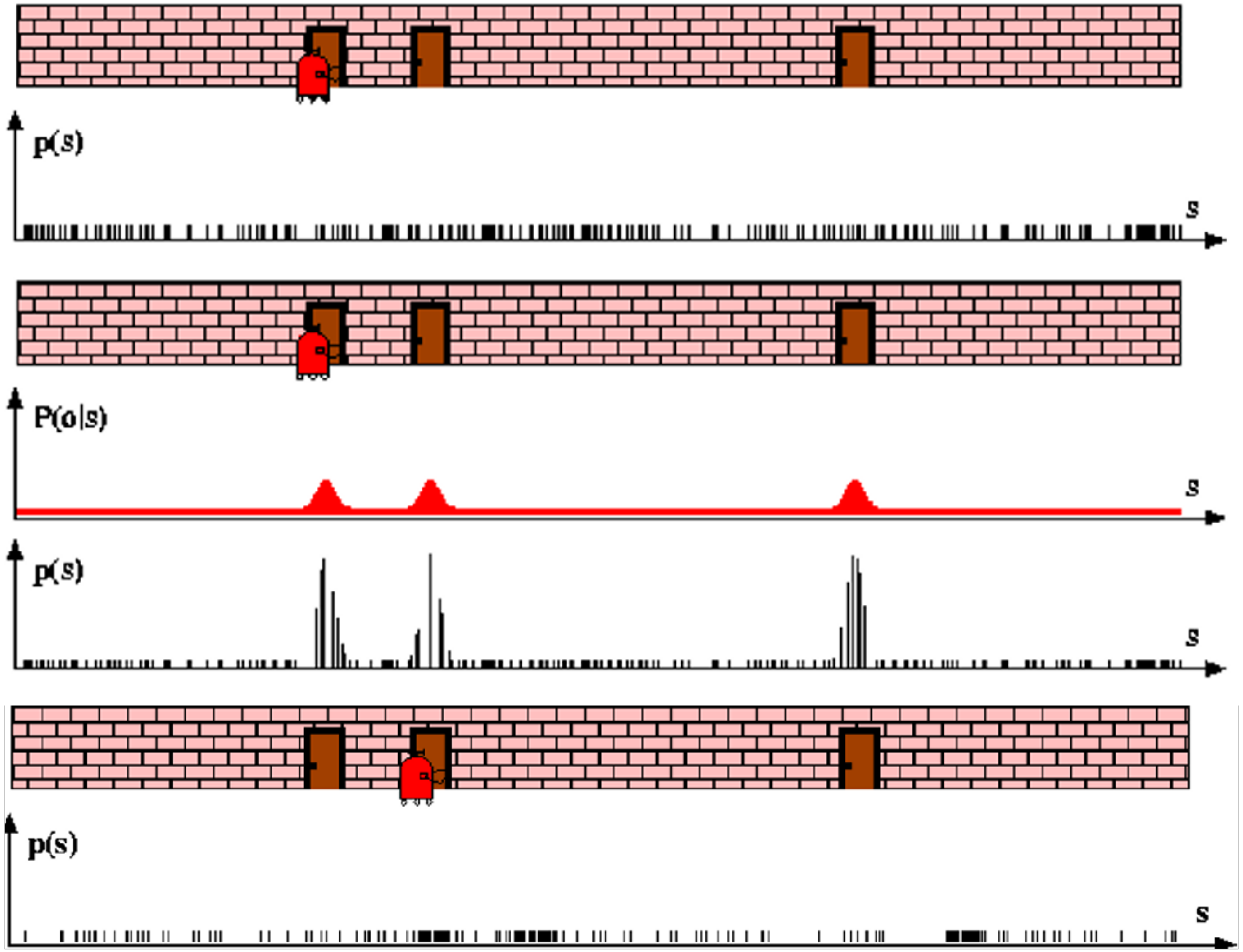
**65 ultrasound scans**

# Particle Filters

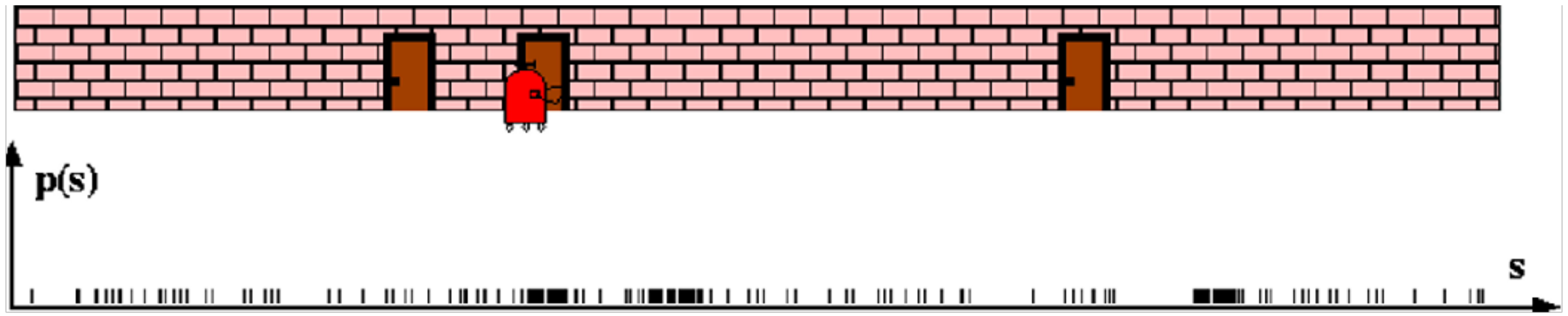
- Belief is represented by a set of particles, $\{(x_i^t, w_i)\}$.
- Robot takes action $u_t$ and then observes $z_t$. Set $w_i \leftarrow \frac{1}{N}$.
- Update particles:
  - $x_{t+1}^i \sim p(\cdot \mid x_t^i, u_t)$
  - $w_i \leftarrow w_i * p(z_t \mid x_{t+1}^i)$
  - Normalize weights so that $\sum_{i=1}^{N} w_i = 1$.
  - Sample N new particles (with replacement) to form a new particle set.

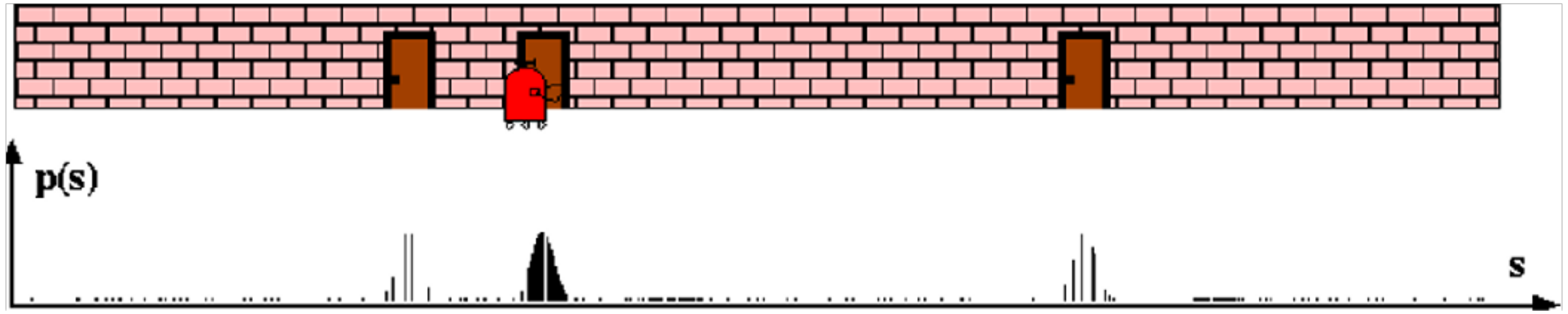
$$\texttt{bel}(x_t) = \sum_{i=1}^{N} w_i \cdot \mathbf{1}\{x_t^i = x_t\}$$

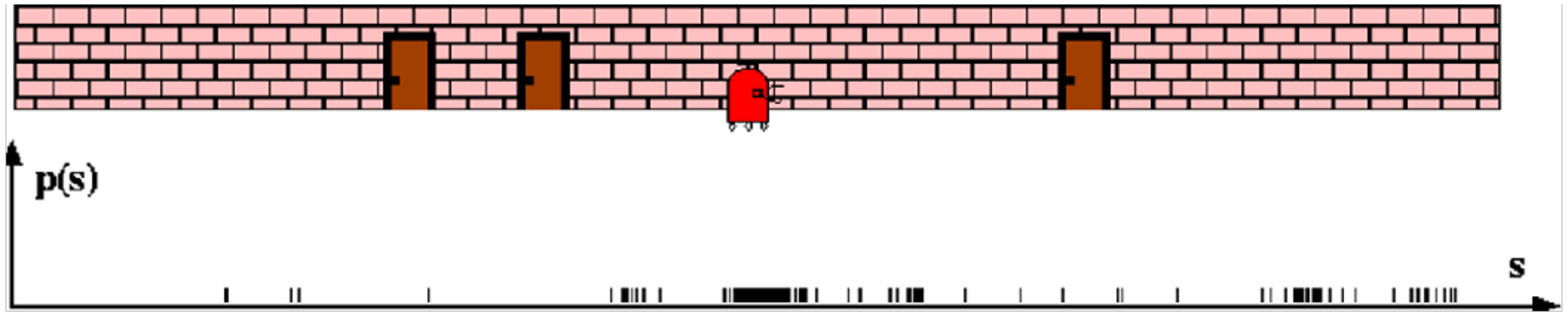
# Particle Filter Illustration



# Particle Filter Illustration



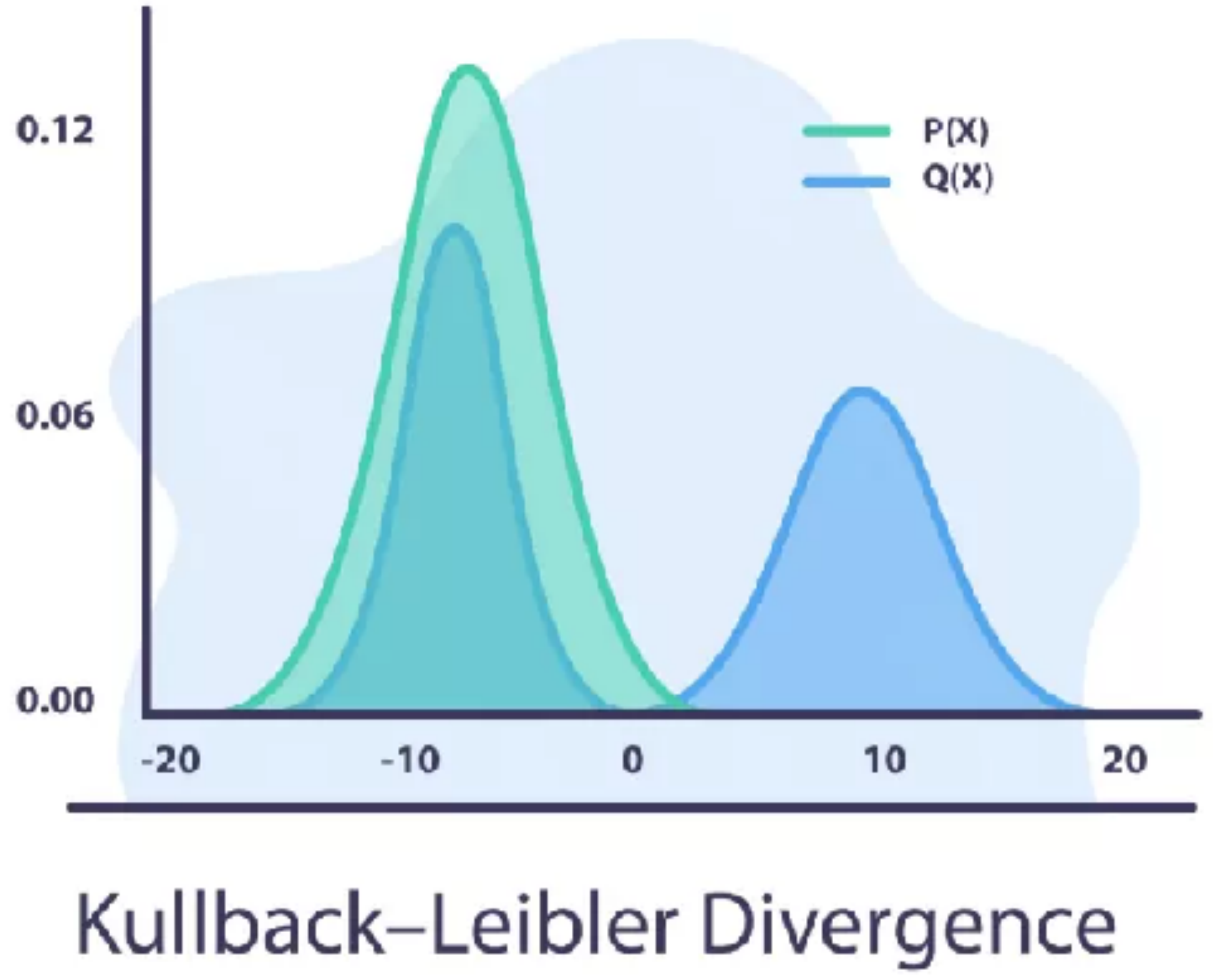



# Comparison to Kalman Filters

- Both filters can work in continuous state spaces.
- Particle filters > (Extended)KF:
  - Can approximate any belief distribution (compare to Kalman/EKF).
  - Approximate inference that scales with computation.
- (Extended)KF > Particle filters
  - Gaussian noise and dynamics are linear or can be linearized.
  - Computationally efficient.
  - Exact inference with fixed computation.

# KLD-Sampler

- Estimate the error between the particle approximation of belief and true posterior, $\texttt{bel}(x_t)$.
  - Use a histogram approximation of the particle-based belief.
- Kullback-Leibler divergence (KLD): a measure of similarity between two probability distributions.
- KLD-sampling determines the number of particles that keep the error (KL-divergence) below some threshold, $\epsilon$, with probability $1 - \delta$.



# Localization Problem

- Estimate a robot's pose as it moves in an environment.
- Small changes to our formal model of the autonomous robot problem:
  - $x_t$ will refer to the robot's pose in some global coordinate system.
  - $p(x_t \mid x_{t-1}, u_t)$ is the robot's motion model, formalizing changes to the pose after an action is taken.
  - State and transition model are no longer attempting to capture all relevant factors.
- Markov localization: Bayes filters and their extensions applied to the localization problem.
- Pose is usually **not** sensed directly.

# Localization Taxonomy

- Position-tracking: estimate the robot's current pose given observations, controls, and knowledge of the initial state.
  - $\texttt{bel}(x_t) = p(x_t \mid x_0, z_{1:t}, u_{1:t})$
- Global localization: estimate the robot's current pose given observations and controls.
  - $\texttt{bel}(x_t) = p(x_t \mid z_{1:t}, u_{1:t})$
- Kidnapped robot problem: the robot is teleported to some other location during operation and must recognize this and then relocalize.

# Position-Tracking

**Actual Trajectory**

**Trajectory from Odometry**

**Trajectory from position-tracking**

# Static vs Dynamic

Static vs. Dynamic

- Static: only the robot's pose changes in the environment.
- Dynamic: other factors change in the environment.

How to handle dynamic factors?

- Add new state variables to track

- Filter out the effect of those variables.

**https://www.epfl.ch/labs/vita/research/planning/crowd-robot-interaction/**

# Passive vs Active

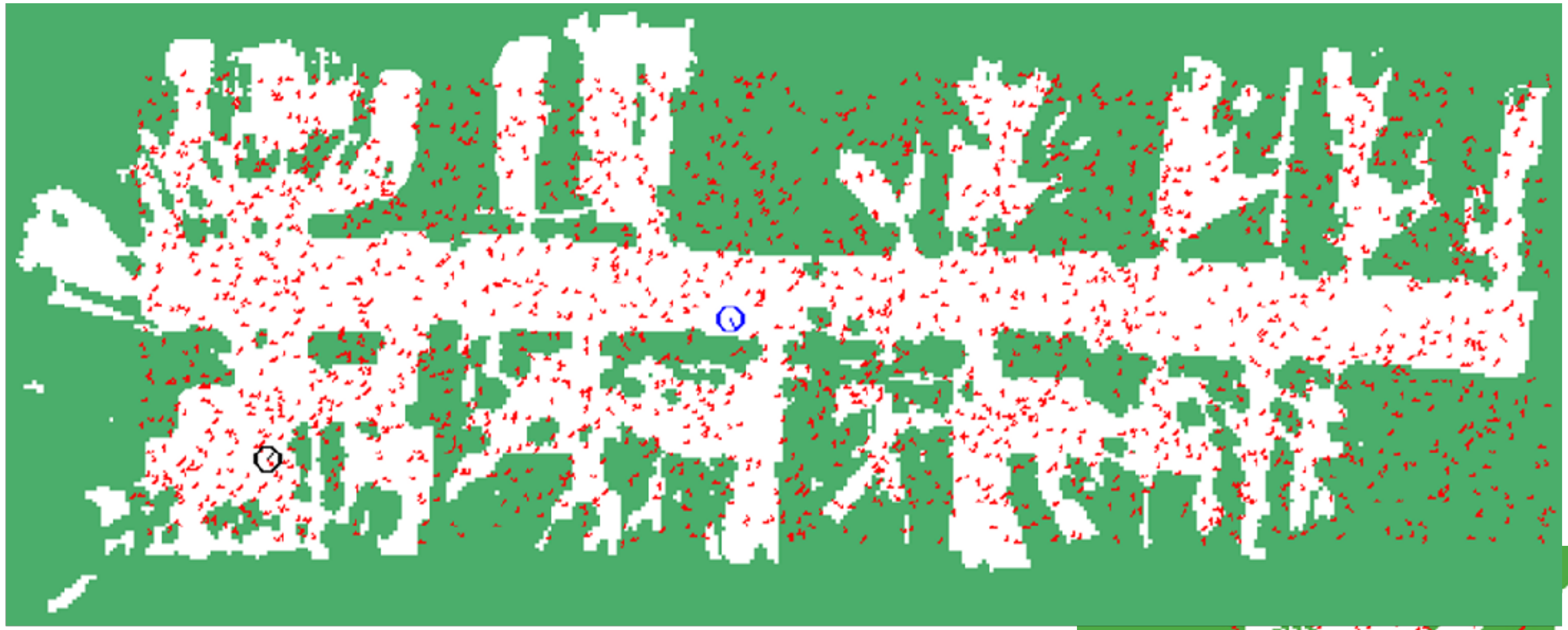
- Active: the robot takes actions that help localization. How?
  - Information-gathering actions
  - Remaining in easy to localize areas, e.g., wall-following.
- Passive: the robot's actions are guided by some other goal.

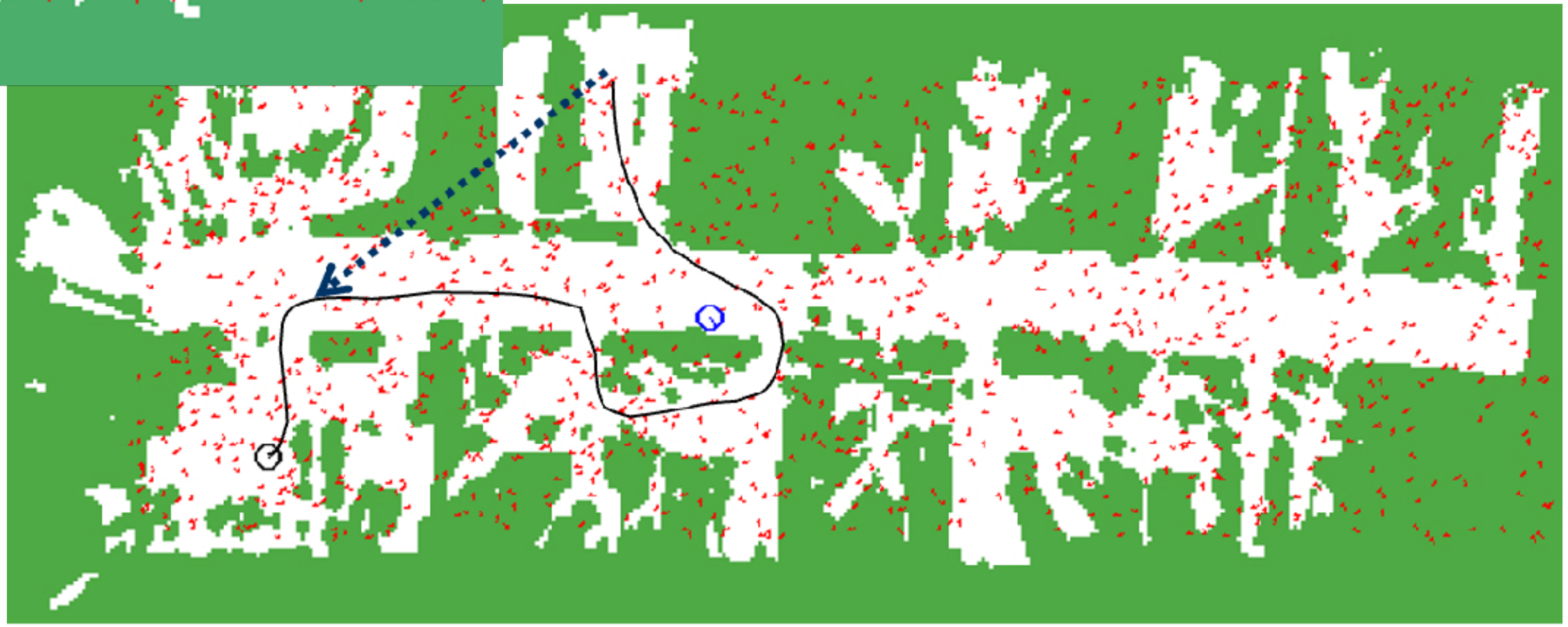
**https://www.epfl.ch/labs/vita/research/planning/crowd-robot-interaction/**

# Monte Carlo Localization

- Particle filter applied for Markov localization.
- How to initialize particles?
- How to handle failures?
    - Add random particles. How many? How to choose them?

# Monte Carlo Localization





# Summary

- Reviewed the particle filter and discussed extensions.
- Introduced the localization problem.
- Discussed considerations of the particle filter for localization problems.

# Action Items

- Work on programming assignment #2.
- Read on SLAM for next week; send a reading response by 12 pm on Monday.