# Autonomous Robotics

## Simultaneous Localization and Mapping

Josiah Hanna
University of Wisconsin — Madison

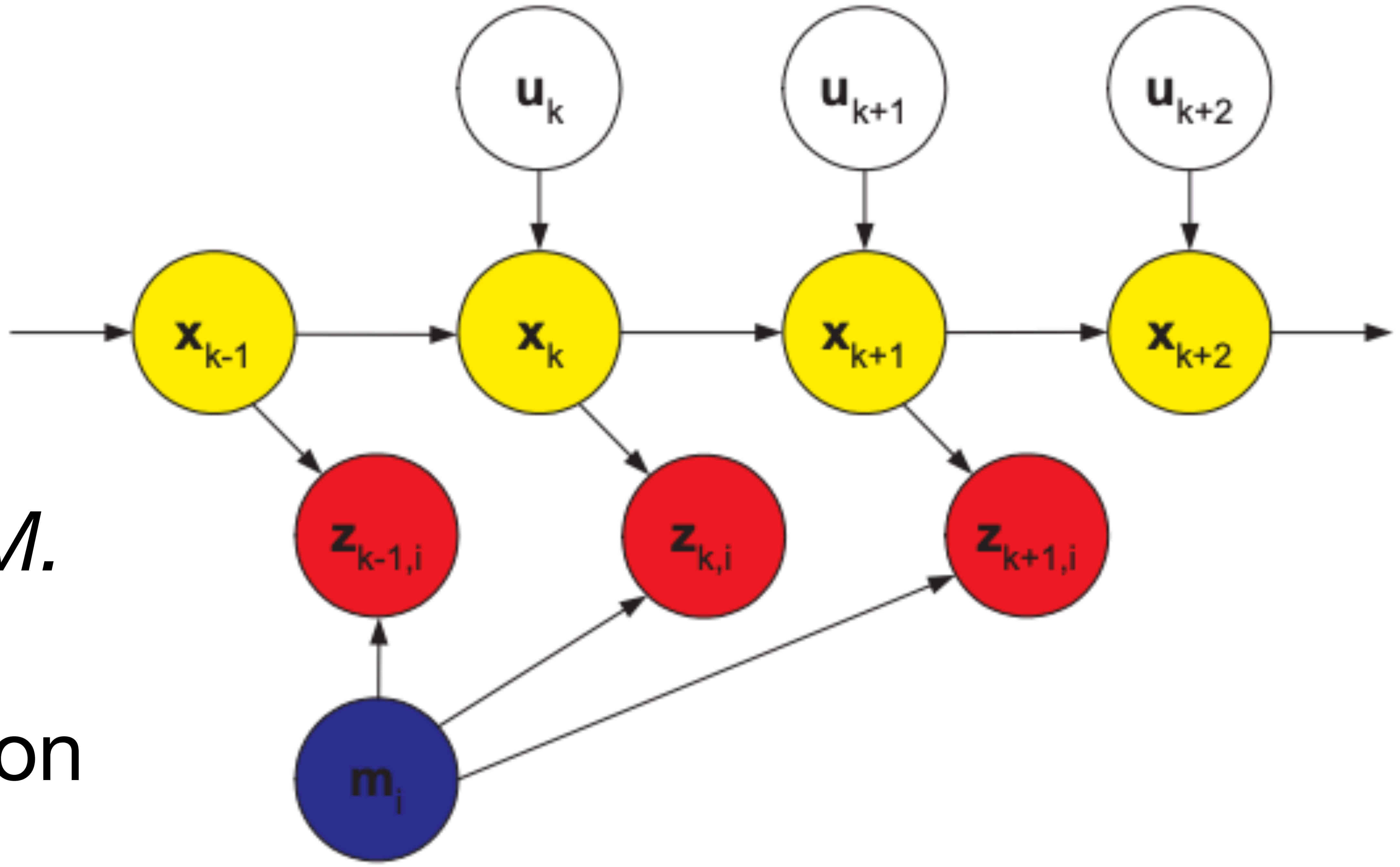
# Learning Outcomes

After today's lecture, you will:

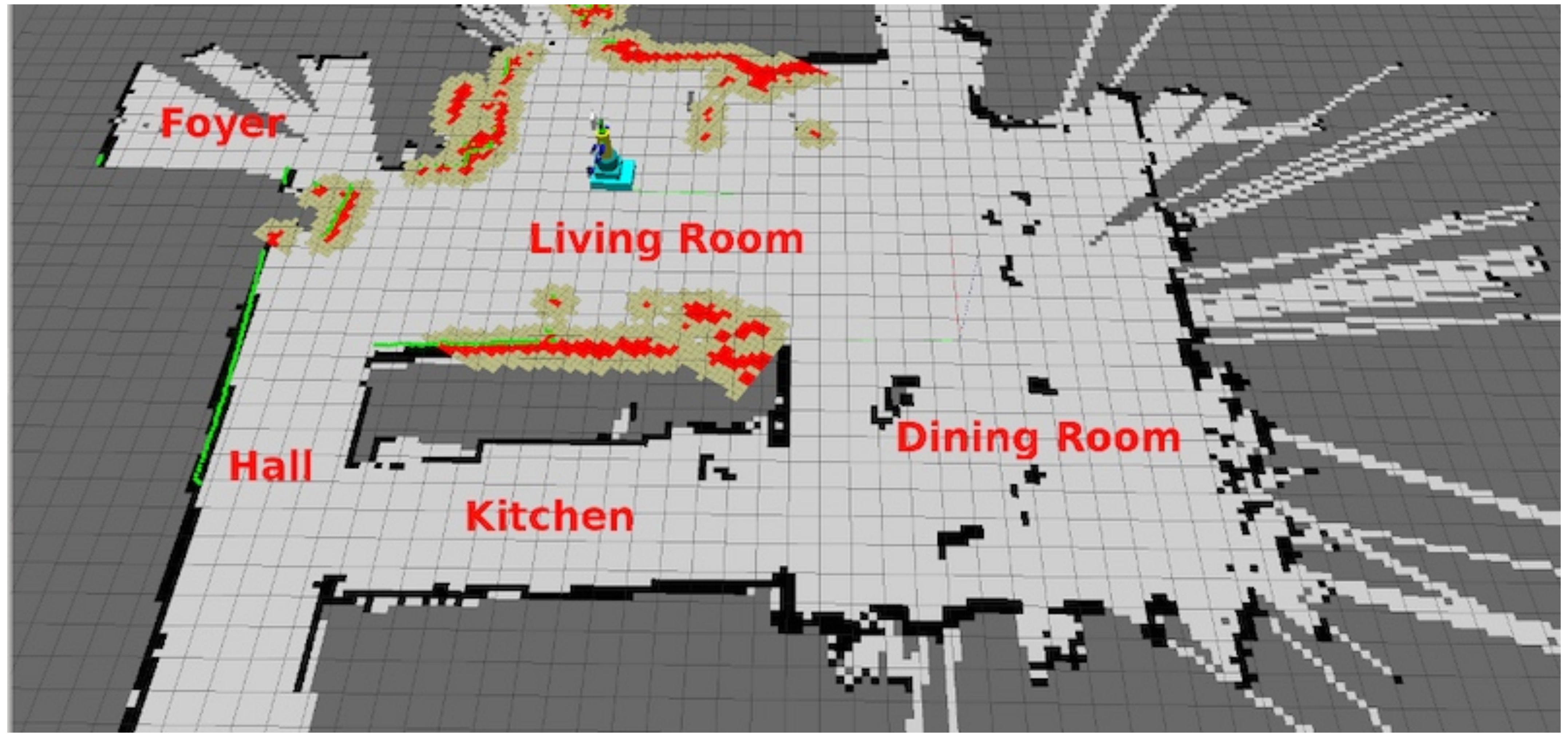
- Understand limitations of applying vanilla particle filters to SLAM
- Understand how the Rao-Blackwellized particle filter overcomes these limitations.

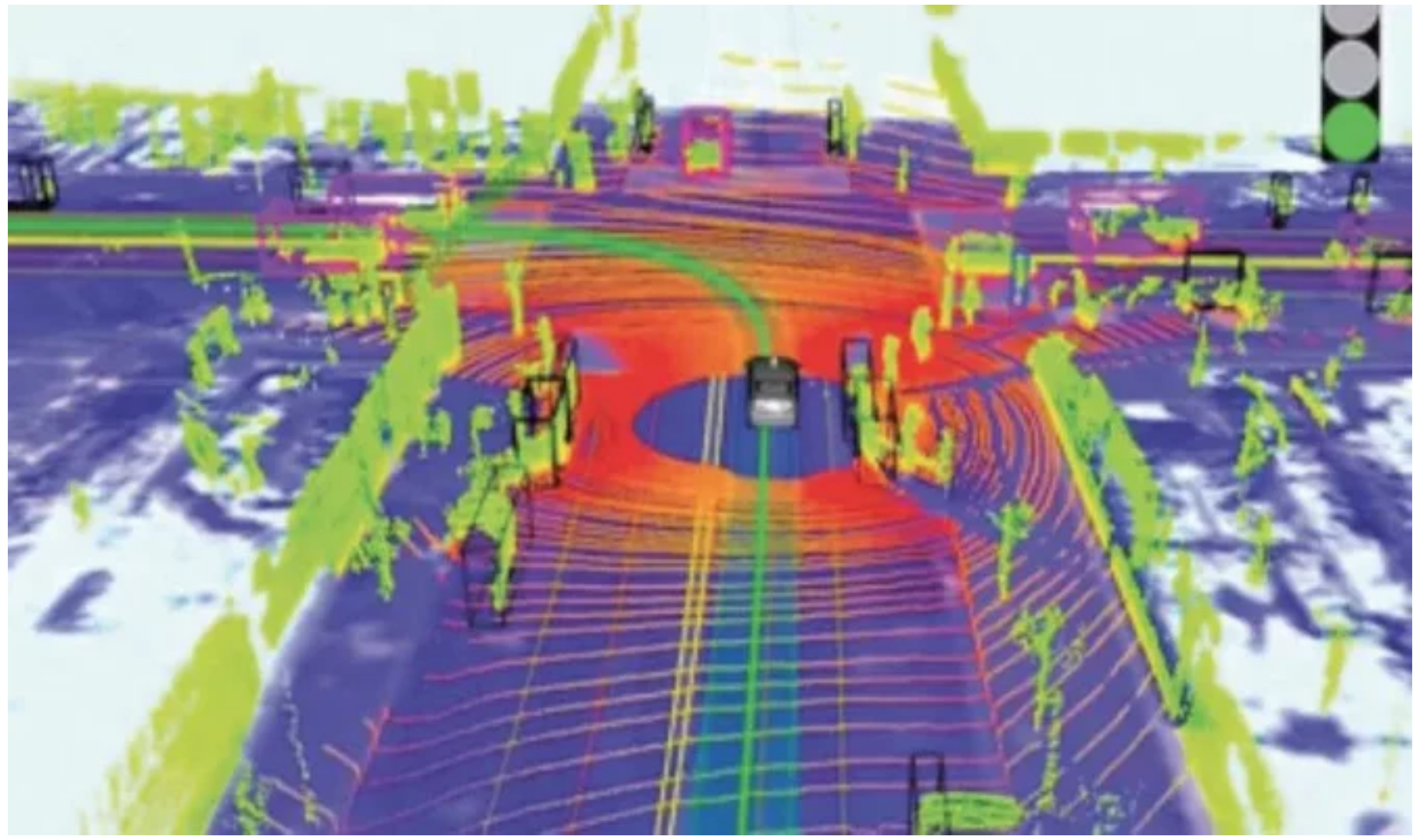
# SLAM

- Localize and map at the same time.
- Formally, estimate $p(x_t, m \mid z_{1:t}, u_{1:t}, x_0)$
  - Or $p(x_{1:t}, m \mid z_{1:t}, u_{1:t}, x_0)$, i.e., *full SLAM.*
- Assume we have a motion and observation model:
  - $p(x_t \mid x_{t-1}, u_t)$ and $g(z_t \mid x_t, m)$.



# Applications



# EKF SLAM with Landmarks

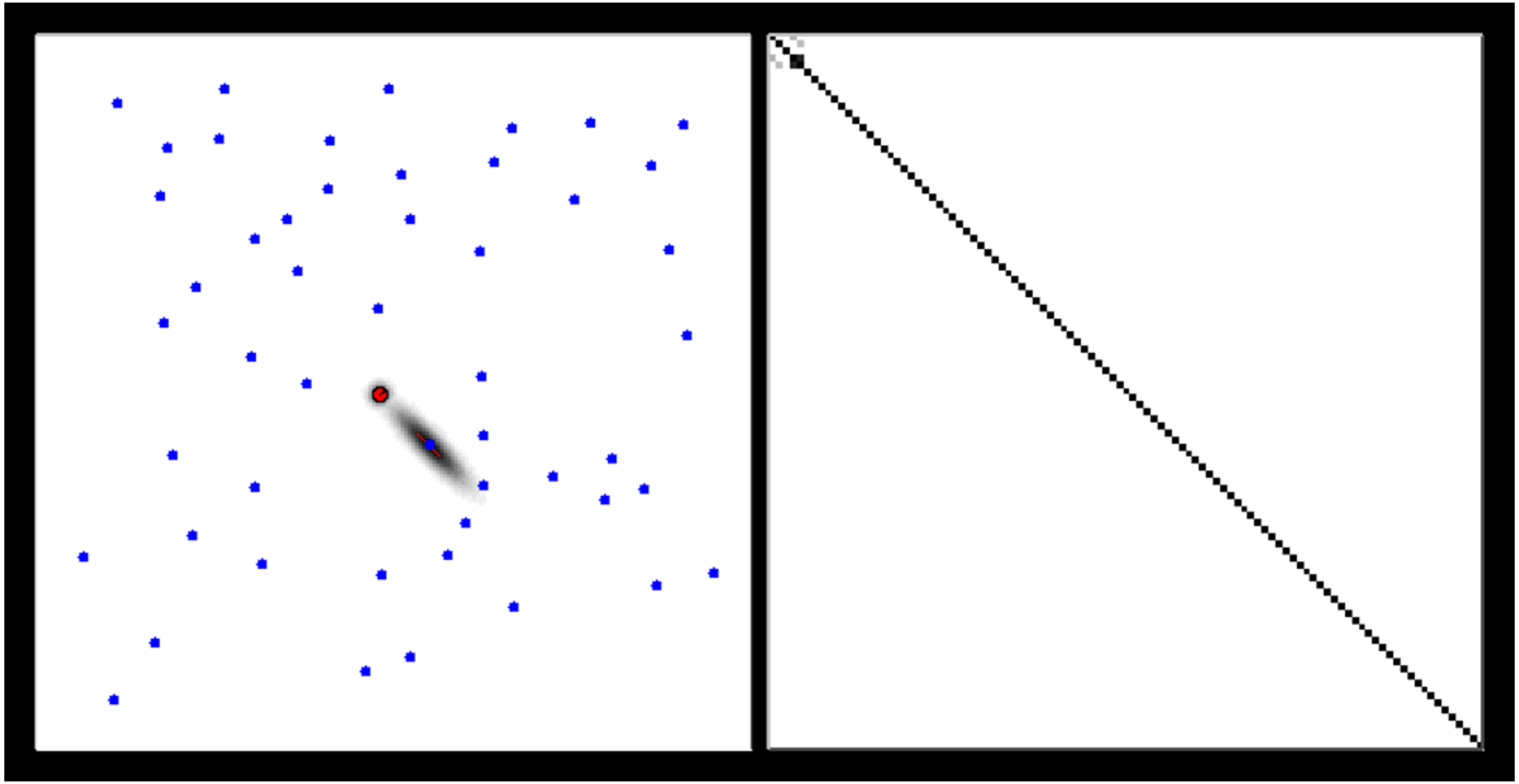
- Key idea: make landmarks part of the state and then run an extended Kalman filter.
- Map representation: a set of landmarks with unknown locations.
    - Let $m_x^i, m_y^i$ be the coordinates of the ith landmark and $m = (m_x^1, m_y^1, \ldots, m_x^k, m_y^k)$ be the vector of all landmark coordinates.
- Define $z_t^i$ as the observation of the ith landmark at time t.
- Assume $p(z_t^i \mid x_t, m_x^i, m_y^i) = \mathcal{N}(h(x_t, m_x^i, m_y^i), R)$.
- Initialize belief $\texttt{bel}(x_0, m) = \mathcal{N}([x_0, m]; \mu_0, \Sigma_0)$
- In practice, incrementally add landmarks as found.
- Must know which landmark an observation is associated with.

$$\mu_0 = \begin{bmatrix} x \\ y \\ \theta \\ m_x^1 \\ m_y^1 \\ \ldots \\ m_x^k \\ m_y^k \end{bmatrix}$$

# EKF SLAM with Landmarks

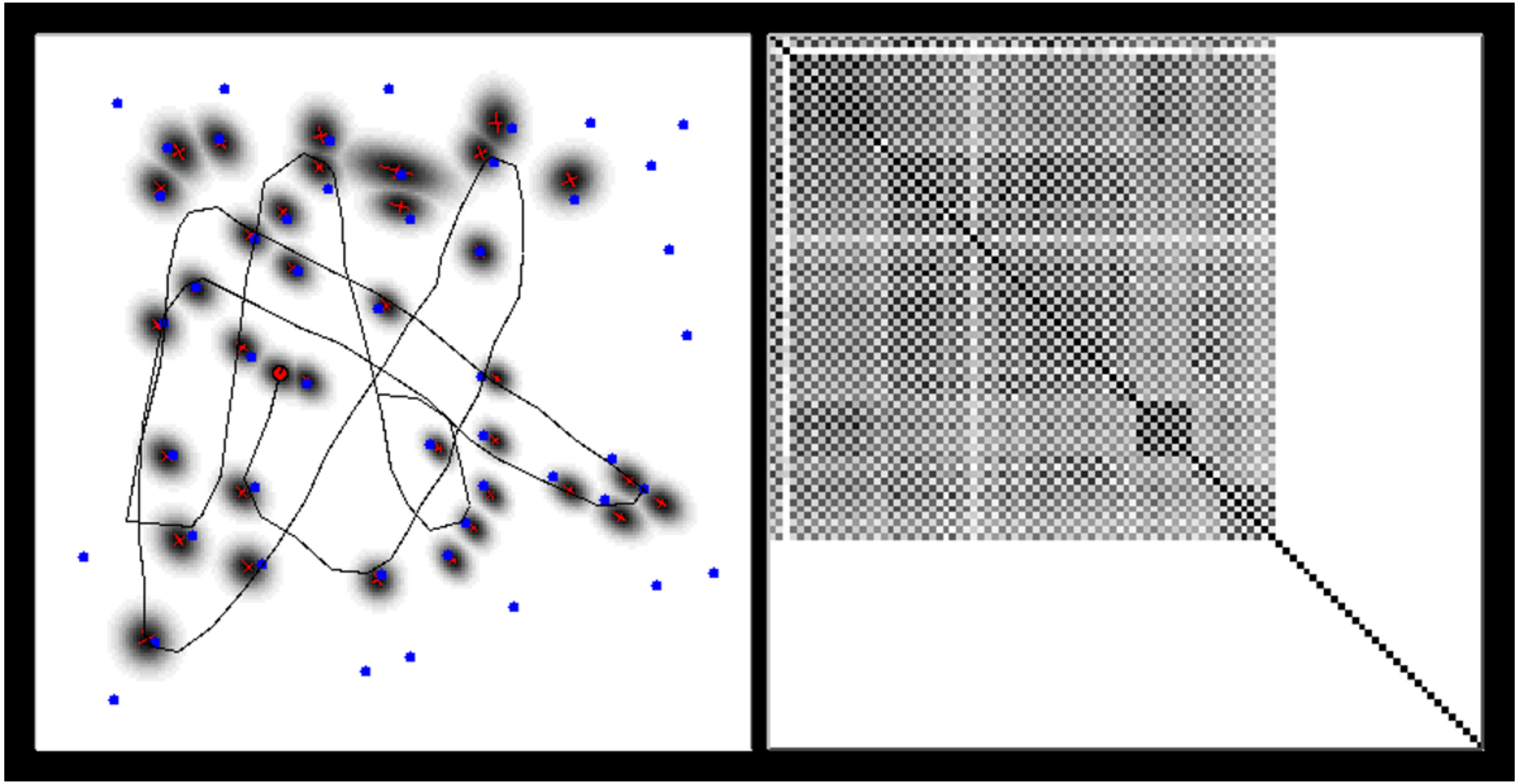
- Covariance matrix $\Sigma_t$ captures correlation between landmarks.
  - Improves estimate landmark estimates in $\mu_t$ even for landmarks that weren't observed at time $t$.
- Prediction step: only changes $\mu_t$ for position components; increases uncertainty for all components.
- Update step: run for each landmark observation $z_t^i$:
  - $\bar{\mu}_t, \overline{\Sigma}_t \leftarrow$ update step with $z_t^i$.

# EKF-SLAM



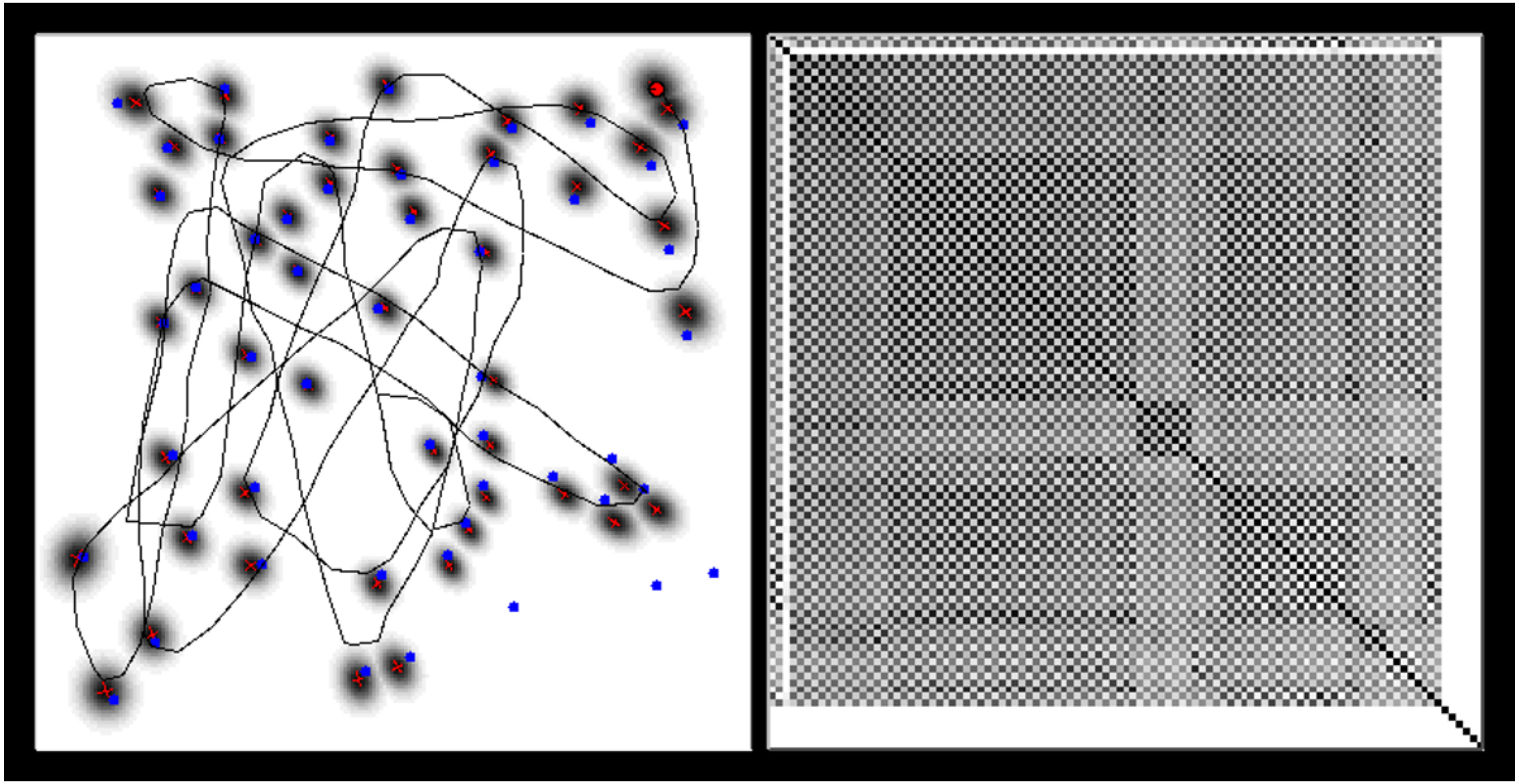
**Map** **Covariance Matrix**

# EKF-SLAM



**Map** **Covariance Matrix**

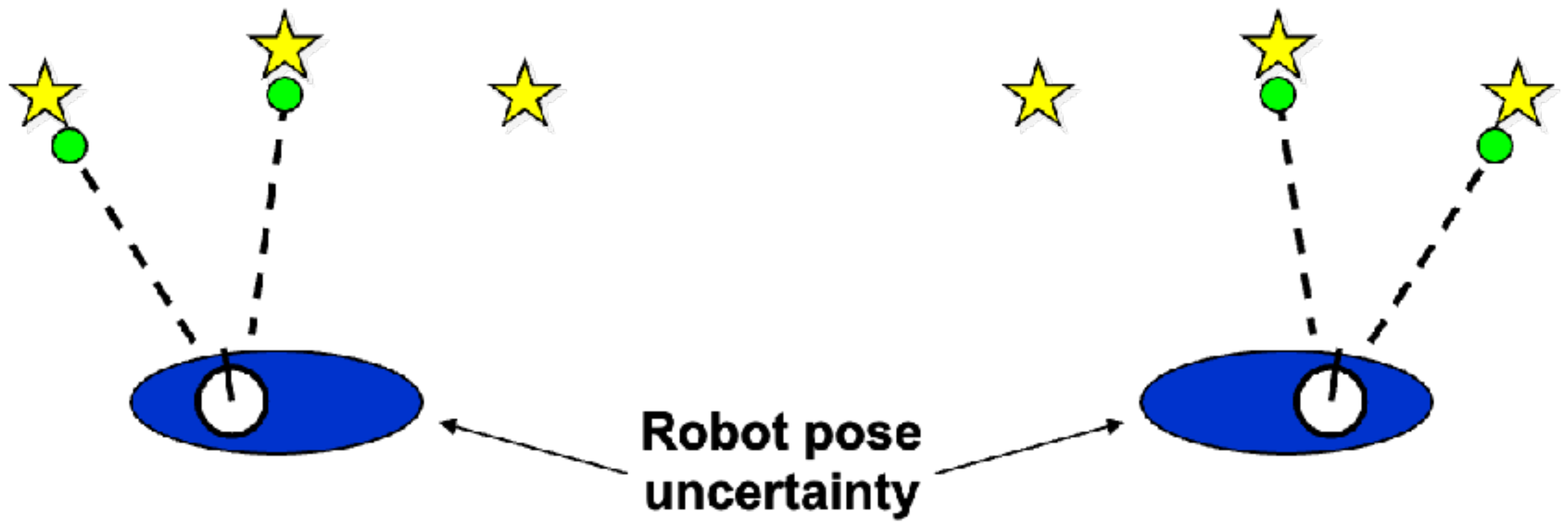
# EKF-SLAM



**Map** **Covariance Matrix**

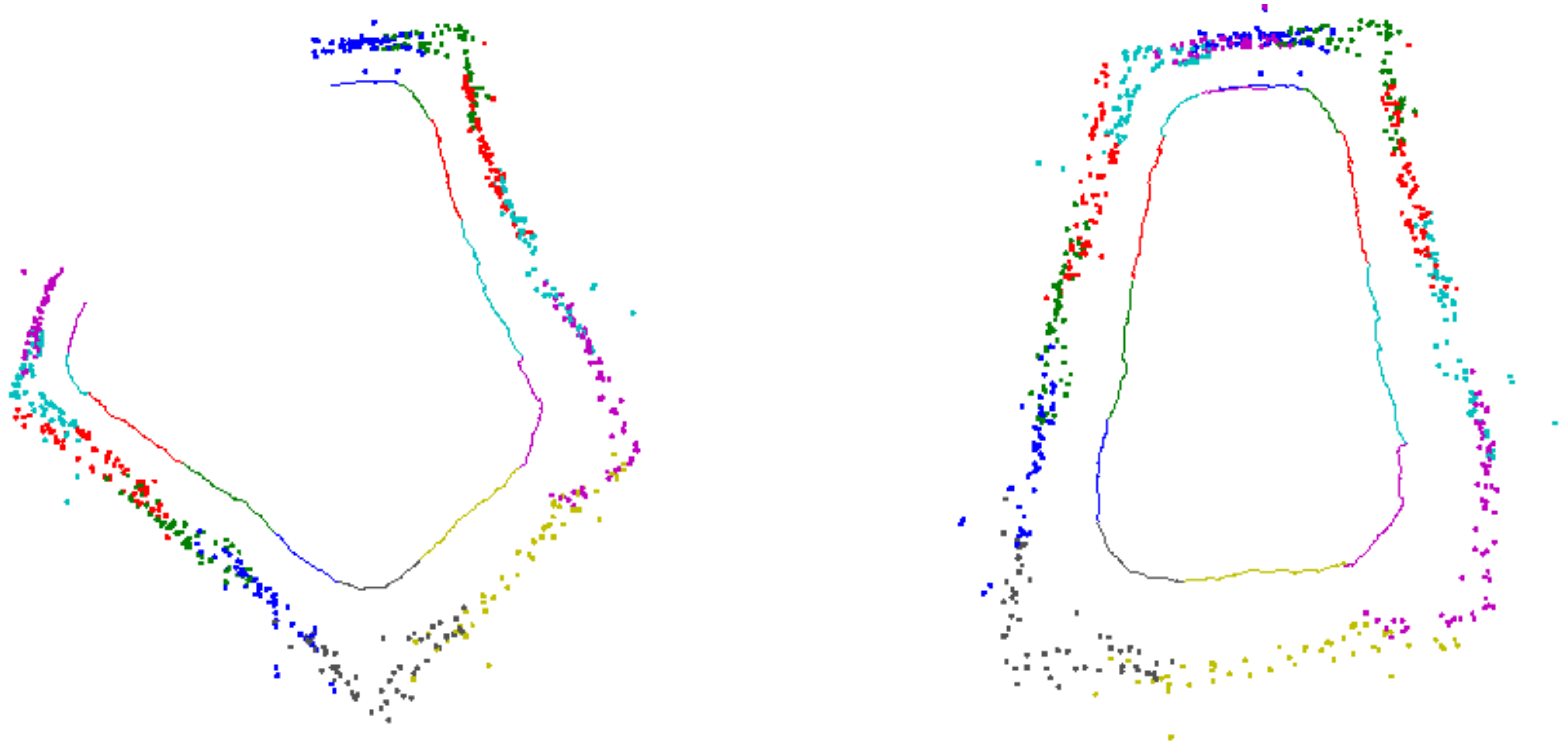
# Data Association

- How to determine which landmark $z_t$ corresponds to?
  - Consider observations based on (noisy) polar coordinates relative to robot. Could be unclear which landmark an observation represents.
- Challenging cases:
  - What if the robot has discovered a new landmark?
  - What if two landmarks are close together?
- Solution:
  - Estimate maximum likelihood correspondence (brittle).
  - Choose spatially far apart and distinctive landmarks for the map.



# Loop Closure

- Detect when a previously visited location is being revisited.



# Limitations of EKF-SLAM

- If uncertainty is high then linearization may be poor.
- Brittle under ambiguity.
- A large number of landmarks requires inverting a large covariance matrix.
    - Polynomial space and time requirement but still bad in practice.

So let's turn to particle filters!

# Vanilla Particle Filters for Mapping

- Recall particle filtering from two weeks ago:
  - Represent belief with a set of weighted particles: $\{(x_i, w_i)\}_{i=1}^{m}$.
  - After new observations are received, resample particles in the set.
- For SLAM:
  - We now represent each particle as: $\{(x_i, m_i, w_i)\}_{i=1}^{m}$ where $m_i$ is a possible map.
  - Problem: maps are high-dimensional, may require an impractical number of particles for proper convergence.

Curse of dimensionality intuition

# Rao-Blackwellized Particle Filters

- Alternative idea: each particle also represents uncertainty on the map.
  - $\{x^i_{0:t}, p(m_i \mid x^i_{0:t}), w_i\}_{i=1}^m$
  - Why does this representation allow us to use fewer particles?
  - Note: particle represents full trajectory. Why useful?
- Use Gaussian belief on map landmarks:
  - $p(m_i = (m_x^1, m_y^1, \ldots m_x^k, m_y^k) \mid x_{0:t}) = \prod_{j=1}^{k} \mathcal{N}([m_x^j, m_y^j]]; \mu_j, \Sigma_j)$
  - Gaussian belief is updated with EKF assuming a known robot trajectory of $x_{0:t}$.
  - Why useful?

# FastSLAM

- Both FastSLAM 1.0 and 2.0 are Rao-Blackwellized Particle Filters.
  - Differ in the proposal distribution for resampling step:

$$p(z_i \mid x_{0:t}) = \int_m p(m \mid x_{0:t}, z_{1:t}) p(z_i \mid m, x_{1:t})$$

$$w_i \propto \frac{p(z_i \mid x_{0:t}) p(x_t \mid x_{t-1}, u_t)}{\pi(x_t \mid x_{0:t-1}, z_t)}$$

$$\pi(x_t \mid x_{0:t-1}, z_t, u_t) = p(x_t \mid x_{t-1}, u_t)$$

FastSlam 1.0
Sampling from the motion model

$$\pi(x_t \mid x_{0:t-1}, z_t, u_t) = p(x_t \mid x_{0:t-1}, u_t, z_t)$$

FastSlam 2.0
Use observation to get better samples

# Rao-Blackwellization

- Why this works.
- Replace sampling of one variable with an analytic expectation.
- Imagine we want to estimate $\theta = E[f(X, Y)] = \sum_x \sum_y p(x, y) f(x, y)$.
    - We do not know $p(x)$ but 1) we can sample from it and 2) for any x, we know $p(y \mid x)$. We can also sample from $p(y \mid x)$.
- Compare estimators:

$$\theta_0 \approx \frac{1}{n} \sum_{i=1}^{n} f(x_i, y_i) \qquad \theta_1 \approx \frac{1}{n} \sum_{i=1}^{n} \sum_y p(y \mid x_i) f(x_i, y)$$

- $\theta_0$ will have higher variance than $\theta_1$ because it uses random sampling for both x and y.

# GMapping

- Both FastSLAM and EKF-SLAM use a feature-based map.
- The GMapping algorithm is a Rao-Blackwellized particle filter that uses a grid map representation.
    - Each particle represents $p(m \mid x_{0:t}, z_{1:t})$ with the most likely map (the maximum a posteori (MAP) estimate — no pun intended) when necessary to integrate over the map for computing weights.
- Also, uses an improved proposal distribution (not discussed here)
- Finally, only performs resampling when effective sampling size drops too low.
- GMapping is a widely used approach with good open source implementations.

# Summary

- Discussed limitations of using particle filters for SLAM.
- Introduced the Rao-Blackwellized particle filter.
- Discussed differences between FastSLAM 1.0 and 2.0

# Action Items

- Kinematics reading for next week; send a reading response by 12 pm on Monday.
- SLAM assignment released soon.