Autonomous Robotics

Simultaneous Localization and Mapping

Josiah Hanna
University of Wisconsin — Madison

Learning Outcomes

After today's lecture, you will:

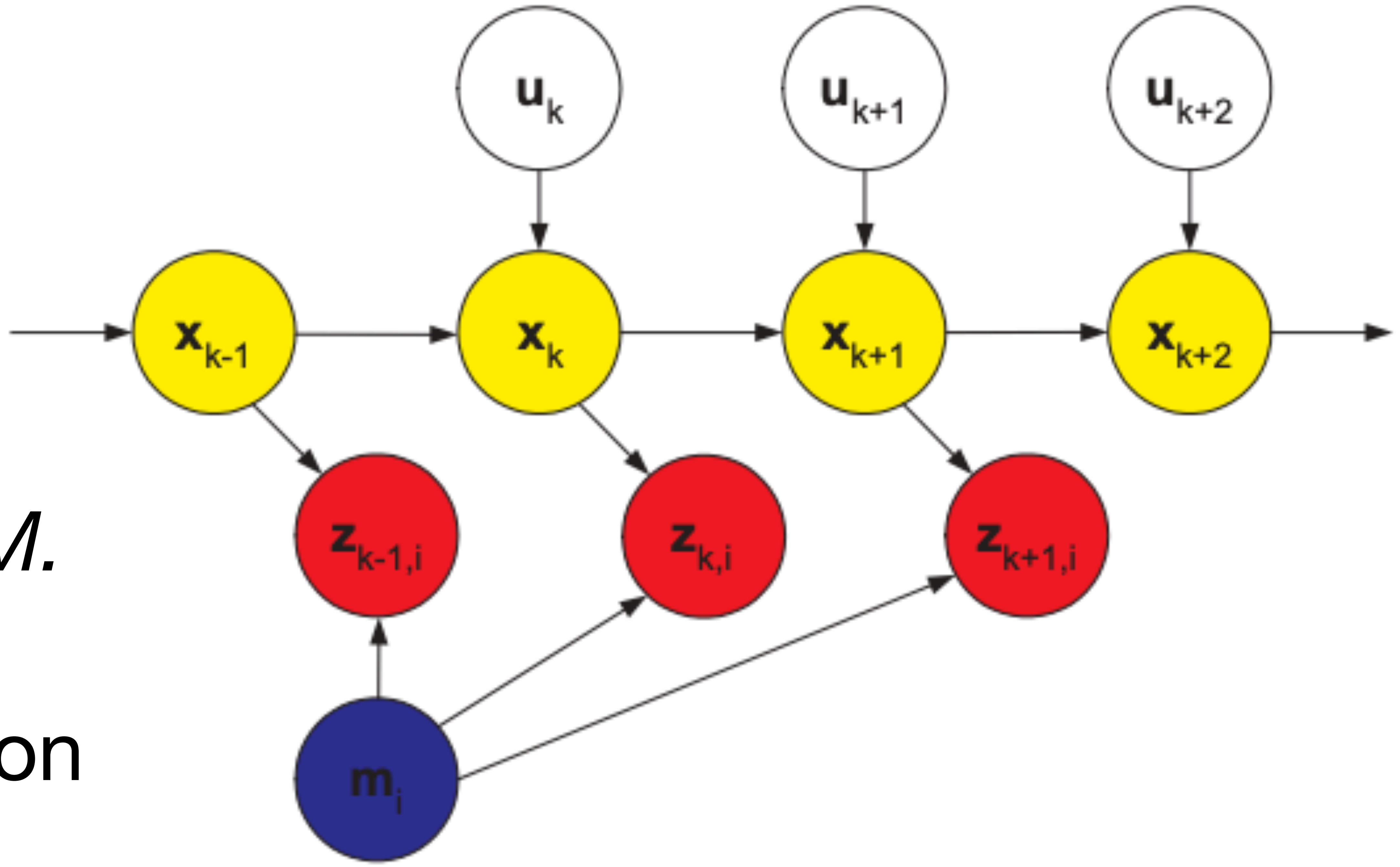
- Be able to explain the GraphSLAM methodology.
- Be able to describe open questions in SLAM research and applications.

Odometry Data

- Odometry is a measurement of movement.
    - Example: $(\Delta x, \Delta y, \Delta \theta)$.
- Technically, a sensor observation but often treated as the robot's control. Why is this reasonable?
- If we know how far the robot has moved, then why must a robot localize?
- *Visual odometry*: determine change in position based on change of visual images.

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 $g(z_t \mid x_t, m)$.



Systems view of SLAM

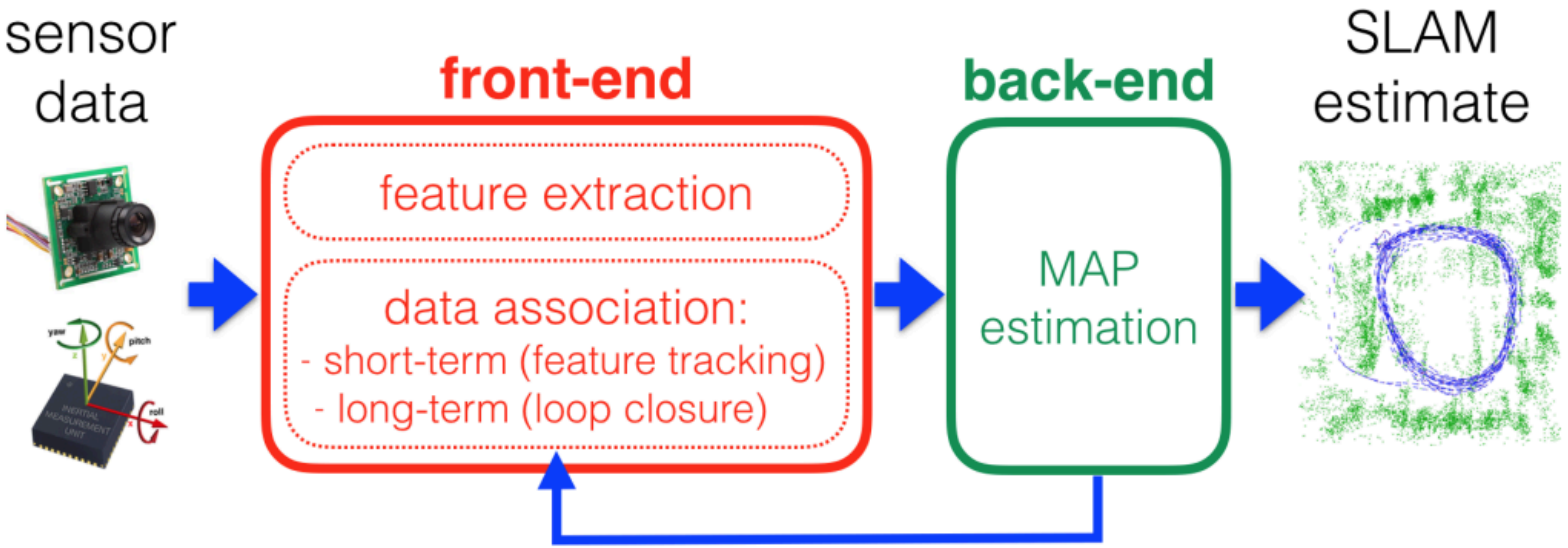


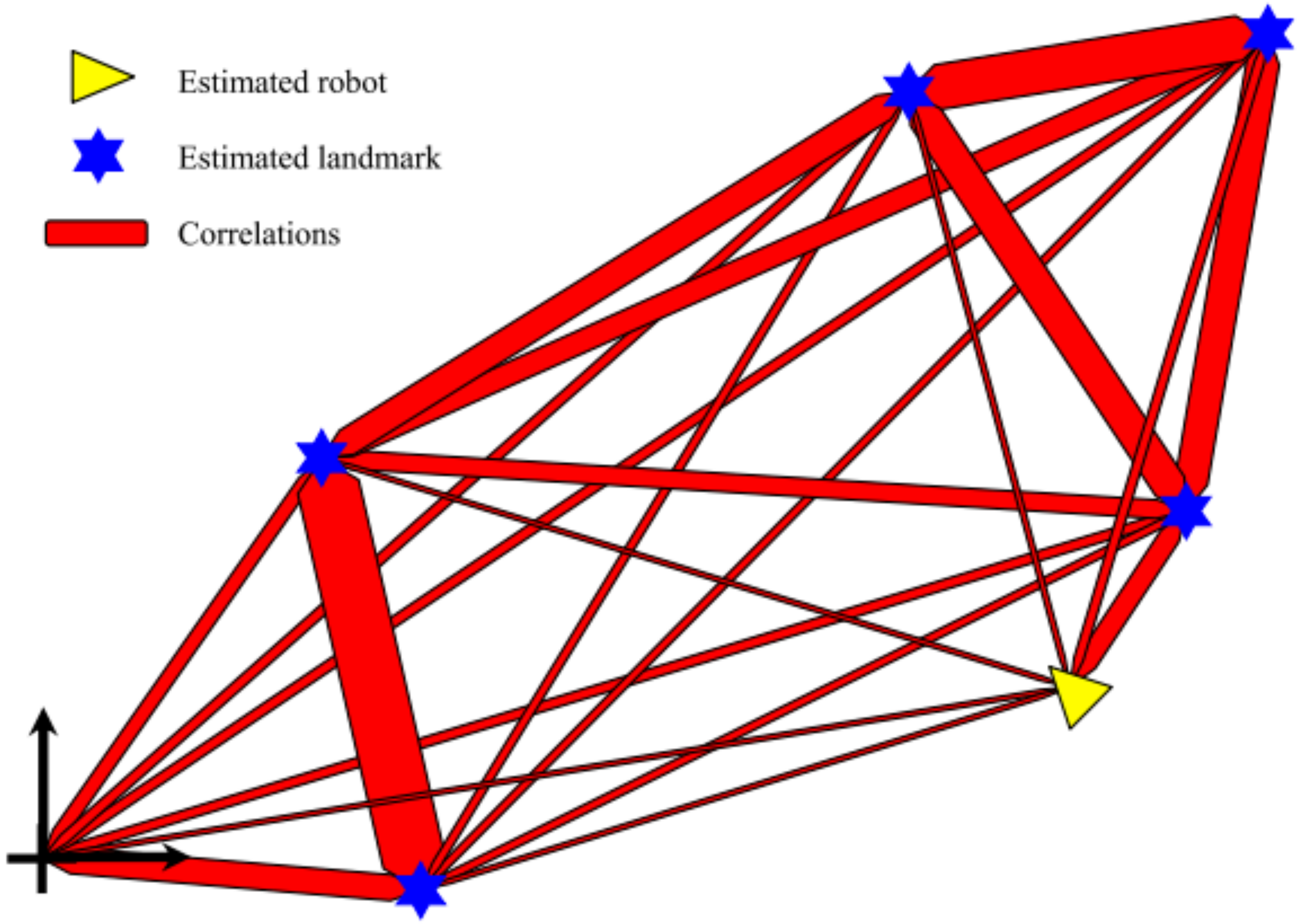
Fig. 2: Front-end and back-end in a typical SLAM system. The back-end can provide feedback to the front-end for loop closure detection and verification.

Algorithm Review

- What are the strengths and weaknesses of:
    - EKF-SLAM
    - Particle Filtering SLAM
    - FastSLAM 1.0 and 2.0
    - GMapping

The SLAM Graph

- Recall the spring analogy from last week's reading.
- Let's formalize this graph:
  - Nodes represent observed landmarks and robot poses.
  - Motion edges connect nodes for x_t and x_{t+1}.
  - Observation edges connect nodes x_t and z_t.



Graph SLAM

- Edges in SLAM graph create constraints:

1-D Measurement constraint: $\frac{(z_t-(x_t+m_t))^2}{\sigma^2}$

1-D Motion constraint: $\frac{(x_t-(x_{t-1}+u_t))^2}{\sigma^2}$

n-D Measurement constraint: $(z_t - h(x_t, m_j))^T Q_t^{-1} (z_t - h(x_t, m_j))$

n-D Motion constraint: $(x_t - g(u_t, x_{t-1}))^T R_t^{-1} (x_t - g(u_t, x_{t-1}))$

- Next, "move" nodes around to minimize sum of constraints.

$$J_{\texttt{GraphSLAM}} = x_0^\top \Omega x_0 + \sum_t (x_t - f(x_{t-1}, u_t))^\top Q^{-1} (x_t - f(x_{t-1}, u_t)) + \sum_t \sum_j (z_t^j - h(x_t, m_t^j))^\top R^{-1} (z_t^j - h(x_t, m_t^j))$$

Moving nodes means optimizing $J_{\texttt{GraphSLAM}}$ with respect to the poses and landmark locations

Graph SLAM Optimization

- In practice, use iterative optimization to find the best position of nodes.
  - Start with a guess $x_{1:t}^0$ and m_0 and then improve guess w.r.t. $J_{\texttt{GraphSLAM}}$ until convergence.
- SLAM graph has special structure that enables fast solving.
- Typically the optimization makes GraphSLAM an offline SLAM algorithm but extensions and fast solvers enable its use for online SLAM.

$$J_{\texttt{GraphSLAM}} = x_0^\top \Omega x_0 + \sum_t (x_t - f(x_{t-1}, u_t))^\top Q^{-1}(x_t - f(x_{t-1}, u_t)) + \sum_t \sum_j (z_t^j - h(x_t, m_t^j))^\top R^{-1}(z_t^j - h(x_t, m_t^j))$$

SLAM as Maximum Likelihood Estimation

- Goal is to estimate most likely map and set of robot poses:

$$p(x_{1:t}, m \,|\, z_{1:t}, u_{1:t}, x_0) \propto p(z_{1:t} \,|\, x_{1:t}, m) p(x_{1:t} \,|\, x_0, u_{1:t})$$

$$= \prod_{i=1}^{t} p(x_i \,|\, x_{i-1}, u_i) \prod_{i=1}^{t} \prod_{j} g(z_i^j \,|\, x_i, m_i^j)$$

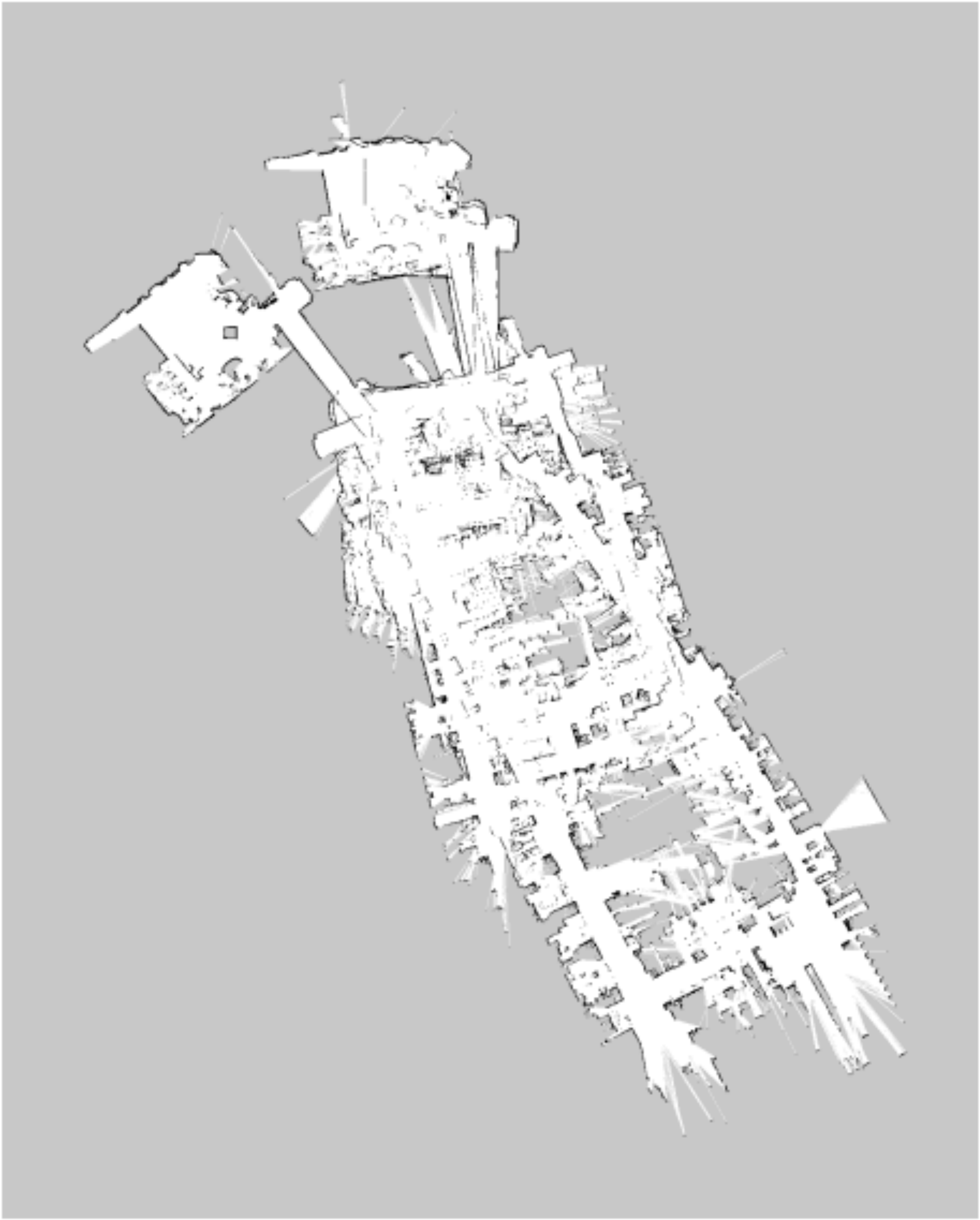
Equivalent to maximizing log likelihood:

$$\sum_{i=1}^{t} \log p(x_i \,|\, x_{i-1}, u_i) + \sum_{i=1}^{t} \sum_{j} \log g(z_i^j \,|\, x_i, m_i^j)$$

Assume Gaussian (but not necessarily linear dynamics and observations):

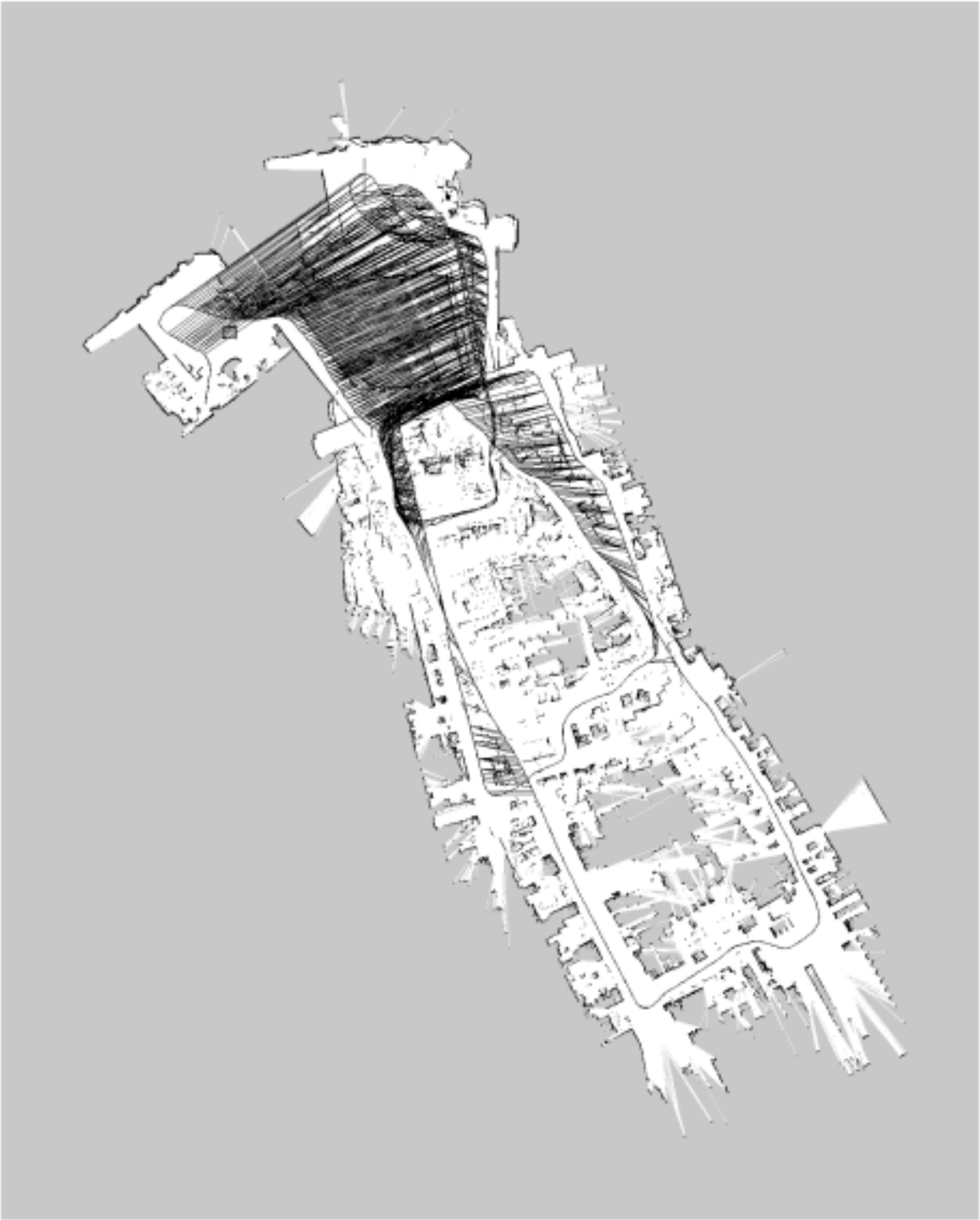
$$J_{\texttt{GraphSLAM}} = x_0^\top \Omega x_0 + \sum_{t} (x_t - f(x_{t-1}, u_t))^\top Q^{-1} (x_t - f(x_{t-1}, u_t)) + \sum_{t} \sum_{j} (z_t^j - h(x_t, m_t^j))^\top R^{-1} (z_t^j - h(x_t, m_t^j))$$

SLAM as Maximum Likelihood Estimation



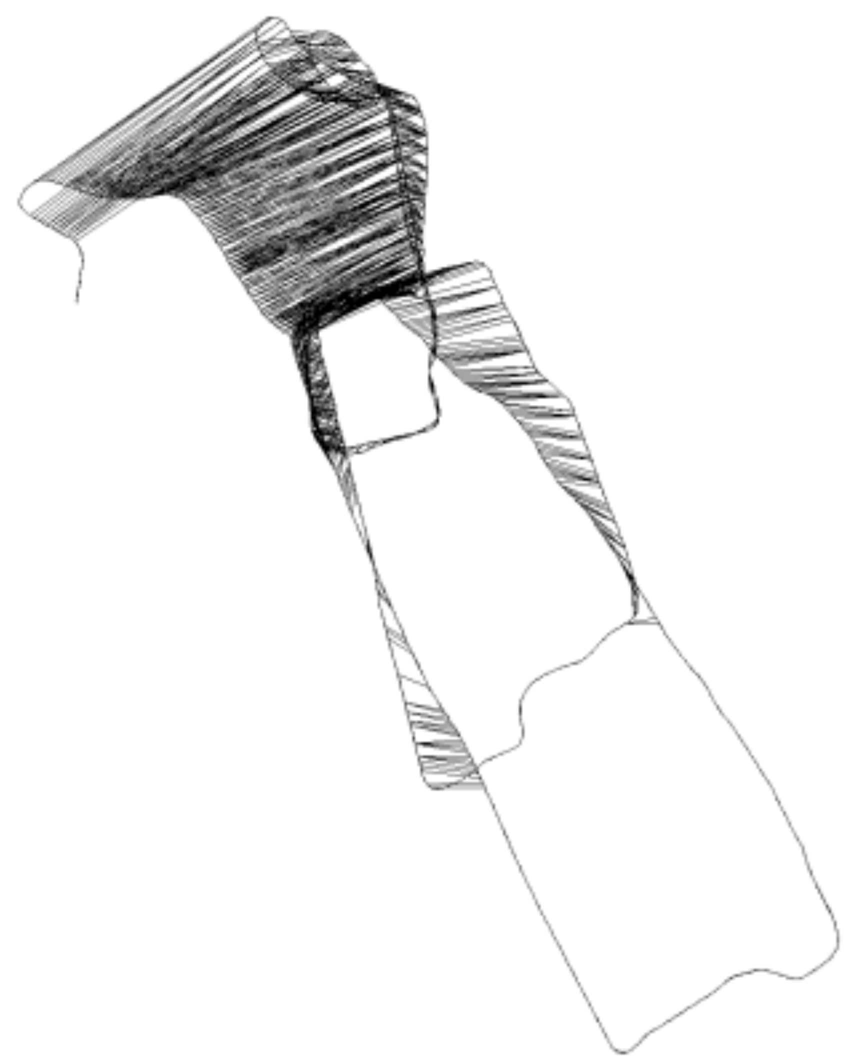
[KUKA Hall 22, courtesy P. Pfaff & G. Grisetti]

SLAM as Maximum Likelihood Estimation



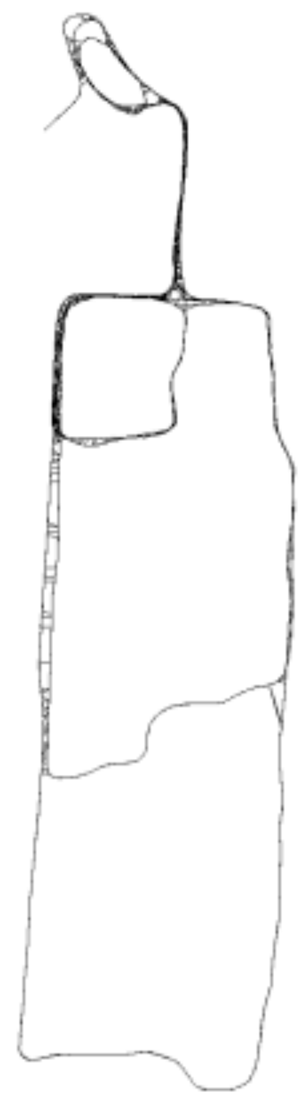
[KUKA Hall 22, courtesy P. Pfaff & G. Grisetti]

SLAM as Maximum Likelihood Estimation



[KUKA Hall 22, courtesy P. Pfaff & G. Grisetti]

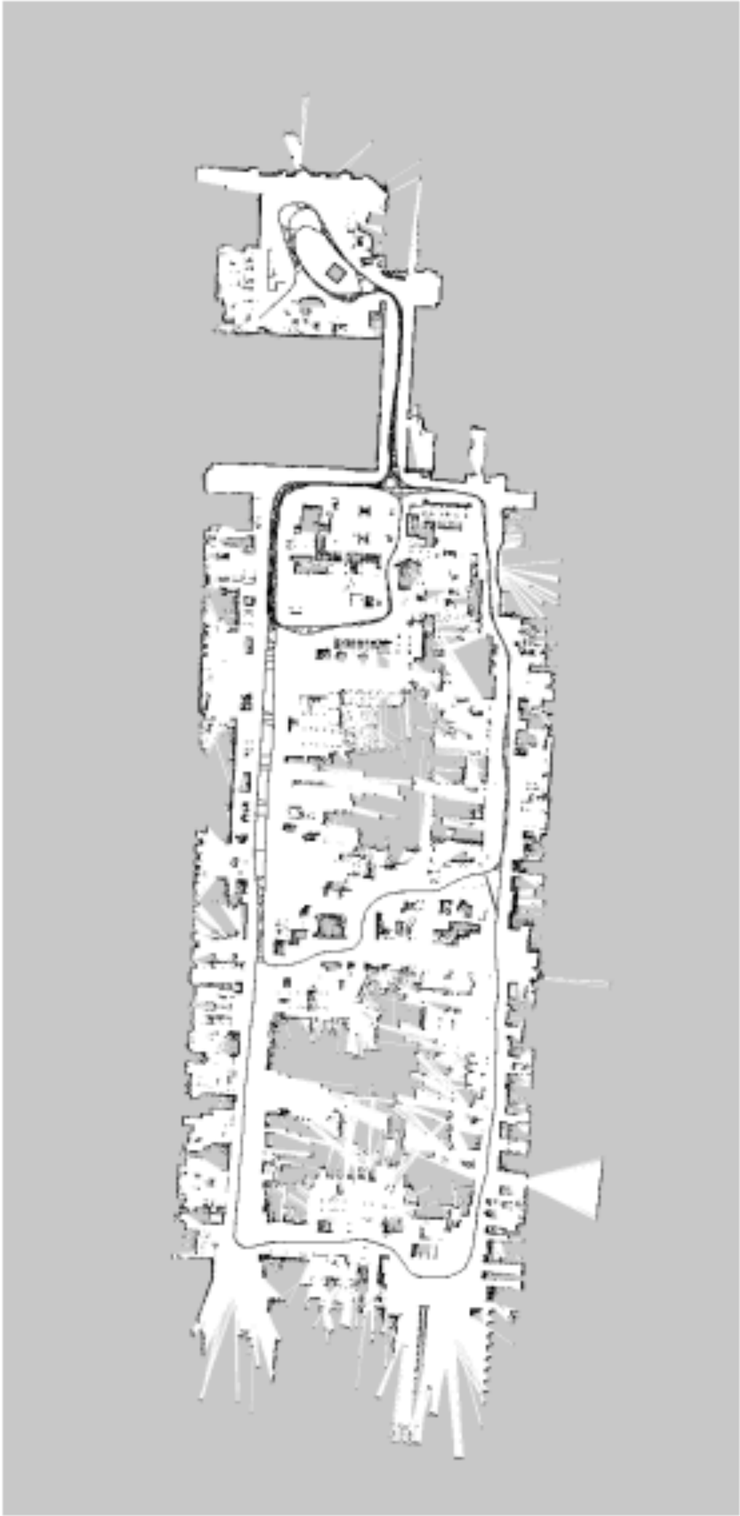
SLAM as Maximum Likelihood Estimation



[KUKA Hall 22, courtesy P. Pfaff & G. Grisetti]

i]

SLAM as Maximum Likelihood Estimation



[KUKA Hall 22, courtesy P. Pfaff & G. Grisetti]

Active SLAM

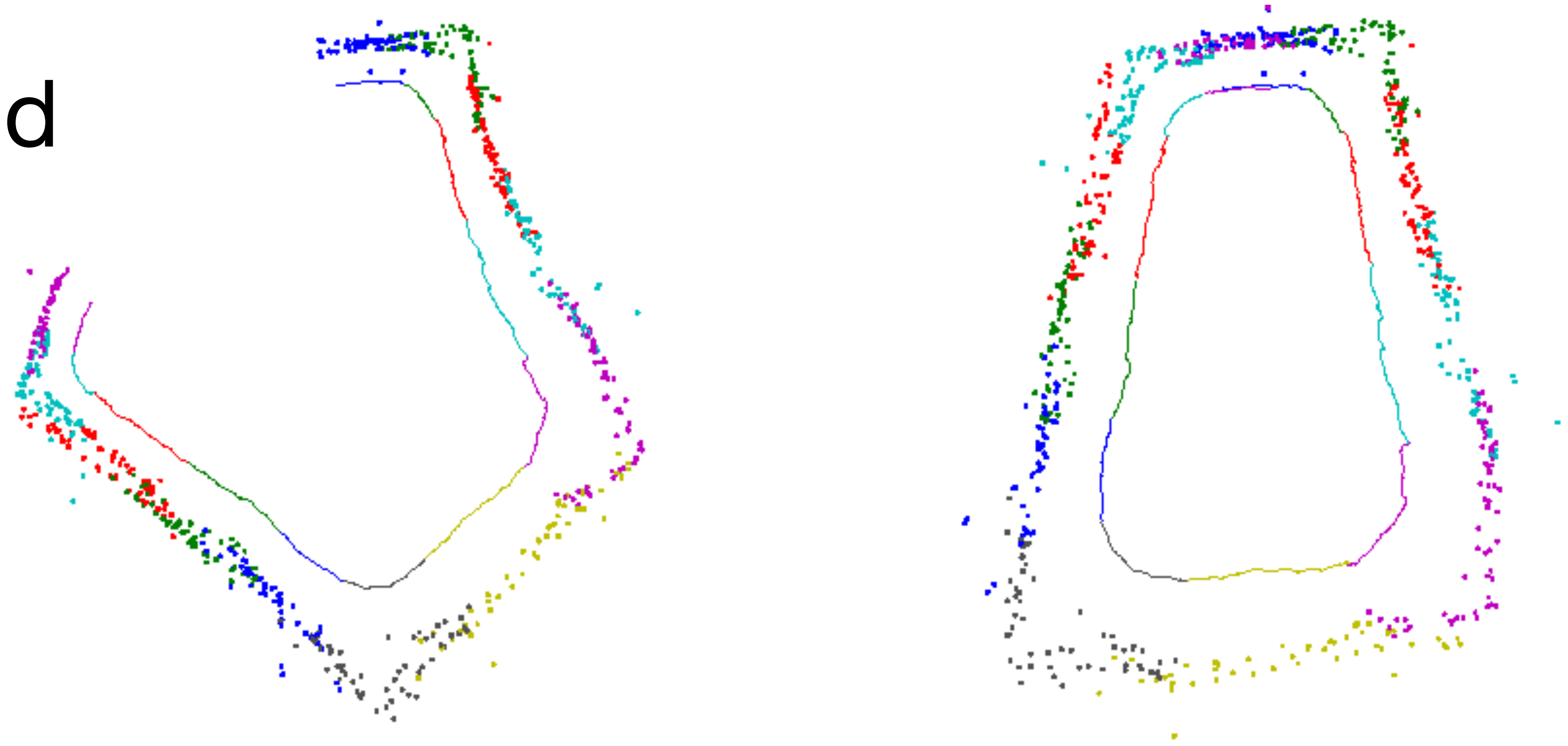
- So far we have ignored the choice of the robot's controls, but they are critical for optimal mapping.
  - If the robot stays in one area or only visits known areas, then the map will not improve.
- Where to go next? What should a robot consider?
  - Discovering new parts of the map.
  - Maintaining its own belief about where it is.
- How to determine where to go?
  - Optimal design
  - Information gain
  - Reinforcement learning / POMDP planning

Handling multiple sensor types

- Modern robots often have multiple sensor types, e.g., lidar, cameras, IMUs.
- The Bayes filter (and descendants) can handle this by running multiple update steps, one for each sensor's observation.
  - $\overline{\texttt{bel}}(x_t) \propto g(z_t^i \mid x_t)$ where $\overline{\texttt{bel}}$ is the belief after the prediction step and the update step for other observations z_t^j.

Robustness

- Hardware failure
- Robustness to incorrect loop closure
- Dynamic maps
- Integration of the front-end and back-end



Scalability

- Need to forget?
- Handling new types of sensors?
  - E.g., event cameras
- Integrating semantic information
- Handling resource constraints

Summary

- Introduced GraphSLAM
- Discussed extensions and future directions for the SLAM problem.

Action Items

- Kinematics reading for next week; send a reading response by 12 pm on Monday.
- SLAM assignment due in 1 week.