

# Principles of Robot Autonomy I

Simultaneous Localization and Mapping (SLAM)



**Stanford**  
University



# Attendance Form

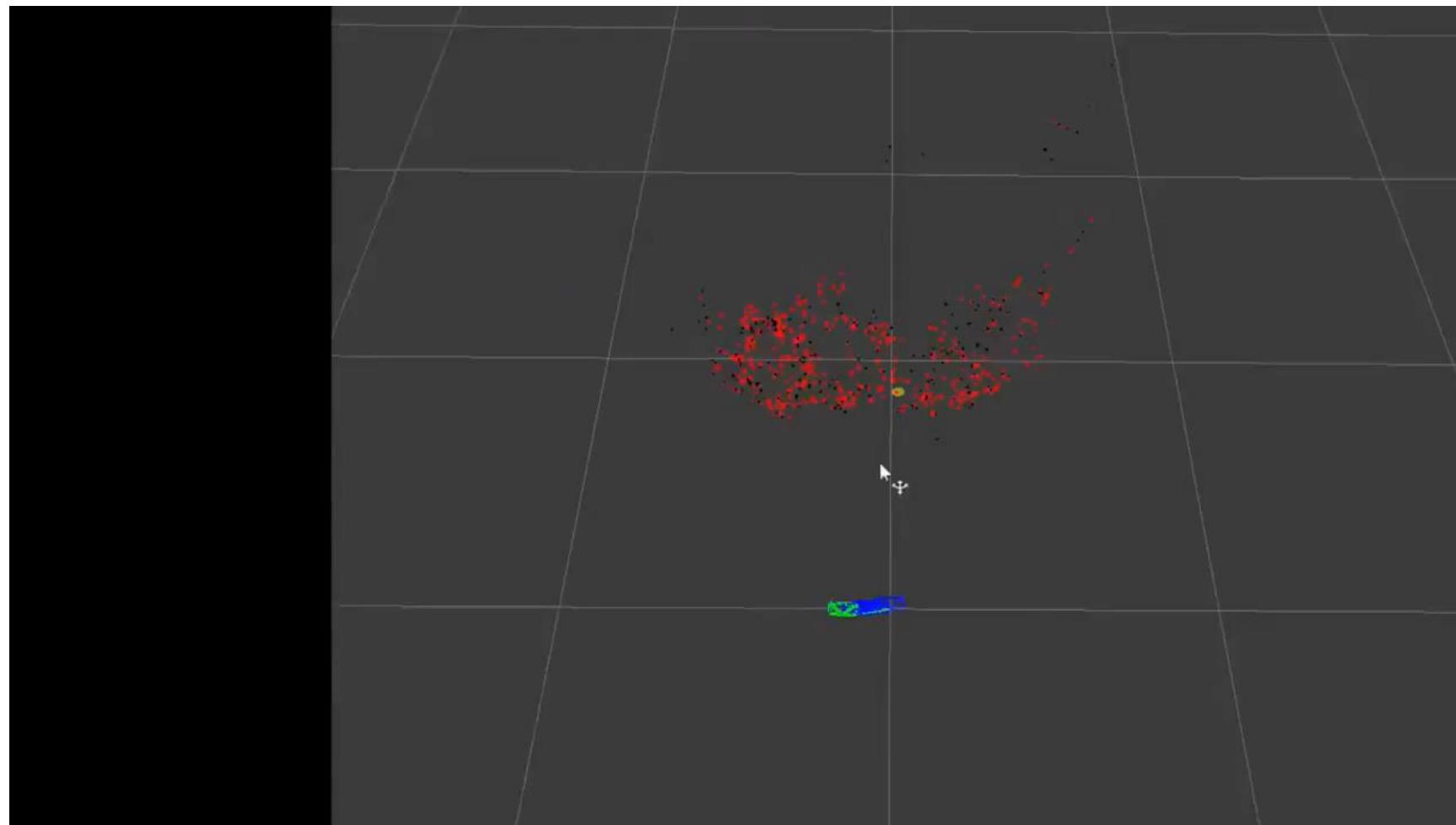


# Agenda

- Aim
  - General SLAM problem
  - EKF SLAM
- Readings
  - Chapter 16, sections 16.1 – 16.4 in D. Gammelli, J. Lorenzetti, K. Luo, G. Zardini, M. Pavone. *Principles of Robot Autonomy*. 2026.

# Simultaneous Localization and Mapping

The SLAM problem:  
**given** measurements  
 $z_{1:t}$  and controls  $u_{1:t}$ ,  
**find** the path (or pose)  
of the robot and  
acquire a map of the  
environment



# Forms of SLAM

- **Online SLAM problem:** estimate the posterior over the momentary pose along with the map

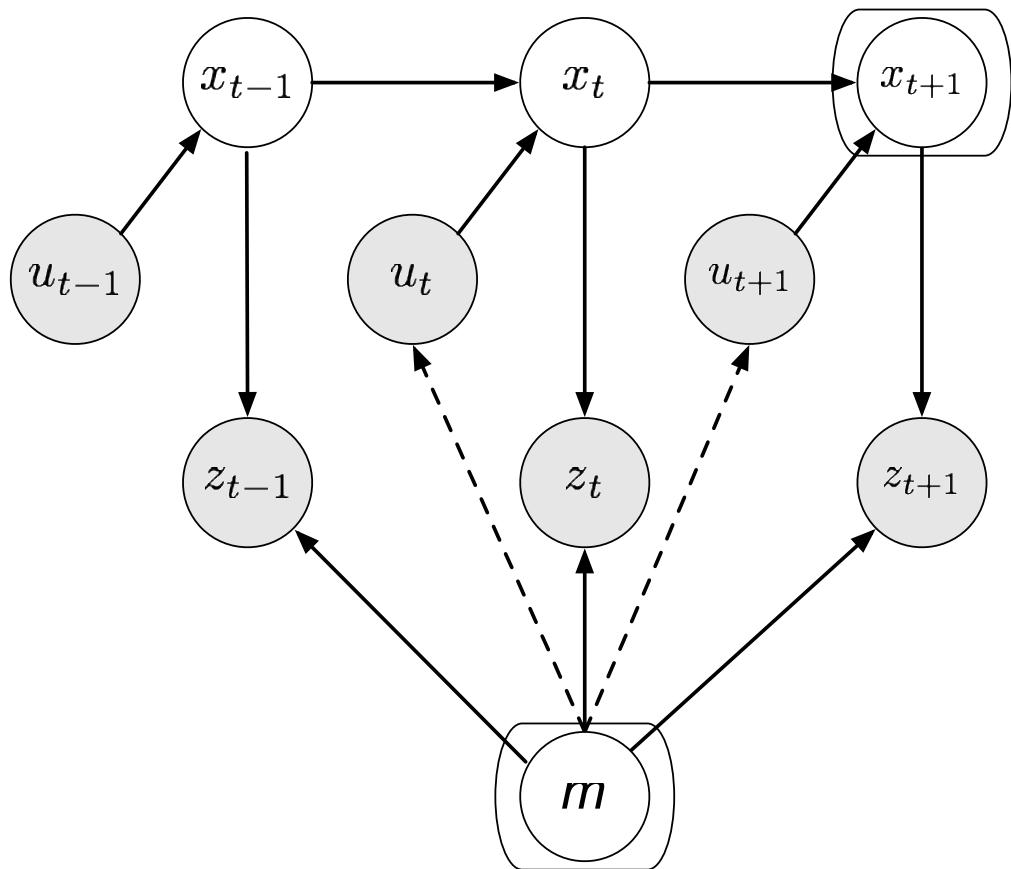
$$p(x_t, m \mid z_{1:t}, u_{1:t}) \quad \text{or} \quad p(x_t, m, c_t \mid z_{1:t}, u_{1:t})$$

- **Full SLAM problem:** estimate posterior over the entire path along with the map

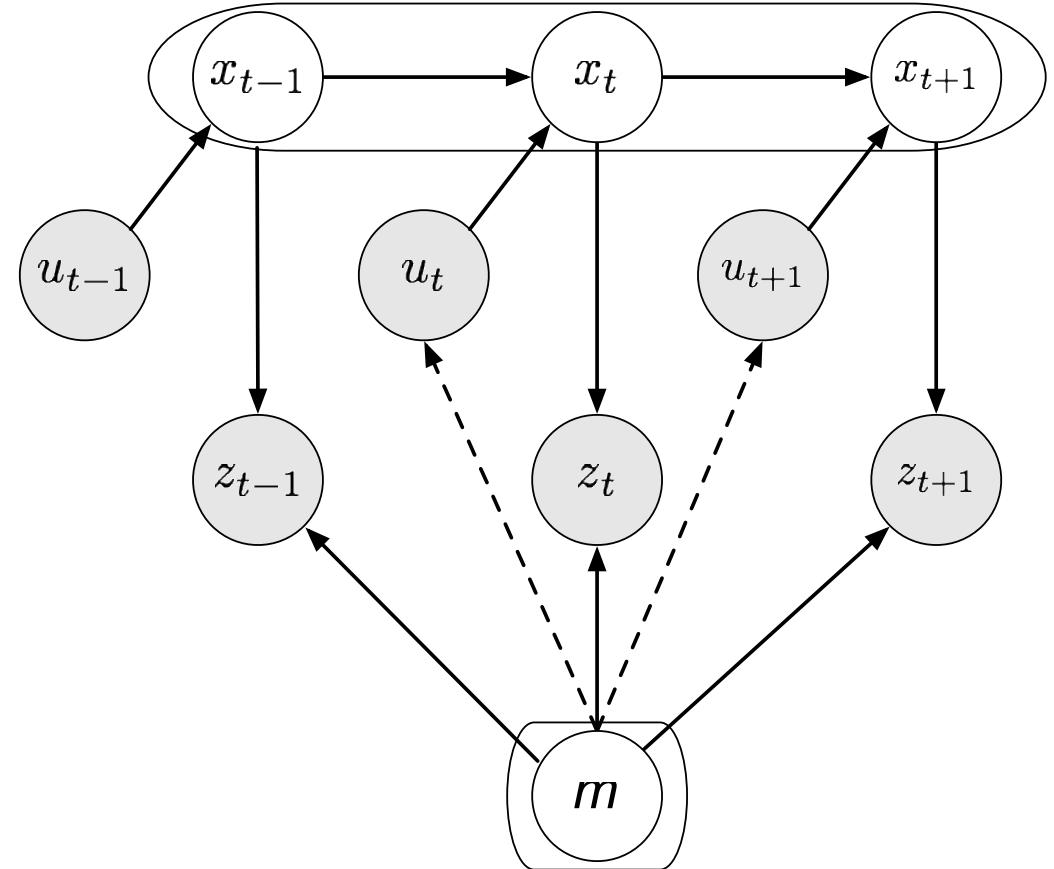
$$p(x_{1:t}, m \mid z_{1:t}, u_{1:t}) \quad \text{or} \quad p(x_{1:t}, m, c_t \mid z_{1:t}, u_{1:t})$$

# Graphical models of SLAM

Online SLAM



Full SLAM



# EKF SLAM

- Historically the earliest SLAM algorithm
- **Key idea:** apply EKF to online SLAM using maximum likelihood data association
- Assumptions:
  1. Gaussian assumption for motion and perception noise, and Gaussian approximation for belief (essential)
  2. Feature-based maps (essential)
- Two versions of the problem
  1. Correspondence variables are known
  2. Correspondence variables are not known (usual case)

# EKF SLAM with known correspondences

- Similar to EKF localization algorithm with known correspondences
- **Key difference:** in addition to estimate the robot pose  $x_t$ , the EKF SLAM algorithm also estimates the coordinates of **all** landmarks
- Define combined state vector

$$y_t := \begin{pmatrix} x_t \\ m \end{pmatrix} = (x, y, \theta, m_{1,x}, m_{1,y}, m_{2,x}, m_{2,y} \dots m_{N,x}, m_{N,y})^T$$

3 + 2N vector

- **Goal:** calculate the **online posterior**

$$p(y_t | z_{1:t}, u_{1:t})$$

# Motion and sensing model

- (Following discussion is for illustration purposes; setup can be generalized to other motion and sensing models)
- Assume motion model with state  $x_t = (x, y, \theta)$

$$y_t = g(u_t, y_{t-1}) + \epsilon_t, \quad \epsilon_t \sim \mathcal{N}(0, R_t), \quad G_t := J_g(u_t, \mu_{t-1})$$

where we assume that the landmarks are *static*, that is

1.  $g(u_t, y_{t-1})$  is a  $3+2N$  vector, whose last  $2N$  components are the same as those in  $y_{t-1}$
2.  $R_t$  has zero entries, except for the top left  $3 \times 3$  block

# Motion and sensing model

- Assume range and bearing measurement model

$$z_t^i = \underbrace{\left( \sqrt{(m_{j,x} - x)^2 + (m_{j,y} - y)^2} \right.}_{:= h(y_t, j)} \left. + \text{atan2}(m_{j,y} - y, m_{j,x} - x) - \theta \right) + \delta_t, \quad \delta_t \sim \mathcal{N}(0, Q_t), \quad Q_t = \begin{pmatrix} \sigma_r^2 & 0 \\ 0 & \sigma_\phi^2 \end{pmatrix}$$

- Usual linear approximation for sensing model (with  $j = c_t^i$ )

$$h(y_t, j) \approx h(\bar{\mu}_t, j) + H_t^i (y_t - \bar{\mu}_t), \quad \text{where } H_t^i := \frac{\partial h(\bar{\mu}_t, j)}{\partial y_t}$$

- Since  $h$  depends only on  $x_t$  and  $m_j$ ,  $H_t^i$  can be factored as

$$H_t^i = h_t^i F_{x,j}$$

# Motion and sensing model

- First term, a  $2 \times 5$  matrix, is the Jacobian of  $h(y_t, j)$  at  $\bar{\mu}_t$  w.r.t.  $x_t$  and  $m_j$ :

$$h_t^i = \frac{\partial h(\bar{\mu}_t, j)}{\partial (x_t, m_j)} = \begin{pmatrix} \frac{\bar{\mu}_{t,x} - \bar{\mu}_{j,x}}{\sqrt{q_{t,j}}} & \frac{\bar{\mu}_{t,y} - \bar{\mu}_{j,y}}{\sqrt{q_{t,j}}} & 0 & \frac{\bar{\mu}_{j,x} - \bar{\mu}_{t,x}}{\sqrt{q_{t,j}}} & \frac{\bar{\mu}_{j,y} - \bar{\mu}_{t,y}}{\sqrt{q_{t,j}}} \\ \frac{\bar{\mu}_{j,y} - \bar{\mu}_{t,y}}{\sqrt{q_{t,j}}} & \frac{\bar{\mu}_{t,x} - \bar{\mu}_{j,x}}{\sqrt{q_{t,j}}} & -1 & \frac{\bar{\mu}_{t,y} - \bar{\mu}_{j,y}}{\sqrt{q_{t,j}}} & \frac{\bar{\mu}_{j,x} - \bar{\mu}_{t,x}}{\sqrt{q_{t,j}}} \end{pmatrix}$$

where  $q_{t,j} := (\bar{\mu}_{j,x} - \bar{\mu}_{t,x})^2 + (\bar{\mu}_{j,y} - \bar{\mu}_{t,y})^2$

- Second term, a  $5 \times (3+2N)$  matrix, maps  $h_t^i$  into  $H_t^i$ :

$$F_{x,j} = \begin{pmatrix} 1 & 0 & 0 & 0 \cdots 0 & 0 & 0 & 0 \cdots 0 \\ 0 & 1 & 0 & 0 \cdots 0 & 0 & 0 & 0 \cdots 0 \\ 0 & 0 & 1 & 0 \cdots 0 & 0 & 0 & 0 \cdots 0 \\ 0 & 0 & 0 & 0 \cdots 0 & 1 & 0 & 0 \cdots 0 \\ 0 & 0 & 0 & \underbrace{0 \cdots 0}_{2j-2} & 0 & 1 & \underbrace{0 \cdots 0}_{2N-2j} \end{pmatrix}$$

# Initialization

- Initial belief expressed as

$$\mu_0 = (0, 0, 0 \dots 0)^T$$

Initialization  
for pose  
variables

$\Sigma_0 = \begin{pmatrix} 0 & 0 & 0 & 0 & \dots & 0 \\ 0 & 0 & 0 & 0 & \dots & 0 \\ 0 & 0 & 0 & 0 & \dots & 0 \\ 0 & 0 & 0 & \infty & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & \dots & \infty \end{pmatrix}$

(3+2N) x (3+2N)

# Initialization

- When a landmark is observed for the first time, the landmark estimate  $(\bar{\mu}_{j,x}, \bar{\mu}_{j,y})^T$  is initialized with the expected position, that is

$$\begin{pmatrix} \bar{\mu}_{j,x} \\ \bar{\mu}_{j,y} \end{pmatrix} = \begin{pmatrix} \bar{\mu}_{t,x} \\ \bar{\mu}_{t,y} \end{pmatrix} + \begin{pmatrix} r_t^i \cos(\phi_t^i + \bar{\mu}_{t,\theta}) \\ r_t^i \sin(\phi_t^i + \bar{\mu}_{t,\theta}) \end{pmatrix}$$

# EKF SLAM algorithm

- Similar to EKF localization; main differences:
  - Augmented state vector
  - Augmented dynamics (with trivial dynamics for the landmarks)
  - Initialization of unseen landmarks
  - Augmented measurement Jacobian

**Data:**  $(\mu_{t-1}, \Sigma_{t-1}), u_t, z_t, c_t$   
**Result:**  $(\mu_t, \Sigma_t)$

$$\bar{\mu}_t = g(u_t, \mu_{t-1});$$
$$\bar{\Sigma}_t = G_t \Sigma_{t-1} G_t^T + R_t;$$

**foreach**  $z_t^i = (r_t^i, \phi_t^i)^T$  **do**

$$j = c_t^i;$$

**if** landmark  $j$  never seen before **then**

$$\begin{pmatrix} \bar{\mu}_{j,x} \\ \bar{\mu}_{j,y} \end{pmatrix} = \begin{pmatrix} \bar{\mu}_{t,x} \\ \bar{\mu}_{t,y} \end{pmatrix} + \begin{pmatrix} r_t^i \cos(\phi_t^i + \bar{\mu}_{t,\theta}) \\ r_t^i \sin(\phi_t^i + \bar{\mu}_{t,\theta}) \end{pmatrix};$$

**end**

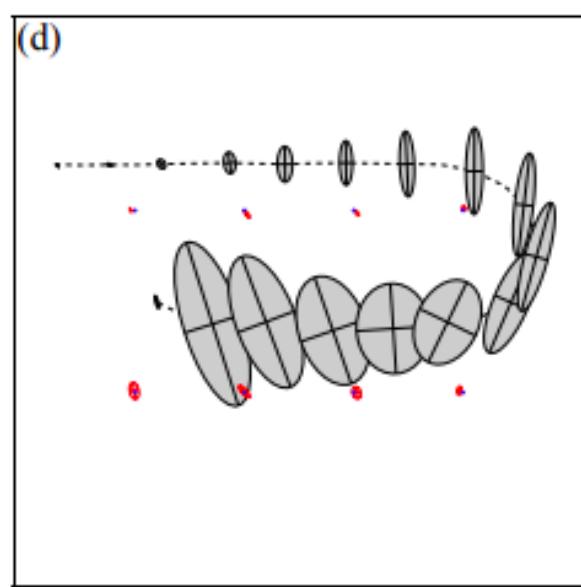
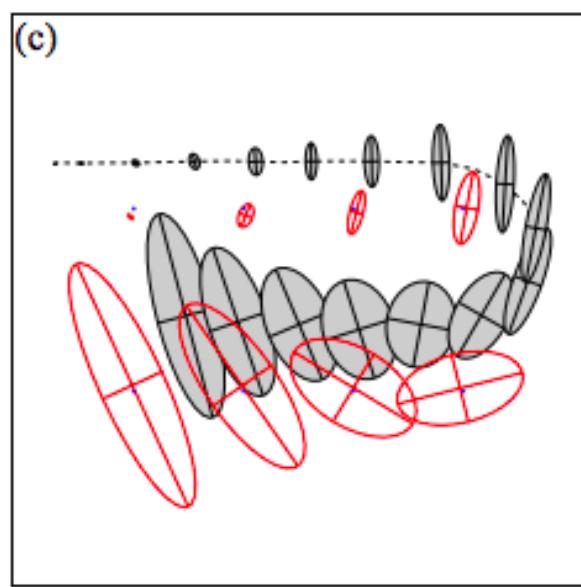
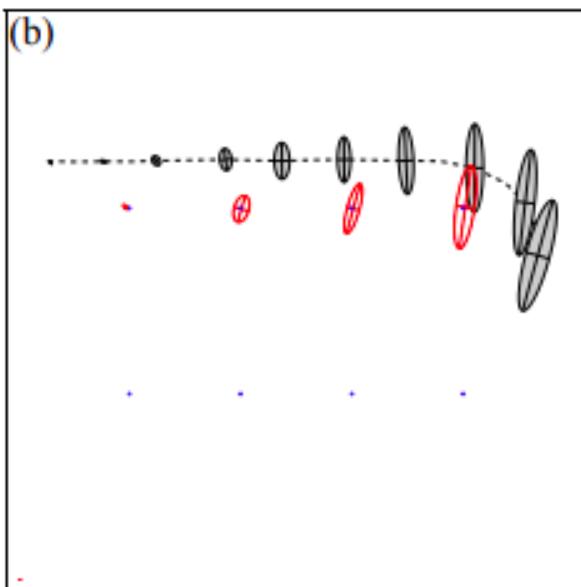
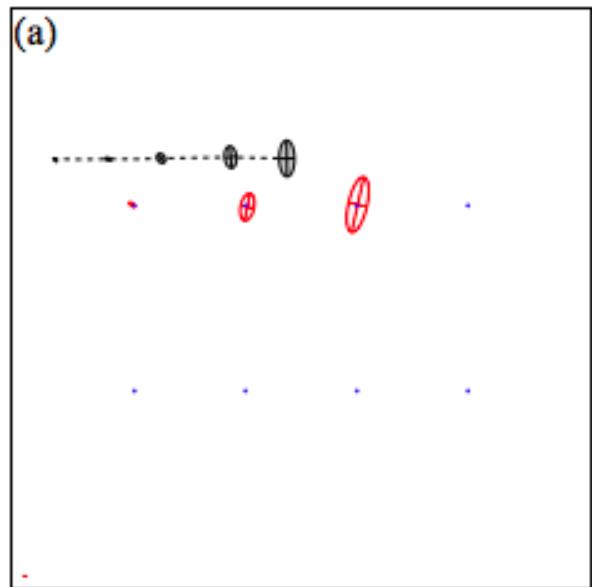
$$\hat{z}_t^i = \begin{pmatrix} \sqrt{(\bar{\mu}_{j,x} - \bar{\mu}_{t,x})^2 + (\bar{\mu}_{j,y} - \bar{\mu}_{t,y})^2} \\ \text{atan2}(\bar{\mu}_{j,y} - \bar{\mu}_{t,y}, \bar{\mu}_{j,x} - \bar{\mu}_{t,x}) - \bar{\mu}_{t,\theta} \end{pmatrix};$$
$$H_t^i = h_t^i F_{x,j};$$
$$S_t^i = H_t^i \bar{\Sigma}_t [H_t^i]^T + Q_t;$$
$$K_t^i = \bar{\Sigma}_t [H_t^i]^T [S_t^i]^{-1};$$
$$\bar{\mu}_t = \bar{\mu}_t + K_t^i (z_t^i - \hat{z}_t^i);$$
$$\bar{\Sigma}_t = (I - K_t^i H_t^i) \bar{\Sigma}_t;$$

**end**

$$\mu_t = \bar{\mu}_t \text{ and } \Sigma_t = \bar{\Sigma}_t;$$

Return  $(\mu_t, \Sigma_t)$

# Example



# EKF SLAM with unknown correspondences

- **Key idea:** use an incremental maximum likelihood estimator to determine correspondences
- Similar to EKF localization with unknown correspondences, but now we also need to create hypotheses for new landmarks
- **Caveat:** maximum likelihood data association often makes the algorithm brittle, as it is not possible to revise past data associations

# EKF SLAM with unknown correspondences

- In the measurement update loop, we first create the hypothesis of a new landmark
- A new landmark is created if the Mahalanobis distance to all existing landmarks exceeds the value  $\alpha$

**Data:**  $(\mu_{t-1}, \Sigma_{t-1}), u_t, z_t, N_{t-1}$   
**Result:**  $(\mu_t, \Sigma_t)$

$$N_t = N_{t-1};$$

$$\bar{\mu}_t = g(u_t, \mu_{t-1});$$

$$\bar{\Sigma}_t = G_t \Sigma_{t-1} G_t^T + R_t;$$

Hypothesis for new landmark

```

foreach  $z_t^i = (r_t^i, \phi_t^i)^T$  do
    
$$\begin{pmatrix} \bar{\mu}_{N_t+1,x} \\ \bar{\mu}_{N_t+1,y} \end{pmatrix} = \begin{pmatrix} \bar{\mu}_{t,x} \\ \bar{\mu}_{t,y} \end{pmatrix} + \begin{pmatrix} r_t^i \cos(\phi_t^i + \bar{\mu}_{t,\theta}) \\ r_t^i \sin(\phi_t^i + \bar{\mu}_{t,\theta}) \end{pmatrix};$$

    for  $k = 1$  to  $N_t + 1$  do
        
$$\hat{z}_t^k = \begin{pmatrix} \sqrt{(\bar{\mu}_{j,x} - \bar{\mu}_{t,x})^2 + (\bar{\mu}_{j,y} - \bar{\mu}_{t,y})^2} \\ \text{atan2}(\bar{\mu}_{j,y} - \bar{\mu}_{t,y}, \bar{\mu}_{j,x} - \bar{\mu}_{t,x}) - \bar{\mu}_{t,\theta} \end{pmatrix};$$

        
$$H_t^k = h_t^k F_{x,k};$$

        
$$S_t^k = H_t^k \bar{\Sigma}_t [H_t^k]^T + Q_t;$$

        
$$\pi_k = (z_t^i - \hat{z}_t^k)^T [S_t^k]^{-1} (z_t^i - \hat{z}_t^k);$$

end

```

Mahalanobis distance

$$\pi_{N_t+1} = \alpha;$$

$$j(i) = \operatorname{argmin}_k \pi_k; \quad \text{Hypothesis test}$$

$$N_t = \max\{N_t, j(i)\};$$

$$K_t^i = \bar{\Sigma}_t [H_t^{j(i)}]^T [S_t^{j(i)}]^{-1};$$

$$\bar{\mu}_t = \bar{\mu}_t + K_t^i (z_t^i - \hat{z}_t^{j(i)});$$

$$\bar{\Sigma}_t = (I - K_t^i H_t^{j(i)}) \bar{\Sigma}_t;$$

**end**

$$\mu_t = \bar{\mu}_t \text{ and } \Sigma_t = \bar{\Sigma}_t;$$

Return  $(\mu_t, \Sigma_t)$

# Making EKF SLAM robust

- A key issue is represented by the fact that **fake landmarks** might be created; furthermore, EKF can **diverge** if nonlinearities are large
- Several techniques exist to mitigate such issues
  1. Outlier rejection schemes, for example via provisional landmark lists
  2. Strategies to enhance the distinctiveness of landmarks
    - Spatial arrangement
    - Signatures
    - Enforcing geometric constraints
- **Dilemma of EKF SLAM:** accurate localization typically requires dense maps, but EKF requires sparse maps due to quadratic update complexity

# Summary: Gaussian filtering

- **Key ideas:**
  - Represent a belief with a Gaussian distribution
  - Assume all uncertainty sources are Gaussian
- **Pros:**
  - Runs online
  - Well understood
  - Works well when uncertainty is low
- **Cons:**
  - Unimodal estimate
  - States must be well approximated by a Gaussian
  - Works poorly when uncertainty is high

# Final considerations

- A quite recent overview of SLAM (with strong focus on graph SLAM): c. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, and J. J. Leonard. "Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age." *IEEE Transactions on Robotics* 32, no. 6 (2016): 1309-1332.
- Popular software packages
  - <https://www.openslam.org/>: comprehensive list of open-source SLAM software
  - <https://github.com/pamela-project/slambench>: popular benchmark framework
  - Commercial SDKs: ARCore/ARKit from Google/Apple, Oculus Insight

Thanks for a great quarter!