Principles of Robot Autonomy I

Simultaneous Localization and Mapping (SLAM)

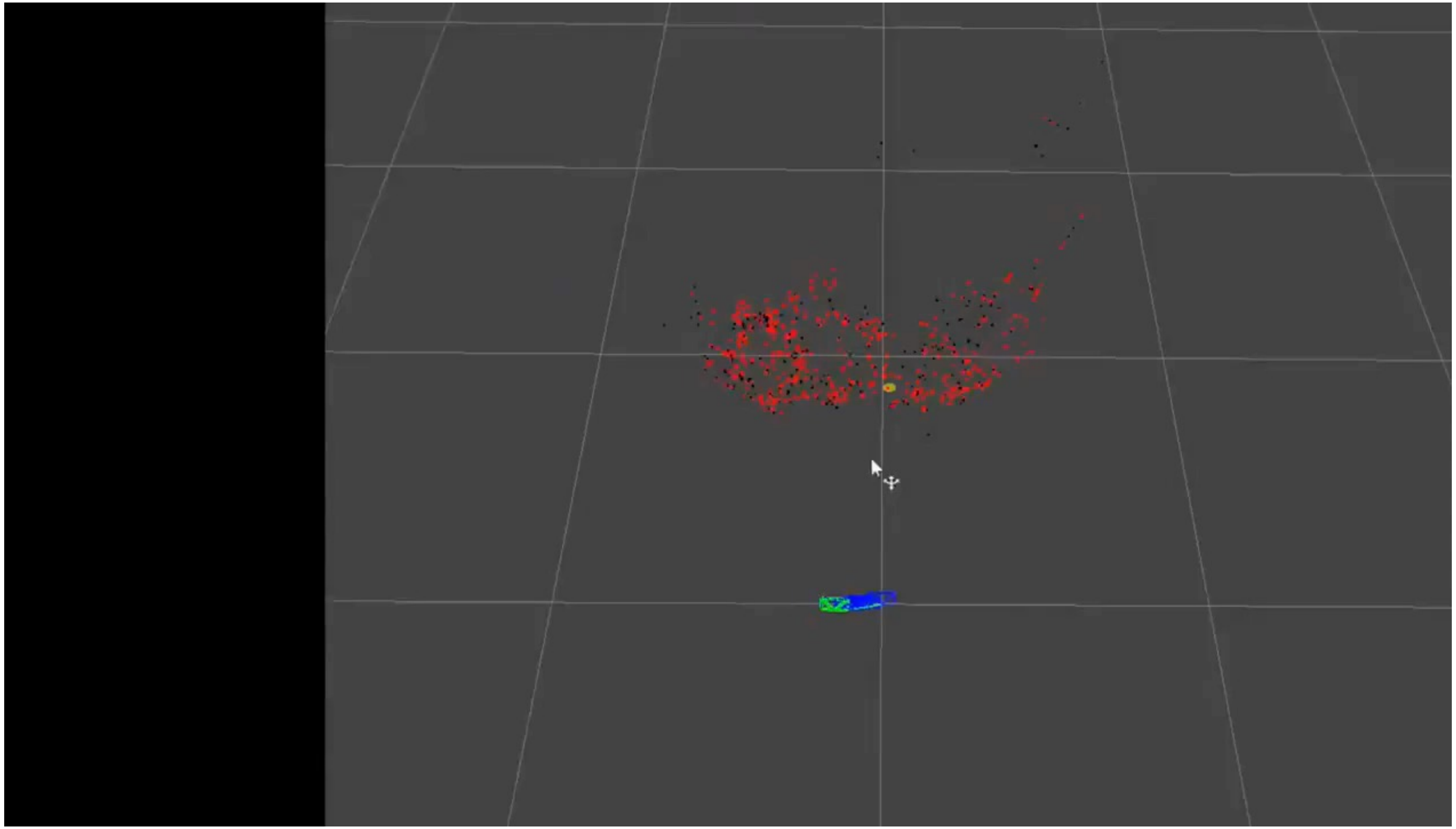
Today's lecture

- Aim
  - Learn about the general SLAM problem
  - Learn about EKF SLAM
  - Introduce particle filter SLAM

- Readings
  - S. Thrun, W. Burgard, and D. Fox. Probabilistic robotics. MIT press, 2005. Sections 8.1 – 8.3, 10.1 – 10.4
  - S. Thrun, W. Burgard, and D. Fox. Probabilistic robotics. MIT press, 2005. Sections 13.1-13.3, 13.5

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

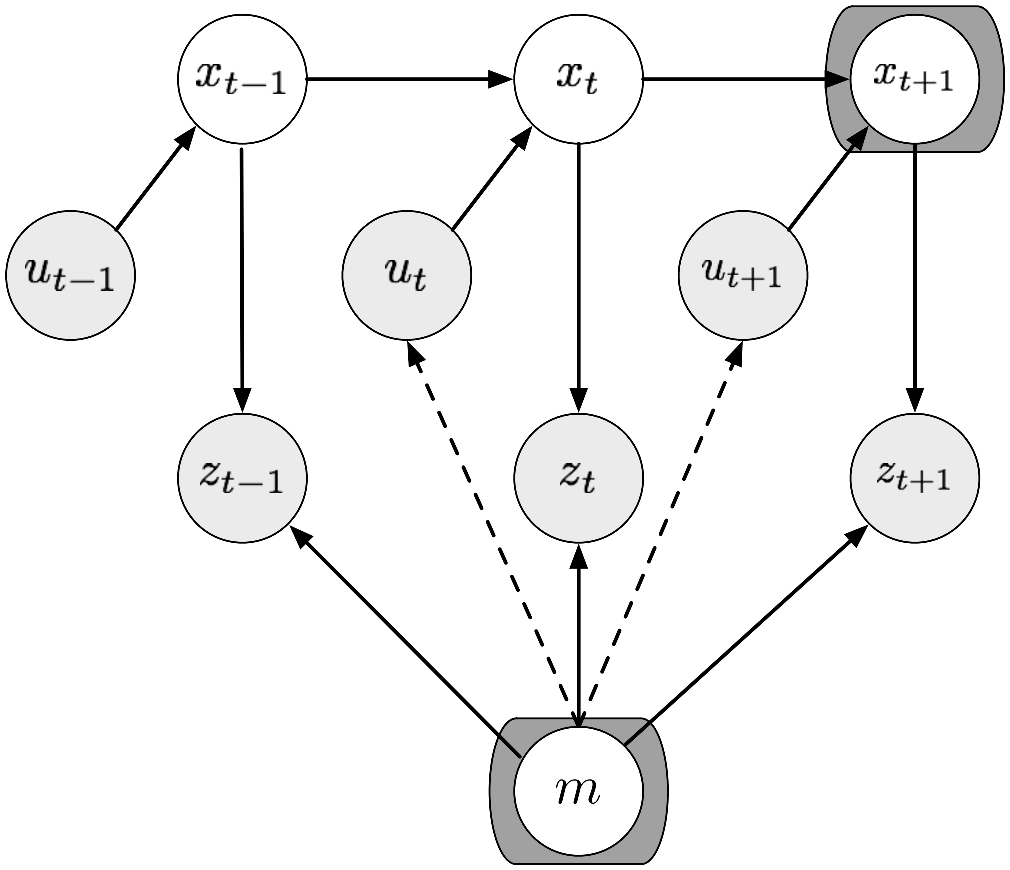
$$p(x_t, m \mid z_{1:t}, u_{1:t}) \qquad \text{or} \qquad p(x_t, m, c_t \mid z_{1:t}, u_{1:t})$$

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

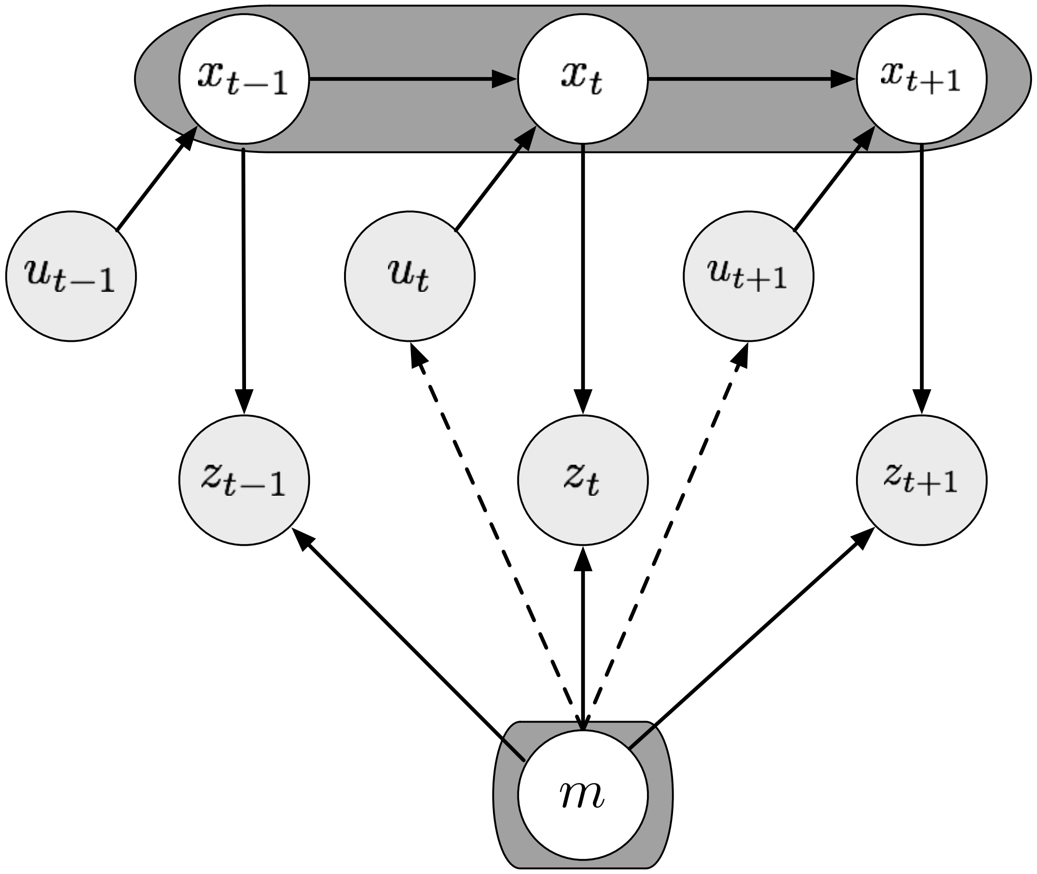
$$p(x_{1:t}, m \mid z_{1:t}, u_{1:t}) \qquad \text{or} \qquad p(x_{1:t}, m, c_t \mid z_{1:t}, u_{1:t})$$

Graphical models of SLAM

Online SLAM

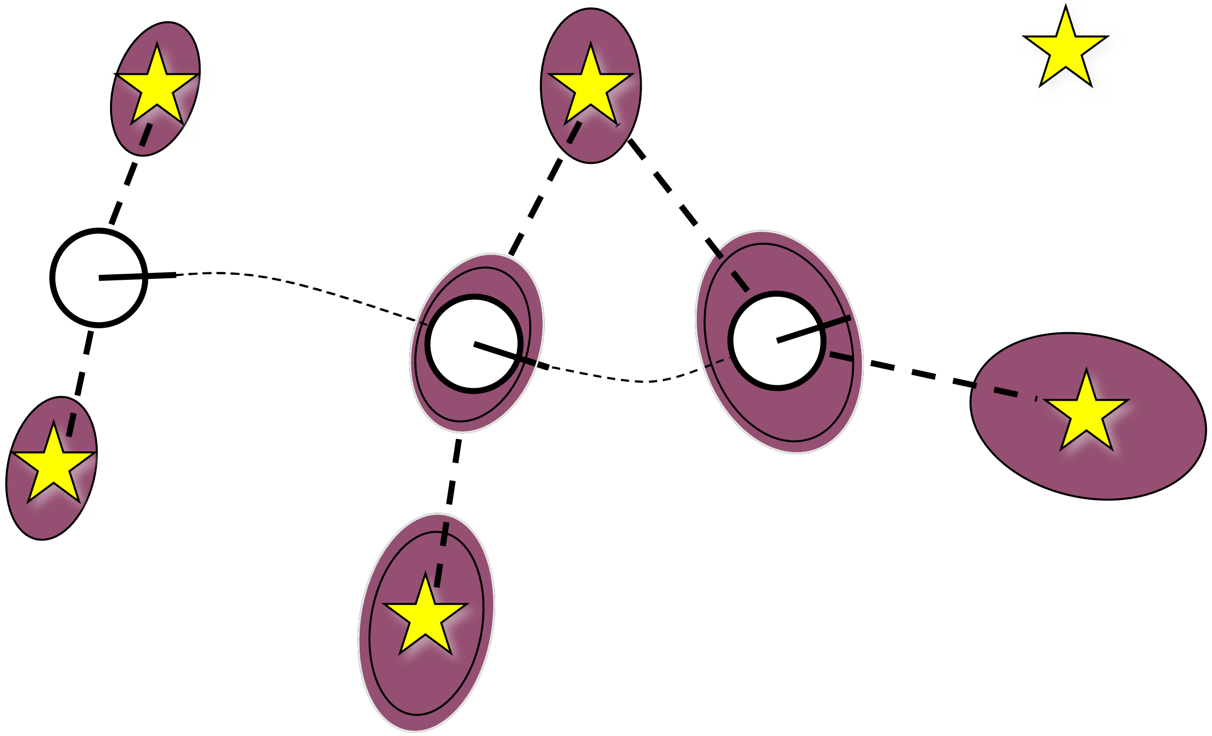


Full SLAM



The challenge of SLAM

- Robot path and map are both unknown



- Path error is correlated with map error

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

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

3 + 2N vector

- Goal: calculate the online posterior

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

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 $x_t = (x, y, \theta)$

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

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 x 3 block

Motion and sensing model

- Assume range and bearing measurement model

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

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

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

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

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

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

- Second term, a 5 x (3+2*N*) 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

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

Initialization for pose variables

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

$$\Sigma_0 = \begin{pmatrix} \boxed{\begin{matrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{matrix}} & \begin{matrix} 0 & \ldots & 0 \\ 0 & \ldots & 0 \\ 0 & \ldots & 0 \end{matrix} \\ \begin{matrix} 0 & 0 & 0 \\ \vdots & \vdots & \vdots \\ 0 & 0 & 0 \end{matrix} & \begin{matrix} \infty & \ldots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \ldots & \infty \end{matrix} \end{pmatrix}$$

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

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

- Bearing only SLAM would require multiple sightings

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

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

$j = c_t^i;$

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

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

end

$$\hat{z}_t^i = \begin{pmatrix} \sqrt{(\overline{\mu}_{j,x} - \overline{\mu}_{t,x})^2 + (\overline{\mu}_{j,y} - \overline{\mu}_{t,y})^2} \\ \operatorname{atan2}(\overline{\mu}_{j,y} - \overline{\mu}_{t,y}, \overline{\mu}_{j,x} - \overline{\mu}_{t,x}) - \overline{\mu}_{t,\theta} \end{pmatrix};$$

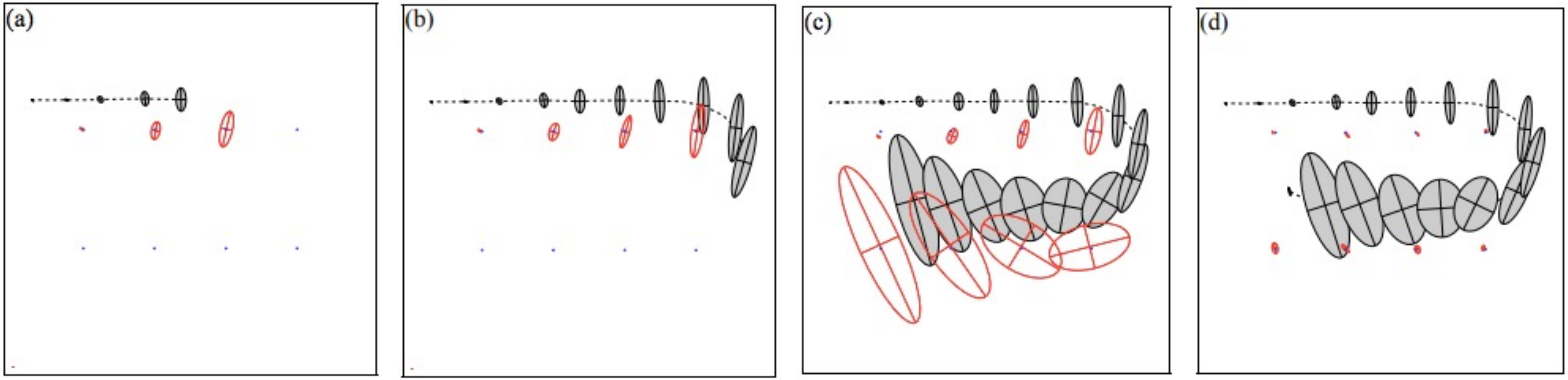
$H_t^i = h_t^i \, F_{x,j};$

$S_t^i = H_t^i \, \overline{\Sigma}_t \, [H_t^i]^T + Q_t;$
$K_t^i = \overline{\Sigma}_t \, [H_t^i]^T \, [S_t^i]^{-1};$
$\overline{\mu}_t = \overline{\mu}_t + K_t^i (z_t^i - \hat{z}_t^i);$
$\overline{\Sigma}_t = (I - K_t^i H_t^i) \, \overline{\Sigma}_t;$

end

$\mu_t = \overline{\mu}_t$ and $\Sigma_t = \overline{\Sigma}_t;$
Return (μ_t, Σ_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 α

Data: $(\mu_{t-1}, \Sigma_{t-1}), u_t, z_t, N_{t-1}$
Result: (μ_t, Σ_t)
$N_t = N_{t-1};$
$\bar{\mu}_t = g(u_t, \mu_{t-1});$
$\bar{\Sigma}_t = G_t \Sigma_{t-1} G_t^T + R_t;$

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

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

Hypothesis for new landmark

for $k = 1$ *to* $N_t + 1$ **do**

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

$H_t^k = h_t^k \, F_{x,k};$
$S_t^k = H_t^k \, \bar{\Sigma}_t \, [H_t^k]^T + Q_t;$
$\pi_k = (z_t^i - \hat{z}_t^k)^T \, [S_t^k]^{-1} \, (z_t^i - \hat{z}_t^k);$

Mahalanobis distance

end

$\pi_{N_t+1} = \alpha;$
$j(i) = \operatorname{argmin}_k \; \pi_k;$ 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

$\mu_t = \bar{\mu}_t$ and $\Sigma_t = \bar{\Sigma}_t;$
Return (μ_t, Σ_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

Particle filter SLAM

- Key idea: use particles to approximate the belief, and particle filter to simultaneously estimate the robot path and the map
- Goal is to solve full-scale SLAM, i.e., estimate

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

- Challenge: naïve implementation of particle filter to SLAM is intractable, due to the excessively large number of particles required
- Key insight: knowledge of the robot's true path renders features conditionally independent -> mapping problem can be *factored* into separate problems, one for each feature in the map

Factoring the posterior

- The key mathematical insight behind particle filter SLAM is the factorization of the posterior

$$p(y_{1:t} \mid z_{1:t}, u_{1:t}, c_{1:t}) = p(x_{1:t} \mid z_{1:t}, u_{1:t}, c_{1:t}) \prod_{n=1}^{N} p(m_n \mid x_{1:t}, z_{1:t}, c_{1:t})$$

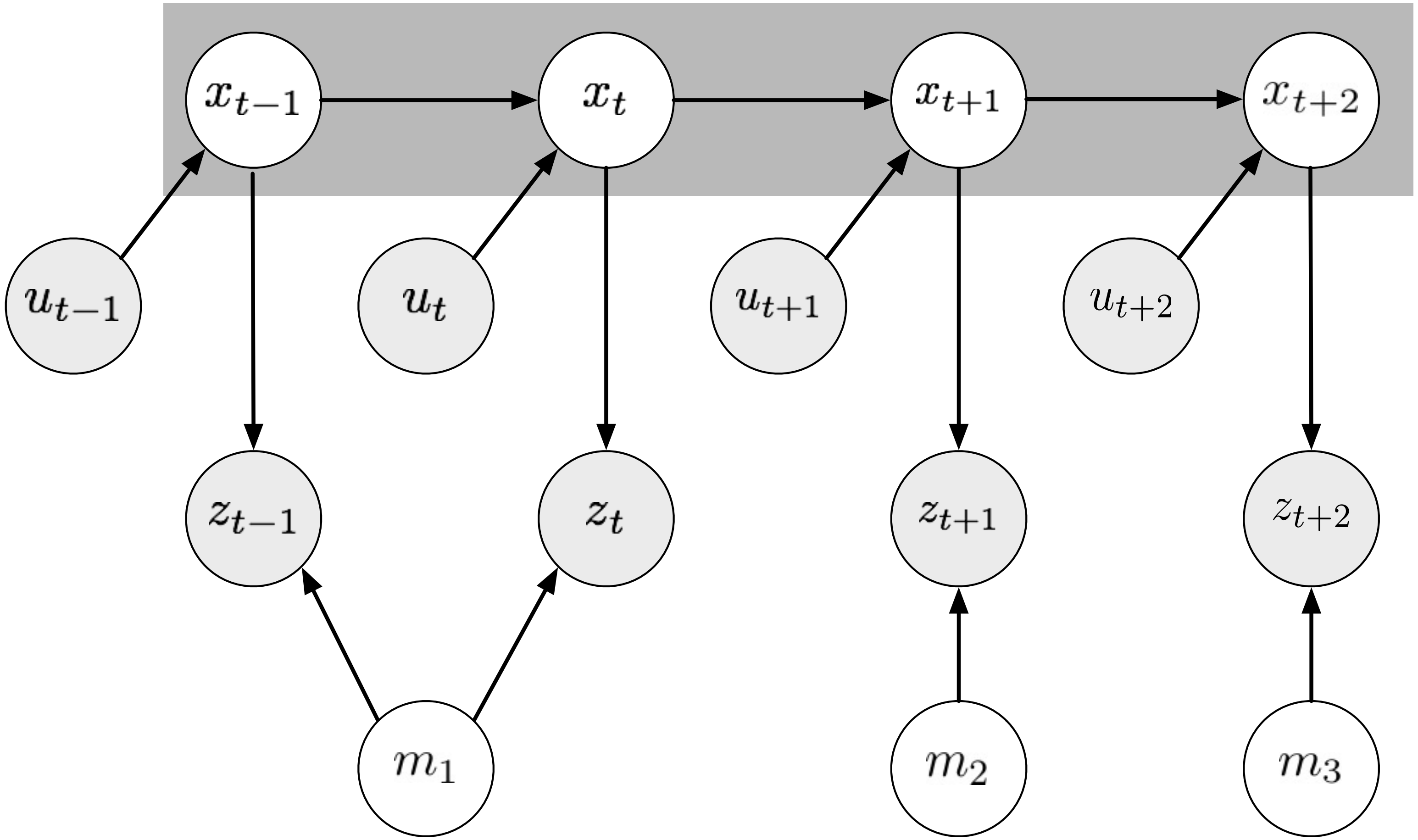
SLAM posterior

Path posterior
(particles)

Feature posterior
(EKF)

Factoring the posterior

- Intuition



Factoring the posterior

- Proof follows from Bayes' rule and induction
- Step #1:

$$
\begin{aligned}
p(y_{1:t} \mid z_{1:t}, u_{1:t}, c_{1:t}) &= p(x_{1:t} \mid z_{1:t}, u_{1:t}, c_{1:t})\, p(m \mid x_{1:t}, z_{1:t}, u_{1:t}, c_{1:t}) \\
&= p(x_{1:t} \mid z_{1:t}, u_{1:t}, c_{1:t})\, p(m \mid x_{1:t}, z_{1:t}, c_{1:t})
\end{aligned}
$$

Factoring the posterior

- Step 2.a: assume $c_t \neq n$

$$p(m_n \,|\, x_{1:t}, z_{1:t}, c_{1:t}) = p(m_n \,|\, x_{1:t-1}, z_{1:t-1}, c_{1:t-1})$$

- Step 2.b: assume $c_t = n$

$$\begin{aligned} p(m_{c_t} \,|\, x_{1:t}, z_{1:t}, c_{1:t}) &= \frac{p(z_t \,|\, m_{c_t}, x_{1:t}, z_{1:t-1}, c_{1:t})\, p(m_{c_t} \,|\, x_{1:t}, z_{1:t-1}, c_{1:t})}{p(z_t \,|\, x_{1:t}, z_{1:t-1}, c_{1:t})} \\ &= \frac{p(z_t \,|\, m_{c_t}, x_t, c_t)\, p(m_{c_t} \,|\, x_{1:t-1}, z_{1:t-1}, c_{1:t-1})}{p(z_t \,|\, x_{1:t}, z_{1:t-1}, c_{1:t})} \end{aligned}$$

Factoring the posterior

- Step 3 (induction): assume at time $t-1$ (induction hypothesis)

$$p(m \mid x_{1:t-1}, z_{1:t-1}, c_{1:t-1}) = \prod_{n=1}^{N} p(m_n \mid x_{1:t-1}, z_{1:t-1}, c_{1:t-1})$$

Factoring the posterior

- Then at time t

$$
\begin{aligned}
p(m \mid x_{1:t}, z_{1:t}, c_{1:t}) &= \frac{p(z_t \mid m, x_{1:t}, z_{1:t-1}, c_{1:t})\, p(m \mid x_{1:t}, z_{1:t-1}, c_{1:t})}{p(z_t \mid x_{1:t}, z_{1:t-1}, c_{1:t})} \\
&= \frac{p(z_t \mid m, x_t, c_t)\, p(m \mid x_{1:t-1}, z_{1:t-1}, c_{1:t-1})}{p(z_t \mid x_{1:t}, z_{1:t-1}, c_{1:t})} \\
&= \frac{p(z_t \mid m, x_t, c_t)}{p(z_t \mid x_{1:t}, z_{1:t-1}, c_{1:t})} \prod_{n=1}^{N} p(m_n \mid x_{1:t-1}, z_{1:t-1}, c_{1:t-1}) \\
&= \frac{p(z_t \mid m, x_t, c_t)}{p(z_t \mid x_{1:t}, z_{1:t-1}, c_{1:t})} \underbrace{p(m_{c_t} \mid x_{1:t-1}, z_{1:t-1}, c_{1:t-1})}_{\text{Steb 2.b}} \prod_{n \neq c_t} \underbrace{p(m_n \mid x_{1:t-1}, z_{1:t-1}, c_{1:t-1})}_{\text{Steb 2.a}} \\
&= p(m_{c_t} \mid x_{1:t}, z_{1:t}, c_{1:t}) \prod_{n=1}^{N} p(m_n \mid x_{1:t}, z_{1:t}, c_{1:t}) = \prod_{n=1}^{N} p(m \mid x_{1:t}, z_{1:t}, c_{1:t})
\end{aligned}
$$

Fast SLAM with known correspondences

- Key idea: exploit factorization result to decompose problem into sub-problems
  - Path posterior is estimated using particle filter
  - Map features are estimated via EKF conditioned on the robot path (one EKF for each feature)

- Accordingly, particles in Fast SLAM are represented as

$$Y_t^{[k]} = \left\langle x_t^{[k]}, \mu_{1,t}^{[k]}, \Sigma_{1,t}^{[k]}, \ldots, \mu_{N,t}^{[k]}, \Sigma_{N,t}^{[k]} \right\rangle$$

Fast SLAM with known correspondences

- Each particle possesses its own set of EKFs!
- In total there are *NM* EKFs
- Filtering involves generating a new particle set Y_t from Y_{t-1} by incorporating a new control u_t and a new measurement z_t with associated correspondence variable c_t
- Update entails three steps
  1. Extend path posterior
  2. Update observed feature estimate
  3. Resample

Step 1: Extending path posterior

- For each particle $Y_t^{[k]}$, sample pose x_t according to motion posterior

$$x_t^k \sim p(x_t \mid x_{t-1}^k, u_t)$$

- Sample $x_t^{[k]}$ is then concatenated with previous poses $x_{1:t-1}^{[k]}$

Step 2: updating observed feature estimate

- This step entails updating the posterior over the feature estimates
- If $c_t \neq n$

$$\left\langle \mu_{n,t}^{[k]}, \Sigma_{n,t}^{[k]} \right\rangle = \left\langle \mu_{n,t-1}^{[k]}, \Sigma_{n,t-1}^{[k]} \right\rangle$$

- If $c_t = n$

$$p(m_{c_t} \mid x_{1:t}, z_{1:t}, c_{1:t}) = \eta\, p(z_t \mid m_{c_t}, x_t, c_t)\, p(m_{c_t} \mid x_{1:t-1}, z_{1:t-1}, c_{1:t-1})$$

$$\sim \mathcal{N}(\mu_{n,t-1}^{[k]}, \Sigma_{n,t-1}^{[k]})$$

Step 2: updating observed feature estimate

- To ensure that the new estimate is Gaussian as well, measurement model is linearized as usual

$$h(m_{c_t}, x_t^{[k]}) \approx h(\mu_{c_t,t-1}^{[k]}, x_t^{[k]}) + \underbrace{h'(\mu_{c_t,t-1}^{[k]}, x_t^{[k]})}_{:=H_t^{[k]}}(m_{c_t} - \mu_{c_t,t-1}^{[k]})$$

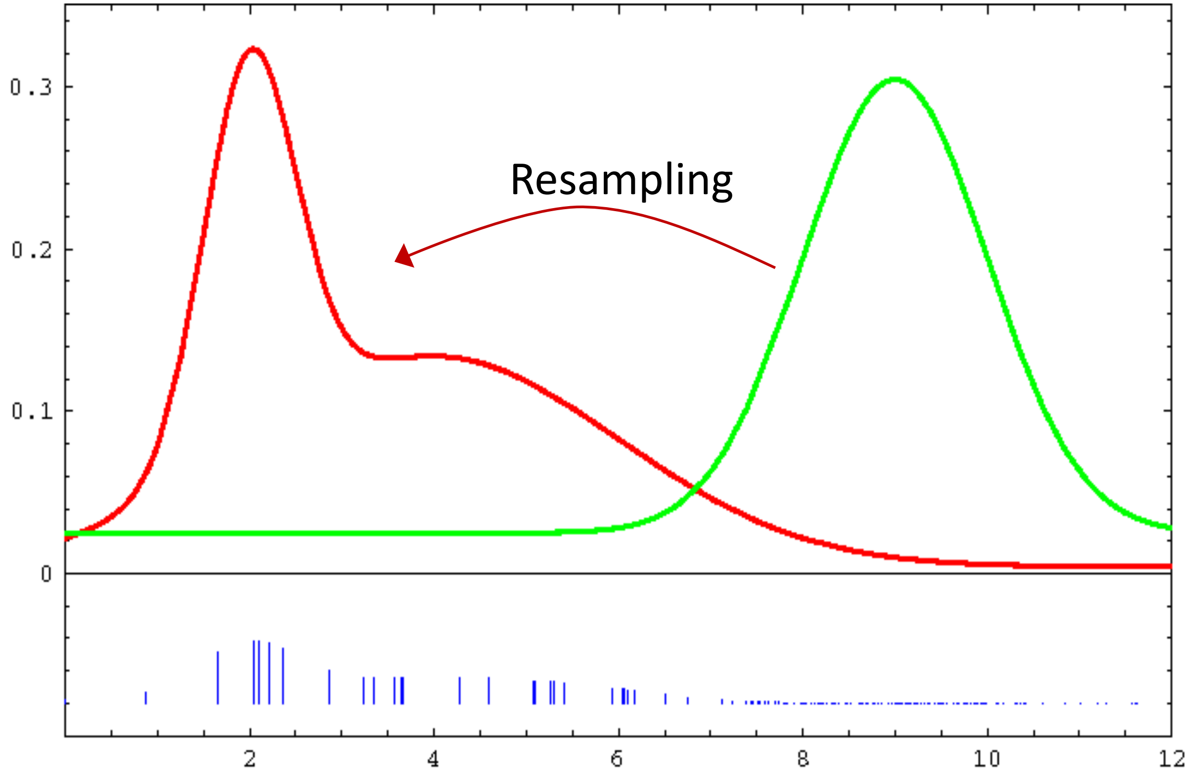
- Mean and covariance are then obtained as per standard EKF

$$K_t^{[k]} = \Sigma_{c_t,t-1}^{[k]}[H_t^{[k]}]^T \, (H_t^{[k]}\Sigma_{c_t,t-1}^{[k]}[H_t^{[k]}]^T + Q_t)^{-1}$$
$$\mu_{c_t,t}^{[k]} = \mu_{c_t,t-1}^{[k]} + K_t^{[k]}(z_t - \hat{z}_t^{[k]})$$
$$\Sigma_{c_t,t}^{[k]} = (I - K_t^{[k]}H_t^{[k]})\Sigma_{c_t,t-1}^{[k]}$$

Step 3: resampling

- Step 1 generates pose x_t only in accordance with the most recent control u_t, paying no attention to the measurement z_t

- Goal: resample particles to correct for this mismatch



Step 3: resampling

- How do we find the weights?
- Path particles at this stage are distributed according to

$$p(x_{1:t}^{[k]} \mid z_{1:t-1}, u_{1:t}, c_{1:t-1}) = \underbrace{p(x_t \mid x_{t-1}^{k}, u_t)}_{\text{Sampling distribution}} \, \underbrace{p(x_{1:t-1}^{[k]} \mid z_{1:t-1}, u_{1:t-1}, c_{1:t-1})}_{\text{Distribution of path particles in } Y_{t-1}^{[k]}}$$

- The target distribution takes into account z_t, along with c_t

$$p(x_{1:t}^{[k]} \mid z_{1:t}, u_{1:t}, c_{1:t})$$

Step 3: resampling

- Importance factor is then given by

$$
\begin{aligned}
w_t^{[k]} &= \frac{p(x_{1:t}^{[k]} \mid z_{1:t}, u_{1:t}, c_{1:t})}{p(x_{1:t}^{[k]} \mid z_{1:t-1}, u_{1:t}, c_{1:t-1})} \\
&= \frac{\eta\, p(z_t \mid x_{1:t}^{[k]}, z_{1:t-1}, u_{1:t}, c_{1:t}) p(x_{1:t}^{[k]} \mid , z_{1:t-1}, u_{1:t}, c_{1:t})}{p(x_{1:t}^{[k]} \mid z_{1:t-1}, u_{1:t}, c_{1:t-1})} \\
&= \frac{\eta\, p(z_t \mid x_t^{[k]}, c_t) p(x_{1:t}^{[k]} \mid , z_{1:t-1}, u_{1:t}, c_{1:t-1})}{p(x_{1:t}^{[k]} \mid z_{1:t-1}, u_{1:t}, c_{1:t-1})} \\
&= \eta\, p(z_t \mid x_t^{[k]}, c_t)
\end{aligned}
$$

Step 3: resampling

- To derive an (approximate) close-form expression for $w_t^{[k]}$, one can then apply the total probability law along with a linearization of the measurement model to obtain

$$w_t^{[k]} = \eta \det(2\pi Q_t^{[k]})^{-\frac{1}{2}} \exp\left\{-\frac{1}{2}(z_t - \hat{z}_t^{[k]})[Q_t^{[k]}]^{-1}(z_t - \hat{z}_t^{[k]})\right\}$$

$$Q_t^{[k]} = [H_t^{[k]}]^T \Sigma_{n,t-1}^{[k]} H_t^{[k]} + Q_t$$

Fast Slam algorithm

- Key fact: only the most recent pose is used in the process of generating a new particle at time *t*!

Data: Y_{t-1}, u_t, z_t, c_t
Result: Y_t
for $k = 1$ **to** M **do**
  $x_t^k \sim p(x_t \mid x_{t-1}^k, u_t)$;
  $j = c_t$;
  if *feature j never seen before* **then**
    initialize feature
  else
    $\hat{z} = h(\mu_{j,t-1}^{[k]}, x_t^{[k]})$;
    calculate Jacobian H;
    $Q = H\Sigma_{j,t-1}^{[k]} H_t^T + Q_t$;
    $K = \Sigma_{j,t-1}^{[k]} H^T Q^{-1}$;
    $\mu_{j,t}^{[k]} = \mu_{j,t-1}^{[k]} + K(z_t - \hat{z})$;
    $\Sigma_{j,t}^{[k]} = (I - KH)\Sigma_{j,t-1}^{[k]}$;
    $w^{[k]} = \det(2\pi Q)^{-\frac{1}{2}} \exp\left\{-\frac{1}{2}(z_t - \hat{z})Q_t^{-1}(z_t - \hat{z})\right\}$;
  end
  for *all other features $n \neq j$* **do**
    $\left\langle \mu_{n,t}^{[k]}, \Sigma_{n,t}^{[k]} \right\rangle = \left\langle \mu_{n,t-1}^{[k]}, \Sigma_{n,t-1}^{[k]} \right\rangle$;
  end
  $Y_t = \emptyset$;
end
for $i = 1$ **to** M **do**
  Draw k with probability $\propto w^{[k]}$;
  Add $\left\langle x_t^{[k]}, \mu_{1,t}^{[k]}, \Sigma_{1,t}^{[k]}, \ldots, \mu_{N,t}^{[k]}, \Sigma_{N,t}^{[k]} \right\rangle$ to Y_t;
end
Return Y_t

Fast SLAM with unknown correspondences

- Key advantage of particle filters: each particle can rely on its own, local data association decisions!
- Key idea: per-particle data association generalizes the per-filter data association to individual particles
- Each particle maintains *a local set* of data association variables, $\hat{c}_t^{[k]}$
- Data association is solved, as usual, via maximum likelihood estimation

$$\hat{c}_t^{[k]} = \arg\max_{c_t} p(z_t \mid c_t, \hat{c}_{1:t-1}^{[k]}, x_{1:t}^{[k]}, z_{1:t-1}, u_{1:t})$$

Computed, as usual, via total probability law + linearization

Summary: Gaussian filtering (EKF, UKF)

- 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

Summary: particle filter approaches

- Key ideas:
  - Approximate belief with particles
  - Use particle filters to perform inference
- Pros:
  - Can handle "any" noise distribution
  - Relatively easy to implement
  - Naturally represents multimodal beliefs
  - Robust to data association errors
- Cons:
  - Does not scale well to large dimensional problems
  - Might require many particles for good convergence
  - Might have issues with loop closure

Final considerations

- A 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.
- Trends: from the classical age, to the algorithmic-analysis age, to the robust perception age
- 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

Next time