Robotic Navigation and Exploration

Unit 05: SLAM Back-end (I)

Min-Chun Hu anitahu@cs.nthu.edu.tw
CS, NTHU

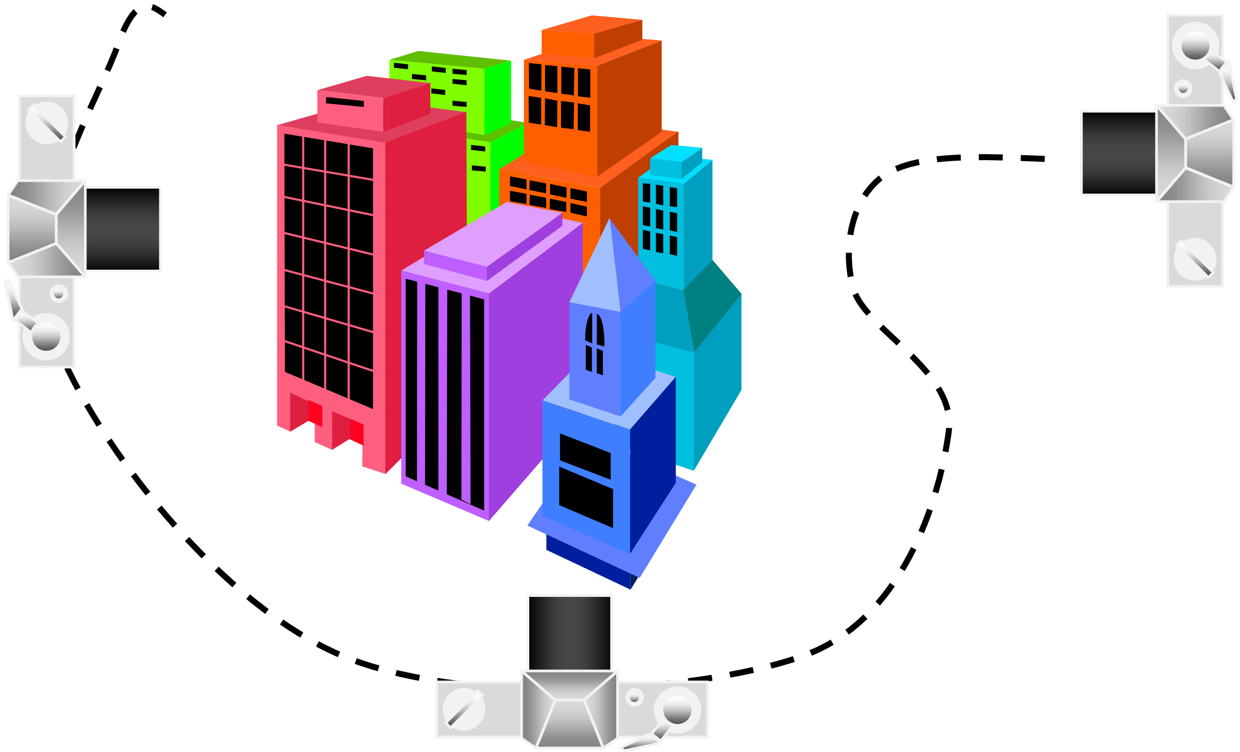
Outline

- State Estimation and SLAM Problem
- SLAM Back-end (Error Compensation)
  - Filter-based Methods
    - Probability Theory and Bayes Filter
    - Kalman Filter (KF) / Extended Kalman Filter (EKF)
      - EKF-SLAM
    - Particle Filter
      - Fast-SLAM
  - Graph-based Methods
    - Pose Graph and Least-square Optimization
    - Gauss-Newton and Levenberg-Marquardt Algorithm
    - Sparse Matrix for Optimization

Outline

- State Estimation and SLAM Problem
- SLAM Back-end (Error Compensation)
  - Filter-based Methods
    - Probability Theory and Bayes Filter
    - Kalman Filter (KF) / Extended Kalman Filter (EKF)
      - EKF-SLAM
    - Particle Filter
      - Fast-SLAM
  - Graph-based Methods
    - Pose Graph and Least-square Optimization
    - Gauss-Newton and Levenberg-Marquardt Algorithm
    - Sparse Matrix for Optimization

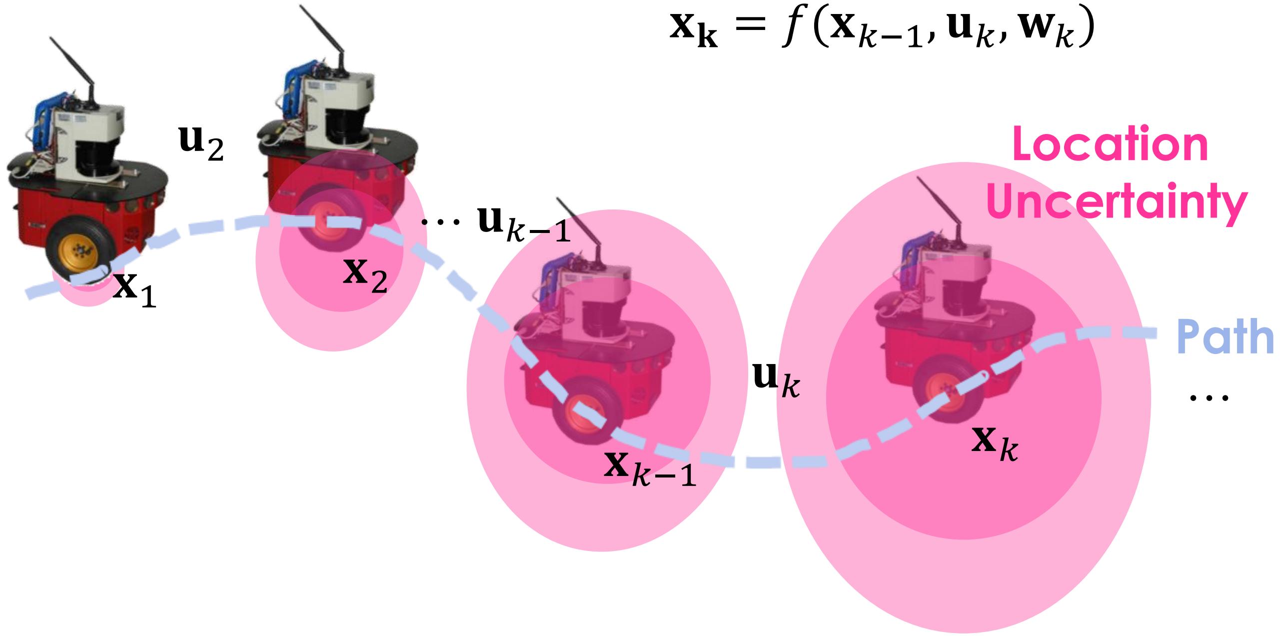
SLAM Problem



Simultaneous **L**ocalization **A**nd **M**apping

SLAM Problem

$$\mathbf{x_k} = f(\mathbf{x}_{k-1}, \mathbf{u}_k, \mathbf{w}_k)$$

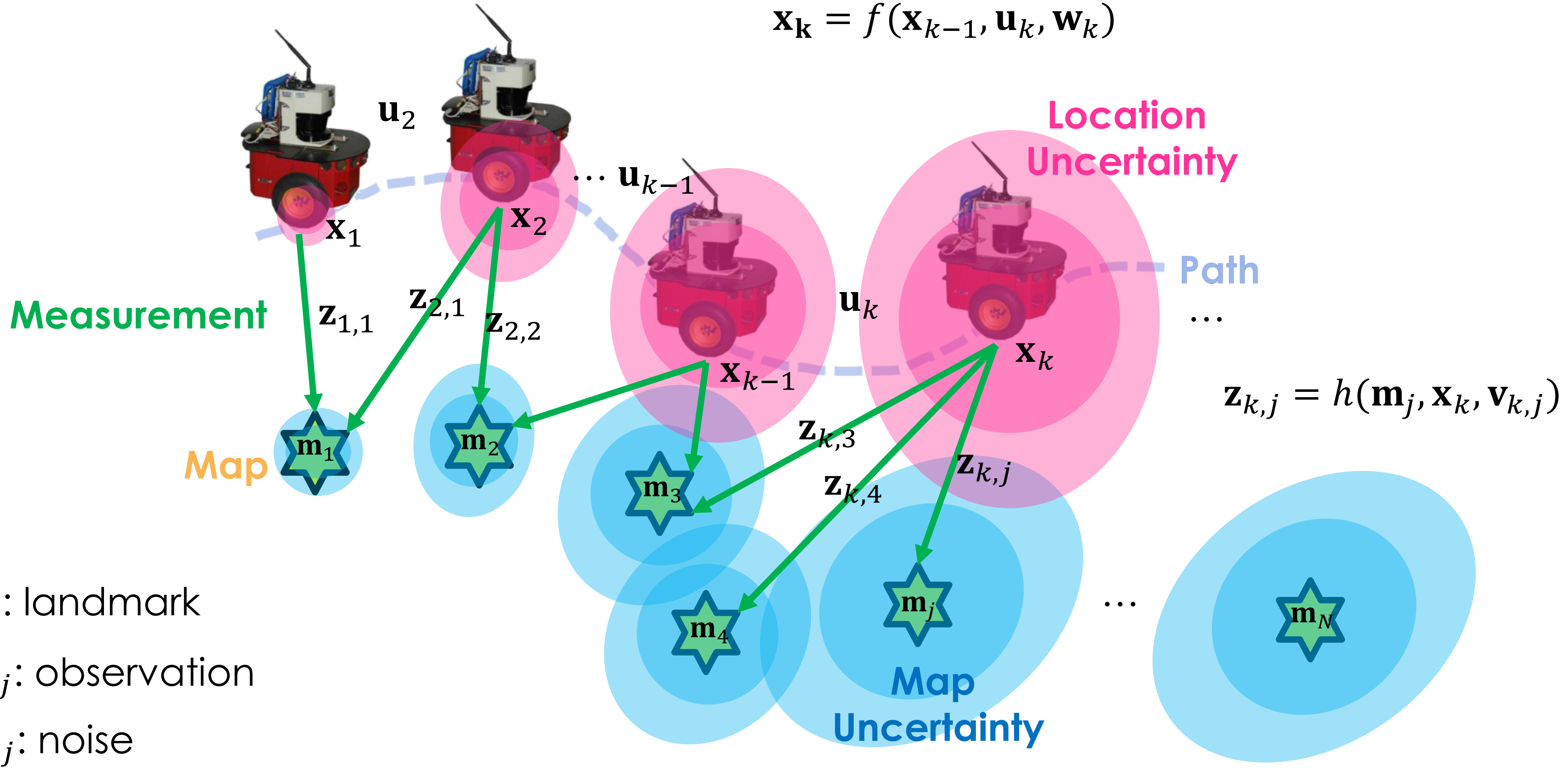


$\mathbf{x}_k$: pose

$\mathbf{u}_k$: control

$\mathbf{w}_k$: noise

SLAM Problem

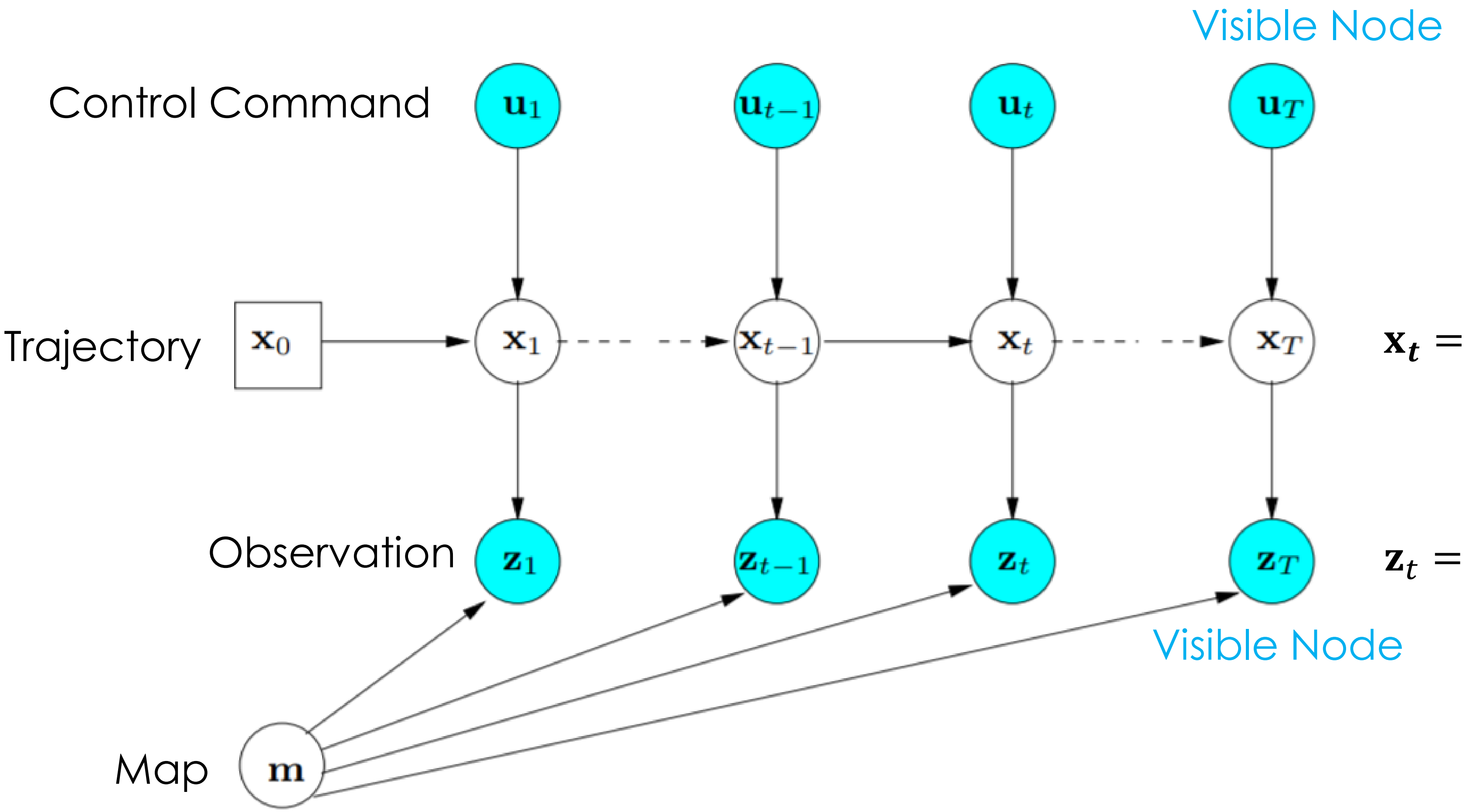


$\boldsymbol{m}_j$: landmark

$\boldsymbol{z}_{k,j}$: observation

$\boldsymbol{v}_{k,j}$: noise

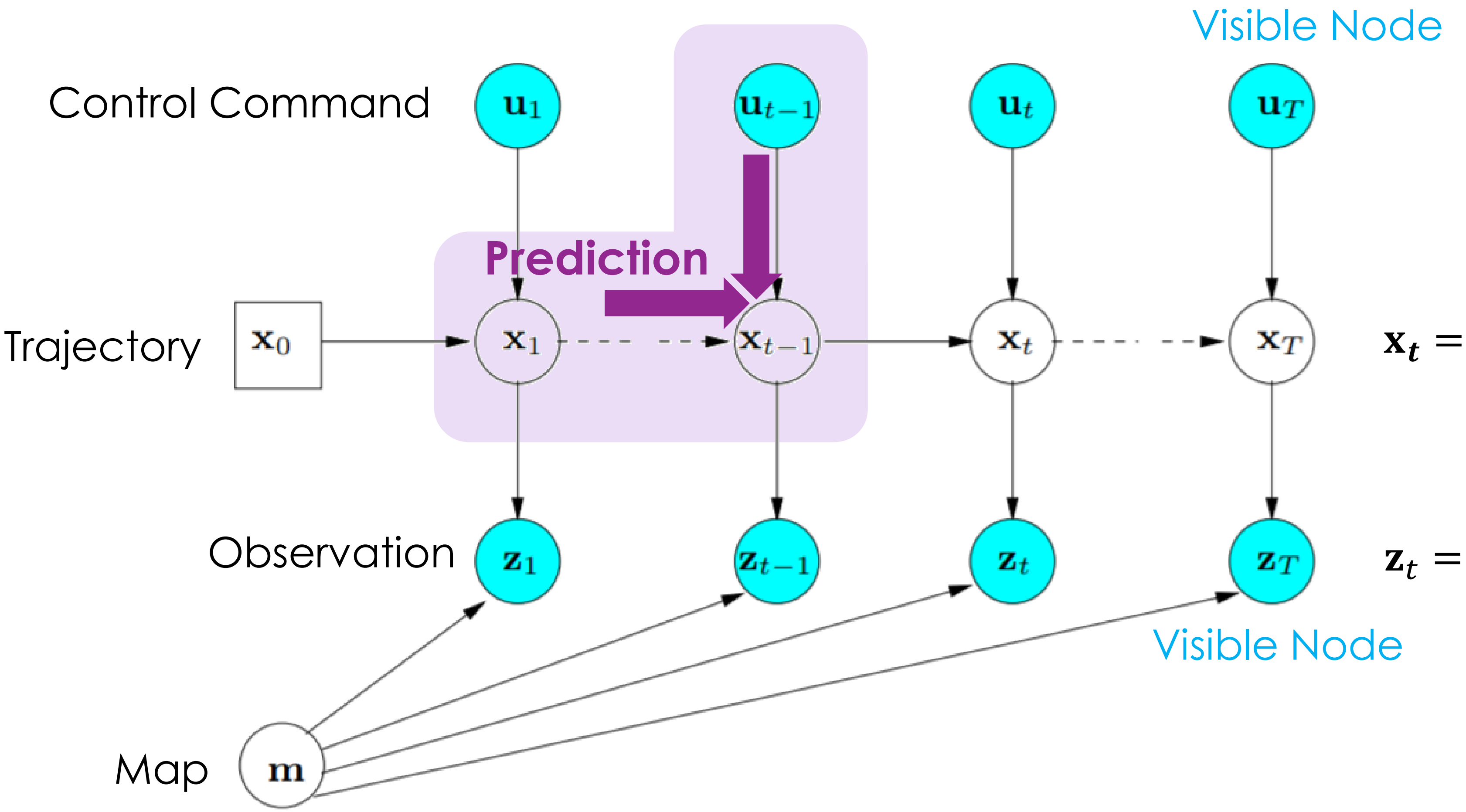
Probability Graphical Model for SLAM Problem



$$\mathbf{x}_t = f(\mathbf{x}_{t-1}, \mathbf{u}_t, \boldsymbol{w}_t)$$

$$\mathbf{z}_t = h(\mathbf{m}, \mathbf{x}_t, \boldsymbol{v}_t)$$

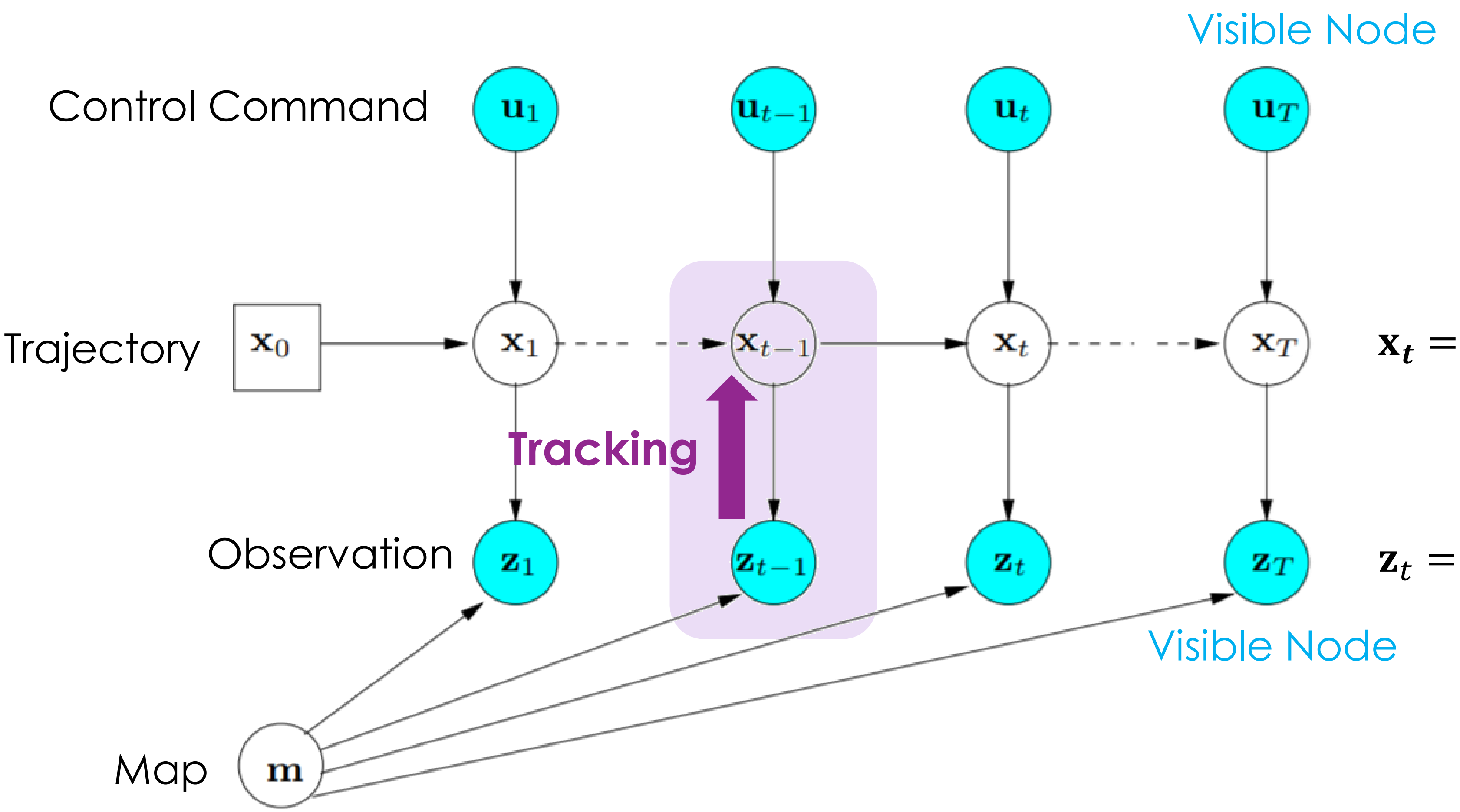
Probability Graphical Model for SLAM Problem



$$\mathbf{x}_t = f(\mathbf{x}_{t-1}, \mathbf{u}_t, \boldsymbol{w}_t)$$

$$\mathbf{z}_t = h(\mathbf{m}, \mathbf{x}_t, \boldsymbol{v}_t)$$

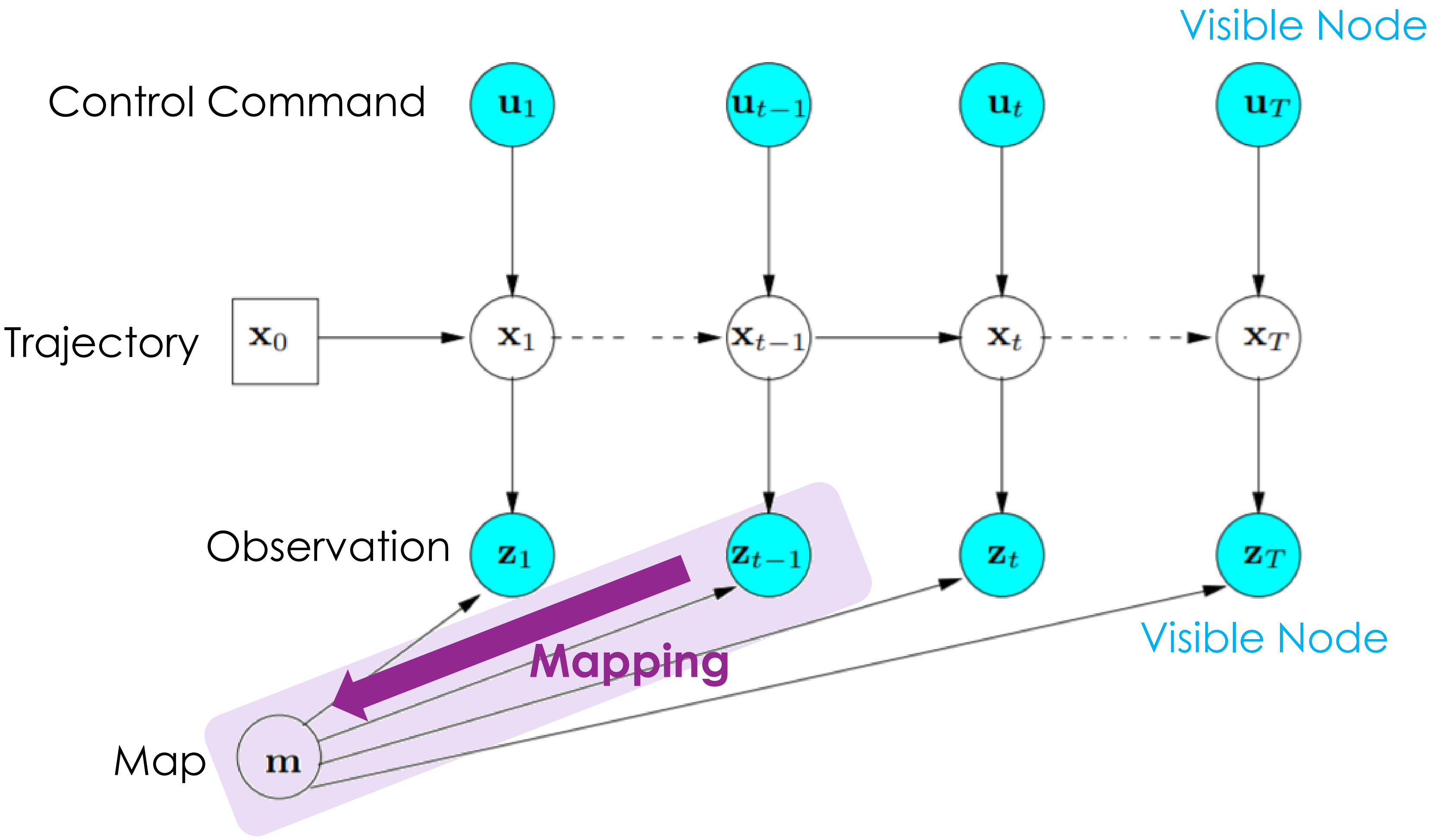
Probability Graphical Model for SLAM Problem



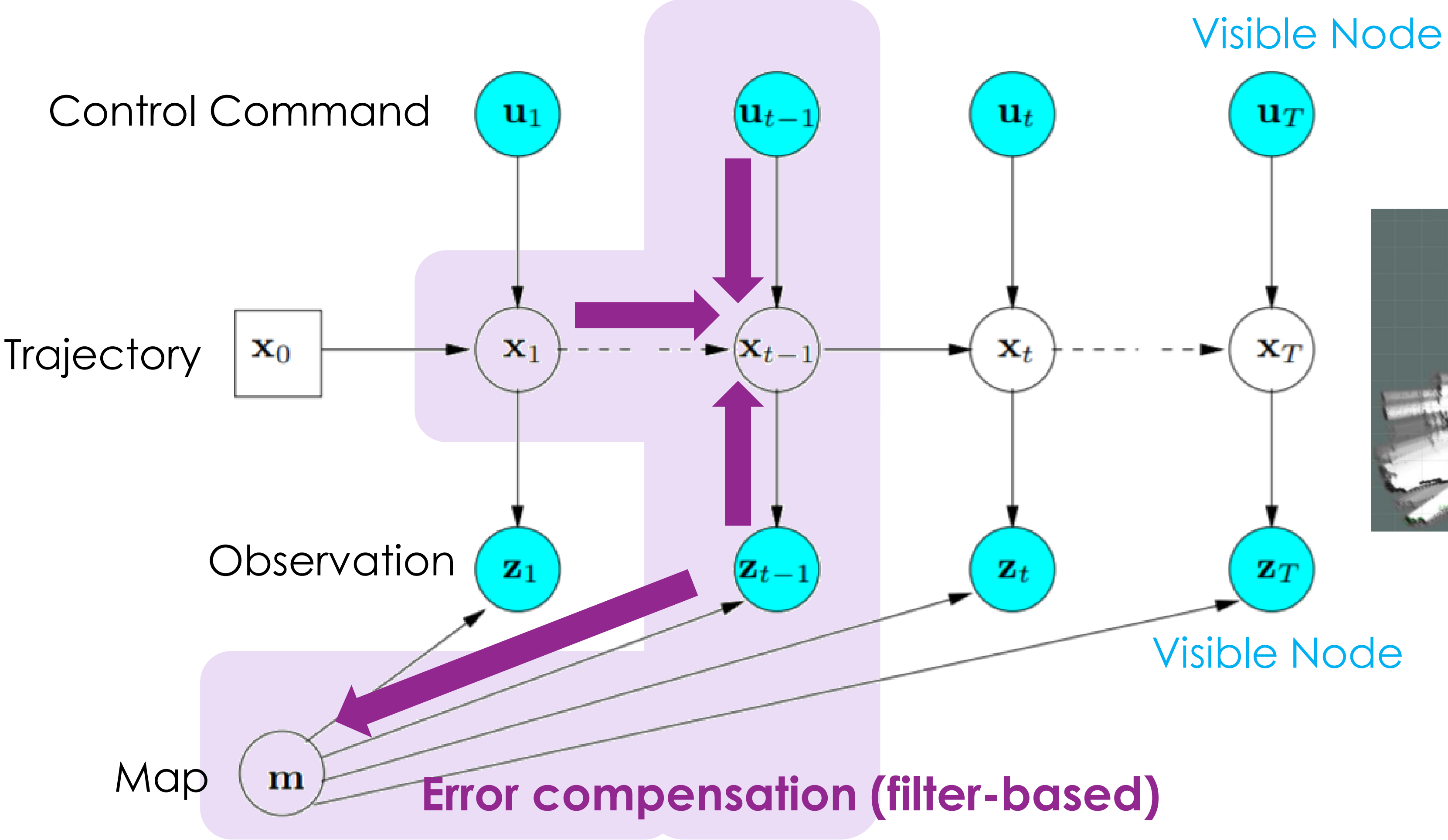
$$\mathbf{x}_t = f(\mathbf{x}_{t-1}, \mathbf{u}_t, \boldsymbol{w}_t)$$

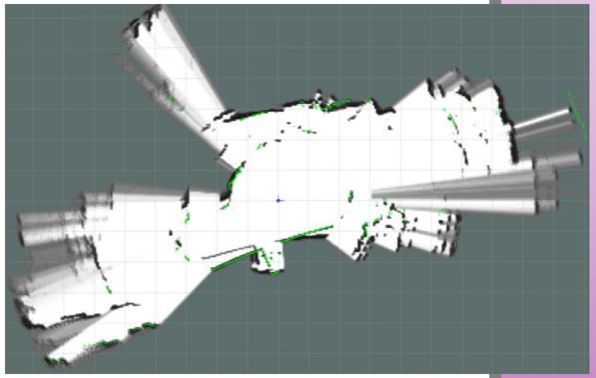
$$\mathbf{z}_t = h(\mathbf{m}, \mathbf{x}_t, \boldsymbol{v}_t)$$

Probability Graphical Model for SLAM Problem

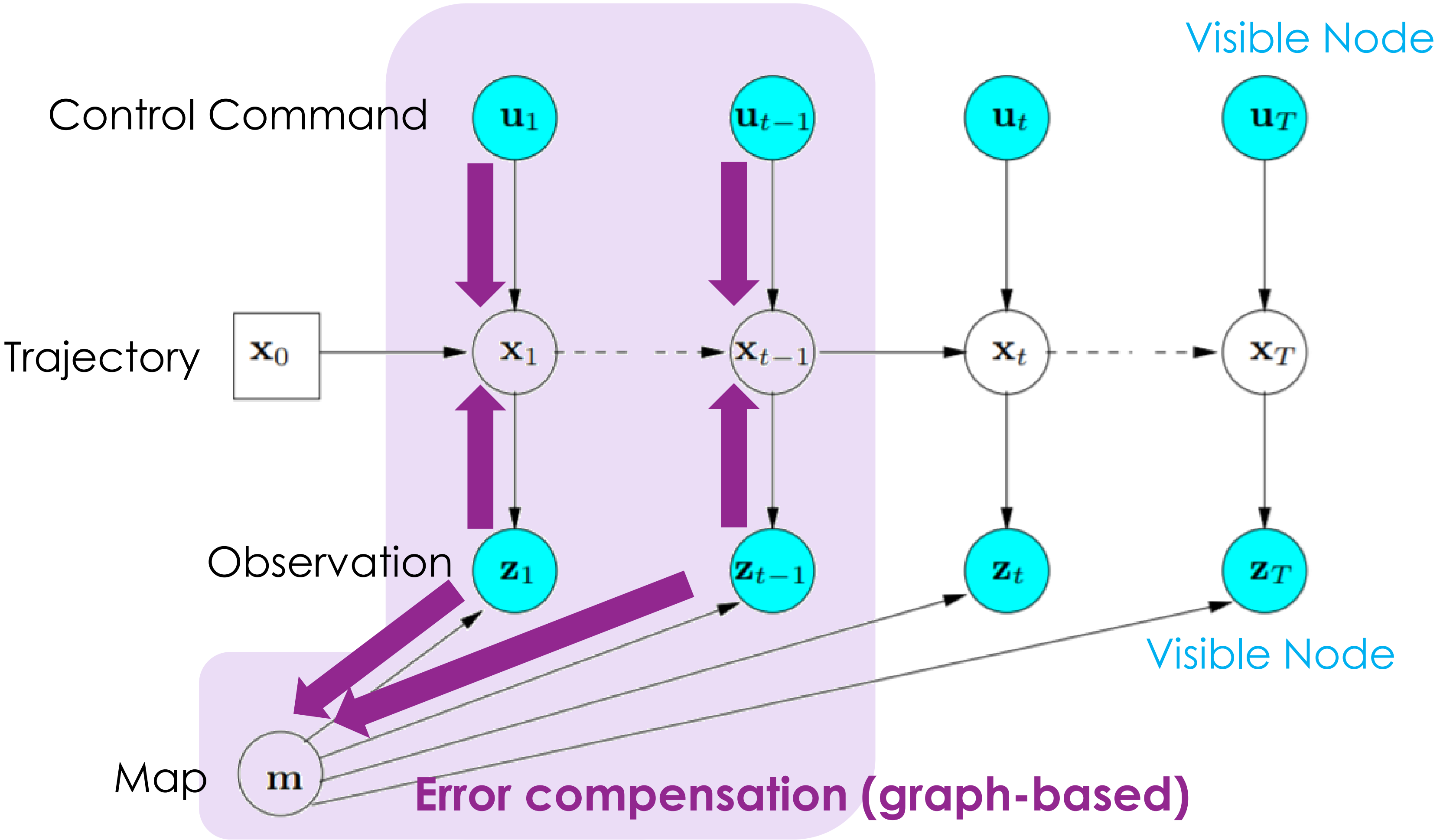


Probability Graphical Model for SLAM Problem





Probability Graphical Model for SLAM Problem

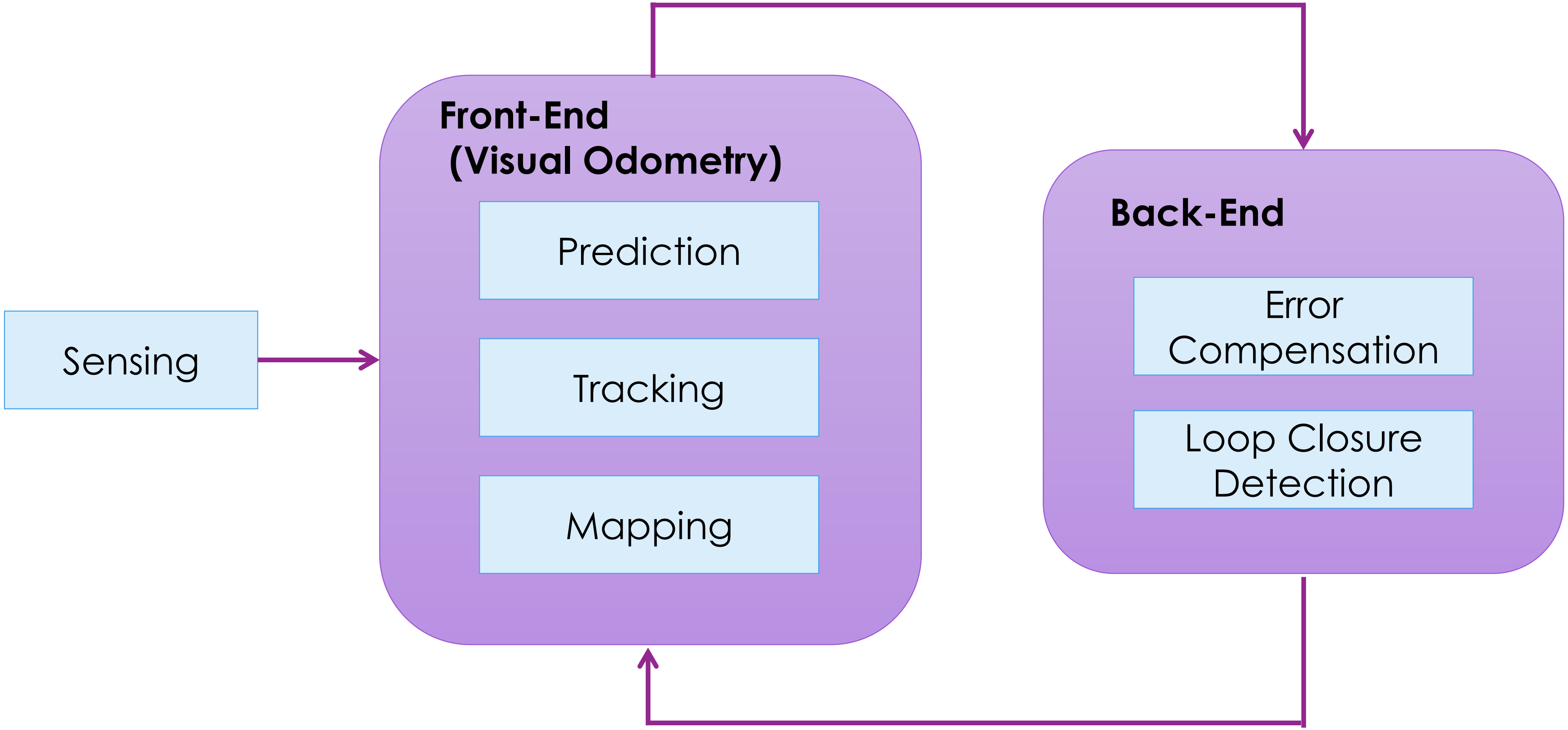


Error Compensation Methods

- Filter-based
  - Less computation
  - On-line optimization
  - Less accurate

- Graph-based
  - Heavier computation
  - Off-line optimization
  - More accurate

SLAM Architecture

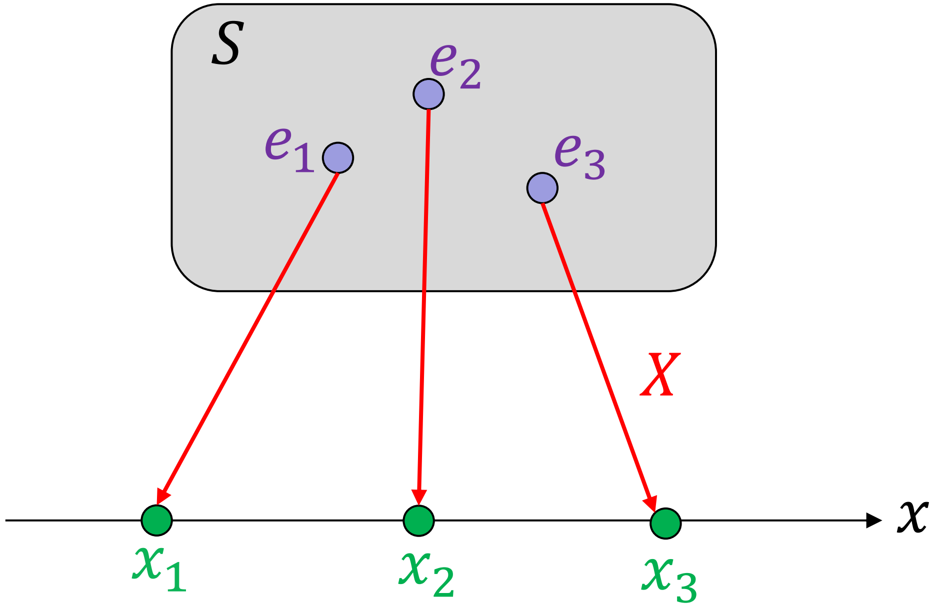


Outline

- State Estimation and SLAM Problem
- SLAM Back-end (Error Compensation)
  - Filter-based Methods
    - Probability Theory and Bayes Filter
    - Kalman Filter (KF) / Extended Kalman Filter (EKF)
      - EKF-SLAM
    - Particle Filter
      - Fast-SLAM
  - Graph-based Methods
    - Pose Graph and Least-square Optimization
    - Gauss-Newton and Levenberg-Marquardt Algorithm
    - Sparse Matrix for Optimization

Random Variable

- A random variable is defined as a function that maps the observation results of unpredictable processes to numerical quantities
- Definition:
  - X:Random Variable
  - S:Sample Space
  - e:event ($e \in S$)
  - $X(e)=x$ ($x \in R$)

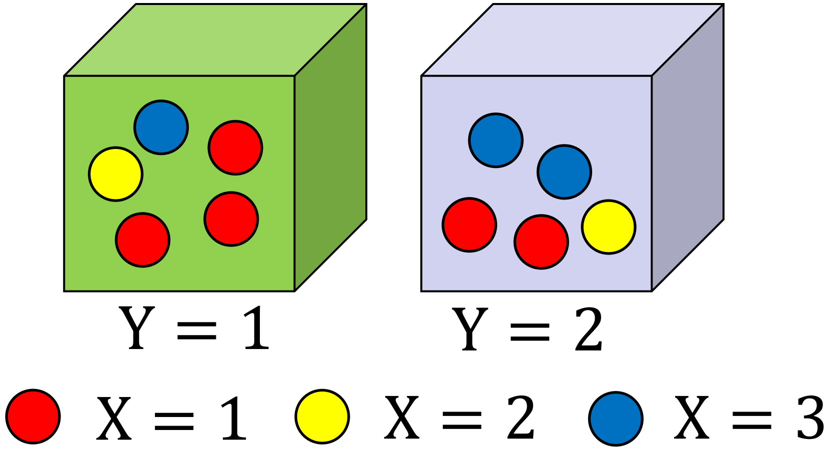


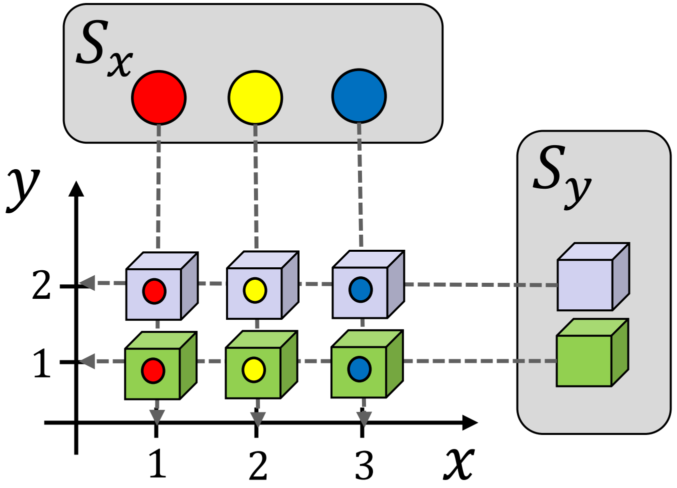
Example of Random Variable

Two Random Variable: X, Y

X: The id of the ball
$X = 1$, if choose the **red** ball
$X = 2$, if choose the **yellow** ball
$X = 3$, if choose the **blue** ball

Y: The id of the box
$Y = 1$, if choose the **green** box
$Y = 2$, if choose the **purple** box





Different Types of Probability

- Joint Probability

$$P(X, Y)$$

- Condition Probability

$$P(X|Y), P(Y|X)$$

- Marginal Probability

$$P(X), P(Y)$$

Sum / Product Rule

- Sum Rule

$$\boxed{P(X = x_i)} = \sum_{Y} P(X, Y)$$

Marginal Probability

- Product Rule

$$\boxed{P(X = x_i, Y = y_j)} = \underline{P(X|Y)P(Y) = P(Y|X)P(X)}$$

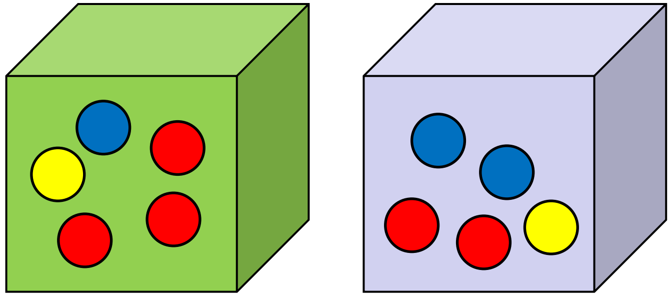
Joint Probability

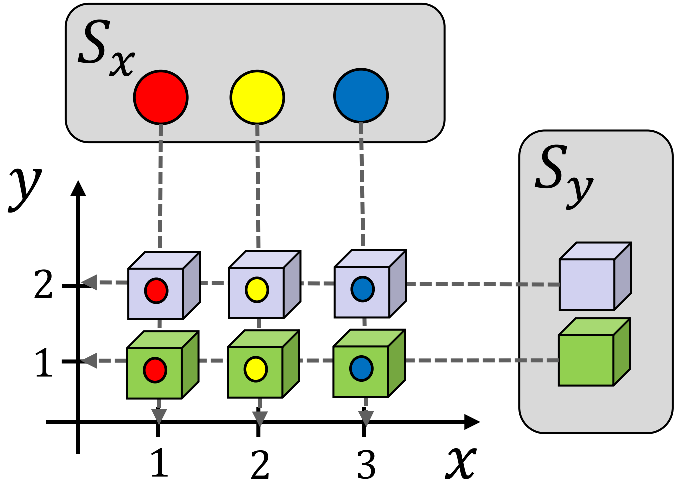
- Bayes Theorem

$$P(Y|X) = \frac{P(X|Y)P(Y)}{P(X)}$$

Sum / Product Rule Example

- Joint Probability and Marginal Probability





	X=1	X=2	X=3
P(X)	5/10	2/10	3/10

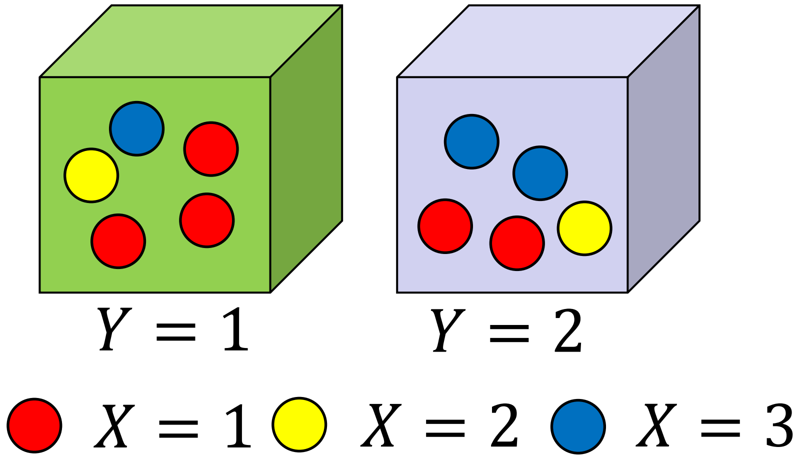
P(X,Y)	X=1	X=2	X=3
Y=1	3/10	1/10	1/10
Y=2	2/10	1/10	2/10

	P(Y)
Y=1	1/2
Y=2	1/2

Independent

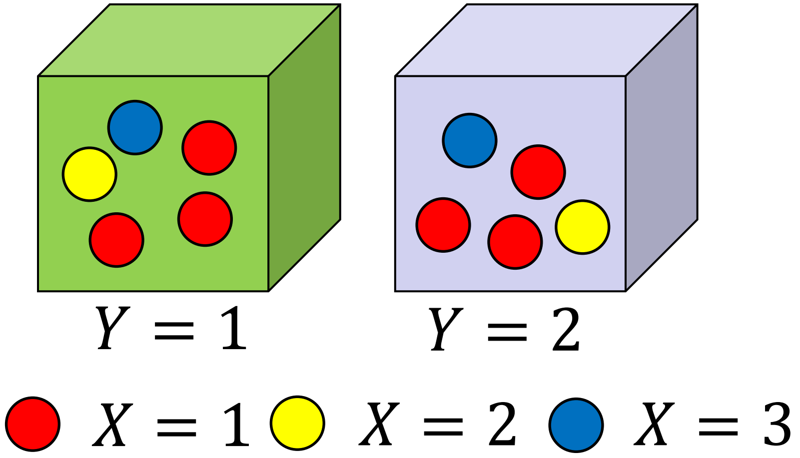
- Independent Event

$$P(Y=1, X=2) = P(Y=1)P(X=2)$$
$$\frac{1}{10} = \frac{1}{2} \times \frac{2}{10}$$
$$P(Y=1) = P(Y=1|X=2)$$
$$\frac{1}{2} = \frac{1/10}{1/10 + 1/10}$$



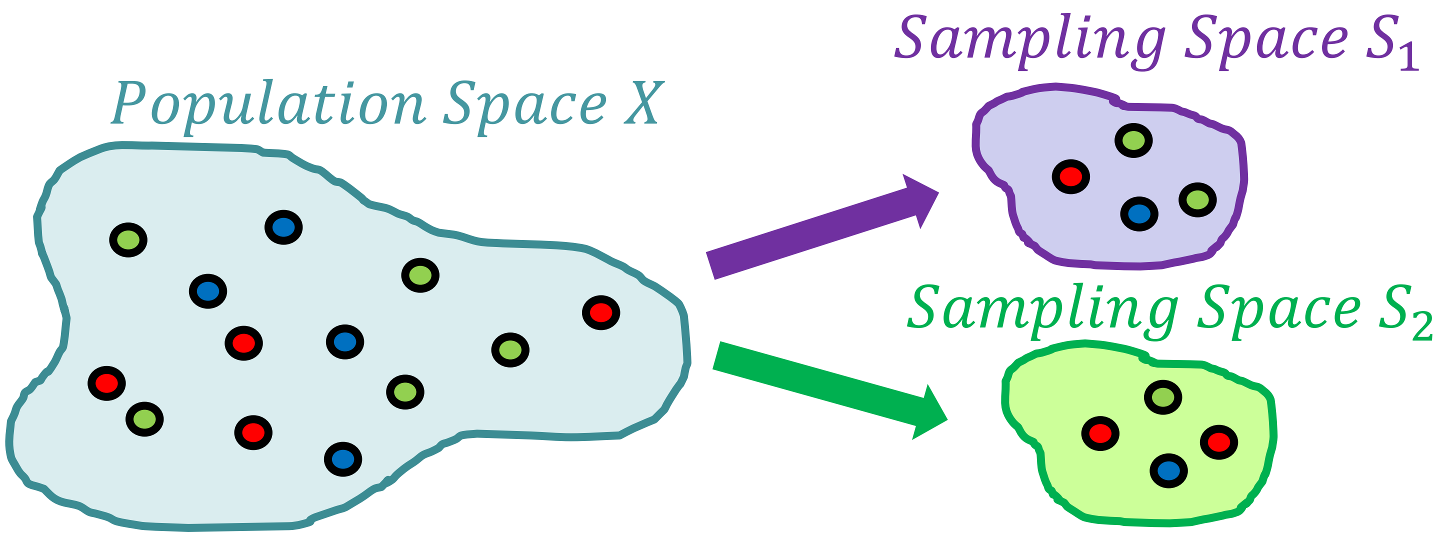
- Independent Random Variable

$$P(Y, X) = P(Y)P(X)$$
$$P(Y|X) = P(Y)$$



Independent and Identically Distributed (i.i.d.)

- We hope that the sampling process is Independent and Identically Distributed (i.i.d)
  - → The probability of each sampling data is independent and came from same probability distribution



$$p(x_i \in S_1 \cap x_i \in S_2) = p(x_i \in S_1)p(x_i \in S_2)$$

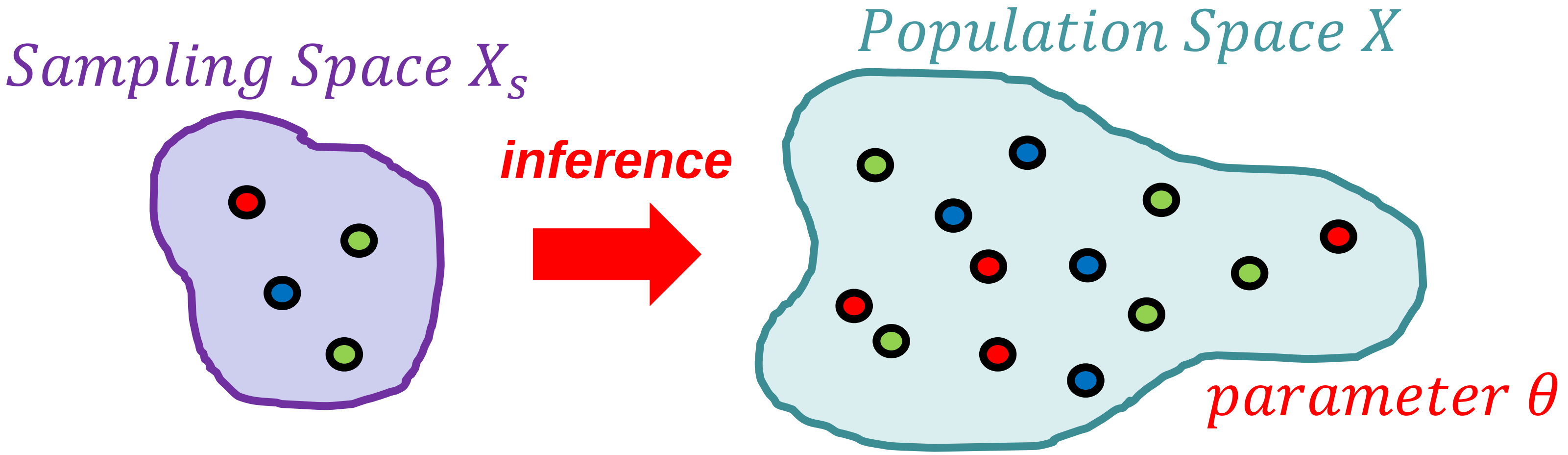
Inference

- Inference:
  - A process to find the logical consequences from premises
  - In machine learning, we want to inference the probability of an event for a given condition $\boldsymbol{p}(\boldsymbol{Event} \mid \boldsymbol{Condition})$

- Example: Supervised Model
  - $\boldsymbol{x}$ is input, $\boldsymbol{y}$ is output, $\boldsymbol{\theta}$ is the parameter of the model
  - Learning and Predicting are both inference tasks
  - Learning Tasks: $p(\theta \mid x, y)$
  - Predicting Tasks: $p_\theta(y \mid x)$ or $p(y \mid \theta, x)$

Statistical Inference

- A process to inference the parameters of population based on the information of sampling data
- $\boldsymbol{x}_s$ is sampled data, $\boldsymbol{\theta}$ is the parameter of distribution over population, statistical inference is to inference $\boldsymbol{p}(\boldsymbol{\theta} \mid \boldsymbol{x}_s)$

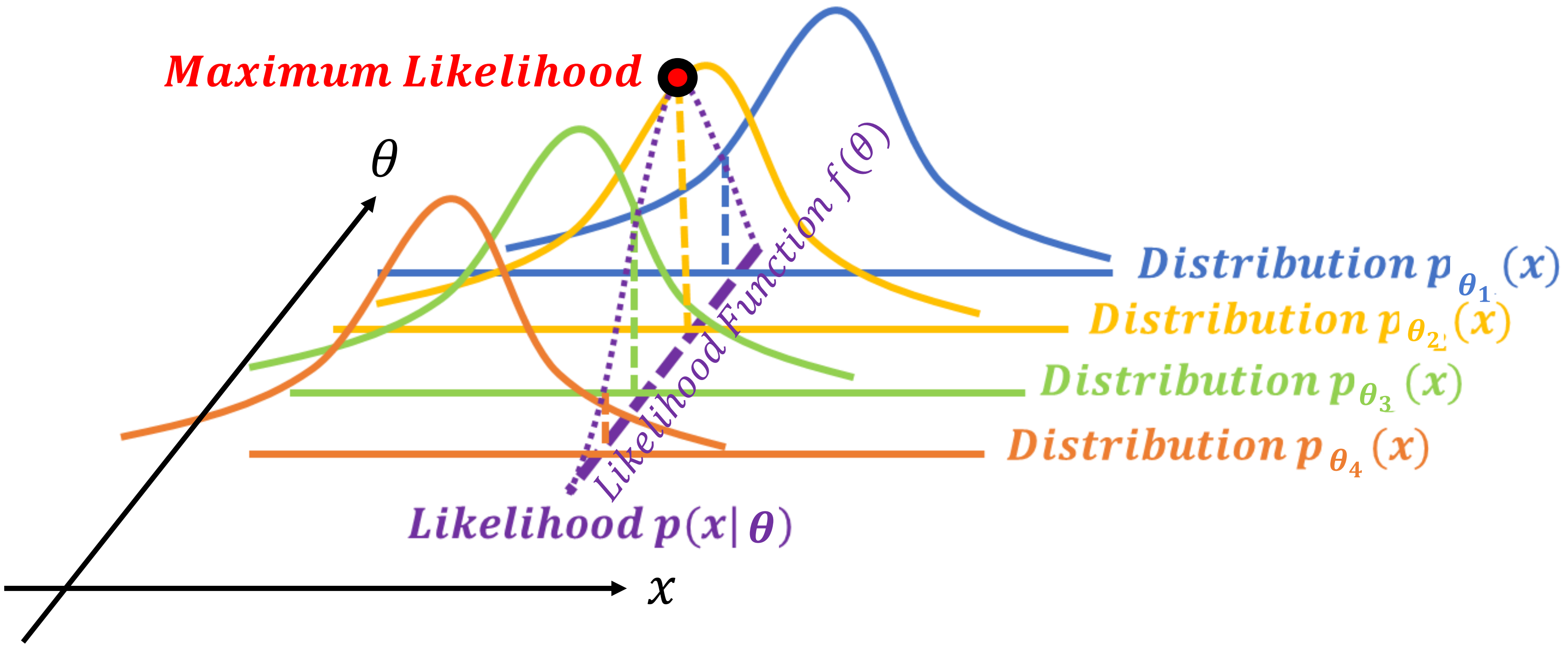


Statistical Inference

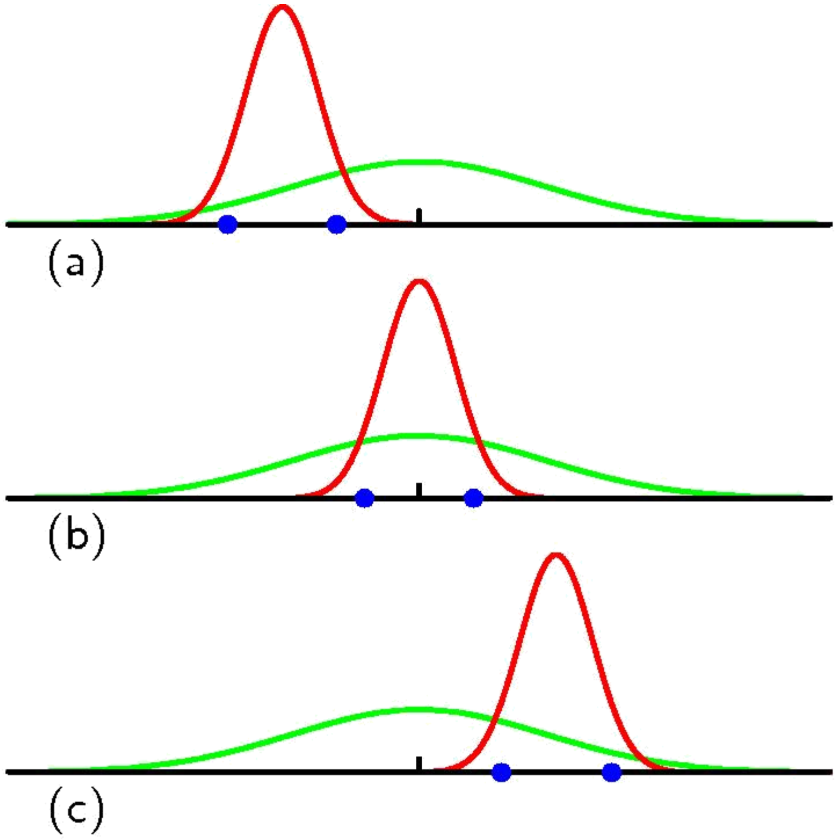
- Two approaches of statistical inference
  - Hypothesis Testing (Top-Down)
    - Given a hypothesis of parameters, evaluate the correctness from sampling data
  - Estimation (Bottom-Up)
    - Find the most likely parameters from sampling data

Maximum Likelihood Estimation (MLE)

- Visualization of likelihood function



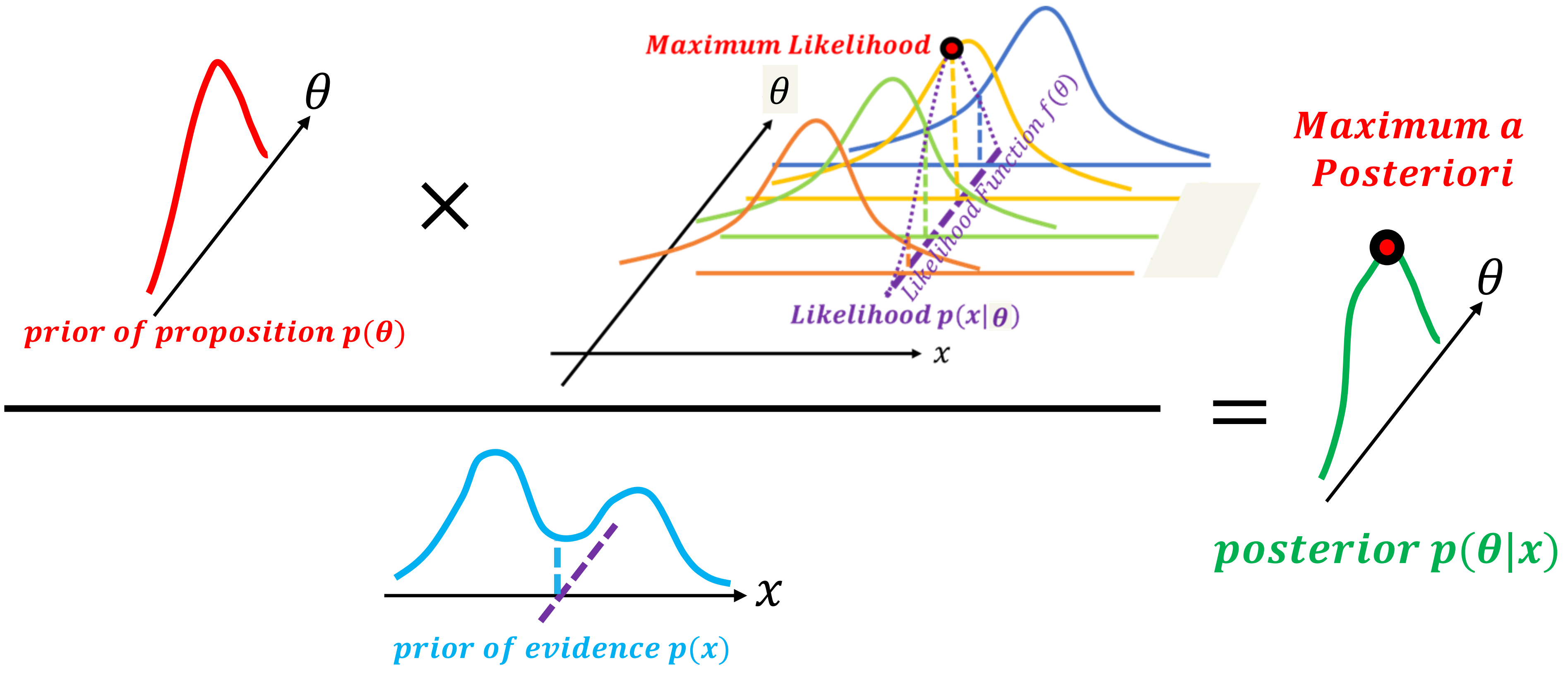
Problem of MLE



Maximum a Posteriori Estimation (MAP)

- Visualization of Posterior Probability

$$\frac{\boldsymbol{p}(\boldsymbol{\theta})\boldsymbol{p}(\boldsymbol{x}|\boldsymbol{\theta})}{\boldsymbol{p}(\boldsymbol{x})} = \boldsymbol{p}(\boldsymbol{\theta}|\boldsymbol{x})$$



Example: Coin Estimation

- Toss a coin
  - [tail, tail, tail, head, tail]

- Likelihood $\boldsymbol{P(x \mid \theta)}$:
  - ***Bernoulli distribution***: $\theta^n (1-\theta)^{m-n}$
  - ***MLE Estimation***:

$$\rightarrow \max_{p} \theta(1-\theta)^4$$

$$\frac{d\theta(1-\theta)^4}{d\theta} = (1-\theta)^4 + 4\theta(1-\theta)^3(-1) = (1-\theta)^3(5\theta-1) = 0$$

$\boldsymbol{\theta}$ **=0.2**

Example: Coin Estimation

- MAP Estimation (Assume Discrete Uniform Prior)

Prior
(Discrete Uniform)

Likelihood
(Bernoulli)
$\theta^n(1-\theta)^{m-n}$

$$\frac{\boldsymbol{p(\theta)p(x|\theta)}}{\boldsymbol{p(x)}} = \boldsymbol{p(\theta|x)}$$

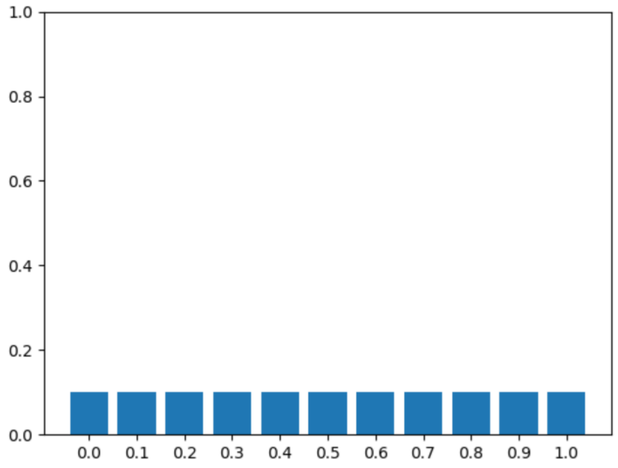
Posterior

$$\frac{\begin{matrix}\theta = 0.0 \\ \theta = 0.1 \\ \theta = 0.2 \\ \vdots\end{matrix}\quad \begin{bmatrix}1/11 \\ 1/11 \\ 1/11 \\ \vdots\end{bmatrix} \times \begin{bmatrix}(0)^1(1)^4 \\ (0.1)^1(0.9)^4 \\ (0.2)^1(0.8)^4 \\ \vdots\end{bmatrix}}{p(x) = \sum_\theta p(x,\theta) = \sum_\theta p(\theta)p(x|\theta)} = \begin{bmatrix}0.000 \\ 0.213 \\ 0.333 \\ \vdots\end{bmatrix}$$

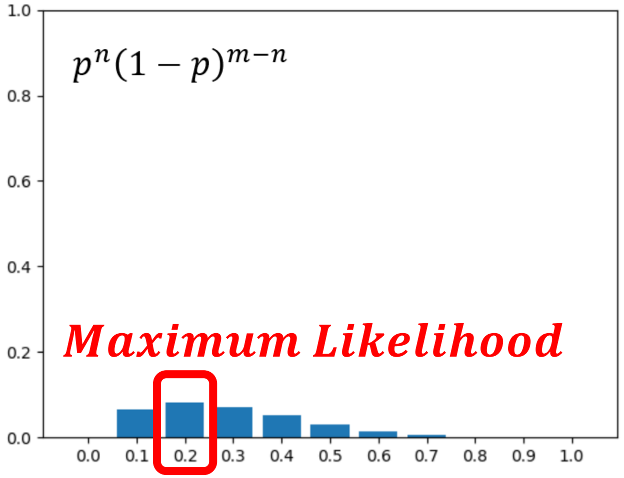
Marginal Probability

Example: Coin Estimation

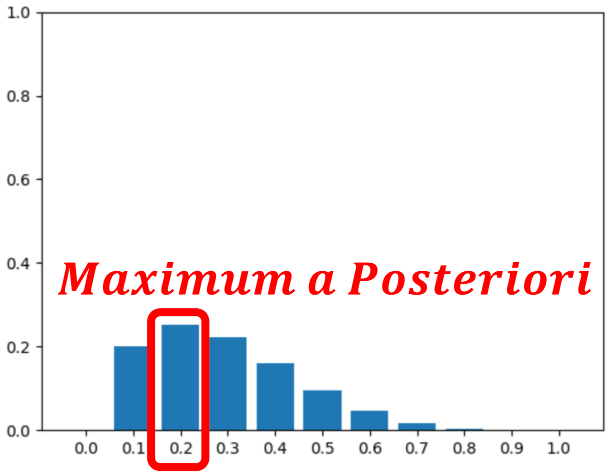
- MAP Estimation
  - Prior: Discrete Uniform Distribution
  - Likelihood: Bernoulli distribution



Prior: $\boldsymbol{P(\theta)}$



Likelihood: $\boldsymbol{P(x|\theta)}$



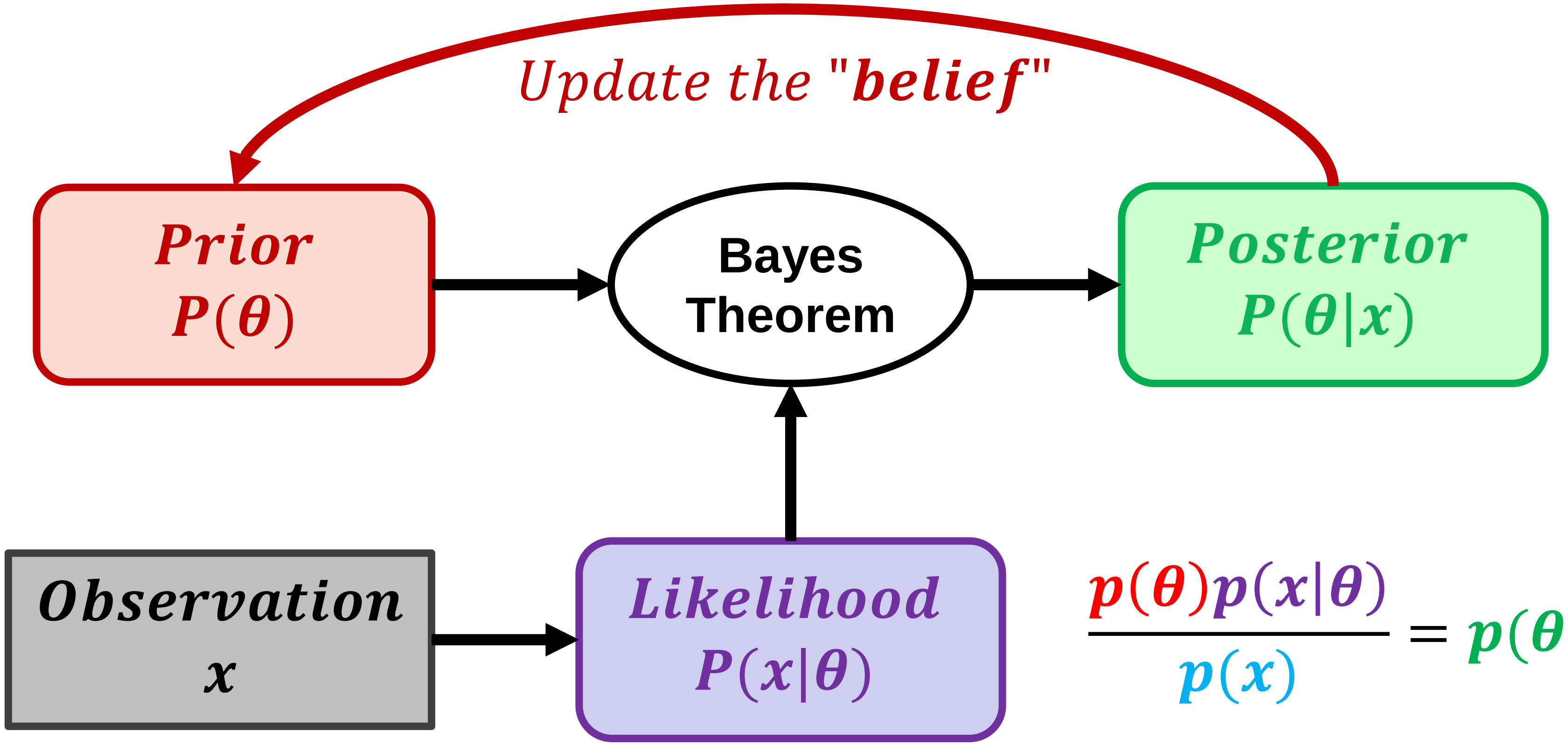
Posterior: $\boldsymbol{P(\theta|x)}$

Bayesian Probability

- Classical Probability View
  - Model parameters have a certain value.
  - The goal of learning is to inference the parameters from sampling data which we call “Estimation”.
- Bayesian Probability View
  - Model parameters have uncertainty.
  - The goal of learning is to inference the probability over every possible parameters, or inference the hyper-parameters.

Bayesian Approach

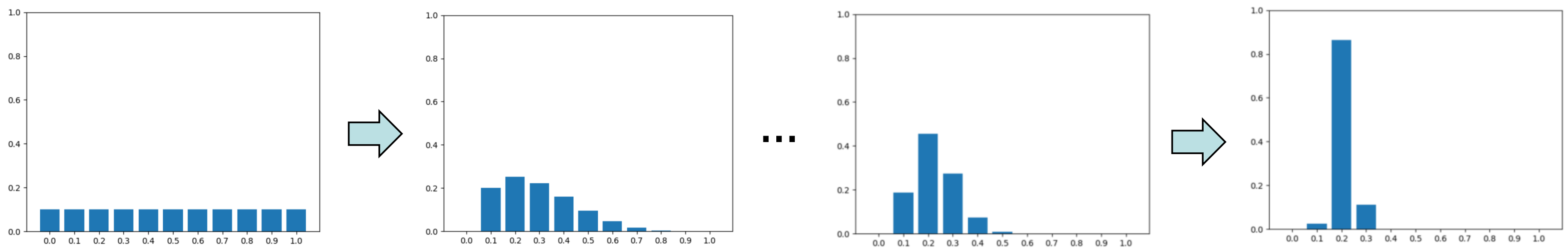
- The current hypothesis of the parameters is the "belief"



$$\frac{p(\theta)p(x|\theta)}{p(x)} = p(\theta|x)$$

Bayesian Approach (Tossing Coins Example)

$$\frac{p(\theta)p(x|\theta)}{p(x)} = p(\theta|x)$$



***Belief*:** $p^{(0)}(\theta)$ $\quad$ $p^{(1)}(\theta)$ $\quad$ … $\quad$ $p^{(N-1)}(\theta)$ $\quad$ $p^{(N)}(\theta)$

***Posterior*:** $p^{(0)}(\theta|x_1)$ $\quad$ $p^{(1)}(\theta|x_2)$ $\quad$ … $\quad$ $p^{(N-1)}(\theta|x_N)$

Bayes Filter

$$\overline{bel}(\boldsymbol{\theta})$$

$$\frac{p(\boldsymbol{\theta})p(x|\boldsymbol{\theta})}{p(x)} = p(\boldsymbol{\theta}|x)$$

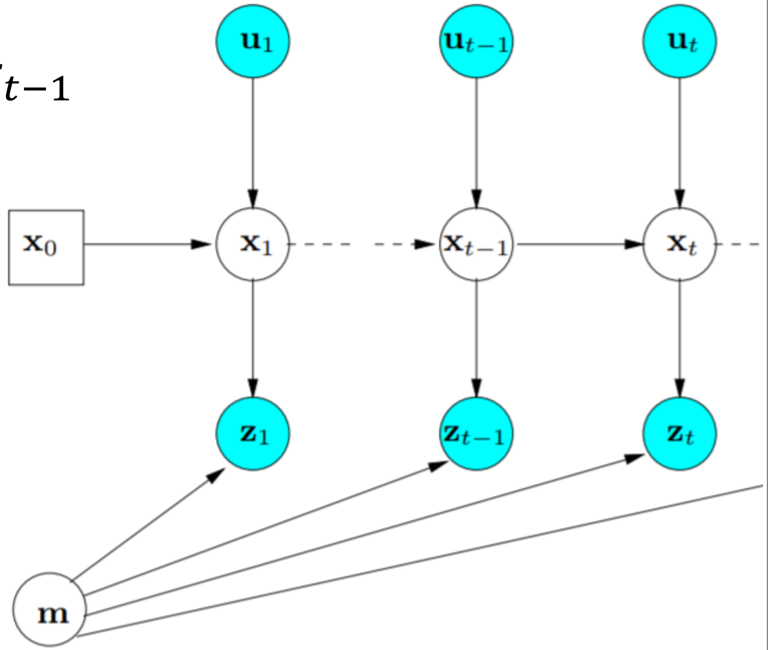
$$bel(\boldsymbol{\theta})$$

State Prediction:

$$P(\mathbf{x_t}|\mathbf{z}_{1:t-1},\mathbf{u}_{1:t}) = \int P(\mathbf{x_t}|\mathbf{x}_{t-1},\mathbf{z}_{1:t-1},\mathbf{u}_{1:t})\mathrm{P}(\mathbf{x}_{t-1}|\mathbf{z}_{1:t-1},\mathbf{u}_{1:t})\, dx_{t-1}$$

$$= \int P(\mathbf{x_t}|\mathbf{x}_{t-1},\mathbf{u}_t)\mathrm{P}(\mathbf{x}_{t-1}|\mathbf{z}_{1:t-1},\mathbf{u}_{1:t})\, dx_{t-1}$$

$$\overline{bel}(\mathbf{x_t}) = \int P(\mathbf{x_t}|\mathbf{x}_{t-1},\mathbf{u}_t)bel(\mathbf{x_{t-1}})dx_{t-1}$$



Bayes Filter

$$\overline{bel}(\boldsymbol{\theta})$$

$$\frac{p(\boldsymbol{\theta})p(x|\boldsymbol{\theta})}{p(x)} = p(\boldsymbol{\theta}|x)$$

$$bel(\boldsymbol{\theta})$$

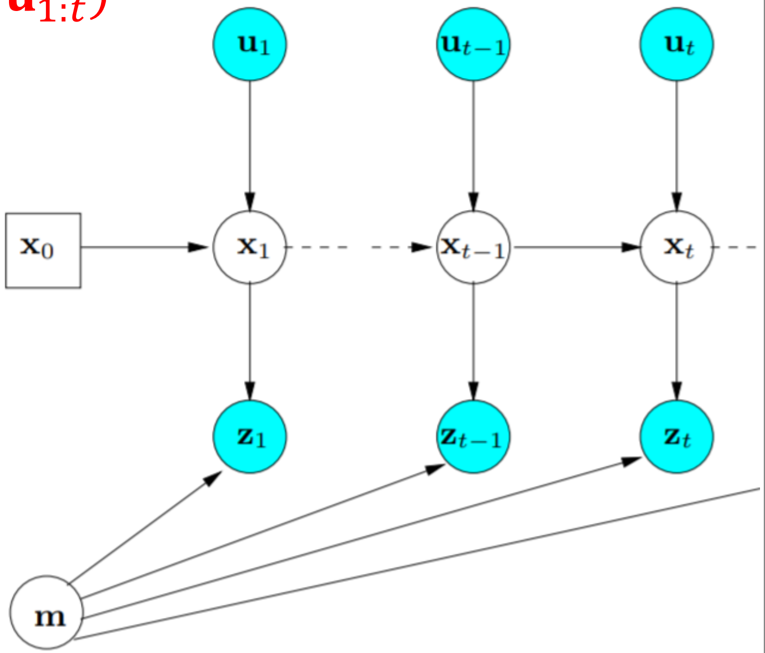
Measurement Update:

$$P(\mathbf{x_t}|\mathbf{z}_{1:t},\mathbf{u}_{1:t}) = \frac{P(\mathbf{z_t}|\mathbf{x_t},\mathbf{z}_{1:t-1},\mathbf{u}_{1:t})P(\mathbf{x_t}|\mathbf{z}_{1:t-1},\mathbf{u}_{1:t})}{P(\mathbf{z_t}|\mathbf{z}_{1:t-1},\mathbf{u}_{1:t})}$$

$$=\eta\ P(\mathbf{z_t}|\mathbf{x_t},\mathbf{z}_{1:t-1},\mathbf{u}_{1:t})P(\mathbf{x_t}|\mathbf{z}_{1:t-1},\mathbf{u}_{1:t})$$

$$=\eta\ P(\mathbf{z_t}|\mathbf{x_t})P(\mathbf{x_t}|\mathbf{z}_{1:t-1},\mathbf{u}_{1:t})$$

$$bel(\mathbf{x_t}) = \eta P(\mathbf{z_t}|\mathbf{x}_t)\overline{bel}(\mathbf{x_t})$$



Bayes Filter

State Prediction: $P(\mathbf{x_t}) = \overline{bel}(\mathbf{x_t}) = \int P(\mathbf{x_t}|\mathbf{x}_{t-1}, \mathbf{u_t})bel(\mathbf{x_{t-1}})dx_{t-1}$

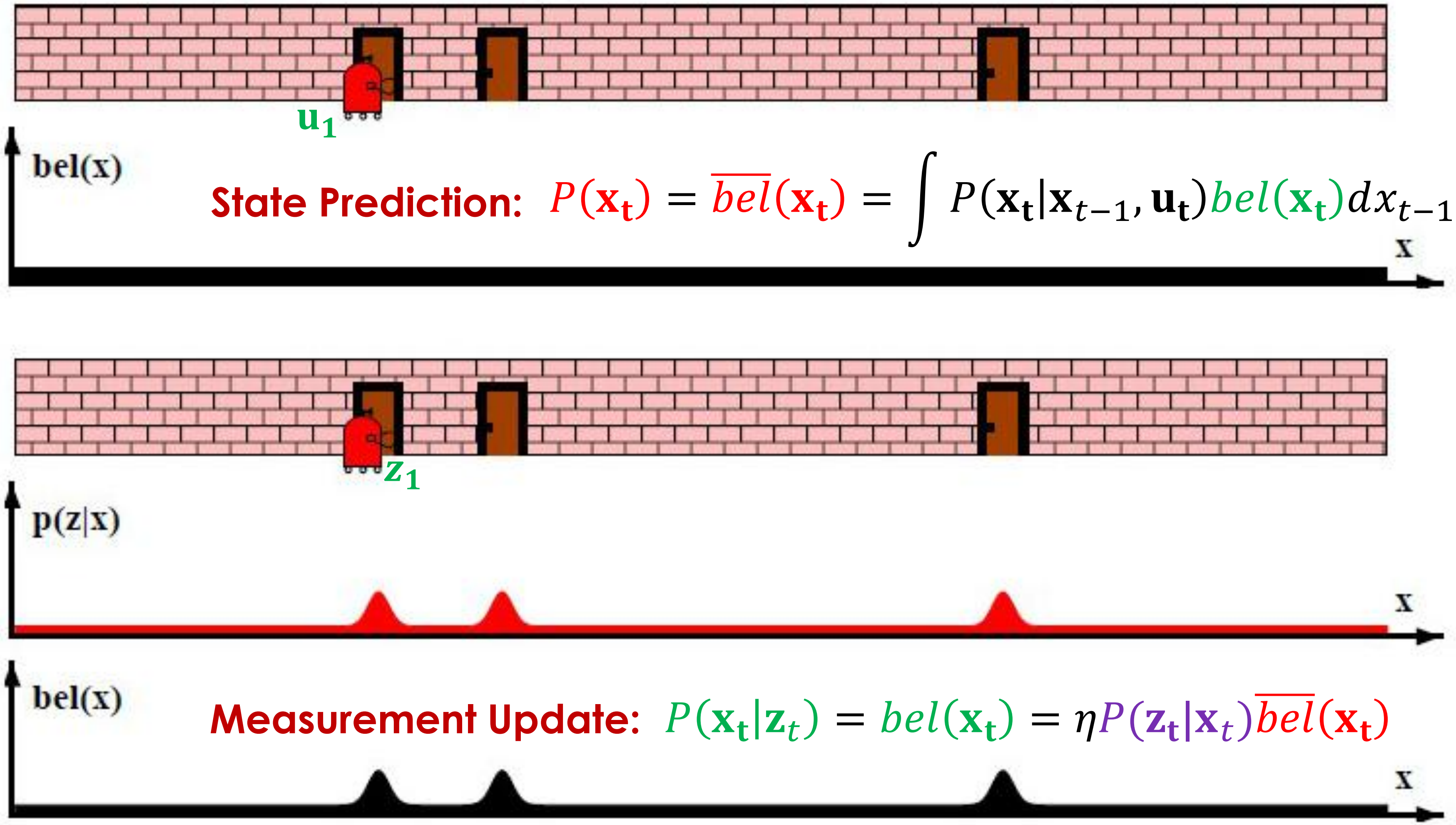
Measurement Update: $P(\mathbf{x_t}|\mathbf{z}_t) = bel(\mathbf{x_t}) = \eta P(\mathbf{z_t}|\mathbf{x}_t)\overline{bel}(\mathbf{x_t})$

Algorithm Bayes_filter($bel(x_{t-1}), u_t, z_t$**):**
for all x_t do
$\overline{bel}(x_t) = \int p(x_t \mid u_t, x_{t-1})\ bel(x_{t-1})\ dx$
$bel(x_t) = \eta\ p(z_t \mid x_t)\ \overline{bel}(x_t)$
endfor
return $bel(x_t)$

Localization

$\overline{bel}(\mathbf{x}_1)$

State Prediction: $P(\mathbf{x}_t) = \overline{bel}(\mathbf{x}_t) = \int P(\mathbf{x}_t | \mathbf{x}_{t-1}, \mathbf{u}_t) bel(\mathbf{x}_t) dx_{t-1}$



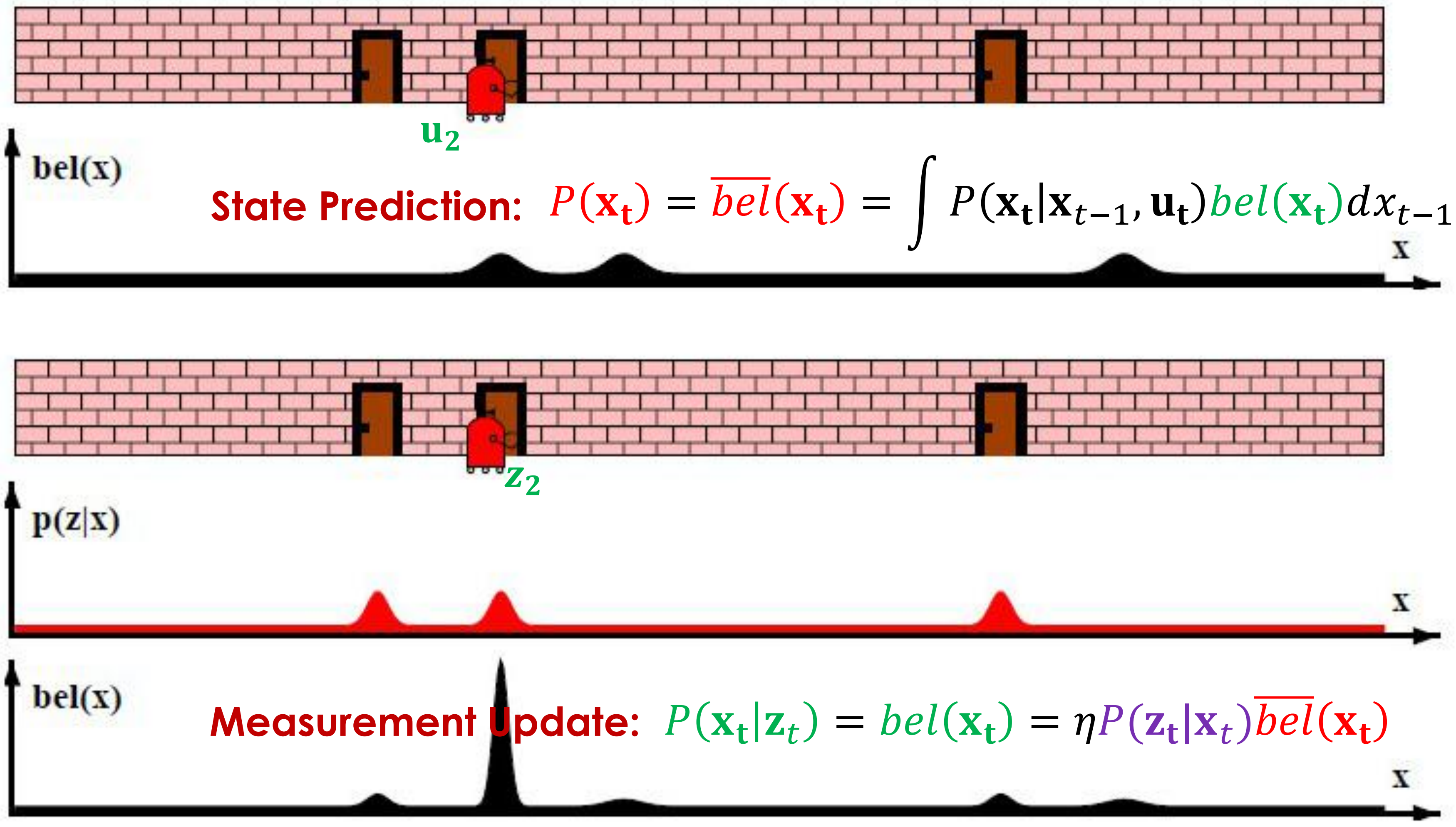
$bel(\mathbf{x}_1)$

Measurement Update: $P(\mathbf{x}_t | \mathbf{z}_t) = bel(\mathbf{x}_t) = \eta P(\mathbf{z}_t | \mathbf{x}_t) \overline{bel}(\mathbf{x}_t)$

Localization

$\overline{bel}(\mathbf{x}_2)$

State Prediction: $P(\mathbf{x}_\mathbf{t}) = \overline{bel}(\mathbf{x}_\mathbf{t}) = \int P(\mathbf{x}_\mathbf{t}|\mathbf{x}_{t-1}, \mathbf{u}_\mathbf{t}) bel(\mathbf{x}_\mathbf{t}) dx_{t-1}$



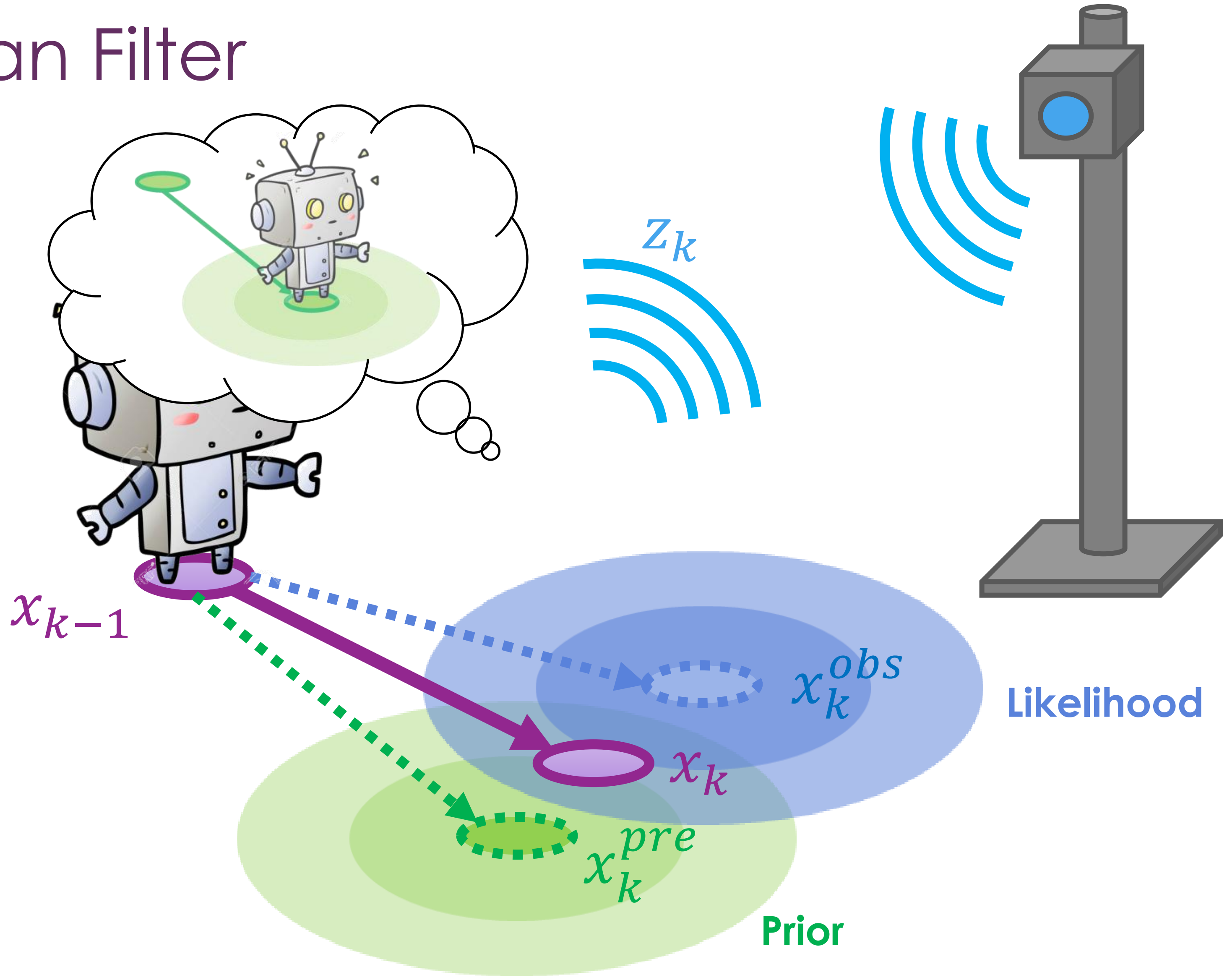
$bel(\mathbf{x}_2)$

Measurement Update: $P(\mathbf{x}_\mathbf{t}|\mathbf{z}_t) = bel(\mathbf{x}_\mathbf{t}) = \eta P(\mathbf{z}_\mathbf{t}|\mathbf{x}_t)\overline{bel}(\mathbf{x}_\mathbf{t})$

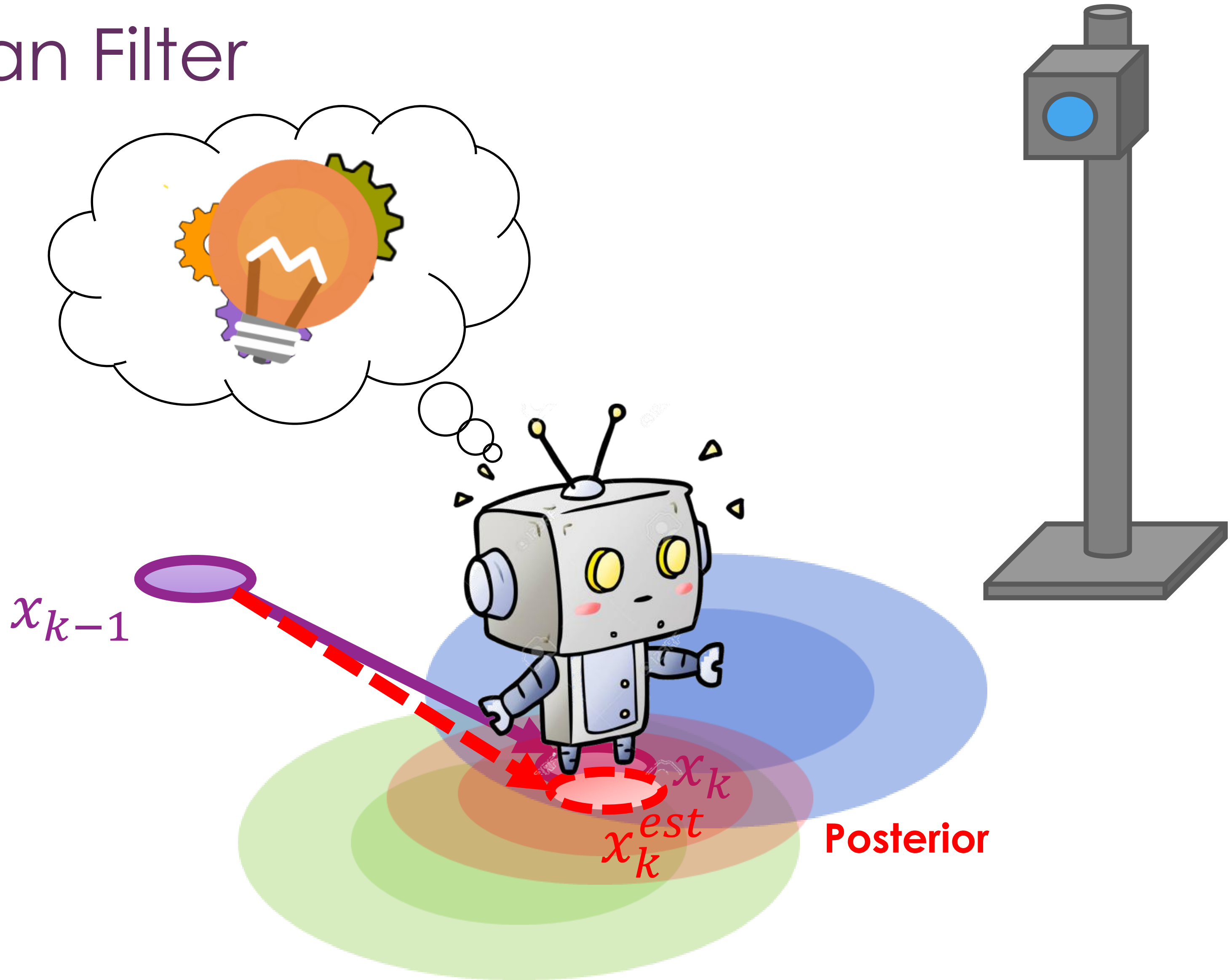
Outline

- State Estimation and SLAM Problem
- SLAM Back-end (Error Compensation)
  - Filter-based Methods
    - Probability Theory and Bayes Filter
    - Kalman Filter (KF) / Extended Kalman Filter (EKF)
      - EKF-SLAM
    - Particle Filter
      - Fast-SLAM
  - Graph-based Methods
    - Pose Graph and Least-square Optimization
    - Gauss-Newton and Levenberg-Marquardt Algorithm
    - Sparse Matrix for Optimization

Kalman Filter

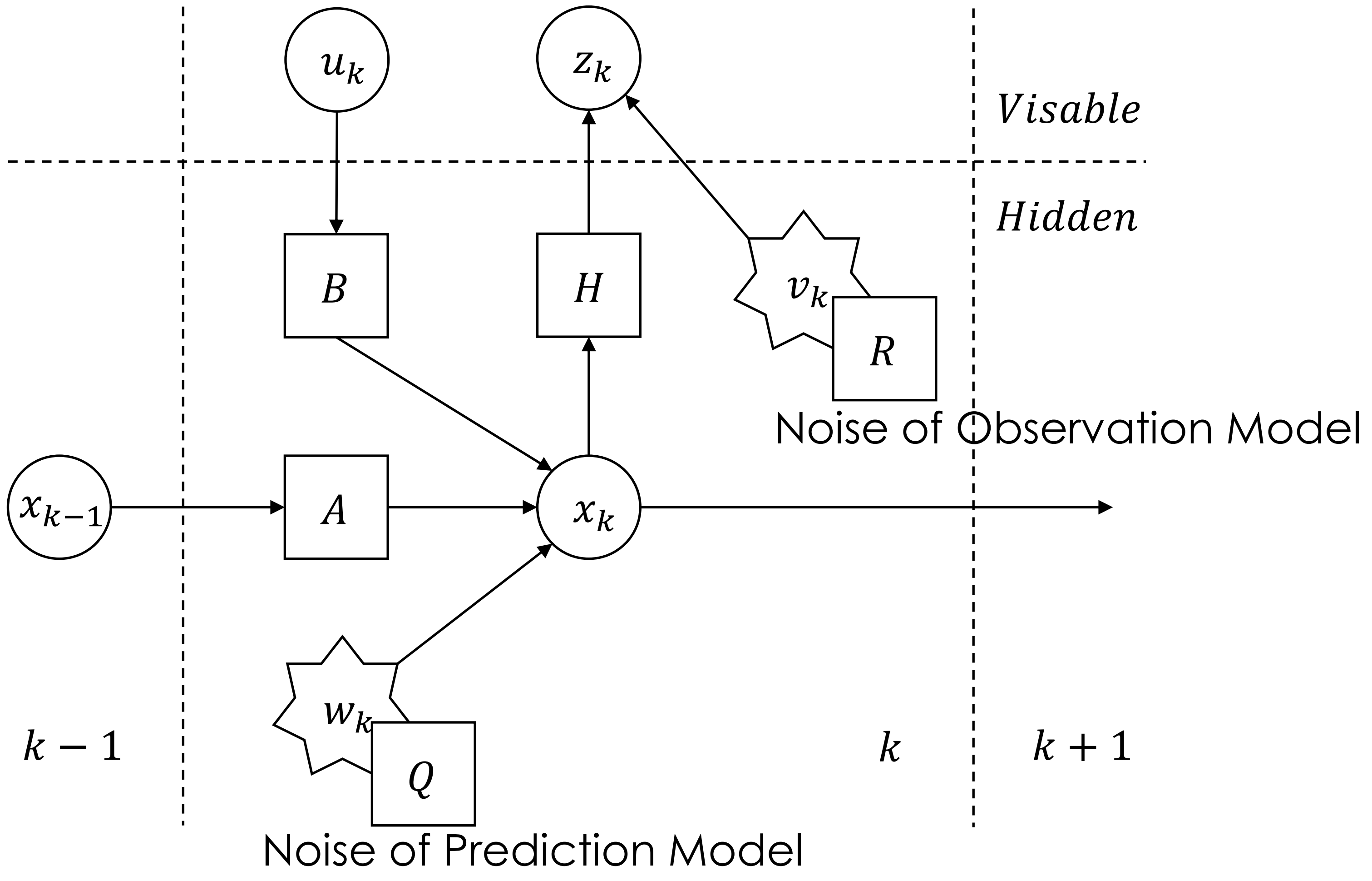


Kalman Filter



Kalman Filter

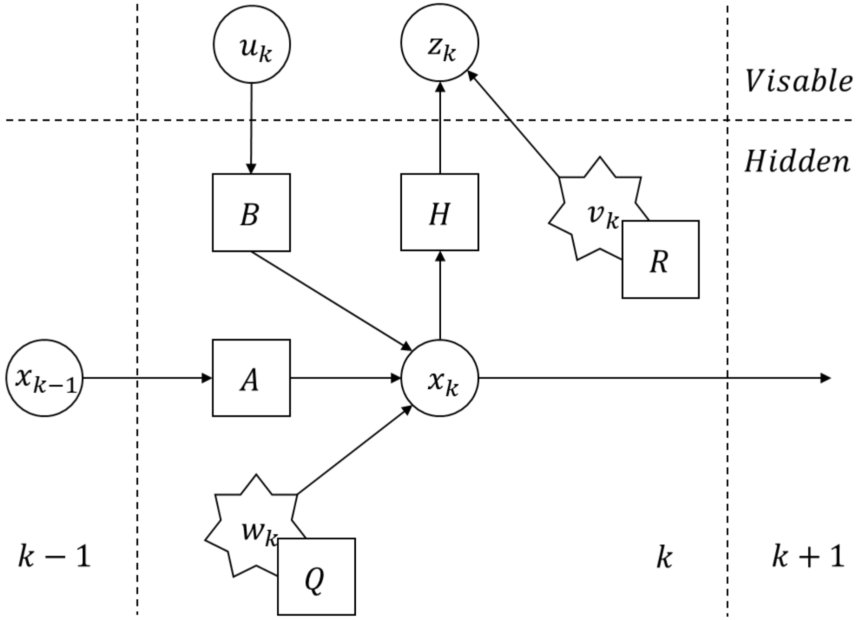
$$x_k = Ax_{k-1} + Bu_k + w_k$$
$$z_k = Hx_k + v_k$$



$$x_k = Ax_{k-1} + Bu_k + w_k$$
$$z_k = Hx_k + v_k$$

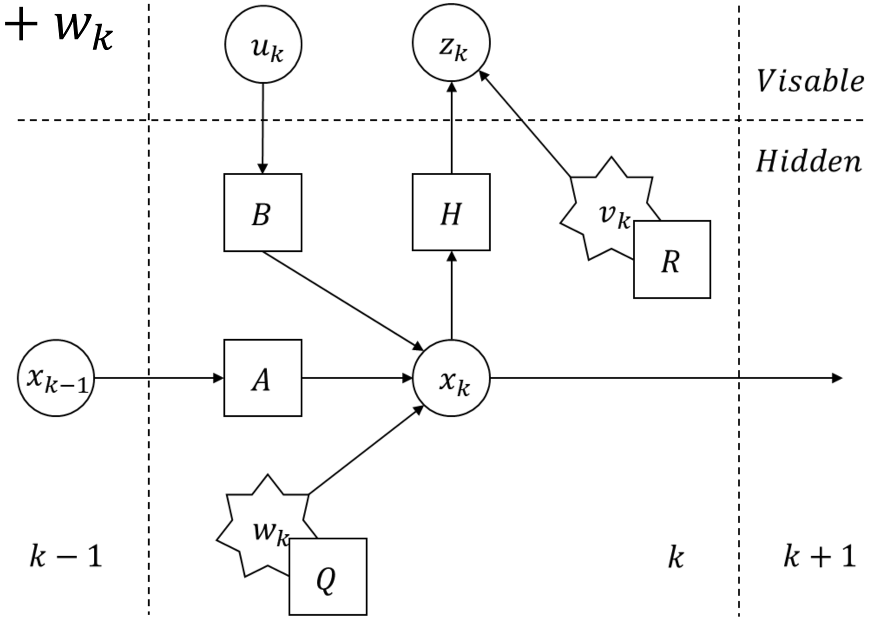
Kalman Filter

- Proof of Kalman Filter
- Notation
  - Truth State: x_k
  - Prediction
    - ✓State: x_k^{pre}
    - ✓Error: $e_k^{pre} = x_k - x_k^{pre}$,
    - ✓Covariance: $P_k^{pre} = E[e_k^{pre} {e_k^{pre}}^T]$
  - Estimation
    - ✓State: x_k^{est}
    - ✓Error: $e_k^{est} = x_k - x_k^{est}$
    - ✓Covariance: $P_k^{est} = E[e_k^{est} {e_k^{est}}^T]$



Kalman Filter

$$x_k = Ax_{k-1} + Bu_k + w_k$$
$$z_k = Hx_k + v_k$$



- The prediction of the state:
  - $x_k^{pre} = Ax_{k-1}^{est} + Bu_k$
- Define the feedback equation:
  - $x_k^{est} = x_k^{pre} + \boldsymbol{K}(z_k - z_k^{pre})$

Kalman Gain — Observation Feedback

- Substitute the observation term of the feedback equation:

$$z_k = Hx_k + v_k\,, z_k^{pre} = Hx_k^{pre}$$
$$x_k^{est} = x_k^{pre} + K\left(Hx_k + v_k - Hx_k^{pre}\right)$$
$$= x_k^{pre} + KH(x_k - x_k^{pre}) + Kv_k$$

- The object is to find the optimal Kalman Gain **K** to minimize the covariance of the estimation :
  - $J = \sum_{min} P_k^{est}$

Kalman Filter

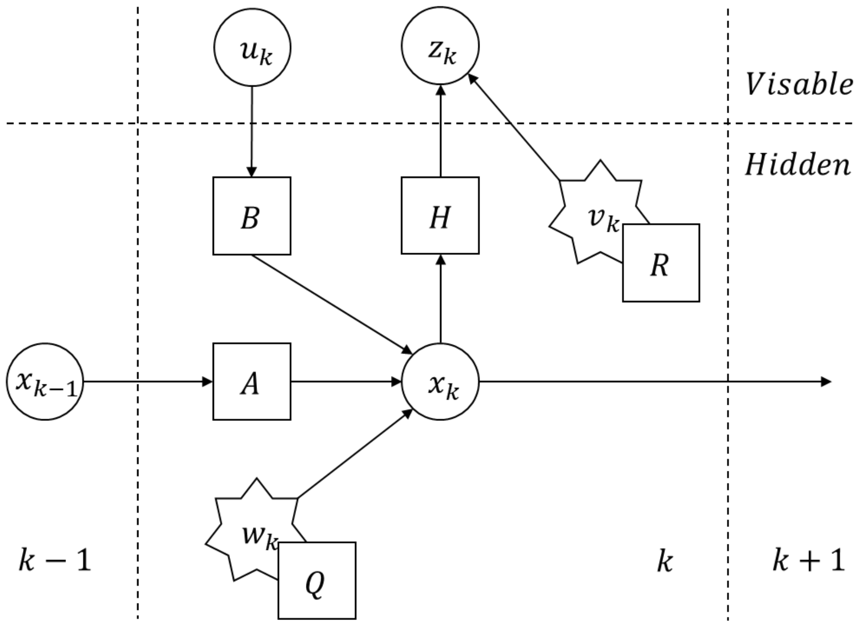
Object: $J = \sum_{min} P_k^{est}$

- Propagate the error along the system.
- Compute the covariance of prediction
  - Prediction error:

$$
\begin{aligned}
e_k^{pre} &= x_k - x_k^{pre} \\
&= (Ax_{k-1} + Bu_k + w_k) - (Ax_{k-1}^{est} + Bu_k) \\
&= A(x_{k-1} - x_{k-1}^{est}) + w_k = Ae_{k-1}^{est} + w_k
\end{aligned}
$$

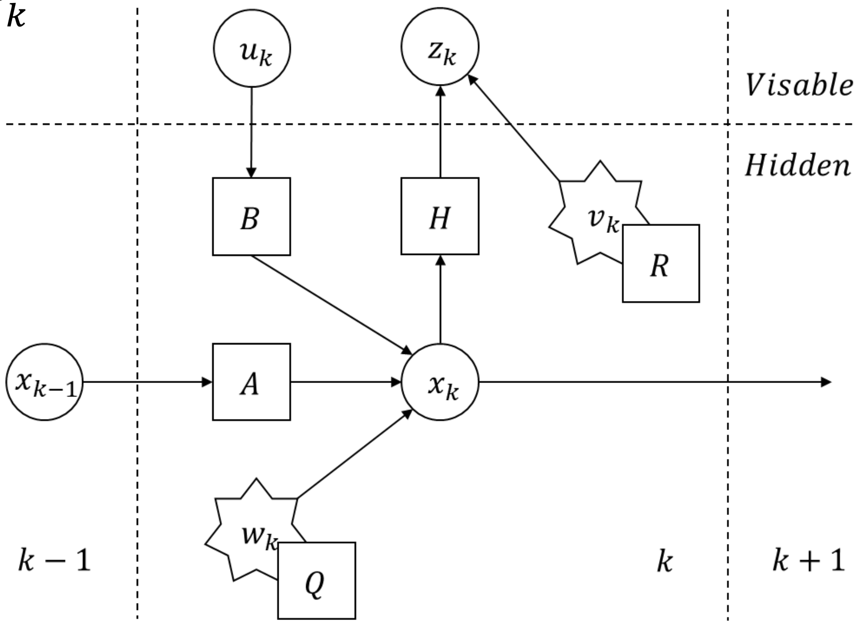
  - Covariance:

$$
\begin{aligned}
P_k^{pre} &= E\left[e_k^{pre} e_k^{pre\,T}\right] \\
&= E[(Ae_{k-1}^{est} + w_k)(Ae_{k-1}^{est} + w_k)^T] \\
&= E\left[Ae_{k-1}^{est} e_{k-1}^{est\;T} A^T\right] + E[w_k w_k^T] \\
&= AP_{k-1}^{est} A^T + Q
\end{aligned}
$$



Object: $J = \sum_{min} P_k^{est}$

Kalman Filter



- Estimate the covariance of posterior
  - Estimation error:

$$\boxed{\begin{aligned} &x_k^{est} \\ &= x_k^{pre} + KH(x_k - x_k^{pre}) + Kv_k \end{aligned}}$$

$$\begin{aligned} e_k^{est} &= x_k - x_k^{est} \\ &= \left(x_k - x_k^{pre}\right) - KH\left(x_k - x_k^{pre}\right) - Kv_k \\ &= (I - KH)e_k^{pre} - Kv_k \end{aligned}$$

  - Covariance:

$$\begin{aligned} P_k^{est} &= E\left[e_k^{est} e_k^{est^T}\right] \\ &= (I - KH)E\left[e_k^{pre} e_k^{pre^T}\right](I - KH)^T + KE\left[v_k v_k^T\right]K^T \overset{=0}{\cancel{- (I - KH)e_k^{pre} KE[v_k] - K^T E\left[v_k^T\right] e_k^{pre^T} (I - KH)^T}} \\ &= (I - KH)P_k^{pre}(I - KH)^T + KRK^T = P_k^{pre} - KHP_k^{pre} - P_k^{pre} H^T K^T + K\left(HP_k^{pre} H_T + R\right)K^T \end{aligned}$$

- Optimize the objective function

$$\frac{\partial P_k^{est}}{\partial K} = -2\left(P_k^{pre} H^T\right) + 2K\left(HP_k^{pre} H^T + R\right) = 0$$

$$K = P_k^{pre} H^T \left(HP_k^{pre} H^T + R\right)^{-1}$$

Kalman Filter

- Kalman Filter Computation Steps:

Set the parameters of Kalman filter A, B, Q, R

1. Predict the next state

$$x_k^{pre} = Ax_{k-1}^{est} + Bu_k$$

2. Compute the prediction covariance

$$P_k^{pre} = AP_{k-1}^{est}A^T + Q$$

3. Compute Kalman-gain

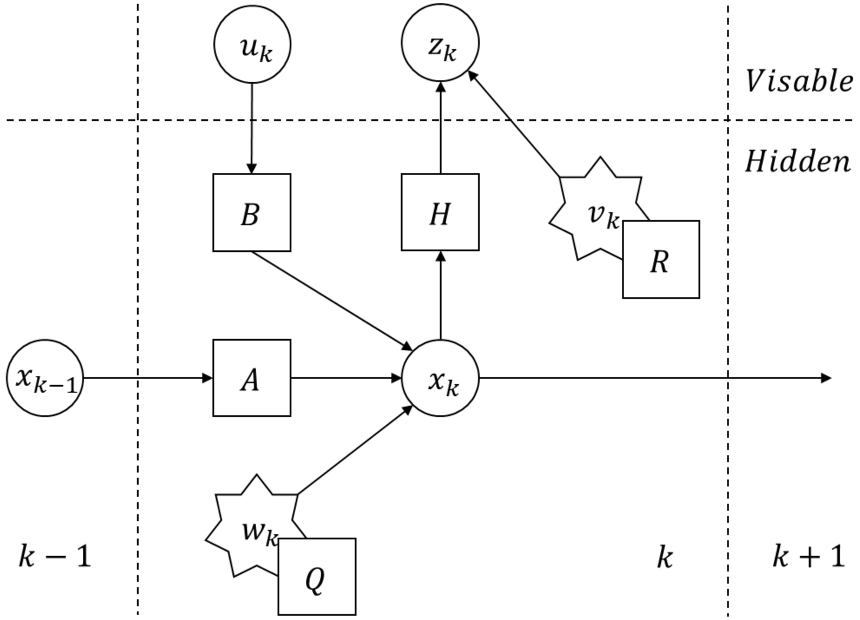
$$K_k = P_k^{pre}H^T\left(HP_k^{pre}H^T + R\right)^{-1}$$

4. Estimate the mean of the state

$$x_k^{est} = x_k^{pre} + K_k\left(z_k - Hx_k^{pre}\right)$$

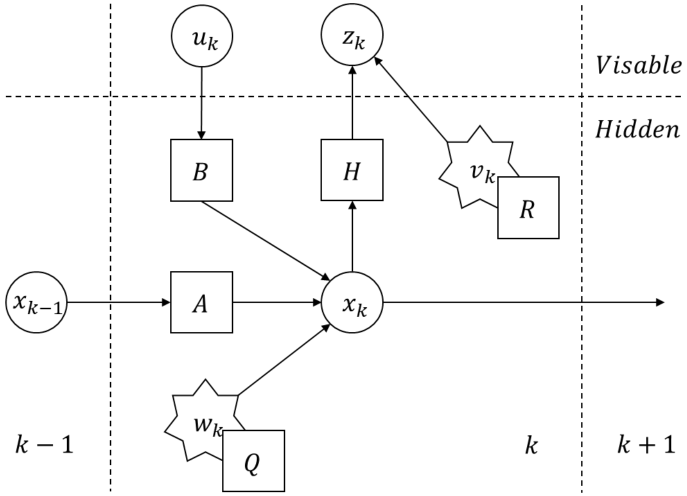
5. Estimate the covariance of the state

$$P_k^{est} = (I - K_kH)P_k^{pre}$$



$$\boxed{\begin{aligned} x_k^{pre} &= Ax_{k-1}^{est} + Bu_k \\ P_k^{pre} &= AP_{k-1}^{est}A^T + Q \\ K_k &= P_k^{pre}H^T\left(HP_k^{pre}H^T + R\right)^{-1} \\ x_k^{est} &= x_k^{pre} + K_k\left(z_k - Hx_k^{pre}\right) \\ P_k^{est} &= (I - K_kH)P_k^{pre} \end{aligned}}$$

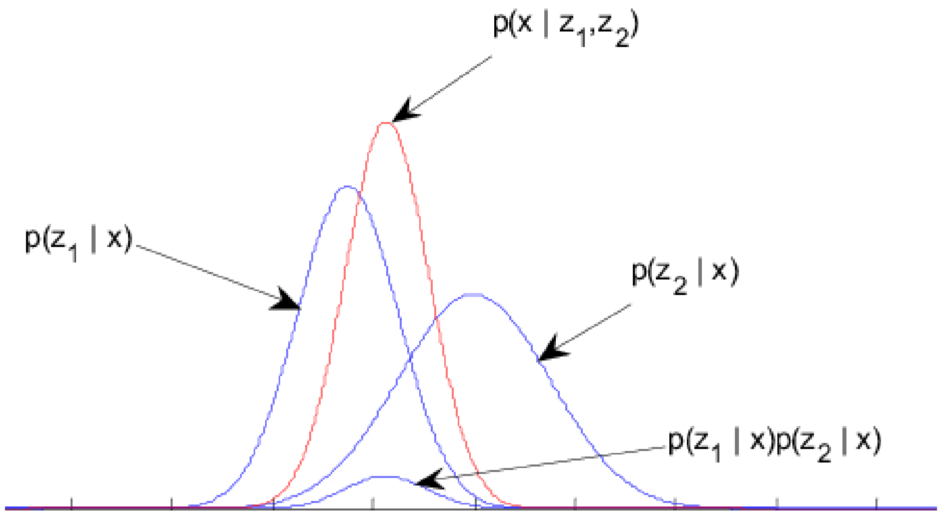
Kalman Filter



- Probabilistic view of Kalman filter
- Prediction Model & Observation Model
  - $x_k = Ax_{k-1} + Bu_k + w_k$
  - $z_k = Hx_k + v_k$

$$\boxed{x_k = H^{-1}(z_k - v_k)}$$

- Probability distribution of the prediction and observation
  - $p\left(x_k^{pre}\right) = \mathcal{N}(Ax_{k-1}^{est} + Bu_k,\ AP_{k-1}^{est}A^T + Q)$
  - $p(x_k^{obs}) = \mathcal{N}(H^{-1}z_k,\ H^{-1}RH^{-T})$
- Fusion of Gaussian Distribution
  - $S = \Sigma_0(\Sigma_0 + \Sigma_1)^{-1}$
  - $\mu = \mu_0 + S(\mu_1 - \mu_0)$
  - $\Sigma = \Sigma_0 - S\Sigma_0$



Kalman Filter

$$S = \Sigma_0(\Sigma_0 + \Sigma_1)^{-1}$$
$$\mu = \mu_0 + S(\mu_1 - \mu_0)$$
$$\Sigma = \Sigma_0 - K\Sigma_0$$

$$p(x_k^{pre}) = \mathcal{N}(Ax_{k-1}^{est} + Bu_k, AP_{k-1}^{est}A + Q)$$
$$p(x_k^{obs}) = \mathcal{N}(H^{-1}z_k, H^{-1}RH^{-T})$$

- Fusion the distribution of prediction and observation
  - Mean: x_k^{est}

$$= x_{k-1}^{pre} + P_k^{pre}\left(P_k^{pre} + H^{-1}RH^{-T}\right)^{-1}\left(x_k^{pre} - H^{-1}z_k\right)$$
$$= x_{k-1}^{pre} + P_k^{pre}H^TH^{-T}\left(P_k^{pre} + H^{-1}RH^{-T}\right)^{-1}H^{-1}H\left(x_k^{pre} - H^{-1}z_k\right)$$
$$= x_{k-1}^{est} + Bu_k + P_k^{pre}H^T\left(HP_k^{pre}H^T + R\right)^{-1}\left(Hx_k^{pre} - z_k\right)$$
$$= x_{k-1}^{est} + Bu_k + K_k\left(Hx_k^{pre} - z_k\right)$$

$$\boxed{A^{-1}B^{-1} = (BA)^{-1}}$$

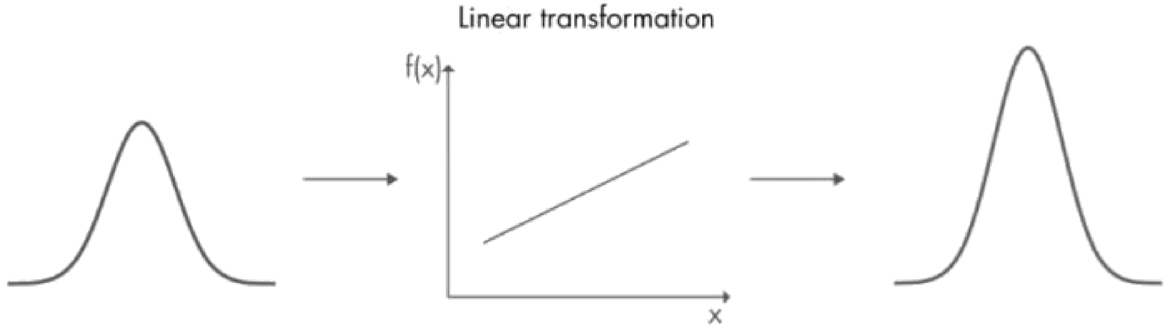
  - Covariance: P_k^{est}

$$= P_k^{pre} - P_k^{pre}\left(P_k^{pre} + H^{-1}RH^{-T}\right)^{-1}P_k^{pre}$$
$$= P_k^{pre} - P_k^{pre}H^TH^{-T}\left(P_k^{pre} + H^{-1}RH^{-T}\right)^{-1}H^{-1}HP_k^{pre}$$
$$= P_k^{pre} - P_k^{pre}H^T\left(HP_k^{pre}H^T + R\right)^{-1}HP_k^{pre}$$
$$= P_k^{pre} - K_kHP_k^{pre} = (I - K_kH)P_k^{pre}$$

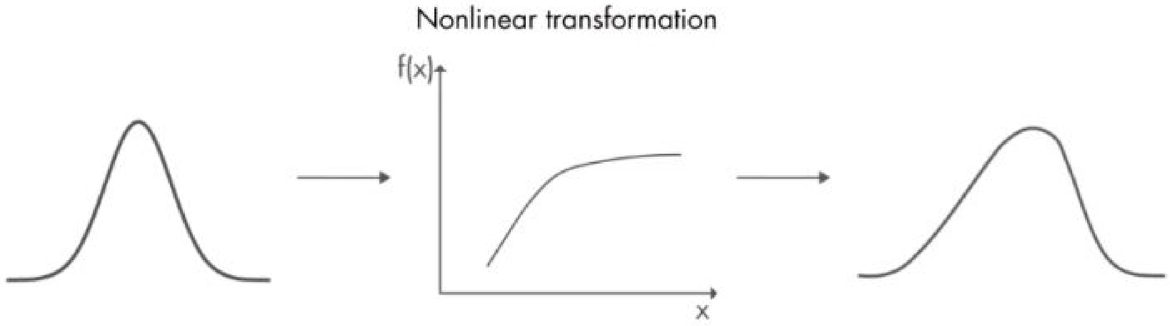
$$x_k^{pre} = Ax_{k-1}^{est} + Bu_k$$
$$P_k^{pre} = AP_{k-1}^{est}A^T + Q$$
$$K_k = P_k^{pre}H^T\left(HP_k^{pre}H^T + R\right)^{-1}$$
$$x_k^{est} = x_k^{pre} + K_k\left(z_k - Hx_k^{pre}\right)$$
$$P_k^{est} = (I - K_kH)P_k^{pre}$$

Extended Kalman Filter (EKF)

- Kalman filter assumes the prediction model to be linear, the Gaussian distribution of the state will transform to another Gaussian :



- However, the prediction model is usually nonlinear, the state distribution after transformation will not be a Gaussian.



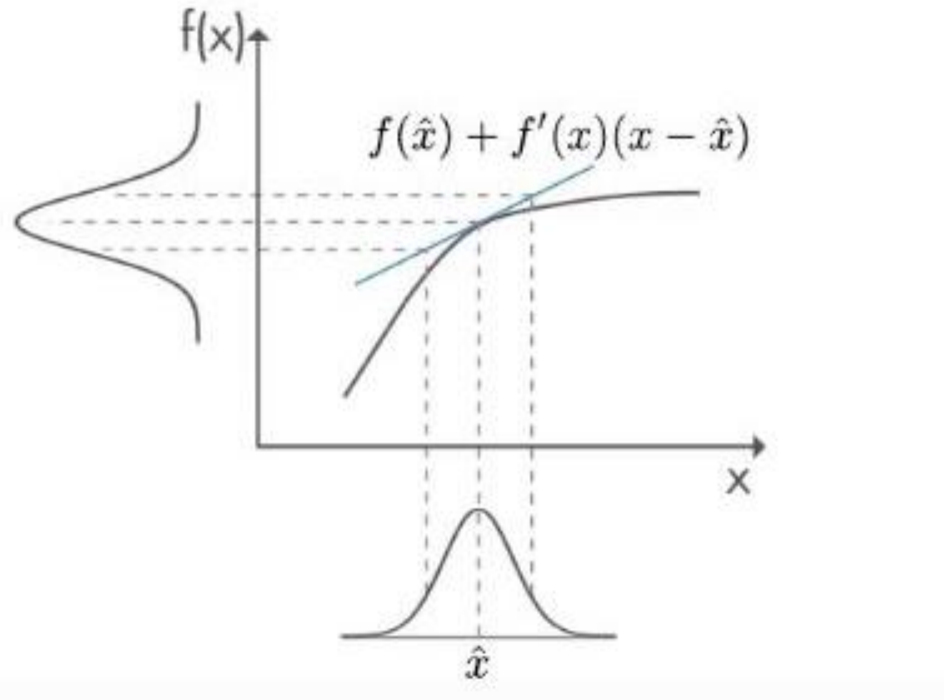
Extended Kalman Filter (EKF)

- In this case, we can approximate the nonlinear transform by utilizing the 1st order Taylor expansion at the mean of the state:

- Prediction Model & Observation Model
  - $x_k = f(x_{k-1}, u_k) + w_k$
  - $z_k = h(x_k) + v_k$

- Jacobian Matrix:
  - $F_k = \frac{\partial f(\hat{x}_{k-1}, u_k)}{\partial x}, H_k = \frac{\partial h(\hat{x}_k)}{\partial x}$

- Linearized System
  - $x_k = f(\hat{x}_{k-1}, u_k) + F_k(x_{k-1} - \hat{x}_{k-1}) + w_k$
  - $z_k = h(\hat{x}_k) + H_k(x_k - \hat{x}_k) + v_k$



Extended Kalman Filter (EKF)

- Linearized System
  - $x_k = f(\hat{x}_{k-1}, u_k) + F_k(x_{k-1} - \hat{x}_{k-1}) + w_k$
  - $z_k = h(\hat{x}_k) + H_k(x_k - \hat{x}_k) + v_k$
- Computation of EKF

$$x_k^{pre} = f(x_{k-1}^{est}, u_k)$$

$$P_k^{pre} = F_k P_{k-1}^{pre} F_k^T + Q$$

$$K_k = P_k^{pre} H^T \left(H P_k^{pre} H^T + R\right)^{-1}$$

$$x_k^{est} = x_k^{pre} + K_k\left(z_k - H x_k^{pre}\right)$$

$$P_k^{est} = (I - K_k H) P_k^{pre}$$

Kalman-Filter

$$x_k^{pre} = A x_{k-1}^{est} + B u_k$$

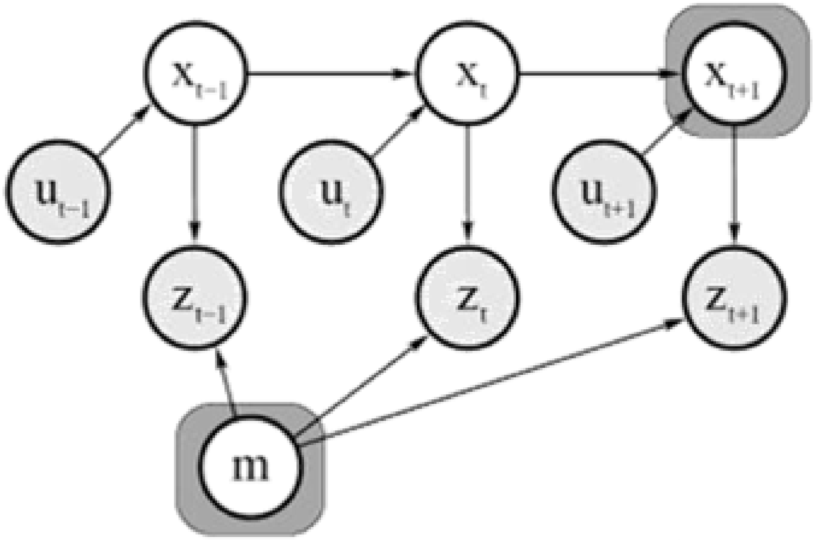
$$P_k^{pre} = A P_{k-1}^{est} A^T + Q$$

$$K_k = P_k^{pre} H^T \left(H P_k^{pre} H^T + R\right)^{-1}$$

$$x_k^{est} = x_k^{pre} + K_k\left(z_k - H x_k^{pre}\right)$$

$$P_k^{est} = (I - K_k H) P_k^{pre}$$

EKF-SLAM



- Consider the SLAM problem
- Define the state as the concatenation of robot's pose and landmarks position:

$$s_k = (\underbrace{x, y, \theta}_{\text{robot's pose}}, \underbrace{m_{1,x}, m_{1,y}}_{\text{Landmark 1}}, \underbrace{m_{2,x}, m_{2,y}}_{\text{Landmark 2}}, \ldots, \underbrace{m_{n,x}, m_{n,y}}_{\text{Landmark n}})^T$$

- Probability distribution of the state:

$$\begin{bmatrix} x \\ y \\ \theta \\ m_{1,x} \\ m_{1,y} \\ \vdots \\ m_{n,x} \\ m_{n,y} \end{bmatrix} \rightarrow \mu = \begin{bmatrix} \mathbf{x} \\ \mathbf{m} \end{bmatrix}, \Sigma = \begin{bmatrix} \Sigma_{\mathbf{xx}} & \Sigma_{\mathbf{xm}} \\ \Sigma_{\mathbf{mx}} & \Sigma_{\mathbf{mm}} \end{bmatrix}$$

Extended Kalman-Filter

$$x_k^{pre} = f(x_k^{est}, u_k)$$
$$P_k^{pre} = F_k P_{k-1}^{est} F_k^T + Q$$
$$K_k = P_k^{pre} H^T (H P_k^{pre} H^T + R)^{-1}$$
$$x_k^{est} = x_k^{pre} + K_k (z_k - H x_k^{pre})$$
$$P_k^{est} = (I - K_k H) P_k^{pre}$$

EKF-SLAM (Prediction Model)

- In the past section, we have learnt the equation of motion model

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} v\cos\theta \\ v\,sin\theta \\ \omega \end{bmatrix}$$

- In simulation process, we utilize the numerical integral to compute the future state with a small interval dt.

- However, in SLAM task we need an accurate state prediction for a given interval Δt, which can be obtained by integrating over the motion equation:

$$\begin{cases} x(t) = \int v\,cos\theta\,dt \\ y(t) = \int v\,sin\theta\,dt \\ \theta(t) = \int \omega\,dt \end{cases}$$

$$\begin{cases} x(t) = \int v\, cos\theta\, dt \\ y(t) = \int v\, sin\theta\, dt \\ \theta(t) = \int \omega\, dt \end{cases}$$

EKF-SLAM (Prediction Model)

- First, we integrate the angle:
  - $\theta(t) = \int \omega\, dt, \quad \theta(t) = \omega t + C$
- Consider the initial condition of angle
  - $\theta(0) = \hat{\theta}$, we can get the scalar term $C = \hat{\theta}$
- Then we can substitute the angle term for integral of x and y
  - $x(t) = \int v \cos(\hat{\theta} + \omega t)\, dt = \frac{v}{\omega}\sin(\hat{\theta} + \omega t) + C$

    $y(t) = \int v \sin(\hat{\theta} + \omega t)\, dt = -\frac{v}{\omega}\cos(\hat{\theta} + \omega t) + C$
- Consider the initial condition of position
  - $x(0) = \hat{x}, y(0) = \hat{y}$, we can get

    $x(t) = \int v \cos(\hat{\theta} + \omega t)\, dt = \frac{v}{\omega}\sin(\hat{\theta} + \omega t) - \frac{v}{\omega}\sin(\hat{\theta}) + \hat{x}$

    $y(t) = \int v \sin(\hat{\theta} + \omega t)\, dt = -\frac{v}{\omega}\cos(\hat{\theta} + \omega t) + \frac{v}{\omega}\cos(\hat{\theta}) + \hat{y}$

Velocity Motion Model

EKF-SLAM (Prediction Model)

- Prediction Model

$$\begin{cases} x' = \hat{x} - \frac{v}{\omega}\sin(\hat{\theta}) + \frac{v}{\omega}\sin(\hat{\theta} + \omega\Delta t) \\ y' = \hat{y} + \frac{v}{\omega}\cos(\hat{\theta}) - \frac{v}{\omega}\cos(\hat{\theta} + \omega\Delta t) \\ \theta' = \omega\Delta t + \hat{\theta} \end{cases}, \quad \begin{bmatrix} x' \\ y' \\ \theta' \end{bmatrix} = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix} + \begin{bmatrix} -\frac{v}{\omega}\sin(\theta) + \frac{v}{\omega}\sin(\theta + \omega_t\Delta t) \\ \frac{v}{\omega}\cos(\theta) - \frac{v}{\omega}\cos(\theta + \omega_t\Delta t) \\ \omega\Delta\mathrm{t} \end{bmatrix}$$

- Linearized the velocity motion model:

  - $$F_t^x = \frac{\partial f}{\partial (x,y,\theta)^T}\left[\begin{bmatrix} x \\ y \\ \theta \end{bmatrix} + \begin{bmatrix} -\frac{v_t}{\omega_t}\sin(\theta) + \frac{v_t}{\omega_t}\sin(\theta + \omega_t\Delta t) \\ \frac{v_t}{\omega_t}\cos(\theta) - \frac{v_t}{\omega_t}\cos(\theta + \omega_t\Delta t) \\ \omega\Delta t + \theta \end{bmatrix}\right] = I + \frac{\partial f}{\partial (x,y,\theta)^T}\begin{bmatrix} -\frac{v_t}{\omega_t}\sin(\theta) + \frac{v_t}{\omega_t}\sin(\theta + \omega_t\Delta t) \\ \frac{v_t}{\omega_t}\cos(\theta) - \frac{v_t}{\omega_t}\cos(\theta + \omega_t\Delta t) \\ \omega_t\Delta t \end{bmatrix}$$

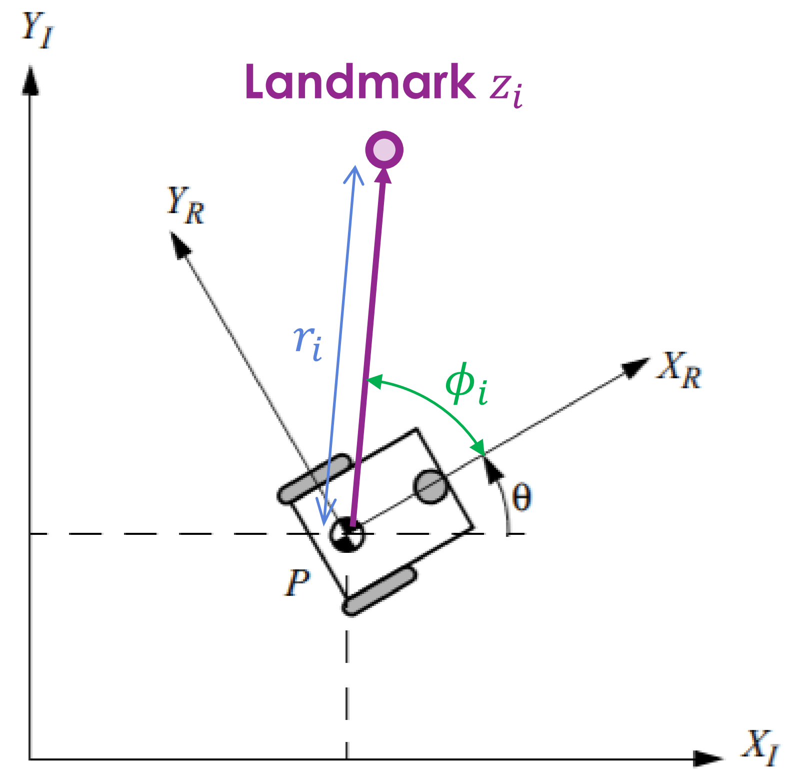
$$= I + \begin{bmatrix} 0 & 0 & -\frac{v_t}{\omega_t}\cos(\theta) + \frac{v_t}{\omega_t}\cos(\theta + \omega_t\Delta t) \\ 0 & 0 & -\frac{v_t}{\omega_t}\sin(\theta) + \frac{v_t}{\omega_t}\sin(\theta + \omega_t\Delta t) \\ 0 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 & -\frac{v_t}{\omega_t}\cos(\theta) + \frac{v_t}{\omega_t}\cos(\theta + \omega_t\Delta t) \\ 0 & 1 & -\frac{v_t}{\omega_t}\sin(\theta) + \frac{v_t}{\omega_t}\sin(\theta + \omega_t\Delta t) \\ 0 & 0 & 1 \end{bmatrix}$$

EKF-SLAM (Observation Model)

- Obtain the relative measurement of landmarks: $z_i = (r_i, \phi_i)^T$
  - $\begin{bmatrix} m_{i,x} \\ m_{i,y} \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} r_i\cos(\phi_i + \theta) \\ r_i\sin(\phi_i + \theta) \end{bmatrix}$

- Define the following term:
  - $\delta = \begin{bmatrix} \delta_x \\ \delta_y \end{bmatrix} = \begin{bmatrix} m_{i,x} - x \\ m_{i,y} - y \end{bmatrix}, \ q = \delta^T\delta$

- The observation can be represented as:
  - $z_i = \begin{bmatrix} \sqrt{q} \\ atan2(\delta_x, \delta_y) - \theta \end{bmatrix}$



EKF-SLAM (Observation Model)

- Given observation model

$$z_i = \begin{bmatrix} \sqrt{q} \\ atan2(\delta_x, \delta_y) - \theta \end{bmatrix}, \delta = \begin{bmatrix} m_{i,x} - x \\ m_{i,y} - y \end{bmatrix}, q = \delta^T \delta$$

- Linearized the observation model :

  - $H^i = \frac{\partial z_i}{\partial(x, y, \theta, m_{i,x}, m_{i,y})} = \begin{bmatrix} \frac{\partial \sqrt{q}}{\partial x} & \frac{\partial \sqrt{q}}{\partial y} & \cdots \\ \frac{\partial atan2(\delta_x, \delta_y)}{\partial x} & \frac{\partial atan2(\delta_x, \delta_y)}{\partial y} & \cdots \end{bmatrix}$

$$= \frac{1}{q} \begin{bmatrix} -\sqrt{q}\delta_x & -\sqrt{q}\delta_y & 0 & \sqrt{q}\delta_x & \sqrt{q}\delta_y \\ \delta_y & -\delta_x & -q & -\delta_y & \delta_x \end{bmatrix}$$

$$\boxed{\frac{\partial \sqrt{q}}{\partial x} = \frac{1}{2} \frac{1}{\sqrt{q}} 2\delta_x(-1) = \frac{1}{q}(-\sqrt{q}\delta_x)}$$

$$\boxed{\begin{aligned} \frac{\partial}{\partial x} \operatorname{atan2}(y, x) &= \frac{\partial}{\partial x} \arctan\left(\frac{y}{x}\right) = -\frac{y}{x^2 + y^2}, \\ \frac{\partial}{\partial y} \operatorname{atan2}(y, x) &= \frac{\partial}{\partial y} \arctan\left(\frac{y}{x}\right) = \frac{x}{x^2 + y^2}. \end{aligned}}$$

EKF-SLAM

- Prediction Model
  - $F_t = \begin{bmatrix} 1 & 0 & 0 & 0\cdots 0 \\ 0 & 1 & 0 & 0\cdots 0 \\ 0 & 0 & 1 & 0\cdots 0 \end{bmatrix}^T * F_t^x$, in which $F_t^x = \begin{bmatrix} 1 & 0 & -\frac{v_t}{\omega_t}\cos(\theta) + \frac{v_t}{\omega_t}\cos(\theta + \omega_t \Delta t) \\ 0 & 1 & -\frac{v_t}{\omega_t}\sin(\theta) + \frac{v_t}{\omega_t}\sin(\theta + \omega_t \Delta t) \\ 0 & 0 & 1 \end{bmatrix}$
- Observation Model
  - $$H_t = \begin{array}{c} \begin{bmatrix} 1 & 0 & 0 & 0\cdots 0 & 0 & 0 & 0\cdots 0 \\ 0 & 1 & 0 & 0\cdots 0 & 0 & 0 & 0\cdots 0 \\ 0 & 0 & 1 & 0\cdots 0 & 0 & 0 & 0\cdots 0 \\ 0 & 0 & 0 & 0\cdots 0 & 1 & 0 & 0\cdots 0 \\ 0 & 0 & 0 & 0\cdots 0 & 0 & 1 & 0\cdots 0 \end{bmatrix}^T \\ \begin{array}{cccccccc} x & y & \theta & & m_{i,x} & m_{i,y} & \end{array} \end{array} * H_t^i$$

Extended Kalman-Filter

$x_k^{pre} = f(x_k^{est}, u_k)$

$P_k^{pre} = F_k P_{k-1}^{est} F_k^T + Q$

$K_k = P_k^{pre} H^T \left(H P_k^{pre} H^T + R\right)^{-1}$

$x_k^{est} = x_k^{pre} + K_k\left(z_k - H x_k^{pre}\right)$

$P_k^{est} = (I - K_k H) P_k^{pre}$

, in which $H_t^i = \frac{1}{q}\begin{bmatrix} -\sqrt{q}\delta_x & -\sqrt{q}\delta_y & 0 & \sqrt{q}\delta_x & \sqrt{q}\delta_y \\ \delta_y & -\delta_x & -q & -\delta_y & \delta_x \end{bmatrix}$, $\delta = \begin{bmatrix} m_{i,x} - x \\ m_{i,y} - y \end{bmatrix}$, $q = \delta^T \delta$

Outline

- State Estimation and SLAM Problem
- SLAM Back-end (Error Compensation)
  - Filter-based Methods
    - Probability Theory and Bayes Filter
    - Kalman Filter (KF) / Extended Kalman Filter (EKF)
      - EKF-SLAM
    - Particle Filter
      - Fast-SLAM
  - Graph-based Methods
    - Pose Graph and Least-square Optimization
    - Gauss-Newton and Levenberg-Marquardt Algorithm
    - Sparse Matrix for Optimization

EKF-SLAM

- Prediction Model

  - $F_t = \begin{bmatrix} 1 & 0 & 0 & 0 \cdots 0 \\ 0 & 1 & 0 & 0 \cdots 0 \\ 0 & 0 & 1 & 0 \cdots 0 \end{bmatrix}^T * F_t^x$, in which $F_t^x = \begin{bmatrix} 1 & 0 & -\frac{v_t}{\omega_t}\cos(\theta) + \frac{v_t}{\omega_t}\cos(\theta + \omega_t \Delta t) \\ 0 & 1 & -\frac{v_t}{\omega_t}\sin(\theta) + \frac{v_t}{\omega_t}\sin(\theta + \omega_t \Delta t) \\ 0 & 0 & 1 \end{bmatrix}$

- Observation Model

  - $$H_t = \begin{array}{c} \begin{bmatrix} 1 & 0 & 0 & 0 \cdots 0 & 0 & 0 & 0 \cdots 0 \\ 0 & 1 & 0 & 0 \cdots 0 & 0 & 0 & 0 \cdots 0 \\ 0 & 0 & 1 & 0 \cdots 0 & 0 & 0 & 0 \cdots 0 \\ 0 & 0 & 0 & 0 \cdots 0 & 1 & 0 & 0 \cdots 0 \\ 0 & 0 & 0 & 0 \cdots 0 & 0 & 1 & 0 \cdots 0 \end{bmatrix}^T \\ \begin{array}{cccccccc} x & y & \theta & & m_{i,x} & m_{i,y} & \end{array} \end{array} * H_t^i$$

Extended Kalman-Filter

$x_k^{pre} = f(x_k^{est}, u_k)$

$P_k^{pre} = F_k P_{k-1}^{est} F_k^T + Q$

$K_k = P_k^{pre} H^T \left(H P_k^{pre} H^T + R\right)^{-1}$

$x_k^{est} = x_k^{pre} + K_k\left(z_k - H x_k^{pre}\right)$

$P_k^{est} = (I - K_k H) P_k^{pre}$

, in which $H_t^i = \frac{1}{q}\begin{bmatrix} -\sqrt{q}\delta_x & -\sqrt{q}\delta_y & 0 & \sqrt{q}\delta_x & \sqrt{q}\delta_y \\ \delta_y & -\delta_x & -q & -\delta_y & \delta_x \end{bmatrix}$, $\delta = \begin{bmatrix} m_{i,x} - x \\ m_{i,y} - y \end{bmatrix}$, $q = \delta^T \delta$

Introduction to Particle Filter

- EKF-SLAM assumes the probability distribution of robot pose and landmarks to be Gaussian, which leads to the following drawbacks:
  - Gaussian distribution can not express the robot pose properly.
  - The time complexity of estimating the covariance matrix for pose and landmarks is ($O(K^2)$), which is time-consuming even when only observing few landmarks.

    (K: number of landmarks)
- Particle filter utilizes **importance sampling** to approximate arbitrary distribution, which can express the robot pose more precisely.
- Furthermore, the time complexity of posterior estimation can be decreased to **O(MlogK)** by disentangling the estimation process of pose and map.

Sampling Process

- In statistical modeling and inference, there are many complex problems that the closed-form descriptions of $\boldsymbol{P}(\boldsymbol{X})$ can not be obtained.
- One can define a function $f(X)$ that computes $P(X)$ up to a normalizing constant:

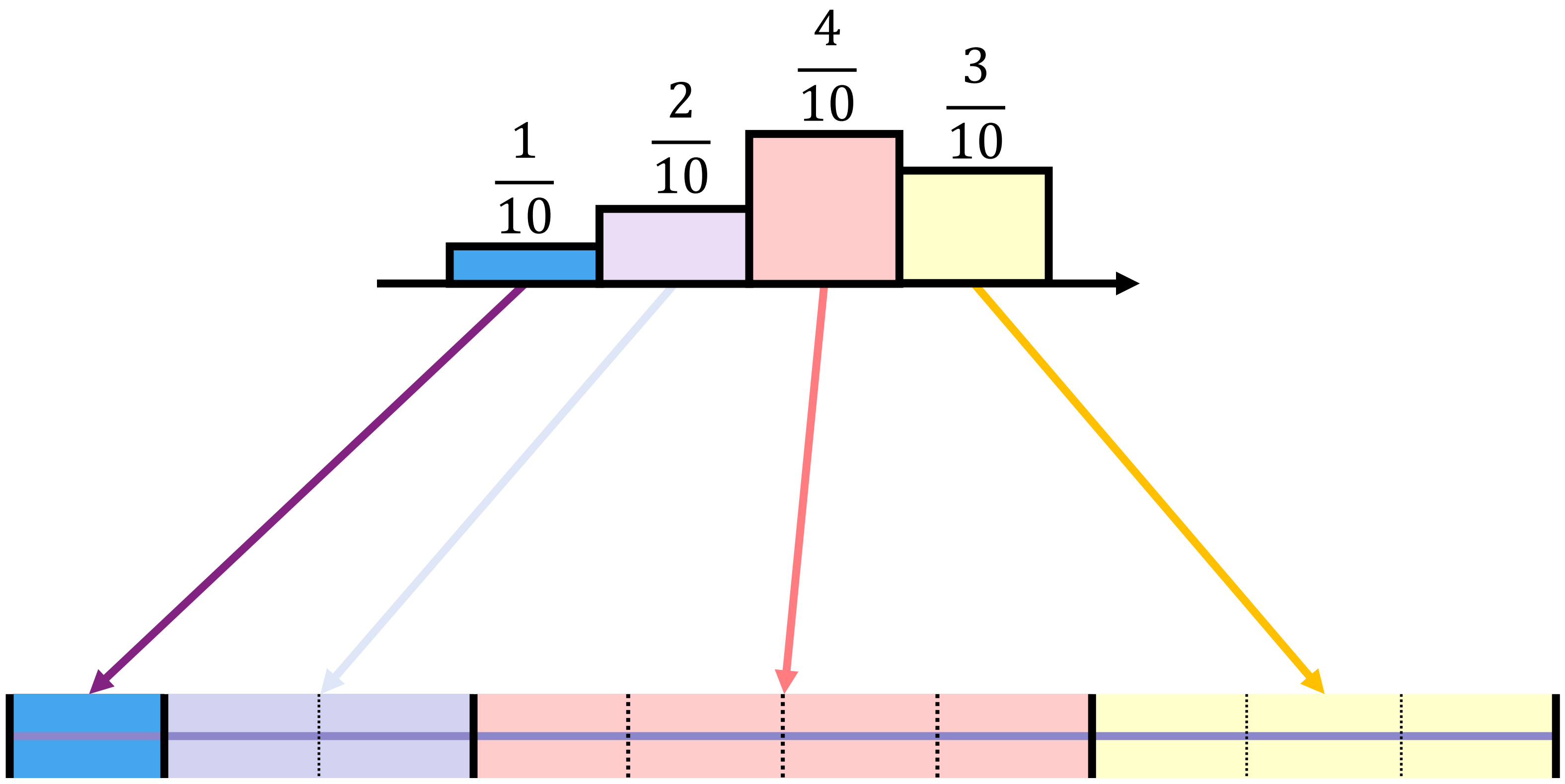
$$p(X) = \frac{f(X)}{Z}$$

where $z = \int_{x \in s} f(x)dx$ can not be computed because $f(X)$ is too complex, or because the state space $\boldsymbol{S}$ is too large to compute the integral.

- Statistical sampling and simulation techniques are used for getting fair samples from target probability distributions.

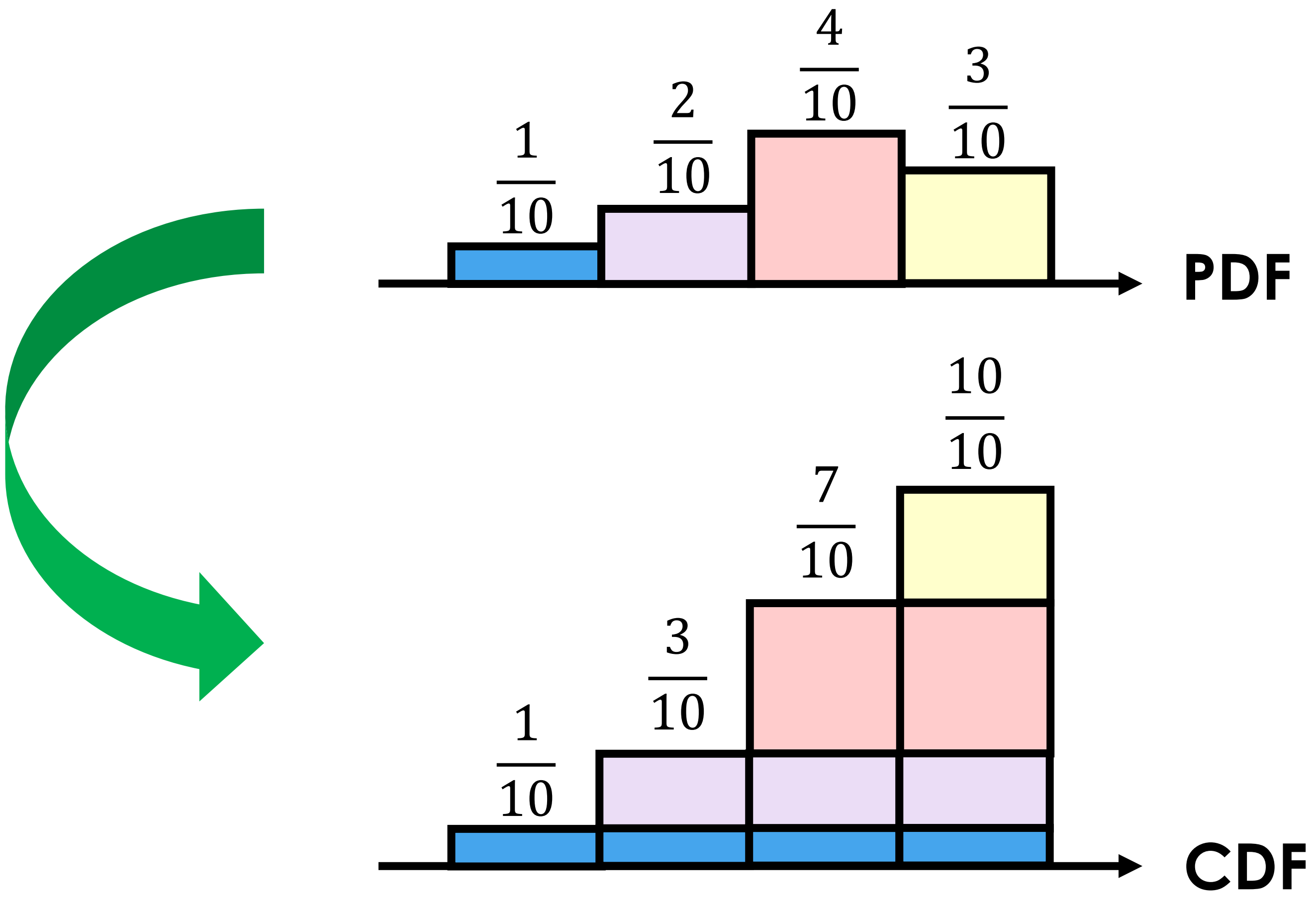
Basic Sampling

- Sampling from Probability Distribution Figure (PDF)



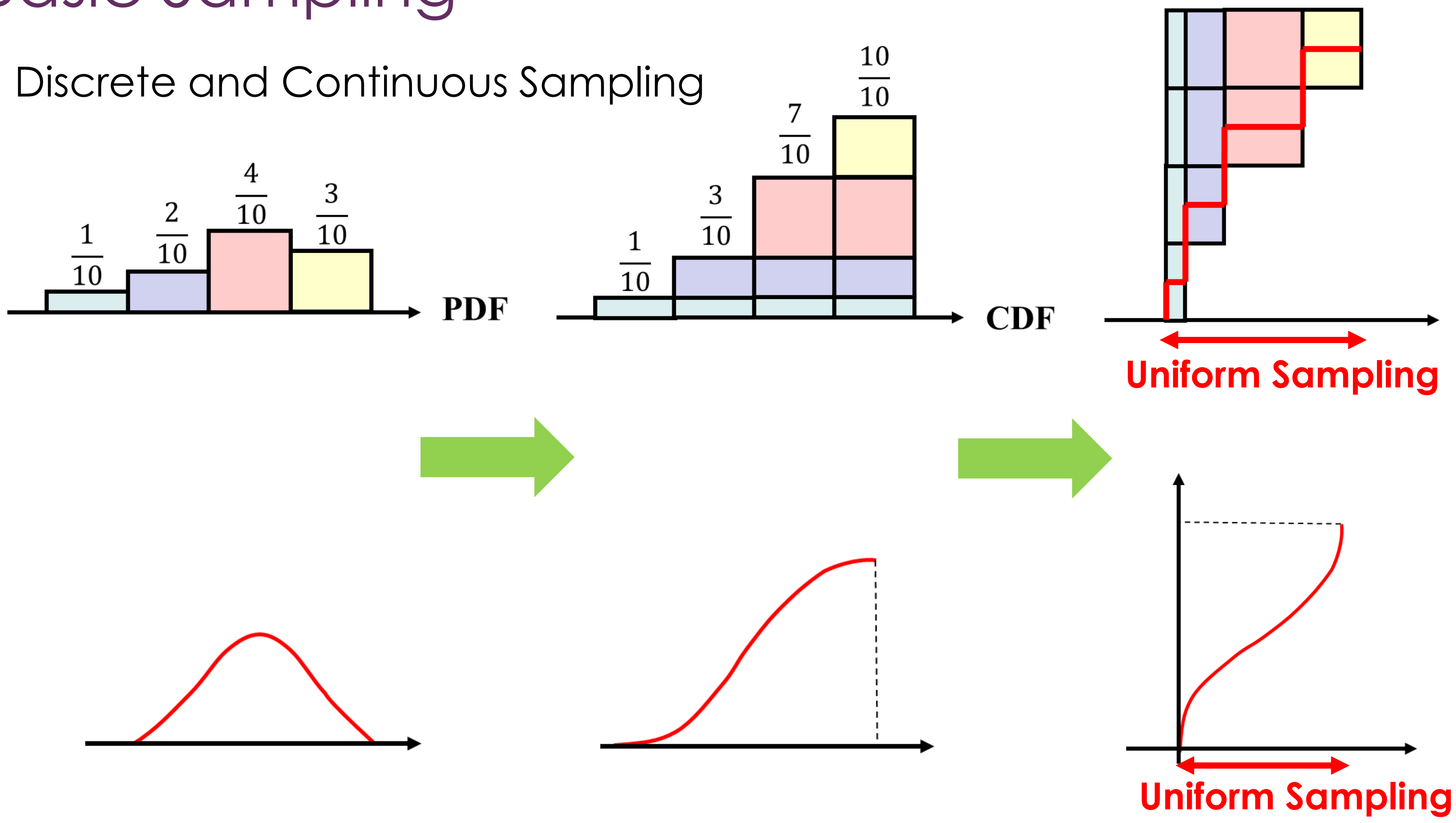
Basic Sampling

- From Probability Distribution Figure (PDF) to Cumulated Distribution Figure (CDF)



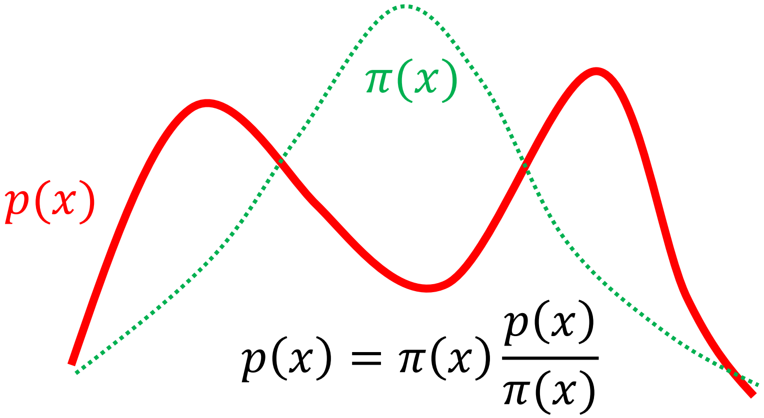
Basic Sampling

- Discrete and Continuous Sampling



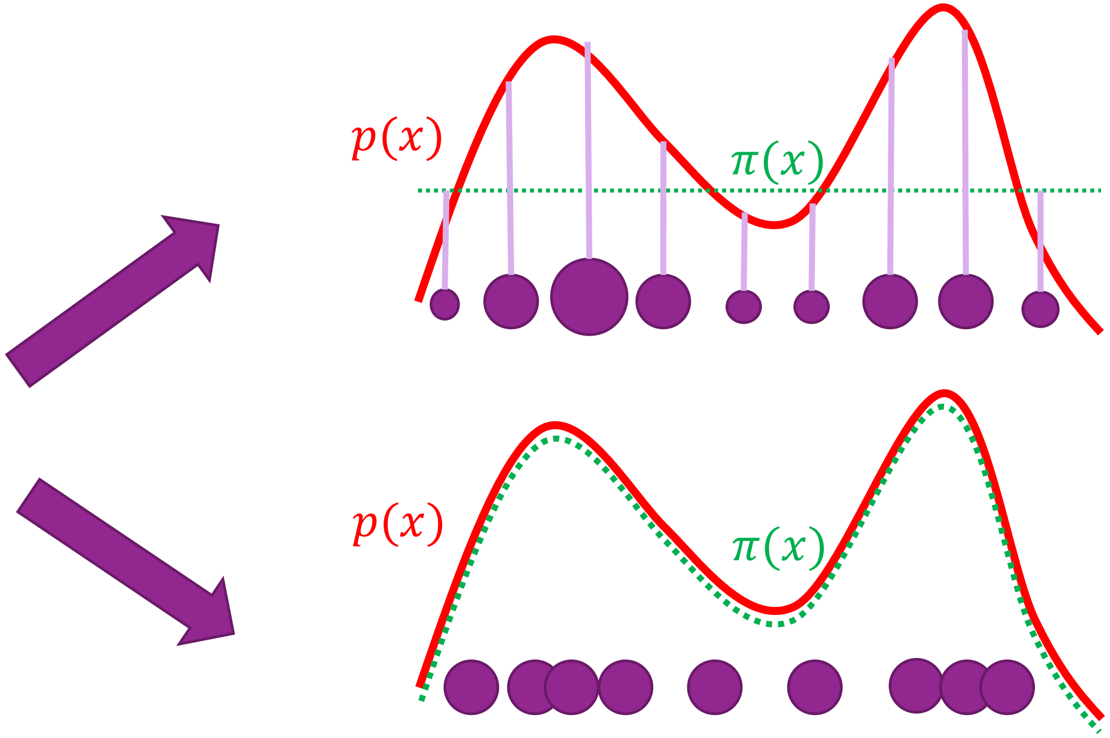
Importance Sampling

- Important sampling adopts discrete multinomial to approximate arbitrary distribution. More sampling particles will have more accurate approximation.



$$p(x) = \pi(x)\frac{p(x)}{\pi(x)}$$

1. $Sampling\ x_i\ from\ \pi(x)$
2. $Calculate\ w_i = \frac{p(x_i)}{\pi(x_i)}$
3. $Sampling\ x\ from\ mul(x_i, w_i)$



Sequential Importance Sampling (SIS)

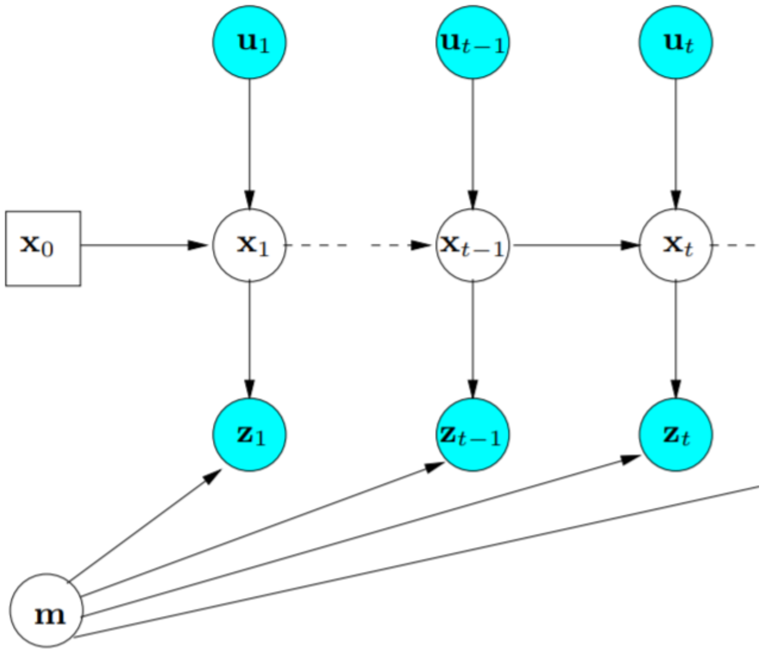
- Consider the localization problem, we utilize several particles to represent the approximation of pose distribution.

- In importance sampling, each particle have its own pose and weighting. The weighting is the division of source distribution and target distribution:

$$w_t^{(i)} = \frac{p(x_{1:t}^{(i)} | z_{1:t-1}, u_{1:t})}{\pi(x_{1:t}^{(i)} | z_{1:t-1}, u_{1:t})}$$

- According to the graphical model, we have

$$w_t^{(i)} = \frac{p\left(x_t^{(i)} \middle| x_{1:t-1}^{(i)}, z_{1:t-1}, u_{1:t}\right)}{\pi(x_t^{(i)} | x_{1:t-1}^{(i)}, z_{1:t-1}, u_{1:t})} \cdot \frac{p\left(x_{1:t-1}^{(i)} \middle| z_{1:t-2}, u_{1:t-1}\right)}{\pi\left(x_{1:t-1}^{(i)} \middle| z_{1:t-2}, u_{1:t-1}\right)}$$



$$w_t^{(i)} = \frac{p(x_{1:t}^{(i)}|z_{1:t-1}, u_{1:t})}{\pi(x_{1:t}^{(i)}|z_{1:t-1}, u_{1:t})}$$

Sequential Importance Sampling (SIS)

$$w_t^{(i)} = \frac{p\left(x_t^{(i)}\middle|x_{1:t-1}^{(i)}, z_{1:t-1}, u_{1:t}\right)}{\pi(x_t^{(i)}|x_{1:t-1}^{(i)}, z_{1:t-1}, u_{1:t})} \cdot \frac{p\left(x_{1:t-1}^{(i)}\middle|z_{1:t-2}, u_{1:t-1}\right)}{\pi\left(x_{1:t-1}^{(i)}\middle|z_{1:t-2}, u_{1:t-1}\right)}$$

- Apply the Bayes theorem, we can get

$$w_t^{(i)} = \frac{\eta p\left(z_{t-1}\middle|x_{1:t}^{(i)}, u_{1:t}\right) p\left(x_t^{(i)}\middle|x_{t-1}^{(i)}, u_t\right)}{\pi(x_t^{(i)}|x_{1:t-1}^{(i)}, z_{1:t-1}, u_{1:t})} \cdot \underbrace{\frac{p\left(x_{1:t-1}^{(i)}\middle|z_{1:t-2}, u_{1:t-1}\right)}{\pi\left(x_{1:t-1}^{(i)}\middle|z_{1:t-2}, u_{1:t-1}\right)}}_{w_{t-1}^{(i)}}$$

$$\propto \frac{p\left(z_{t-1}\middle|m_{t-1}, x_t^{(i)}\right) p\left(x_t^{(i)}\middle|x_{t-1}^{(i)}, u_{t-1}\right)}{\pi\left(x_t\middle|x_{1:t-1}^{(i)}, z_{1:t-1}, u_{1:t-1}\right)} \cdot w_{t-1}^{(i)}$$

, in which $\eta = \dfrac{1}{p(z_{t-1}|z_{1:t-2}, u_{1:t})}$

Sequential Importance Sampling (SIS)

- Now, we select the distribution of last timestep as the source distribution:

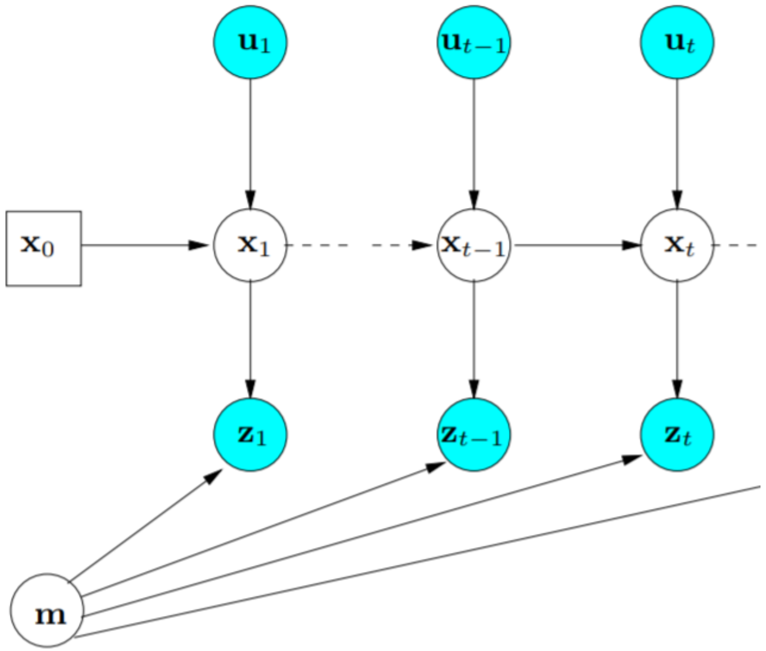
$$\pi\left(x_t \middle| x_{1:t-1}^{(i)}, z_{1:t-1}, u_{1:t}\right) = p(x_t^{(i)} | x_{t-1}^{(i)}, u_t)$$

$$\boxed{w_t^{(i)} = \frac{\eta p\left(z_{t-1} \middle| m_{t-1}, x_t^{(i)}\right) p\left(x_t^{(i)} \middle| x_{t-1}^{(i)}, u_{t-1}\right)}{\pi\left(x_t \middle| x_{1:t-1}^{(i)}, z_{1:t-1}, u_{1:t-1}\right)} \cdot w_{t-1}^{(i)}}$$

- We can get the update weighting:

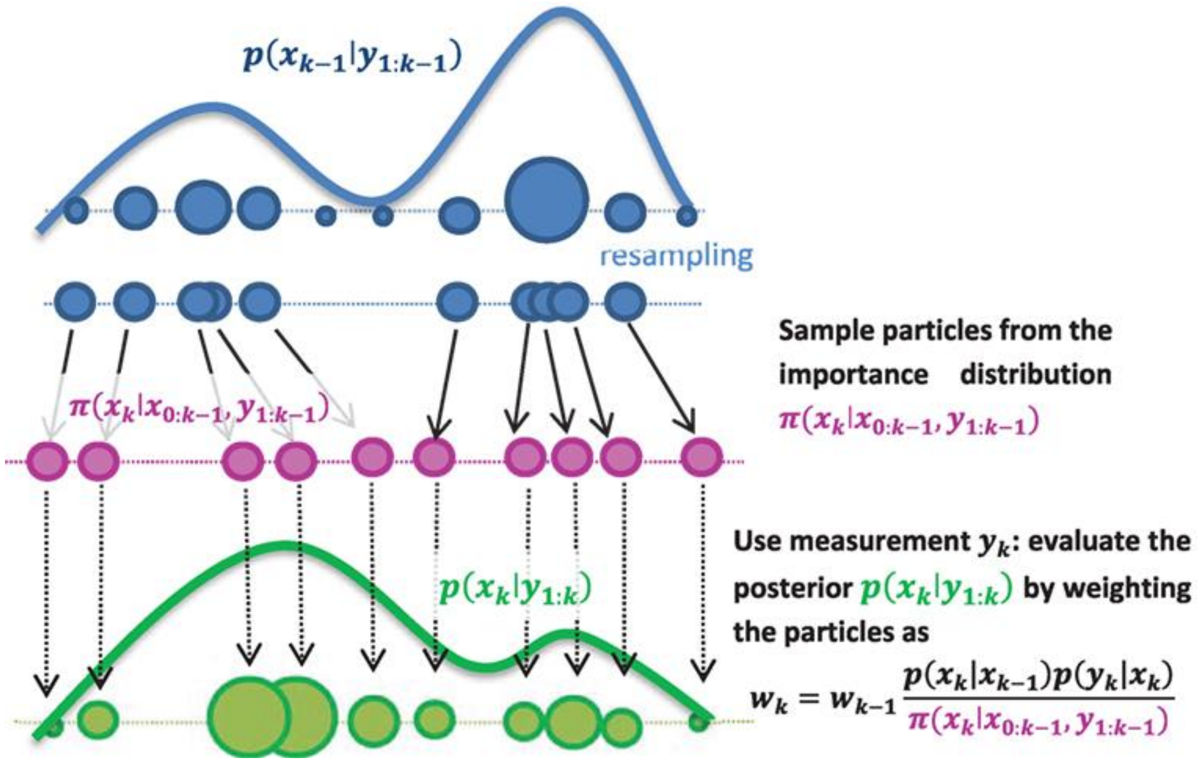
$$w_t^{(i)} = w_{t-1}^{(i)} \frac{\eta p\left(z_{t-1} \middle| m_{t-1}, x_t^{(i)}\right) p\left(x_t^{(i)} \middle| x_{t-1}^{(i)}, u_t\right)}{p(x_t^{(i)} | x_{t-1}^{(i)}, u_t)}$$

$$\propto w_{t-1}^{(i)} \cdot p(z_{t-1} | m_{t-1}, x_t^{(i)})$$

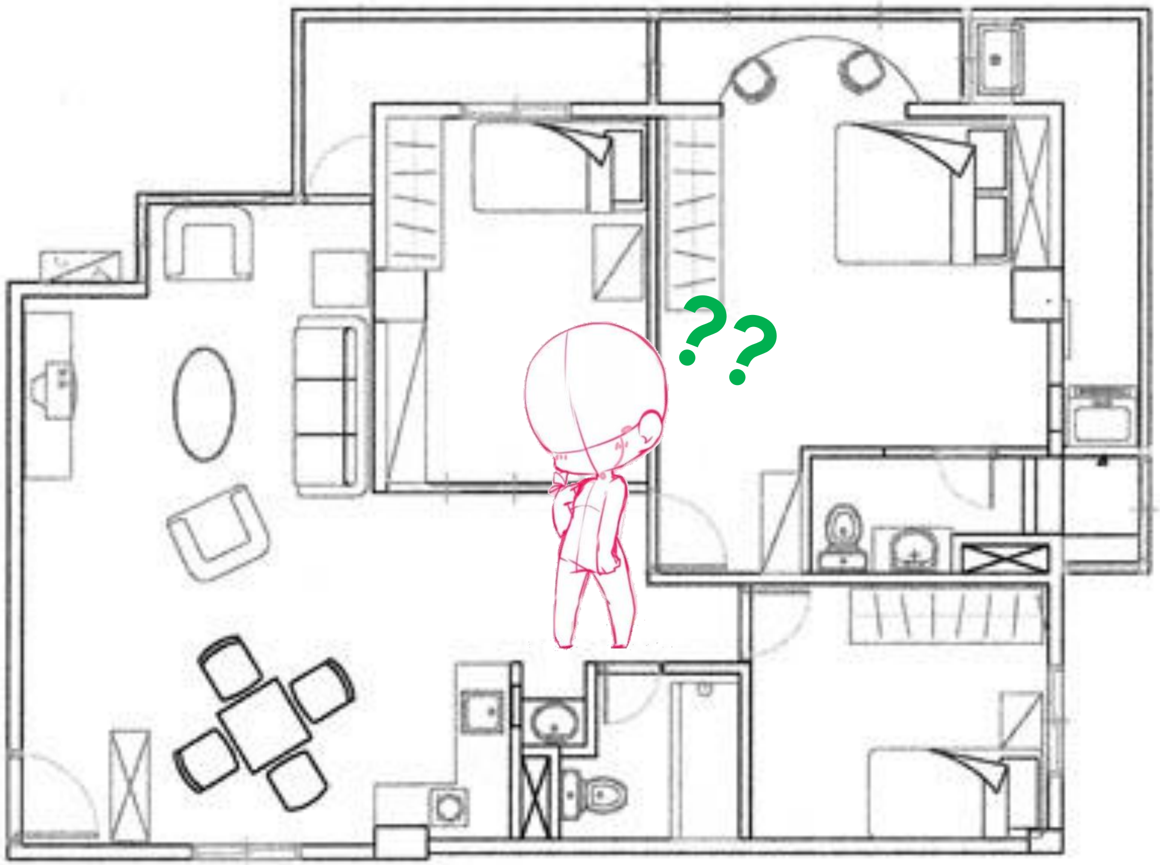


Sequential Importance Resampling (SIR)

- After several steps, the weightings of most particles in SIS particle filter will decrease to close to zero.
- To avoid this problem, we can utilize the resampling process:



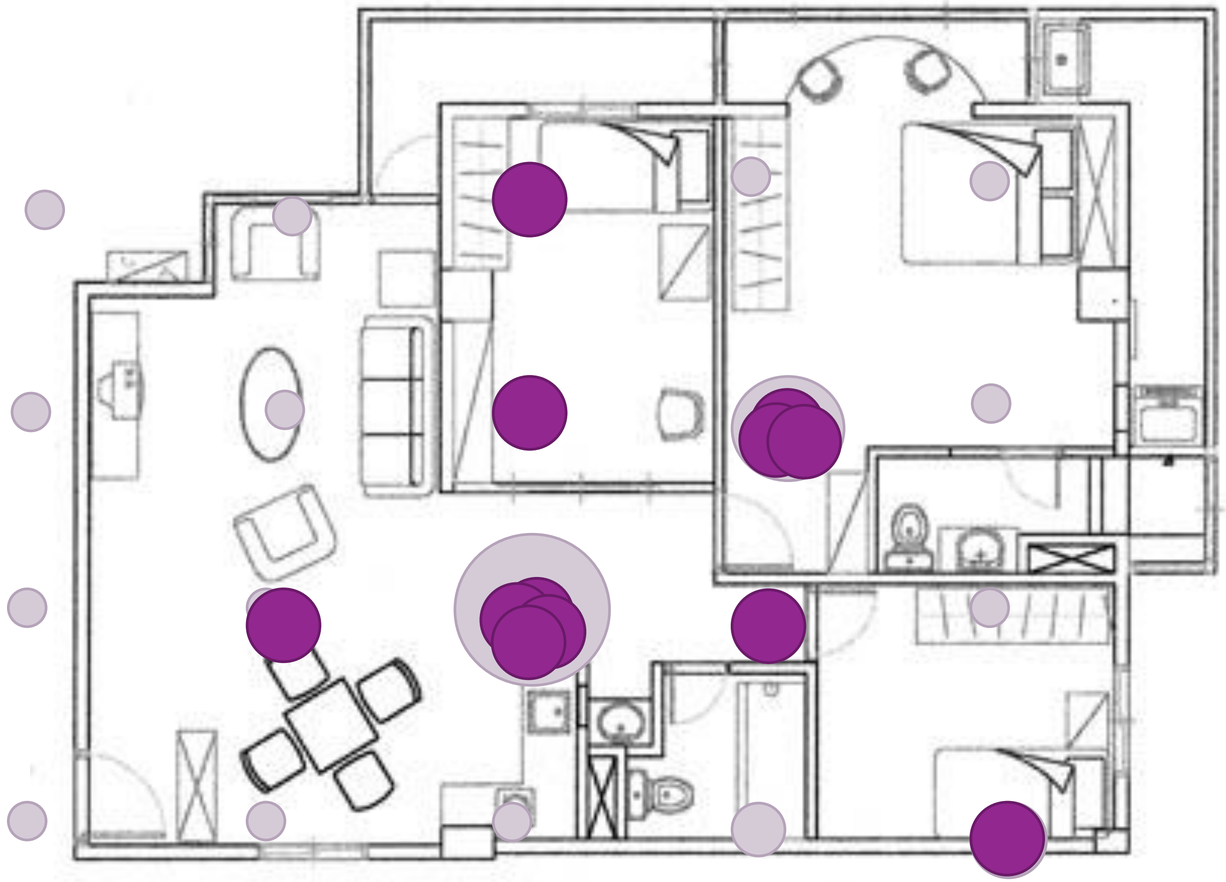
Monte-Carlo Localization Example



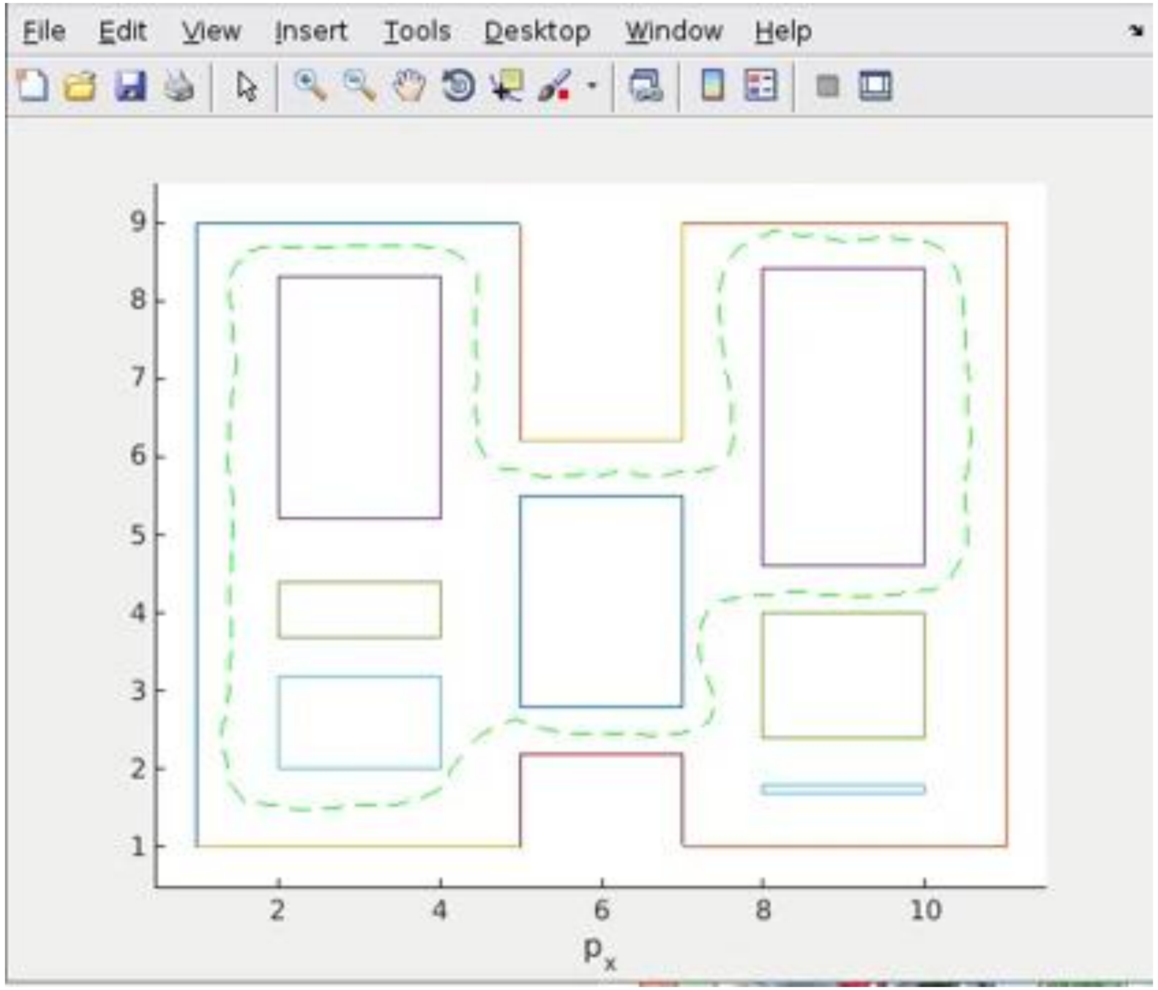
Monte-Carlo Localization Example

Monte-Carlo Localization Example

Monte-Carlo Localization Example



Monte-Carlo Localization



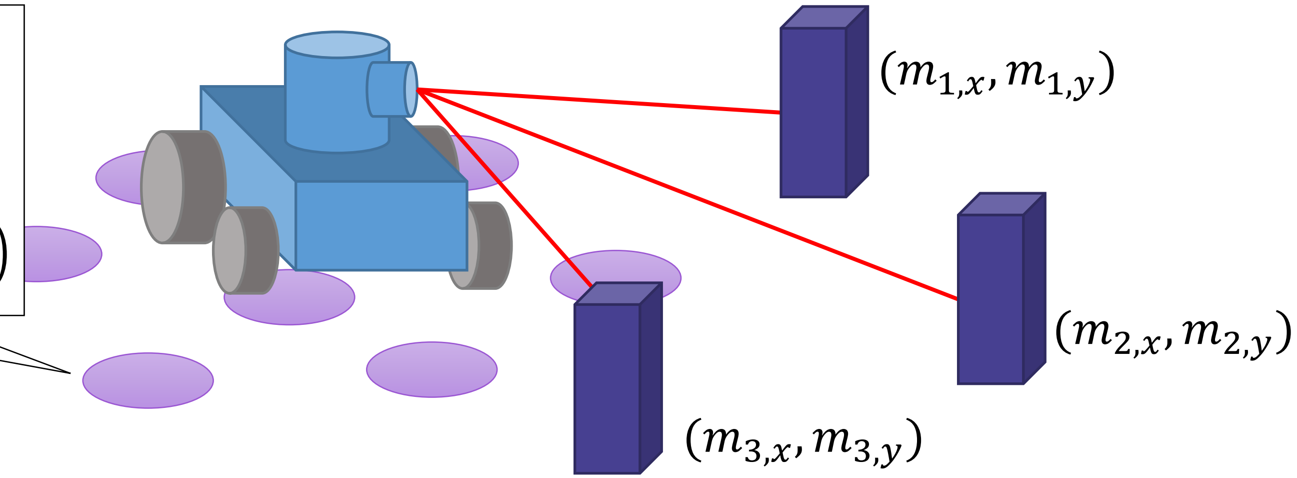
Fast-SLAM

- Now consider the full SLAM problem (localization and mapping), we can divide the full process to localization and mapping steps. This method is called Rao-Blackwellization.

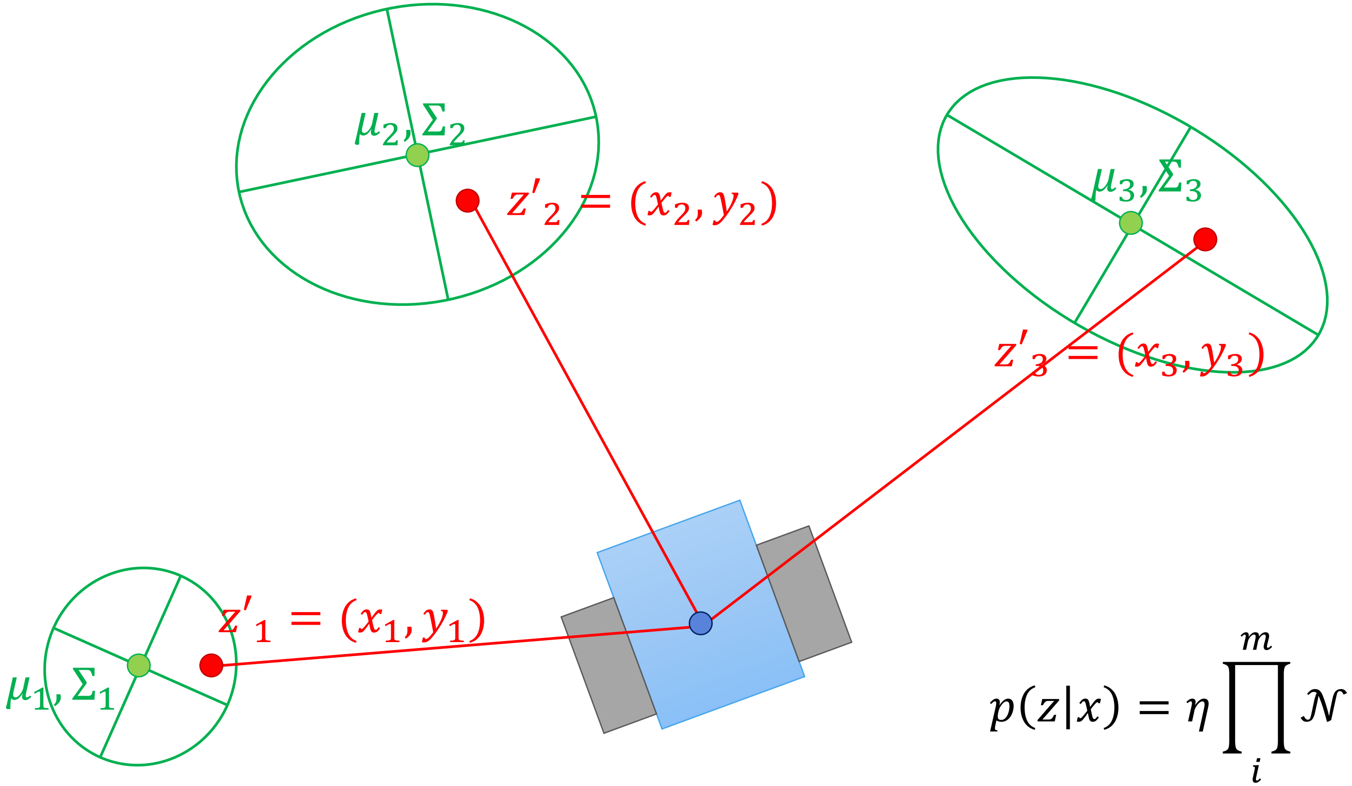
$$p(x_{1:t}, m_t | z_{1:t}, u_{1:t}) = p(x_{1:t} | z_{1:t}, u_{1:t}) p(m_t | x_{1:t}, z_{1:t})$$

- In Fast-SLAM, the robot pose is represented by the multivariate distribution of several weighted particles, and each particle adopts **K** extended Kalman filter to estimate the landmarks independently.

Particle Weights: $w^{(i)}$
Robot Pose: $(x\ y\ \theta)^{(i)}$
Landmarks:
$\left(\mu_1^{(i)}, \Sigma_1^{(i)}\right), \left(\mu_2^{(i)}, \Sigma_2^{(i)}\right), \left(\mu_3^{(i)}, \Sigma_3^{(i)}\right)$



Likelihood of Measurement



$$p(z|x) = \eta \prod_{i}^{m} \mathcal{N}(z'_i; \mu_i, \Sigma_i)$$

Fast-SLAM

- Steps of Fast-SLAM

1. Predict the next pose $x_t^{(i)}$ by motion model.

$$x_t^{(i)} \sim p(x_t^{(i)} | x_{t-1}^{(i)}, u_{t-1})$$

2. Update the distribution of each landmark $(\mu_{j,t}^{(i)}, \Sigma_{j,t}^{(i)})$ via measurement z_k.

$$Q = H\Sigma_{j,t-1}^{(i)}H^T + R, \qquad K_t = \Sigma_{j,t-1}^{(i)}H^T Q^{-1}$$

$$\mu_{j,t}^{(i)} = \mu_{j,t-1}^{(i)} + K_k\left(z_k - h(\mu_{j,t-1}^{(i)}, x_t^{(i)})\right)$$

$$\Sigma_{j,t}^{(i)} = (I - K_t H)\Sigma_{j,t-1}^{(i)}$$

3. Update the importance weight of particles.

$$w^{(i)} \sim |2\pi Q|^{-\frac{1}{2}} \exp\{-\frac{1}{2}\left(z_k - h\left(\mu_{j,t-1}^{(i)}, x_t^{(i)}\right)\right)^T Q^{-1}\left(z_k - h(\mu_{j,t-1}^{(i)}, x_t^{(i)})\right)\}$$

4. Resampling.

Fast SLAM

- Measure of how well the target distribution is approximated by samples drawn from the proposal.

$$N_{eff} = \frac{1}{\sum_i \left(w_t^{(i)}\right)^2}$$

- N_{eff} denotes the inverse variance of the normalized particle weights. For equal weights, the results is the number of the particles. And the sample approximation is close to the target.

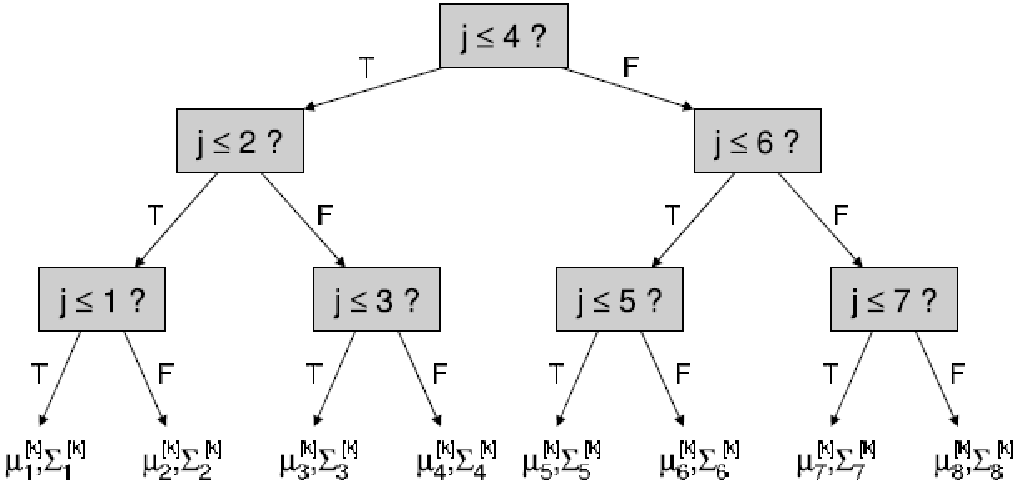
$$N_{eff}^* = \frac{1}{\sum_i \frac{1}{N^2}} = \frac{1}{N \frac{1}{N^2}} = N$$

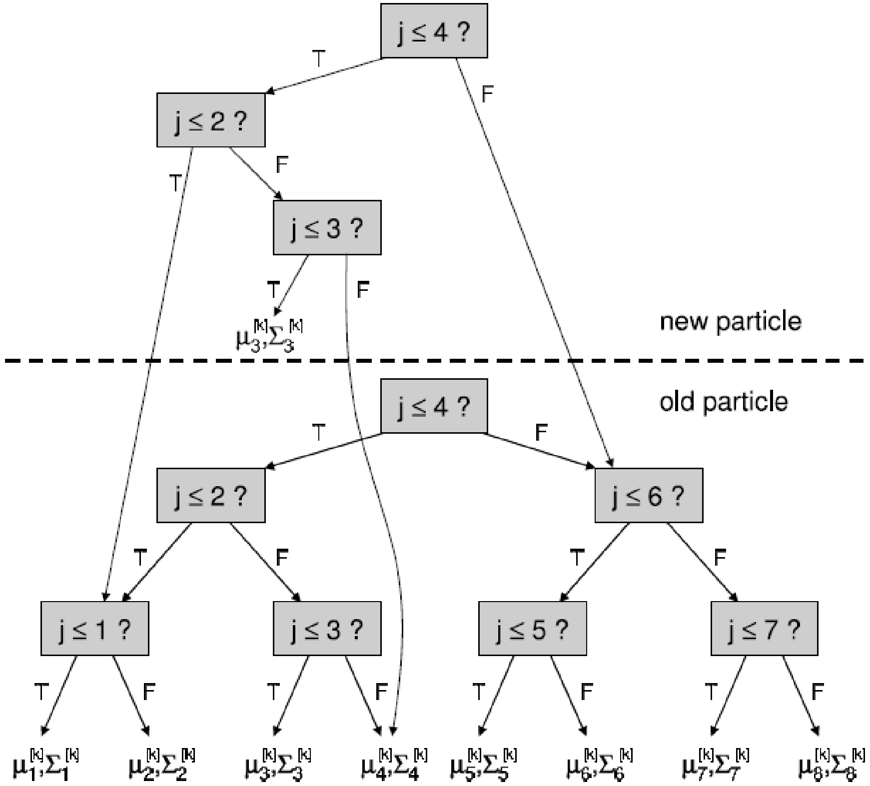
- If N_{eff} drops below a given threshold (usually set to half of the particles), we will resample the particle.

$$N_{eff} < \frac{N}{2}$$

Fast-SLAM

- Efficient implementation of Fast-SLAM. The basic idea is that the set of Gaussians in each particle is represented by a balanced binary tree.





Fast-SLAM Demo