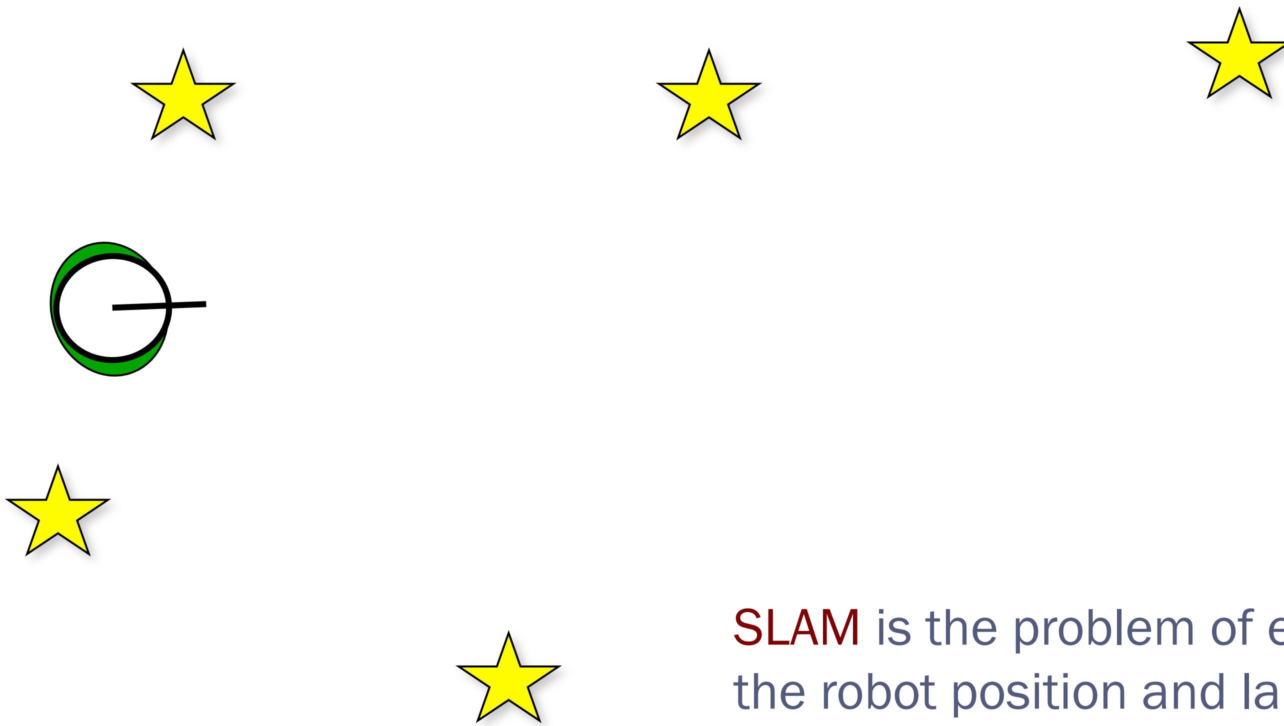
AUSTRALIAN CENTRE FOR ROBOTIC VISION

Tutorial on SLAM

Viorela Ila

SLAM

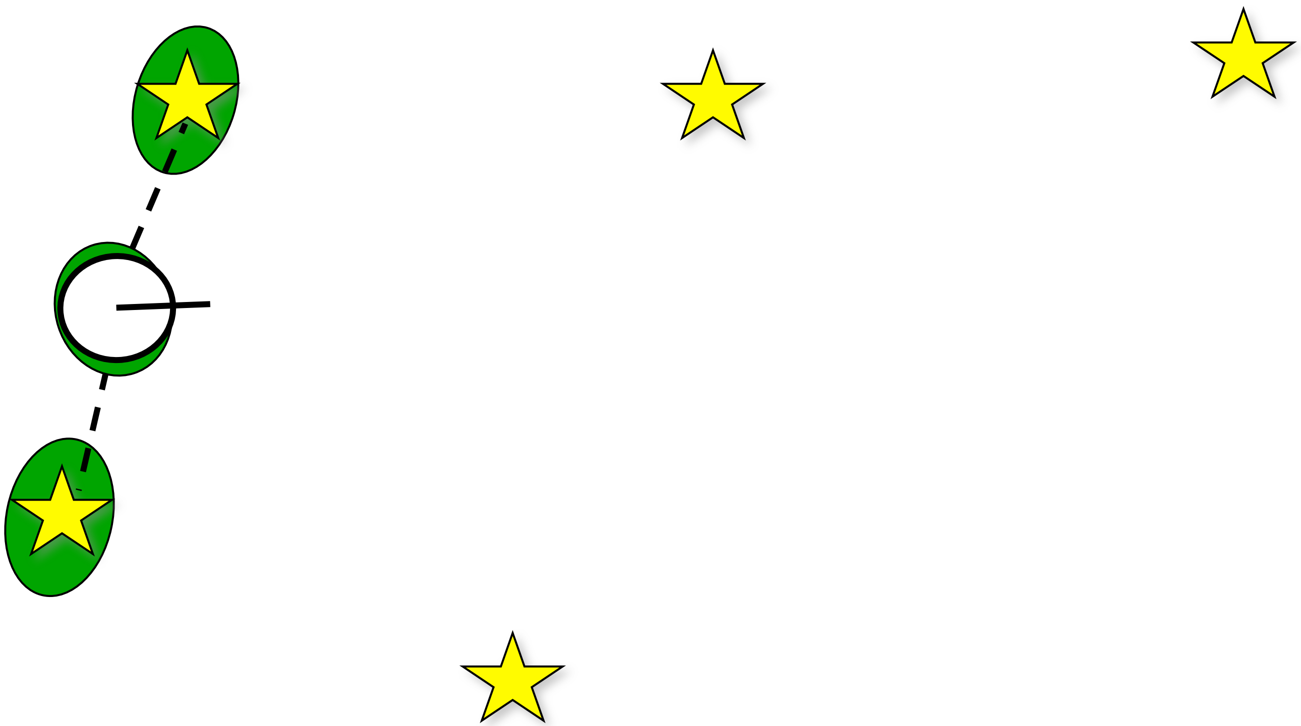
Starts from known position but unknown environment



SLAM is the problem of estimating the robot position and landmarks in the environment given the sensor data and control inputs.

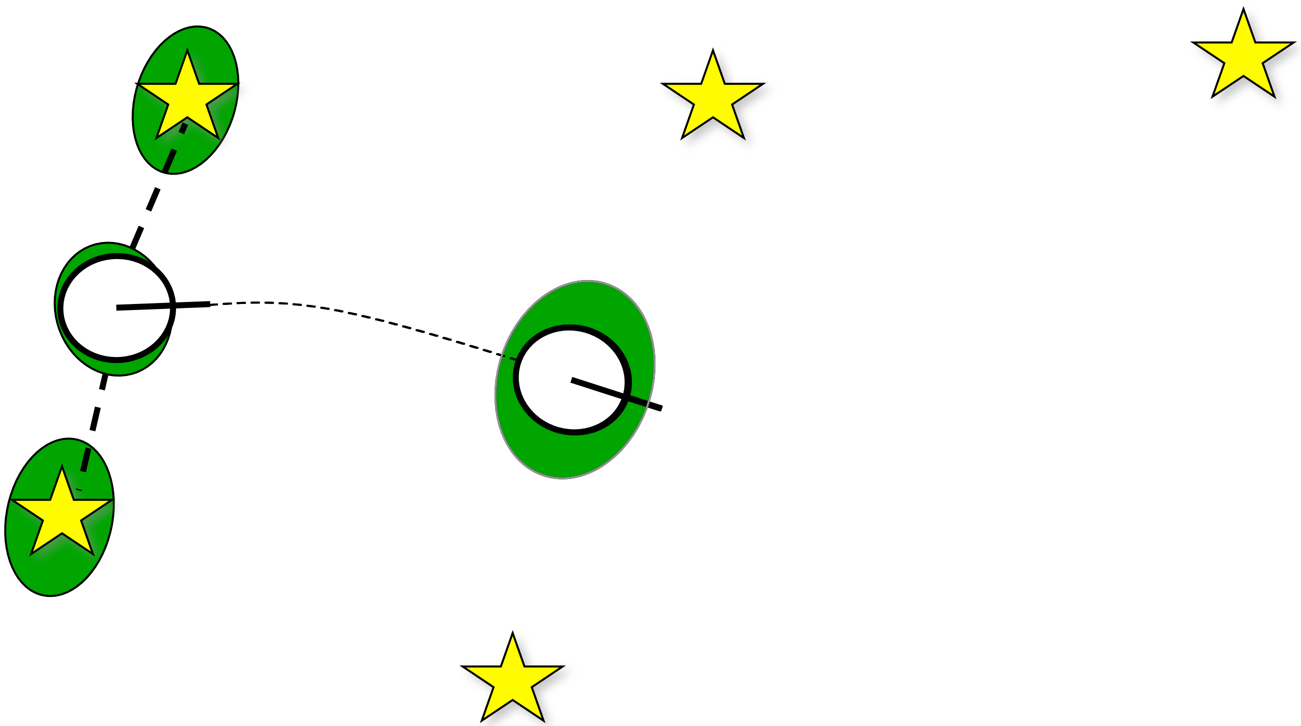
SLAM

Observes landmarks in the environment



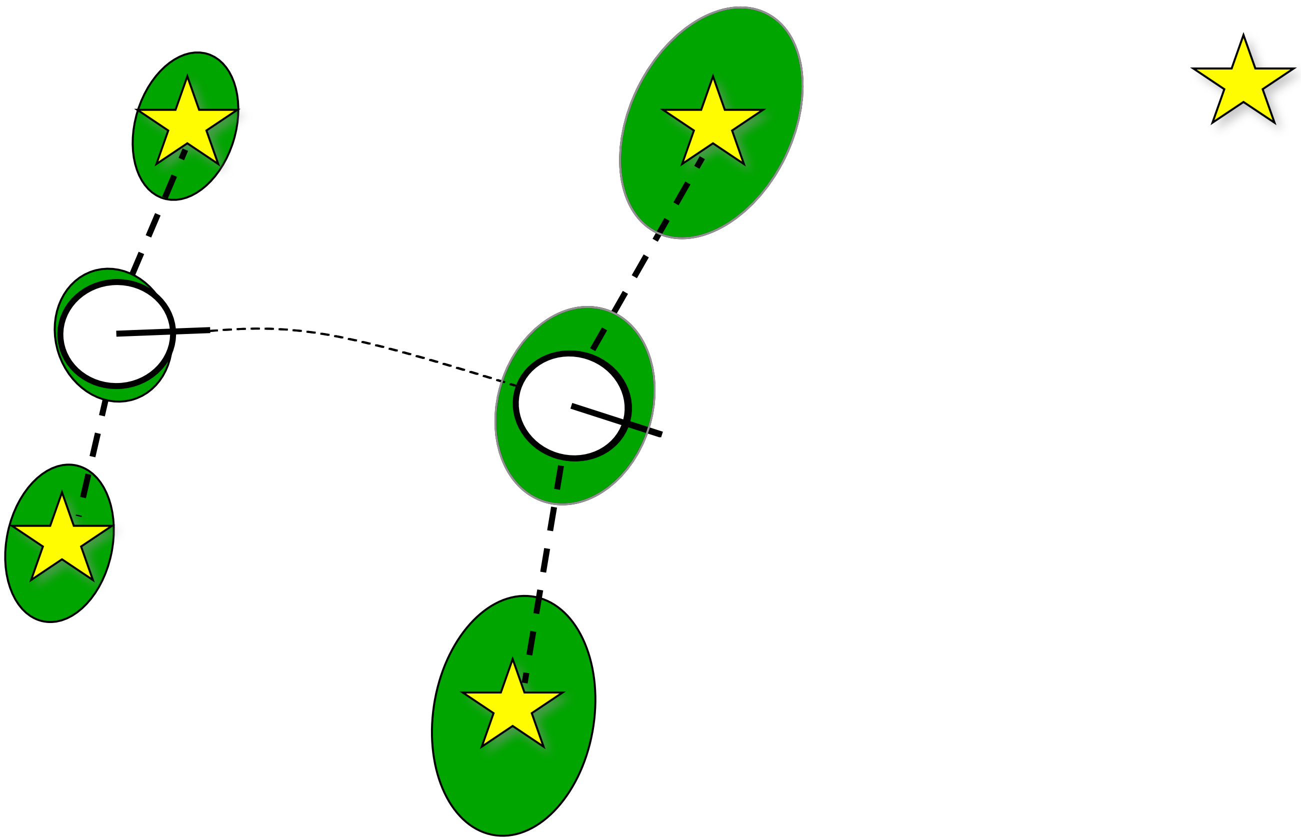
SLAM

Move in the environment.



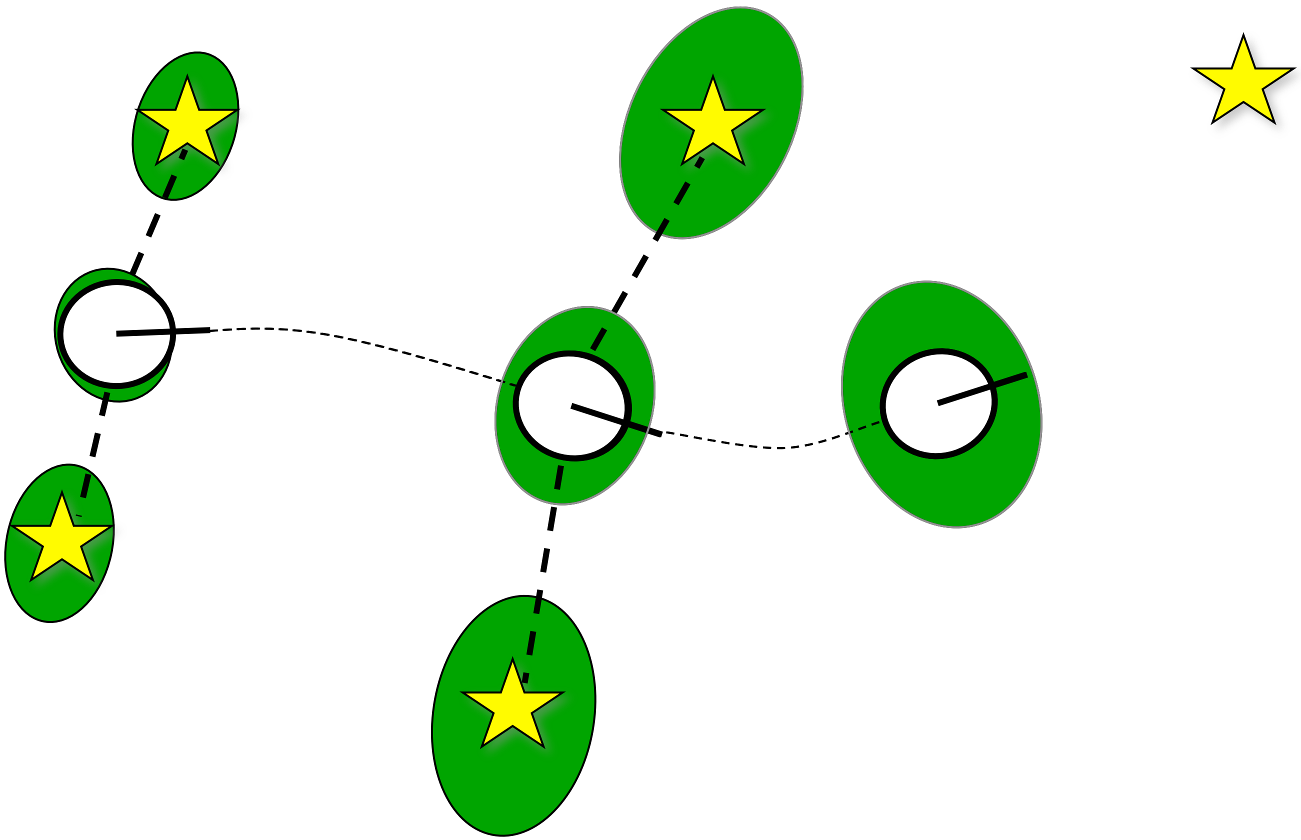
SLAM

Observes landmarks in the environment



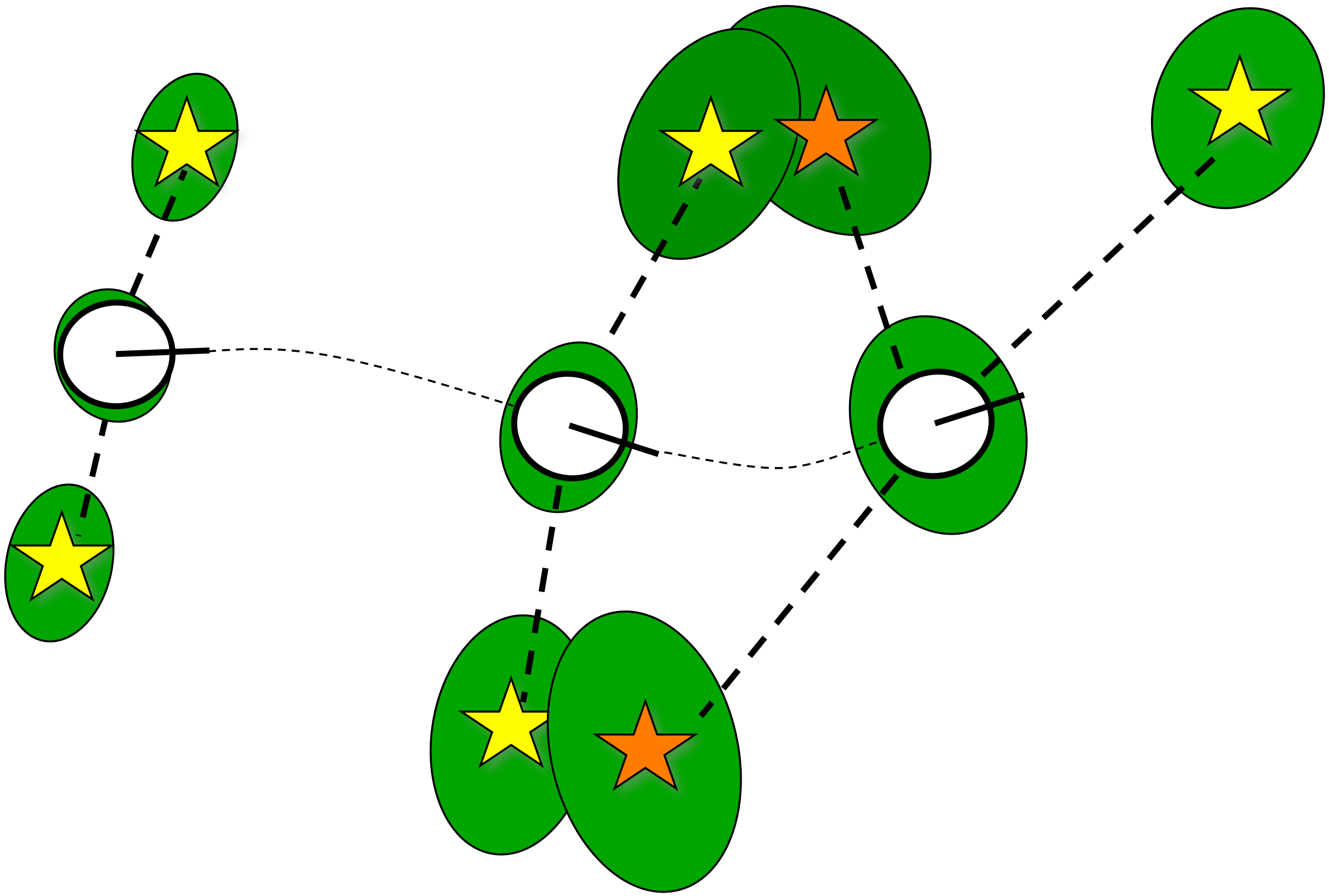
SLAM

Move again.



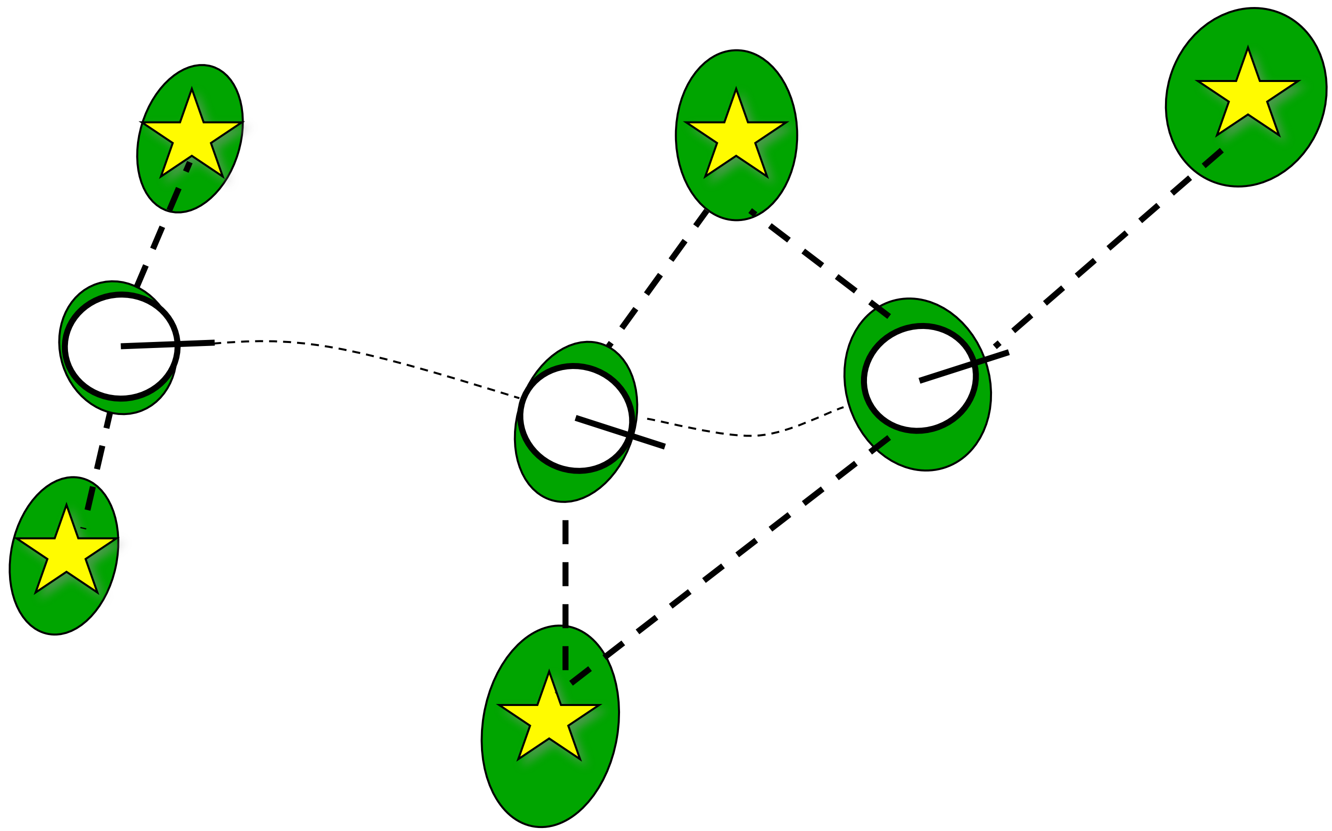
SLAM

Re-observe landmarks in the environment – Data association

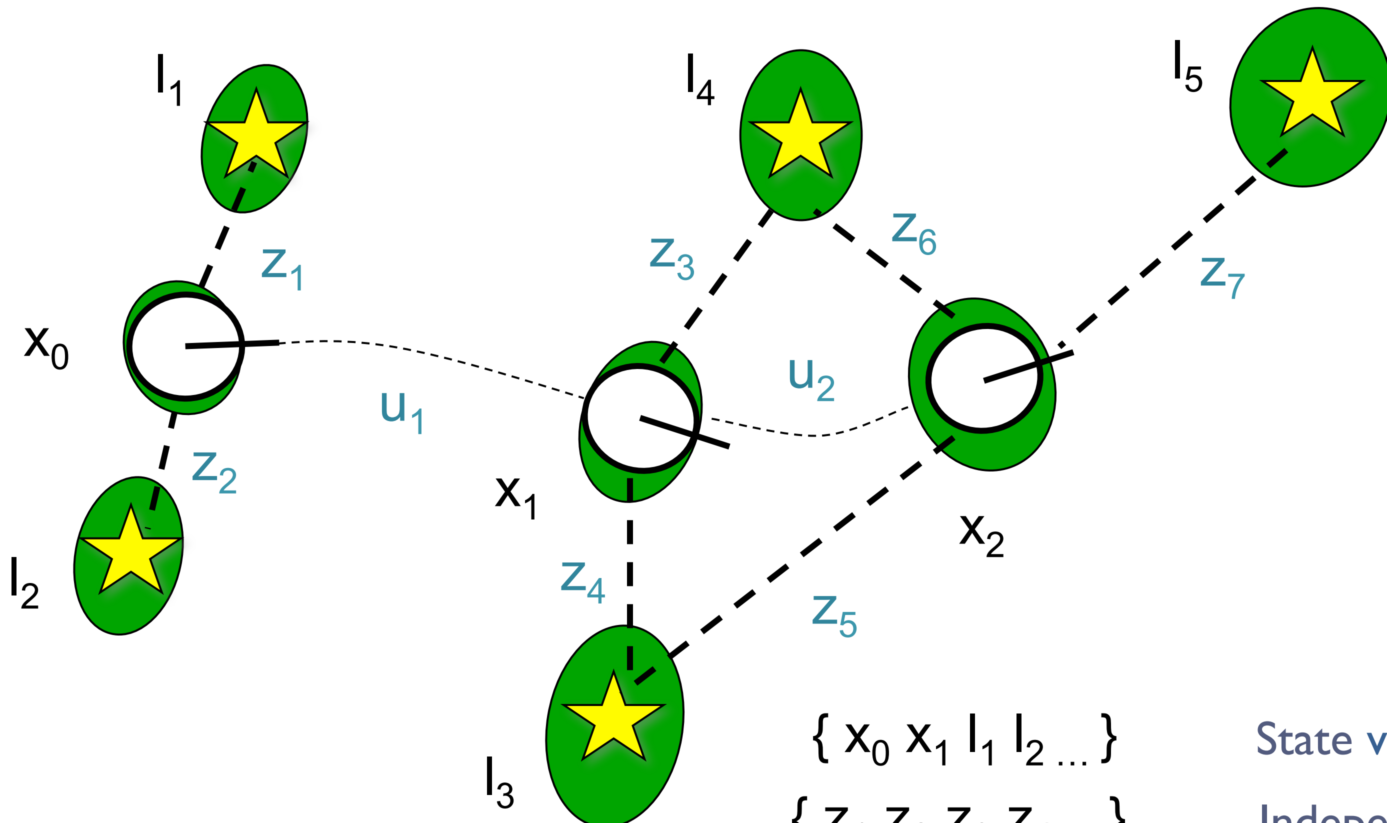


SLAM

Re-observe landmarks in the environment – Reduces the error



SLAM – variables and measurements

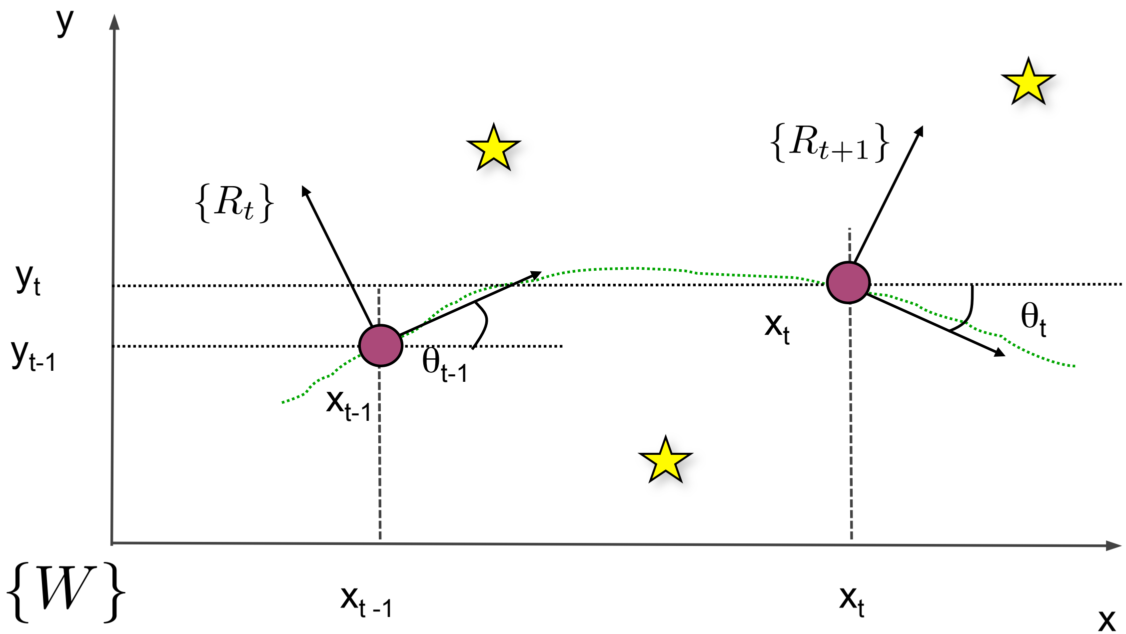


$\{ x_0\ x_1\ l_1\ l_2 \ldots \}$ State variables

$\{ z_1\ z_2\ z_3\ z_4 \ldots \}$ Independent data

$\{ u_1 \ldots \}$ Control data

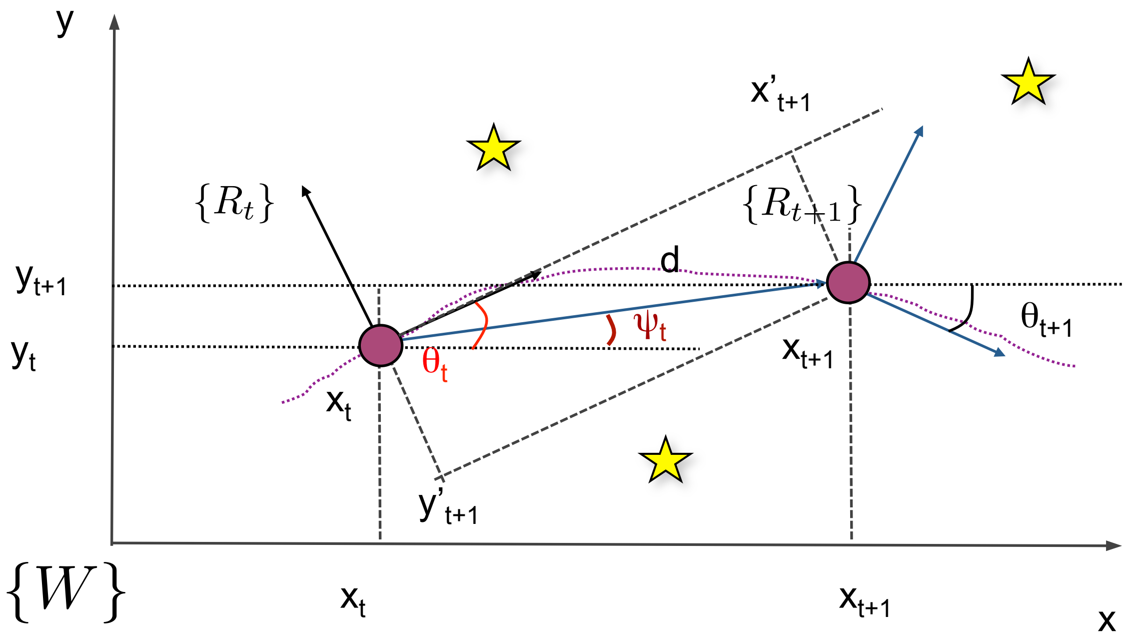
2D SLAM



$$^{\{W\}}\mathbf{x}_{t-1} = [x_{t-1}, y_{t-1}, \theta_{t-1}]^\top$$

$$^{\{W\}}\mathbf{x}_t = [x_t, y_t, \theta_t]^\top$$

Motion Model



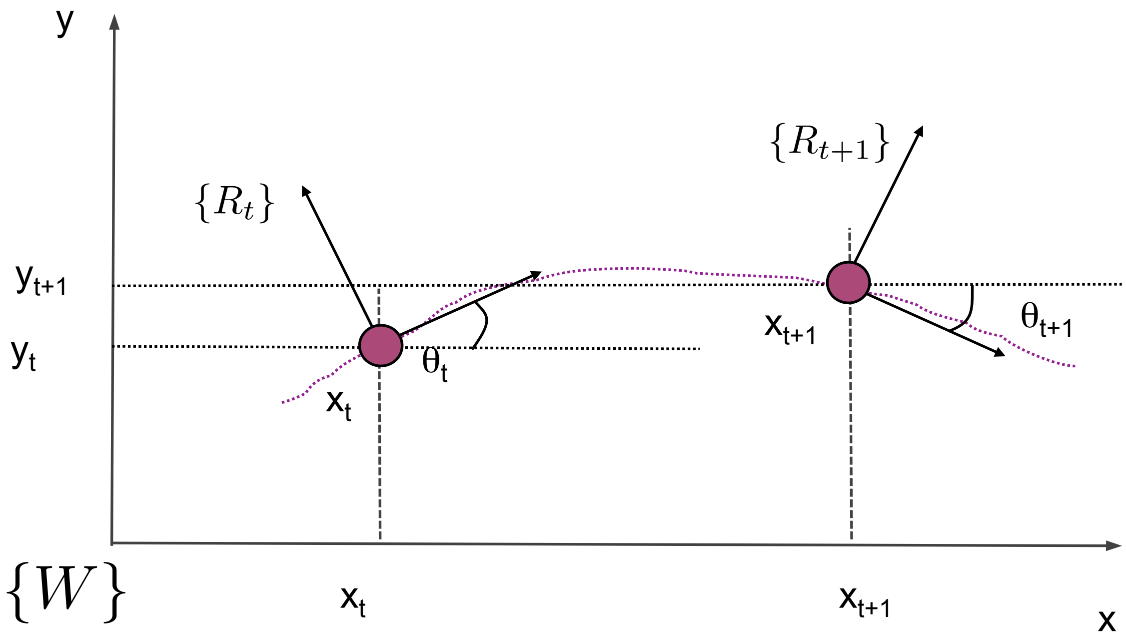
$$
\begin{aligned}
{}^{\{R_t\}}\Delta x &= d * \cos(\psi) \\
{}^{\{R_t\}}\Delta y &= d * \sin(\psi) \\
{}^{\{R_t\}}\Delta \theta &= \theta_{t+1} - \theta_t
\end{aligned}
$$

we don't need to calculate explicit d and ψ!

Motion model

$$
\begin{aligned}
x_{t+1} &= x_t + \Delta x \cos(\theta_t) - \Delta y \sin(\theta_t) \\
y_{t+1} &= y_t + \Delta x \sin(\theta_t) + \Delta y \cos(\theta_t) \\
\theta_{t+1} &= \theta_t + \Delta \theta
\end{aligned}
$$

Motion Model



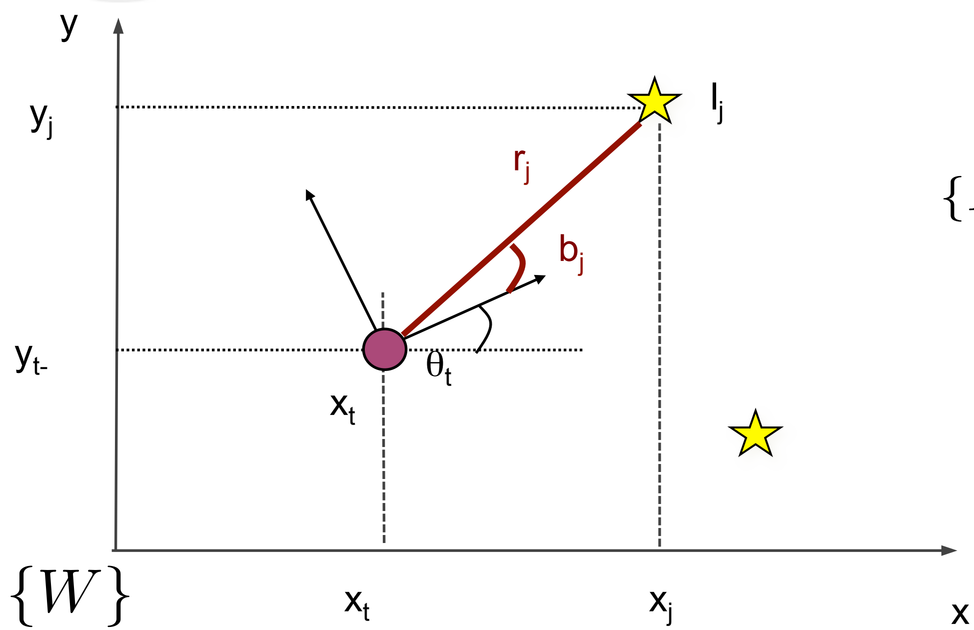
$$^{\{W\}}\mathbf{x}_{t+1} = [x_{t+1}, y_{t+1}, \theta_{t+1}]^\top$$

$$^{\{R_t\}}\mathbf{x}_{t+1} = [\Delta x, \Delta y, \Delta \theta]^\top$$

$$^{W}R_{R_t} = \begin{bmatrix} cos(o) & -sin(o) & 0 \\ sin(o) & cos(o) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\boxed{^{\{W\}}\mathbf{x}_{t+1} = ^{\{W\}}\mathbf{x}_t + ^{\{W\}}R_{\{R_t\}} \cdot ^{\{R_t\}}\mathbf{x}_{t+1}}$$

Observation Model



$$
{}^{\{R_{t+1}\}}l_j = \begin{bmatrix} {}^{\{R_{t+1}\}}x \\ {}^{\{R_{t+1}\}}y \end{bmatrix} = \begin{bmatrix} r\cos(b) \\ r\sin(b) \end{bmatrix}
$$

$$
\begin{aligned}
r_t^j &= \sqrt{(x_j - x_t)^2 + (y_j - y_t)^2} \\
b_t^j &= atan\left(\frac{y_j - y_t}{x_j - x_t}\right) - \theta_t
\end{aligned}
$$

$$
\boxed{{}^{\{W\}}l_j = \begin{bmatrix} {}^{\{W\}}x_{l_j} \\ {}^{\{W\}}y_{l_j} \end{bmatrix} = \begin{bmatrix} x_{t+1} \\ y_{t+1} \end{bmatrix} + {}^{\{W\}}R_{\{R_{t+1}\}} \begin{bmatrix} {}^{\{R_{t+1}\}}x_{l_j} \\ {}^{\{R_{t+1}\}}y_{l_j} \end{bmatrix}}
$$

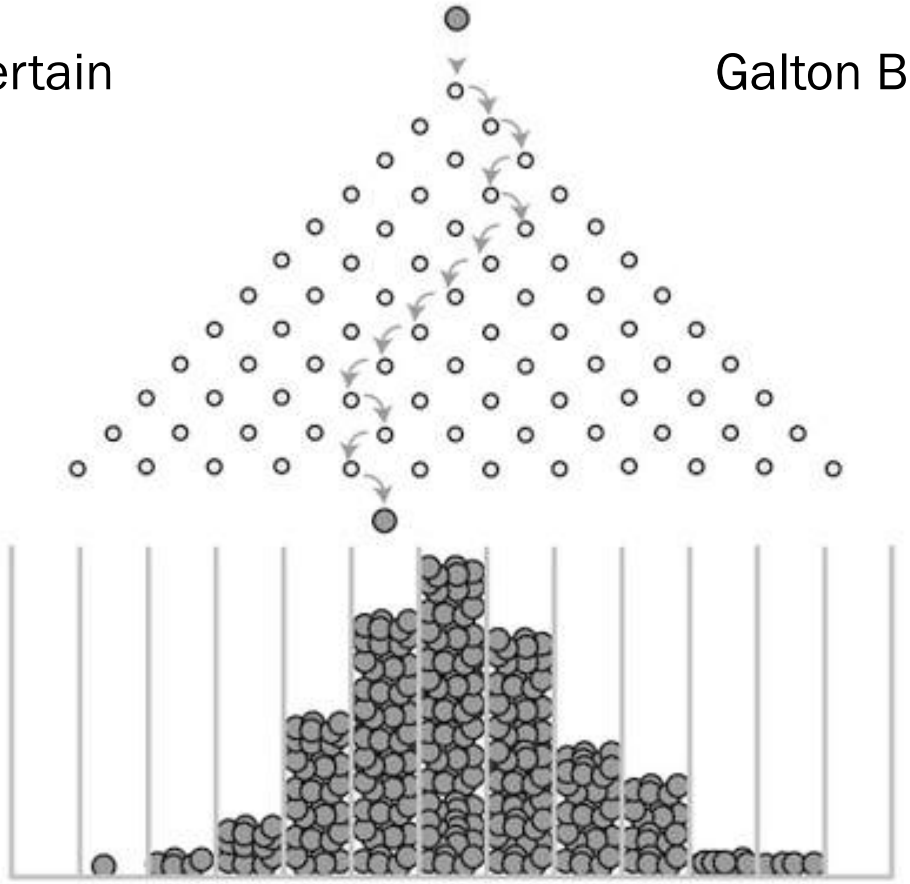
Challenges of Robot Navigation

- Most mobile robots work in an **unstructured, uncertain environment.**
- Absolute position information (e.g. via GPS or other global localization systems such as VICON) is often unavailable, inaccurate, or insufficient
- **Uncertainties** are present in sensors readings, motion as well as in the model.
  - Sensor noise
  - Sensor aliasing
  - Effecter/Actuator noise
  - Position integration
  - Simple models

Probability and Gaussian

Example of un uncertain physical process.

Galton Board

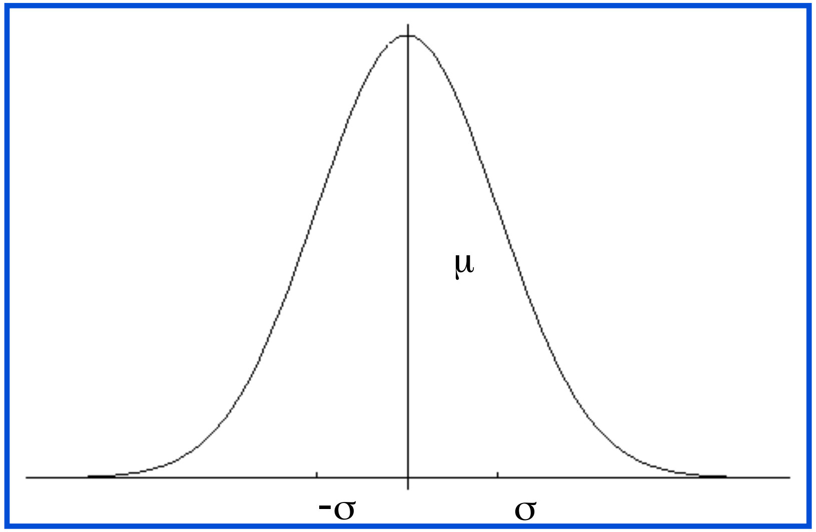


Gaussian

Univariate

$$p(x) \sim N(\mu, \sigma^2):$$

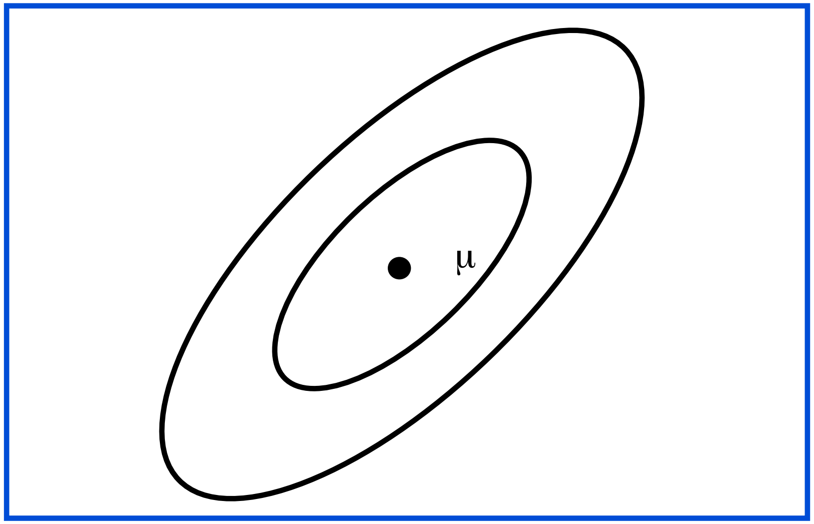
$$p(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{1}{2}\frac{(x-\mu)^2}{\sigma^2}}$$



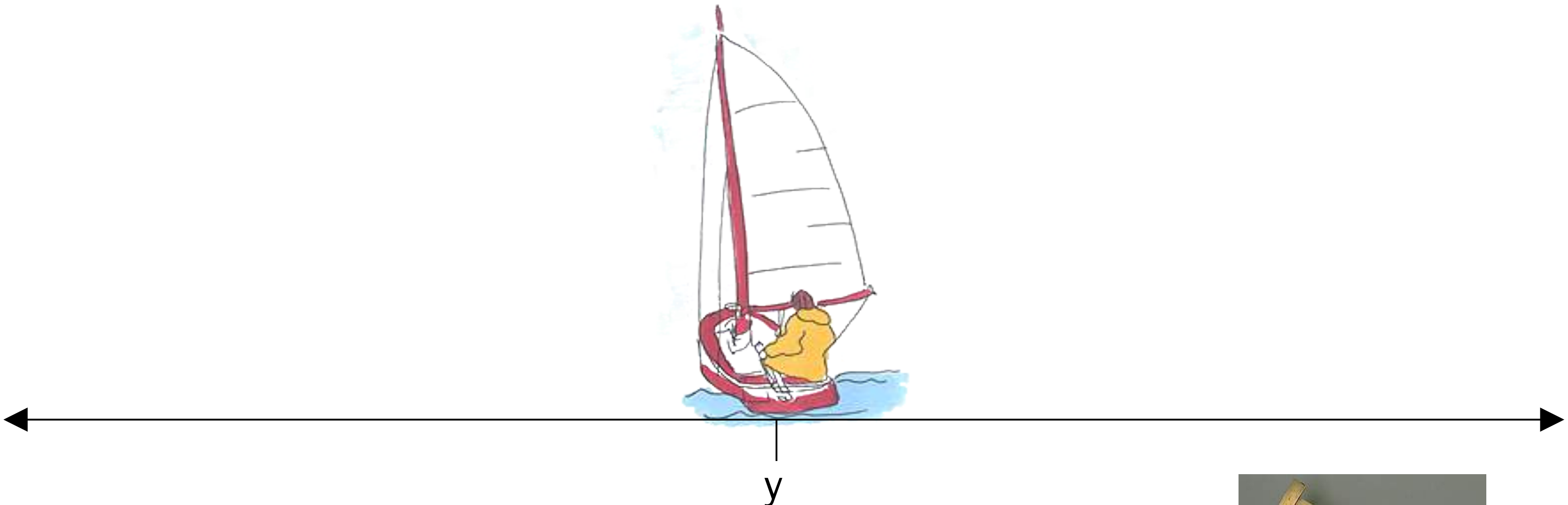
Multivariate

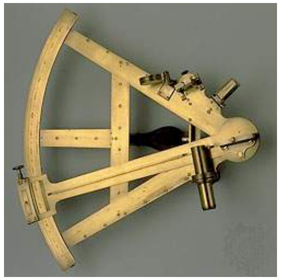
$$p(\mathbf{x}) \sim N(\boldsymbol{\mu}, \boldsymbol{\Sigma}):$$

$$p(\mathbf{x}) = \frac{1}{(2\pi)^{d/2}|\boldsymbol{\Sigma}|^{1/2}} e^{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^t \boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})}$$



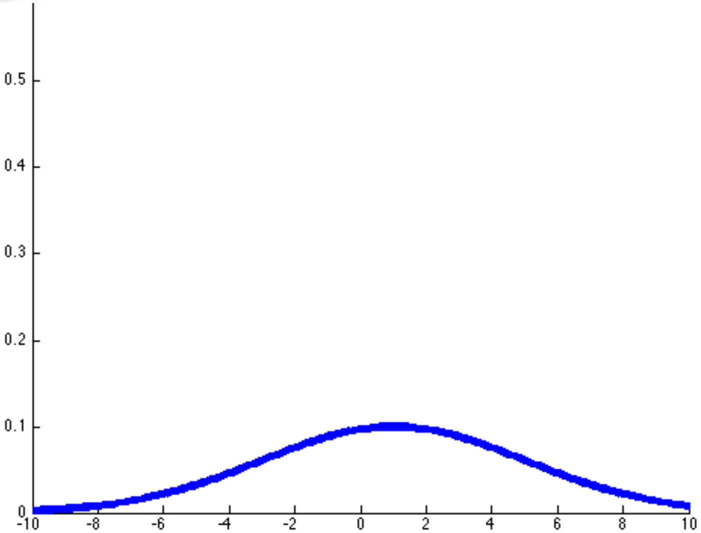
1D Example

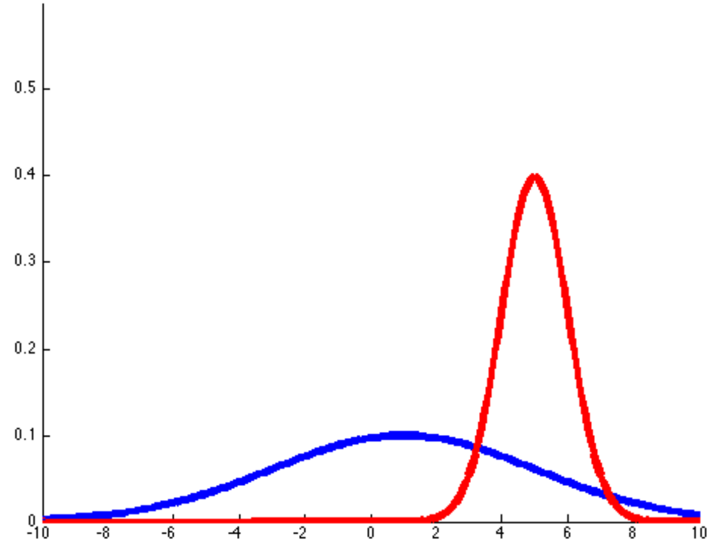




- Lost on the 1-D line
- Position : $x(t)$
- Assume Gaussian distributed measurements

Prior Belief





- Current belief $bel(x_t)$

prior

μ_prior = 1;

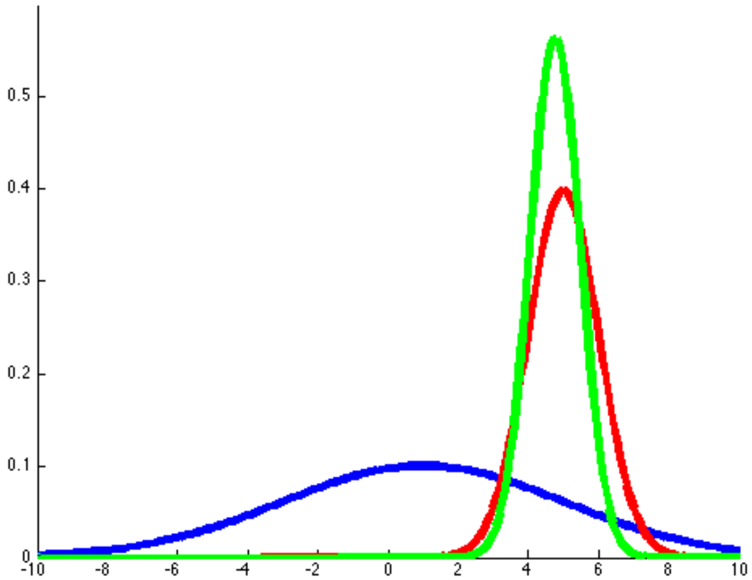
σ_prior = 4;

measurement

μ_meas = 5;

σ_meas = 1;

Kalman Filter - Posterior



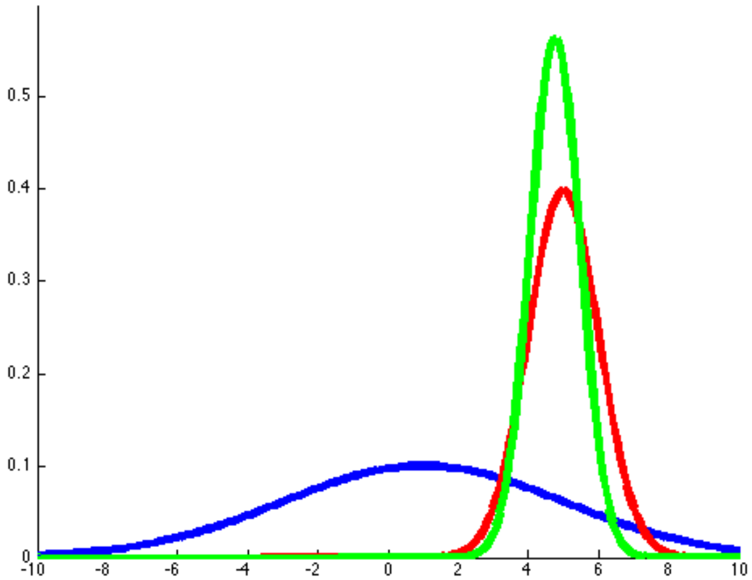
- Corrected mean is the new optimal estimate of the position
- New variance is smaller than either of the previous two variances

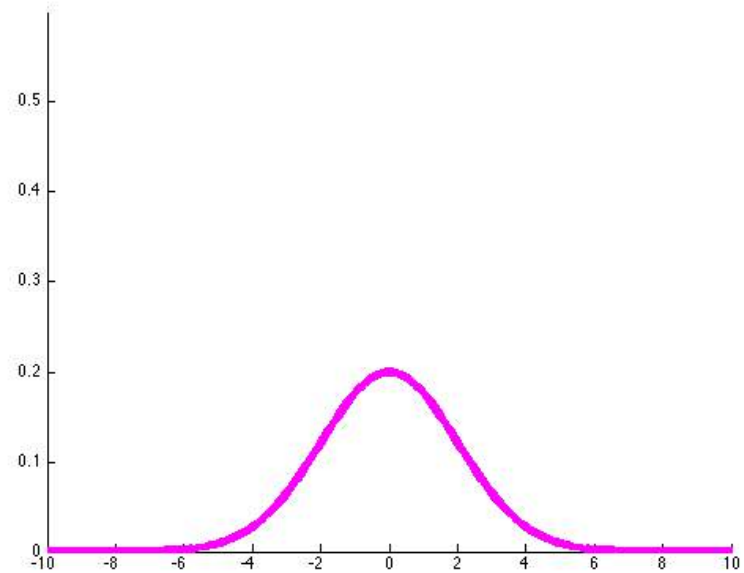
$$\mu_{posterior} = \frac{\sigma^2_{meas}\mu_{prior} + \sigma^2_{prior}\mu_{meas}}{\sigma^2_{prior} + \sigma^2_{meas}}$$

$$bel(x_k) \sim \mathcal{N}(x_k; \mu_{x_k}, \sigma_k^2)$$

$$\sigma^2_{posterior} = \frac{1}{\frac{1}{\sigma^2_{prior}} + \frac{1}{\sigma^2_{meas}}}$$

Kalman Filter - Motion

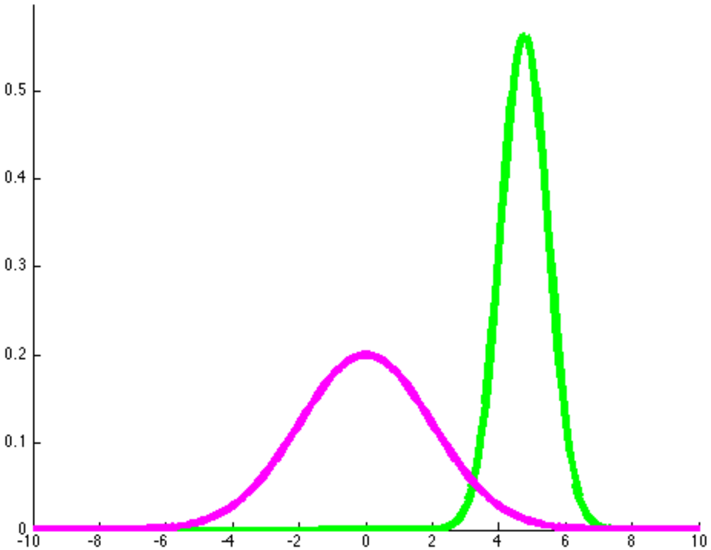


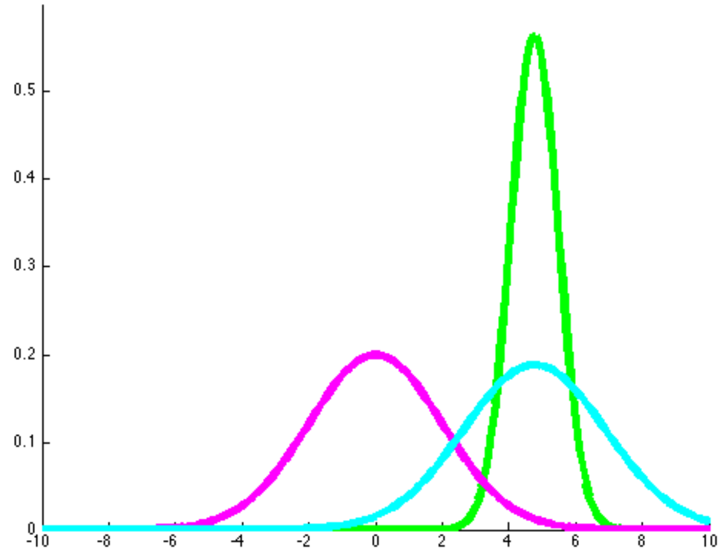


Now we apply a motion to our state.

mu_move = 0;
sigma_move = 2;

Kalman Filter - Prior





$$\overline{bel}(x_k) \sim \mathcal{N}(\bar{x}_k; \bar{\mu}_{x_k}, \bar{\sigma}_k^2)$$

$$\mu_{prior} = \mu_{posterior} + \mu_{move}$$

$$\sigma^2_{prior} = \sigma^2_{posterior} + \sigma^2_{move}$$

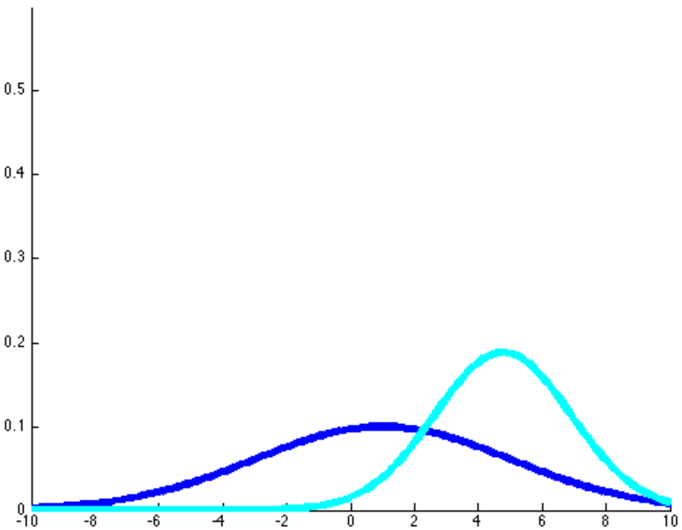
- With a bigger covariance than both because we sum them.

Kalman Filter – Prior vs. Posterior

Prediction: $\overline{bel}(x_k) \sim \mathcal{N}(\bar{x}_k; \bar{\mu}_{x_k}, \bar{\sigma}_k^2)$

$$\bar{\mu}_{x_k} = \mu_{x_{k-1}}$$

$$\bar{\sigma}_k^2 = \sigma_{k-1}^2 + \sigma_w^2$$



Update: $bel(x_k) \sim \mathcal{N}(x_k; \mu_{x_k}, \sigma_k^2)$

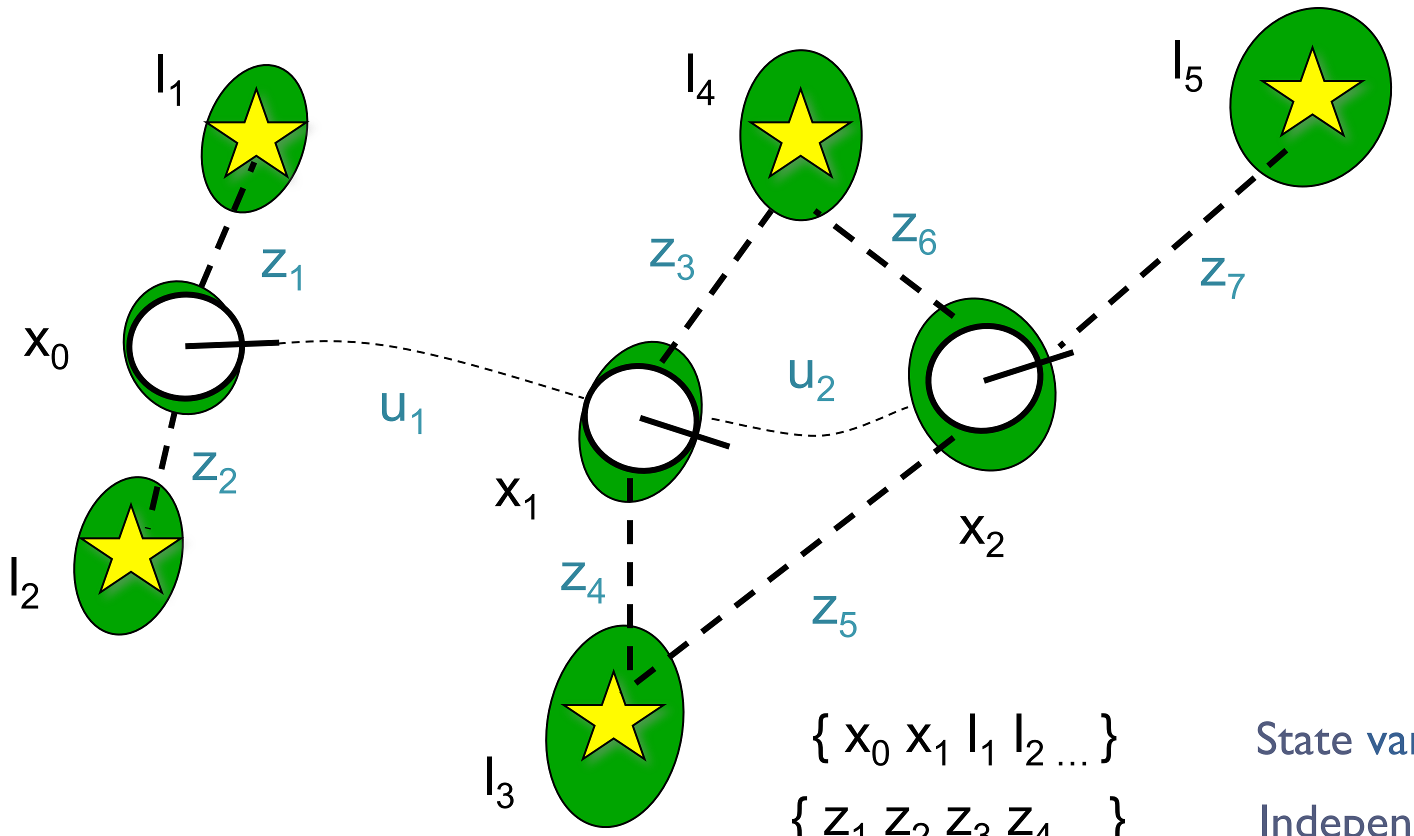
$$\mu_{x_k} = \bar{\mu}_{x_k} + K_k(z_k - \bar{\mu}_{x_k})$$

$$\sigma_k^2 = (1 - K_k)\bar{\sigma}_k^2$$

$$K_k = \bar{\sigma}_k^2\,(\bar{\sigma}_k^2 + \sigma_v^2)^{-1}$$

- When we compare the two priors, the one after measurement is more certain.

SLAM – variables and measurements



$\{ x_0\ x_1\ l_1\ l_2 \ldots \}$ State variables

$\{ z_1\ z_2\ z_3\ z_4 \ldots \}$ Independent data

$\{ u_1 \ldots \}$ Control data

SLAM Random Variables

$$\mathbf{x_i} \sim \mathcal{N}(\mu_{\mathbf{x_i}}, \boldsymbol{\Sigma}_{\mathbf{x_i}}) \qquad \mathbf{l_i} \sim \mathcal{N}(\mu_{\mathbf{l_i}}, \boldsymbol{\Sigma}_{\mathbf{l_i}})$$

$$\boldsymbol{\mu}_{x_i} \qquad \boldsymbol{\Sigma}_{x_i}$$

$$\begin{array}{c} x \\ y \\ \theta \end{array} \begin{bmatrix} \\ \\ \end{bmatrix} \qquad \begin{array}{c} \begin{array}{ccc} x & y & \theta \end{array} \\ \begin{bmatrix} \sigma_{11} & & \\ & \sigma_{22} & \\ & & \sigma_{33} \end{bmatrix} \end{array}$$

$$\boldsymbol{\mu}_{l_j} \qquad \boldsymbol{\Sigma}_{l_j}$$

$$\begin{array}{c} x \\ y \end{array} \begin{bmatrix} \\ \end{bmatrix} \qquad \begin{array}{c} \begin{array}{cc} x & y \end{array} \\ \begin{bmatrix} \sigma_{11} & \\ & \sigma_{22} \end{bmatrix} \end{array}$$

Noisy Models

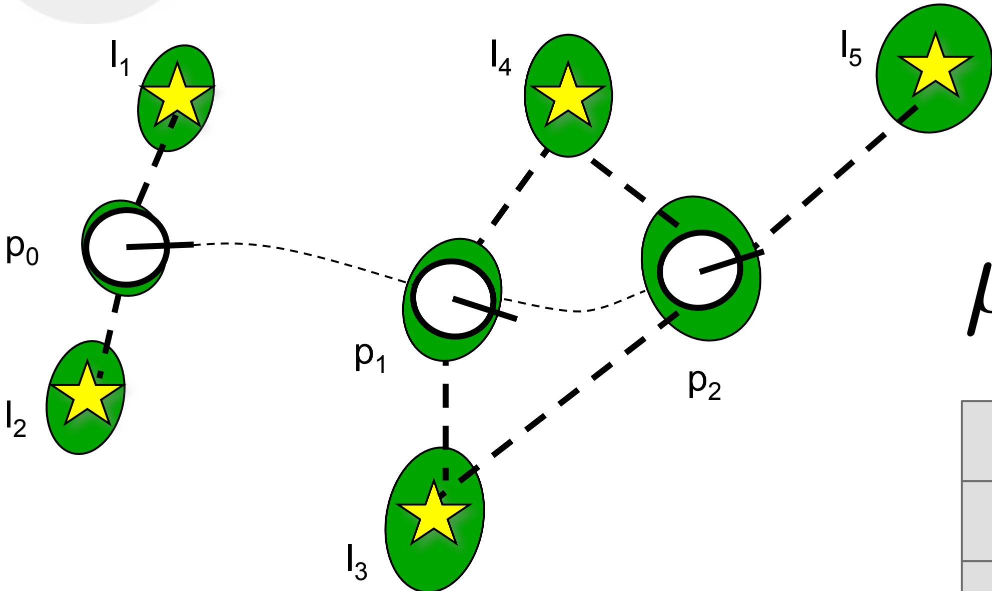
Motion model:

$$
\begin{aligned}
x_t &= f_i(x_{t-1}, u_t) + v_t \\
P(x_t \mid x_{t-1}, u_t) &\propto \exp\left(-\frac{1}{2} \parallel f_i(x_{t-1}, u_t) - x_t \parallel^2_{\Sigma_{x_t}}\right)
\end{aligned}
$$

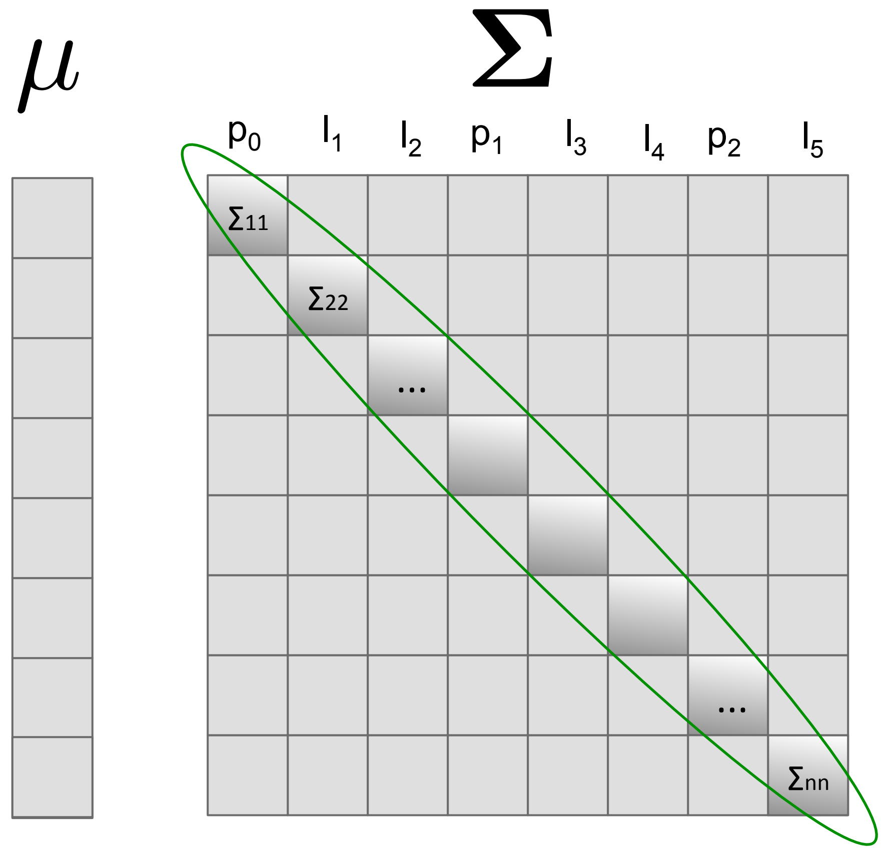
Observation model:

$$
\begin{aligned}
z_t^j &= h_k(x_t, l_j) + v_n \\
P(z_t^j \mid x_t, l_j) &\propto \exp\left(-\frac{1}{2} \parallel h(\mu_{x_t}, \mu_{l_j}) - z_t^j \parallel^2_{\Sigma_{z_t^j}}\right)
\end{aligned}
$$

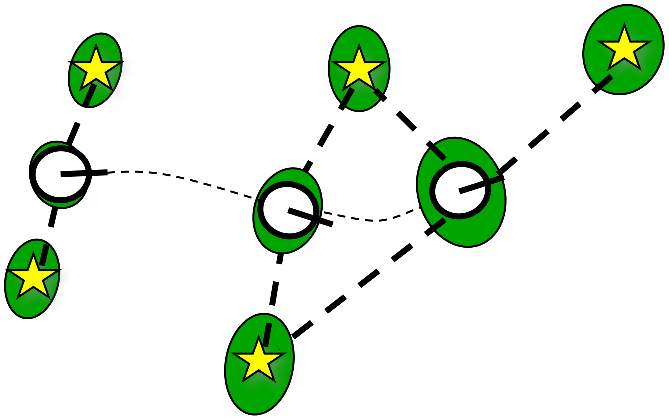
Full SLAM State



$$P(\mathbf{x}) \sim \mathcal{N}(\mathbf{x}; \mu, \mathbf{\Sigma}),$$



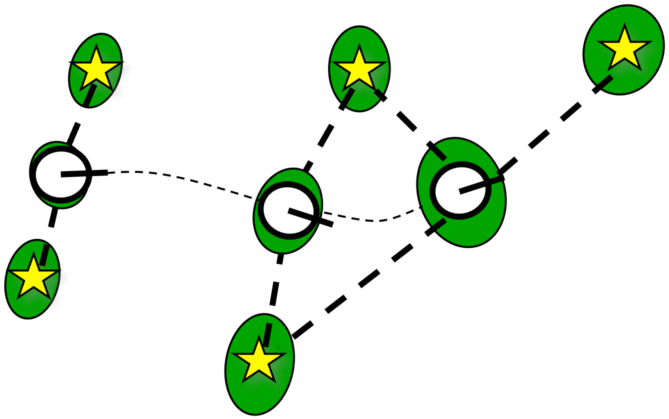
Covariance



Cross Covariance

$$\mu_t = \begin{bmatrix} \mu_{x_t} \\ \mu_{l_1} \\ \cdots \\ \mu_{l_n} \end{bmatrix}, \ \Sigma_t = \begin{bmatrix} \Sigma_{x_t,x_t} & \Sigma_{x_t,l_1} & \cdots & \Sigma_{x_t,l_n} \\ \Sigma_{x_t,l_1}^\top & \Sigma_{l_1,l_1} & \cdots & \Sigma_{l,l_n} \\ \cdots & \cdots & \cdots & \cdots \\ \Sigma_{x_t,l_n}^\top & \Sigma_{l_1,l_n}^\top & \cdots & \Sigma_{l_n,l_n} \end{bmatrix}$$

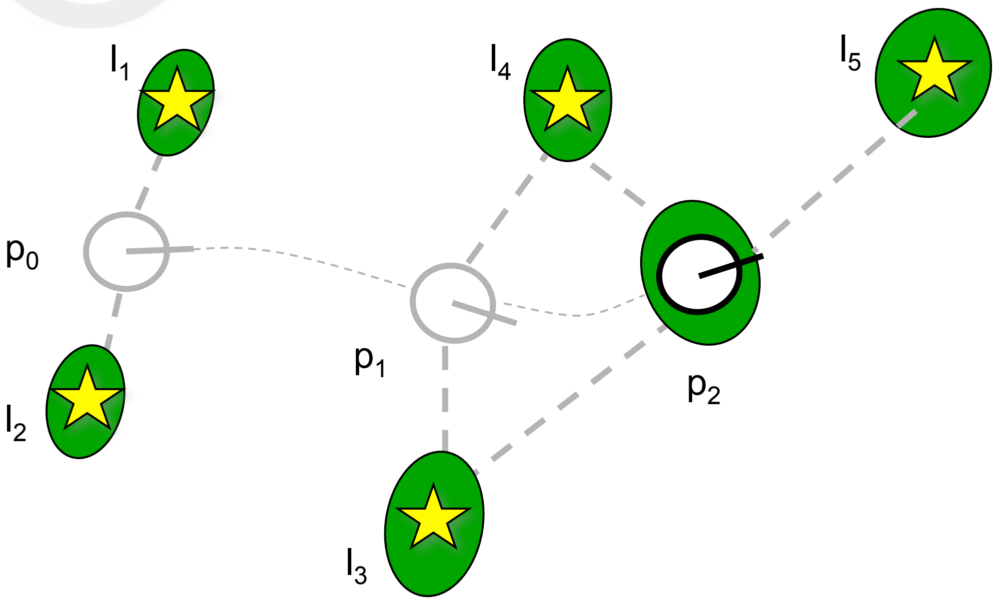
Covariance

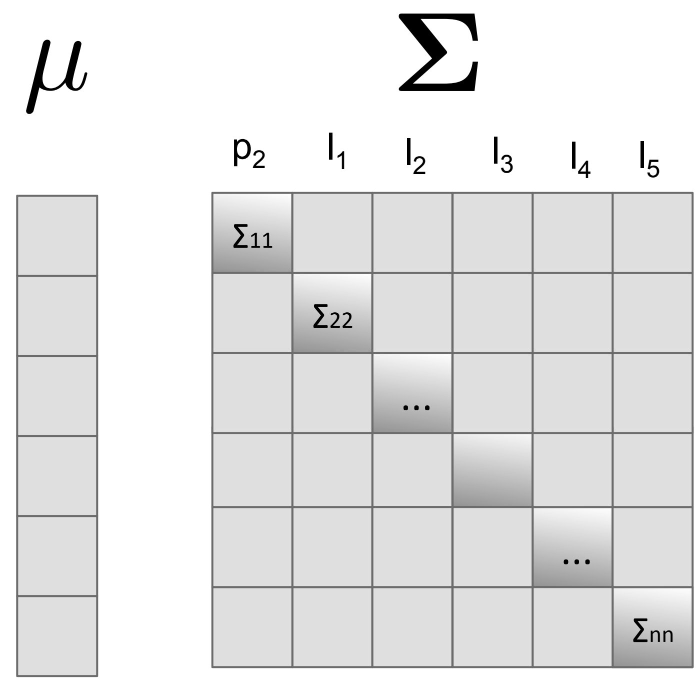


Marginal Covariance

$$\mu_t = \begin{bmatrix} \mu_{x_t} \\ \mu_{l_1} \\ \ldots \\ \mu_{l_n} \end{bmatrix}, \ \Sigma_t = \begin{bmatrix} \Sigma_{x_t,x_t} & \Sigma_{x_t,l_1} & \ldots & \Sigma_{x_t,l_n} \\ \Sigma^\top_{x_t,l_1} & \Sigma_{l_1,l_1} & \ldots & \Sigma_{l,l_n} \\ \ldots & \ldots & \ldots & \ldots \\ \Sigma^\top_{x_t,l_n} & \Sigma^\top_{l_1,l_n} & \ldots & \Sigma_{l_n,l_n} \end{bmatrix}$$

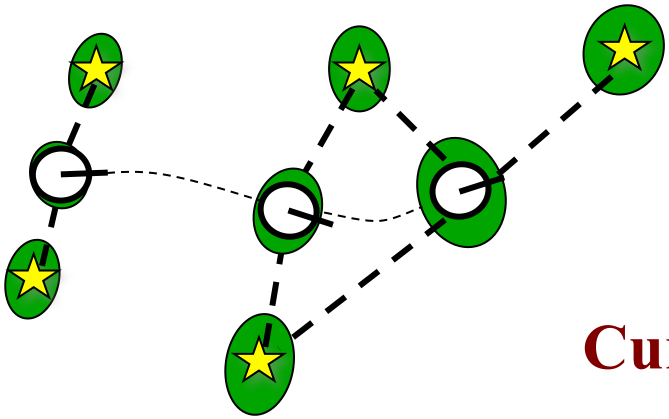
Landmark SLAM - State





$$P(\mathbf{x}) \sim \mathcal{N}(\mathbf{x}; \mu, \mathbf{\Sigma}),$$

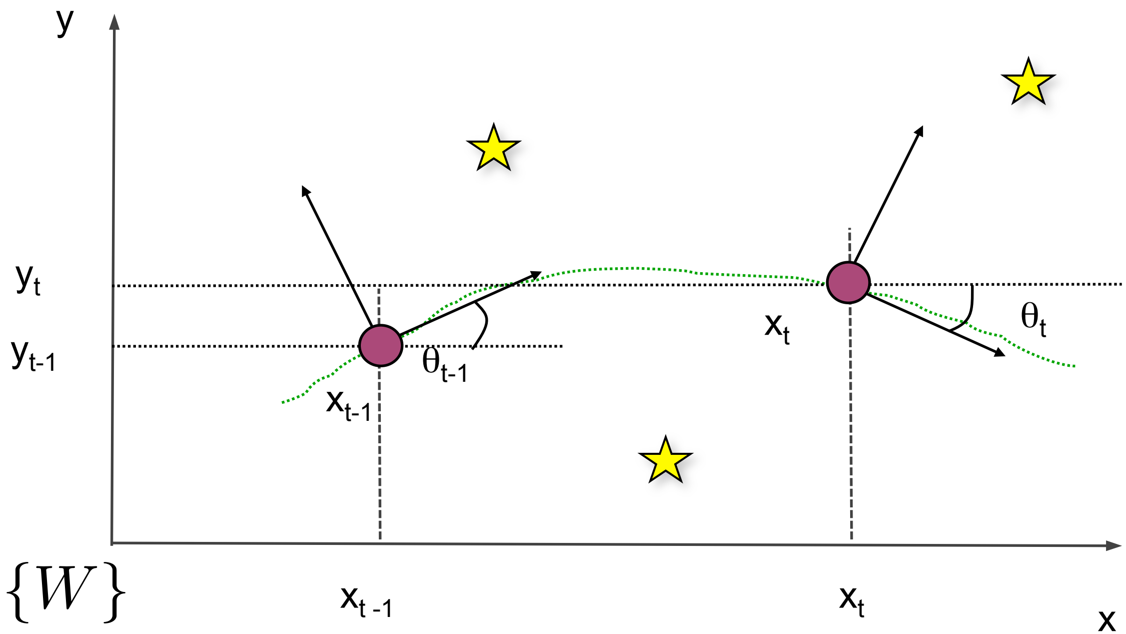
Landmark SLAM - State



Current robot pose + All landmarks

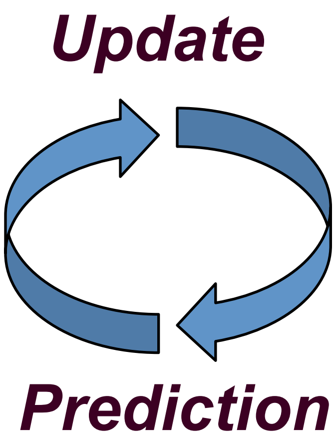
$$\mu_t = \begin{bmatrix} \boxed{\mu_{x_t}} \\ \mu_{l_1} \\ \cdots \\ \mu_{l_n} \end{bmatrix}, \quad \Sigma_t = \begin{bmatrix} \boxed{\Sigma_{x_t,x_t}} & \Sigma_{x_t,l_1} & \cdots & \Sigma_{x_t,l_n} \\ \Sigma_{x_t,l_1}^\top & \Sigma_{l_1,l_1} & \cdots & \Sigma_{l,l_n} \\ \cdots & \cdots & \cdots & \cdots \\ \Sigma_{x_t,l_n}^\top & \Sigma_{l_1,l_n}^\top & \cdots & \Sigma_{l_n,l_n} \end{bmatrix}$$

2D SLAM



Nonlinear functions!!!

$$
\begin{aligned}
x_t &= f(x_{t-1}, u_t) + v_t \\
z_t^j &= h(x_t, l_j) + w_t^j
\end{aligned}
$$



Linearize Motion Model

$$x_t \quad = \quad f(x_{t-1}, u_t) + v_t$$

Random variables:

$$x_t \sim \mathcal{N}(\mu_{x_t}, \Sigma_{x_t})$$

$$u_t \sim \mathcal{N}(\mu_{u_t}, \Sigma_{u_t})$$

$$v_t \sim \mathcal{N}(0, \Sigma_{v_t})$$

Linearize:

$$f(x_t, u_t) \approx f(\mu_{x_t}, \mu_{u_t}) + F(x_t - \mu_{x_t}) + W(u_t - \mu_{u_t})$$

Jacobians:

$$F = \left.\frac{\partial f(x_t, u_t)}{\partial x_t}\right|_{x_t=\mu_{x_t}} \qquad W = \left.\frac{\partial f(x_t, u_t)}{\partial u_t}\right|_{u_t=\mu_{u_t}}$$

EKF - Prediction

From the belief at time t and the noisy action u:

$$bel(x_t) \sim \mathcal{N}(x_t; \mu_{x_t}, \Sigma_{x_t}) \qquad bel(u_t) \sim \mathcal{N}(u_t; \mu_{u_t}, \Sigma_{u_t})$$

The mean is calculated using the nonlinear function:

$$\boxed{\bar{\mu}_{x_t} = f(\mu_{x_{t-1}}, \mu_{u_t})}$$

$$\overline{\mu}_t = \begin{bmatrix} \overline{\mu}_{x_t} \\ \mu_{l_1} \\ \mu_{l_2} \\ \dots \end{bmatrix}$$

The covariance is calculated using the Jacobians:

$$\boxed{\overline{\Sigma}_t = F\,\Sigma_{t-1}F^\top + W\,\Sigma_{u_t}W^\top}$$

Prediction – Linear Algebra

$$\bar{\Sigma}_t = \begin{bmatrix} F_t & 0 & \dots \\ 0 & I & \dots \\ \dots & \dots & I \end{bmatrix} \begin{bmatrix} \Sigma_{x,x} & \Sigma_{x,l_1} & \dots \\ \Sigma_{x,l_1}^\top & \Sigma_{l,l_1} & \dots \\ \dots & \dots & \dots \end{bmatrix} \begin{bmatrix} F_t & 0 & \dots \\ 0 & I & \dots \\ \dots & \dots & I \end{bmatrix}^\top + W \begin{bmatrix} \Sigma_{u_t} & 0 & \dots \\ 0 & 0 & \dots \\ \dots & \dots & 0 \end{bmatrix} W^\top$$

For multi-dimensional covariance matrix we need to scatter the elements.

$$\bar{\Sigma}_t = \begin{bmatrix} F_t \Sigma_{x,x} F_t^\top + \Sigma_{u_t} & F_t \Sigma_{x,l_1} & \dots \\ \Sigma_{x,l_1}^\top F_t^\top & \Sigma_{l_1,l_1} & \dots \\ \dots & \dots & \dots \end{bmatrix}$$

EKF - Prediction

Algorithm Extended Kalman_filter $(\mu_{t-1}, \Sigma_{t-1}, u_t, z_t)$:

Prediction:

$$\bar{\mu}_{x_t} = f(\mu_{x_{t-1}}, \mu_{u_t})$$

$$\overline{\Sigma}_t = F\,\Sigma_{t-1}F^\top + W\,\Sigma_{u_t}W^\top$$

Update:

Return μ_t, Σ_t

Linearize Observation

Random variables:

$$z_t^j \sim \mathcal{N}(\mu_{z_t^j}, \Sigma_{z_t^j})$$
$$x_t \sim \mathcal{N}(\mu_{x_t}, \Sigma_{x_t})$$
$$l_j \sim \mathcal{N}(\mu_{l_j}, \Sigma_{l_j})$$
$$w_t^j \sim \mathcal{N}(0, \Sigma_{w_t^j})$$

$$\boxed{z_t^j \quad = \quad h(x_t, l_j) + w_t^j}$$

Linearize:

$$\boxed{h(x_t, l_j) \approx h(\mu_{x_t}, \mu_{l_j}) + H_t^j(x_t - \bar{\mu}_{x_t}) + J_j^t(l_j - \mu_{l_j})}$$

Jacobians:

$$H_t^j = \left.\frac{\partial h(x_t, l_j)}{\partial x_t}\right|_{x_t = \bar{\mu}_{x_t}} \qquad J_j^t = \left.\frac{\partial h(x_t, l_j)}{\partial l_j}\right|_{l_j = \mu_{l_j}}$$

EKF - Update

From the belief at time t

$$\overline{bel}(x_t) \sim \mathcal{N}(x_t; \overline{\mu}_{x_t}, \overline{\Sigma}_{x_t})$$

and the measurement probability:

$$p(z_t^j | x_t, l_j) \sim \mathcal{N}(z_t^j; h(x_t, l_j), \Sigma_{z_t^j})$$

The Kalman gain is calculated as:

$$\boxed{K = \overline{\Sigma}_t \, C_t^\top (C_t \, \overline{\Sigma}_t \, C_t^\top + \Sigma_{z_t^j})^{-1}}$$

Kalman Gain – Linear Algebra

$$\boxed{K = \bar{\Sigma}_t\, C_t^\top (C_t\, \bar{\Sigma}_t\, C_t^\top + \Sigma_{z_t^j})^{-1}}$$

$$C_t^j = [H_t^j, 0, \ldots, J_j^t, 0, \ldots]$$

Jacobian of one measurement

$$K = \begin{bmatrix} \Sigma_{t,t} & \ldots & \Sigma_{t,l_j} & \ldots \\ \ldots & \ldots & \ldots & \ldots \\ \Sigma_{t,l_j}^\top & \ldots & \Sigma_{l_j,l_j} & \ldots \\ \ldots & \ldots & \ldots & \ldots \end{bmatrix} \begin{bmatrix} {H_t^j}^\top \\ \ldots \\ {J_j^t}^\top \\ \ldots \end{bmatrix} \cdot (\ldots)^{-1}$$

Linearize Observation

$$\boxed{z_t^j \quad = \quad h(x_t, l_j) + w_t^j}$$

The Kalman gain:

$$K_t = \overline{\Sigma}_t\, C_t^\top (C_t\, \overline{\Sigma}_t\, C_t^\top + \Sigma_{z_t^j})^{-1}$$

The posterior mean:

$$\mu_t = \bar{\mu}_t + K_t(z_t - h(\bar{\mu}_{\mathbf{x}_t}, \mu_{\mathbf{l}_j}))$$

The posterior covariance:

$$\Sigma_t = (I - K_t\, C_t)\, \overline{\Sigma}_t$$

EKF

Algorithm Extended Kalman_filter (μ_{t-1}, Σ_{t-1}, u_t, z_t):

Prediction:

$$\bar{\mu}_{x_t} = f(\mu_{x_{t-1}}, \mu_{u_t})$$

$$\overline{\Sigma}_t = F\,\Sigma_{t-1} F^\top + W\,\Sigma_{u_t} W^\top$$

Update:

$$K_t = \overline{\Sigma}_t\, C_t^\top (C_t\, \overline{\Sigma}_t\, C_t^\top + \Sigma_{z_t^j})^{-1}$$

$$\mu_t = \bar{\mu}_t + K_t(z_t - h(\bar{\mu}_{\mathbf{x}_t}, \mu_{\mathbf{l}_j}))$$

$$\Sigma_t = (I - K_t\, C_t)\, \overline{\Sigma}_t$$

Return μ_t, Σ_t

Initialize a Landmark

$$\mu_{\mathbf{l}_t} = g(\mathbf{x}_t, z_t)$$

When seeing a landmark for the first time the landmark needs to be initialized.

$$G_1 = \left.\frac{\partial g(\mathbf{x}_t, \mathbf{z}_t)}{\partial \mathbf{x}_t}\right|_{\mathbf{x}_t = \mu_{\mathbf{x}_t}} \qquad G_2 = \left.\frac{\partial g(\mathbf{x}_t, \mathbf{z}_t)}{\partial \mathbf{z}_t}\right|_{\mathbf{z}_t = \mu_{\mathbf{z}_t}}$$

$$\mu_t = \begin{bmatrix} \mu_{x_t} \\ \mu_{l_1} \\ \dots \\ \boxed{\mu_{l_t}} \end{bmatrix}, \quad \Sigma_1 = \begin{bmatrix} \Sigma_{x_t,x_t} & \Sigma_{x_t,l_1} & \dots & \boxed{\Sigma_{x_t,l_t}} \\ \Sigma^\top_{x_t,l_1} & \Sigma_{l_1,l_1} & \dots & \boxed{\Sigma_{l_1,l_t}} \\ \dots & \dots & \dots & \dots \\ \boxed{\Sigma^\top_{x_t,l_t}} & \boxed{\Sigma^\top_{l_1,l_t}} & \boxed{\dots} & \boxed{\Sigma_{l_t,l_t}} \end{bmatrix}$$

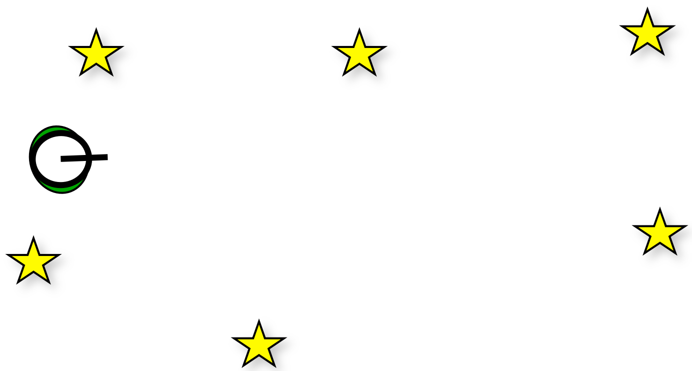
Initialize a Landmark

$$\Sigma_{l_t,l_t} = G_1 \, \bar{\Sigma}_{x_t,x_t} \, G_1^\top + G_2 \, Q_{z_t} \, G_2^\top$$

$$[\Sigma_{x_t,l_t}, \Sigma_{l_1,l_t} \ldots]^\top = G_1 \; [\bar{\Sigma}_{x_t,x_t}, \bar{\Sigma}_{x_t,l_1}, \ldots]^\top$$

$$\mu_t = \begin{bmatrix} \mu_{x_t} \\ \mu_{l_1} \\ \ldots \\ \mu_{l_t} \end{bmatrix}, \; \Sigma_1 = \begin{bmatrix} \Sigma_{x_t,x_t} & \Sigma_{x_t,l_1} & \ldots & \Sigma_{x_t,l_t} \\ \Sigma_{x_t,l_1}^\top & \Sigma_{l_1,l_1} & \ldots & \Sigma_{l_1,l_t} \\ \ldots & \ldots & \ldots & \ldots \\ \Sigma_{x_t,l_t}^\top & \Sigma_{l_1,l_t}^\top & \ldots & \Sigma_{l_t,l_t} \end{bmatrix}$$

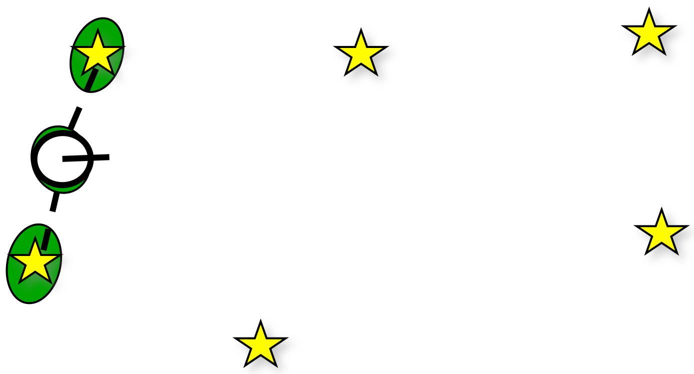
EKF-SLAM



- Assumes it starts in known location

$$\overline{\mu}_0 = \left[\mu_{x_0}\right],\ \overline{\Sigma}_0 = \left[\Sigma_{x_0,x_0}\right]$$

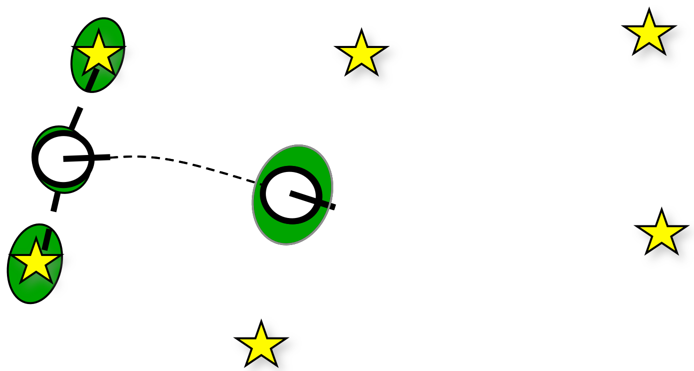
EKF-SLAM



- Assumes it starts in known location
- Observe the landmarks

$$\mu_0 = \begin{bmatrix} \mu_{x_0} \\ \mu_{l_1} \\ \mu_{l_2} \end{bmatrix}, \ \Sigma_0 = \begin{bmatrix} \Sigma_{x_0,x_0} & \Sigma_{x_0,l_1} & \Sigma_{x_0,l_2} \\ \Sigma_{x_0,l_1}^\top & \Sigma_{l_1,l_1} & \Sigma_{l_1,l_2} \\ \Sigma_{x_0,l_2}^\top & \Sigma_{l_1,l_2} & \Sigma_{l_2,l_2} \end{bmatrix}$$

EKF-SLAM

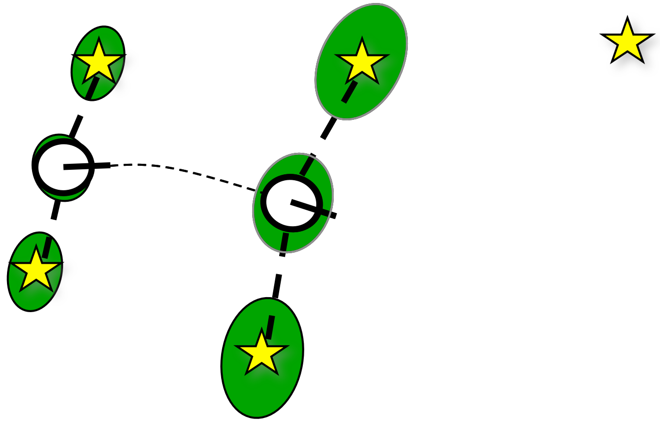


- Assumes it starts in known location
- Observe the landmarks
- Move

The prediction step only affects pose-related state elements

$$\overline{\mu}_1 = \begin{bmatrix} \boxed{\mu_{x_1}} \\ \mu_{l_1} \\ \mu_{l_2} \end{bmatrix}, \; \overline{\Sigma}_1 = \begin{bmatrix} \Sigma_{x_1,x_1} & \Sigma_{x_1,l_1} & \Sigma_{x_1,l_2} \\ \Sigma_{x_1,l_1}^\top & \Sigma_{l_1,l_1} & \Sigma_{l_1,l_2} \\ \Sigma_{x_1,l_2}^\top & \Sigma_{l_1,l_2} & \Sigma_{l_2,l_2} \end{bmatrix}$$

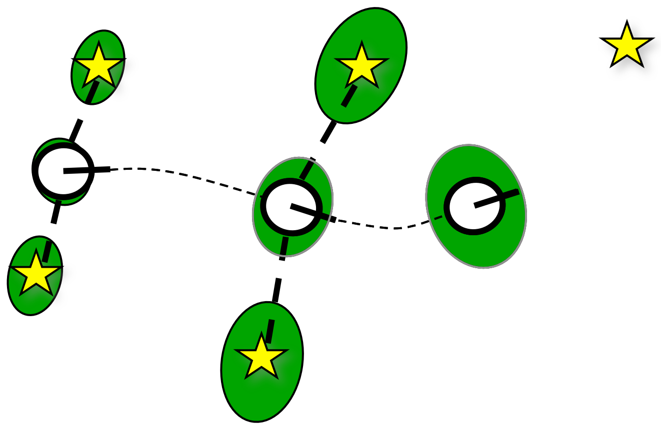
EKF-SLAM



- Assumes it starts in known location
- Observe the landmarks
- Move
- Observe the landmarks

$$\mu_1 = \begin{bmatrix} \mu_{x_1} \\ \mu_{l_1} \\ \mu_{l_2} \\ \cdots \end{bmatrix}, \ \Sigma_1 = \begin{bmatrix} \Sigma_{x_1,x_1} & \Sigma_{x_1,l_1} & \Sigma_{x_1,l_2} & \cdots \\ \Sigma^\top_{x_1,l_1} & \Sigma_{l_1,l_1} & \Sigma_{l_1,l_2} & \cdots \\ \Sigma^\top_{x_1,l_2} & \Sigma_{l_1,l_2} & \Sigma_{l_2,l_2} & \cdots \\ \cdots & \cdots & \cdots & \cdots \end{bmatrix}$$

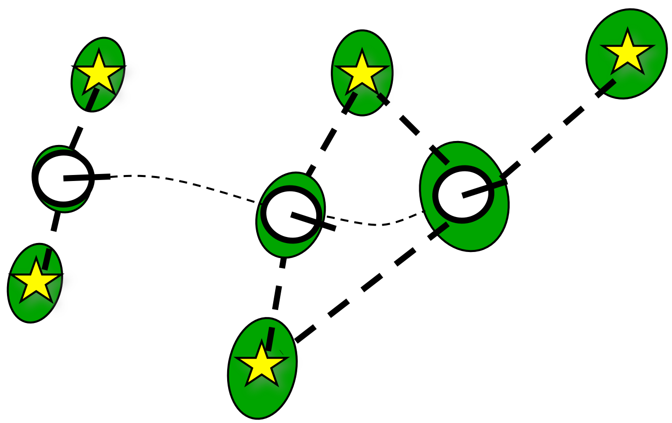
EKF-SLAM



- Assumes it starts in known location
- Observe the landmarks
- Move
- Observe the landmarks
- Move

$$\overline{\mu}_2 = \begin{bmatrix} \mu_{x_2} \\ \mu_{l_1} \\ \mu_{l_2} \\ \cdots \end{bmatrix}, \quad \overline{\Sigma}_2 = \begin{bmatrix} \Sigma_{x_2,x_2} & \Sigma_{x_2,l_1} & \Sigma_{x_2,l_2} & \cdots \\ \Sigma_{x_2,l_1}^\top & \Sigma_{l_1,l_1} & \Sigma_{l_1,l_2} & \cdots \\ \Sigma_{x_2,l_2}^\top & \Sigma_{l_1,l_2} & \Sigma_{l_2,l_2} & \cdots \\ \cdots & \cdots & \cdots & \cdots \end{bmatrix}$$

EKF-SLAM

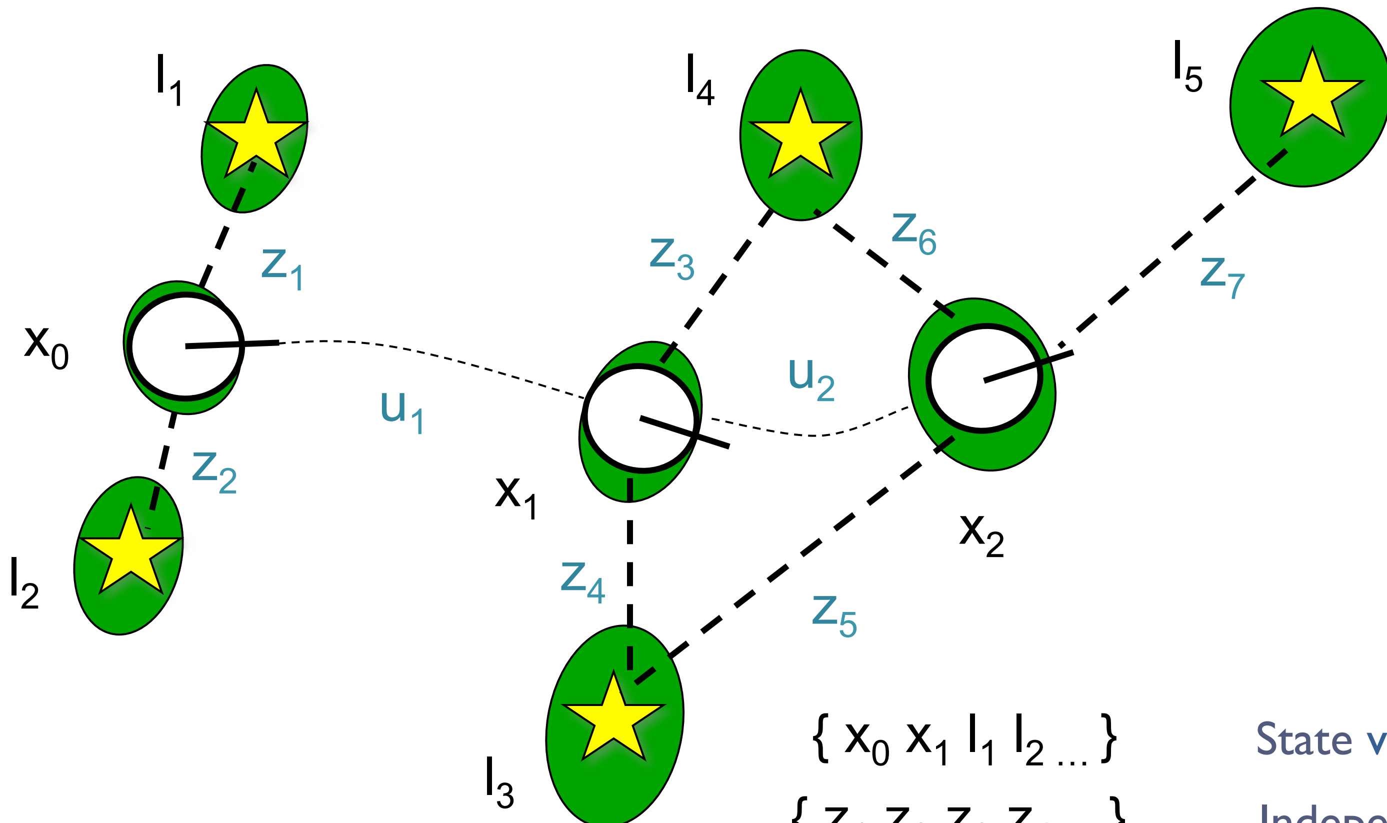


- Assumes it starts in known location
- Observe the landmarks
- Move
- Observe the landmarks
- Move
- Observe + Associate landmarks

$$\mu_2 = \begin{bmatrix} \mu_{x_2} \\ \dots \\ \mu_{l_4} \\ \mu_{l_5} \end{bmatrix}, \; \Sigma_2 = \begin{bmatrix} \Sigma_{x_2,x_2} & \dots & \Sigma_{x_2,l_4} & \Sigma_{x_2,l_5} \\ \dots & \dots & \dots & \dots \\ \Sigma^\top_{x_2,l_14} & \dots & \Sigma_{l_4,l_4} & \Sigma_{l_4,l_5} \\ \Sigma^\top_{x_2,l_5} & \dots & \Sigma^\top_{l_4,l_5} & \Sigma_{l_5,l_5} \end{bmatrix}$$

Maximum Likelihood Estimation
The SLAM Example

SLAM – variables and measurements

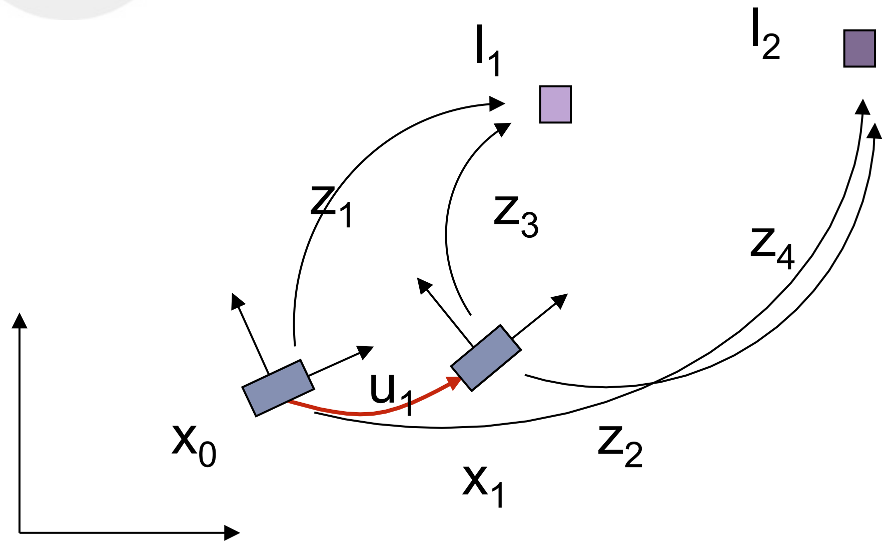


$\{ x_0\ x_1\ l_1\ l_2 \ldots \}$ State variables

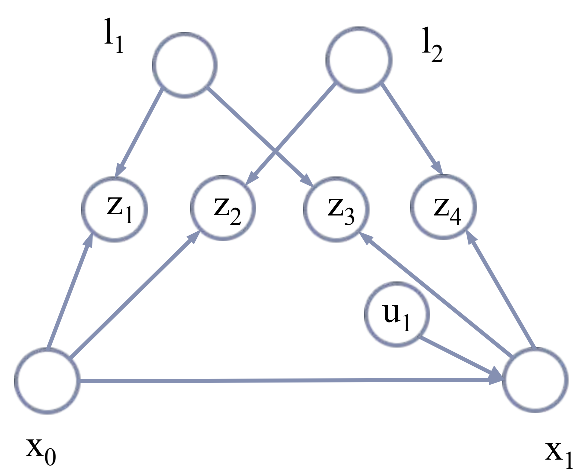
$\{ z_1\ z_2\ z_3\ z_4 \ldots \}$ Independent data

$\{ u_1 \ldots \}$ Control data

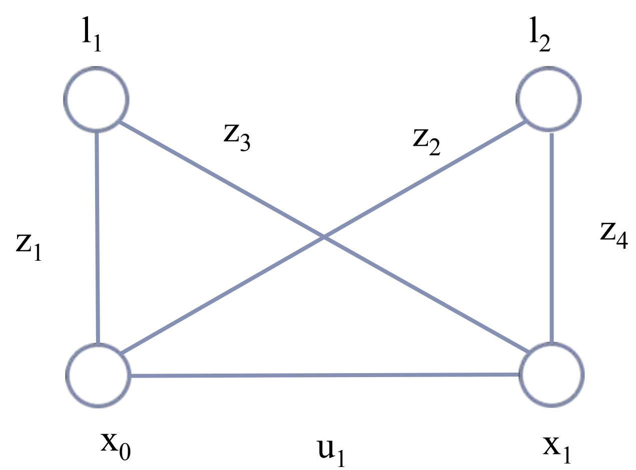
Graphical Models for SLAM



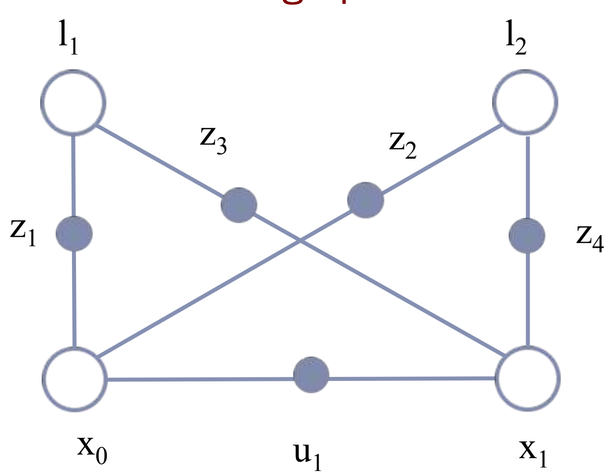
Bayesian belief network



Markov Networks



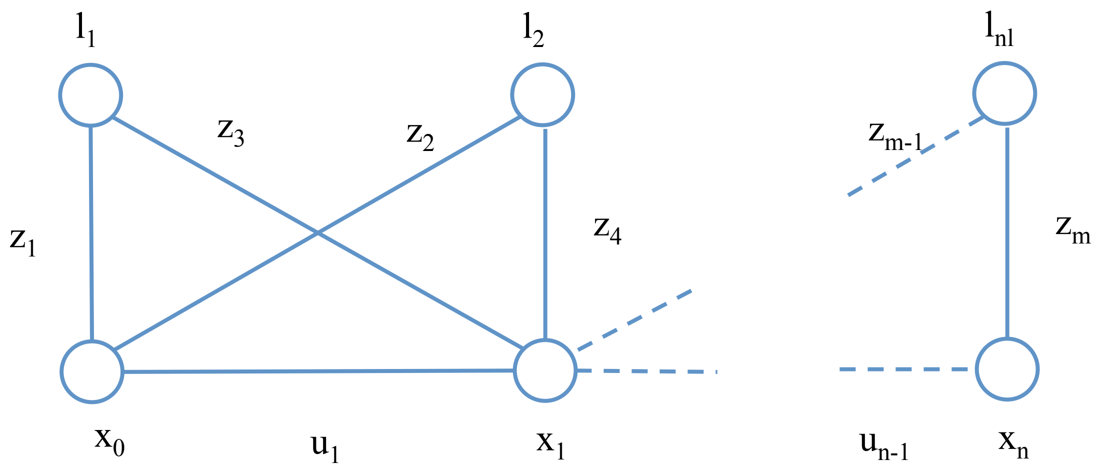
Factor graphs



Maximum Likelihood Estimation

Maximum A Posteriori estimate (MAP)

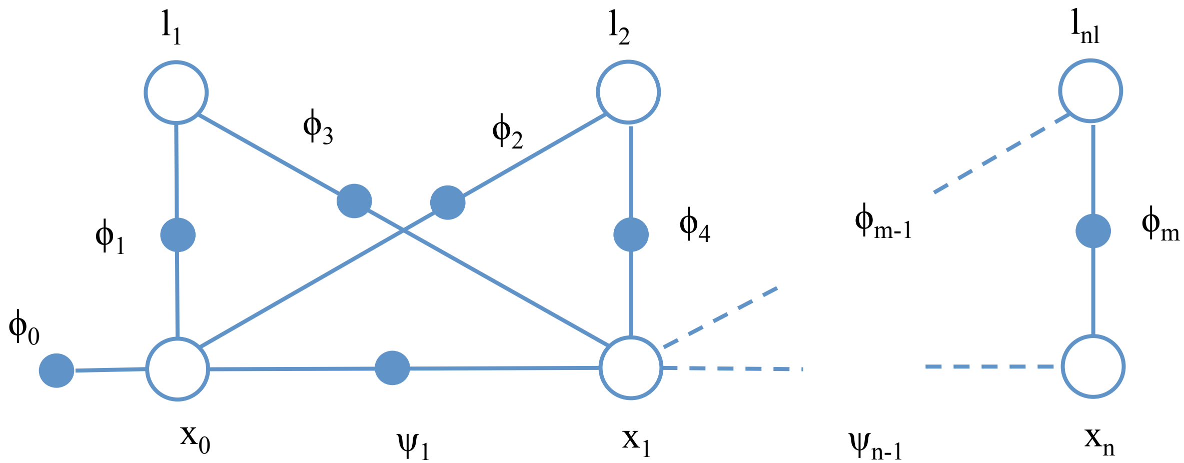
$$\boxed{P(X, L) =} \; P(\mathbf{x}_0) \prod_i^n P(x_i \mid x_{i-1}, u_i) \prod_k^m P(z_k \mid x_{i_k}, l_{j_k})$$



The configuration that maximizes the joint probability distribution

Maximum Likelihood Estimation

$$\boxed{P(X, L) =} \ \phi(\mathbf{x}_0) \prod_{i}^{n} \psi(x_{i-1}, u_i) \prod_{k}^{m} \phi(x_{i_k}, l_{j_k})$$



Factor graph expression of the joint probability distribution

Maximum Likelihood Estimation

$$P(X, L) = P(\mathbf{x}_0) \prod_{i}^{n} P(x_i \mid x_{i-1}, u_i) \prod_{k}^{m} P(z_k \mid x_{i_k}, l_{j_k})$$

Replace the multivariate normal distributions

$$\begin{aligned}\max\{P(X, L)\} \quad &= \quad \max \left\{ \prod_{k}^{m} \exp\left(-\frac{1}{2} \left\| h(x_{i_k}, l_{j_k}) - z_k \right\|_{\Sigma_z}^{2} \right) \right. \\ &\left. \prod_{i}^{n} \exp\left(-\frac{1}{2} \left\| f(x_{i-1}, u_i) - x_i \right\|_{\Sigma_u}^{2} \right) \right\}\end{aligned}$$

NIGHTMARE!!!

- log(x)

$$\operatorname{argmax}\left\{-\log\left(\prod_k^m \exp(r_k)\right)\right\} = \operatorname{argmin}\left\{\sum_k^m r_k\right\}$$

Makes everything easier!

$$\{L^*, X^*\} = \min\left\{\frac{1}{2}\sum_{k=1}^{m} \|\overbrace{h(x_{i_k}, l_{j_k}) - z_k}^{\text{errors}}\|^2_{\Sigma_z} + \sum_{i=1}^{n}\frac{1}{2}\|\underbrace{f(x_{i-1}, u_i) - x_i}_{\text{errors}}\|^2_{\Sigma_u}\right\}$$

Nonlinear Least Squares Problem

Nonlinear Least Squares

A standard nonlinear least squares

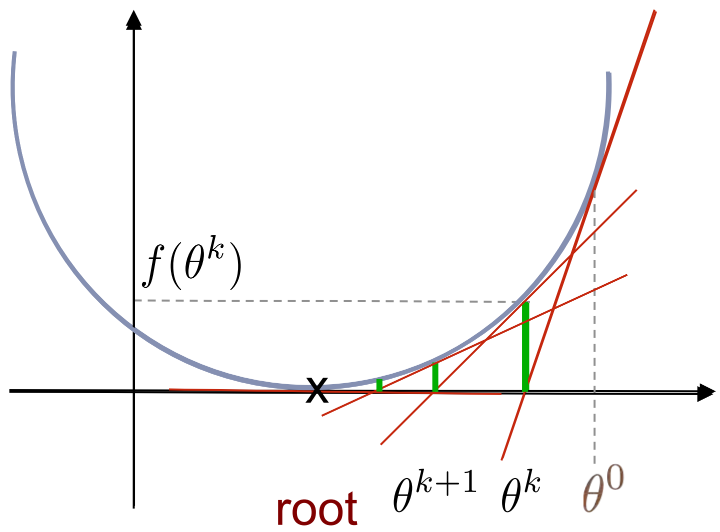
$$\boldsymbol{\theta} = \{L, X\}$$

stationary point $$\boldsymbol{\theta}^* \quad = \quad \min \{ \; F(\boldsymbol{\theta}) \; \}$$

$$F(\boldsymbol{\theta}) \quad = \quad \frac{1}{2} \sum_{k=1}^{m} \|\mathbf{r}_k(\boldsymbol{\theta})\|^2$$

Newton Method

Newton methods can be used to find the root of a function.



- Start with an initial estimate: θ^0
- Calculate the tangent in this point:

$$t(\theta) = f'(\theta^k)(\theta - \theta^k) + f(\theta^k)$$

- Find the intercept:

$$t(\theta^{k+1}) = 0$$

- Iterate:

$$\theta^{k+1} = \theta^k - \frac{f(\theta^k)}{f'(\theta^k)}$$

Newton Method in Optimization

For minimizing a nonlinear function, one applies Newton method to the **first derivative**.

$$f'(\theta^*) = 0$$

stationary point

$$f'(\theta^k) \propto f'(\theta^k) + f''(\theta^k)\,\Delta\theta = 0$$

$$\Delta\theta = \theta - \theta^k$$

$$\theta^{k+1} = \theta^k - \frac{f'(\theta^k)}{f''(\theta^k)}$$

Needs the second derivative

Nonlinear Least Squares

$$\boxed{F(\boldsymbol{\theta}) \quad = \quad \frac{1}{2}\sum_{k=1}^{m}\|\mathbf{r}_k(\boldsymbol{\theta})\|^2} \qquad \boldsymbol{\theta}^* \quad = \quad \min\{\, F(\boldsymbol{\theta}) \,\}$$

Nonlinear residuals: $\mathbf{r}(\boldsymbol{\theta}) = [r_1, \ldots, r_m]^\top$

Linearize: $\tilde{\mathbf{r}}(\boldsymbol{\theta}) = \mathbf{r}(\boldsymbol{\theta}^0) + J(\boldsymbol{\theta}^0)\underbrace{(\boldsymbol{\theta} - \boldsymbol{\theta}^0)}_{\text{correction } \boldsymbol{\delta}}$

Linear Least Squares:

$$\frac{1}{2}\sum_{k=1}^{m}\|r_{0_k} + J_k\delta_k\|^2 \quad = \quad \frac{1}{2}\|\mathbf{r_0}\|^2 + \boldsymbol{\delta}^\top J^\top \mathbf{r_0} + \frac{1}{2}\boldsymbol{\delta}^\top J^\top J\boldsymbol{\delta}$$

Linear Least Squares

We need to find the minimum of :

$$L(\boldsymbol{\delta}) \quad = \quad \frac{1}{2}\left\|\mathbf{r_0}\right\|^2 + \boldsymbol{\delta}^\top J^\top \mathbf{r_0} + \frac{1}{2}\boldsymbol{\delta}^\top J^\top J \boldsymbol{\delta}$$

1 st derivative:

$$L(\boldsymbol{\delta})' \quad = \quad J^\top \mathbf{r_0} + J^\top J \boldsymbol{\delta}$$

The minimum is where the 1 st derivative cancels

$$J^\top \mathbf{r_0} + J^\top J \boldsymbol{\delta} = 0$$

Correction:

$$\boldsymbol{\delta}^*$$

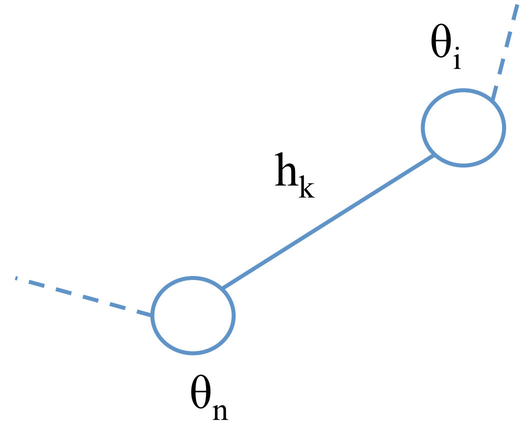
Jacobians and Hessians

$$L(\boldsymbol{\delta})' = \underbrace{J^\top \mathbf{r_0}}_{\text{Jacobian}} + \underbrace{J^\top J}_{\text{Hessian}} \boldsymbol{\delta}$$

$$J_k = \begin{bmatrix} \frac{\delta r_k}{\delta \theta_1} \\ \frac{\delta r_k}{\delta \theta_2} \\ \vdots \\ \frac{\delta r_k}{\delta \theta_n} \end{bmatrix} \quad H_k = \begin{bmatrix} \frac{\delta^2 r_k}{\delta \theta_1 \delta \theta_1} & \frac{\delta^2 r_k}{\delta \theta_1 \delta \theta_2} & \cdots & \frac{\delta^2 r_k}{\delta \theta_1 \delta \theta_n} \\ \frac{\delta^2 r_k}{\delta \theta_2 \delta \theta_1} & \frac{\delta^2 r_k}{\delta \theta_2 \delta \theta_2} & \cdots & \frac{\delta^2 r_k}{\delta \theta_2 \delta \theta_n} \\ \vdots & & & \\ \frac{\delta^2 r_k}{\delta \theta_n \delta \theta_1} & \frac{\delta^2 r_k}{\delta \theta_n \delta \theta_2} & \cdots & \frac{\delta^2 r_k}{\delta \theta_n \delta \theta_n} \end{bmatrix}$$

Jacobians and Hessians

Each measurement affects few variables (2 in general):



$$J_k = \begin{bmatrix} 0 \\ \vdots \\ \boxed{\dfrac{\delta r_k}{\delta \theta_i}} \\ 0 \\ \vdots \\ \boxed{\dfrac{\delta r_k}{\delta \theta_n}} \end{bmatrix} \qquad H_k = \begin{bmatrix} 0 & \dots & 0 & \dots & 0 \\ 0 & \dots & \boxed{\dfrac{\delta^2 r_k}{\delta \theta_i \delta \theta_i}} & 0 \dots 0 & \boxed{\dfrac{\delta^2 r_k}{\delta \theta_i \delta \theta_n}} \\ \vdots & & \vdots & & \vdots \\ 0 & \dots & \boxed{\dfrac{\delta^2 r_k}{\delta \theta_n \delta \theta_i}} & 0 \dots 0 & \boxed{\dfrac{\delta^2 r_k}{\delta \theta_n \delta \theta_n}} \end{bmatrix}$$

Gauss-Newton

$$F(\boldsymbol{\theta}) \quad = \quad \frac{1}{2}\sum_{k=1}^{m} \left\|\mathbf{r}_k(\boldsymbol{\theta})\right\|^2$$

while 1

linearize $F(\theta)$ in θ^i ➔ $L(\delta)$

solve $L(\delta)' = 0$ obtain δ^*

if norm(δ^*) < threshold

done

update $\theta^{i+1} = \theta^i + \delta^*$

SLAM - Solve

$$\boxed{L(\boldsymbol{\delta}) \quad = \quad \|\mathbf{b}\|^2 + \boldsymbol{\delta}^\top A^\top \mathbf{b} + \frac{1}{2}\boldsymbol{\delta}^\top A^\top A\boldsymbol{\delta}}$$

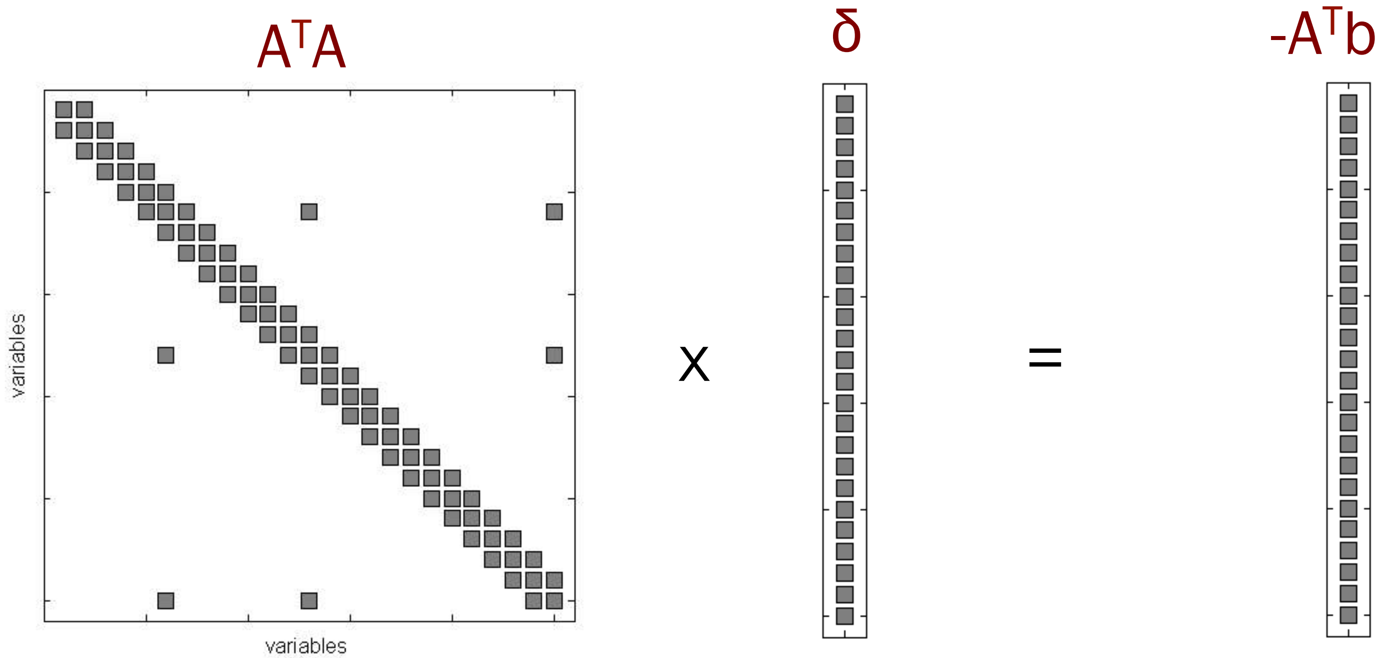
The min is where the first derivative cancels!

$$L(\boldsymbol{\delta})' \quad = \quad A^\top \mathbf{b} + A^\top A\boldsymbol{\delta} = 0$$

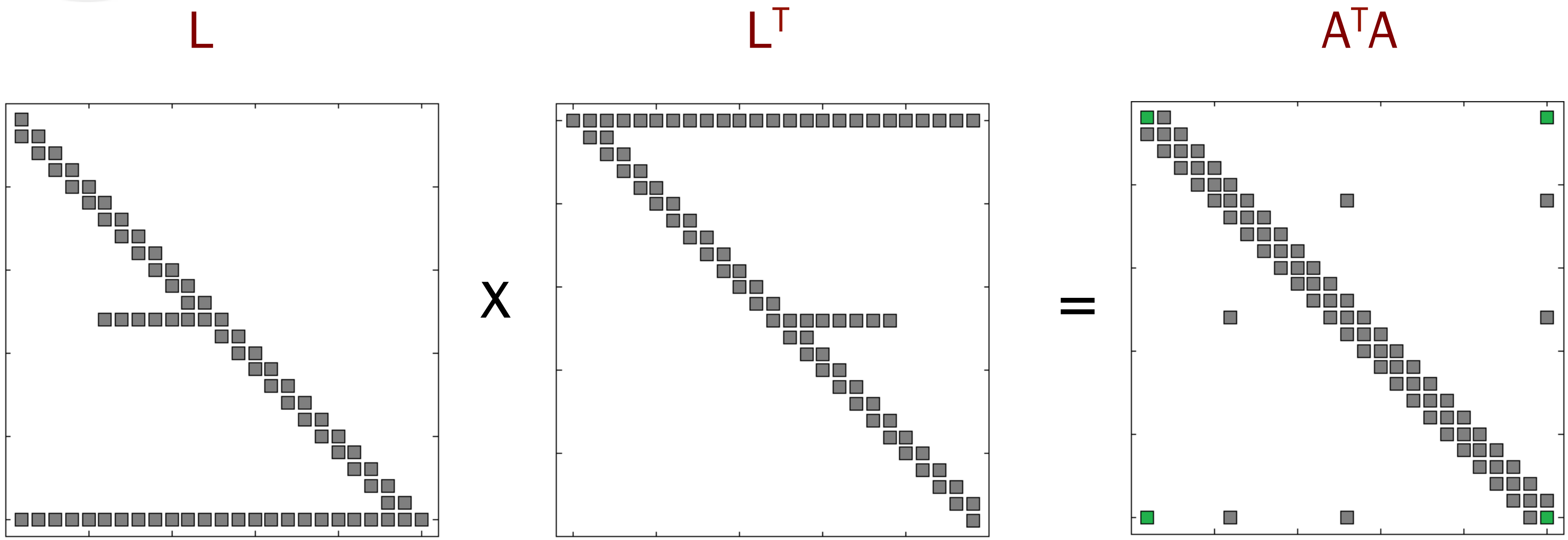
$$A^\top A\boldsymbol{\delta} = -A^\top \mathbf{b}$$

Normal Equation

$$\boxed{A^\top A\boldsymbol{\delta} = -A^\top \mathbf{b}}$$

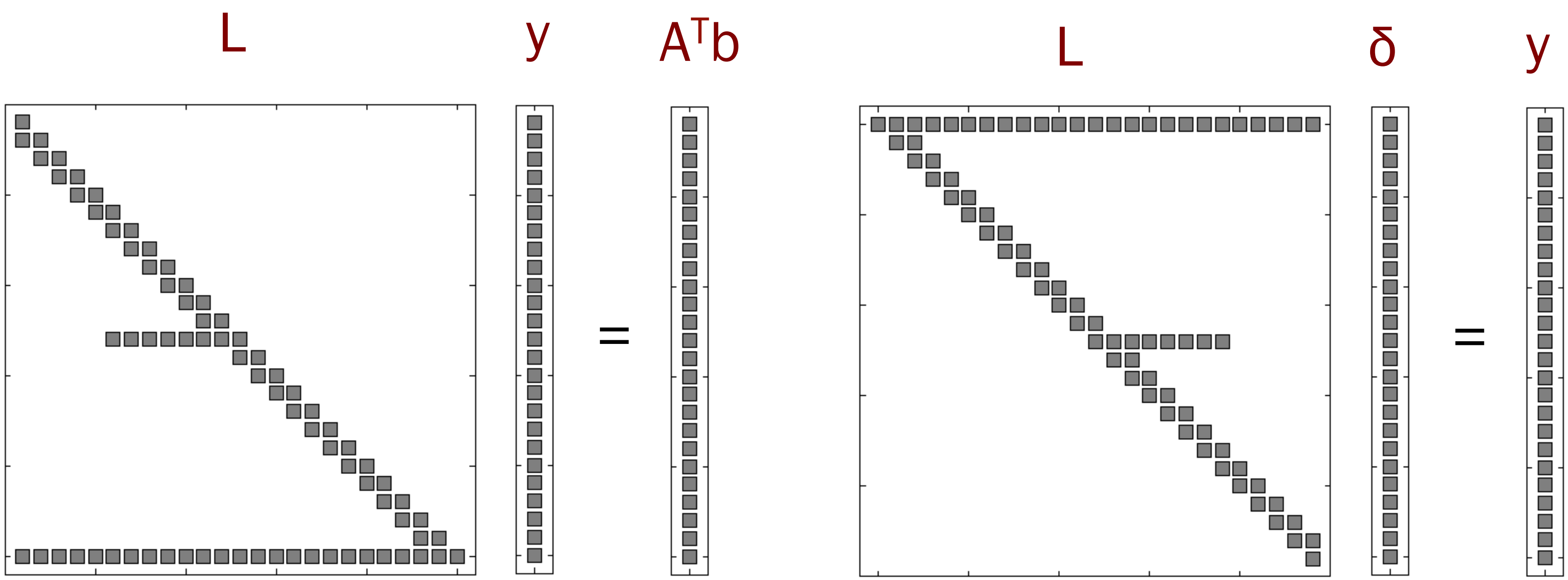


Matrix Factorization



- Symmetric positive definite matrix A^TA has Cholesky factorization $A^TA = LL^T$ where L is lower triangular matrix with positive diagonal entries.

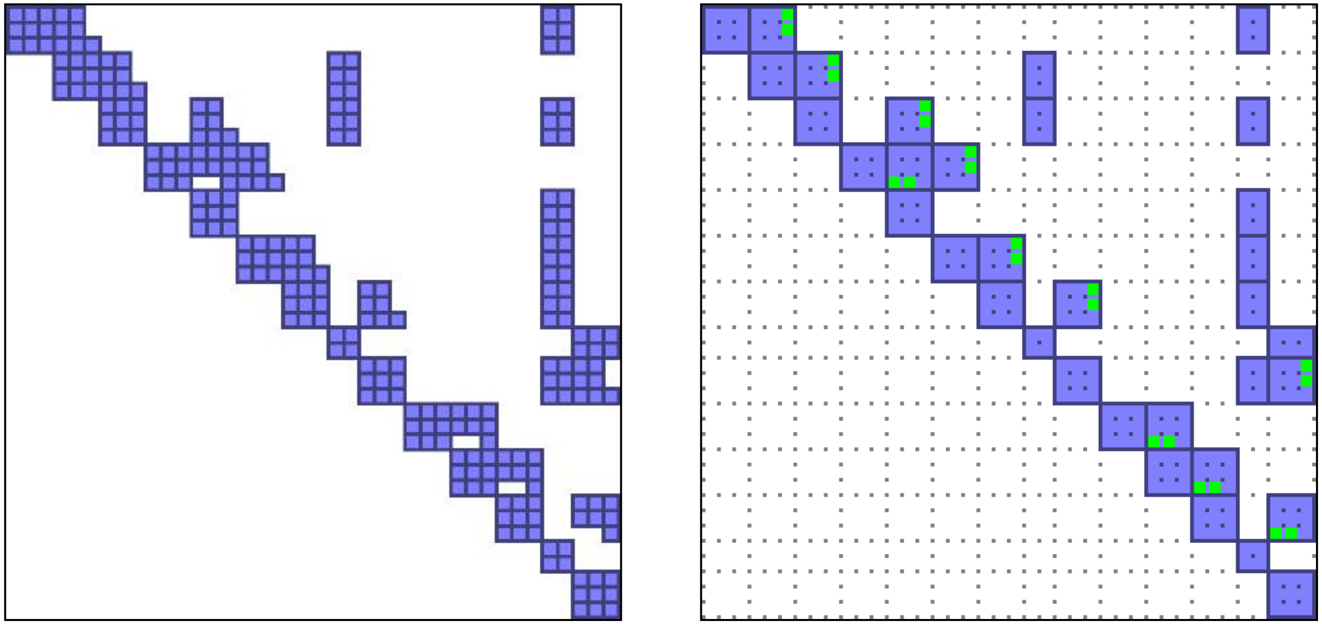
Matrix Factorization



- Linear system $A^TA\ \delta = A^Tb$ can then be solved by forward substitution in lower triangular system $Ly = A^Tb$, followed by back-substitution in upper triangular system $L^T\ \delta = y$

Sparse Matrices

- A matrix is called **sparse** if many of its entries are zero



- A **block matrix** is a matrix which is interpreted as partitioned into sections called blocks that can be manipulated at once

Sparse Algebra

http://faculty.cse.tamu.edu/davis/suitesparse.html

SLAM++

high-performance nonlinear least squares solver for graph problems

Brought to you by: iviorela, swajnautcz

http://sourceforge.net/projects/slam-plus-plus/

Covariance vs. Information Matrix

$$\Sigma_t = \begin{bmatrix} \Sigma_{0,0} & \Sigma_{0,1} & \dots & \Sigma_{0,t} \\ \Sigma_{1,0} & \Sigma_{1,1} & \dots & \Sigma_{1,t} \\ \dots & \dots & \dots & \dots \\ \Sigma_{t,0} & \Sigma_{t,1} & \dots & \Sigma_{t,t} \end{bmatrix}$$

Covariance - Dense matrix

$$\boxed{\Lambda_t = \Sigma_t^{-1}}$$

$$\Lambda_t = \begin{bmatrix} \Lambda_{0,0} & \Lambda_{0,1} & \dots & 0 \\ \Lambda_{1,0} & \Lambda_{1,1} & \dots & 0 \\ \dots & \dots & \dots & \dots \\ 0 & \dots & \Lambda_{t-1,t-1} & \Lambda_{t-1,t} \\ 0 & \dots & \Lambda_{t,t-1} & \Lambda_{t,t} \end{bmatrix}$$

Information - Sparse matrix

AUSTRALIAN CENTRE FOR ROBOTIC VISION

Australian Government
Australian Research Council

QUT

THE UNIVERSITY of ADELAIDE

THE AUSTRALIAN NATIONAL UNIVERSITY

MONASH University

CSIRO

UNIVERSITY OF OXFORD

Inria
INVENTORS FOR THE DIGITAL WORLD

Imperial College London

ETH zürich

Georgia Tech