Slide credit: https://cvg.ethz.ch/teaching/3dvision/slides/class06eth18_annotated.pdf

# CS 395T
# Lecture 13: Optimization for SLAM


Qixing Huang

October 13st 2018

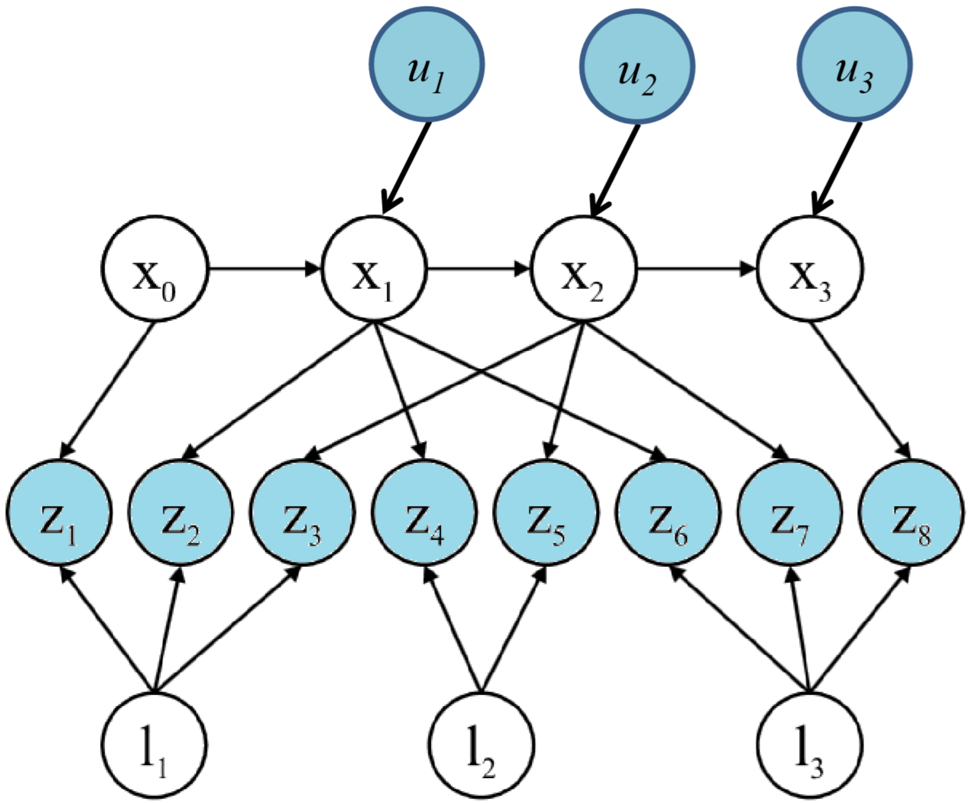
# Simultaneous Localization & Mapping (SLAM)

- Robot navigates in unknown environment:
  - Estimate its own pose
  - Acquire a map model of its environment

- Chicken-and-Egg problem:
  - Map is needed for localization (pose estimation)
  - Pose is needed for mapping

- Highly related to Structure-From-Motion

# Full SLAM: Problem Definition



# Full SLAM: Problem Definition

- Maximum a Posteriori (MAP) solution:

$$\begin{aligned}\underset{X,L}{\operatorname{argmax}}\, P(X,L|Z,U) &= \underset{X,L}{\operatorname{argmax}} P(X_0)\cdot\prod_{i=1}^{M} P(\boldsymbol{x}_i|\boldsymbol{x}_{i-1},\boldsymbol{u}_i)\prod_{i=1}^{K} P(z_{ik}|x_{ik},l_{jk})\\ &= -\underset{X,L}{\operatorname{argmin}}\sum_{i=1}^{M}\log\big(P(\boldsymbol{x}_i|\boldsymbol{x}_{i-1},\boldsymbol{u}_i)\big)+\sum_{i=1}^{K}\log\big(P(\boldsymbol{z}_k|x_{ik},l_{jk})\big)\end{aligned}$$

Likelihoods:

$$P(\boldsymbol{x}_i|\boldsymbol{x}_{i-1},\boldsymbol{u}_i)\propto\exp\big(-\|f(\boldsymbol{x}_{i-1},\boldsymbol{u}_i)-\boldsymbol{x}_i\|^2_{\Lambda_i}\big)$$

Process model

$$P(\boldsymbol{z}_k|x_{ik},l_{jk})\propto\exp\big(-\|h(x_{ik},l_{jk})-\boldsymbol{z}_k\|^2_{\Sigma_k}\big)$$

Measurement model

# Full SLAM

$$\underset{X,L}{\operatorname{argmax}}\ P(X,L|Z,U) = \underset{X,L}{\operatorname{argmax}} P(X_0) \cdot \prod_{i=1}^{M} P(\boldsymbol{x}_i|\boldsymbol{x}_{i-1},\boldsymbol{u}_i) \prod_{i=1}^{K} P(z_{ik}|x_{ik},l_{jk})$$
$$= -\underset{X,L}{\operatorname{argmin}} \sum_{i=1}^{M} \log\left(P(\boldsymbol{x}_i|\boldsymbol{x}_{i-1},\boldsymbol{u}_i)\right) + \sum_{i=1}^{K} \log\left(P(\boldsymbol{z}_k|x_{ik},l_{jk})\right)$$

- Putting the likelihoods into the equation:

$$\underset{X,L}{\operatorname{argmax}}\ P(X,L|Z,U) = \underset{X,L}{\operatorname{argmin}} \left( \sum_{i=1}^{M} \|f(\boldsymbol{x}_{i-1},\boldsymbol{u}_i) - \boldsymbol{x}_i\|^2_{\Lambda_i} + \sum_{k=1}^{K} \|h(x_{ik},l_{jk}) - \boldsymbol{z}_k\|^2_{\Sigma_k} \right)$$

Minimization can be done with Levenberg-Marquardt
(similar to bundle adjustment problem)!

# Full SLAM

Normal Equations:

Weight made up of $\Lambda_i, \Sigma_k$

$$(J^T W J + \lambda I)\delta = -J^T W d\boldsymbol{z}$$

Jacobian made up of $\frac{\partial f}{\partial \boldsymbol{x}}, \frac{\partial f}{\partial \boldsymbol{u}}, \frac{\partial h}{\partial \boldsymbol{x}}, \frac{\partial h}{\partial l}$

Can be solved with sparse matrix factorization or iterative methods

# Online SLAM: Problem Definition

- Estimate current pose $x_t$ and full map $L$ :

$$P(\boldsymbol{x}_t, L|Z, U) = \int\int \cdots \int P(X, L|Z, U) d\boldsymbol{x}_1 d\boldsymbol{x}_2 \cdots d\boldsymbol{x}_{t-1}$$

- Inference with:
  - (Extended) KalmanFilter (EKF SLAM)
  - Particle Filter (FastSLAM)

# EKF SLAM

- Assumes: pose $x_t$ and map *L* are random variables that follow Gaussian distributions
- Hence,

$$P(\boldsymbol{x}_t, L | Z, U) \sim \mathcal{N}(\mu, \Sigma)$$

Mean

Error covariance

- (Extended) Kalman Filter iteratively
  - Predicts pose & map based on process model
  - Corrects prediction based on observations

# EKF SLAM

- Prediction:

$$\overline{\mu}_t = f(\boldsymbol{u}_t, \mu_{t-1})$$ Process model

$$\overline{\Sigma}_t = F_t \Sigma_{t-1} F_t^T + R_t$$ Error propagation with process noise

- Correction:

$$\boldsymbol{y}_t = \boldsymbol{z}_t - h(\overline{\mu}_t)$$ Measurement residual (innovation)

$$K_t = \overline{\Sigma}_t H_t^T \left( H_t \overline{\Sigma}_t H_t^T + Q_t \right)^{-1}$$ Kalman gain

$$\mu_t = \overline{\mu}_t + K_t \boldsymbol{y}_t$$ Update mean

$$\Sigma_t = (I - K_t H_t) \overline{\Sigma}_t$$ Update covariance

Measurement Jacobian $H_t = \dfrac{\partial h(\overline{\mu}_t)}{\partial \boldsymbol{x}_t}$ Process Jacobian $F_t = \dfrac{\partial f(\boldsymbol{u}_t, \mu_{t-1})}{\partial \boldsymbol{x}_{t-1}}$

# Structure of Mean and Covariance

$$\mu_t = \begin{pmatrix} x \\ y \\ \theta \\ l_1 \\ l_2 \\ \vdots \\ l_N \end{pmatrix}, \quad \Sigma_t = \begin{pmatrix} \sigma_x^2 & \sigma_{xy} & \sigma_{x\theta} & \sigma_{xl_1} & \sigma_{xl_2} & \cdots & \sigma_{xl_N} \\ \sigma_{xy} & \sigma_y^2 & \sigma_{y\theta} & \sigma_{yl_1} & \sigma_{yl_2} & \cdots & \sigma_{yl_N} \\ \sigma_{x\theta} & \sigma_{y\theta} & \sigma_\theta^2 & \sigma_{\theta l_1} & \sigma_{\theta l_2} & \cdots & \sigma_{\theta l_N} \\ \sigma_{xl_1} & \sigma_{yl_1} & \sigma_{\theta l_1} & \sigma_{l_1}^2 & \sigma_{l_1 l_2} & \cdots & \sigma_{l_1 l_N} \\ \sigma_{xl_2} & \sigma_{yl_2} & \sigma_{\theta l_2} & \sigma_{l_1 l_2} & \sigma_{l_2}^2 & \cdots & \sigma_{l_2 l_N} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ \sigma_{xl_N} & \sigma_{yl_N} & \sigma_{\theta l_N} & \sigma_{l_1 l_N} & \sigma_{l_2 l_N} & \cdots & \sigma_{l_N}^2 \end{pmatrix}$$

Covariance is a dense matrix that grows with increasing map features!

True robot and map states might not follow unimodal Gaussian distribution!

# Particle Filtering: FastSLAM

- Particles represents samples from the posterior distribution $P(x_t, L|Z,U)$
- $P(x_t, L|Z,U)$ can be any distribution (need not be Gaussian)

# FastSLAM

- Each particle represents:

$$p_t^m = \{ \underset{\text{Robot state}}{x_t^m}, < \mu_{1,t}^m, \Sigma_{1,t}^m >, \underbrace{< \mu_{2,t}^m, \Sigma_{2,t}^m >}_{\substack{\text{Landmark state} \\ \text{(mean and covariance)}}} \ldots < \mu_{N,t}^m, \Sigma_{N,t}^m > \}$$

$$x_t^m \sim p(x_t \mid x_{t-1}, u_t)$$ ← Sample the robot state from the process model

$$p(L_{n,t}^m \mid x_t^m, z_t)$$ ← N Kalman filter Landmark updates

$$w_t^m \propto p(z_t \mid L_t^m, x_t^m)$$ ← Weight update

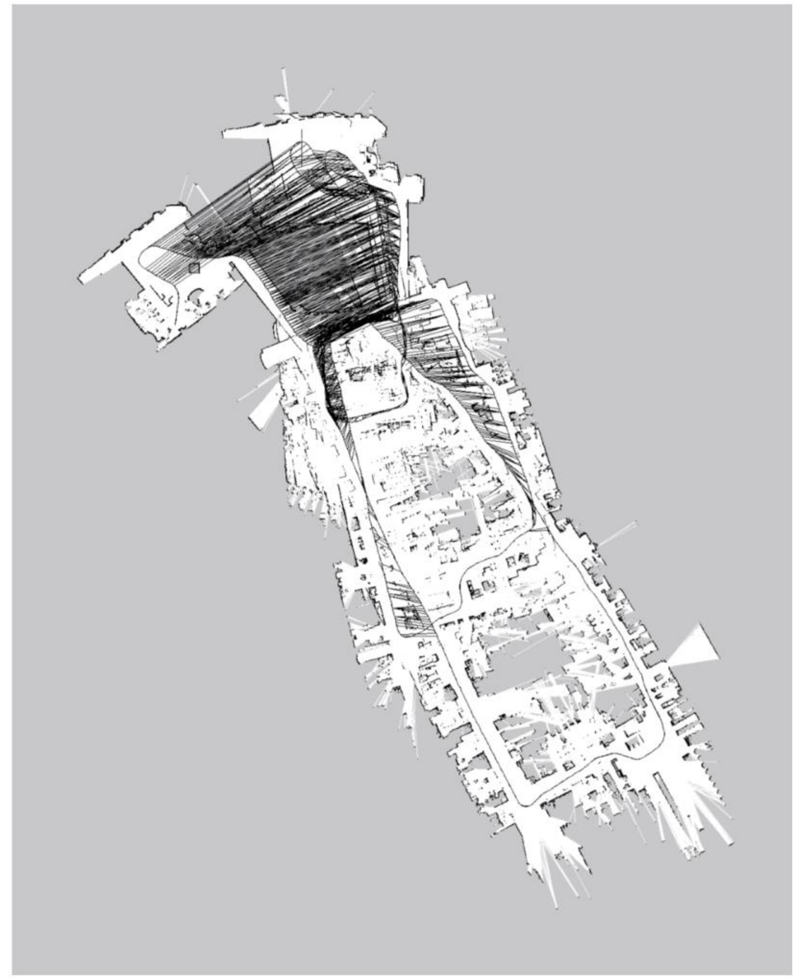
Resampling based on current state

# FastSLAM

- Many particles needed for accurate results
- Computationally expensive for high state dimensions

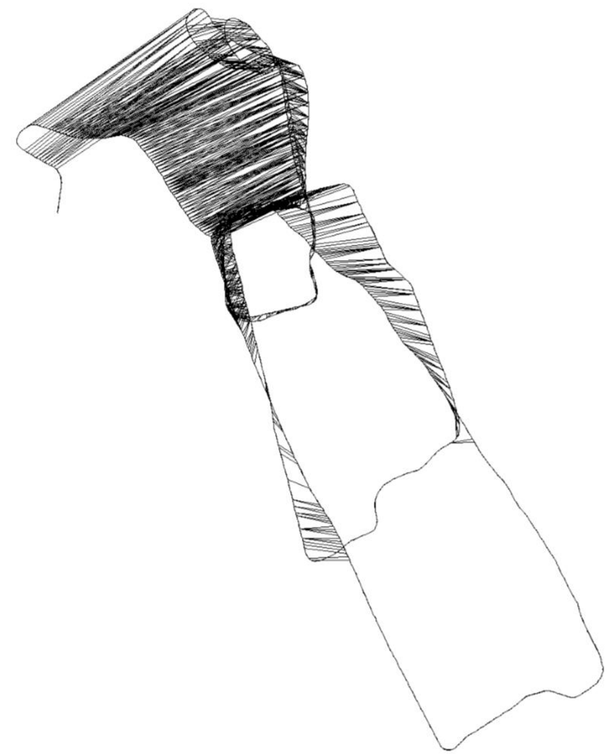
# Pose-Graph SLAM

- Every node in the graph corresponds to a robot position and a laser measurement
- An edge between two nodes represents a spatial constraint between the nodes



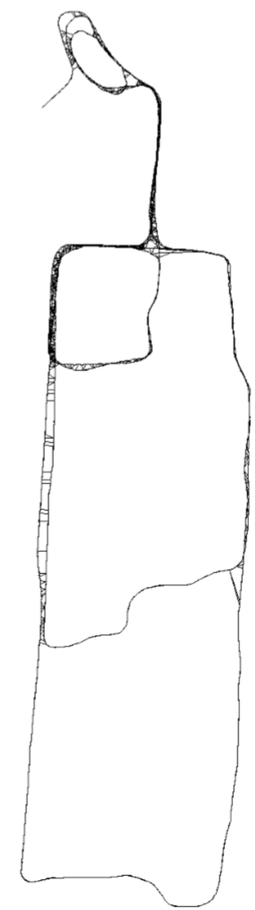
# Pose-Graph SLAM

- Once we have the graph, we determine the most likely map by correcting the nodes

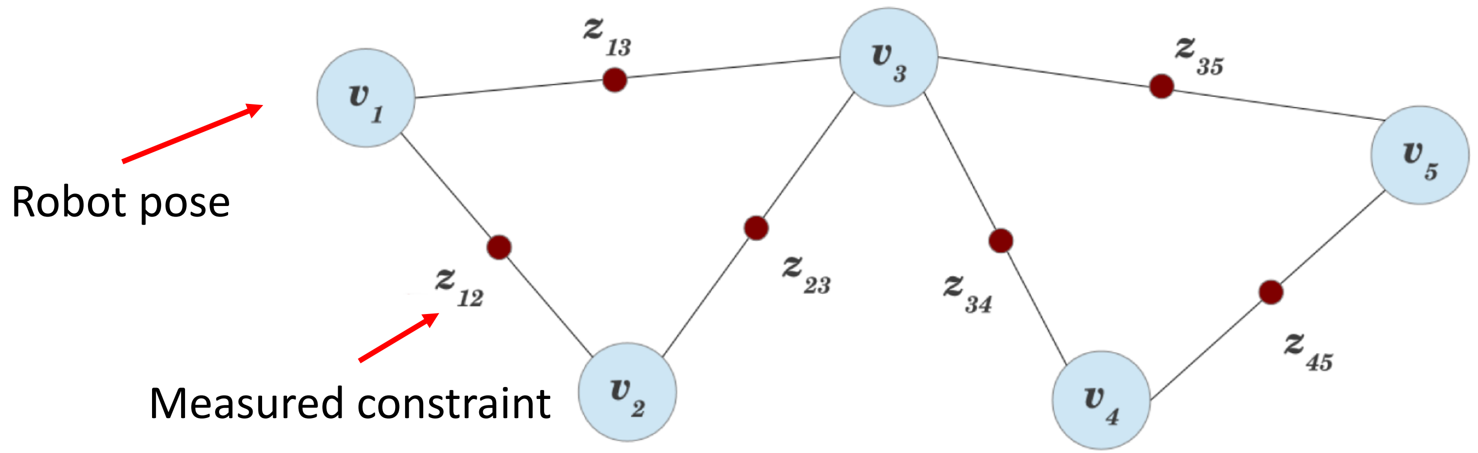


# Pose-Graph SLAM

- Once we have the graph, we determine the most likely map by correcting the nodes



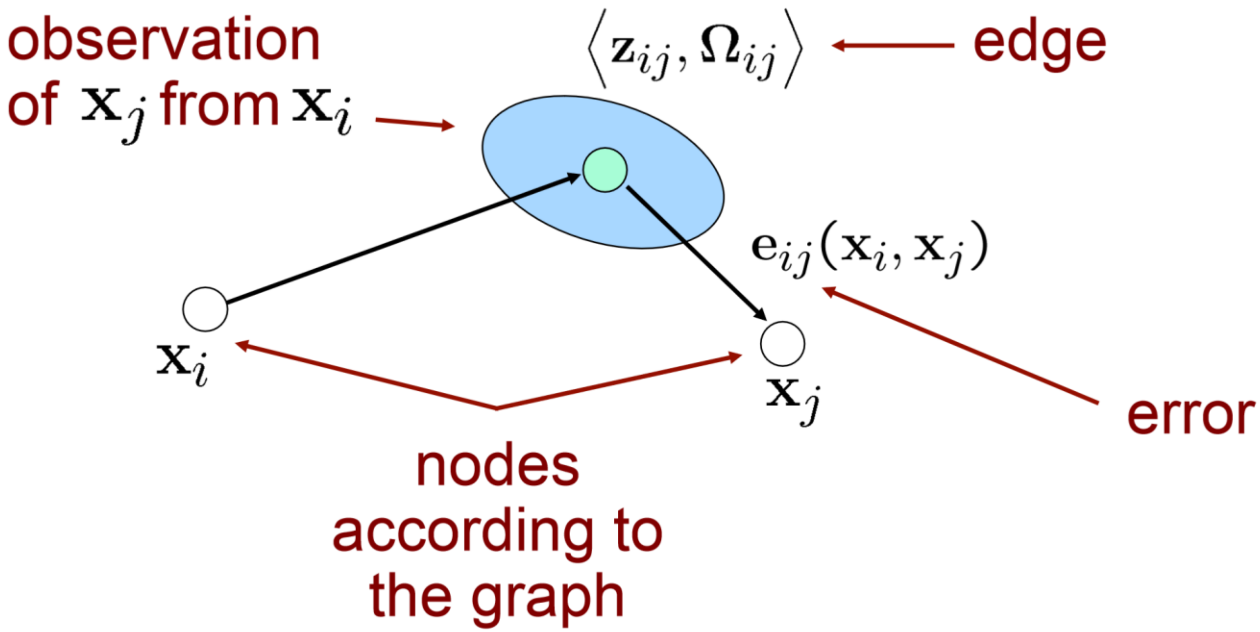
# Pose-Graph SLAM




- Constraints: Relative pose estimates from 3D structure
- Don't update 3D structure (fixed wrt. to some pose)
- Optimizes poses as $\underset{X}{\operatorname{argmin}} \sum_{ij} \| \boldsymbol{z}_{ij} - h(\boldsymbol{v}_i, \boldsymbol{v}_j) \|^2_{\Sigma_{ij}}$

- Can be used to minimize loop-closure errors

# Pose-Graph SLAM




- **Goal:** $\mathbf{x}^* = \operatorname*{argmin}_{\mathbf{x}} \sum_{ij} \mathbf{e}_{ij}^T \mathbf{\Omega}_{ij} \mathbf{e}_{ij}$

# Gauss-Newton +LM

- Define the error function

- Linearize the error function
- Compute its derivative
- Set the derivative to zero

- Solve the linear system
- Iterate this procedure until convergence