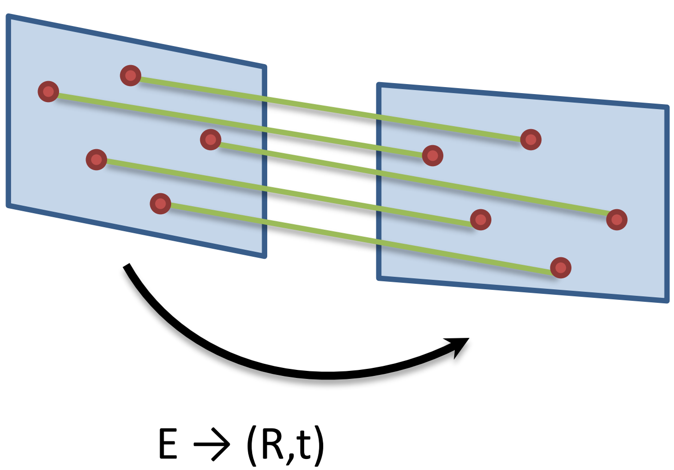
3D Photography: Bundle Adjustment and SLAM

31 March 2014

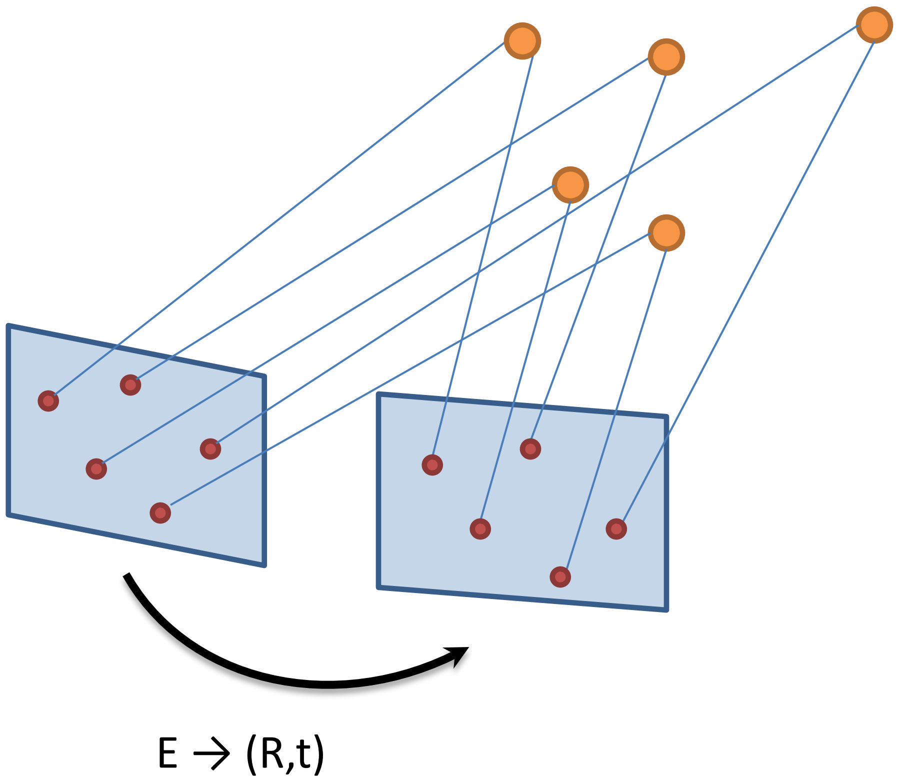
Structure-From-Motion

- Two views initialization:
  - 5-Point algorithm (Minimal Solver)
  - 8-Point linear algorithm
  - 7-Point algorithm



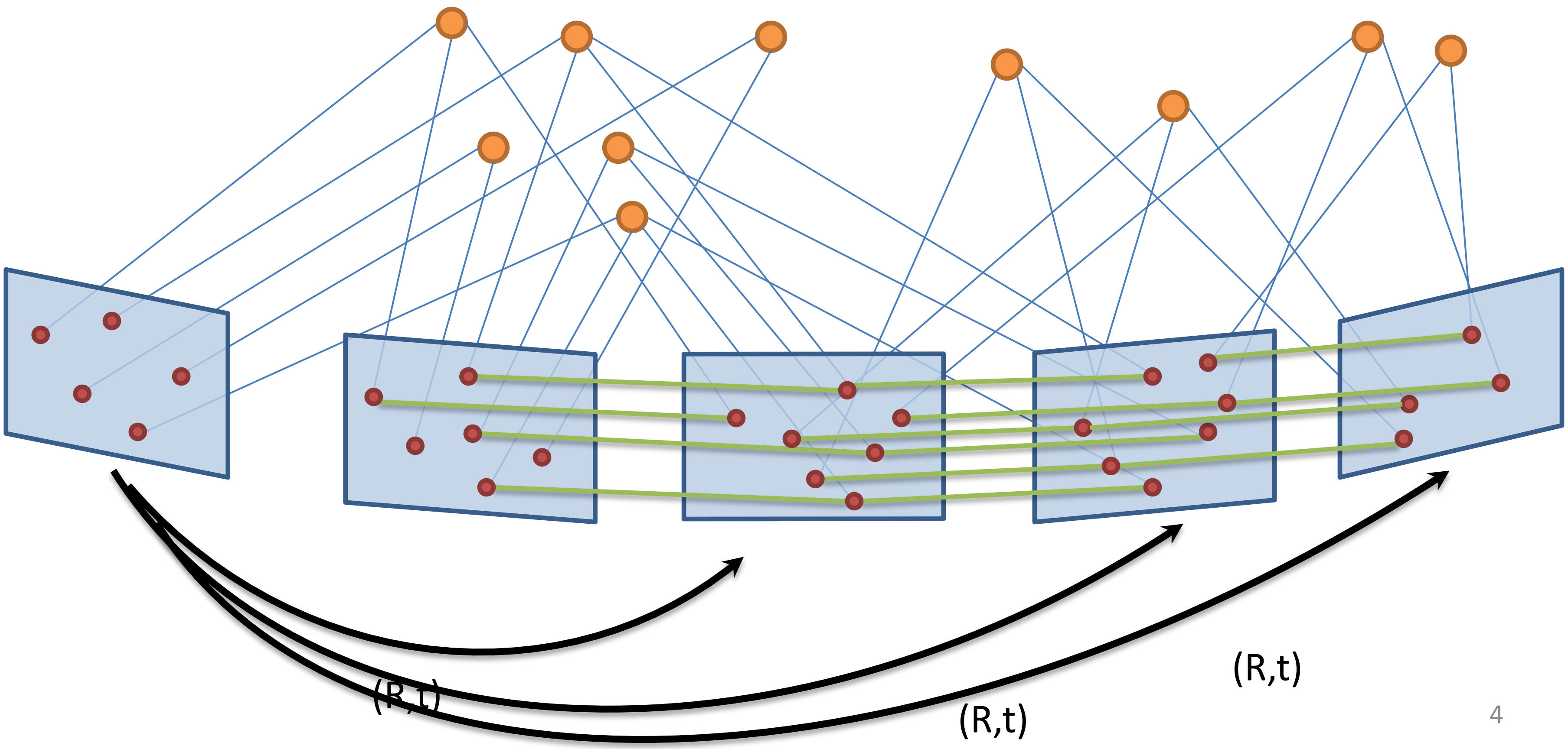
Structure-From-Motion

- Triangulation: 3D Points



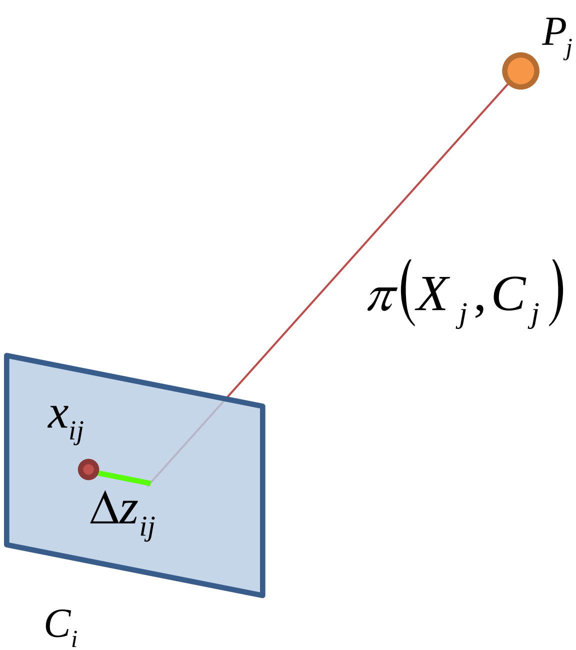
Structure-From-Motion

- Subsequent views: Perspective pose estimation



Bundle Adjustment

- Final step in Structure-from-Motion.
- Refine a visual reconstruction to produce jointly optimal 3D structures P and camera poses C.
- Minimize total reprojection errors Δz.



Cost Function:

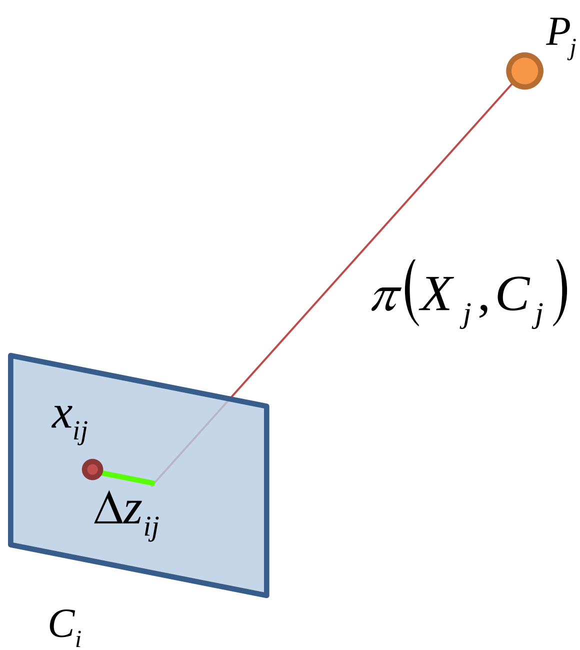
$$\underset{X}{\operatorname{argmin}} \sum_i \sum_j \| \underbrace{x_{ij} - \pi(P_j, C_i)}_{\Delta z_{ij}} \|_{W_{ij}}^2$$

W_{ij}^{-1} : Measurement error covariance

$X = [P, C]$

Bundle Adjustment

- Final step in Structure-from-Motion.
- Refine a visual reconstruction to produce jointly optimal 3D structures P and camera poses C.
- Minimize total reprojection errors Δz.



Cost Function:

$$\underset{X}{\operatorname{argmin}} \underbrace{\sum_i \sum_j \Delta z_{ij}^T W_{ij} \Delta z_{ij}}_{f(X)}$$

W_{ij}^{-1} : Measurement error covariance

$$X = [P, C]$$

Bundle Adjustment

- Minimize the cost function: $\underset{X}{\operatorname{argmin}}\ f(X)$
    1. Gradient Descent
    2. Newton Method
    3. Gauss-Newton
    4. Levenberg-Marquardt

Bundle Adjustment

1. Gradient Descent

Initialization: $X_k = X_0$

Iterate until convergence

Compute gradient: $g = \left.\dfrac{\partial f(X)}{\partial X}\right|_{X=X_k} = \Delta Z^T W J$

Update: $X_k \leftarrow X_k - \eta g$

η : Step size

$J = \dfrac{\partial \pi}{\partial X}$: Jacobian

Slow convergence near minimum point!

Bundle Adjustment

2. Newton Method

2^{nd} order approximation (Quadratic Taylor Expansion):

$$f(X+\delta)\big|_{X=X_K} \approx f(X) + g\delta + \tfrac{1}{2}\delta^T H\delta\big|_{X=X_K}$$

Hessian matrix $: H = \left.\dfrac{\partial^2 f\left(X+\delta\right)}{\partial\delta^2}\right|_{X=X_k}$

Find δ that minimizes $f(X+\delta)\big|_{X=X_K}$!

Bundle Adjustment

2. Newton Method

Differentiate and set to 0 gives:

$$\delta = -H^{-1}g$$

Update: $X_k \leftarrow X_k + \delta$

Computation of H is not trivial and might get stuck at saddle point!

Bundle Adjustment

3. Gauss-Newton

$$H = J^T W J + \cancel{\sum_i \sum_j \Delta Z_{ij} W_{ij} \frac{\partial^2 \pi_{ij}}{\partial X^2}}$$

$$\Downarrow$$

$$H \approx J^T W J$$

Normal equation:

$$J^T W J \delta = -J^T W \Delta Z$$

Update: $X_k \leftarrow X_k + \delta$

Might get stuck and slow convergence at saddle point !

Bundle Adjustment

4. Levenberg-Marquardt

Regularized Gauss-Newton with damping factor λ .

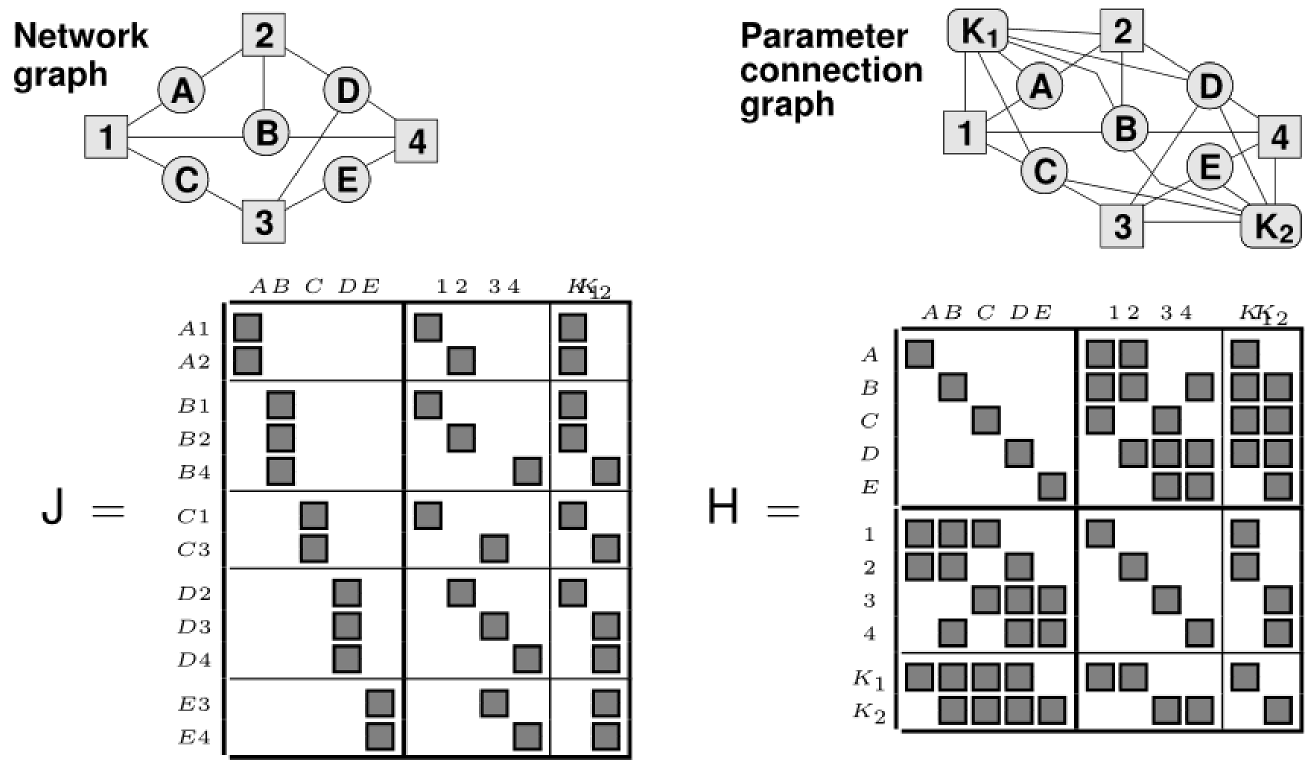
$$\underbrace{\left(J^T W J + \lambda I\right)}_{H_{LM}} \delta = -J^T W \Delta Z$$

$\lambda \to 0$: Gauss-Newton (when convergence is rapid)

$\lambda \to \infty$: Gradient descent (when convergence is slow)

Structure of the Jacobian and Hessian Matrices

- Sparse matrices since 3D structures are locally observed.

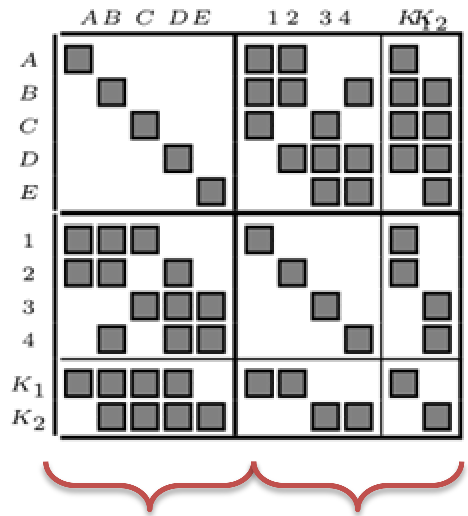


Solving the Normal Equation

- Schur Complement

$$H_{LM}\delta = -J^T W \Delta Z$$

$$H_{LM} =$$



3D Structures

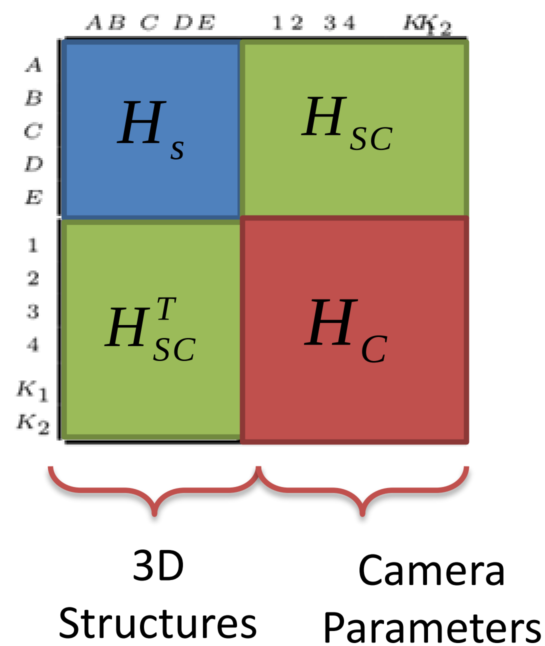
Camera Parameters

Solving the Normal Equation

- Schur Complement

$$H_{LM}\delta = -J^{T}W\Delta Z$$

$$H_{LM} =$$



Solving the Normal Equation

- Schur Complement

$$H_{LM}\delta = -J^T W \Delta Z$$

$$\begin{bmatrix} H_S & H_{SC} \\ H_{SC}^T & H_C \end{bmatrix}\begin{bmatrix} \delta_S \\ \delta_C \end{bmatrix} = \begin{bmatrix} \varepsilon_S \\ \varepsilon_C \end{bmatrix} \begin{array}{l} \leftarrow \text{3D Structures} \\ \leftarrow \text{Camera Parameters} \end{array}$$

Multiply both sides by: $\begin{bmatrix} I & 0 \\ -H_{SC}^T H_S^{-1} & I \end{bmatrix}$

$$\begin{bmatrix} H_S & H_{SC} \\ 0 & H_C - H_{SC}^T H_S^{-1} H_{SC} \end{bmatrix}\begin{bmatrix} \delta_S \\ \delta_C \end{bmatrix} = \begin{bmatrix} \varepsilon_S \\ \varepsilon_C - \varepsilon_S H_{SC}^T H_S^{-1} \end{bmatrix}$$

Solving the Normal Equation

- Schur Complement

$$\begin{bmatrix} H_S & H_{SC} \\ 0 & H_C - H_{SC}^T H_S^{-1} H_{SC} \end{bmatrix} \begin{bmatrix} \delta_S \\ \delta_C \end{bmatrix} = \begin{bmatrix} \varepsilon_S \\ \varepsilon_C - \varepsilon_S H_{SC}^T H_S^{-1} \end{bmatrix}$$

First solve for δ_C from:

Easy to invert a block diagonal matrix

$$\underbrace{(H_C - H_{SC}^T H_S^{-1} H_{SC})}_{\substack{\text{Schur Complement} \\ \text{(Sparse and Symmetric Positive Definite Matrix)}}} \delta_C = \varepsilon_C - \varepsilon_S H_{SC}^T H_S^{-1}$$

Solve for δ_{SC} by backward substitution.

Solving the Normal Equation

$$(H_C - H_{SC}^T H_S^{-1} H_{SC})\delta_C = \varepsilon_C - \varepsilon_S H_{SC}^T H_S^{-1} \quad \equiv \quad Ax = b$$

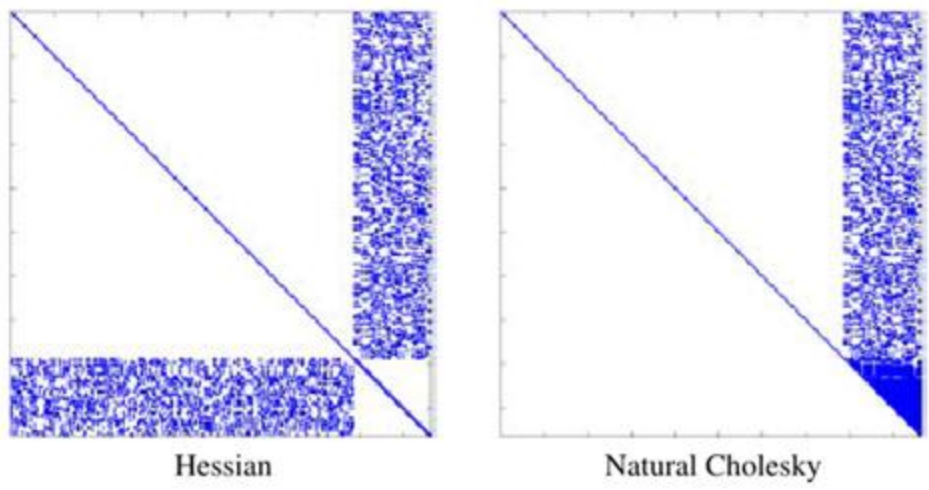
Can be solved without inverting A since it is a sparse matrix!

- Sparse matrix factorization
  1. LU Factorization $\longrightarrow$ $A = LU$
  2. QR factorization $\longrightarrow$ $A = QR$
  3. Cholesky Factorization $\longrightarrow$ $A = LL^T$

Solve for x by forward backward substitutions.

- Iterative methods
  1. Conjugate gradient
  2. Gauss-Seidel

Problem of Fill-In



Problem of Fill-In

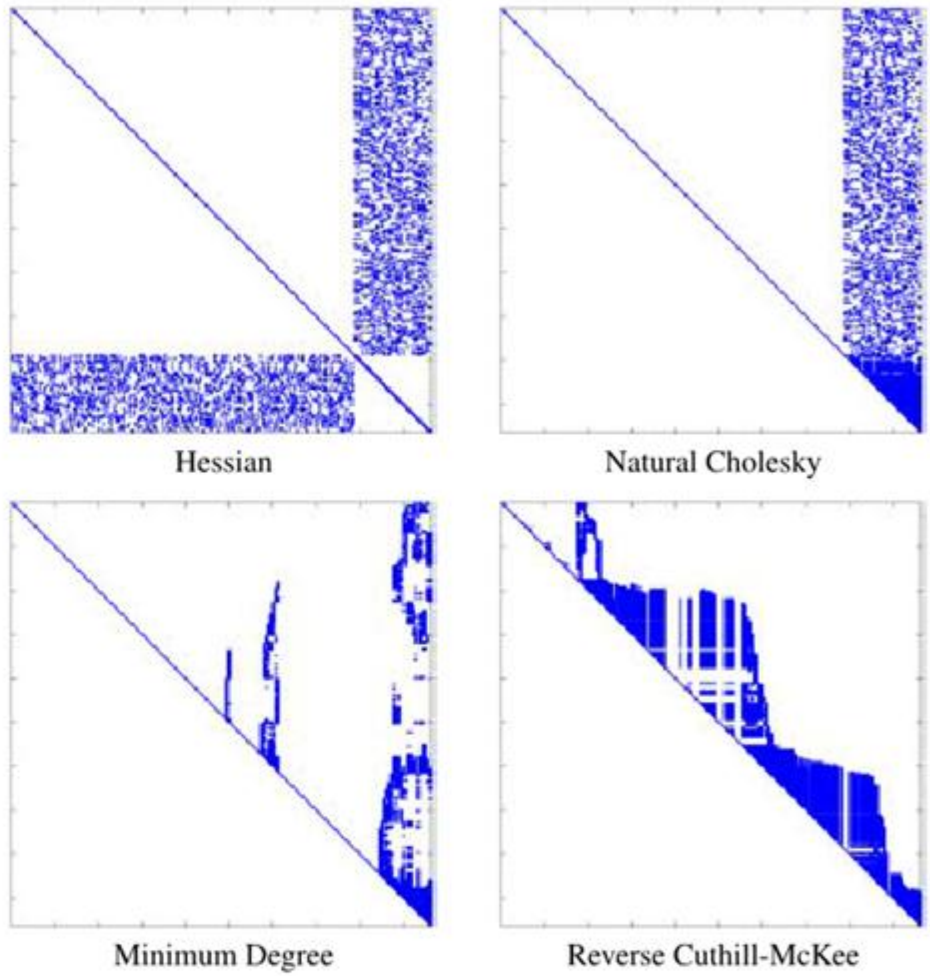
- Reorder sparse matrix to minimize fill-in.

$$\left(P^T A P\right)\left(P^T x\right) = P^T b$$

Permutation matrix to reorder A

- NP-Complete problem.
- Approximate solutions:
    1. Minimum degree
    2. Column approximate minimum degree permutation
    3. Reverse Cuthill-Mckee.
    4. …

Problem of Fill-In



Robust Cost Function

- Non-linear least squares: $\underset{X}{\operatorname{argmin}} \sum_{ij} \Delta z_{ij}^T W_{ij} \Delta z_{ij}$
- Maximum log-likelihood solution:

$$-\underset{X}{\operatorname{argmin}} \ln p(Z \mid X)$$

- Assume that:
  1. X is a random variable that follows Gaussian distribution.
  2. All observations are independent.

$$-\underset{X}{\operatorname{argmin}} \ln p(X \mid Z) = -\underset{X}{\operatorname{argmin}} \ln \left\{ \prod_{ij} c_{ij} \exp\left(-\Delta z_{ij}^T W_{ij} \Delta z_{ij} \right) \right\}$$

$$= \underset{X}{\operatorname{argmin}} \sum_{ij} \Delta z_{ij}^T W_{ij} \Delta z_{ij}$$

Robust Cost Function

- Gaussian distribution assumption is not true in the presence of outliers!
- Causes wrong convergences.

Robust Cost Function

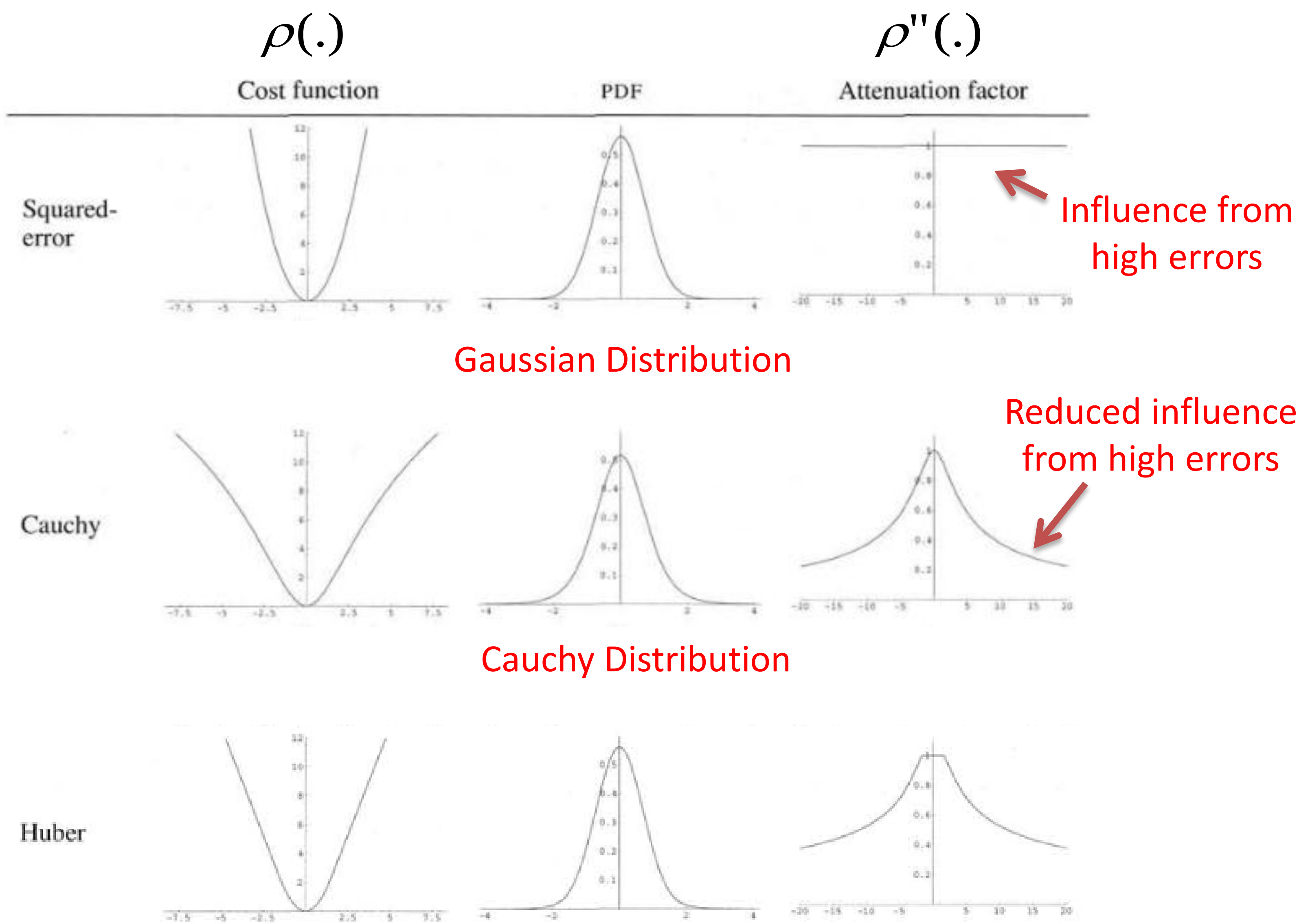
$$\underset{X}{\operatorname{argmin}} \sum_{ij} \rho_{ij}\left(\Delta z_{ij}\right) \equiv \underset{X}{\operatorname{argmin}} \sum_{ij} \Delta z_{ij}^{T} S_{ij} \Delta z_{ij}$$

Robust Cost Function

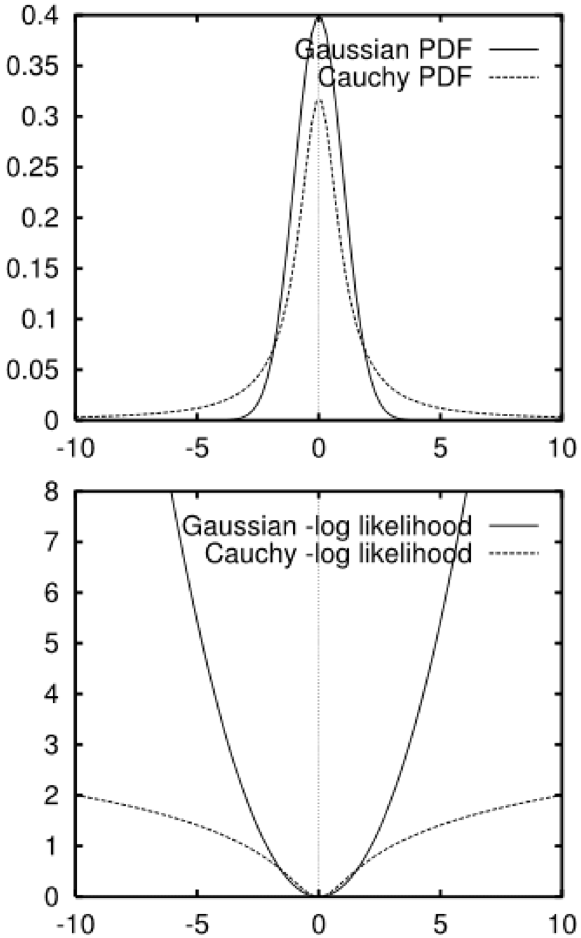
W_{ij} scaled with ρ''_{ij}

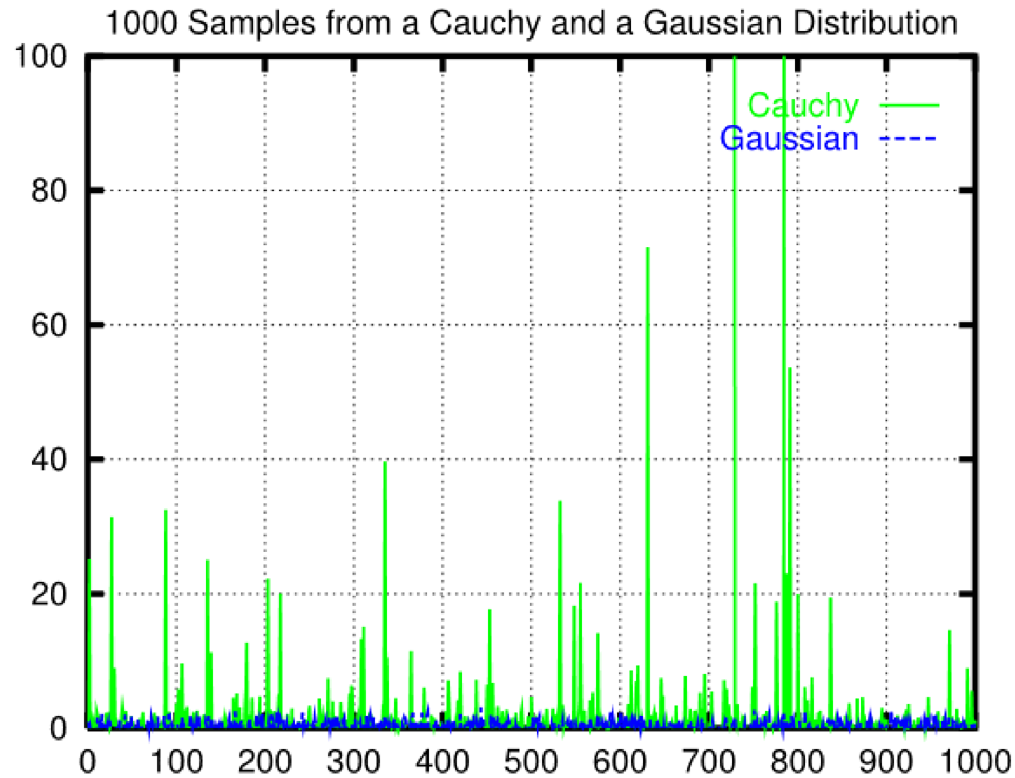
- Similar to iteratively re-weighted least-squares.
- Weight is iteratively rescaled with the attenuating factor ρ''_{ij} .
- Attenuating factor is computed based on current error.

Robust Cost Function



Robust Cost Function





Outliers are taken into account in Cauchy!

State-of-the-Art Solvers

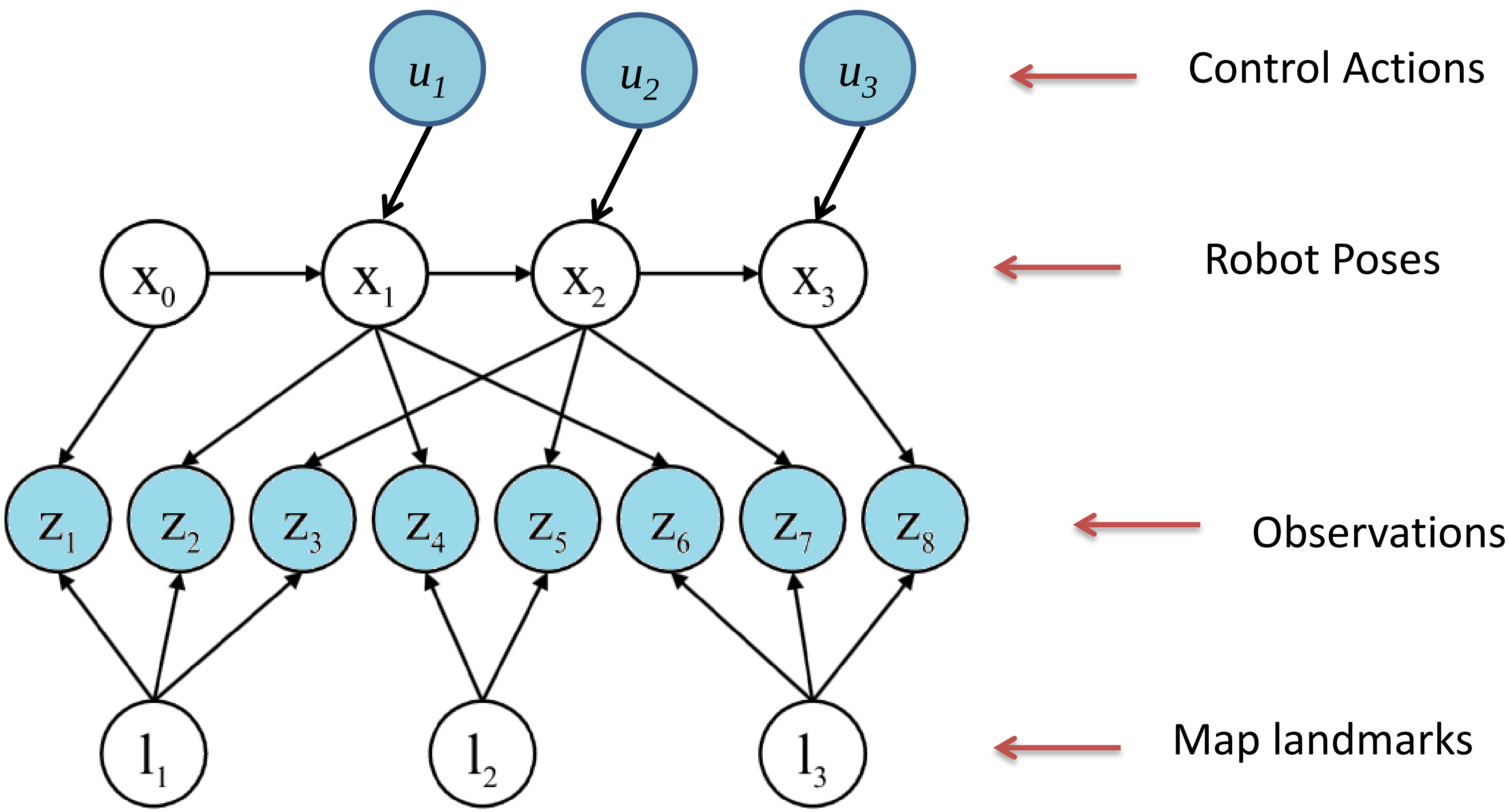
- Google Ceres:
  - https://code.google.com/p/ceres-solver/
- g2o:
  - https://openslam.org/g2o.html
- GTSAM:
  - https://collab.cc.gatech.edu/borg/gtsam/

Simultaneous Localization and Mapping (SLAM)

- For a robot to estimate its own pose and acquire a map model of its environment.

- Chicken-and-Egg problem:
  - Map is needed for localization.
  - Pose is needed for mapping.

Full SLAM: Problem Definition



$$\underset{X,L}{\operatorname{argmax}}\ p(X, L \mid Z, U) = \underset{X,L}{\operatorname{argmax}}\ p(X_0) \prod_{i=1}^{M} p(x_i \mid x_{i-1}, u_i) \prod_{k=1}^{K} p(z_k \mid x_{ik}, l_{jk})$$

Simultaneous Localization and Mapping (SLAM)

$$\underset{X,L}{\operatorname{argmax}}\ p\left(X,L\,|\,Z,U\right)=\underset{X,L}{\operatorname{argmax}}\ p(X_0)\prod_{i=1}^{M}p(x_i\,|\,x_{i-1},u_i)\prod_{k=1}^{K}p(z_k\,|\,x_{ik},l_{jk})$$

Negative log-likelihood →

$$=-\underset{X,L}{\operatorname{argmin}}\left\{\sum_{i=1}^{M}\ln p(x_i\,|\,x_{i-1},u_i)+\sum_{k=1}^{K}\ln p(z_k\,|\,x_{ik},l_{jk})\right\}$$

Likelihoods:

$$p(x_i\,|\,x_{i-1},u_i)\propto\exp\{-\left\|f(x_{i-1},u)-x_i\right\|_{\Lambda_i}^2\}$$

Process model

$$p(z_k\,|\,x_{ik},l_{jk})\propto\exp\{-\left\|h(x_{ik},l_{jk})-z_k\right\|_{\Sigma_k}^2\}$$

Measurement model

Simultaneous Localization and Mapping (SLAM)

$$\underset{X,L}{\operatorname{argmax}}\; p(X,L\,|\,Z,U) = \text{-}\underset{X,L}{\operatorname{argmin}}\left\{\sum_{i=1}^{M}\ln p(x_i\,|\,x_{i-1},u_i) + \sum_{k=1}^{K}\ln p(z_k\,|\,x_{ik},l_{jk})\right\}$$

Putting the likelihoods into the equation:

$$\underset{X,L}{\operatorname{argmax}}\; p(X,L\,|\,Z,U) = \underset{X,L}{\operatorname{argmin}}\left\{\sum_{i=1}^{M}\left\| f(x_{i-1},u_i) - x_i\right\|_{\Lambda_i}^2 + \sum_{k=1}^{K}\left\| h(x_{ik},l_{jk}) - z_k\right\|_{\Sigma_k}^2\right\}$$

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

Simultaneous Localization and Mapping (SLAM)

Normal Equations:

Weight made up of Λ_i , Σ_k

$$\left(J^T W J + \lambda I\right)\delta = -J^T W \Delta Z$$

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

Can be solved with sparse matrix factorization or iterative methods

Online SLAM: Problem Definition

- Estimate current pose x_t and full map L.

$$p(x_t, L \mid Z, U) = \underbrace{\int\int \dots \int p(X, L \mid Z, U)\, dx_1 dx_2 \dots dx_{t-1}}_{\text{Previous poses are marginalized out}}$$

- Inference with:
  1. Kalman Filtering (EKF SLAM)
  2. Particle Filtering (FastSLAM)

EKF SLAM

- Assumes: pose x_t and map L are random variables that follows Gaussian distribution.
- Hence,

$$p(x_t, L \mid Z, U) \sim \mathrm{N}(\mu, \Sigma)$$

mean Error covariance

EKF SLAM

Prediction:

$$\bar{\mu}_t = f(u_t, \mu_{t-1})$$ Process model

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

Correction:

$$y_t = z_t - h(\bar{\mu}_t)$$ Measurement residual (innovation)

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

$$\mu_t = \bar{\mu}_t + K_t y_t$$ Update mean

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

Measurement Jacobian $H_t = \dfrac{\partial h(\bar{\mu}_t)}{\partial x_t}$ Process Jacobian $F_t = \dfrac{\partial f(u_t, \mu_{t-1})}{\partial x_{t-1}}$

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 $p(x_t, L \mid Z, U)$.
- $p(x_t, L \mid Z, U)$ can be any distribution (need not be Gaussian).

FastSLAM

Each particle represents:

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

Robot state

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

Sample the robot state from the process model

$$p(L_{n,t}^m \mid x_t^m, z_t)$$

N Kalman filter measurement updates

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

Weight update

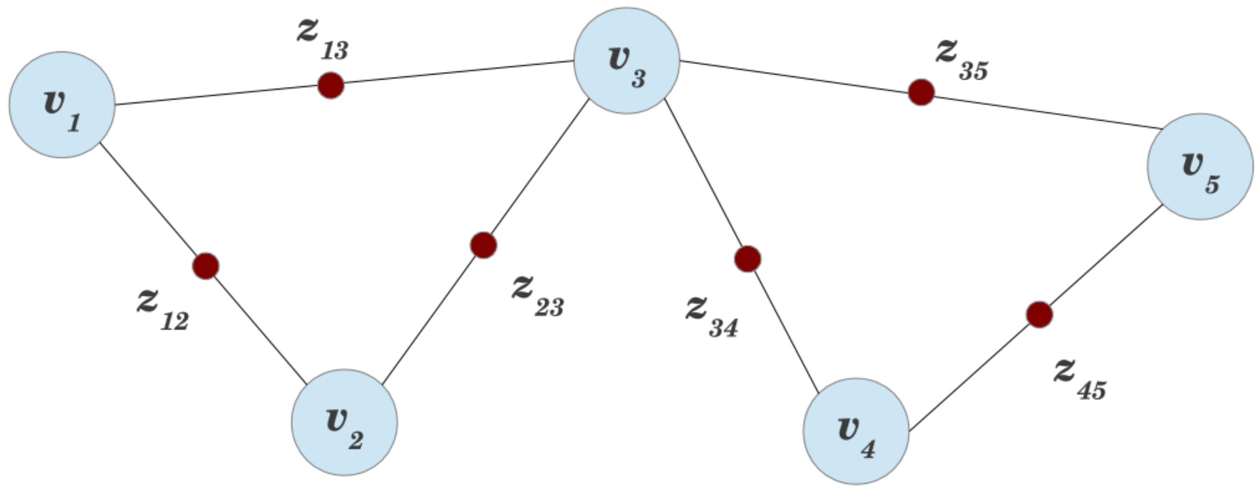
Resampling

FastSLAM

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

Pose-Graph SLAM

- 3D structures are removed (fixed).
- Constraints are relative pose estimates from 3D structures.
- Minimizes loop-closure errors.



$$\underset{X}{\operatorname{argmin}} \sum_{ij} \left\| z_{ij} - h(v_i, v_j) \right\|_{\Sigma_{ij}}^2$$

Conclusion

- Bundle Adjustment
- Simultaneous Localization and Mapping
  - Full SLAM
  - Online SLAM
    - EKF SLAM
    - FastSLAM
- Pose-Graph SLAM