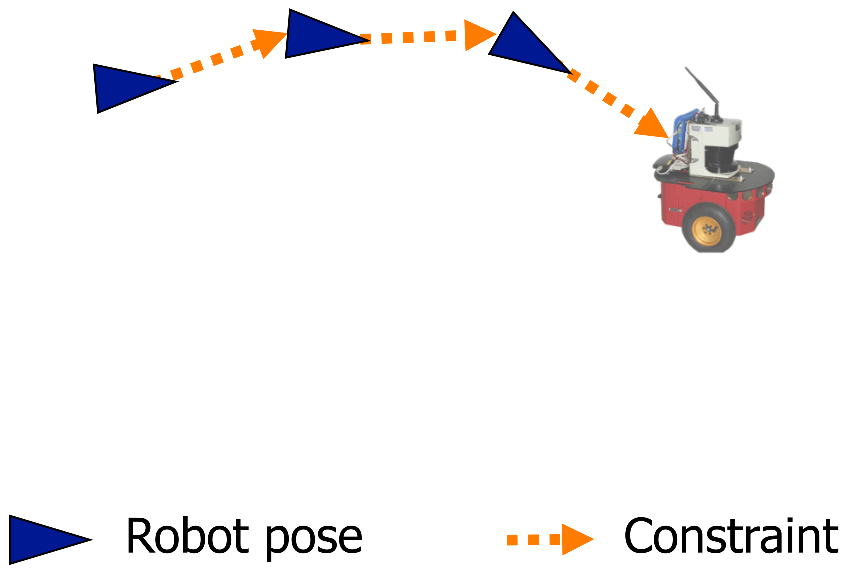
g2o: A General Framework for Graph Optimization

Rainer Kümmerle Giorgio Grisetti
Hauke Strasdat Kurt Konolige
Wolfram Burgard

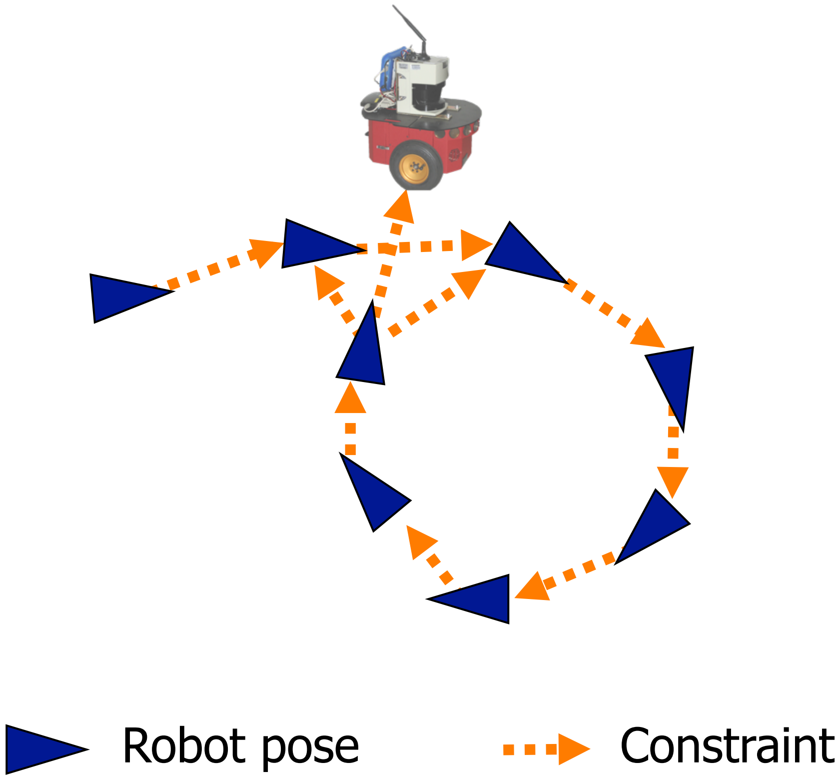
Graph-Based SLAM

- Constraints connect the poses of the robot while it is moving
- Constraints are inherently uncertain



Graph-Based SLAM

- Observing previously seen areas generates constraints between non-successive poses

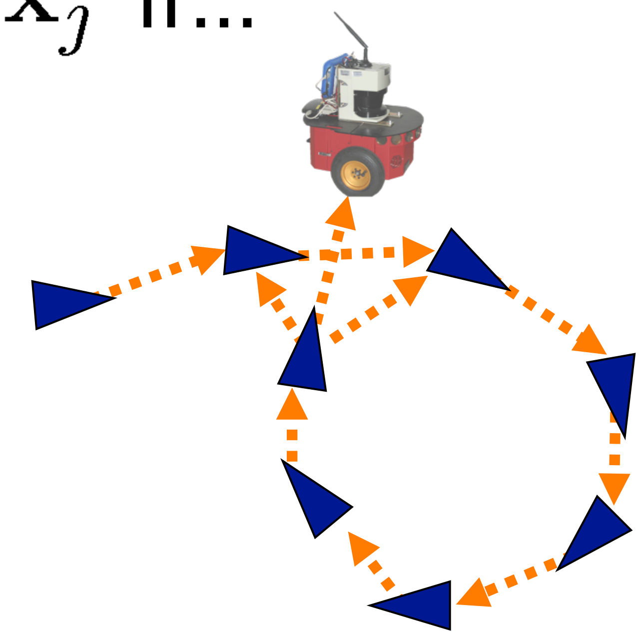


Idea of Graph-Based SLAM

- Use a **graph** to represent the problem
- Every **node** in the graph corresponds to a pose of the robot during mapping
- Every **edge** between two nodes corresponds to a spatial constraint between them
- **Graph-Based SLAM:** Build the graph and find a node configuration that minimize the error introduced by the constraints

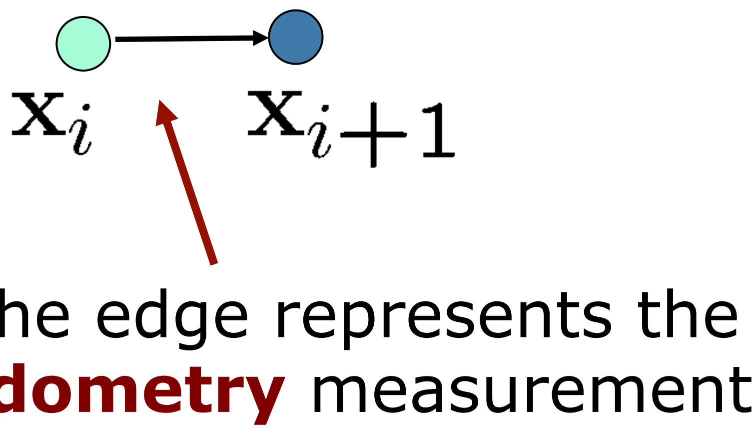
The Graph

- It consists of n nodes $\mathbf{x} = \mathbf{x}_{1:n}$
- Each $\mathbf{x}_i$ is a 2D or 3D transformation (the pose of the robot at time t_i)
- A constraint/edge exists between the nodes $\mathbf{x}_i$ and $\mathbf{x}_j$ if…



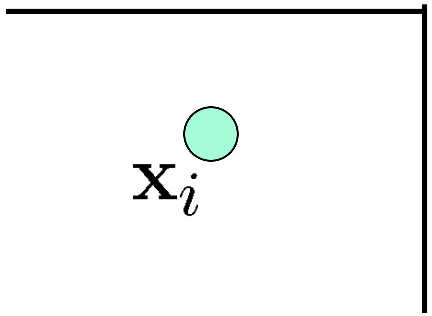
Create an Edge If... (1)

- ...the robot moves from $\mathbf{x}_i$ to $\mathbf{x}_{i+1}$
- Edge corresponds to odometry

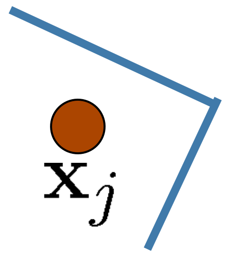


Create an Edge If... (2)

- ...the robot observes the same part of the environment from $\mathbf{x}_i$ and from $\mathbf{x}_j$



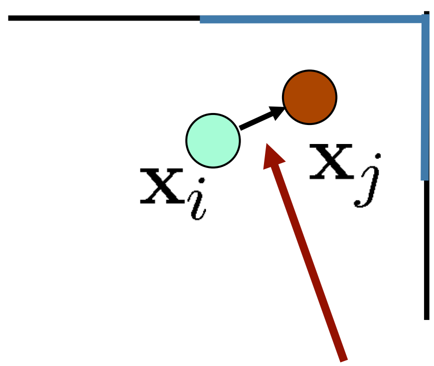
Measurement from $\mathbf{x}_i$



Measurement from $\mathbf{x}_j$

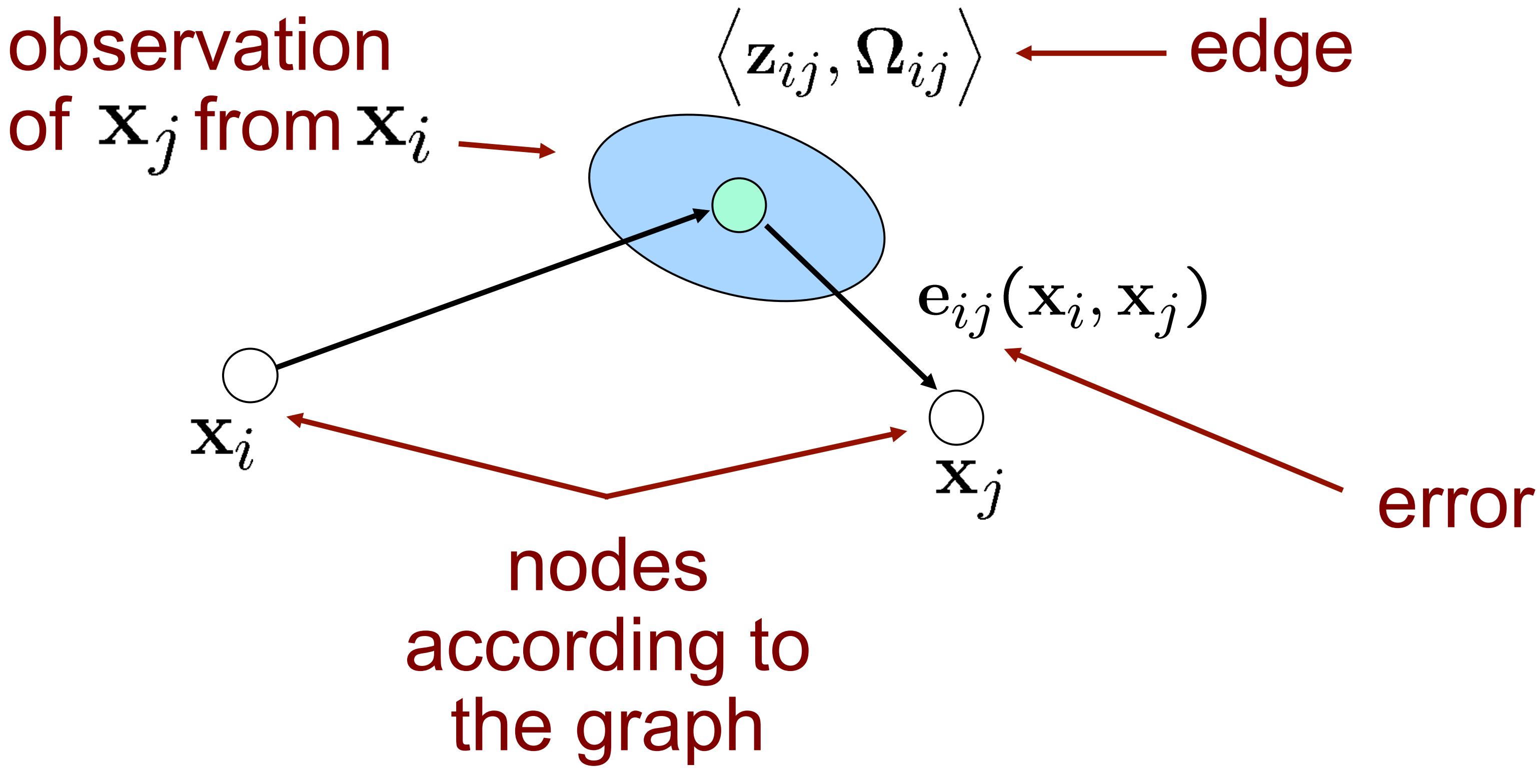
Create an Edge If… (2)

- …the robot observes the same part of the environment from $\mathbf{x}_i$ and from $\mathbf{x}_j$
- Construct a **virtual measurement** about the position of $\mathbf{x}_j$ seen from $\mathbf{x}_i$



Edge represents the position of $\mathbf{x}_j$ seen from $\mathbf{x}_i$ based on the **observation**

Pose Graph



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

The Error Function

- Error function for a single constraint

$$\mathbf{e}_{ij}(\mathbf{x}_i, \mathbf{x}_j) = \text{t2v}(\underbrace{\mathbf{Z}_{ij}^{-1}}_{\text{measurement}}(\underbrace{\mathbf{X}_i^{-1}\mathbf{X}_j}_{\mathbf{x}_j \text{ referenced w.r.t. } \mathbf{x}_i}))$$

- Error takes a value of zero if

$$\mathbf{Z}_{ij} = (\mathbf{X}_i^{-1}\mathbf{X}_j)$$

Gauss-Newton: The Overall Error Minimization Procedure

- 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

Linearizing the Error Function

- We can approximate the error functions around an initial guess $\mathbf{x}$ via Taylor expansion

$$\mathbf{e}_{ij}(\mathbf{x} + \mathbf{\Delta x}) \simeq \mathbf{e}_{ij}(\mathbf{x}) + \mathbf{J}_{ij}\mathbf{\Delta x}$$

with $\mathbf{J}_{ij} = \frac{\partial \mathbf{e}_{ij}(\mathbf{x})}{\partial \mathbf{x}}$

Jacobians and Sparsity

- Error $\mathbf{e}_{ij}(\mathbf{x})$ depends only on the two parameter blocks $\mathbf{x}_i$ and $\mathbf{x}_j$

$$\mathbf{e}_{ij}(\mathbf{x}) = \mathbf{e}_{ij}(\mathbf{x}_i, \mathbf{x}_j)$$

- The Jacobian will be zero everywhere except in the columns of $\mathbf{x}_i$ and $\mathbf{x}_j$

$$\mathbf{J}_{ij} = \left(\mathbf{0} \cdots \mathbf{0} \quad \underbrace{\frac{\partial \mathbf{e}(\mathbf{x}_i)}{\partial \mathbf{x}_i}}_{\mathbf{A}_{ij}} \quad \mathbf{0} \cdots \mathbf{0} \quad \underbrace{\frac{\partial \mathbf{e}(\mathbf{x}_j)}{\partial \mathbf{x}_j}}_{\mathbf{B}_{ij}} \quad \mathbf{0} \cdots \mathbf{0} \right)$$

Consequences of the Sparsity

- We need to compute the coefficient vector $\mathbf{b}$ and matrix $\mathbf{H}$:

$$\begin{aligned} \mathbf{b}^T &= \sum_{ij} \mathbf{b}_{ij}^T &= \sum_{ij} \mathbf{e}_{ij}^T \boldsymbol{\Omega}_{ij} \mathbf{J}_{ij} \\ \mathbf{H} &= \sum_{ij} \mathbf{H}_{ij} &= \sum_{ij} \mathbf{J}_{ij}^T \boldsymbol{\Omega}_{ij} \mathbf{J}_{ij} \end{aligned}$$

- The sparse structure of $\mathbf{J}_{ij}$ will result in a sparse structure of $\mathbf{H}$
- This structure reflects the adjacency matrix of the graph

Illustration of the Structure

$$\mathbf{b}_{ij} = \mathbf{J}_{ij}^T \boldsymbol{\Omega}_{ij} \mathbf{e}_{ij}$$

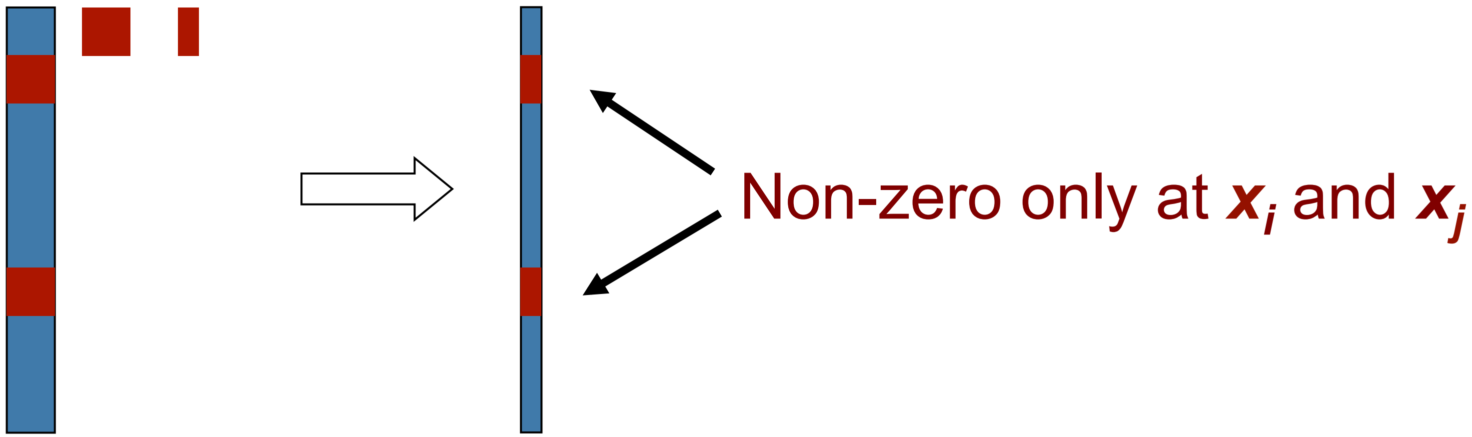
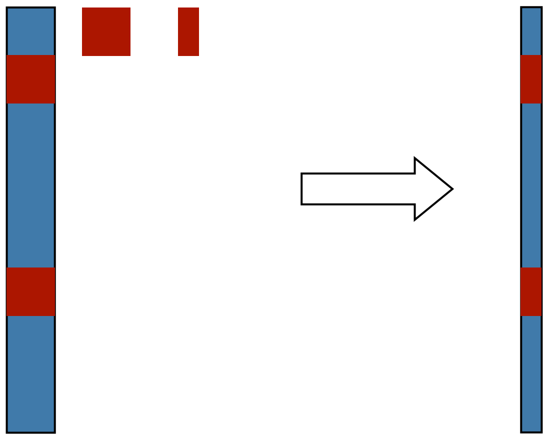


Illustration of the Structure

$$\mathbf{b}_{ij} = \mathbf{J}_{ij}^T \mathbf{\Omega}_{ij} \mathbf{e}_{ij}$$



Non-zero only at $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$

$$\mathbf{H}_{ij} = \mathbf{J}_{ij}^T \mathbf{\Omega}_{ij} \mathbf{J}_{ij}$$

Non-zero on the main diagonal at $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$

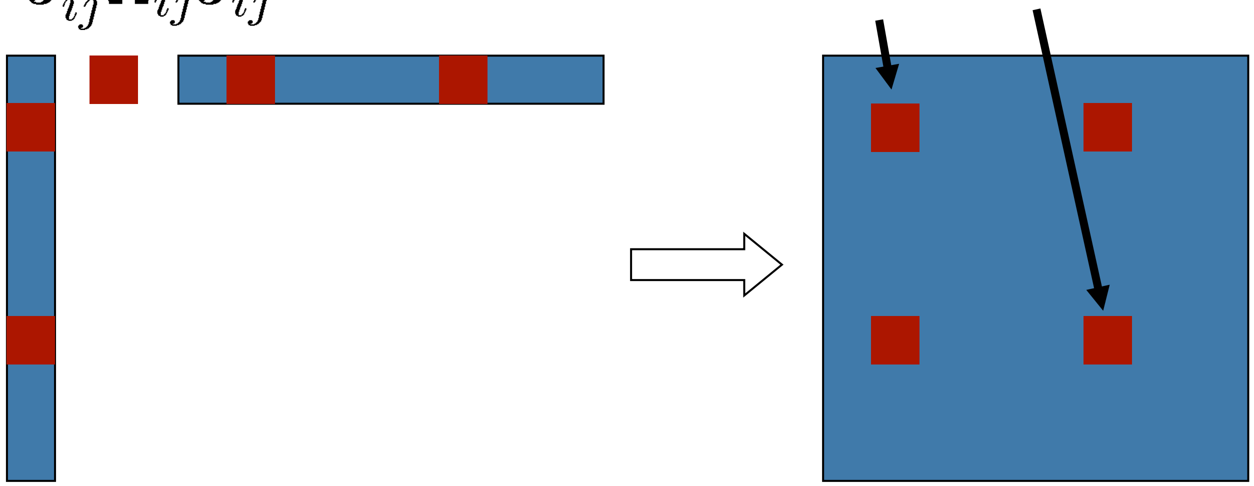
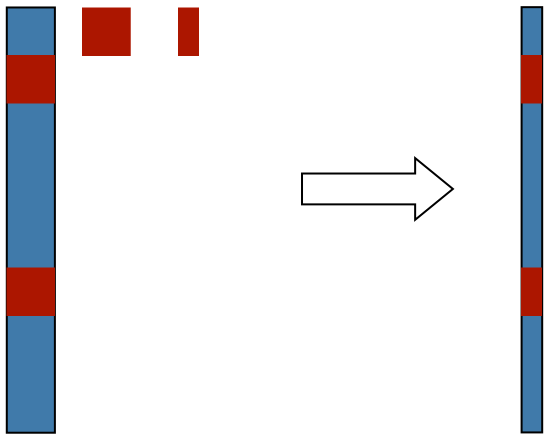


Illustration of the Structure

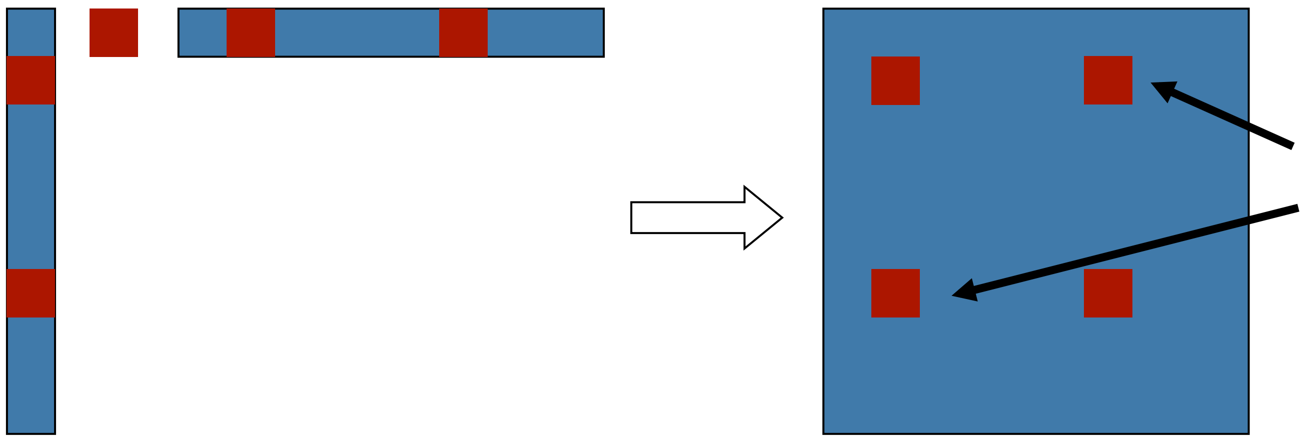
$$\mathbf{b}_{ij} = \mathbf{J}_{ij}^T \mathbf{\Omega}_{ij} \mathbf{e}_{ij}$$



Non-zero only at $\boldsymbol{x_i}$ and $\boldsymbol{x_j}$

$$\mathbf{H}_{ij} = \mathbf{J}_{ij}^T \mathbf{\Omega}_{ij} \mathbf{J}_{ij}$$

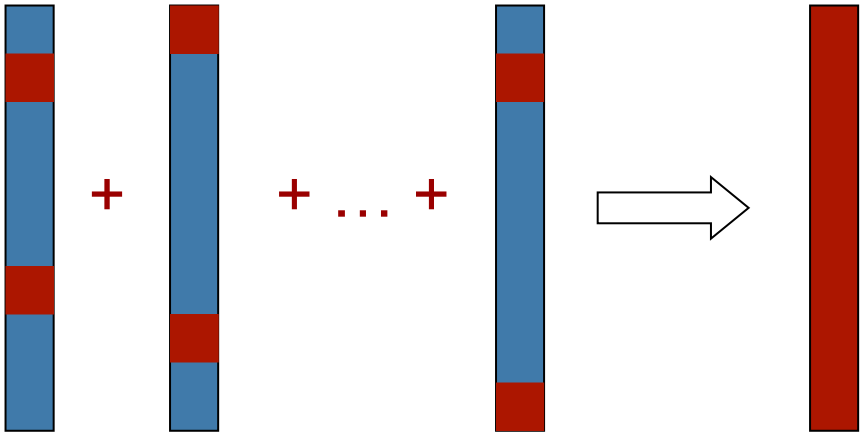
Non-zero on the main diagonal at $\boldsymbol{x_i}$ and $\boldsymbol{x_j}$



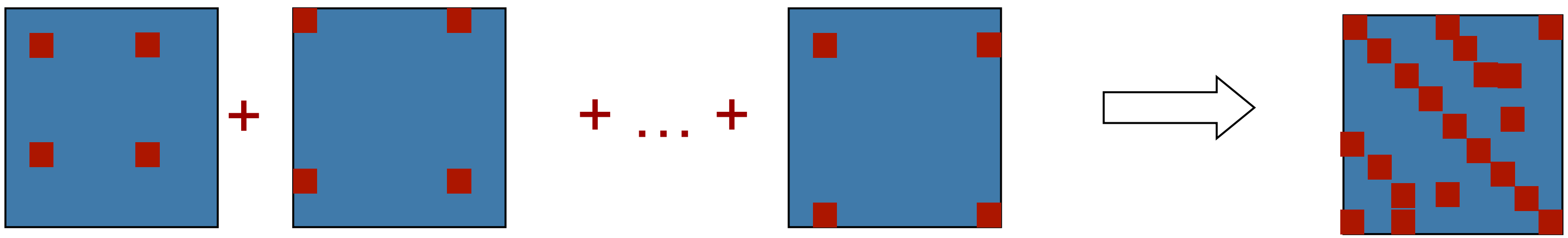
... and at the blocks *ij*,*ji*

Illustration of the Structure

$$\mathrm{b} = \sum_{ij} \mathrm{b}_{ij}$$



$$\mathrm{H} = \sum_{ij} \mathrm{H}_{ij}$$



The Linear System

- Vector of the states increments:

$$\mathbf{\Delta x}^T = \begin{pmatrix} \mathbf{\Delta x}_1^T & \mathbf{\Delta x}_2^T & \cdots & \mathbf{\Delta x}_n^T \end{pmatrix}$$

- Coefficient vector:

$$\mathbf{b}^T = \begin{pmatrix} \bar{\mathbf{b}}_1^T & \bar{\mathbf{b}}_2^T & \cdots & \bar{\mathbf{b}}_n^T \end{pmatrix}$$

- System matrix:

$$\mathbf{H} = \begin{pmatrix} \bar{\mathbf{H}}^{11} & \bar{\mathbf{H}}^{12} & \cdots & \bar{\mathbf{H}}^{1n} \\ \bar{\mathbf{H}}^{21} & \bar{\mathbf{H}}^{22} & \cdots & \bar{\mathbf{H}}^{2n} \\ \vdots & \ddots & & \vdots \\ \bar{\mathbf{H}}^{n1} & \bar{\mathbf{H}}^{n2} & \cdots & \bar{\mathbf{H}}^{nn} \end{pmatrix}$$

Building the Linear System

For each constraint:

- Compute error $\mathbf{e}_{ij} = \text{t2v}(\mathbf{Z}_{ij}^{-1}(\mathbf{X}_i^{-1}\mathbf{X}_j))$
- Compute the blocks of the Jacobian:

$$\mathbf{A}_{ij} = \frac{\partial \mathbf{e}(\mathbf{x}_i, \mathbf{x}_j)}{\partial \mathbf{x}_i} \qquad \mathbf{B}_{ij} = \frac{\partial \mathbf{e}(\mathbf{x}_i, \mathbf{x}_j)}{\partial \mathbf{x}_j}$$

- Update the coefficient vector:

$$\bar{\mathbf{b}}_i^T += \mathbf{e}_{ij}^T \boldsymbol{\Omega}_{ij} \mathbf{A}_{ij} \qquad \bar{\mathbf{b}}_j^T += \mathbf{e}_{ij}^T \boldsymbol{\Omega}_{ij} \mathbf{B}_{ij}$$

- Update the system matrix:

$$\bar{\mathbf{H}}^{ii} += \mathbf{A}_{ij}^T \boldsymbol{\Omega}_{ij} \mathbf{A}_{ij} \qquad \bar{\mathbf{H}}^{ij} += \mathbf{A}_{ij}^T \boldsymbol{\Omega}_{ij} \mathbf{B}_{ij}$$

$$\bar{\mathbf{H}}^{ji} += \mathbf{B}_{ij}^T \boldsymbol{\Omega}_{ij} \mathbf{A}_{ij} \qquad \bar{\mathbf{H}}^{jj} += \mathbf{B}_{ij}^T \boldsymbol{\Omega}_{ij} \mathbf{B}_{ij}$$

Algorithm

optimize($\mathbf{x}$)**:**

while (!*converged*)

$(\mathbf{H}, \mathbf{b}) = \text{buildLinearSystem}(\mathbf{x})$

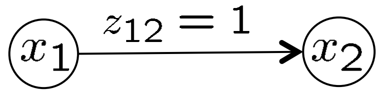
$\mathbf{\Delta x} = \text{solveSparse}(\mathbf{H}\mathbf{\Delta x} = -\mathbf{b})$

$\mathbf{x} = \mathbf{x} + \mathbf{\Delta x}$

end

return $\mathbf{x}$

Trivial 1D Example



- Two nodes and one observation

$$
\begin{aligned}
\mathbf{x} &= (x_1\, x_2)^T = (0\, 0) \\
\mathbf{z}_{12} &= 1 \\
\mathbf{\Omega} &= 2 \\
\mathbf{e}_{12} &= = z_{12} - (x_2 - x_1) = 1 - (0 - 0) = 1 \\
\mathbf{J}_{12} &= (1 \;\; -1) \\
\mathbf{b}_{12}^T &= \mathbf{e}_{12}^T \mathbf{\Omega}_{12} \mathbf{J}_{12} = (2 \;\; -2) \\
\mathbf{H}_{12} &= \mathbf{J}_{12}^T \mathbf{\Omega} \mathbf{J}_{12} = \begin{pmatrix} 2 & -2 \\ -2 & 2 \end{pmatrix} \\
\Delta\mathbf{x} &= -\mathbf{H}_{12}^{-1} b_{12}
\end{aligned}
$$

BUT $\det(\mathbf{H}) = 0$ **???**

What Went Wrong?

- The constraint specifies a **relative constraint** between both nodes
- Any poses for the nodes would be fine as long a their relative coordinates fit
- **One node needs to be "fixed"**

$$\mathbf{H} = \begin{pmatrix} 2 & -2 \\ -2 & 2 \end{pmatrix} + \boxed{\begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}}$$

$$\Delta \mathbf{x} = -\mathbf{H}^{-1} b_{12}$$

$$\Delta \mathbf{x} = (0\;1)^T$$

constraint that sets ***dx_1=0***

2D Pose-Graph of the Intel Research Lab

https://www.youtube.com/watch?v=8BUhMhk3JB0