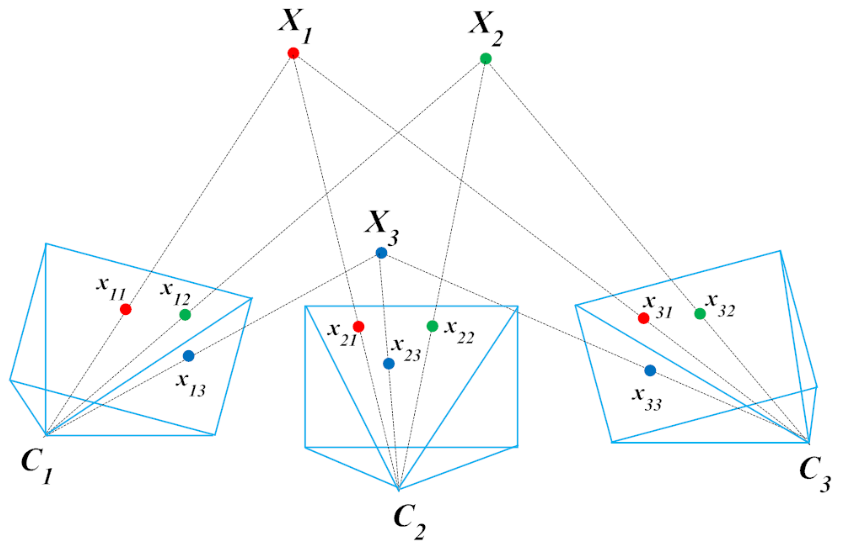
Bundle Adjustment

刘浩敏

Introduction of BA

- Bundle Adjustment (BA) is to jointly optimize all cameras and points, by minimizing reprojection errors

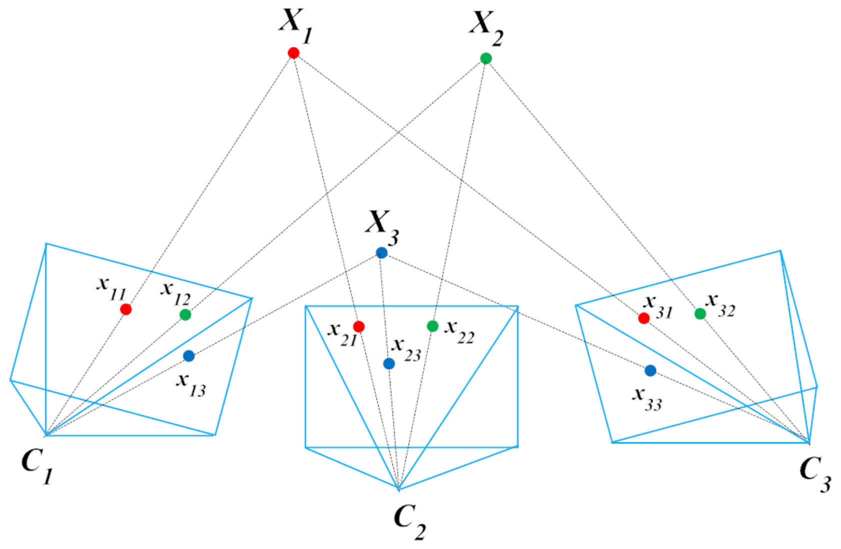
$$\underset{C_1,\ldots C_{N_c},X_1,\ldots,X_{N_p}}{\operatorname{argmin}} \sum \left\| \pi(C_i, X_j) - x_{ij} \right\|^2$$



Introduction of BA

- Bundle Adjustment (BA) is to jointly optimize all cameras and points, by minimizing reprojection errors

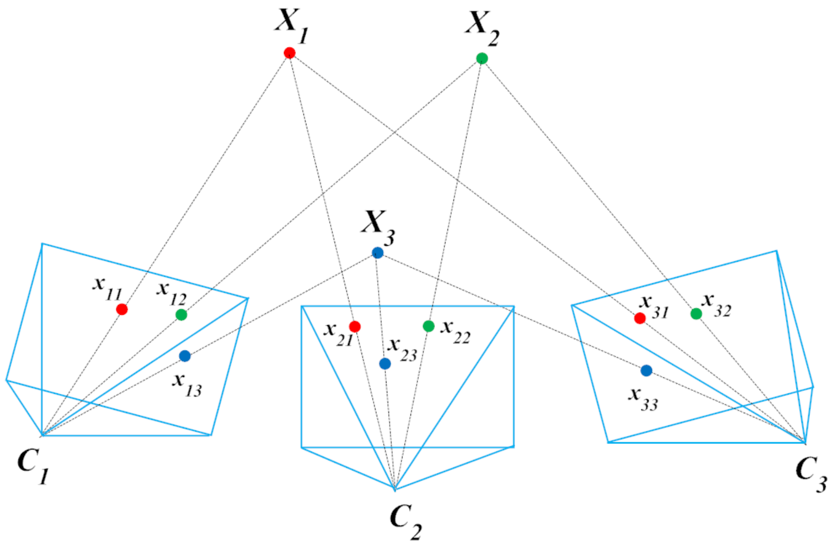
$$\underset{C_1,\dots C_{N_c}, X_1,\dots,X_{N_p}}{\operatorname{argmin}} \sum \left\| \pi(C_i, X_j) - x_{ij} \right\|^2$$



Introduction of BA

- Bundle Adjustment (BA) is to jointly optimize all cameras and points, by minimizing the reprojection errors

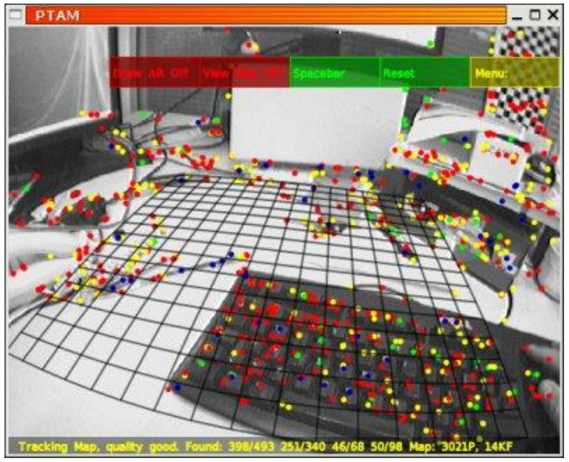
$$\underset{C_1,\ldots C_{N_c},X_1,\ldots,X_{N_p}}{\operatorname{argmin}} \sum \left\| \pi(C_i, X_j) - x_{ij} \right\|^2$$



Introduction of BA

- BA is a golden step for almost all SfM and SLAM systems

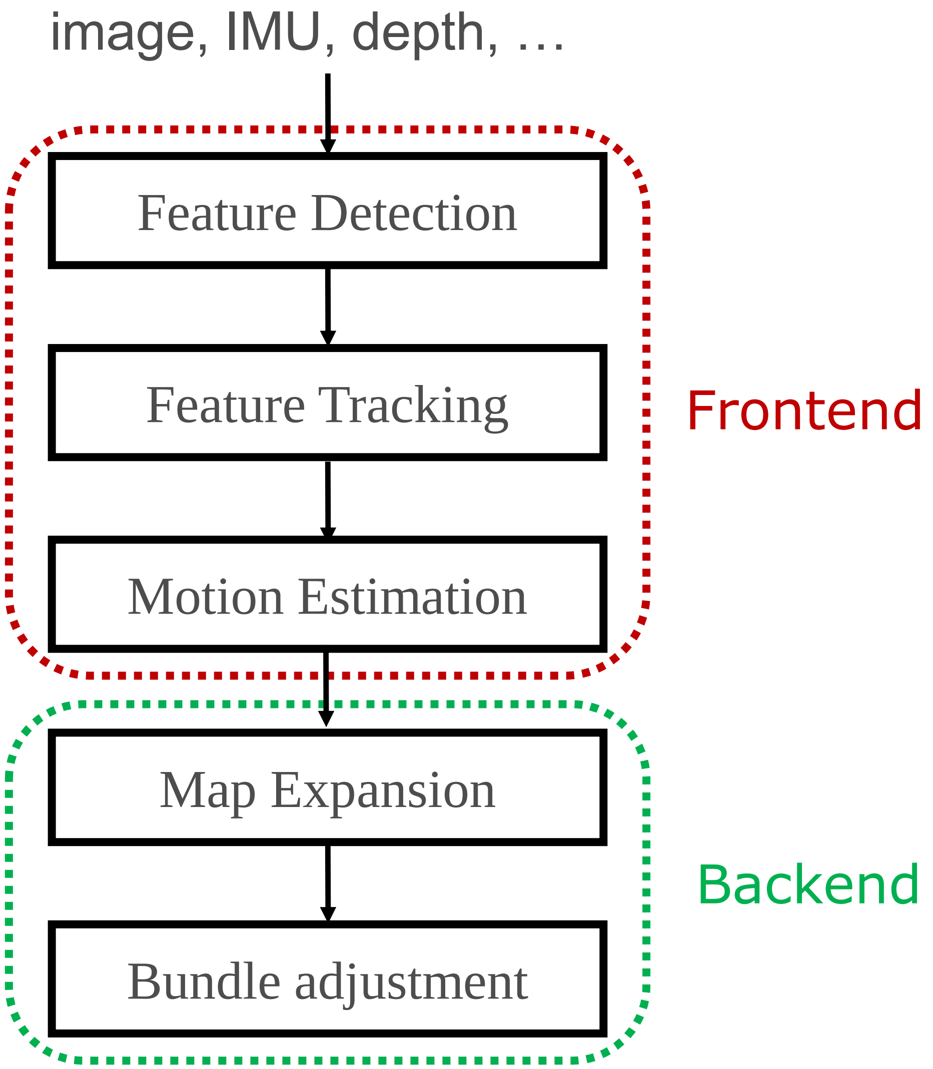
Bundler （SfM)



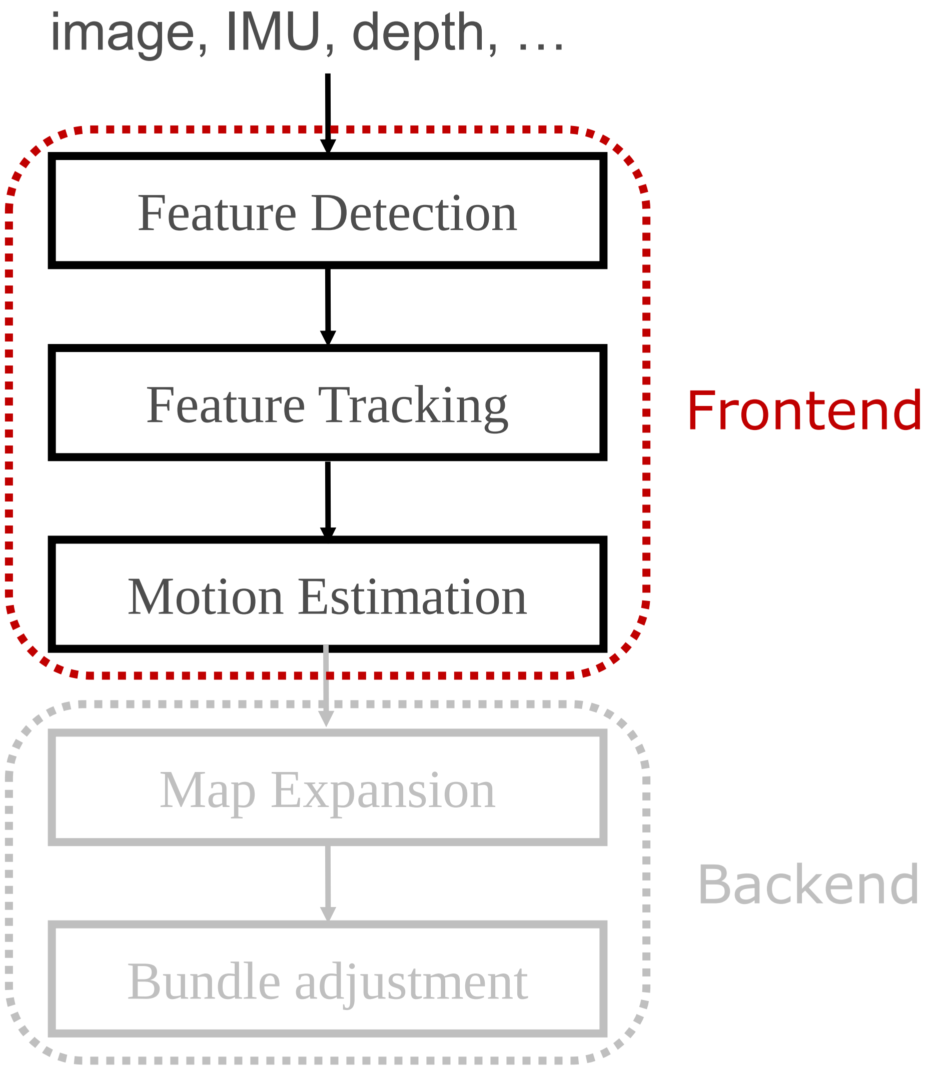
```
Tracking Map, quality good. Found: 398/493 251/340 46/68 50/98 Map: 3021P, 14KF
```

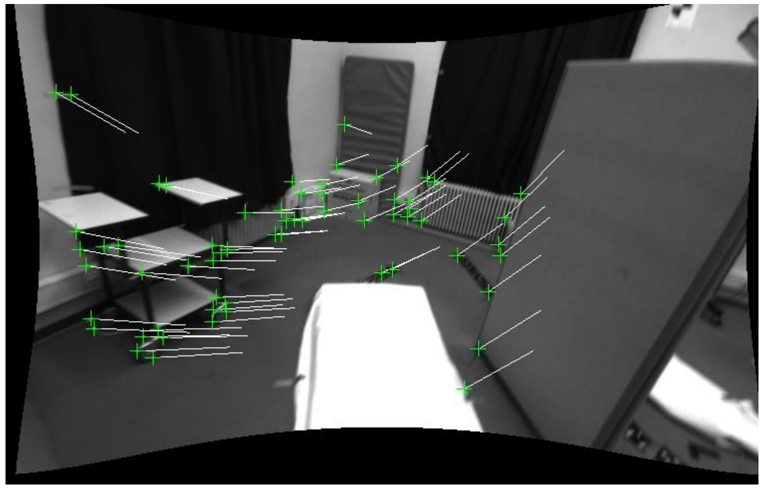
PTAM （SLAM)

Flowchart of SLAM

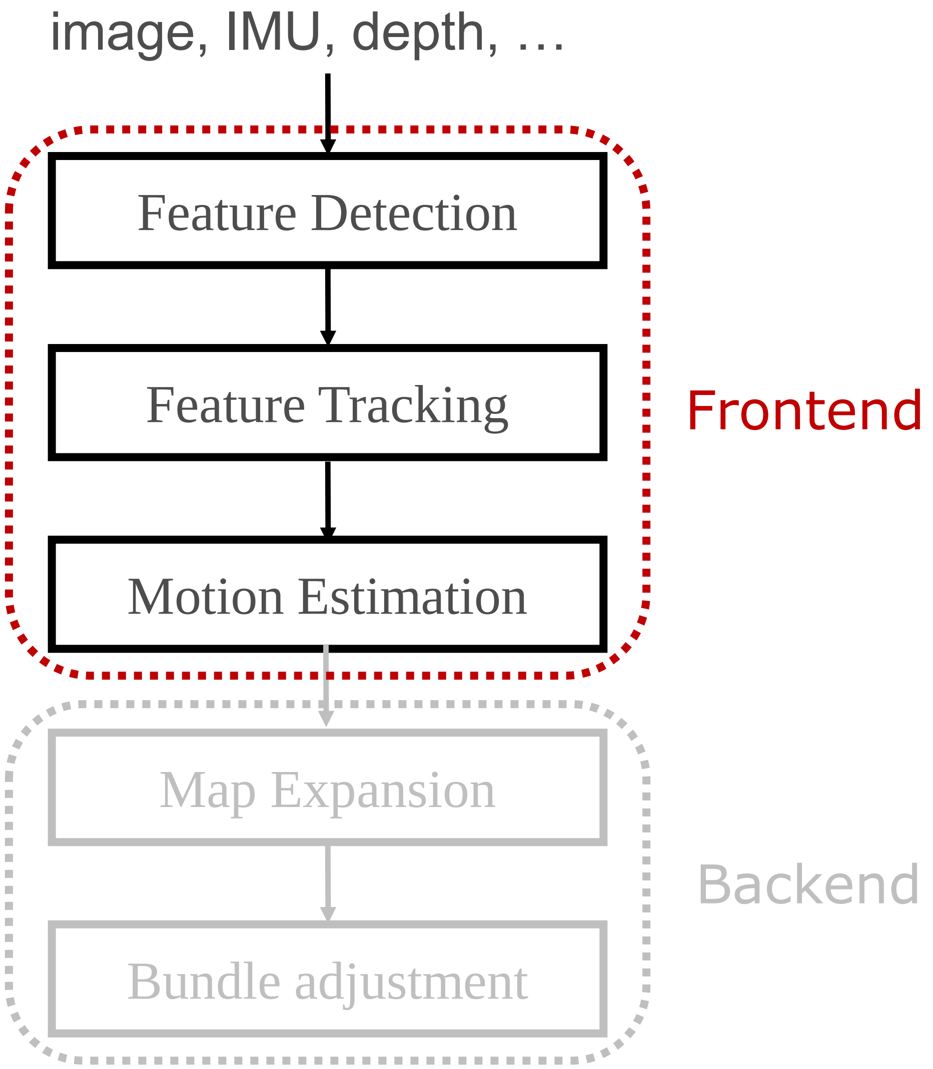


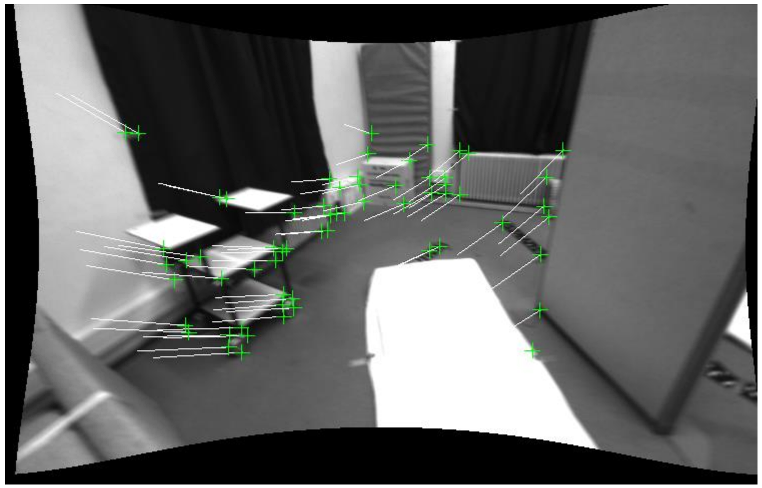
Flowchart of SLAM



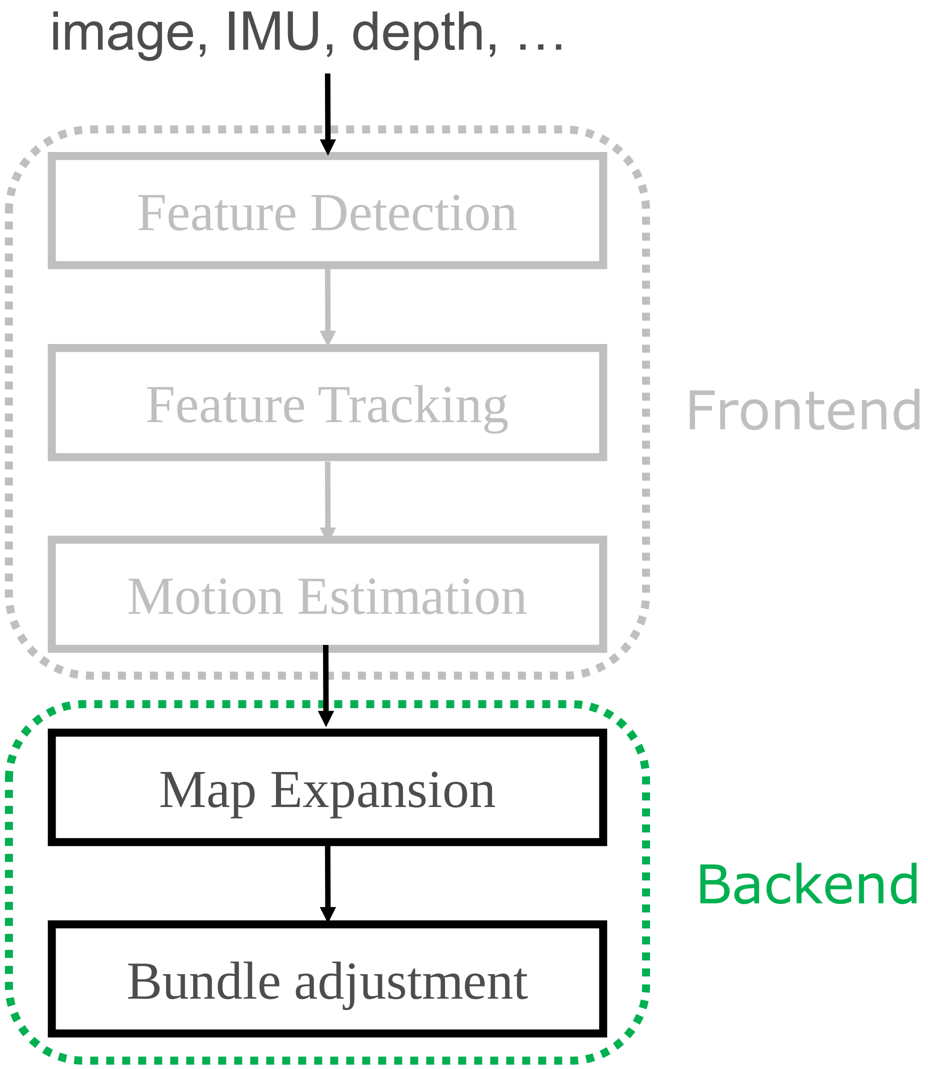


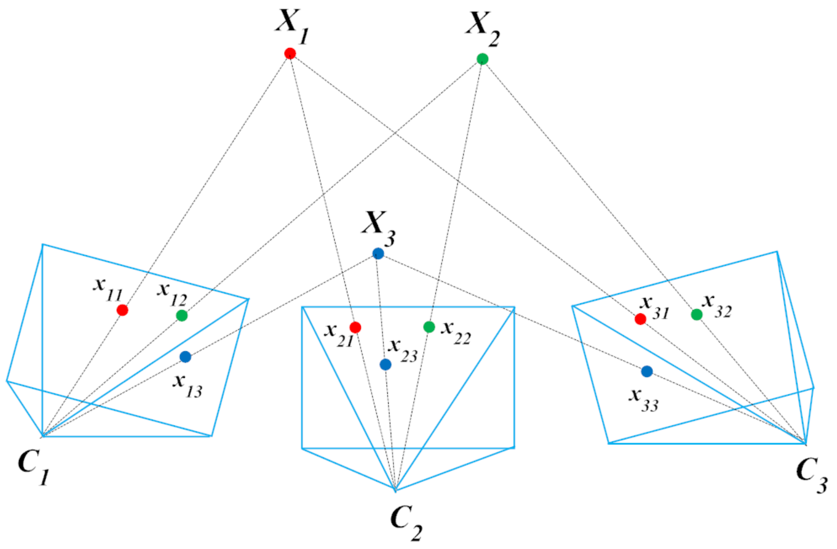
Flowchart of SLAM





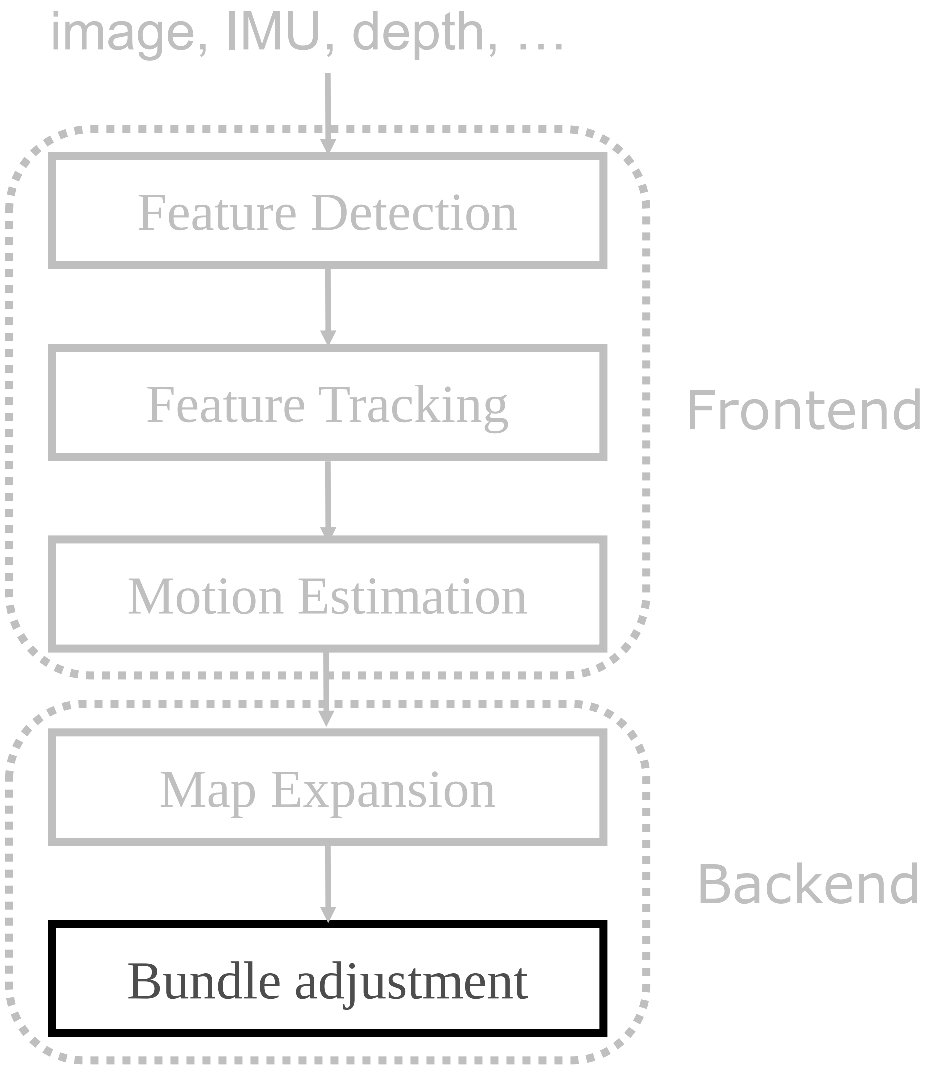
Flowchart of SLAM

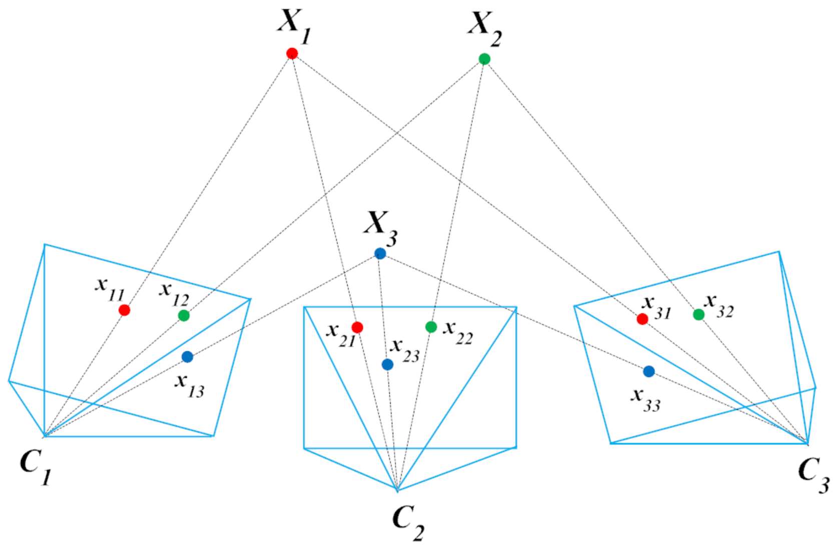




$$\underset{C_1,\dots C_{N_c},X_1,\dots,X_{N_p}}{\operatorname{argmin}} \sum \left\|\pi(X_i, C_j) - x_{ij}\right\|^2$$

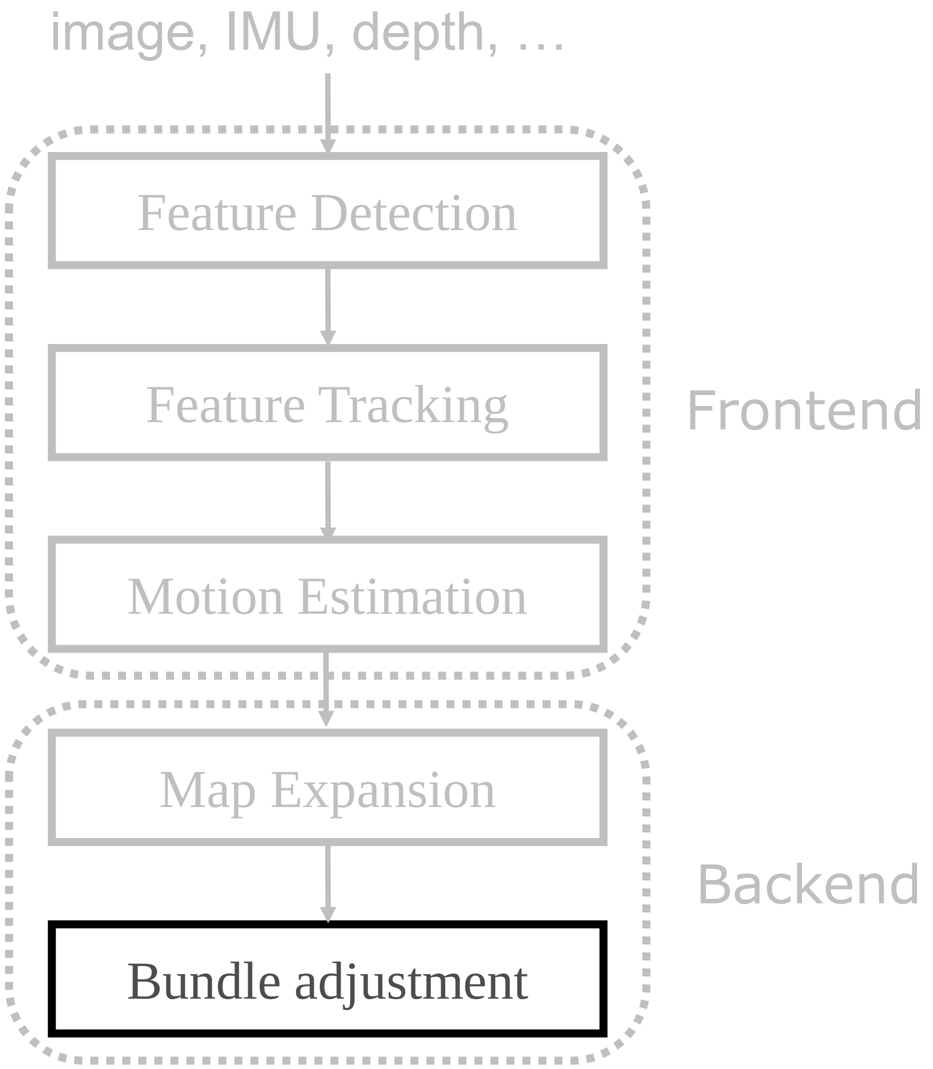
Flowchart of SLAM

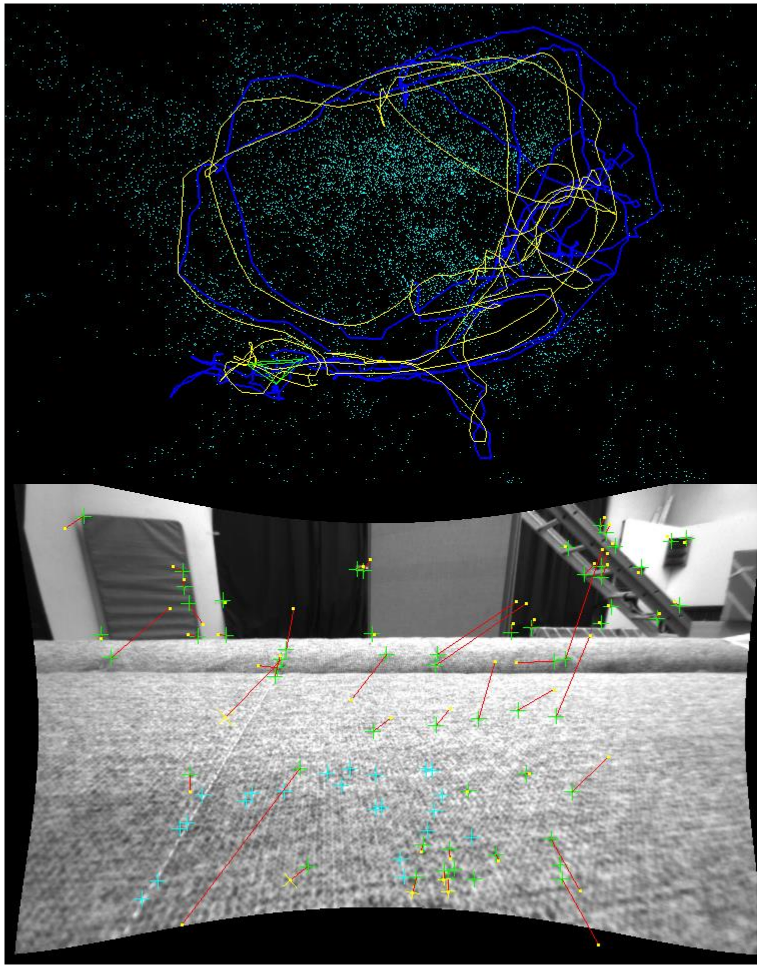




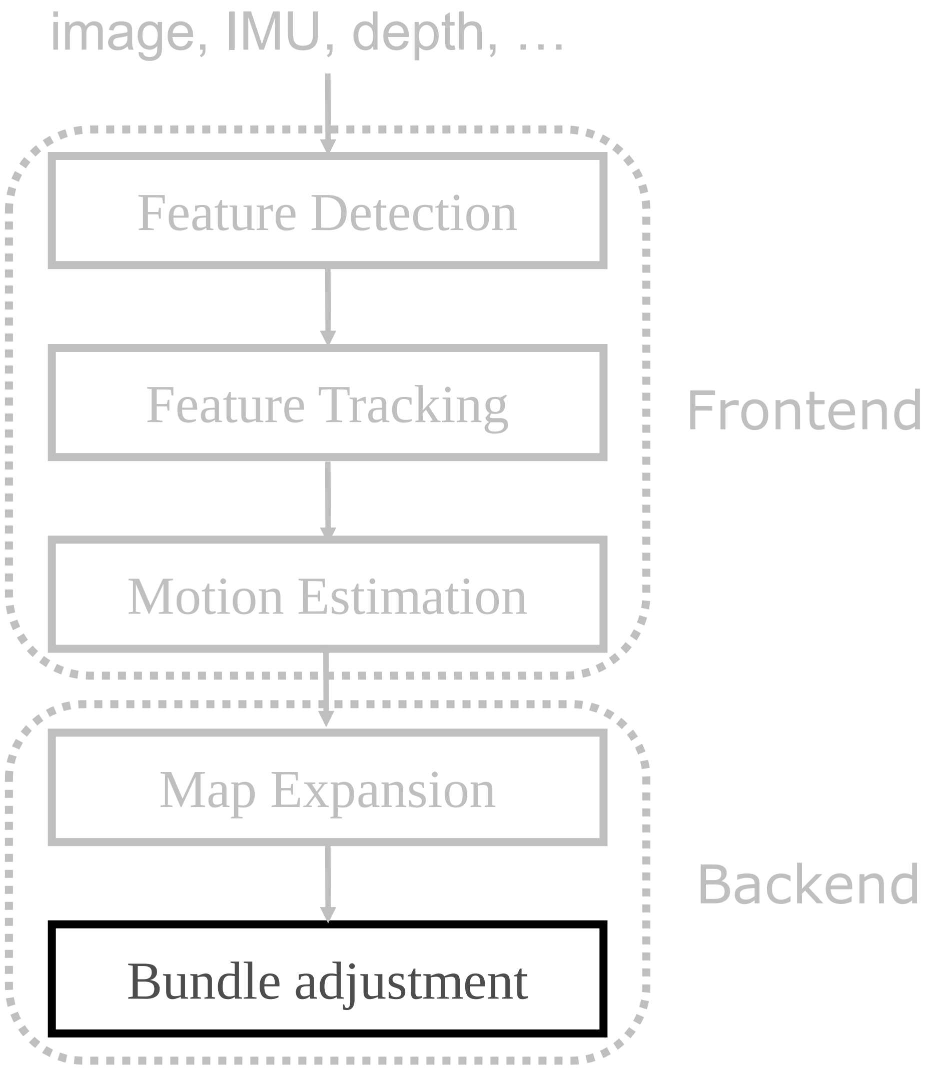
$$\underset{C_1,\dots C_{N_c},X_1,\dots,X_{N_p}}{\operatorname{argmin}} \sum \left\|\pi(X_i, C_j) - x_{ij}\right\|^2$$

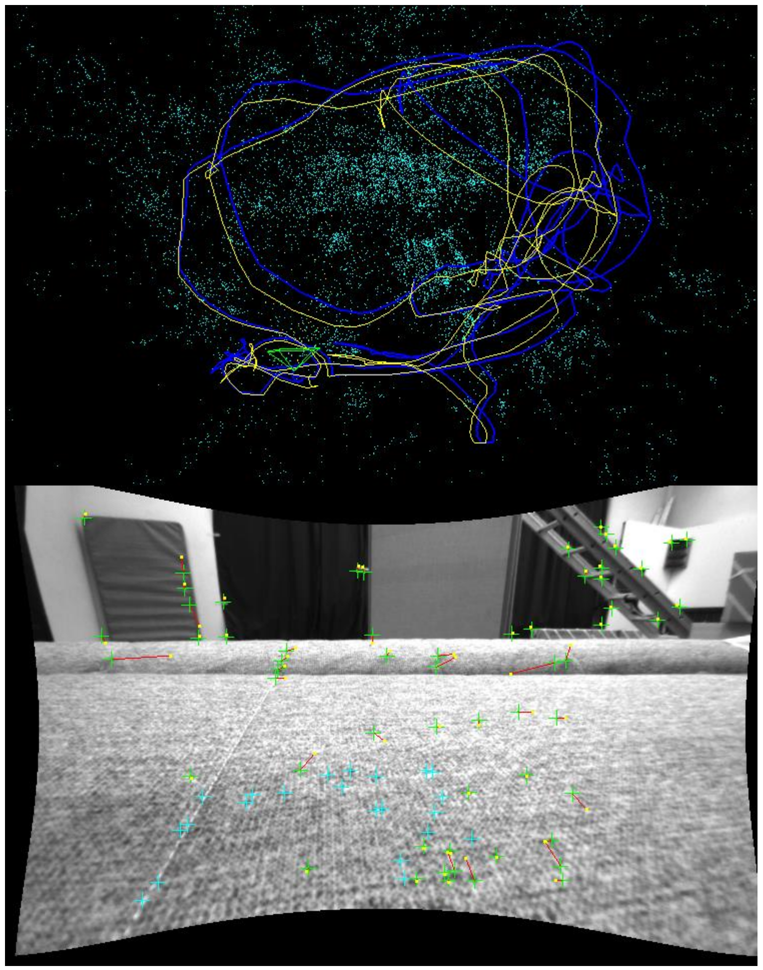
Flowchart of SLAM



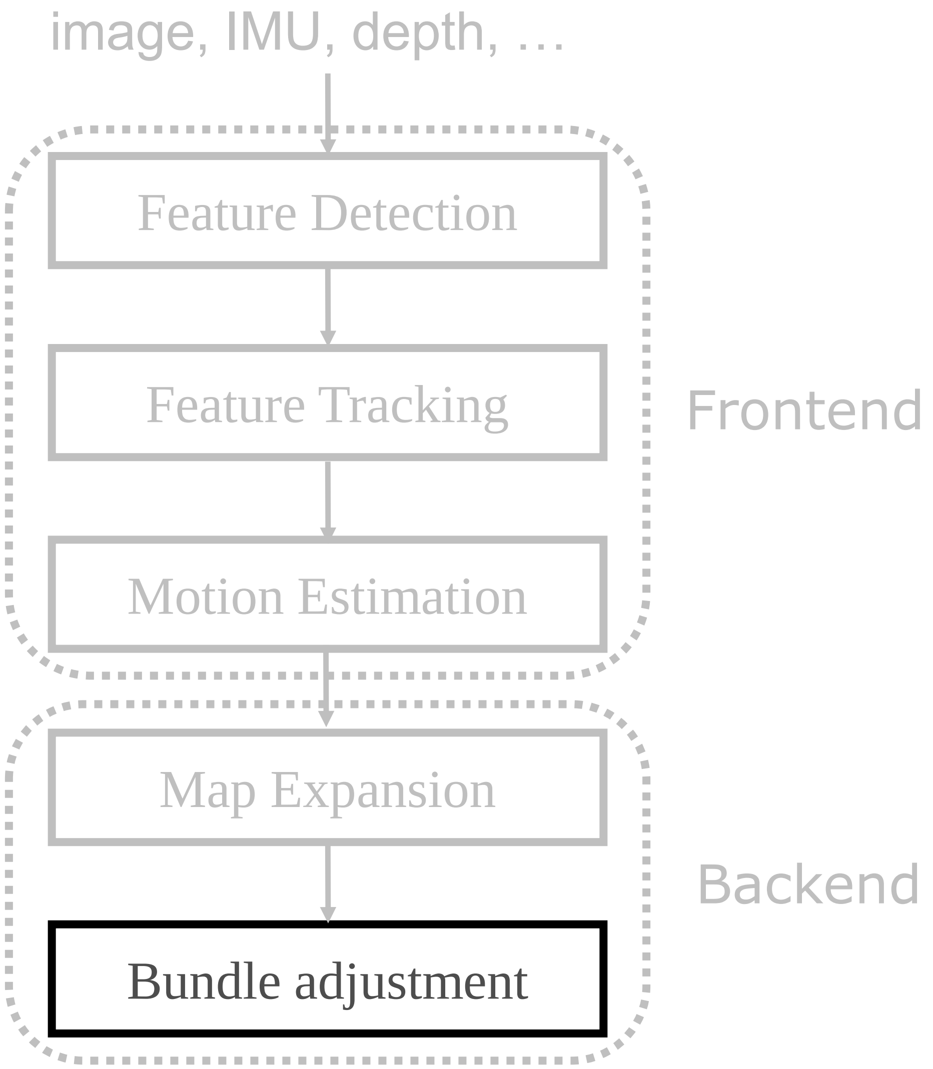


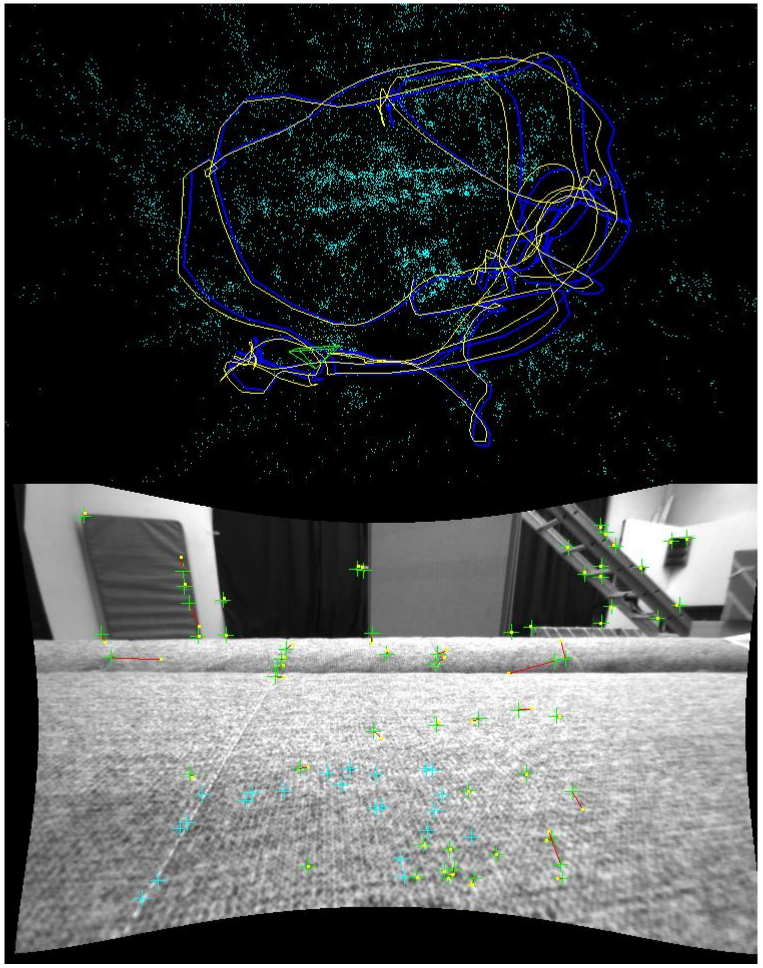
Flowchart of SLAM





Flowchart of SLAM





Outline

- Theories in BA
- BA for large scale SfM
- BA for realtime SLAM

Outline

- Theories in BA
- BA for large scale SfM
- BA for realtime SLAM

Nonlinear Least Sqaures

$$x^* = \underset{x}{\operatorname{argmin}}\, E(x)$$

Nonlinear Least Sqaures

$$x^* = \underset{x}{\operatorname{argmin}} E(x)$$

Linea case

$$E(x) = \|Ax + b\|^2$$

Nonlinear Least Sqaures

$$x^* = \operatorname*{argmin}_x E(x)$$

Linea case

$$E(x) = \|Ax + b\|^2$$

$$= x^T(A^T A)x + 2(A^T b)x + b^T b$$

Nonlinear Least Sqaures

$$x^* = \underset{x}{\operatorname{argmin}} E(x)$$

Linea case

$$E(x) = \|Ax + b\|^2$$

$$= x^T(A^TA)x + 2(A^Tb)x + b^Tb$$

$$\frac{\partial E(x)}{\partial x} = 2(A^TAx + A^Tb) = 0$$

Nonlinear Least Sqaures

$$x^* = \operatorname*{argmin}_x E(x)$$

Linea case

$$E(x) = \|Ax + b\|^2$$

$$= x^T(A^T A)x + 2(A^T b)x + b^T b$$

$$\frac{\partial E(x)}{\partial x} = 2(A^T Ax + A^T b) = 0$$

$$A^T A\, x = -A^T b$$

Nonlinear Least Sqaures

$$x^* = \underset{x}{\operatorname{argmin}}\, E(x)$$

Linea case

$$E(x) = \|Ax + b\|^2$$

$$= x^T(A^TA)x + 2(A^Tb)x + b^Tb$$

$$\frac{\partial E(x)}{\partial x} = 2(A^TAx + A^Tb) = 0$$

$$A^TA\,x = -A^Tb$$

Nonlinear case

$$E(x) = \|\varepsilon(x)\|^2$$

Nonlinear Least Sqaures

$$x^* = \underset{x}{\operatorname{argmin}}\, E(x)$$

Linea case

$$E(x) = \|Ax + b\|^2$$

$$= x^T(A^TA)x + 2(A^Tb)x + b^Tb$$

$$\frac{\partial E(x)}{\partial x} = 2(A^TAx + A^Tb) = 0$$

$$A^TA\,x = -A^Tb$$

Nonlinear case

$$E(x) = \|\varepsilon(x)\|^2$$

$$x^* = \hat{x} + \delta_x$$

Nonlinear Least Sqaures

$$x^* = \underset{x}{\operatorname{argmin}} E(x)$$

Linea case

$$E(x) = \|Ax + b\|^2$$

$$= x^T(A^TA)x + 2(A^Tb)x + b^Tb$$

$$\frac{\partial E(x)}{\partial x} = 2(A^TAx + A^Tb) = 0$$

$$A^TA\,x = -A^Tb$$

Nonlinear case

$$E(x) = \|\varepsilon(x)\|^2$$

$$x^* = \hat{x} + \delta_x$$

$$\varepsilon(x^*) \approx \varepsilon(\hat{x}) + J\ \delta_x$$

Nonlinear Least Sqaures

$$x^* = \underset{x}{\operatorname{argmin}} E(x)$$

Linea case

$$E(x) = \|Ax + b\|^2$$

$$= x^T(A^T A)x + 2(A^T b)x + b^T b$$

$$\frac{\partial E(x)}{\partial x} = 2(A^T A x + A^T b) = 0$$

$$A^T A\, x = -A^T b$$

Nonlinear case

$$E(x) = \|\varepsilon(x)\|^2$$

$$x^* = \hat{x} + \delta_x$$

$$\varepsilon(x^*) \approx \varepsilon(\hat{x}) + \boxed{J}\, \delta_x$$

Jacobian matrix

$$J = \left.\frac{\partial \varepsilon}{\partial x}\right|_{x=\hat{x}}$$

Nonlinear Least Sqaures

$$x^* = \underset{x}{\operatorname{argmin}}\, E(x)$$

Linea case

$$E(x) = \|Ax + b\|^2$$

$$= x^T(A^T A)x + 2(A^T b)x + b^T b$$

$$\frac{\partial E(x)}{\partial x} = 2(A^T A x + A^T b) = 0$$

$$A^T A\, x = -A^T b$$

Nonlinear case

$$E(x) = \|\varepsilon(x)\|^2$$

$$x^* = \hat{x} + \delta_x$$

$$\varepsilon(x^*) \approx \varepsilon(\hat{x}) + \boxed{J}\, \delta_x$$

Jacobian matrix

$$J = \left.\frac{\partial \varepsilon}{\partial x}\right|_{x=\hat{x}}$$

$$E(x) \approx {\delta_x}^T (J^T J)\delta_x + 2(J^T \varepsilon)\delta_x + \varepsilon^T \varepsilon$$

$$J^T J\, \delta_x = -J^T \varepsilon$$

Nonlinear Least Sqaures

$$x^* = \operatorname*{argmin}_x E(x)$$

Linea case

$$E(x) = \|Ax + b\|^2$$

$$= x^T(A^TA)x + 2(A^Tb)x + b^Tb$$

$$\frac{\partial E(x)}{\partial x} = 2(A^TAx + A^Tb) = 0$$

$$\boxed{A^TA}\, x = -A^Tb$$

Nonlinear case

$$E(x) = \|\varepsilon(x)\|^2$$

$$x^* = \hat{x} + \delta_x$$

$$\varepsilon(x^*) \approx \varepsilon(\hat{x}) + \boxed{J}\, \delta_x$$

$$E(x) \approx \delta_x{}^T(J^TJ)\delta_x + 2(J^T\varepsilon)\delta_x + \varepsilon^T\varepsilon$$

$$\boxed{J^TJ}\, \delta_x = -J^T\varepsilon$$

Jacobian matrix

$$J = \left.\frac{\partial \varepsilon}{\partial x}\right|_{x=\hat{x}}$$

Hessian matrix

Nonlinear Least Sqaures

$$x^* = \underset{x}{\operatorname{argmin}} E(x)$$

Linea case

$$E(x) = \|Ax + b\|^2$$

$$= x^T(A^TA)x + 2(A^Tb)x + b^Tb$$

$$\frac{\partial E(x)}{\partial x} = 2(A^TAx + A^Tb) = 0$$

$$\boxed{\boxed{A^TA}\,x = -A^Tb}$$

Nonlinear case

$$E(x) = \|\varepsilon(x)\|^2$$

$$x^* = \hat{x} + \delta_x$$

$$\varepsilon(x^*) \approx \varepsilon(\hat{x}) + \boxed{J}\,\delta_x$$

Jacobian matrix

$$J = \left.\frac{\partial \varepsilon}{\partial x}\right|_{x=\hat{x}}$$

$$E(x) \approx {\delta_x}^T(J^TJ)\delta_x + 2(J^T\varepsilon)\delta_x + \varepsilon^T\varepsilon$$

$$\boxed{\boxed{J^TJ}\,\delta_x = -J^T\varepsilon}$$

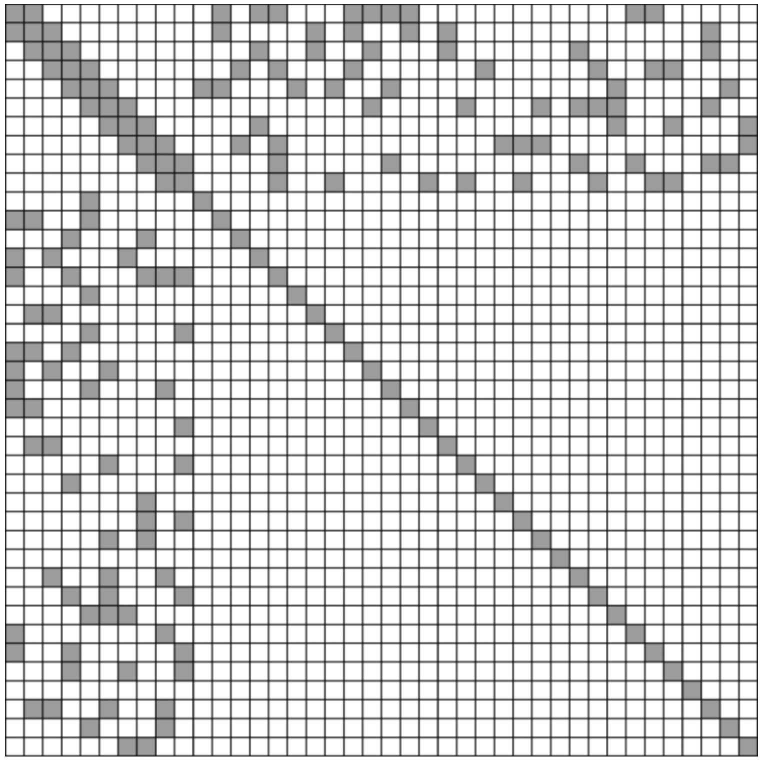
Hessian matrix

normal equation

Sparse Bundle Adjustment

$$\underset{C_1,\ldots C_{N_c},X_1,\ldots,X_{N_p}}{\operatorname{argmin}} \sum \left\| \pi(\underbrace{C_i}_{\boxed{\text{1 camera}}}, \underbrace{X_j}_{\boxed{\text{1 point}}}) - x_{ij} \right\|^2$$

Sparsity patten of Hessian



Sparse Bundle Adjustment

- An simple example
  - 3 cameras
  - 4 points
  - all points are visible in all cameras

Sparse Bundle Adjustment

$$J=\left(\begin{array}{ccccccc}
\multicolumn{3}{c}{\overbrace{\qquad\qquad\qquad}^{\text{3 cameras}}} & \multicolumn{4}{c}{\overbrace{\qquad\qquad\qquad\qquad}^{\text{4 points}}} \\
A_{11} & 0 & 0 & B_{11} & 0 & 0 & 0 \\
A_{12} & 0 & 0 & 0 & B_{12} & 0 & 0 \\
A_{13} & 0 & 0 & 0 & 0 & B_{13} & 0 \\
A_{14} & 0 & 0 & 0 & & 0 & B_{14} \\
0 & A_{21} & 0 & B_{21} & 0 & 0 & 0 \\
0 & A_{22} & 0 & 0 & B_{22} & 0 & 0 \\
0 & A_{23} & 0 & 0 & 0 & B_{23} & 0 \\
0 & A_{24} & 0 & 0 & 0 & 0 & B_{34} \\
0 & 0 & A_{31} & B_{31} & 0 & 0 & 0 \\
0 & 0 & A_{32} & 0 & B_{32} & 0 & 0 \\
0 & 0 & A_{33} & 0 & 0 & B_{33} & 0 \\
0 & 0 & A_{34} & 0 & 0 & 0 & B_{34}
\end{array}\right),e=\left.\begin{pmatrix} e_{11} \\ e_{12} \\ e_{13} \\ e_{14} \\ e_{21} \\ e_{22} \\ e_{23} \\ e_{24} \\ e_{31} \\ e_{32} \\ e_{33} \\ e_{34} \end{pmatrix}\right\}\text{12 re-projections}$$

Sparse Bundle Adjustment

$$J^T J \delta_x = -J^T \varepsilon$$

Sparse Bundle Adjustment

$$\boxed{J^T J}\delta_x = -J^T \varepsilon$$

$$J^T J = \begin{pmatrix} U & W \\ W^T & V \end{pmatrix} = \begin{pmatrix} U_1 & 0 & 0 & W_{11} & W_{12} & W_{13} & W_{14} \\ 0 & U_2 & 0 & W_{21} & W_{22} & W_{23} & W_{24} \\ 0 & 0 & U_3 & W_{31} & W_{32} & W_{33} & W_{34} \\ W_{11}^T & W_{21}^T & W_{31}^T & V_1 & 0 & 0 & 0 \\ W_{12}^T & W_{22}^T & W_{32}^T & 0 & V_2 & 0 & 0 \\ W_{13}^T & W_{21}^T & W_{33}^T & 0 & 0 & V_3 & 0 \\ W_{14}^T & W_{24}^T & W_{34}^T & 0 & 0 & 0 & V_4 \end{pmatrix}$$

$$U_i = \sum_{j=1}^{4} A_{ij}^T A_{ij}, V_j = \sum_{i=1}^{3} B_{ij}^T B_{ij}, W_{ij} = A_{ij}^T B_{ij}$$

Sparse Bundle Adjustment

$$J^T J \boxed{\delta_x} = -J^T \varepsilon$$

$$d_x = \begin{pmatrix} d_C \\ d_X \end{pmatrix} = \begin{pmatrix} d_{C_1}^T & d_{C_2}^T & d_{C_3}^T & d_{X_1}^T & d_{X_2}^T & d_{X_3}^T & d_{X_4}^T \end{pmatrix}^T$$

Sparse Bundle Adjustment

$$J^T J \delta_x = -\boxed{J^T \varepsilon}$$

$$J^T e = \begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} u_1 & u_2 & u_3 & v_1 & v_2 & v_3 & v_4 \end{pmatrix}^T$$

$$u_i = \sum_{j=1}^{4} A_{ij}^T e_{ij}$$

$$v_j = \sum_{i=1}^{3} B_{ij}^T e_{ij}$$

Sparse Bundle Adjustment

- In general, NOT all points are visible in all cameras

$$U_i = \sum_{j=1}^{4} A_{ij}^T A_{ij}\,, V_j = \sum_{i=1}^{3} B_{ij}^T B_{ij}\,, W_{ij} = A_{ij}^T B_{ij}$$

  - $A_{ij} = B_{ij} = 0$ if j-th point is not observed in i-th camera
  - More sparse structure, more speedup

Sparse Bundle Adjustment

$$J^T J \delta_x = -J^T \varepsilon$$

$$\begin{pmatrix} U & W \\ W^T & V \end{pmatrix} \begin{pmatrix} d_C \\ d_X \end{pmatrix} = - \begin{pmatrix} u \\ v \end{pmatrix}$$

$$\begin{pmatrix} U - WV^{-1}W^T & 0 \\ W^T & V \end{pmatrix} \begin{pmatrix} d_C \\ d_X \end{pmatrix} = - \begin{pmatrix} u - WV^{-1}v \\ v \end{pmatrix}$$

$S = U - WV^{-1}W^T$ Schur Complement

$Sd_C = -(u - WV^{-1}v)$ Compute cameras first (# cameras << # points)

$Vd_X = -v - W^T d_C$ back substitution for points

Schur Complement for Cameras

$$(U - WV^{-1}W^T)d_C = -(u - WV^{-1}v)$$

Schur Complement for Cameras

$$(U - \boxed{WV^{-1}W^T})d_C = -(u - WV^{-1}v)$$

$$WV^{-1}W^T = \begin{pmatrix} S_{11} & S_{12} & S_{13} \\ S_{12}^T & S_{22} & S_{23} \\ S_{13}^T & S_{23}^T & S_{33} \end{pmatrix}$$

$$S_{i_1 i_2} = \sum_{j=1}^{4} W_{i_1 j} V_j^{-1} W_{i_2 j}^T$$

Schur Complement for Cameras

$$(U - WV^{-1}W^T)d_C = -(u - \boxed{WV^{-1}v})$$

$$WV^{-1}e_X = \begin{pmatrix} g_1 \\ g_2 \\ g_3 \end{pmatrix}$$

$$g_i = \sum_{j=1}^{4} W_{ij} V_j^{-1} v_j$$

Schur Complement for Cameras

- Again, in general NOT all points are visible in all cameras

$$S_{i_1 i_2} = \sum_{j=1}^{4} W_{i_1 j} V_j^{-1} W_{i_2 j}^T$$

  - $S_{i_1 i_2} = 0$ if i_1-th camera has no common points with i_2-th camera
  - More sparse structure more speedup

Back Substitution for Points

$$\begin{pmatrix} U & W \\ W^T & V \end{pmatrix} \begin{pmatrix} d_C \\ d_X \end{pmatrix} = -\begin{pmatrix} u \\ v \end{pmatrix}$$

$$(U - WV^{-1}W^T)d_C = -(u - WV^{-1}v)$$

$$W^T d_C + V d_X = -v$$

$$V_{ij} \quad d_{X_j} = -v_j - \sum_{i=1}^{3} W_{ij}^T d_{C_i}$$

- Each point can be solve independently
- Again, $W_{ij} = 0$ if j-th point is not observed in i-th camera

Probability Interpretation

$$\begin{pmatrix} U & W \\ W^T & V \end{pmatrix}\begin{pmatrix} d_C \\ d_X \end{pmatrix} = -\begin{pmatrix} u \\ v \end{pmatrix}$$

joint density $\mathrm{P}(\delta_C, \delta_X) = P(\delta_C)P(\delta_X|\delta_C)$

$$(U - WV^{-1}W^T)d_C = -(u - WV^{-1}v)$$

marginalize out $\mathrm{P}(\delta_X)$ to get $\mathrm{P}(\delta_C)$

$$W^T d_C + V d_X = -v$$

conditional probability $P(\delta_X|\delta_C)$

Factor Graph Interpretation

$$\begin{pmatrix} U & W \\ W^T & V \end{pmatrix} \begin{pmatrix} d_C \\ d_X \end{pmatrix} = -\begin{pmatrix} u \\ v \end{pmatrix}$$

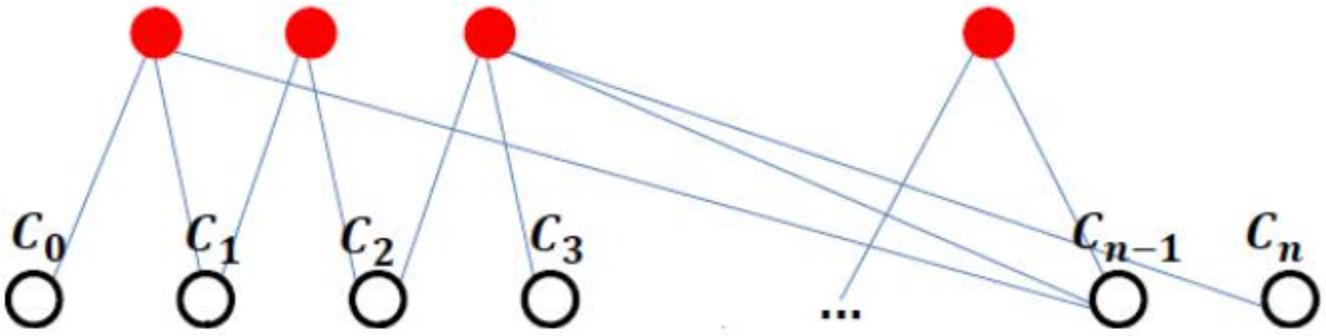
joint density $\mathrm{P}(\delta_C, \delta_X) = P(\delta_C)P(\delta_X|\delta_C)$

$$(U - WV^{-1}W^T)d_C = -(u - WV^{-1}v)$$

marginalize out $\mathrm{P}(\delta_X)$ to get $\mathrm{P}(\delta_C)$

$$W^T d_C + V d_X = -v$$

conditional probability $P(\delta_X|\delta_C)$



Factor Graph Interpretation

$$\begin{pmatrix} U & W \\ W^T & V \end{pmatrix}\begin{pmatrix} d_C \\ d_X \end{pmatrix} = -\begin{pmatrix} u \\ v \end{pmatrix}$$

joint density $\mathrm{P}(\delta_C, \delta_X) = P(\delta_C)P(\delta_X|\delta_C)$

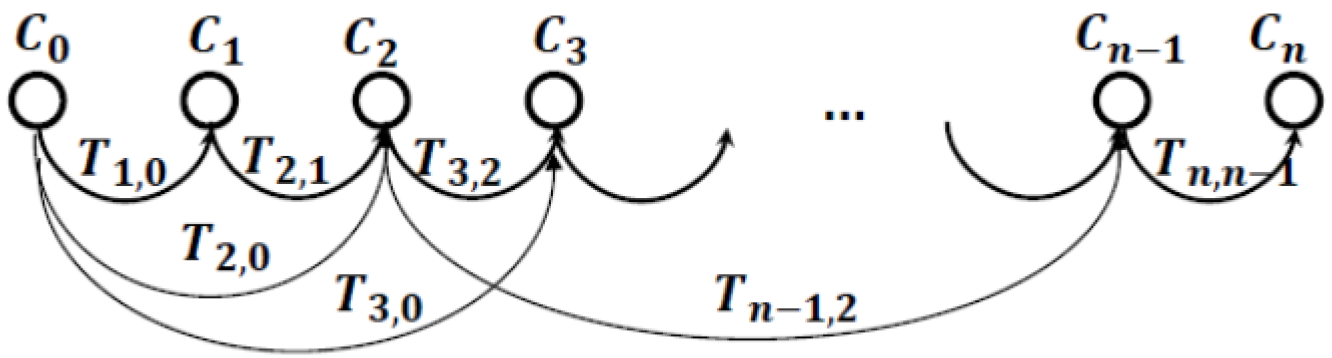
$$(U - WV^{-1}W^T)d_C = -(u - WV^{-1}v)$$

marginalize out $\mathrm{P}(\delta_X)$ to get $\mathrm{P}(\delta_C)$

$$W^T d_C + V d_X = -v$$

conditional probability $P(\delta_X|\delta_C)$

$$\underset{C_1,\ldots C_{N_C}}{\operatorname{argmin}} \sum \left\| C_i \ominus C_j \ominus T_{i,j} \right\|^2$$



Pose Graph Optimization

$$\begin{pmatrix} U & W \\ W^T & V \end{pmatrix}\begin{pmatrix} d_C \\ d_X \end{pmatrix} = -\begin{pmatrix} u \\ v \end{pmatrix}$$

joint density $\mathrm{P}(\delta_C, \delta_X) = P(\delta_C)P(\delta_X|\delta_C)$

$$(U - WV^{-1}W^T)d_C = -(u - WV^{-1}v)$$

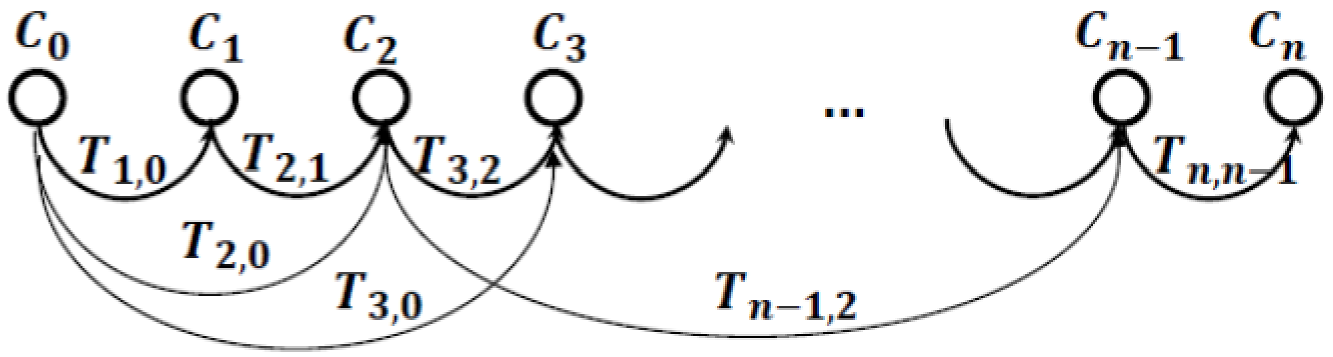
marginalize out $\mathrm{P}(\delta_X)$ to get $\mathrm{P}(\delta_C)$

~~$W^T d_C + V d_X = -v$~~

conditional probability $P(\delta_X|\delta_C)$

$$\operatorname*{argmin}_{C_1,\dots C_{N_C}} \sum \left\|C_i \ominus C_j \ominus T_{i,j}\right\|^2$$

Pose graph optimization is an approximation of BA



Steps in BA

$$\begin{pmatrix} \boxed{\begin{matrix} U & W \\ W^T & V \end{matrix}} \end{pmatrix} \begin{pmatrix} d_C \\ d_X \end{pmatrix} = - \begin{pmatrix} \boxed{\begin{matrix} u \\ v \end{matrix}} \end{pmatrix}$$

$$(U - WV^{-1}W^T)d_C = -(u - WV^{-1}v)$$

$$W^T d_C + V d_X = -v$$

1. Construct normal equation

```
$\mathbf{U} = \mathbf{0}$; $\mathbf{V} = \mathbf{0}$; $\mathbf{W} = \mathbf{0}$; $\mathbf{u} = \mathbf{0}$; $\mathbf{v} = \mathbf{0}$
for each point $j$ and each camera $i \in \mathcal{V}_j$ do
    Construct linearized equation (11)
    $\mathbf{U}_{ii} += \mathbf{J}_{\mathbf{C}_{ij}}^\top \mathbf{J}_{\mathbf{C}_{ij}}$
    $\mathbf{V}_{jj} += \mathbf{J}_{\mathbf{X}_{ij}}^\top \mathbf{J}_{\mathbf{X}_{ij}}$
    $\mathbf{u}_i += \mathbf{J}_{\mathbf{C}_{ij}}^\top \mathbf{e}_{ij}$
    $\mathbf{v}_j += \mathbf{J}_{\mathbf{X}_{ij}}^\top \mathbf{e}_{ij}$
    $\mathbf{W}_{ij} = \mathbf{J}_{\mathbf{C}_{ij}}^\top \mathbf{J}_{\mathbf{X}_{ij}}$
end for
```

Steps in BA

$$\begin{pmatrix} U & W \\ W^T & V \end{pmatrix} \begin{pmatrix} d_C \\ d_X \end{pmatrix} = -\begin{pmatrix} u \\ v \end{pmatrix}$$

$$\left(\boxed{U - WV^{-1}W^T}\right) d_C = -\left(\boxed{u - WV^{-1}v}\right)$$

$$W^T d_C + V d_X = -v$$

1. Construct normal equation
2. Construct Schur complement

```
$\mathbf{S} = \mathbf{U}$
for each point $j$ and each camera pair $(i_1, i_2) \in \mathcal{V}_j \times \mathcal{V}_j$
do
    $\mathbf{S}_{i_1 i_2} -= \mathbf{W}_{i_1 j} \mathbf{V}_{jj}^{-1} \mathbf{W}_{i_2 j}^{\top}$
end for
$\mathbf{g} = \mathbf{u}$
for each point $j$ and each camera $i \in \mathcal{V}_j$ do
    $\mathbf{g}_i -= \mathbf{W}_{ij} \mathbf{V}_{jj}^{-1} \mathbf{v}_j$
end for
```

Steps in BA

$$\begin{pmatrix} U & W \\ W^T & V \end{pmatrix}\begin{pmatrix} d_C \\ d_X \end{pmatrix} = -\begin{pmatrix} u \\ v \end{pmatrix}$$

$$(U - WV^{-1}W^T)\boxed{d_C} = -(u - WV^{-1}v)$$

$$W^T d_C + V d_X = -v$$

1. Construct normal equation
2. Construct Schur complement
3. Solve cameras
   - Sparse Cholesky factorization
   - Preconditioned Conjugate Gradient (PCG)
     - explicitly leverages the sparseness

Steps in BA

$$\begin{pmatrix} U & W \\ W^T & V \end{pmatrix} \begin{pmatrix} d_C \\ d_X \end{pmatrix} = -\begin{pmatrix} u \\ v \end{pmatrix}$$

$$(U - WV^{-1}W^T)d_C = -(u - WV^{-1}v)$$

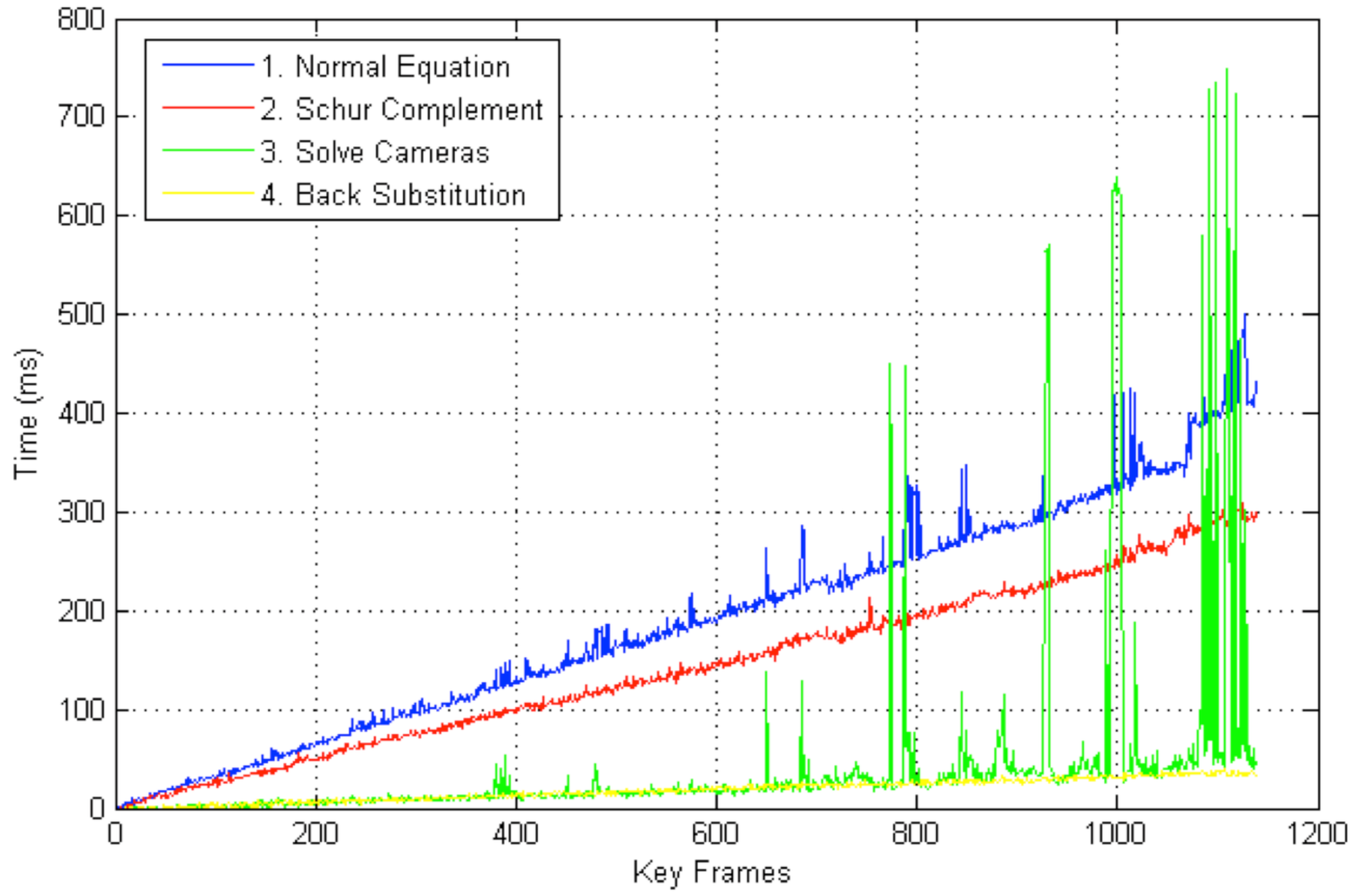
$$W^T d_C + V\boxed{d_X} = -v$$

1. Construct normal equation
2. Construct Schur complement
3. Solve cameras
4. Solve points

```
for each point $j$ do
    $\delta \mathbf{x}_j = \mathbf{V}_{jj}^{-1}\left(\mathbf{v}_j - \sum_{i \in \mathcal{V}_j} \mathbf{W}_{ij}^\top \delta \mathbf{c}_i\right)$
end for
```

Runtime for Each Steps

- Runtime increases with #cameras



Challenge of BA

- Efficiency is the main challenge of BA
- Keyframe or pose graph simplification cannot completely solve this problem
- Two scenarios
    - Large scale SfM
    - Realtime SLAM

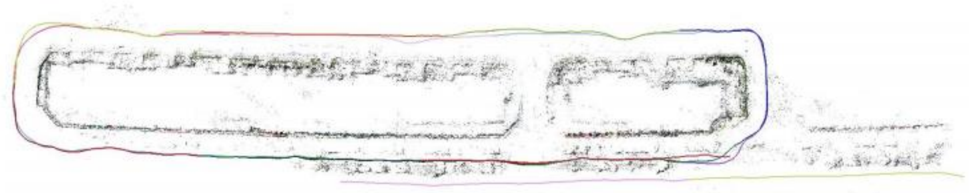
Outline

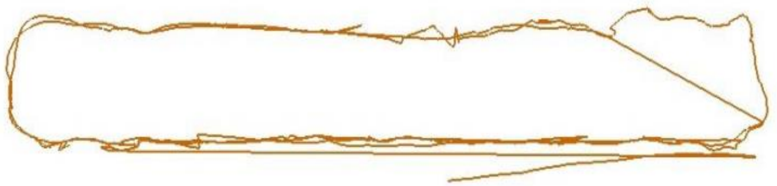
- Theories in BA
- BA for large scale SfM
- BA for realtime SLAM

Outline

- Theories in BA
- BA for large scale SfM
- BA for realtime SLAM

Challenges for Large-scale SfM





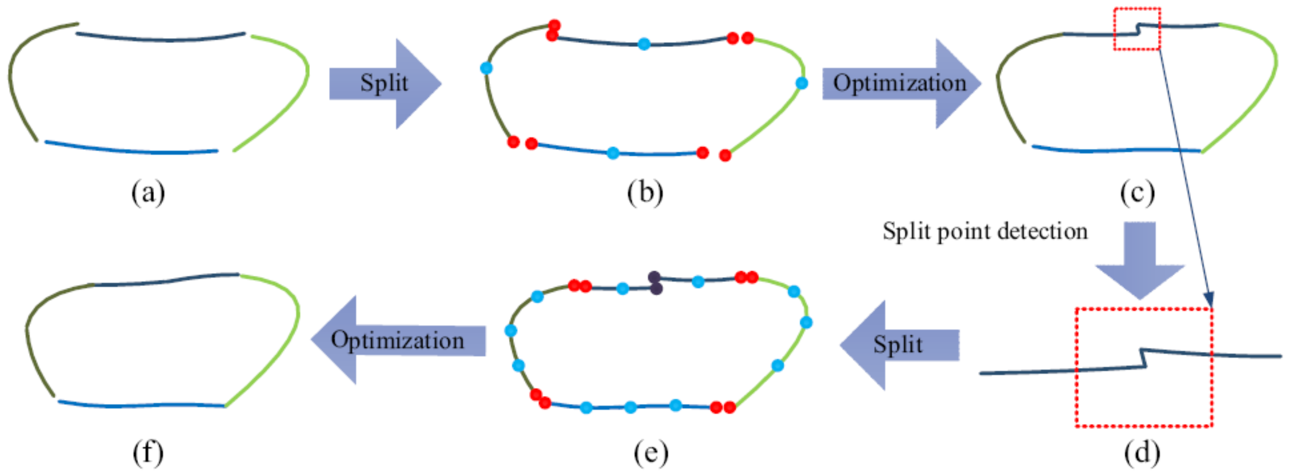
- Global BA
  - Huge #variables
  - Memory limit
  - Time-consuming
- Iterative local BA
  - Large error is difficult to be propagated to whole scene
  - Easily stuck in local optimum
- Pose graph optimization
  - Approximation of BA
  - May not sufficiently minimize the error
- Solutions
  - Hierarchical BA
  - Distributed BA

Segment-based Hierarchical BA

Zhang G, Liu H, Dong Z, et al. Efficient non-consecutive feature tracking for robust structure-from-motion[J]. IEEE Transactions on Image Processing, 2016, 25(12): 5957-5970.

Segment-based Hierarchical BA

- Observations
  - Incremental SfM results in high local accuracy, but low global accuracy
  - The DoF is unnecessarily large by traditional BA
- Solution
  - Split a long sequence to multiple short sub-sequences
  - 7-DoF similarity transformation for each sub-sequence
  - Only optimize overlapping points
  - Hierarchically align sub-sequences



Split Point Detection

- The split point should be at the place where the relative pose error is large, which is unknown in advance
- Naïve solution
  - large re-projection error
  - cannot reliably reflect the relative pose error
- Our solution
  - Revisit the normal equation

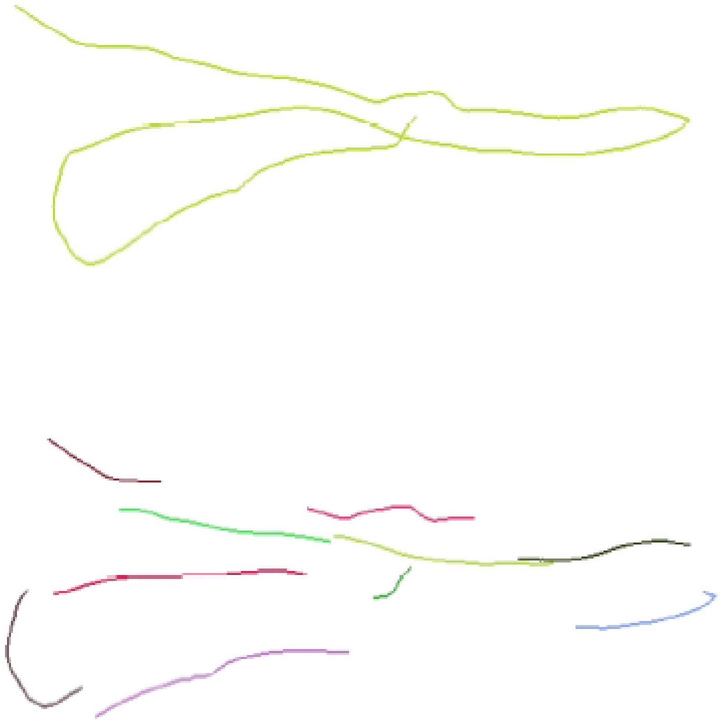
$$\begin{pmatrix} U & W \\ W^T & V \end{pmatrix} \begin{pmatrix} \delta_C \\ \delta_X \end{pmatrix} = -\begin{pmatrix} u \\ v \end{pmatrix}$$

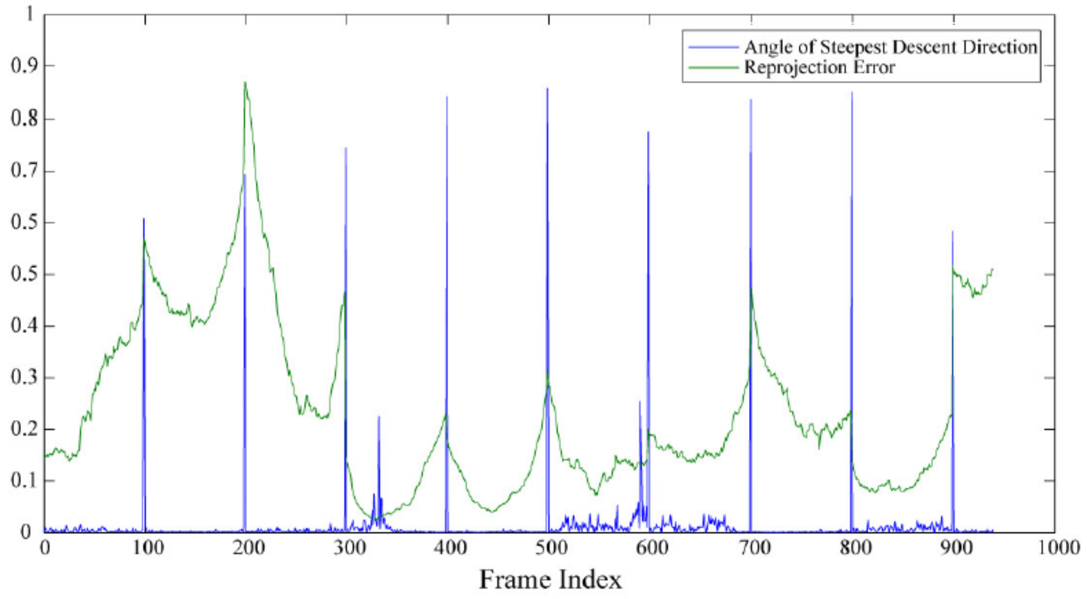
$$u_i = \sum_j A_{ij}^T \varepsilon_{ij}$$

  - ε_{ij} in i-th frame will be best minimized along u_i
  - The inconsistency between i-th and $(i+1)$-th frame

$$E(i, i+1) = \arccos\left(\frac{u_i^T u_{i+1}}{\|u_i\| \|u_{i+1}\|}\right)$$

Split Point Detection



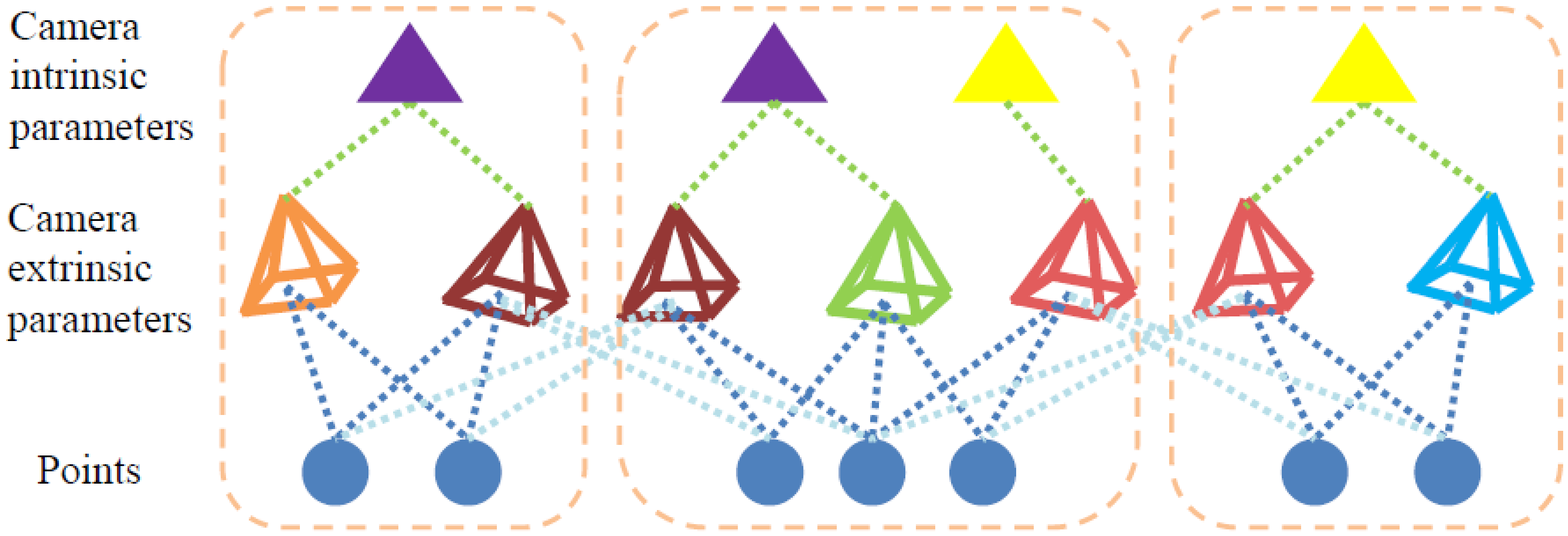


Distributed BA by Global Camera Consensus

Zhang R, Zhu S, Fang T, et al. Distributed very large scale bundle adjustment by global camera consensus[C]//Proceedings of the IEEE International Conference on Computer Vision. 2017: 29-38.

Split Cameras or Points

- Split cameras
  - Broadcast overlapping points, huge overhead
- Split points
  - Broadcast overlapping cameras, called camera consensus



ADMM for Constrained Optimization

- Constrained optimization

$$\begin{aligned} &\text{minimize} && f(\mathbf{x}) + g(\mathbf{z}) \\ &\text{subject to} && \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{z} = \mathbf{w} \end{aligned}$$

- The ADMM algorithm

$$\begin{aligned} L_\rho(\mathbf{x}, \mathbf{z}, \mathbf{y}) = f(\mathbf{x}) &+ g(\mathbf{z}) \\ &+ \mathbf{y}^T(\mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{z} - \mathbf{w}) \\ &+ \frac{\rho}{2}||\mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{z} - \mathbf{w}||_2^2 \end{aligned}$$

$$\begin{aligned} \mathbf{x}^{t+1} &= \arg\min_{\mathbf{x}} L_\rho(\mathbf{x}, \mathbf{z}^t, \mathbf{y}^t) \\ \mathbf{z}^{t+1} &= \arg\min_{\mathbf{z}} L_\rho(\mathbf{x}^{t+1}, \mathbf{z}, \mathbf{y}^t) \\ \mathbf{y}^{t+1} &= \mathbf{y}^t + \rho(\mathbf{A}\mathbf{x}^{t+1} + \mathbf{B}\mathbf{z}^{t+1} - \mathbf{w}) \end{aligned}$$

ADMM for Distributed BA

- Constrained optimization

$$\begin{aligned} &\text{minimize} && f(\mathbf{x}) + g(\mathbf{z}) \\ &\text{subject to} && \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{z} = \mathbf{w} \end{aligned}$$

- The ADMM algorithm

$$\begin{aligned} L_\rho(\mathbf{x}, \mathbf{z}, \mathbf{y}) = f(\mathbf{x}) &+ g(\mathbf{z}) \\ &+ \mathbf{y}^T(\mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{z} - \mathbf{w}) \\ &+ \frac{\rho}{2}||\mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{z} - \mathbf{w}||_2^2 \end{aligned}$$

$$\begin{aligned} \mathbf{x}^{t+1} &= \arg\min_{\mathbf{x}} L_\rho(\mathbf{x}, \mathbf{z}^t, \mathbf{y}^t) \\ \mathbf{z}^{t+1} &= \arg\min_{\mathbf{z}} L_\rho(\mathbf{x}^{t+1}, \mathbf{z}, \mathbf{y}^t) \\ \mathbf{y}^{t+1} &= \mathbf{y}^t + \rho(\mathbf{A}\mathbf{x}^{t+1} + \mathbf{B}\mathbf{z}^{t+1} - \mathbf{w}) \end{aligned}$$

- Distributed BA

$$\text{minimize} \sum_{i=1}^{n} f_i(\mathbf{x}_i)$$

$$\text{subject to } \mathbf{x}_i = \mathbf{z}, i = 1, \ldots, n$$

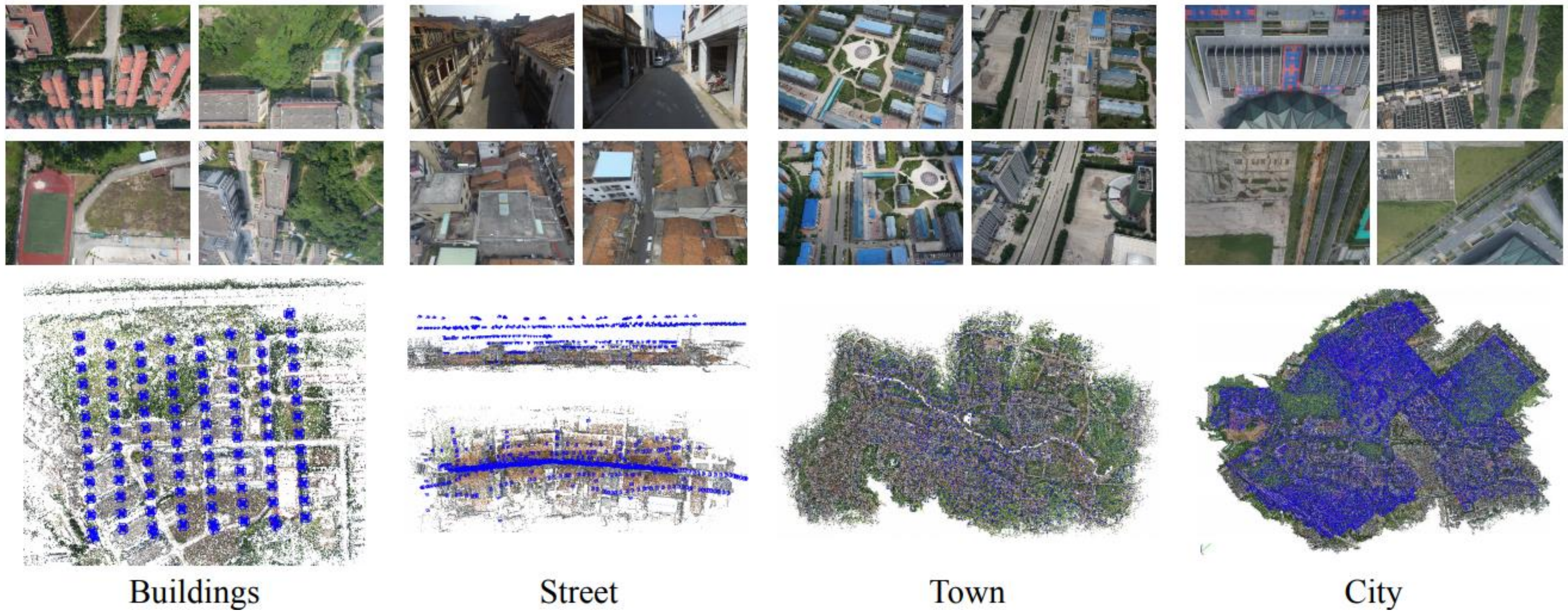
- Applying ADMM

$$\begin{aligned} \mathbf{x}_i^{t+1} = \arg\min_{\mathbf{x}_i} \Big(f_i(\mathbf{x}_i) + \left(\mathbf{y}_i^t\right)^T (\mathbf{x}_i - \mathbf{z}^t) \\ + \frac{\rho}{2}||\mathbf{x}_i - \mathbf{z}^t||_2^2 \Big) \end{aligned}$$

$$\mathbf{z}^{t+1} = \frac{1}{n}\sum_{i=1}^{n} \mathbf{x}_i^{t+1}$$

$$\mathbf{y}_i^{t+1} = \mathbf{y}_i^t + \rho(\mathbf{x}_i^{t+1} - \mathbf{z}^{t+1}), i = 1, \ldots, n$$

Large-scale SfM Results



Outline

- Theories in BA
- BA for large scale SfM
- BA for realtime SLAM

Outline

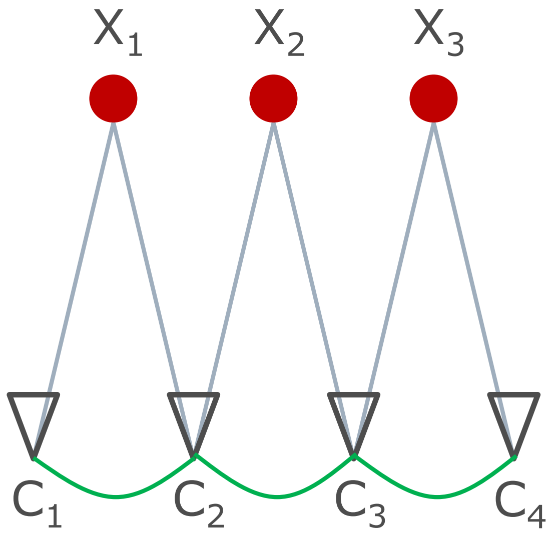
- Theories in BA
- BA for large scale SfM
- BA for realtime SLAM

Significance of BA Efficiency to SLAM

- Higher efficiency of BA means
  - Lower hardware requirement & power consumption
  - Longer sliding window to improve accuracy & robustness
  - Faster map expansion, better robustness

Batch VS Incremental BA

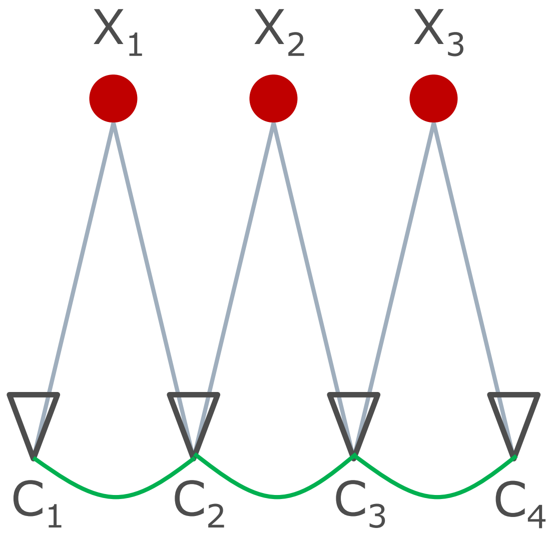
Batch BA



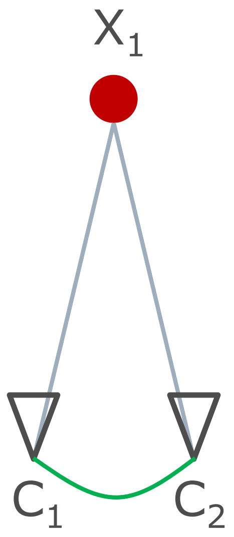
Incremental BA

Batch VS Incremental BA

Batch BA

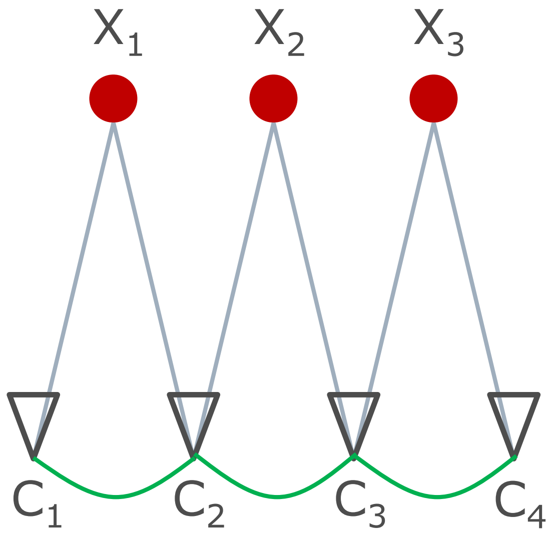


Incremental BA

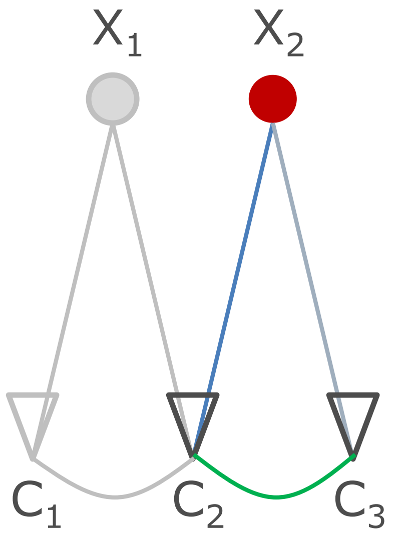


Batch VS Incremental BA

Batch BA

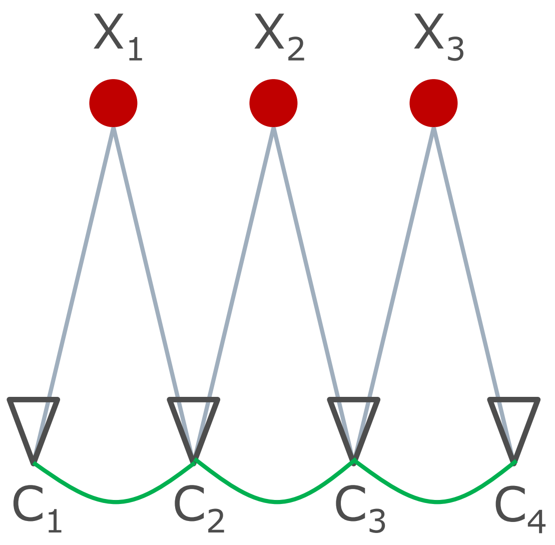


Incremental BA

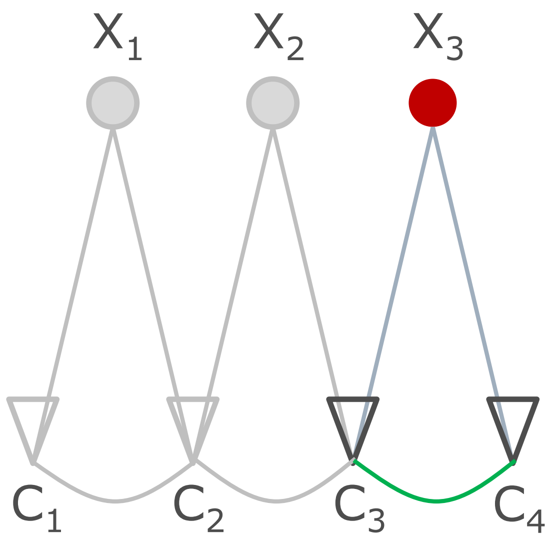


Batch VS Incremental BA

Batch BA



Incremental BA



Representative Methods of Incremental BA

- iSAM/iSAM2
  - Kaess M, Ranganathan A, Dellaert F. iSAM: Incremental smoothing and mapping[J]. IEEE Transactions on Robotics, 2008, 24(6): 1365-1378.
  - Kaess M, Johannsson H, Roberts R, et al. iSAM2: Incremental smoothing and mapping using the Bayes tree[J]. The International Journal of Robotics Research, 2012, 31(2): 216-235.
  - https://bitbucket.org/gtborg/gtsam/
- ICE-BA
  - Liu H, Chen M, Zhang G, et al. Ice-ba: Incremental, consistent and efficient bundle adjustment for visual-inertial slam[C]//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2018: 1974-1982.
  - https://github.com/baidu/ICE-BA
- SLAM++
  - Ila V, Polok L, Solony M, et al. Fast incremental bundle adjustment with covariance recovery[C]//2017 International Conference on 3D Vision (3DV). IEEE, 2017: 175-184.
  - https://sourceforge.net/p/slam-plus-plus/wiki/Home/

Incremental BA by iSAM2

Kaess, M., Johannsson, H., Roberts, R., Ila, V., Leonard, J. J., & Dellaert, F. (2012). iSAM2: Incremental smoothing and mapping using the Bayes tree. The International Journal of Robotics Research, 31(2), 216-235.

Solving Least Squares by QR Factorization

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \left\| A\boldsymbol{\theta} - \mathbf{b} \right\|^2$$

b: error vector

A: Jacobian matrix $\frac{\partial b}{\partial \theta}$

Solving Least Squares by QR Factorization

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \|A\boldsymbol{\theta} - \mathbf{b}\|^2$$

$$A = Q \begin{bmatrix} R \\ 0 \end{bmatrix}$$

b: error vector

A: Jacobian matrix $\frac{\partial b}{\partial \theta}$

R: upper triangular matrix

Solving Least Squares by QR Factorization

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \|A\boldsymbol{\theta} - \mathbf{b}\|^2$$

b: error vector

A: Jacobian matrix $\frac{\partial b}{\partial \theta}$

R: upper triangular matrix

$$A = Q \begin{bmatrix} R \\ 0 \end{bmatrix}$$

$$\begin{aligned}
\|A\boldsymbol{\theta} - \mathbf{b}\|^2 &= \left\| Q \begin{bmatrix} R \\ 0 \end{bmatrix} \boldsymbol{\theta} - \mathbf{b} \right\|^2 \\
&= \left\| Q^T Q \begin{bmatrix} R \\ 0 \end{bmatrix} \boldsymbol{\theta} - Q^T \mathbf{b} \right\|^2 \\
&= \left\| \begin{bmatrix} R \\ 0 \end{bmatrix} \boldsymbol{\theta} - \begin{bmatrix} \mathbf{d} \\ \mathbf{e} \end{bmatrix} \right\|^2 \\
&= \|R\boldsymbol{\theta} - \mathbf{d}\|^2 + \|\mathbf{e}\|^2
\end{aligned}$$

Solving Least Squares by QR Factorization

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \|A\boldsymbol{\theta} - \mathbf{b}\|^2$$

b: error vector

A: Jacobian matrix $\frac{\partial b}{\partial \theta}$

R: upper triangular matrix

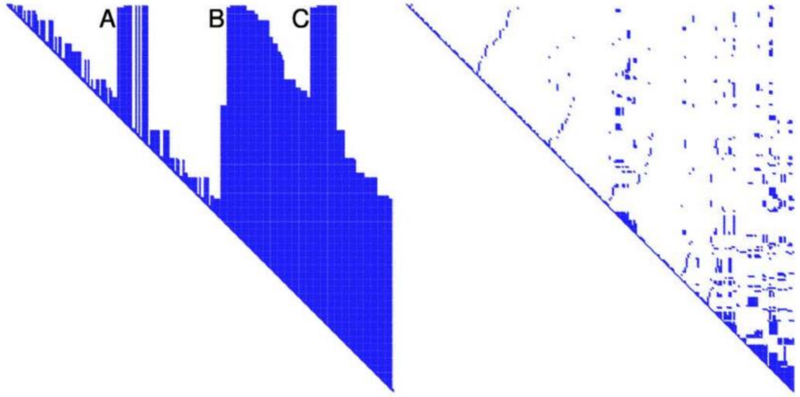
$$A = Q \begin{bmatrix} R \\ 0 \end{bmatrix}$$

$$\begin{aligned} \|A\boldsymbol{\theta} - \mathbf{b}\|^2 &= \left\| Q \begin{bmatrix} R \\ 0 \end{bmatrix} \boldsymbol{\theta} - \mathbf{b} \right\|^2 \\ &= \left\| Q^T Q \begin{bmatrix} R \\ 0 \end{bmatrix} \boldsymbol{\theta} - Q^T \mathbf{b} \right\|^2 \\ &= \left\| \begin{bmatrix} R \\ 0 \end{bmatrix} \boldsymbol{\theta} - \begin{bmatrix} \mathbf{d} \\ \mathbf{e} \end{bmatrix} \right\|^2 \\ &= \|R\boldsymbol{\theta} - \mathbf{d}\|^2 + \|\mathbf{e}\|^2 \end{aligned}$$

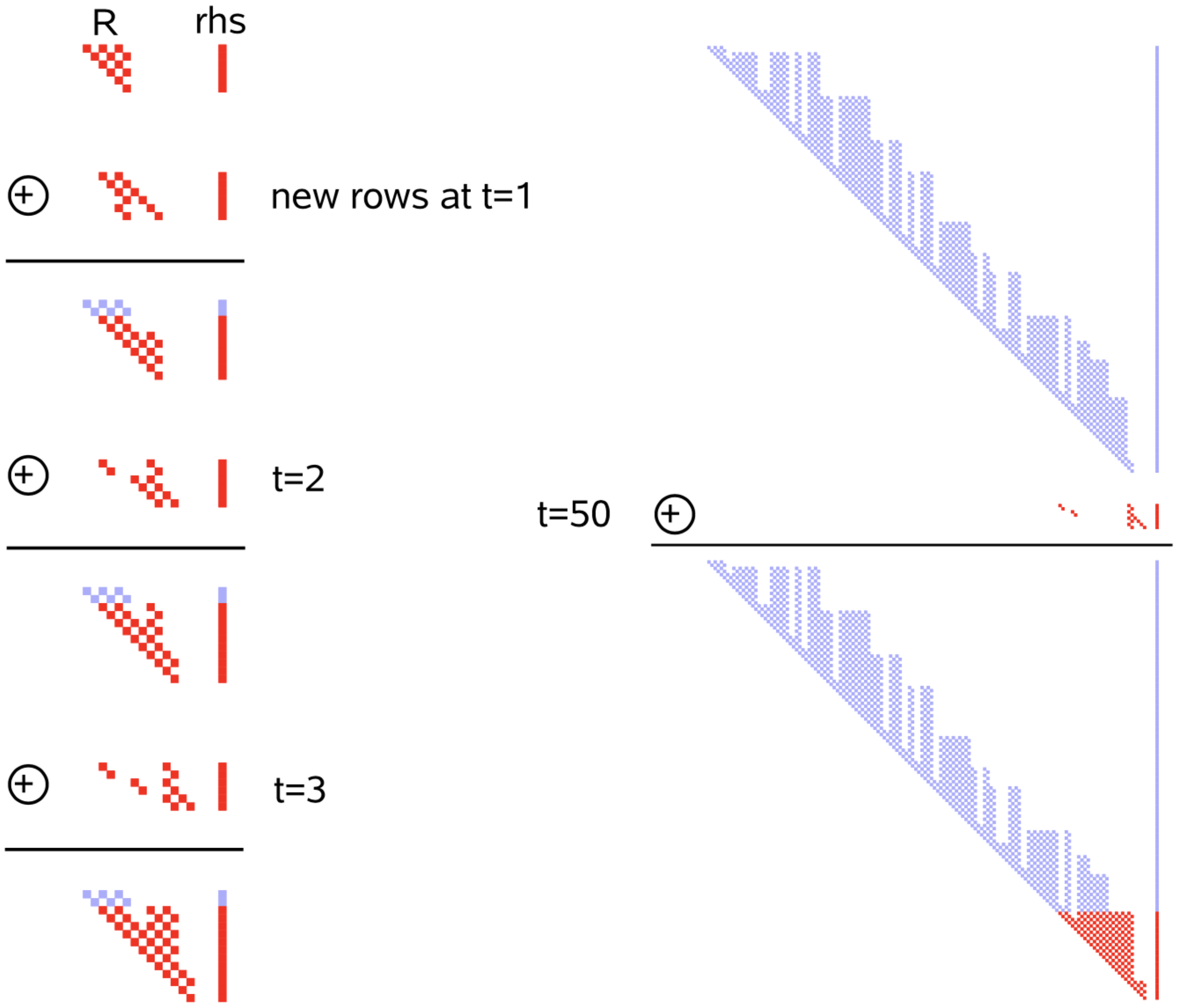
$$R\boldsymbol{\theta}^* = \mathbf{d}$$

QR Factorization VS Normal Equation

- Normal equation $(A^T A)\,\theta = -A^T b$
- QR Factorization $A = Q \begin{bmatrix} R \\ 0 \end{bmatrix}$ $R\boldsymbol{\theta}^* = \mathbf{d}$
  - Directly works on Jacobian A, numerically more stable
    - $\text{cond}(A) < \text{cond}(A^T A)$
  - The upper triangular matrix R can also be update incrementally
  - Efficiency largely depends on variable ordering

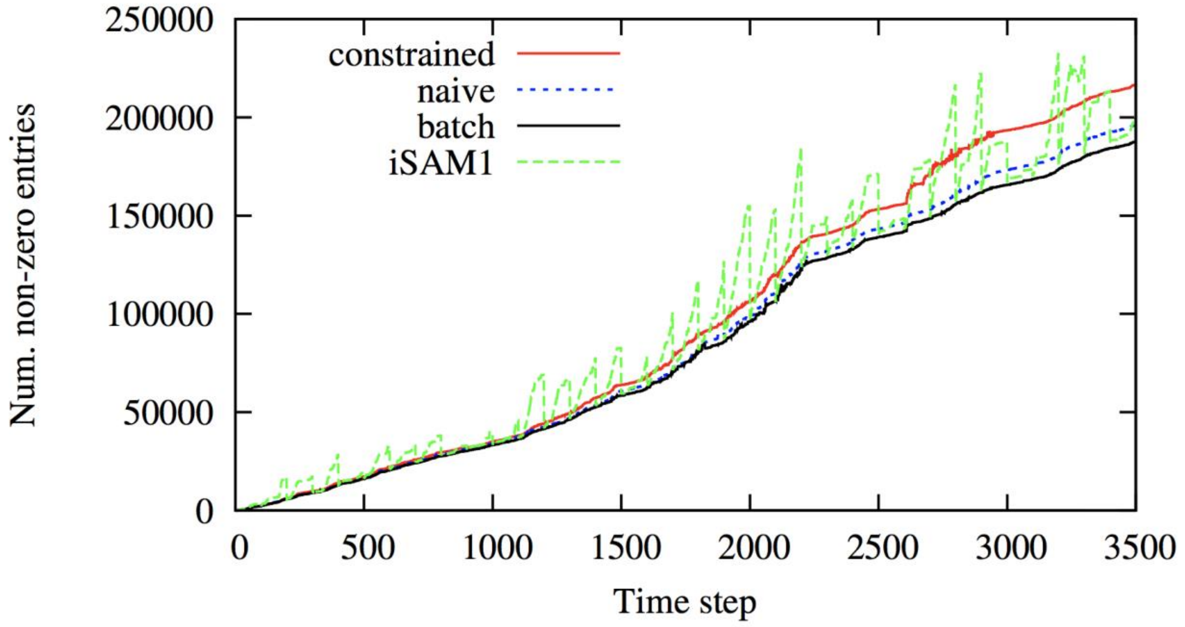


Example of iSAM



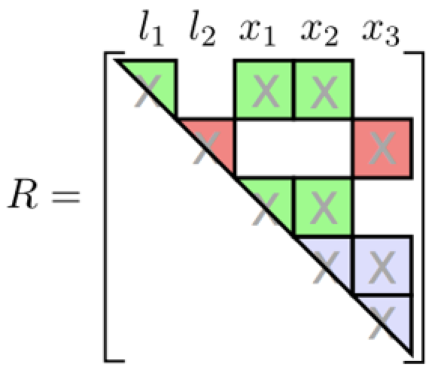
Limitation of iSAM

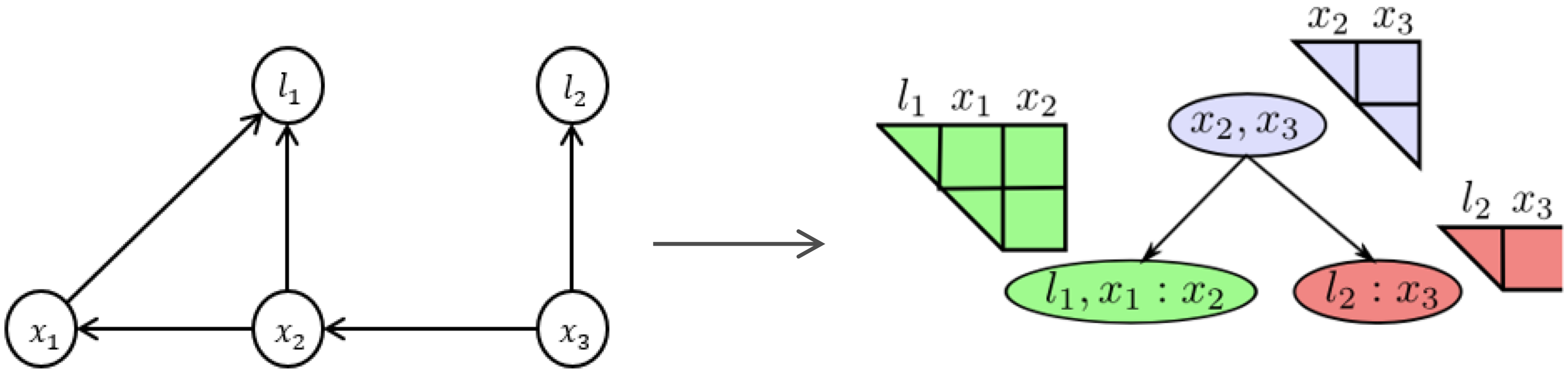
- Need periodically variable reordering by minimizing fill-ins
- It is difficult to provide the best ordering by algebraic method



iSAM2 by Bayes tree

- Alleviate the limitation of iSAM by Bayes tree
- Bayes tree encode the dependency relationship among variables

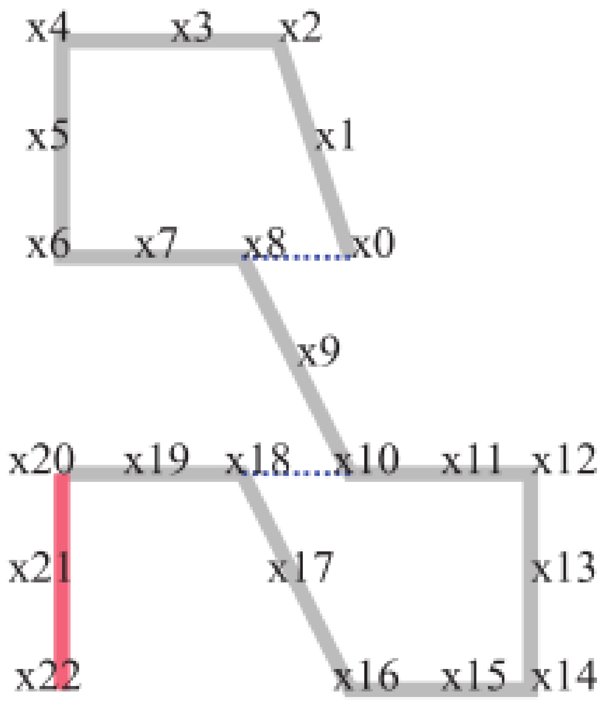


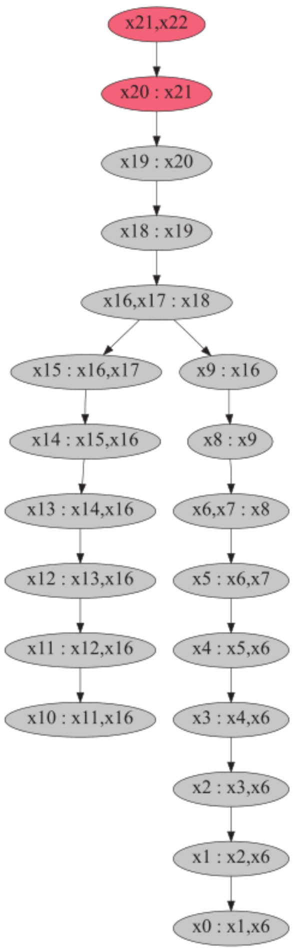


$l_1, x_1 : x_2$

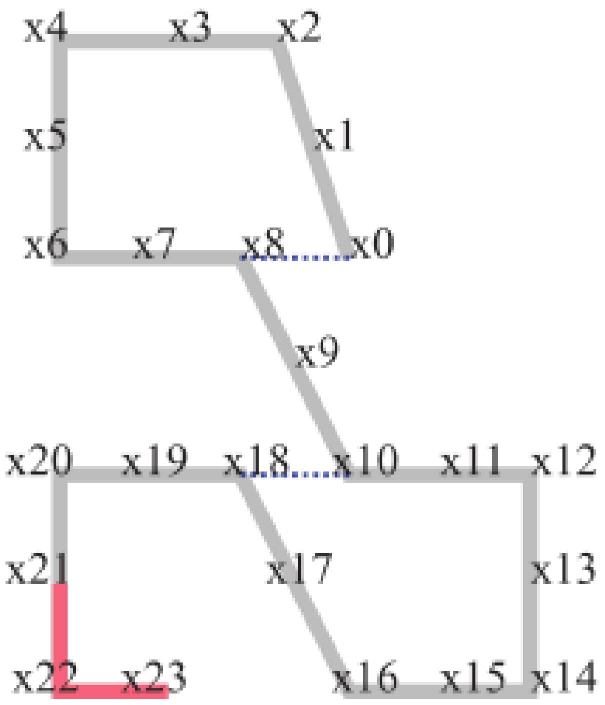
Clique encodes the conditional density

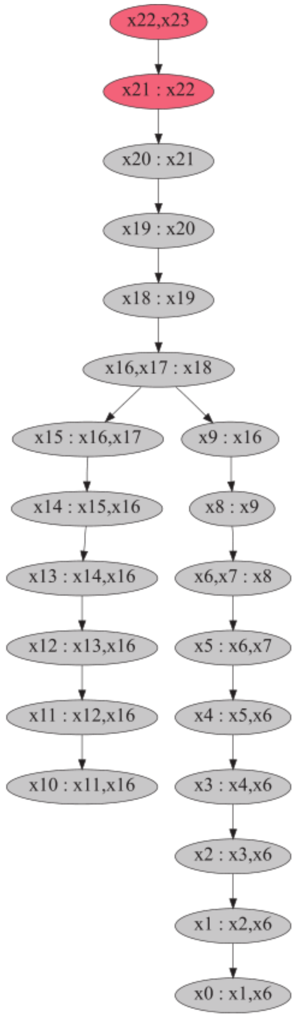
Example of iSAM2: Forward Motion



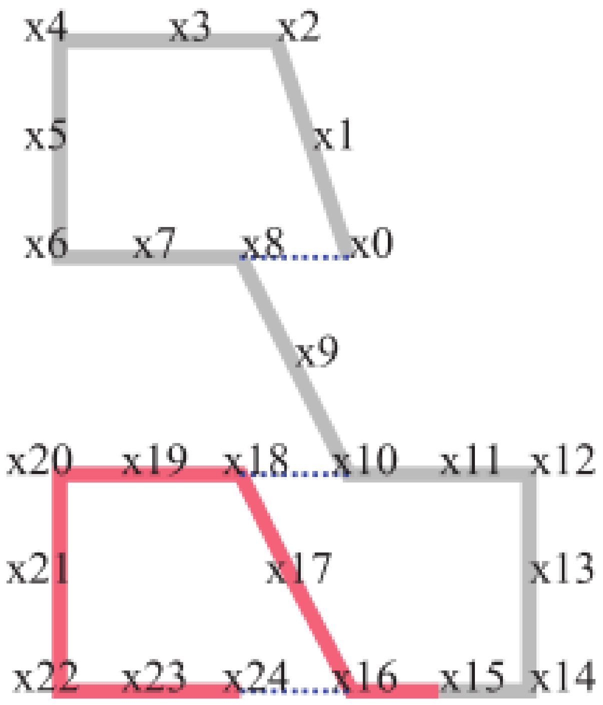


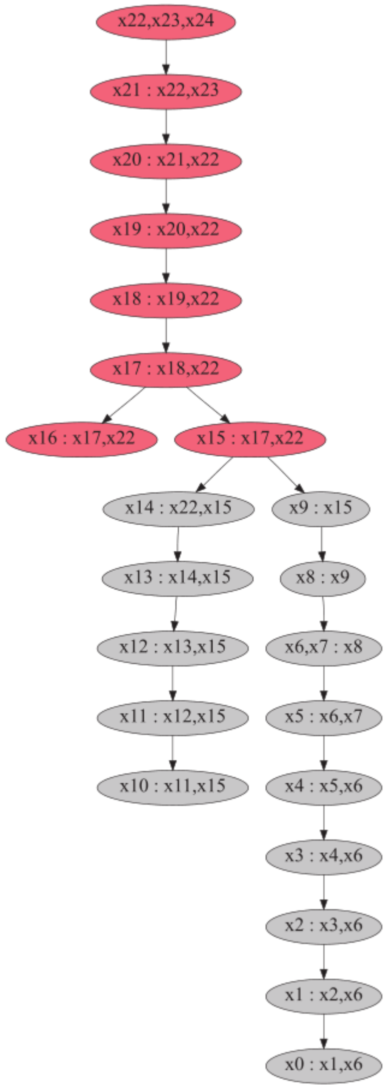
Example of iSAM2: Forward Motion



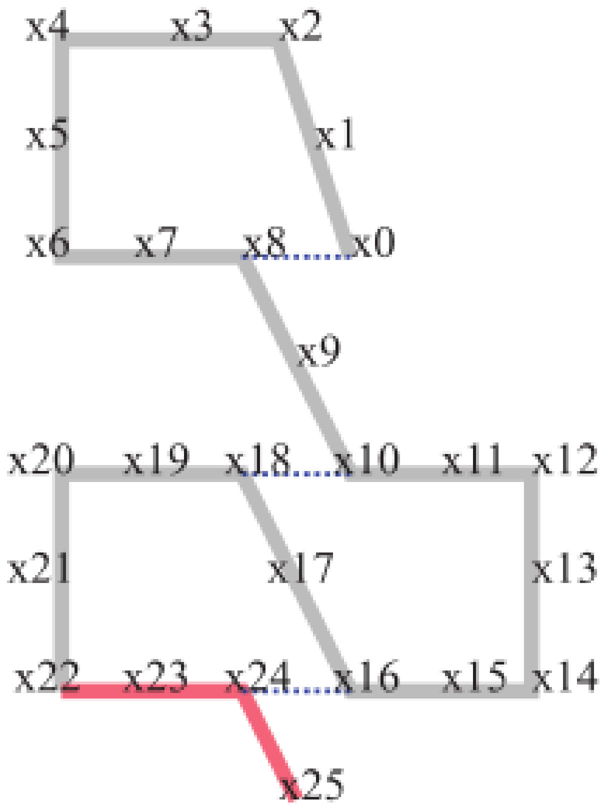


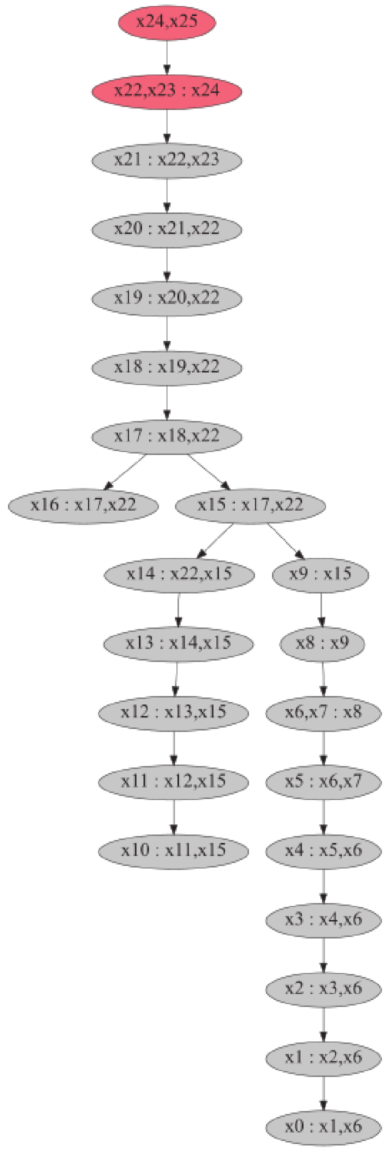
Example of iSAM2: Loop Detected



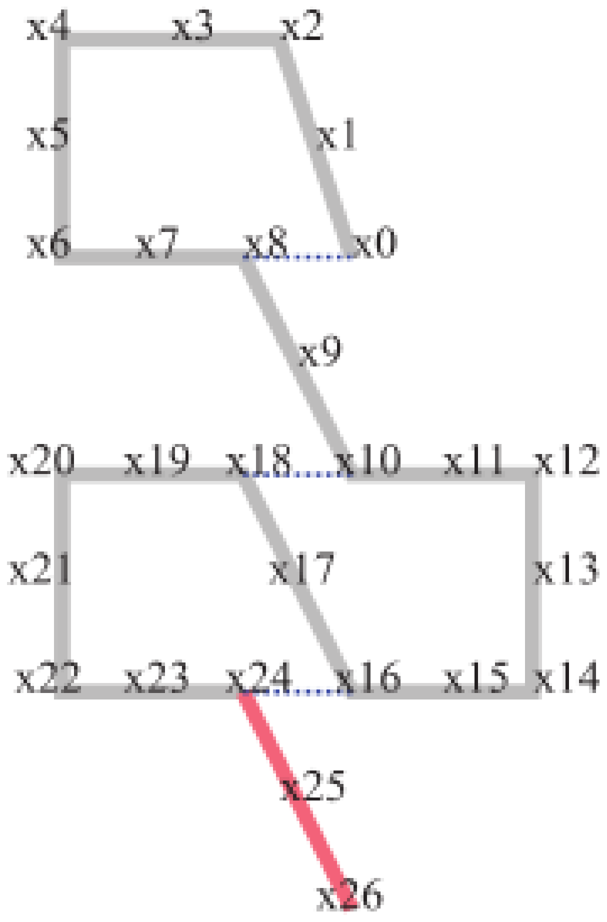


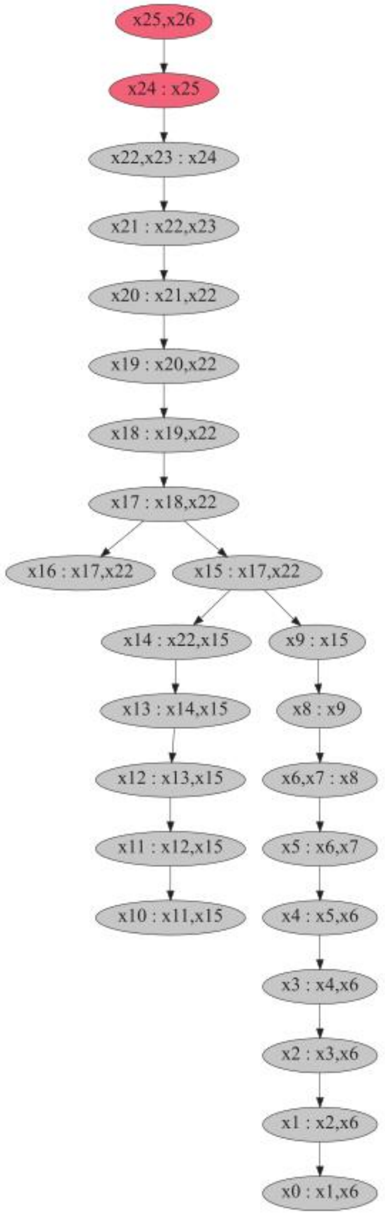
Example of iSAM2: Keep Going





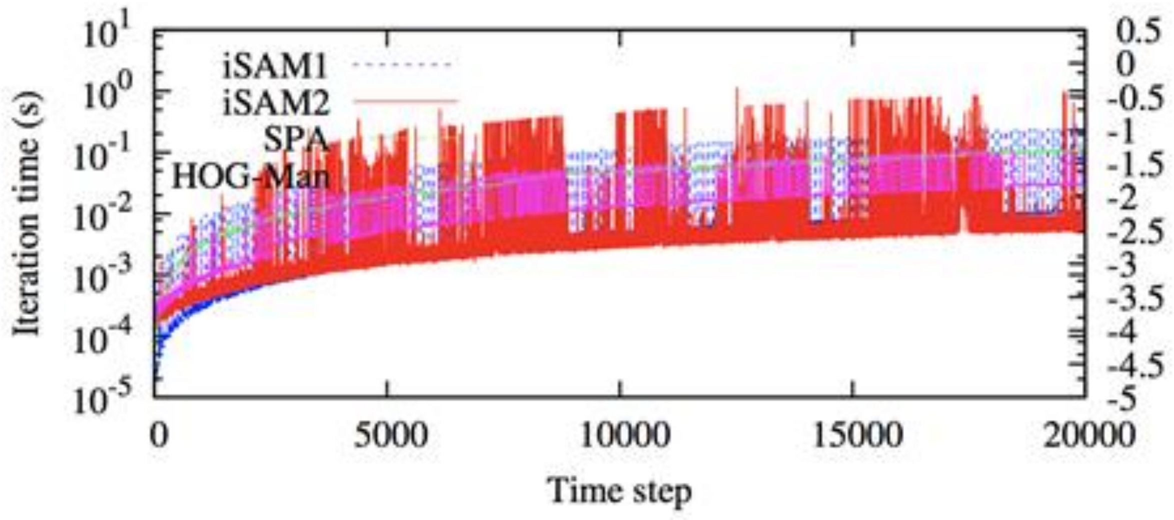
Example of iSAM2: Keep Going





Efficiency of iSAM2

- Improve iSAM most of time
- Many spikes
  - keep forward ✓
  - to and fro ✗
- It is difficult to provide the best ordering by algebraic method
  - Marginalizing points first is always better



Incremental BA by ICE-BA

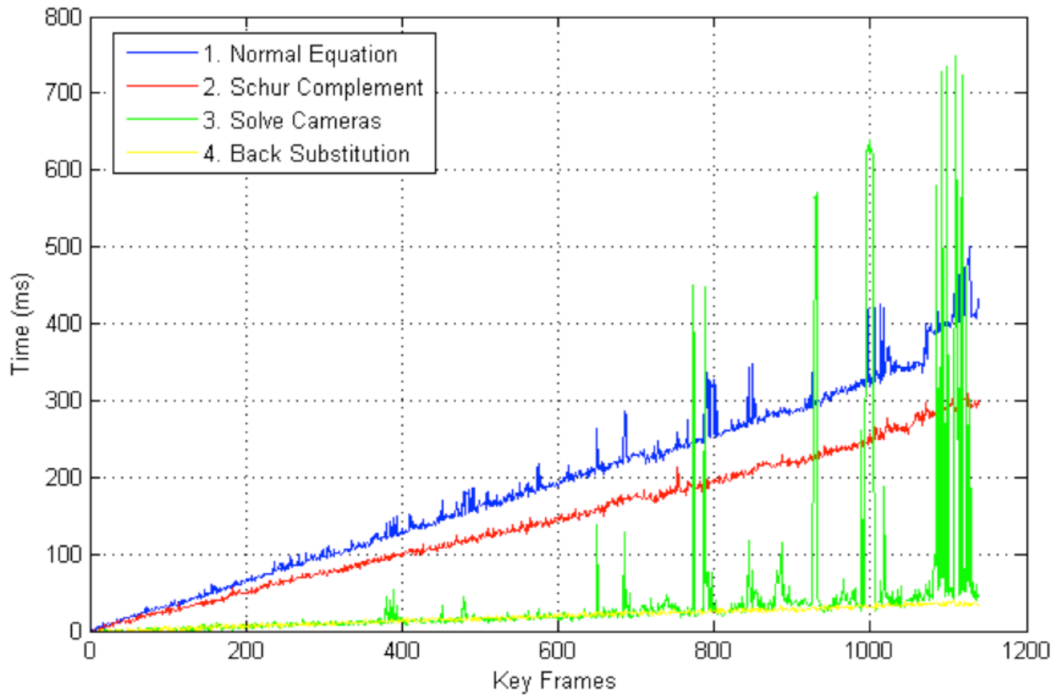
Liu H, Chen M, Zhang G, et al. ICE-BA: Incremental, Consistent and Efficient Bundle Adjustment for Visual-Inertial SLAM. CVPR 2018.

Steps of Standard BA

- Steps in one iteration
  1. normal equation
  2. Schur complement
  3. solve cameras
  4. solve points

Observations in Standard BA

- Runtime for steps 1, 2 >> 3, 4
  - #projections >> #cameras

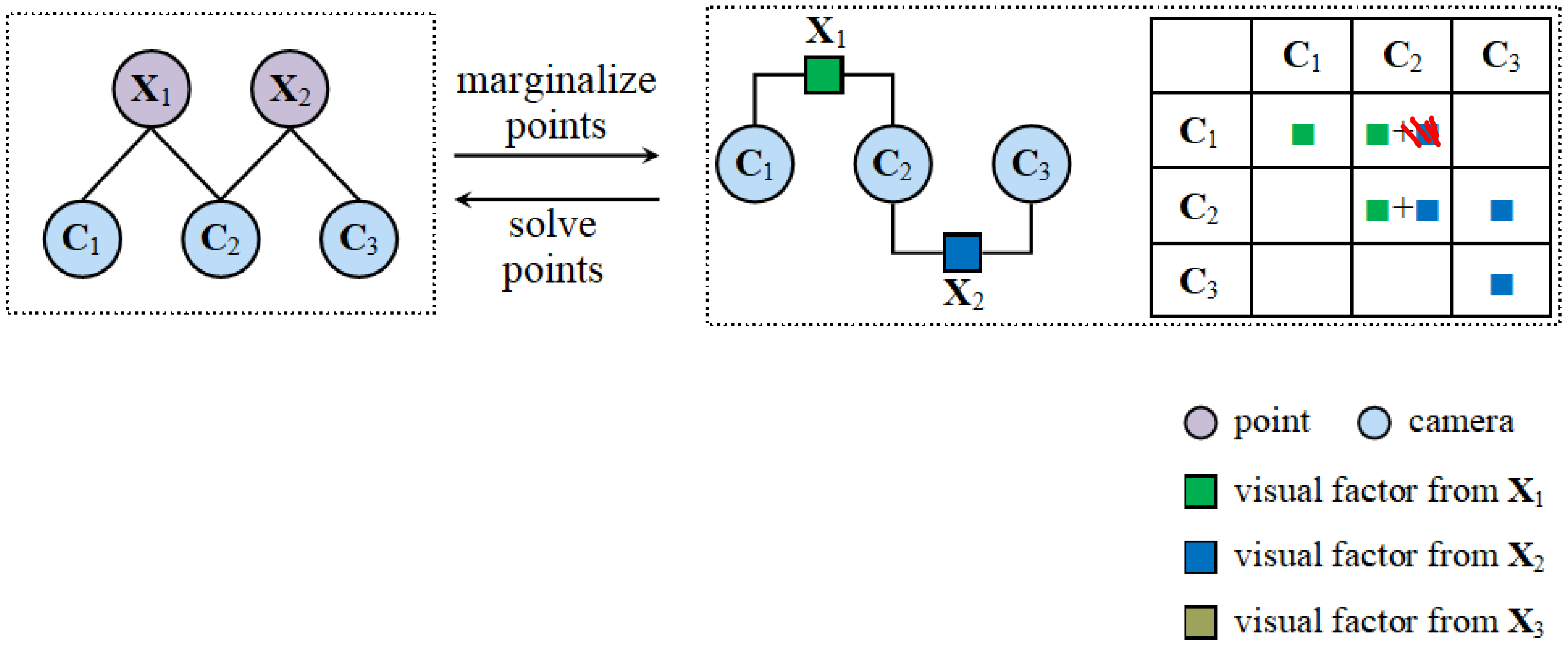


Observations in Standard BA

- Runtime for steps 1, 2 >> 3, 4
- Most cameras and points are nearly unchanged
  - Contribution of most functions nearly unchanged
  - No need to re-compute at each iteration

ICE-BA

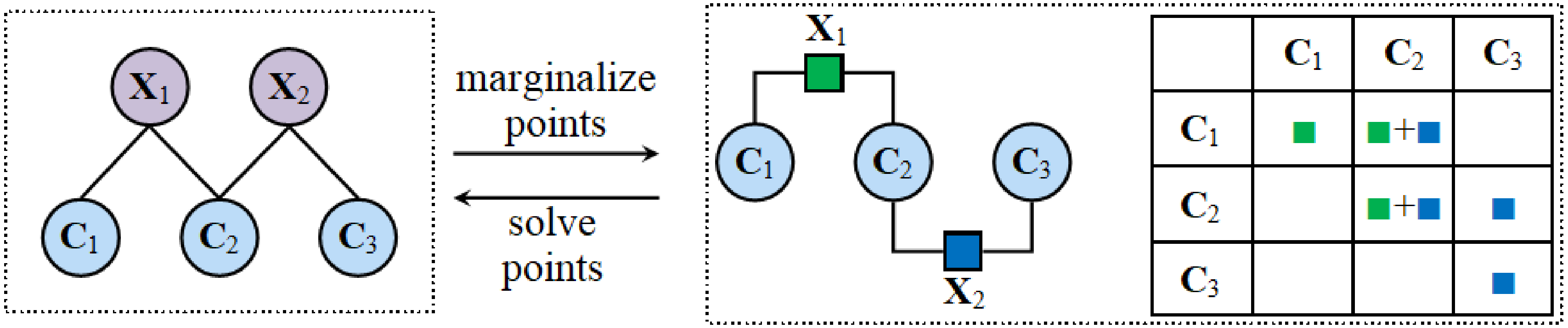
- Factor graph representation



	C_1	C_2	C_3
C_1	🟩	🟩+~~🟦~~	
C_2		🟩+🟦	🟦
C_3			🟦

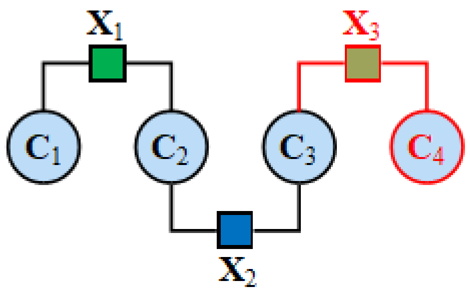
ICE-BA

- Factor graph representation



	C_1	C_2	C_3
C_1	🟩	🟩+🟦	
C_2		🟩+🟦	🟦
C_3			🟦

- New cameras or points come

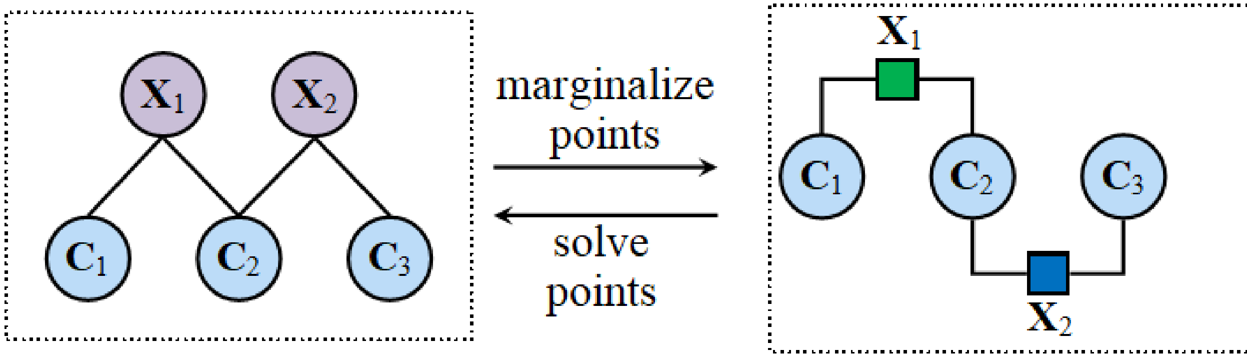


	C_1	C_2	C_3	C_4
C_1	🟩	🟩+~~🟦~~		
C_2		🟩+🟦	🟦+~~(🟫)~~	
C_3			🟦+(🟫)	(🟫)
C_4				(🟫)

● point	● camera
🟩	visual factor from X_1
🟦	visual factor from X_2
🟫	visual factor from X_3

ICE-BA

- Factor graph representation



	C_1	C_2	C_3
C_1	🟩	🟩+🟦	
C_2		🟩+🟦	🟦
C_3			🟦

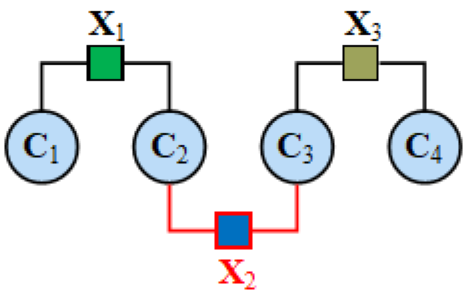
- Points have changed after iteration

○ point ○ camera

🟩 visual factor from X_1

🟦 visual factor from X_2

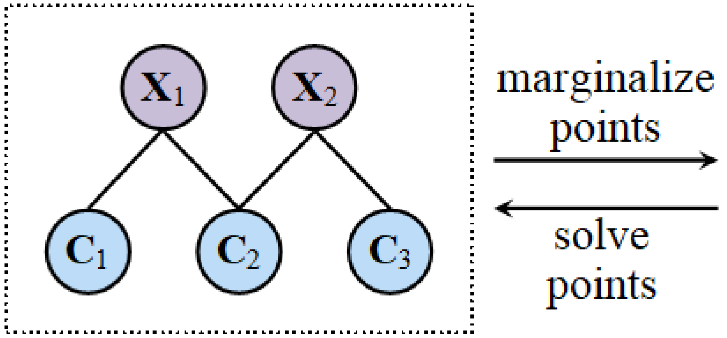
🟫 visual factor from X_3

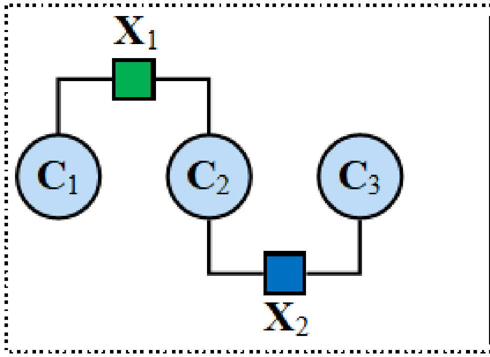


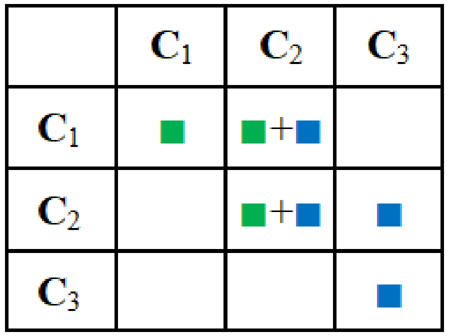
	C_1	C_2	C_3	C_4
C_1	🟩	🟩+(🟦)		
C_2		🟩+(🟦)	(🟦)+🟫	
C_3			(🟦)+🟫	🟫
C_4				🟫

ICE-BA

- Factor graph representation

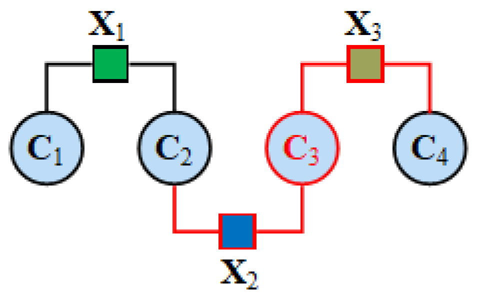


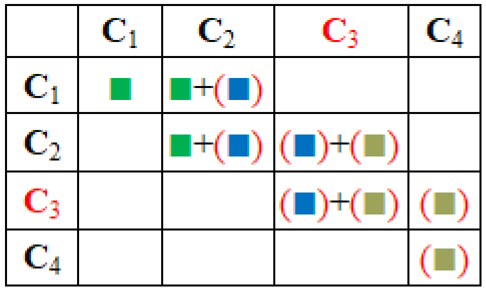




	C_1	C_2	C_3
C_1	🟩	🟩+🟦	
C_2		🟩+🟦	🟦
C_3			🟦

- Cameras have changed after iteration





	C_1	C_2	C_3	C_4
C_1	🟩	🟩+(🟦)		
C_2		🟩+(🟦)	(🟦)+(🟫)	
C_3			(🟦)+(🟫)	(🟫)
C_4				(🟫)

● point ● camera

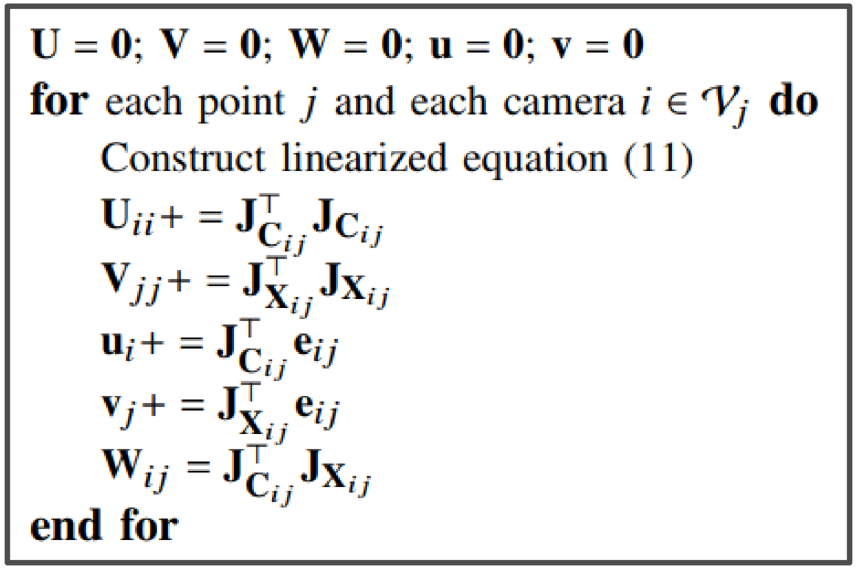
🟩 visual factor from X_1

🟦 visual factor from X_2

🟫 visual factor from X_3

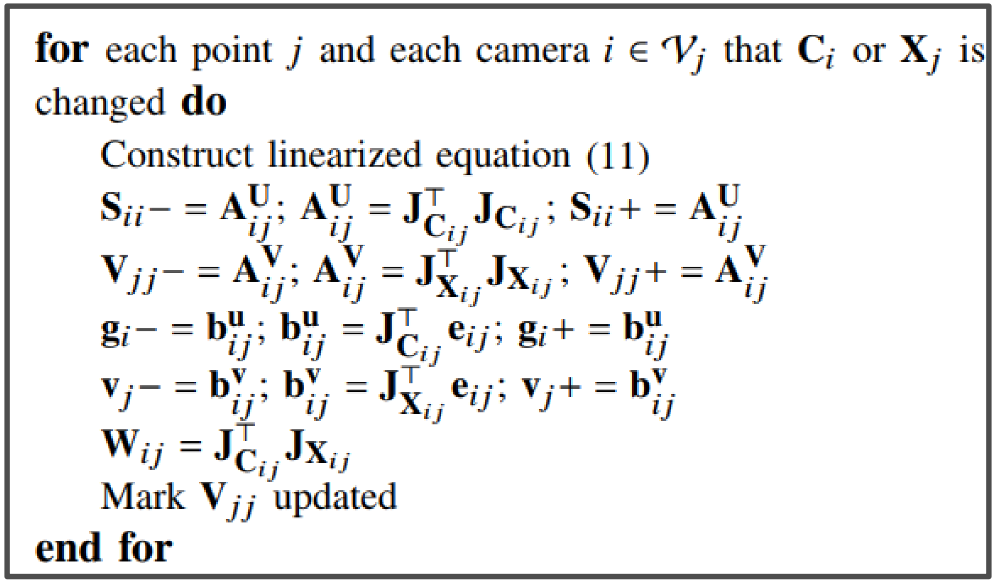
Step1: Normal Equation

- Batch BA



```
$\mathbf{U} = \mathbf{0}$; $\mathbf{V} = \mathbf{0}$; $\mathbf{W} = \mathbf{0}$; $\mathbf{u} = \mathbf{0}$; $\mathbf{v} = \mathbf{0}$
for each point $j$ and each camera $i \in \mathcal{V}_j$ do
    Construct linearized equation (11)
    $\mathbf{U}_{ii} += \mathbf{J}_{\mathbf{C}_{ij}}^\top \mathbf{J}_{\mathbf{C}_{ij}}$
    $\mathbf{V}_{jj} += \mathbf{J}_{\mathbf{X}_{ij}}^\top \mathbf{J}_{\mathbf{X}_{ij}}$
    $\mathbf{u}_i += \mathbf{J}_{\mathbf{C}_{ij}}^\top \mathbf{e}_{ij}$
    $\mathbf{v}_j += \mathbf{J}_{\mathbf{X}_{ij}}^\top \mathbf{e}_{ij}$
    $\mathbf{W}_{ij} = \mathbf{J}_{\mathbf{C}_{ij}}^\top \mathbf{J}_{\mathbf{X}_{ij}}$
end for
```

- ICE-BA



```
for each point $j$ and each camera $i \in \mathcal{V}_j$ that $\mathbf{C}_i$ or $\mathbf{X}_j$ is changed do
    Construct linearized equation (11)
    $\mathbf{S}_{ii} -= \mathbf{A}_{ij}^{\mathbf{U}}$; $\mathbf{A}_{ij}^{\mathbf{U}} = \mathbf{J}_{\mathbf{C}_{ij}}^\top \mathbf{J}_{\mathbf{C}_{ij}}$; $\mathbf{S}_{ii} += \mathbf{A}_{ij}^{\mathbf{U}}$
    $\mathbf{V}_{jj} -= \mathbf{A}_{ij}^{\mathbf{V}}$; $\mathbf{A}_{ij}^{\mathbf{V}} = \mathbf{J}_{\mathbf{X}_{ij}}^\top \mathbf{J}_{\mathbf{X}_{ij}}$; $\mathbf{V}_{jj} += \mathbf{A}_{ij}^{\mathbf{V}}$
    $\mathbf{g}_i -= \mathbf{b}_{ij}^{\mathbf{u}}$; $\mathbf{b}_{ij}^{\mathbf{u}} = \mathbf{J}_{\mathbf{C}_{ij}}^\top \mathbf{e}_{ij}$; $\mathbf{g}_i += \mathbf{b}_{ij}^{\mathbf{u}}$
    $\mathbf{v}_j -= \mathbf{b}_{ij}^{\mathbf{v}}$; $\mathbf{b}_{ij}^{\mathbf{v}} = \mathbf{J}_{\mathbf{X}_{ij}}^\top \mathbf{e}_{ij}$; $\mathbf{v}_j += \mathbf{b}_{ij}^{\mathbf{v}}$
    $\mathbf{W}_{ij} = \mathbf{J}_{\mathbf{C}_{ij}}^\top \mathbf{J}_{\mathbf{X}_{ij}}$
    Mark $\mathbf{V}_{jj}$ updated
end for
```

Step2: Schur Complement

- Batch BA

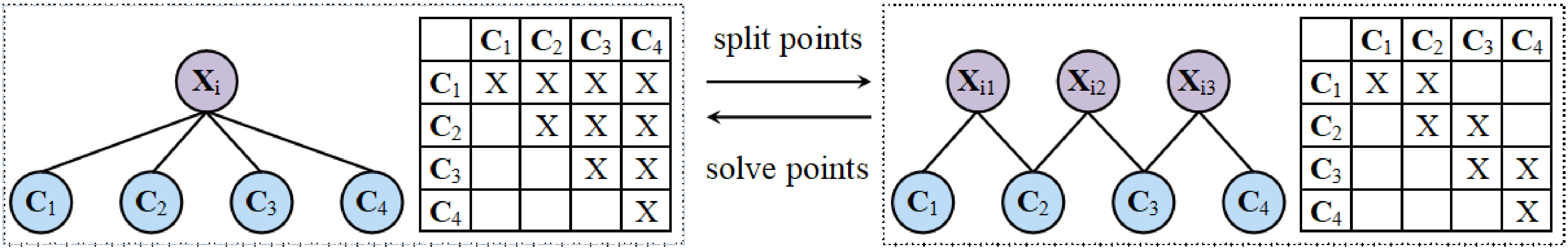
```
$\mathbf{S} = \mathbf{U}$
**for** each point $j$ and each camera pair $(i_1, i_2) \in \mathcal{V}_j \times \mathcal{V}_j$ **do**
    $\mathbf{S}_{i_1 i_2} -= \mathbf{W}_{i_1 j} \mathbf{V}_{jj}^{-1} \mathbf{W}_{i_2 j}^{\top}$
**end for**
$\mathbf{g} = \mathbf{u}$
**for** each point $j$ and each camera $i \in \mathcal{V}_j$ **do**
    $\mathbf{g}_i -= \mathbf{W}_{ij} \mathbf{V}_{jj}^{-1} \mathbf{v}_j$
**end for**
```

- ICE-BA

```
**for** each point $j$ that $\mathbf{V}_{jj}$ is updated and each camera pair $(i_1, i_2) \in \mathcal{V}_j \times \mathcal{V}_j$ **do**
    $\mathbf{S}_{i_1 i_2} += \mathbf{A}^{\mathbf{S}}_{i_1 i_2 j}$
    $\mathbf{A}^{\mathbf{S}}_{i_1 i_2 j} = \mathbf{W}_{i_1 j} \mathbf{V}_{jj}^{-1} \mathbf{W}_{i_2 j}^{\top}$
    $\mathbf{S}_{i_1 i_2} -= \mathbf{A}^{\mathbf{S}}_{i_1 i_2 j}$
**end for**
**for** each point $j$ that $\mathbf{V}_{jj}$ is updated and each camera $i \in \mathcal{V}_j$ **do**
    $\mathbf{g}_i += \mathbf{b}^{\mathbf{g}}_{ij}$; $\mathbf{b}^{\mathbf{g}}_{ij} = \mathbf{W}_{ij} \mathbf{V}_{jj}^{-1} \mathbf{v}_j$; $\mathbf{g}_i -= \mathbf{b}^{\mathbf{g}}_{ij}$
**end for**
```

Sub-track Improvement for Local BA

- In LBA, most points may be observed by most frames in the sliding window
  - Dense Schur complement
  - A large portion need to be re-computed
- Split the origin long feature track X_i into several short overlapping sub-tracks $X_{i_1}, X_{i_2}, \cdots$

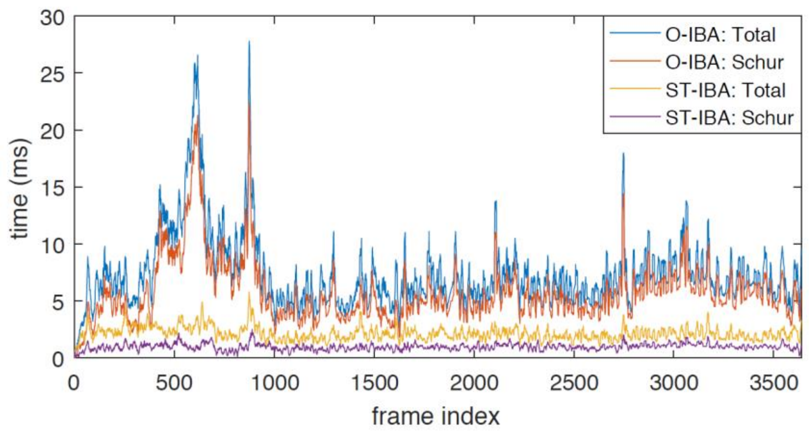


	C_1	C_2	C_3	C_4
C_1	X	X	X	X
C_2		X	X	X
C_3			X	X
C_4				X

	C_1	C_2	C_3	C_4
C_1	X	X		
C_2		X	X	
C_3			X	X
C_4				X

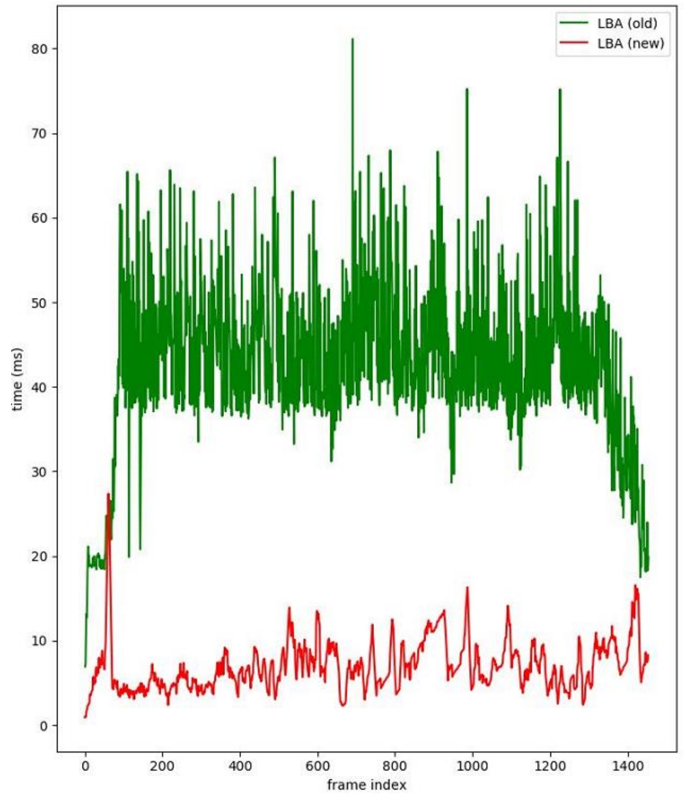
Sub-track Improvement for Local BA

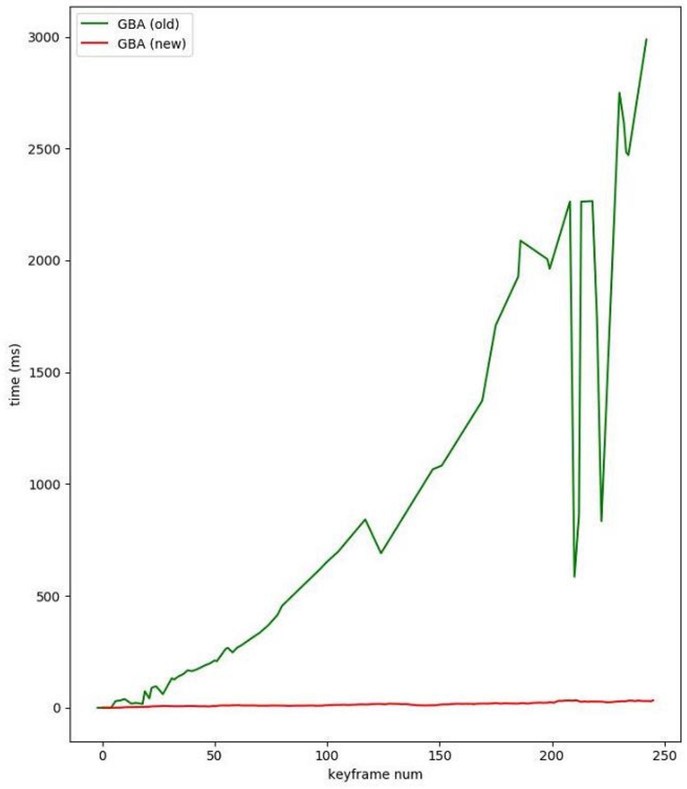
- In LBA, most points may be observed by most frames in the sliding window
  - Dense Schur complement
  - A large portion need to be re-computed
- Split the origin long feature track X_i into several short overlapping sub-tracks $X_{i_1}, X_{i_2}, \cdots$



Efficiency Comparison

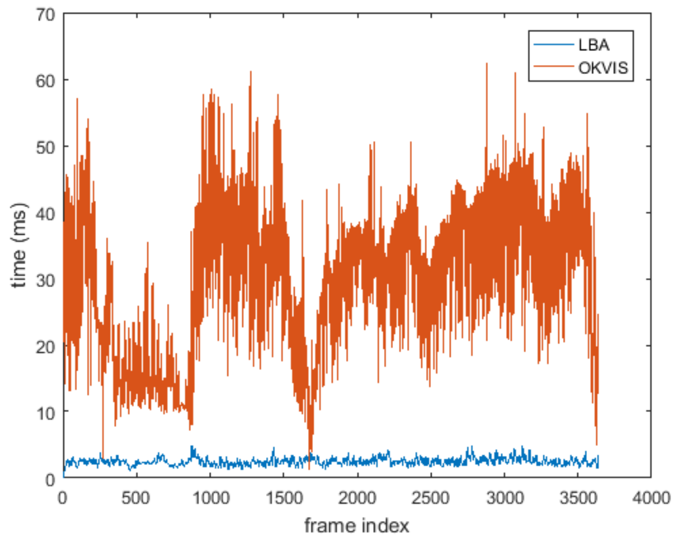
- Local BA (LBA)
  - ICE-BA (50 frames)
  - Ceres (10 frames)
- Global BA (GBA)
  - ICE-BA: $O(1)$
  - Ceres: $O(n^2)$

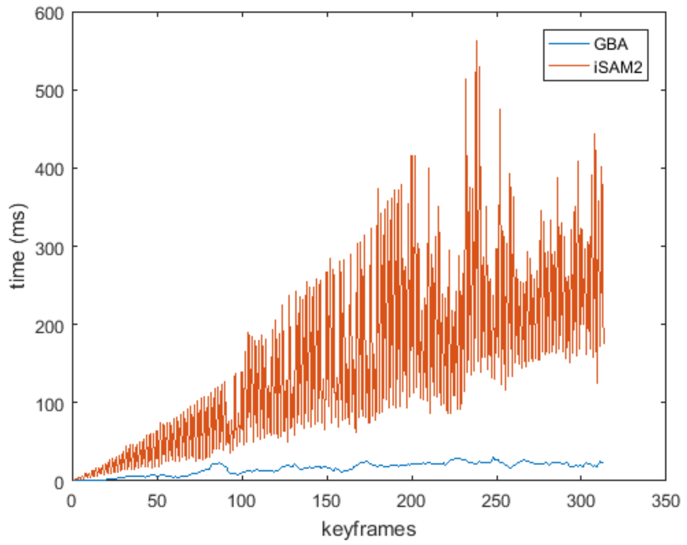




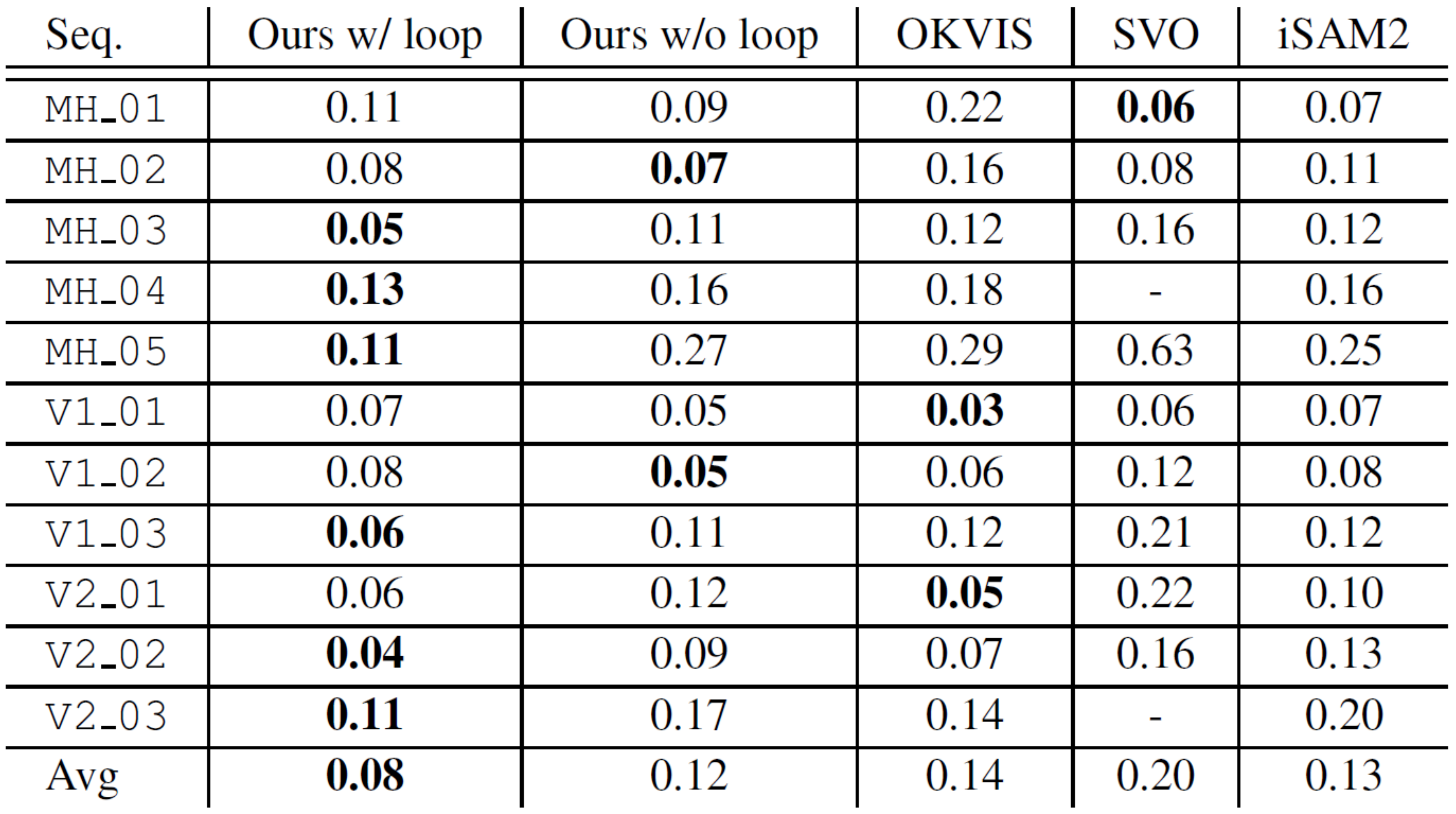
Efficiency Comparison

- Local BA (LBA)
  - ICE-BA (50 frames)
  - OKVIS (8 frames)
- Global BA (GBA)
  - ICE-BA: steady and smooth
  - iSAM2: steep and peaks





Accuracy Comparison



Seq.	Ours w/ loop	Ours w/o loop	OKVIS	SVO	iSAM2
MH_01	0.11	0.09	0.22	**0.06**	0.07
MH_02	0.08	**0.07**	0.16	0.08	0.11
MH_03	**0.05**	0.11	0.12	0.16	0.12
MH_04	**0.13**	0.16	0.18	-	0.16
MH_05	**0.11**	0.27	0.29	0.63	0.25
V1_01	0.07	0.05	**0.03**	0.06	0.07
V1_02	0.08	**0.05**	0.06	0.12	0.08
V1_03	**0.06**	0.11	0.12	0.21	0.12
V2_01	0.06	0.12	**0.05**	0.22	0.10
V2_02	**0.04**	0.09	0.07	0.16	0.13
V2_03	**0.11**	0.17	0.14	-	0.20
Avg	**0.08**	0.12	0.14	0.20	0.13

Open-source Solver & BA

- Bundler: http://www.cs.cornell.edu/~snavely/bundler
- g2o: https://github.com/RainerKuemmerle/g2o
- Ceres Solver: http://ceres-solver.org
- SegmentBA: https://github.com/zju3dv/SegmentBA
- iSAM2: https://bitbucket.org/gtborg/gtsam
- ICE-BA: https://github.com/baidu/ICE-BA
- SLAM++: https://sourceforge.net/p/slam-plus-plus/wiki/Home/

Questions