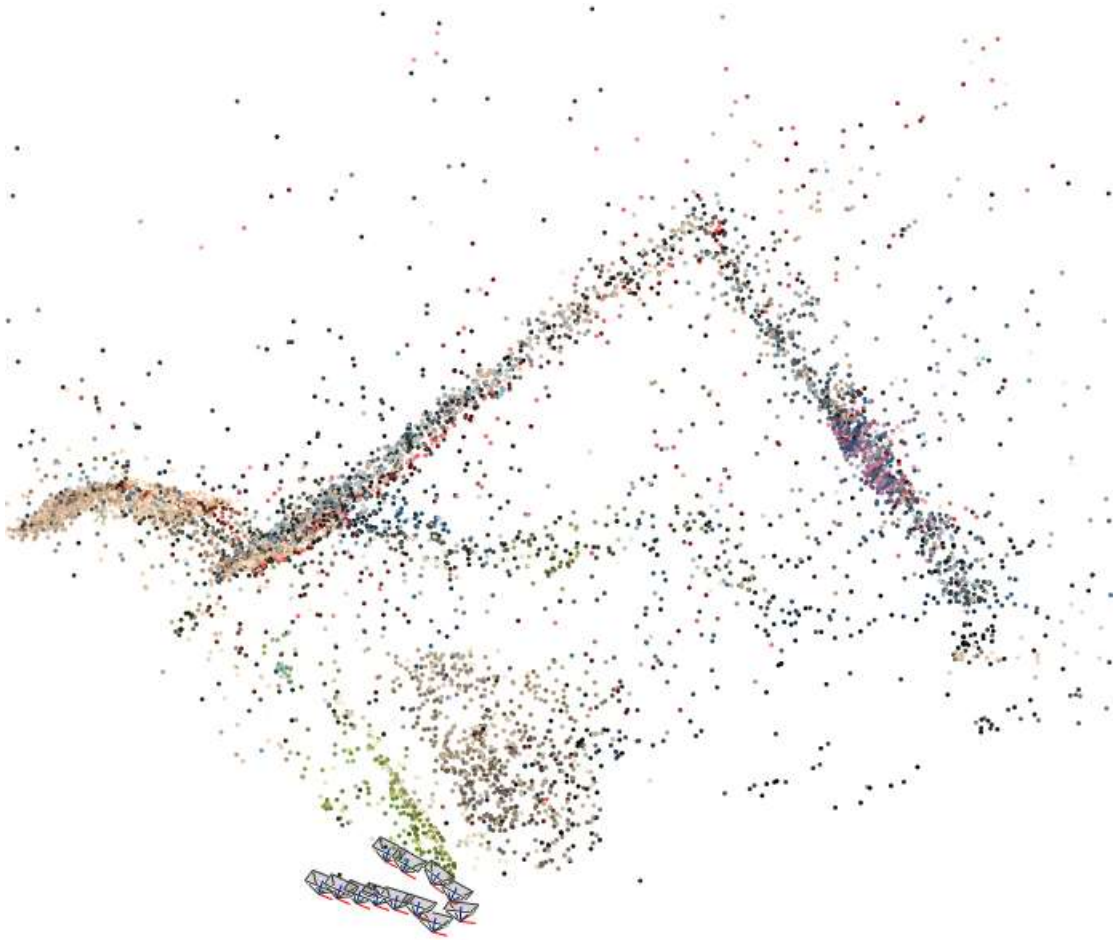


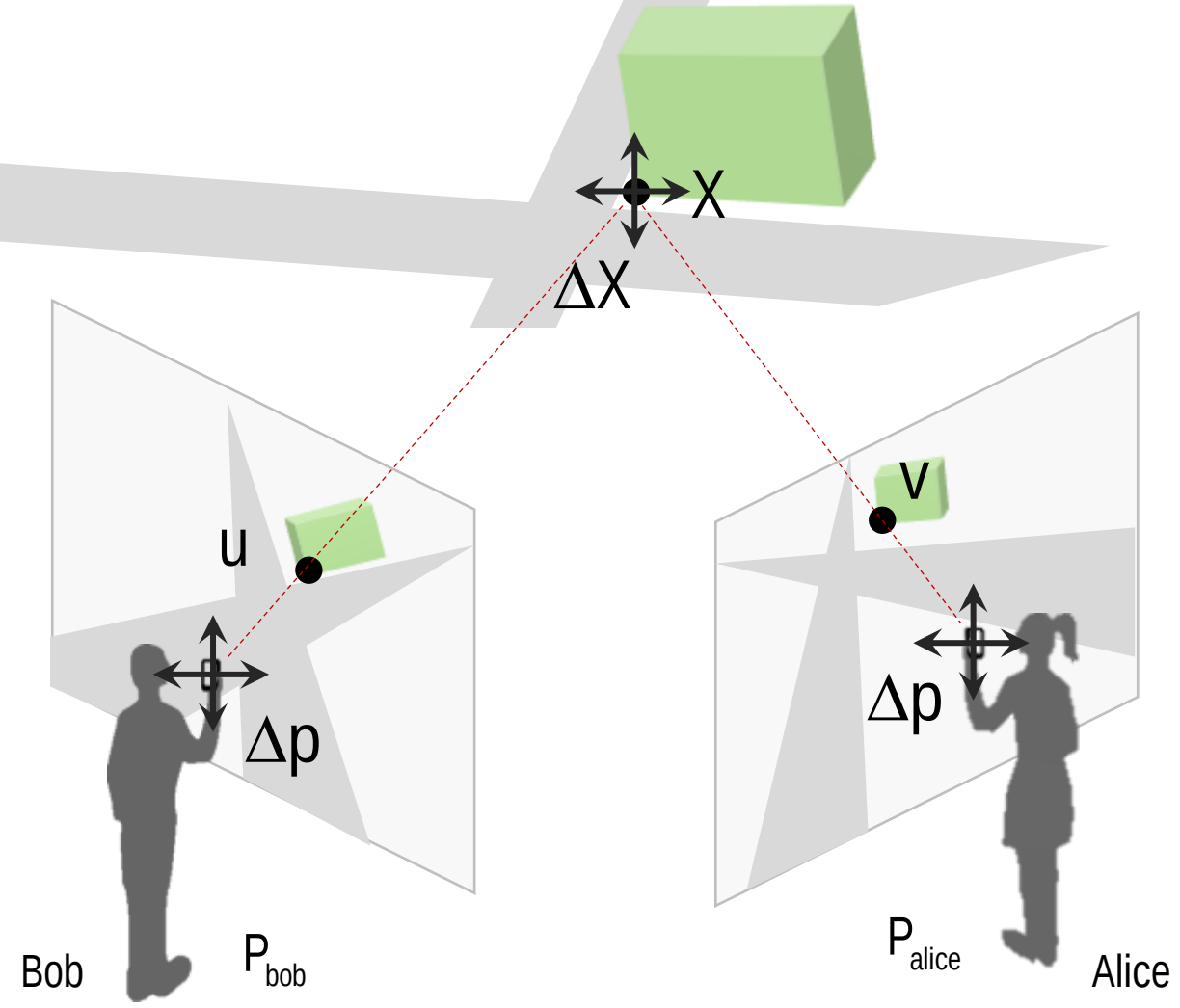
Bundle Adjustment

Black: given variables
Red: unknowns

Camera & Point Jacobian

$$E_{geom} = \left\| \hat{u} - u \right\|^2$$

$$= \left(\frac{P_1 X}{P_3 X} - x \right)^2 + \left(\frac{P_2 X}{P_3 X} - y \right)^2$$

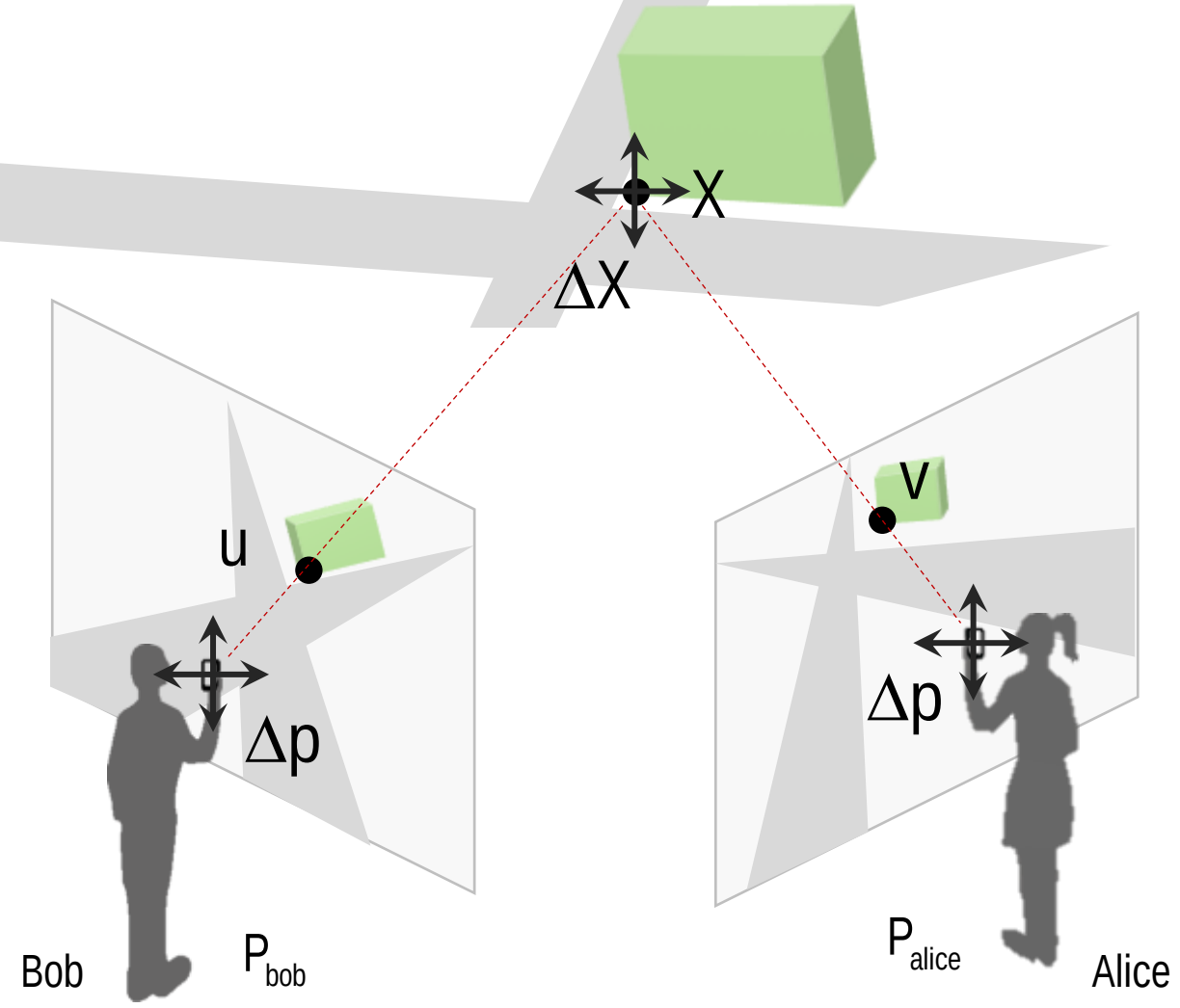


Black: given variables
Red: unknowns

Camera & Point Jacobian

$$E_{geom} = \left\| \hat{u} - u \right\|^2$$

$$= \left(\frac{P_1 X}{P_3 X} - x \right)^2 + \left(\frac{P_2 X}{P_3 X} - y \right)^2$$



$$f(p, X) = \begin{bmatrix} \frac{u}{w} \\ \frac{v}{w} \end{bmatrix}$$

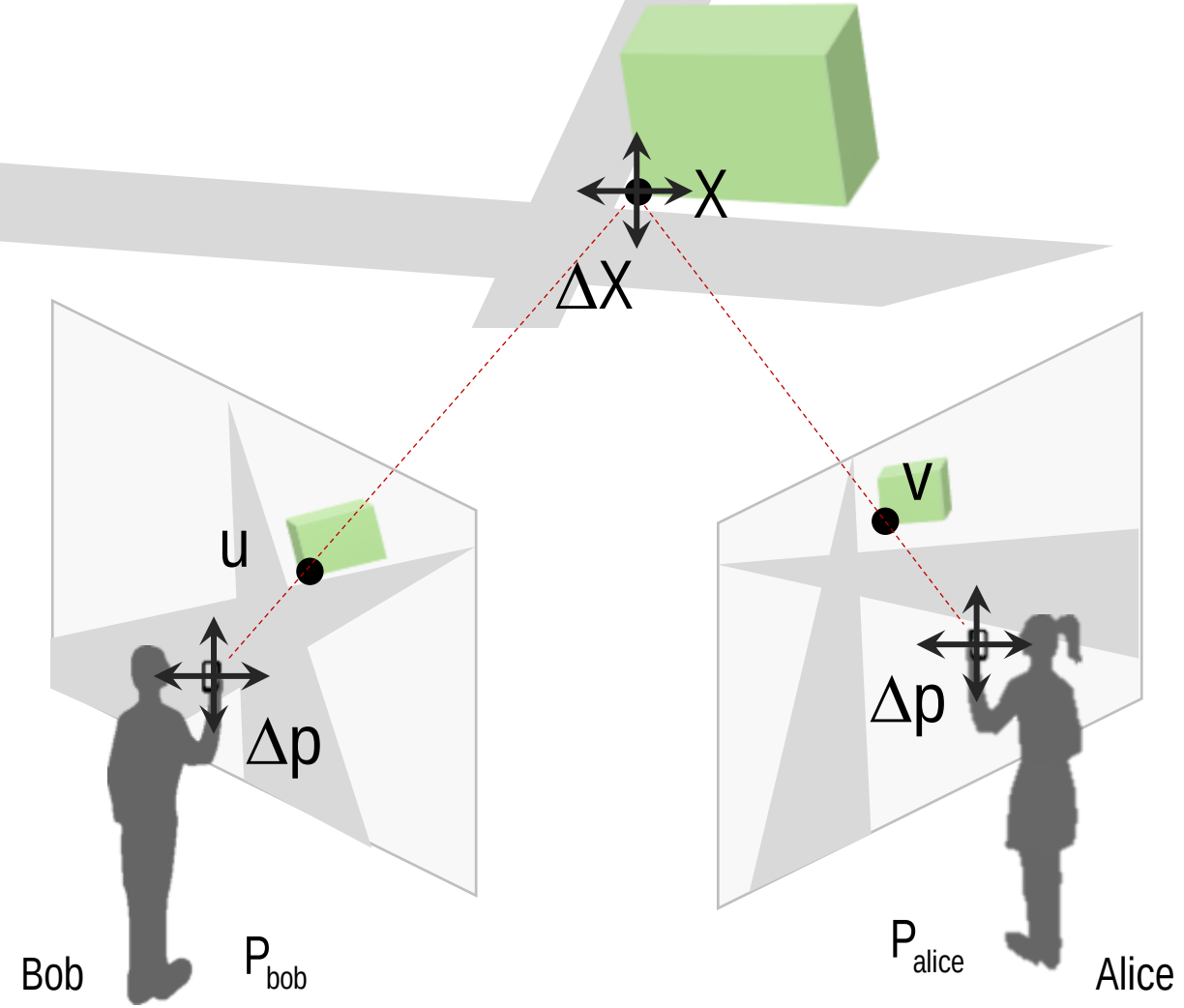
Black: given variables
Red: unknowns

Camera & Point Jacobian

$$E_{geom} = \left\| \hat{u} - u \right\|^2$$

$$= \left(\frac{P_1 X}{P_3 X} - x \right)^2 + \left(\frac{P_2 X}{P_3 X} - y \right)^2$$

$$f(p, X) = \begin{bmatrix} \frac{u}{w} \\ \frac{v}{w} \end{bmatrix} \longrightarrow \frac{\partial f(p, X)}{\partial p} = \frac{\partial}{\partial p} \begin{bmatrix} \frac{u}{w} \\ \frac{v}{w} \end{bmatrix} = \begin{bmatrix} \frac{w \frac{\partial u}{\partial p} - u \frac{\partial w}{\partial p}}{w^2} \\ \frac{v \frac{\partial u}{\partial p} - v \frac{\partial w}{\partial p}}{w^2} \end{bmatrix}$$

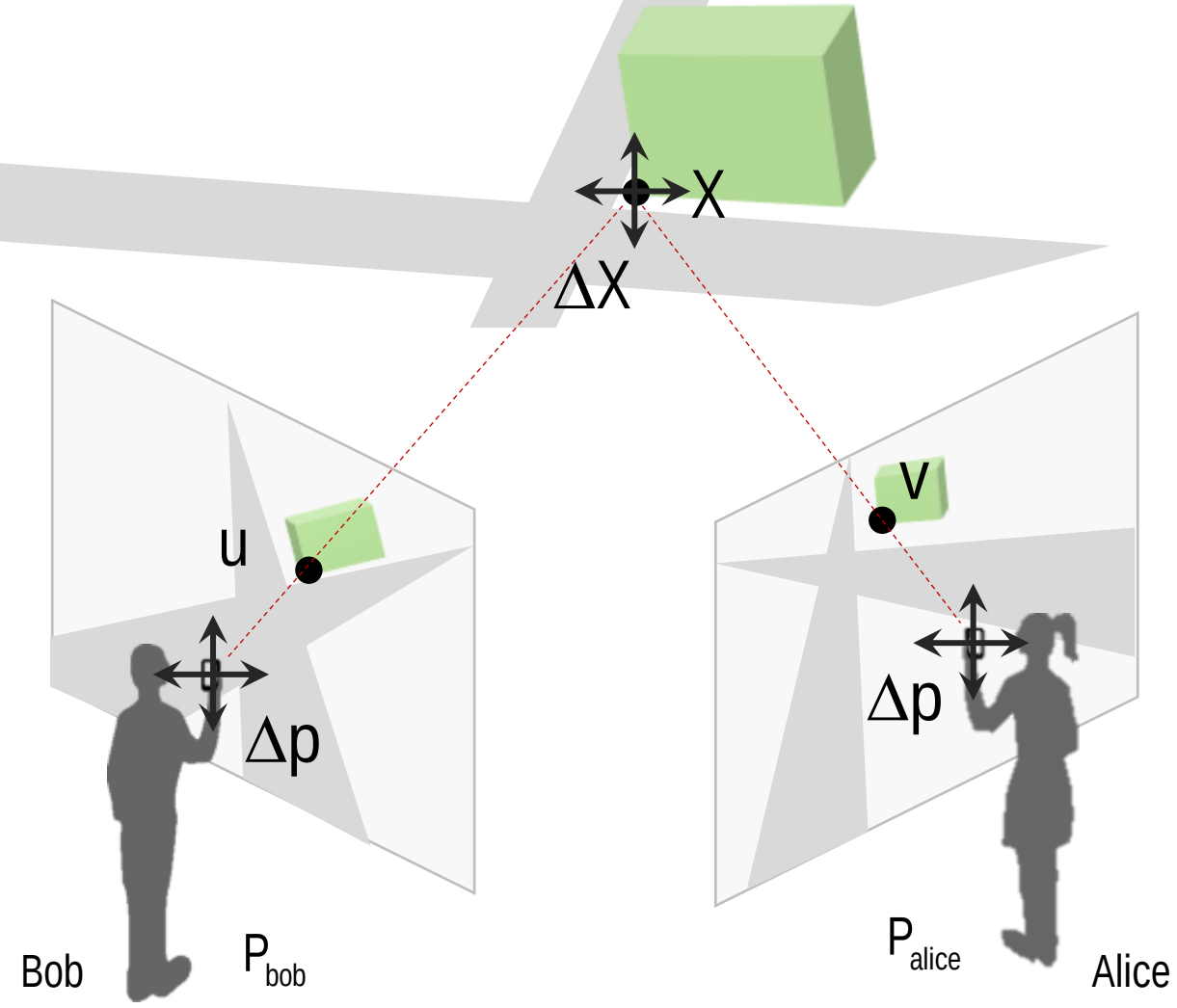


Black: given variables
Red: unknowns

Camera & Point Jacobian

$$E_{geom} = \left\| \hat{u} - u \right\|^2$$

$$= \left(\frac{P_1 X}{P_3 X} - x \right)^2 + \left(\frac{P_2 X}{P_3 X} - y \right)^2$$



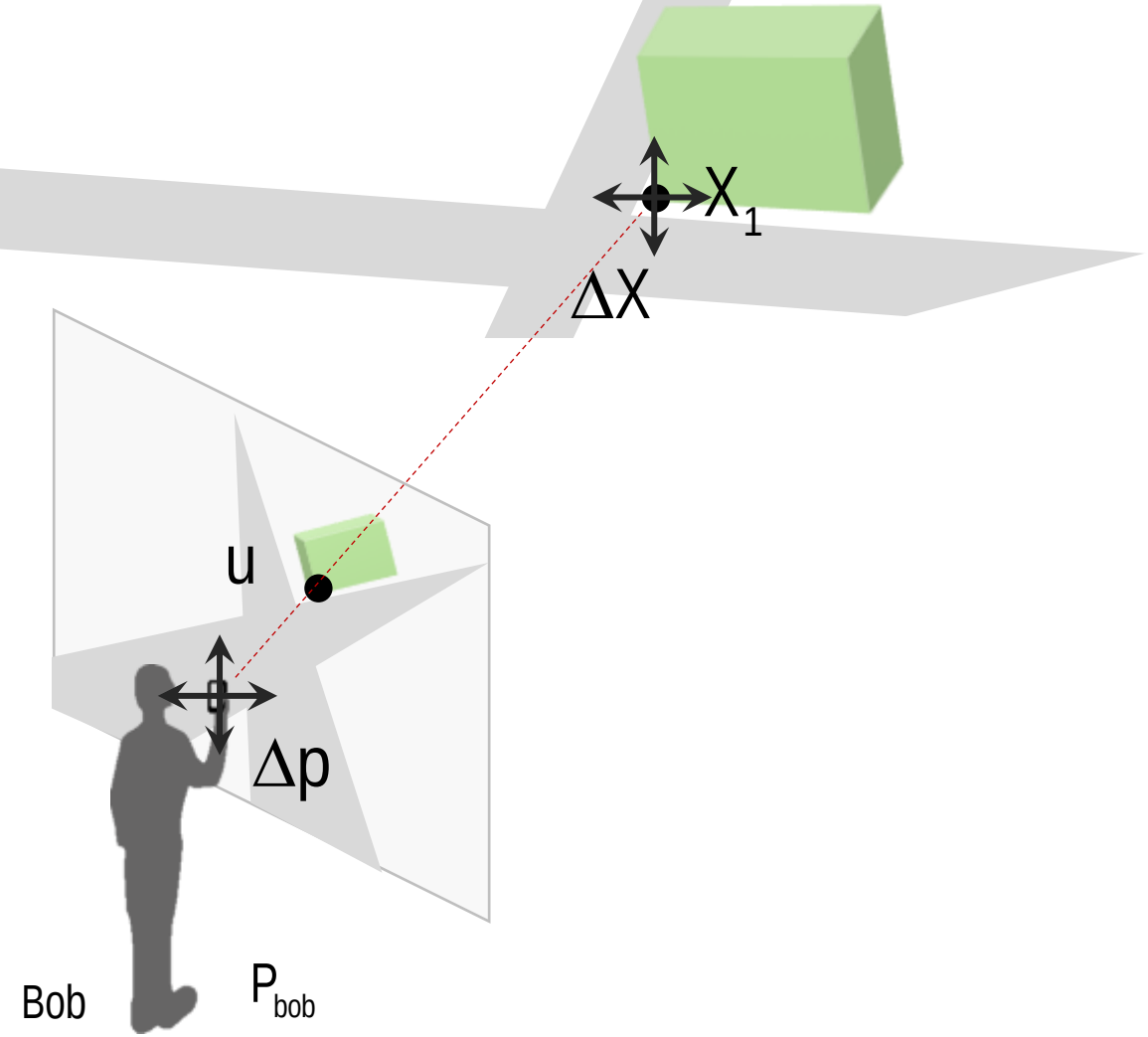
$$f(p, X) = \begin{bmatrix} \frac{u}{w} \\ \frac{v}{w} \end{bmatrix} \longrightarrow \frac{\partial f(p, X)}{\partial p} = \frac{\partial}{\partial p} \begin{bmatrix} \frac{u}{w} \\ \frac{v}{w} \end{bmatrix} = \begin{bmatrix} \frac{w \frac{\partial u}{\partial p} - u \frac{\partial w}{\partial p}}{w^2} \\ \frac{v \frac{\partial u}{\partial p} - v \frac{\partial w}{\partial p}}{w^2} \end{bmatrix}$$

$$\longrightarrow \frac{\partial f(p, X)}{\partial X} = \frac{\partial}{\partial X} \begin{bmatrix} \frac{u}{w} \\ \frac{v}{w} \end{bmatrix} = \begin{bmatrix} \frac{w \frac{\partial u}{\partial X} - u \frac{\partial w}{\partial X}}{w^2} \\ \frac{v \frac{\partial u}{\partial X} - v \frac{\partial w}{\partial X}}{w^2} \end{bmatrix}$$

Black: given variables
Red: unknowns

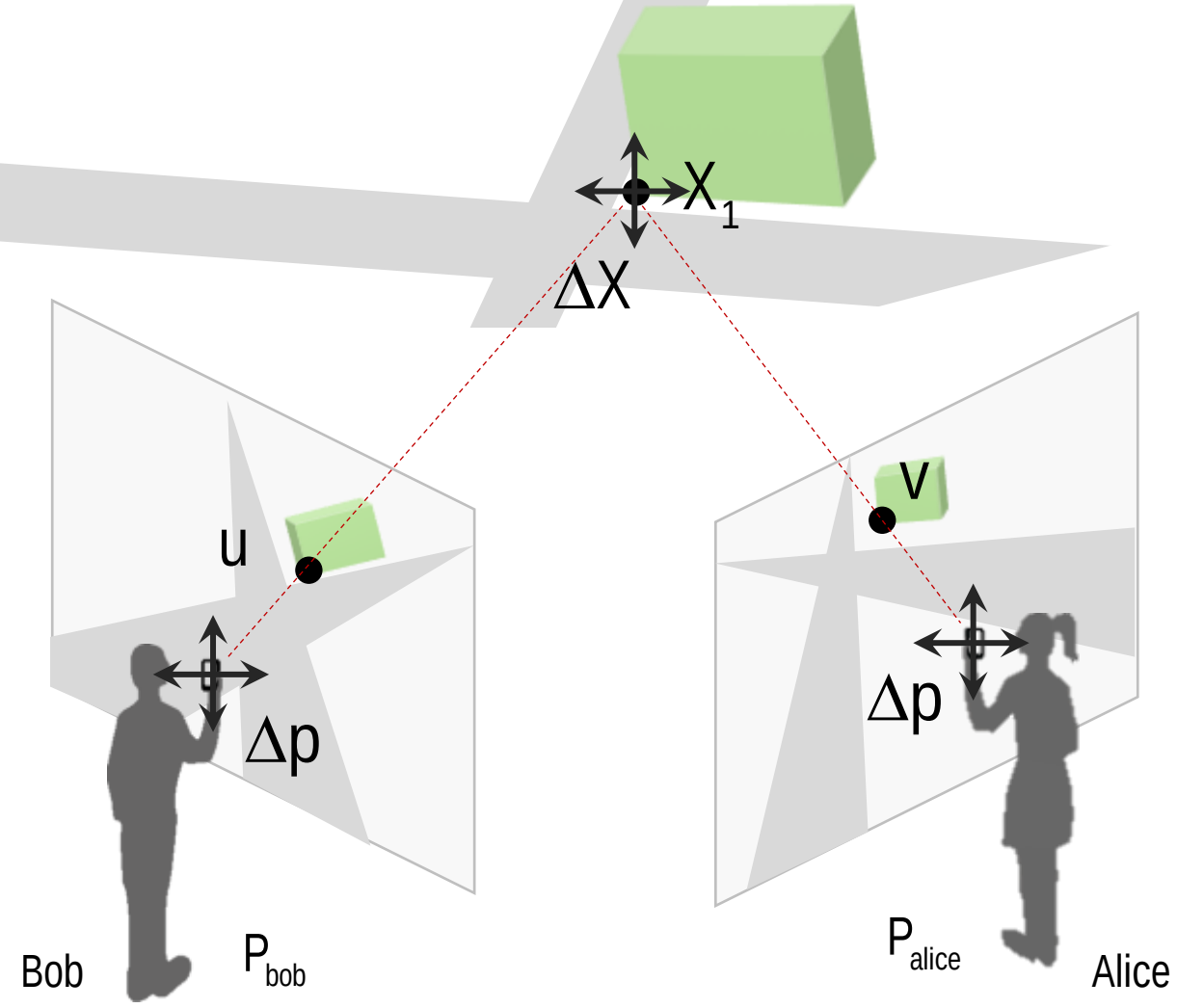
Camera & Point Jacobian

$$J_{ij} = \left[\frac{\partial f(p_j, X_i)}{\partial p_j} \quad \frac{\partial f(p_j, X_i)}{\partial X_i} \right] = \left[J_{pij} \quad J_{Xij} \right]$$



Black: given variables
Red: unknowns

Camera & Point Jacobian

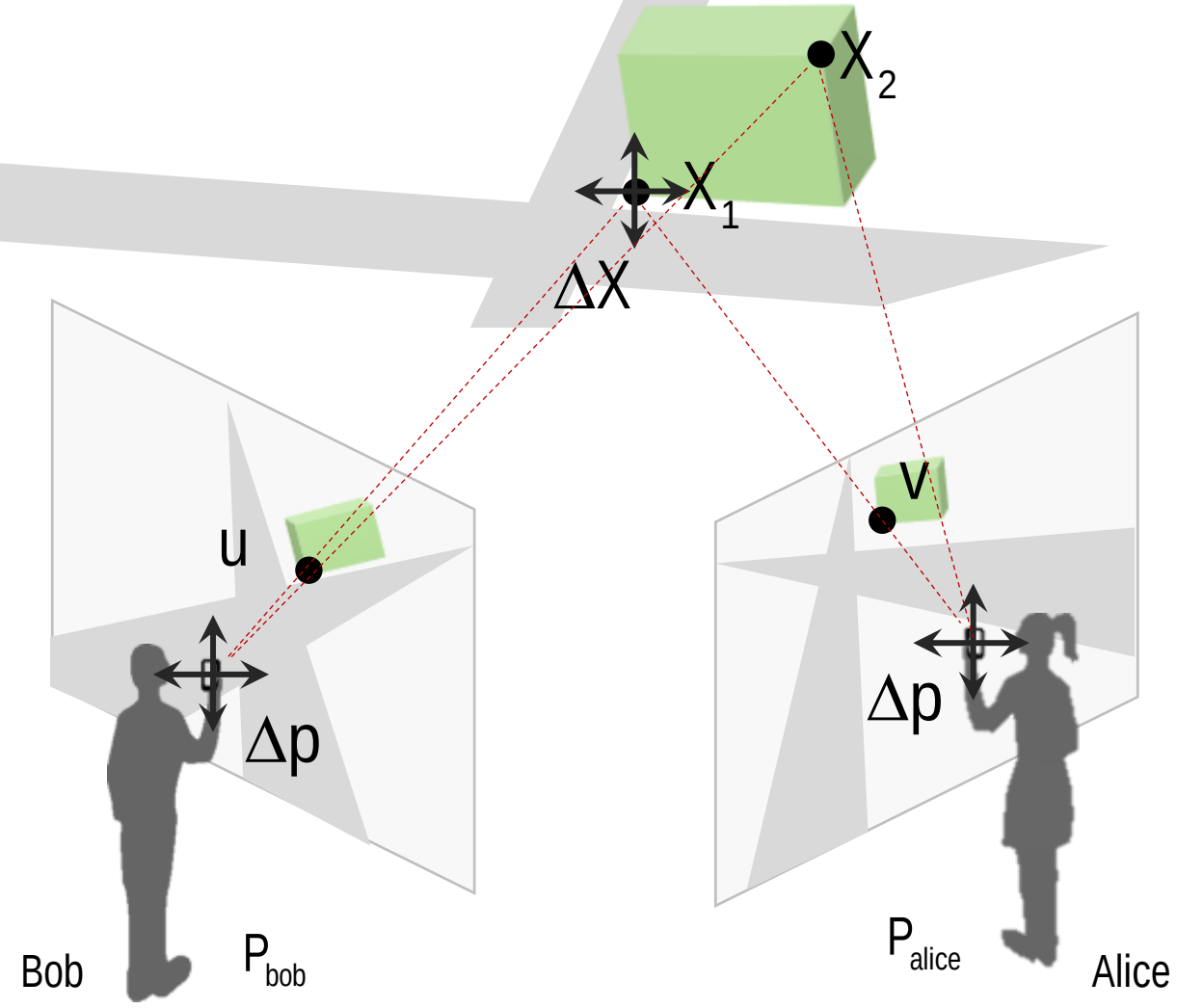


$$J_{ij} = \left[\frac{\partial f(p_j, X_i)}{\partial p_j} \quad \frac{\partial f(p_j, X_i)}{\partial X_i} \right] = \left[J_{pij} \quad J_{Xij} \right]$$

$$J = \begin{bmatrix} J_{p1,bob} & 0_{2\times7} & J_{X1,bob} \\ 0_{2\times7} & J_{p1,alice} & J_{X1,alice} \end{bmatrix}$$

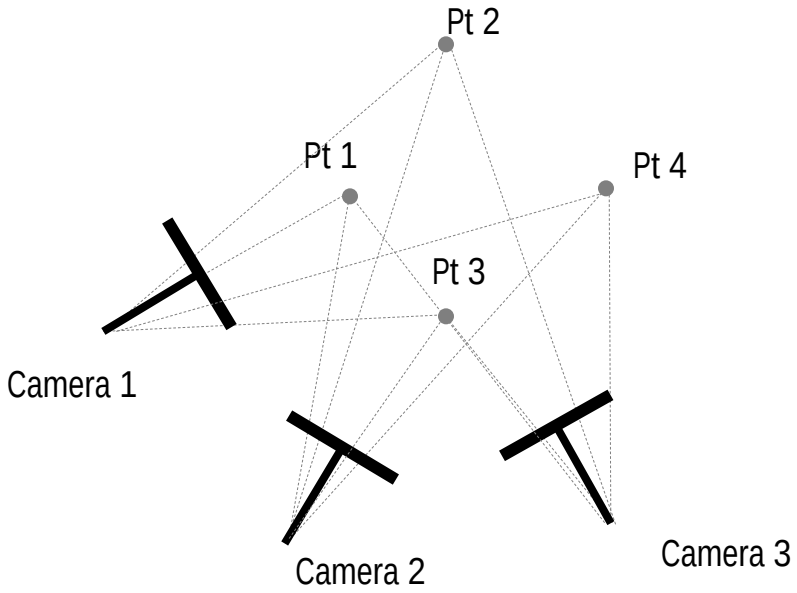
Black: given variables
Red: unknowns

Camera & Point Jacobian

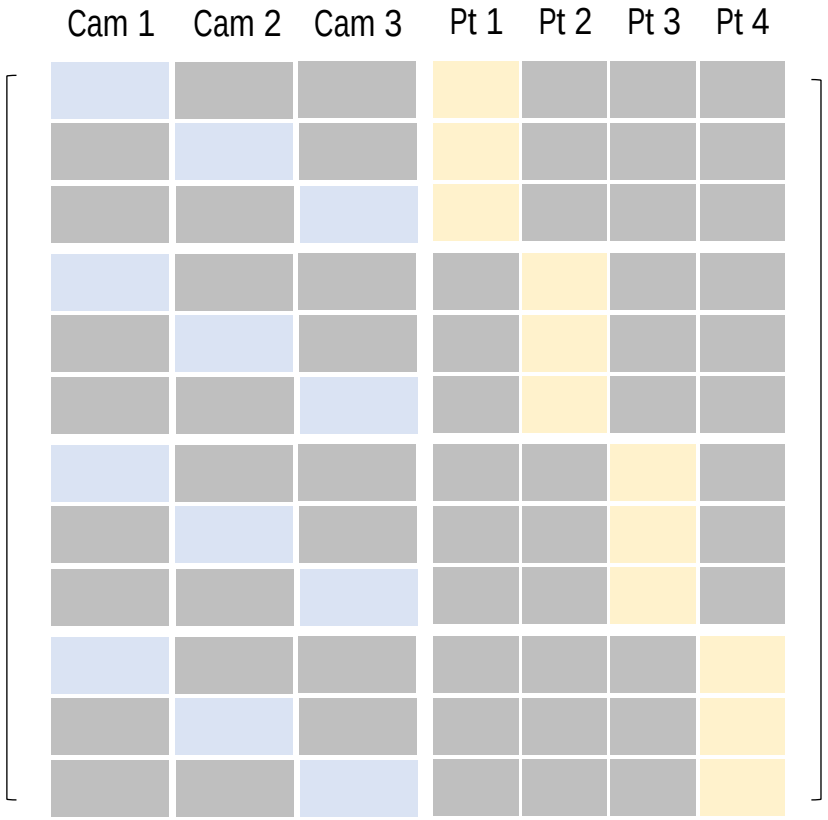


$$J_{ij} = \left[\frac{\partial f(p_j, X_i)}{\partial p_j} \quad \frac{\partial f(p_j, X_i)}{\partial X_i} \right] = \left[J_{pij} \quad J_{Xij} \right]$$

$$J = \begin{bmatrix} J_{p1,bob} & 0_{2\times7} & J_{X1,bob} & 0_{2\times3} \\ 0_{2\times7} & J_{p1,alice} & J_{X1,alice} & 0_{2\times3} \\ J_{p2,bob} & 0_{2\times7} & 0_{2\times3} & J_{X2,bob} \\ 0_{2\times7} & J_{p2,alice} & 0_{2\times3} & J_{X2,alice} \end{bmatrix}$$

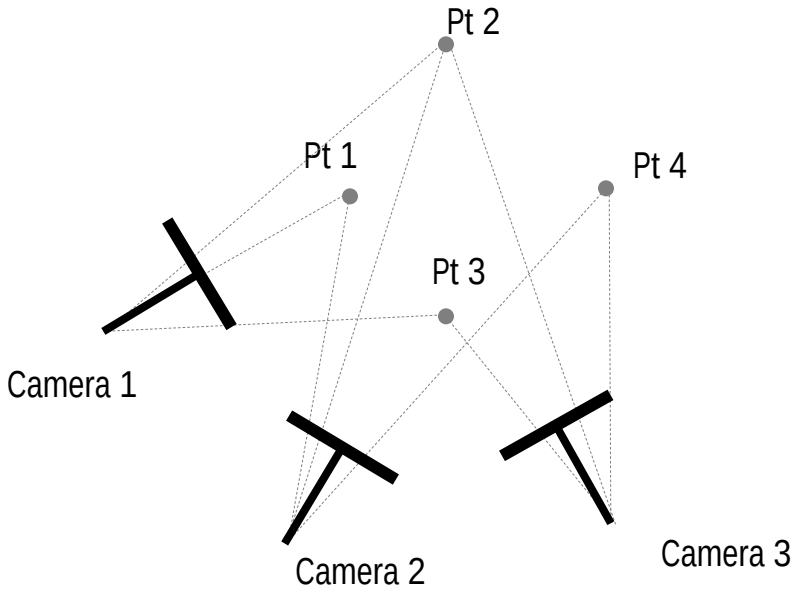


J =

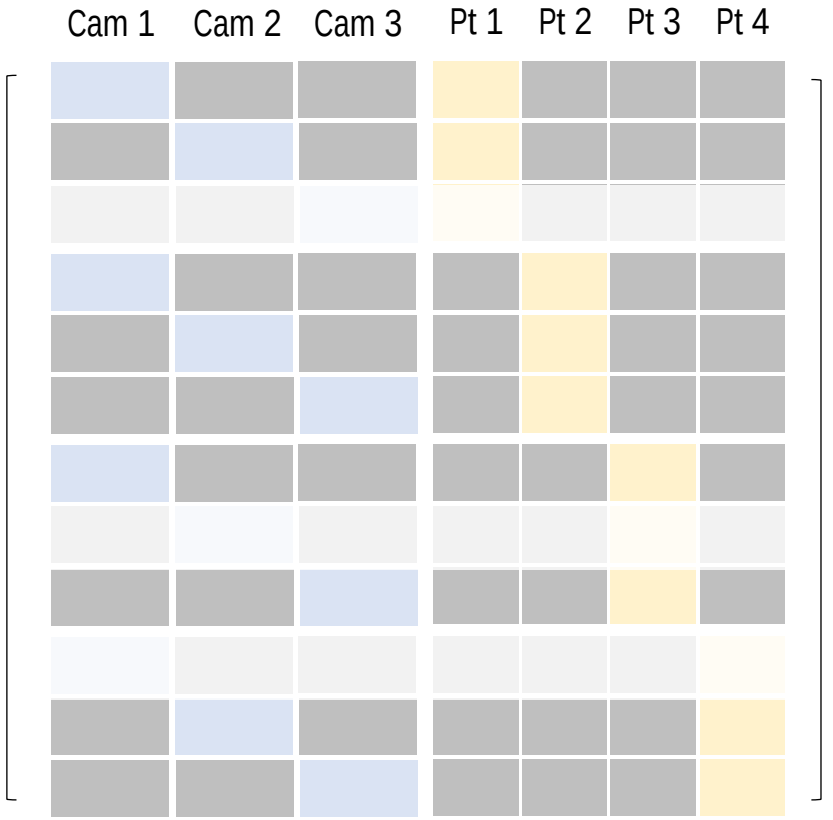


# of unknowns: 3x7+4x3

# of projections: 3x4



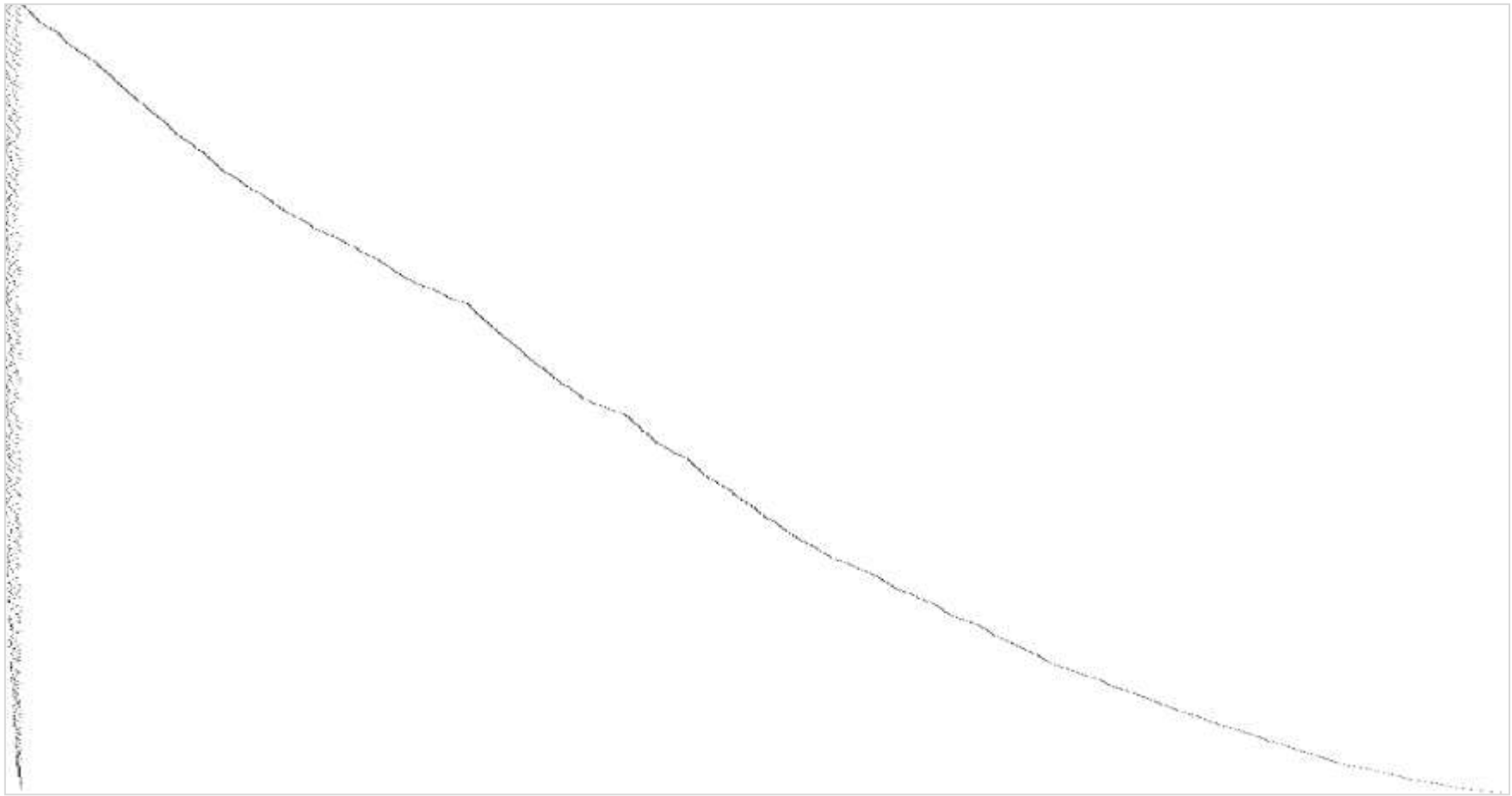
J =



# of unknowns: 3x7+4x3

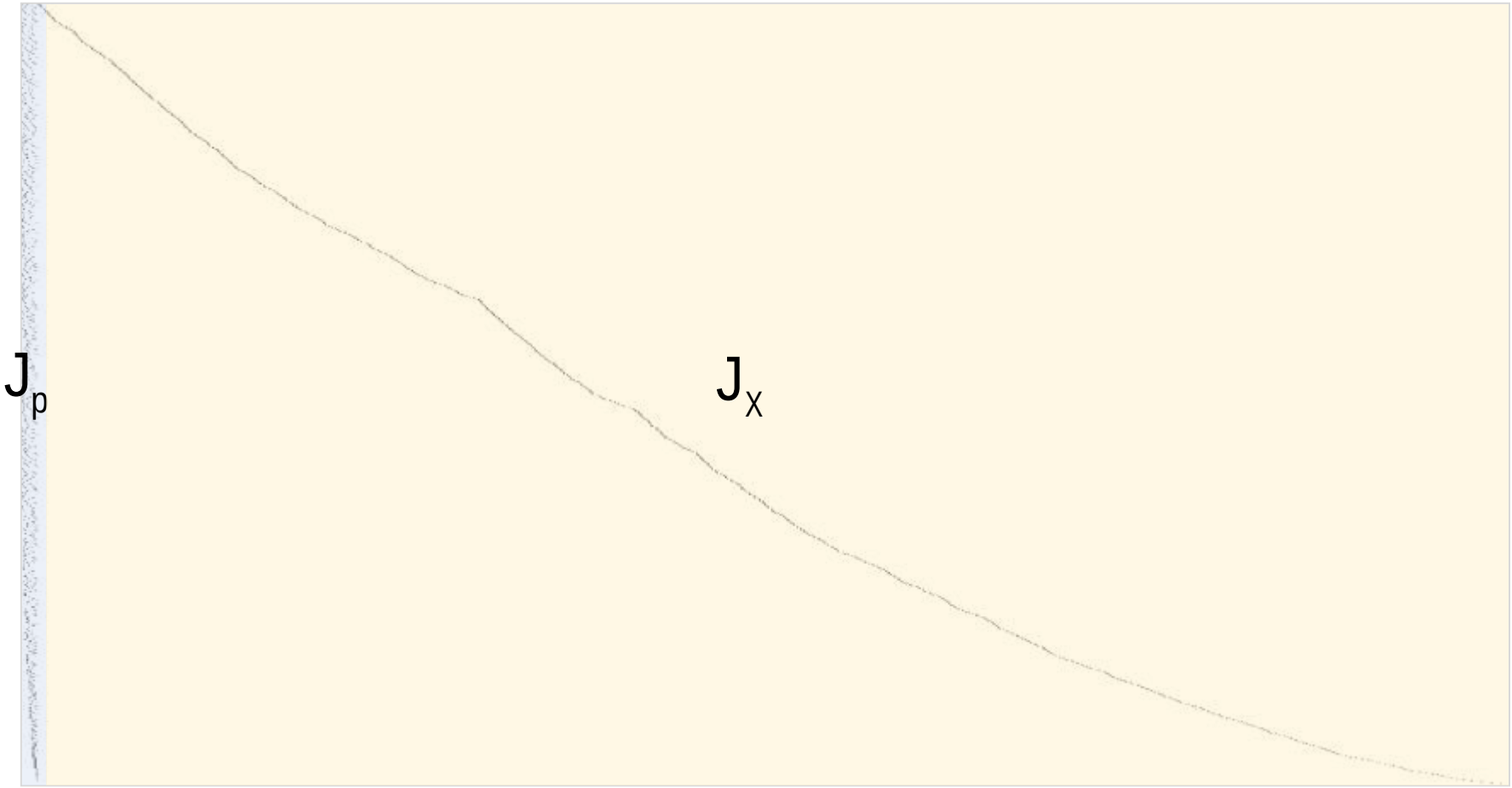
# of projections: 9 (not all points are visible from cameras)

$$J =$$

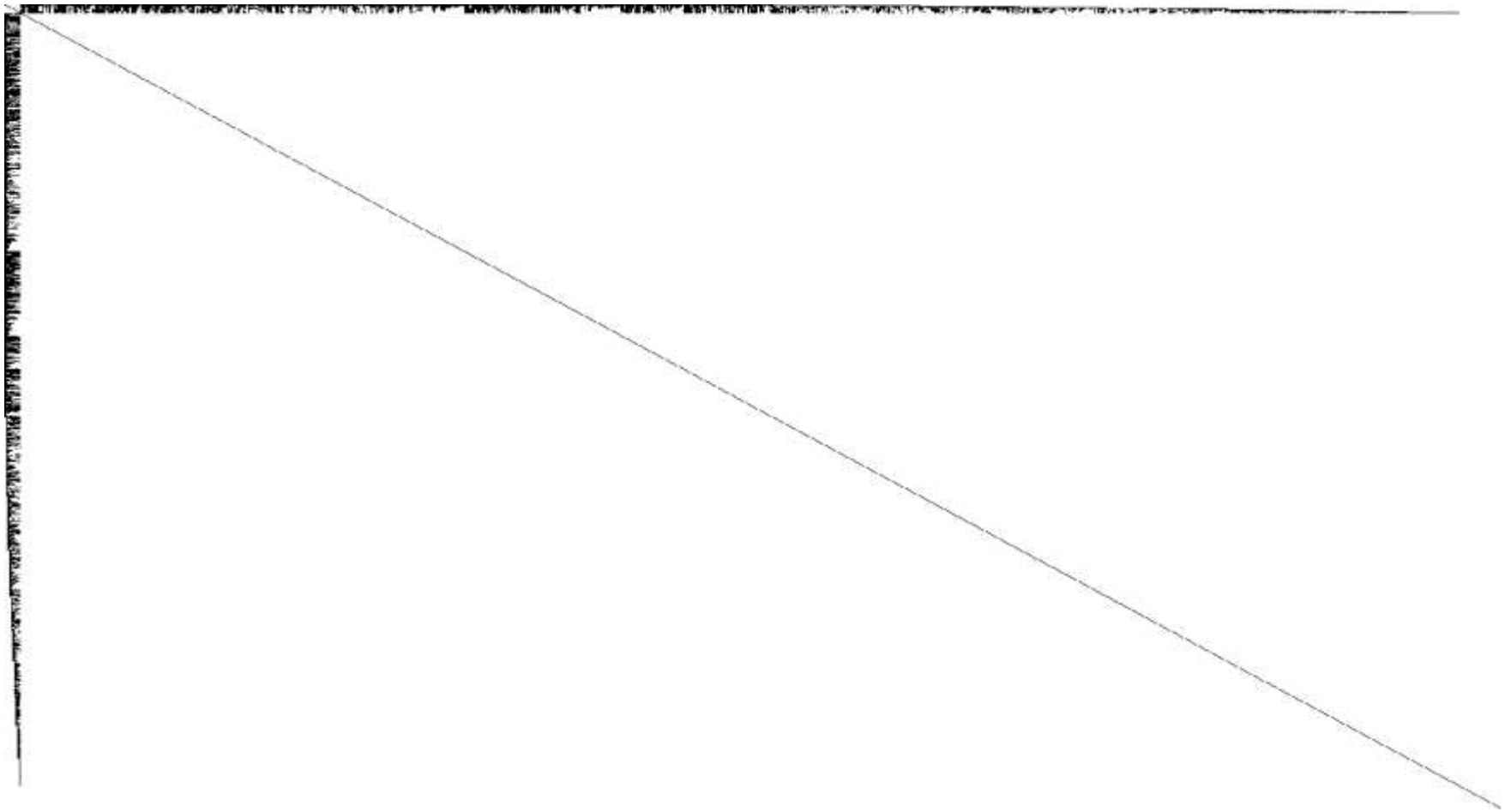


Camera 3D point

$$J = \begin{bmatrix} J_p & J_X \end{bmatrix}$$



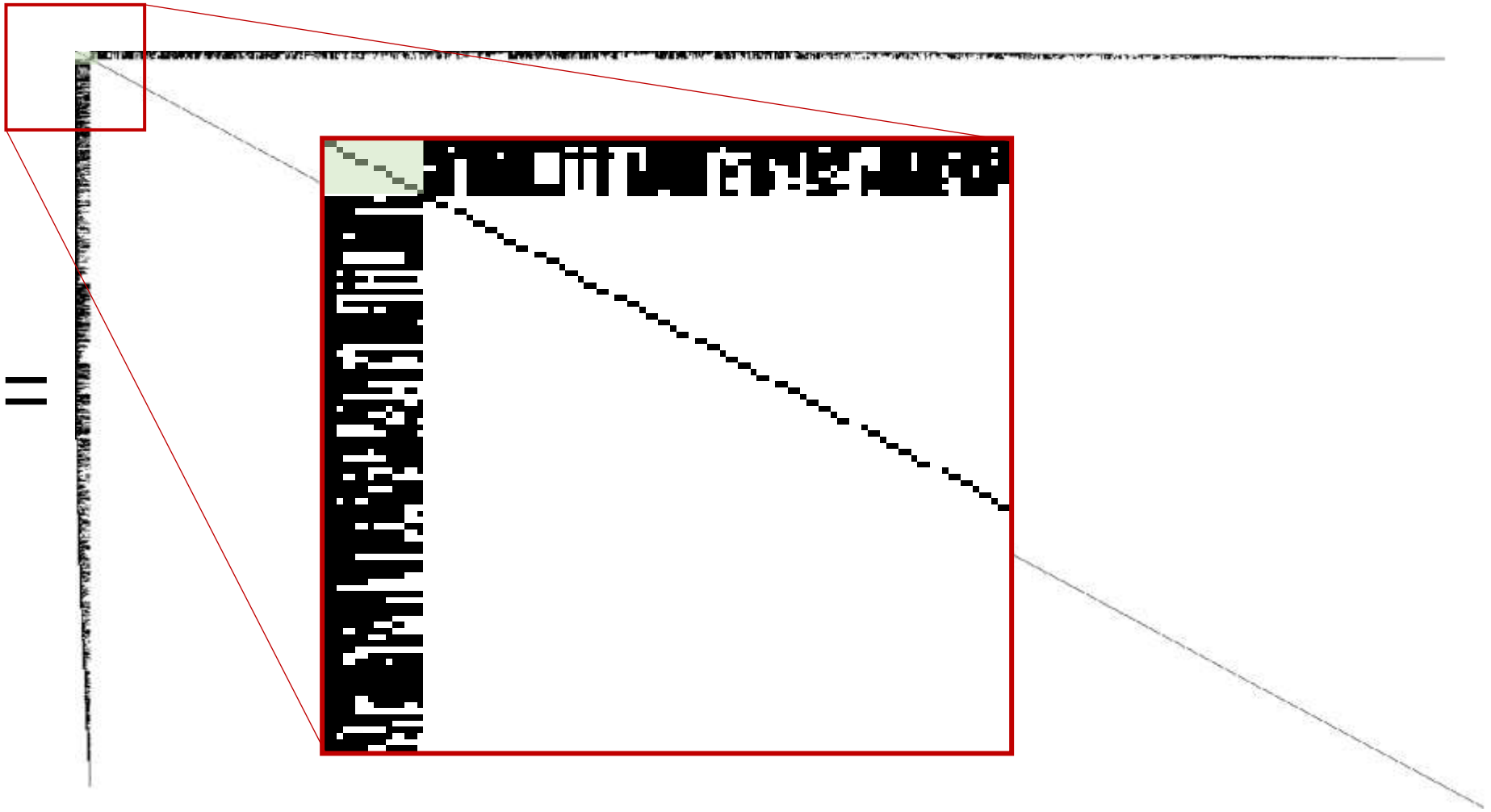
$$J^T J = \quad$$



$$J = \begin{bmatrix} J_p & J_X \end{bmatrix}$$

$$J^T J = \begin{bmatrix} J_p^T J_p & J_p^T J_X \\ J_X^T J_p & J_X^T J_X \end{bmatrix}$$

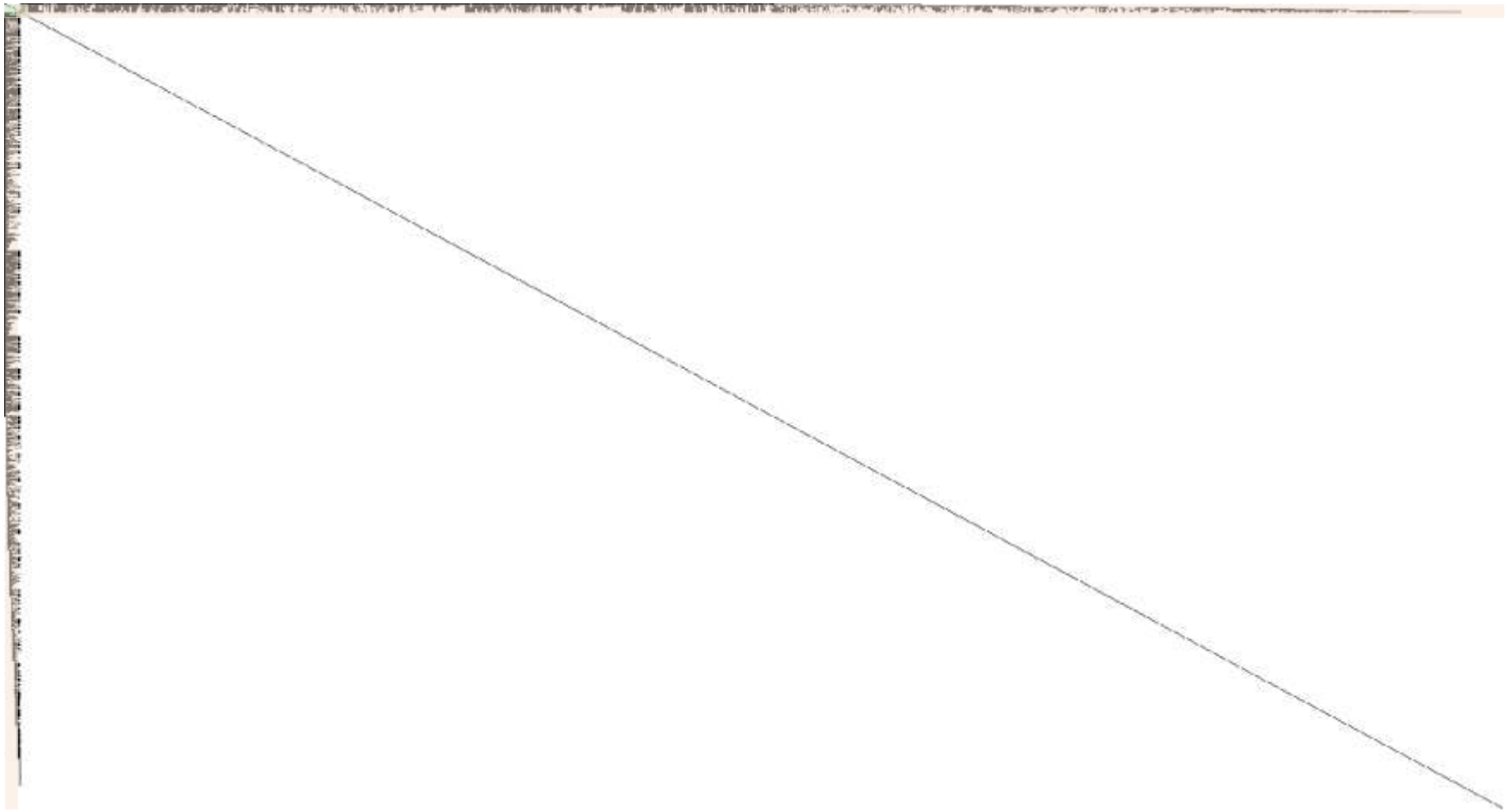
$J^T J =$



$$J = \begin{bmatrix} J_p & J_X \end{bmatrix}$$

$$J^T J = \begin{bmatrix} J_p^T J_p & J_p^T J_X \\ J_X^T J_p & J_X^T J_X \end{bmatrix} = \begin{bmatrix} A & B \\ B^T & D \end{bmatrix}$$

$$J^T J = \quad$$

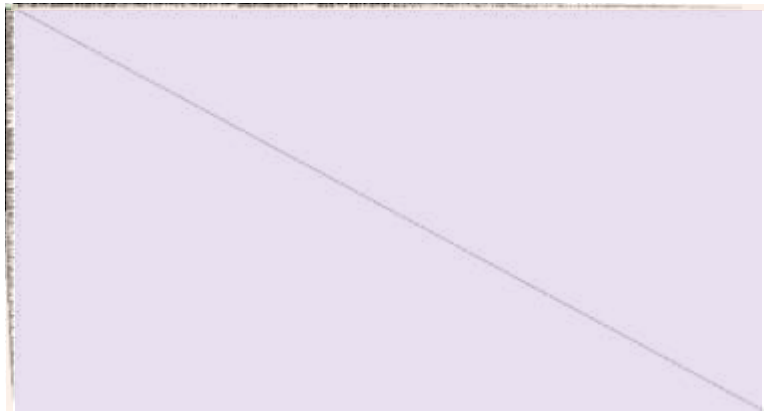


$$J = \begin{bmatrix} J_p & J_X \end{bmatrix}$$

$$J^T J = \begin{bmatrix} J_p^T J_p & J_p^T J_X \\ J_X^T J_p & J_X^T J_X \end{bmatrix} = \begin{bmatrix} A & B \\ B^T & D \end{bmatrix}$$

Normal equation:

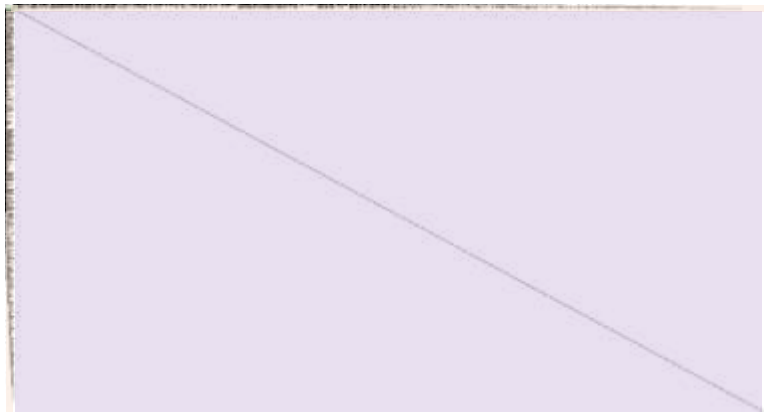
$$J^T J \begin{bmatrix} \Delta p \\ \Delta X \end{bmatrix} = J^T \left(b - f(X) \right)$$

$J^T J =$ 

$$J^T J = \begin{bmatrix} J_p^T J_p & J_p^T J_X \\ J_X^T J_p & J_X^T J_X \end{bmatrix} = \begin{bmatrix} A & B \\ B^T & D \end{bmatrix}$$

Normal equation:

$$J^T J \begin{bmatrix} \Delta p \\ \Delta X \end{bmatrix} = J^T \left(b - f(X) \right)$$

$$J^T J = \text{  }$$

$$\rightarrow \begin{bmatrix} J_p^T J_p & J_p^T J_X \\ J_X^T J_p & J_X^T J_X \end{bmatrix} \begin{bmatrix} \Delta p \\ \Delta X \end{bmatrix} = \begin{bmatrix} J_p^T \\ J_X^T \end{bmatrix} \left(b - f(X) \right) = \begin{bmatrix} e_p \\ e_X \end{bmatrix}$$

$$J^T J = \begin{bmatrix} J_p^T J_p & J_p^T J_X \\ J_X^T J_p & J_X^T J_X \end{bmatrix} = \begin{bmatrix} A & B \\ B^T & D \end{bmatrix}$$

Normal equation:

$$J^T J \begin{bmatrix} \Delta p \\ \Delta X \end{bmatrix} = J^T \left(b - f(X) \right)$$

$$J^T J = \begin{bmatrix} J_p^T J_p & J_p^T J_X \\ J_X^T J_p & J_X^T J_X \end{bmatrix} = \begin{bmatrix} A & B \\ B^T & D \end{bmatrix}$$

$$\rightarrow \begin{bmatrix} J_p^T J_p & J_p^T J_X \\ J_X^T J_p & J_X^T J_X \end{bmatrix} \begin{bmatrix} \Delta p \\ \Delta X \end{bmatrix} = \begin{bmatrix} J_p^T \\ J_X^T \end{bmatrix} \left(b - f(X) \right) = \begin{bmatrix} e_p \\ e_X \end{bmatrix}$$

or

$$\begin{bmatrix} J_p^T J_p + \mu I & J_p^T J_X \\ J_X^T J_p & J_X^T J_X + \mu I \end{bmatrix} \begin{bmatrix} \Delta p \\ \Delta X \end{bmatrix} = \begin{bmatrix} e_p \\ e_X \end{bmatrix}$$

or

$$\begin{bmatrix} A & B \\ B^T & D \end{bmatrix} \begin{bmatrix} \Delta p \\ \Delta X \end{bmatrix} = \begin{bmatrix} e_p \\ e_X \end{bmatrix}$$

Normal equation:

$$J^T J \begin{bmatrix} \Delta p \\ \Delta X \end{bmatrix} = J^T \left(b - f(X) \right)$$

$$J^T J = \begin{bmatrix} D & \\ & \ddots \\ & & d_M \end{bmatrix} \longrightarrow D^{-1} = \begin{bmatrix} d_1^{-1} & & \\ & \ddots & \\ & & d_M^{-1} \end{bmatrix}$$

Inversion of block diagonal matrix can be efficiently computed.

$$\longrightarrow \begin{bmatrix} J_p^T J_p & J_p^T J_X \\ J_X^T J_p & J_X^T J_X \end{bmatrix} \begin{bmatrix} \Delta p \\ \Delta X \end{bmatrix} = \begin{bmatrix} J_p^T \\ J_X^T \end{bmatrix} \left(b - f(X) \right) = \begin{bmatrix} e_p \\ e_X \end{bmatrix}$$

or

$$\begin{bmatrix} J_p^T J_p + \mu I & J_p^T J_X \\ J_X^T J_p & J_X^T J_X + \mu I \end{bmatrix} \begin{bmatrix} \Delta p \\ \Delta X \end{bmatrix} = \begin{bmatrix} e_p \\ e_X \end{bmatrix}$$

or

$$\begin{bmatrix} A & B \\ B^T & D \end{bmatrix} \begin{bmatrix} \Delta p \\ \Delta X \end{bmatrix} = \begin{bmatrix} e_p \\ e_X \end{bmatrix}$$

$$J^T J = \begin{bmatrix} J_p^T J_p & J_p^T J_X \\ J_X^T J_p & J_X^T J_X \end{bmatrix} = \begin{bmatrix} A & B \\ B^T & D \end{bmatrix}$$

Normal equation:

$$J^T J \begin{bmatrix} \Delta p \\ \Delta X \end{bmatrix} = J^T \left(b - f(X) \right)$$

$$\rightarrow \begin{bmatrix} A & B \\ B^T & D \end{bmatrix} \begin{bmatrix} \Delta p \\ \Delta X \end{bmatrix} = \begin{bmatrix} e_p \\ e_X \end{bmatrix}$$

$$J^T J = \begin{bmatrix} D & \\ & \end{bmatrix}$$

$$D = \begin{bmatrix} d_1 & & \\ & \ddots & \\ & & d_M \end{bmatrix} \longrightarrow D^{-1} = \begin{bmatrix} d_1^{-1} & & \\ & \ddots & \\ & & d_M^{-1} \end{bmatrix}$$

Inversion of block diagonal matrix can be efficiently computed.

$$J^T J = \begin{bmatrix} J_p^T J_p & J_p^T J_X \\ J_X^T J_p & J_X^T J_X \end{bmatrix} = \begin{bmatrix} A & B \\ B^T & D \end{bmatrix}$$

Normal equation:

$$J^T J \begin{bmatrix} \Delta p \\ \Delta X \end{bmatrix} = J^T \left(b - f(X) \right)$$

$$\rightarrow \begin{bmatrix} A & B \\ B^T & D \end{bmatrix} \begin{bmatrix} \Delta p \\ \Delta X \end{bmatrix} = \begin{bmatrix} e_p \\ e_X \end{bmatrix}$$

$$\rightarrow \begin{bmatrix} I & -BD^{-1} \\ 0 & I \end{bmatrix} \begin{bmatrix} A & B \\ B^T & D \end{bmatrix} \begin{bmatrix} \Delta p \\ \Delta X \end{bmatrix} = \begin{bmatrix} I & -BD^{-1} \\ 0 & I \end{bmatrix} \begin{bmatrix} e_p \\ e_X \end{bmatrix}$$

$$J^T J = \begin{bmatrix} J_p^T J_p & J_p^T J_X \\ J_X^T J_p & J_X^T J_X \end{bmatrix} = \begin{bmatrix} A & B \\ B^T & D \end{bmatrix}$$

$$J^T J = \begin{bmatrix} D & \\ & \end{bmatrix}, \quad D = \begin{bmatrix} d_1 & & \\ & \ddots & \\ & & d_M \end{bmatrix} \rightarrow D^{-1} = \begin{bmatrix} d_1^{-1} & & \\ & \ddots & \\ & & d_M^{-1} \end{bmatrix}$$

Inversion of block diagonal matrix can be efficiently computed.

$$J^T J = \begin{bmatrix} D & \\ & \end{bmatrix}, \quad D = \begin{bmatrix} d_1 & & \\ & \ddots & \\ & & d_M \end{bmatrix} \longrightarrow D^{-1} = \begin{bmatrix} d_1^{-1} & & \\ & \ddots & \\ & & d_M^{-1} \end{bmatrix}$$

Inversion of block diagonal matrix can be efficiently computed.

$$J^T J = \begin{bmatrix} J_p^T J_p & J_p^T J_X \\ J_X^T J_p & J_X^T J_X \end{bmatrix} = \begin{bmatrix} A & B \\ B^T & D \end{bmatrix}$$

Normal equation:

$$J^T J \begin{bmatrix} \Delta p \\ \Delta X \end{bmatrix} = J^T \left(b - f(X) \right)$$

$$\longrightarrow \begin{bmatrix} A & B \\ B^T & D \end{bmatrix} \begin{bmatrix} \Delta p \\ \Delta X \end{bmatrix} = \begin{bmatrix} e_p \\ e_X \end{bmatrix}$$

$$\longrightarrow \begin{bmatrix} I & -BD^{-1} \\ 0 & I \end{bmatrix} \begin{bmatrix} A & B \\ B^T & D \end{bmatrix} \begin{bmatrix} \Delta p \\ \Delta X \end{bmatrix} = \begin{bmatrix} I & -BD^{-1} \\ 0 & I \end{bmatrix} \begin{bmatrix} e_p \\ e_X \end{bmatrix}$$

$$\longrightarrow \begin{bmatrix} A - BD^{-1}B^T & 0 \\ B^T & D \end{bmatrix} \begin{bmatrix} \Delta p \\ \Delta X \end{bmatrix} = \begin{bmatrix} e_p - BD^{-1}e_X \\ e_X \end{bmatrix}$$

$$J^T J = \begin{bmatrix} D & \\ & \end{bmatrix}, \quad D = \begin{bmatrix} d_1 & & \\ & \ddots & \\ & & d_M \end{bmatrix} \longrightarrow D^{-1} = \begin{bmatrix} d_1^{-1} & & \\ & \ddots & \\ & & d_M^{-1} \end{bmatrix}$$

Inversion of block diagonal matrix can be efficiently computed.

$$J^T J = \begin{bmatrix} J_p^T J_p & J_p^T J_X \\ J_X^T J_p & J_X^T J_X \end{bmatrix} = \begin{bmatrix} A & B \\ B^T & D \end{bmatrix}$$

Normal equation:

$$J^T J \begin{bmatrix} \Delta p \\ \Delta X \end{bmatrix} = J^T \left(b - f(X) \right)$$

$$\longrightarrow \begin{bmatrix} A & B \\ B^T & D \end{bmatrix} \begin{bmatrix} \Delta p \\ \Delta X \end{bmatrix} = \begin{bmatrix} e_p \\ e_X \end{bmatrix}$$

$$\longrightarrow \begin{bmatrix} I & -BD^{-1} \\ 0 & I \end{bmatrix} \begin{bmatrix} A & B \\ B^T & D \end{bmatrix} \begin{bmatrix} \Delta p \\ \Delta X \end{bmatrix} = \begin{bmatrix} I & -BD^{-1} \\ 0 & I \end{bmatrix} \begin{bmatrix} e_p \\ e_X \end{bmatrix}$$

$$\longrightarrow \begin{bmatrix} A - BD^{-1}B^T & 0 \\ B^T & D \end{bmatrix} \begin{bmatrix} \Delta p \\ \Delta X \end{bmatrix} = \begin{bmatrix} e_p - BD^{-1}e_X \\ e_X \end{bmatrix}$$

$$\longrightarrow \Delta p = \left(A - BD^{-1}B^T \right)^{-1} \left(e_p - BD^{-1}e_X \right)$$

$$\Delta X = D^{-1} \left(e_X - B^T \Delta p \right)$$

Note: $A - BD^{-1}B^T$ is Schur complement of D

$$J^T J = \begin{bmatrix} D & \\ & \end{bmatrix}, \quad D = \begin{bmatrix} d_1 & & \\ & \ddots & \\ & & d_M \end{bmatrix} \longrightarrow D^{-1} = \begin{bmatrix} d_1^{-1} & & \\ & \ddots & \\ & & d_M^{-1} \end{bmatrix}$$

Inversion of block diagonal matrix can be efficiently computed.

$$J^T J = \begin{bmatrix} J_p^T J_p & J_p^T J_X \\ J_X^T J_p & J_X^T J_X \end{bmatrix} = \begin{bmatrix} A & B \\ B^T & D \end{bmatrix}$$

Normal equation:

$$J^T J \begin{bmatrix} \Delta p \\ \Delta X \end{bmatrix} = J^T \left(b - f(X) \right)$$

$$\longrightarrow \begin{bmatrix} A & B \\ B^T & D \end{bmatrix} \begin{bmatrix} \Delta p \\ \Delta X \end{bmatrix} = \begin{bmatrix} e_p \\ e_X \end{bmatrix}$$

$$\longrightarrow \begin{bmatrix} I & -BD^{-1} \\ 0 & I \end{bmatrix} \begin{bmatrix} A & B \\ B^T & D \end{bmatrix} \begin{bmatrix} \Delta p \\ \Delta X \end{bmatrix} = \begin{bmatrix} I & -BD^{-1} \\ 0 & I \end{bmatrix} \begin{bmatrix} e_p \\ e_X \end{bmatrix}$$

$$\longrightarrow \begin{bmatrix} A - BD^{-1}B^T & 0 \\ B^T & D \end{bmatrix} \begin{bmatrix} \Delta p \\ \Delta X \end{bmatrix} = \begin{bmatrix} e_p - BD^{-1}e_X \\ e_X \end{bmatrix}$$

Small size matrix

$$\longrightarrow \Delta p = \left(A - BD^{-1}B^T \right)^{-1} \left(e_p - BD^{-1}e_X \right)$$

$$\Delta X = D^{-1} \left(e_X - B^T \Delta p \right)$$

Note: $A - BD^{-1}B^T$ is Schur complement of D

Algorithm 4 Bundle Adjustment

1: $\widehat{p} = \begin{bmatrix} \mathbf{p}_1^\mathsf{T} & \cdots & \mathbf{p}_I^\mathsf{T} \end{bmatrix}^\mathsf{T}$ and $\widehat{\mathbf{X}} = \begin{bmatrix} \mathbf{X}_1^\mathsf{T} & \cdots & \mathbf{X}_M^\mathsf{T} \end{bmatrix}$
2: **for** iter $= 1 :$ nIters **do**
3: Empty $\mathbf{J_p}$, $\mathbf{J_X}$, $\mathbf{b}$, $\mathbf{f}$, $\mathbf{D}_{\text{inv}}$.

$$J = \begin{bmatrix} J_p & J_X \end{bmatrix} \qquad (b - f(p)) \qquad D^{-1} = \begin{bmatrix} d_1^{-1} & & \\ & \ddots & \\ & & d_M^{-1} \end{bmatrix}$$

Algorithm 4 Bundle Adjustment

1: $\widehat{p} = \begin{bmatrix} \mathbf{p}_1^\mathsf{T} & \cdots & \mathbf{p}_I^\mathsf{T} \end{bmatrix}^\mathsf{T}$ and $\widehat{\mathbf{X}} = \begin{bmatrix} \mathbf{X}_1^\mathsf{T} & \cdots & \mathbf{X}_M^\mathsf{T} \end{bmatrix}$
2: **for** iter $= 1 :$ nIters **do**
3: Empty $\mathbf{J_p}$, $\mathbf{J_X}$, $\mathbf{b}$, $\mathbf{f}$, $\mathbf{D}_{\text{inv}}$.
4: **for** $i = 1 : M$ **do** ← # of points
5: $\mathbf{d} = \mathbf{0}_{3\times3}$
6: **for** $j = 1 : I$ **do** ← # of images
7: **if** the i^{th} point is visible from the j^{th} image **then** ← if visible

$$J = \begin{bmatrix} J_p & J_X \end{bmatrix} \qquad (b - f(p)) \qquad D^{-1} = \begin{bmatrix} d_1^{-1} & & \\ & \ddots & \\ & & d_M^{-1} \end{bmatrix}$$

Algorithm 4 Bundle Adjustment

1: $\widehat{p} = \begin{bmatrix} \mathbf{p}_1^\mathsf{T} & \cdots & \mathbf{p}_I^\mathsf{T} \end{bmatrix}^\mathsf{T}$ and $\widehat{\mathbf{X}} = \begin{bmatrix} \mathbf{X}_1^\mathsf{T} & \cdots & \mathbf{X}_M^\mathsf{T} \end{bmatrix}$
2: **for** iter $= 1 :$ nIters **do**
3: Empty $\mathbf{J_p}$, $\mathbf{J_X}$, $\mathbf{b}$, $\mathbf{f}$, $\mathbf{D}_{\text{inv}}$.
4: **for** $i = 1 : M$ **do** ← # of points
5: $\mathbf{d} = \mathbf{0}_{3\times3}$
6: **for** $j = 1 : I$ **do** ← # of images
7: **if** the i^{th} point is visible from the j^{th} image **then** ← if visible
8: $\mathbf{J}_1 = \mathbf{0}_{2\times7I}$ and $\mathbf{J}_2 = \mathbf{0}_{2\times3M}$
9: $\mathbf{J}_1(:, 7(j-1)+1 : 7j) = \frac{\partial f(\mathbf{p}_j, \mathbf{X}_i)}{\partial \mathbf{p}_j}$ ← J_p

$$J = \begin{bmatrix} J_p & J_X \end{bmatrix} \qquad (b - f(p)) \qquad D^{-1} = \begin{bmatrix} d_1^{-1} & & \\ & \ddots & \\ & & d_M^{-1} \end{bmatrix}$$

Algorithm 4 Bundle Adjustment

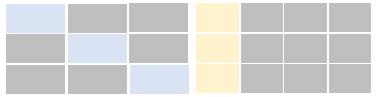
1: $\widehat{p} = \begin{bmatrix} \mathbf{p}_1^\mathsf{T} & \cdots & \mathbf{p}_I^\mathsf{T} \end{bmatrix}^\mathsf{T}$ and $\widehat{\mathbf{X}} = \begin{bmatrix} \mathbf{X}_1^\mathsf{T} & \cdots & \mathbf{X}_M^\mathsf{T} \end{bmatrix}$
2: **for** iter $= 1 :$ nIters **do**
3: Empty $\mathbf{J_p}$, $\mathbf{J_X}$, $\mathbf{b}$, $\mathbf{f}$, $\mathbf{D}_{\text{inv}}$.
4: **for** $i = 1 : M$ **do** ← # of points
5: $\mathbf{d} = \mathbf{0}_{3\times3}$
6: **for** $j = 1 : I$ **do** ← # of images
7: **if** the i^{th} point is visible from the j^{th} image **then** ← if visible
8: $\mathbf{J}_1 = \mathbf{0}_{2\times7I}$ and $\mathbf{J}_2 = \mathbf{0}_{2\times3M}$
9: $\mathbf{J}_1(:, 7(j-1)+1 : 7j) = \frac{\partial f(\mathbf{p}_j, \mathbf{X}_i)}{\partial \mathbf{p}_j}$ ← J_p
10: $\mathbf{J}_2(:, 3(i-1)+1 : 3i) = \frac{\partial f(\mathbf{p}_j, \mathbf{X}_i)}{\partial \mathbf{X}_i}$ ← J_X

$$J = \begin{bmatrix} J_p & J_X \end{bmatrix} \qquad (b - f(p)) \qquad D^{-1} = \begin{bmatrix} d_1^{-1} & & \\ & \ddots & \\ & & d_M^{-1} \end{bmatrix}$$

Algorithm 4 Bundle Adjustment

1: $\widehat{p} = \begin{bmatrix} \mathbf{p}_1^\mathsf{T} & \cdots & \mathbf{p}_I^\mathsf{T} \end{bmatrix}^\mathsf{T}$ and $\widehat{\mathbf{X}} = \begin{bmatrix} \mathbf{X}_1^\mathsf{T} & \cdots & \mathbf{X}_M^\mathsf{T} \end{bmatrix}$
2: **for** iter $= 1 :$ nIters **do**
3: Empty $\mathbf{J_p}$, $\mathbf{J_X}$, $\mathbf{b}$, $\mathbf{f}$, $\mathbf{D}_{\text{inv}}$.
4: **for** $i = 1 : M$ **do** ← # of points
5: $\mathbf{d} = \mathbf{0}_{3\times3}$
6: **for** $j = 1 : I$ **do** ← # of images
7: **if** the i^{th} point is visible from the j^{th} image **then** ← if visible
8: $\mathbf{J}_1 = \mathbf{0}_{2\times7I}$ and $\mathbf{J}_2 = \mathbf{0}_{2\times3M}$
9: $\mathbf{J}_1(:, 7(j-1)+1 : 7j) = \frac{\partial f(\mathbf{p}_j, \mathbf{X}_i)}{\partial \mathbf{p}_j}$ ← J_p
10: $\mathbf{J}_2(:, 3(i-1)+1 : 3i) = \frac{\partial f(\mathbf{p}_j, \mathbf{X}_i)}{\partial \mathbf{X}_i}$ ← J_X
11: $\mathbf{J_p} = \begin{bmatrix} \mathbf{J}_\mathbf{p}^\mathsf{T} & \mathbf{J}_1^\mathsf{T} \end{bmatrix}^\mathsf{T}$ and $\mathbf{J_X} = \begin{bmatrix} \mathbf{J}_\mathbf{X}^\mathsf{T} & \mathbf{J}_2^\mathsf{T} \end{bmatrix}^\mathsf{T}$ ←

$$J = \begin{bmatrix} J_p & J_X \end{bmatrix} \qquad (b - f(p)) \qquad D^{-1} = \begin{bmatrix} d_1^{-1} & & \\ & \ddots & \\ & & d_M^{-1} \end{bmatrix}$$



Algorithm 4 Bundle Adjustment

$\widehat{p} = \begin{bmatrix} \mathbf{p}_1^\mathsf{T} & \cdots & \mathbf{p}_I^\mathsf{T} \end{bmatrix}^\mathsf{T}$ and $\widehat{\mathbf{X}} = \begin{bmatrix} \mathbf{X}_1^\mathsf{T} & \cdots & \mathbf{X}_M^\mathsf{T} \end{bmatrix}$

for iter $= 1 :$ nIters **do**

  Empty $\mathbf{J_p}$, $\mathbf{J_X}$, $\mathbf{b}$, $\mathbf{f}$, $\mathbf{D}_{\text{inv}}$.

  for $i = 1 : M$ **do** ← # of points

    $\mathbf{d} = \mathbf{0}_{3\times3}$

    for $j = 1 : I$ **do** ← # of images

      if the i^{th} point is visible from the j^{th} image **then** ← if visible

        $\mathbf{J}_1 = \mathbf{0}_{2\times7I}$ and $\mathbf{J}_2 = \mathbf{0}_{2\times3M}$

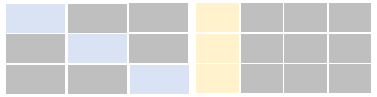
        $\mathbf{J}_1(:, 7(j-1)+1 : 7j) = \frac{\partial f(\mathbf{p}_j, \mathbf{X}_i)}{\partial \mathbf{p}_j}$ ← J_p

        $\mathbf{J}_2(:, 3(i-1)+1 : 3i) = \frac{\partial f(\mathbf{p}_j, \mathbf{X}_i)}{\partial \mathbf{X}_i}$ ← J_X

        $\mathbf{J_p} = \begin{bmatrix} \mathbf{J}_\mathbf{p}^\mathsf{T} & \mathbf{J}_1^\mathsf{T} \end{bmatrix}^\mathsf{T}$ and $\mathbf{J_X} = \begin{bmatrix} \mathbf{J}_\mathbf{X}^\mathsf{T} & \mathbf{J}_2^\mathsf{T} \end{bmatrix}^\mathsf{T}$ ←

        $\mathbf{d} = \mathbf{d} + \frac{\partial f(\mathbf{p}_j, \mathbf{X}_i)}{\partial \mathbf{X}_i}^\mathsf{T} \frac{\partial f(\mathbf{p}_j, \mathbf{X}_i)}{\partial \mathbf{X}_i}$ ← d

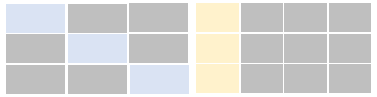
$$J = \begin{bmatrix} J_p & J_X \end{bmatrix} \qquad \begin{pmatrix} b - f(p) \end{pmatrix} \qquad D^{-1} = \begin{bmatrix} d_1^{-1} & & \\ & \ddots & \\ & & d_M^{-1} \end{bmatrix}$$



Algorithm 4 Bundle Adjustment

1: $\widehat{p} = \begin{bmatrix} \mathbf{p}_1^\mathsf{T} & \cdots & \mathbf{p}_I^\mathsf{T} \end{bmatrix}^\mathsf{T}$ and $\widehat{\mathbf{X}} = \begin{bmatrix} \mathbf{X}_1^\mathsf{T} & \cdots & \mathbf{X}_M^\mathsf{T} \end{bmatrix}$
2: **for** iter $= 1 :$ nIters **do**
3: Empty $\mathbf{J_p}$, $\mathbf{J_X}$, $\mathbf{b}$, $\mathbf{f}$, $\mathbf{D}_{\text{inv}}$.
4: **for** $i = 1 : M$ **do** ← # of points
5: $\mathbf{d} = \mathbf{0}_{3\times3}$
6: **for** $j = 1 : I$ **do** ← # of images
7: **if** the i^{th} point is visible from the j^{th} image **then** ← if visible
8: $\mathbf{J}_1 = \mathbf{0}_{2\times7I}$ and $\mathbf{J}_2 = \mathbf{0}_{2\times3M}$
9: $\mathbf{J}_1(:, 7(j-1)+1 : 7j) = \frac{\partial f(\mathbf{p}_j, \mathbf{X}_i)}{\partial \mathbf{p}_j}$ ← J_p
10: $\mathbf{J}_2(:, 3(i-1)+1 : 3i) = \frac{\partial f(\mathbf{p}_j, \mathbf{X}_i)}{\partial \mathbf{X}_i}$ ← J_X
11: $\mathbf{J_p} = \begin{bmatrix} \mathbf{J}_\mathbf{p}^\mathsf{T} & \mathbf{J}_1^\mathsf{T} \end{bmatrix}^\mathsf{T}$ and $\mathbf{J_X} = \begin{bmatrix} \mathbf{J}_\mathbf{X}^\mathsf{T} & \mathbf{J}_2^\mathsf{T} \end{bmatrix}^\mathsf{T}$ ←
12: $\mathbf{d} = \mathbf{d} + \frac{\partial f(\mathbf{p}_j, \mathbf{X}_i)}{\partial \mathbf{X}_i}^\mathsf{T} \frac{\partial f(\mathbf{p}_j, \mathbf{X}_i)}{\partial \mathbf{X}_i}$ ← d
13: $\mathbf{b} = \begin{bmatrix} \mathbf{b}^\mathsf{T} & \mathbf{u}_{ij}^\mathsf{T} \end{bmatrix}$
14: $\mathbf{f} = \begin{bmatrix} \mathbf{f}^\mathsf{T} & \mathbf{x}_{ij}^\mathsf{T} \end{bmatrix}$ where $\mu \begin{bmatrix} \mathbf{x}_{ij} \\ 1 \end{bmatrix} = \mathbf{R}_j \begin{bmatrix} \mathbf{I} & -\mathbf{C}_j \end{bmatrix}$ ← $(b - f(p))$

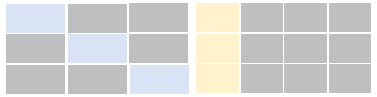
$$J = \begin{bmatrix} J_p & J_X \end{bmatrix} \qquad (b - f(p)) \qquad D^{-1} = \begin{bmatrix} d_1^{-1} & & \\ & \ddots & \\ & & d_M^{-1} \end{bmatrix}$$



Algorithm 4 Bundle Adjustment

$\widehat{p} = \begin{bmatrix} \mathbf{p}_1^\mathsf{T} & \cdots & \mathbf{p}_I^\mathsf{T} \end{bmatrix}^\mathsf{T}$ and $\widehat{\mathbf{X}} = \begin{bmatrix} \mathbf{X}_1^\mathsf{T} & \cdots & \mathbf{X}_M^\mathsf{T} \end{bmatrix}$
for iter $= 1 :$ nIters **do**
  Empty $\mathbf{J_p}$, $\mathbf{J_X}$, $\mathbf{b}$, $\mathbf{f}$, $\mathbf{D}_{\text{inv}}$.
  for $i = 1 : M$ **do** ← # of points
    $\mathbf{d} = \mathbf{0}_{3\times3}$
    for $j = 1 : I$ **do** ← # of images
      if the i^{th} point is visible from the j^{th} image **then** ← if visible
        $\mathbf{J}_1 = \mathbf{0}_{2\times7I}$ and $\mathbf{J}_2 = \mathbf{0}_{2\times3M}$
        $\mathbf{J}_1(:, 7(j-1)+1 : 7j) = \frac{\partial f(\mathbf{p}_j, \mathbf{X}_i)}{\partial \mathbf{p}_j}$ ← J_p
        $\mathbf{J}_2(:, 3(i-1)+1 : 3i) = \frac{\partial f(\mathbf{p}_j, \mathbf{X}_i)}{\partial \mathbf{X}_i}$ ← J_X
        $\mathbf{J_p} = \begin{bmatrix} \mathbf{J}_\mathbf{p}^\mathsf{T} & \mathbf{J}_1^\mathsf{T} \end{bmatrix}^\mathsf{T}$ and $\mathbf{J_X} = \begin{bmatrix} \mathbf{J}_\mathbf{X}^\mathsf{T} & \mathbf{J}_2^\mathsf{T} \end{bmatrix}^\mathsf{T}$ ←
        $\mathbf{d} = \mathbf{d} + \frac{\partial f(\mathbf{p}_j, \mathbf{X}_i)}{\partial \mathbf{X}_i}^\mathsf{T} \frac{\partial f(\mathbf{p}_j, \mathbf{X}_i)}{\partial \mathbf{X}_i}$ ← d
        $\mathbf{b} = \begin{bmatrix} \mathbf{b}^\mathsf{T} & \mathbf{u}_{ij}^\mathsf{T} \end{bmatrix}$
        $\mathbf{f} = \begin{bmatrix} \mathbf{f}^\mathsf{T} & \mathbf{x}_{ij}^\mathsf{T} \end{bmatrix}$ where $\mu \begin{bmatrix} \mathbf{x}_{ij} \\ 1 \end{bmatrix} = \mathbf{R}_j \begin{bmatrix} \mathbf{I} & -\mathbf{C}_j \end{bmatrix}$ ← $(b - f(p))$
      end if
    end for
    $\mathbf{d} = \mathbf{d} + \lambda \mathbf{I}$ ← $d = d + \lambda I$
    $\mathbf{D}_{\text{inv}} = \texttt{blkdiag}(\mathbf{D}_{\text{inv}}, \mathbf{d}^{-1})$

$J = \begin{bmatrix} J_p & J_X \end{bmatrix}$ $(b - f(p))$ $D^{-1} = \begin{bmatrix} d_1^{-1} & & \\ & \ddots & \\ & & d_M^{-1} \end{bmatrix}$

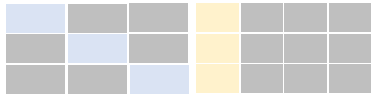


$D^{-1} = \begin{bmatrix} d_1^{-1} & & \\ & \ddots & \\ & & d_M^{-1} \end{bmatrix}$

Algorithm 4 Bundle Adjustment

$\widehat{p} = \begin{bmatrix} \mathbf{p}_1^\mathsf{T} & \cdots & \mathbf{p}_I^\mathsf{T} \end{bmatrix}^\mathsf{T}$ and $\widehat{\mathbf{X}} = \begin{bmatrix} \mathbf{X}_1^\mathsf{T} & \cdots & \mathbf{X}_M^\mathsf{T} \end{bmatrix}$
for iter $= 1 :$ nIters **do**
  Empty $\mathbf{J_p}$, $\mathbf{J_X}$, $\mathbf{b}$, $\mathbf{f}$, $\mathbf{D}_{\text{inv}}$.
  for $i = 1 : M$ **do** ← # of points
    $\mathbf{d} = \mathbf{0}_{3\times3}$
    for $j = 1 : I$ **do** ← # of images
      if the i^{th} point is visible from the j^{th} image **then** ← if visible
        $\mathbf{J}_1 = \mathbf{0}_{2\times7I}$ and $\mathbf{J}_2 = \mathbf{0}_{2\times3M}$
        $\mathbf{J}_1(:, 7(j-1)+1 : 7j) = \frac{\partial f(\mathbf{p}_j, \mathbf{X}_i)}{\partial \mathbf{p}_j}$ ← J_p
        $\mathbf{J}_2(:, 3(i-1)+1 : 3i) = \frac{\partial f(\mathbf{p}_j, \mathbf{X}_i)}{\partial \mathbf{X}_i}$ ← J_X
        $\mathbf{J_p} = \begin{bmatrix} \mathbf{J}_\mathbf{p}^\mathsf{T} & \mathbf{J}_1^\mathsf{T} \end{bmatrix}^\mathsf{T}$ and $\mathbf{J_X} = \begin{bmatrix} \mathbf{J}_\mathbf{X}^\mathsf{T} & \mathbf{J}_2^\mathsf{T} \end{bmatrix}^\mathsf{T}$ ←
        $\mathbf{d} = \mathbf{d} + \frac{\partial f(\mathbf{p}_j, \mathbf{X}_i)}{\partial \mathbf{X}_i}^\mathsf{T} \frac{\partial f(\mathbf{p}_j, \mathbf{X}_i)}{\partial \mathbf{X}_i}$ ← d
        $\mathbf{b} = \begin{bmatrix} \mathbf{b}^\mathsf{T} & \mathbf{u}_{ij}^\mathsf{T} \end{bmatrix}$
        $\mathbf{f} = \begin{bmatrix} \mathbf{f}^\mathsf{T} & \mathbf{x}_{ij}^\mathsf{T} \end{bmatrix}$ where $\mu \begin{bmatrix} \mathbf{x}_{ij} \\ 1 \end{bmatrix} = \mathbf{R}_j \begin{bmatrix} \mathbf{I} & -\mathbf{C}_j \end{bmatrix}$ ← $(b - f(p))$
      end if
    end for
    $\mathbf{d} = \mathbf{d} + \lambda \mathbf{I}$ ← $d = d + \lambda I$
    $\mathbf{D}_{\text{inv}} = \texttt{blkdiag}(\mathbf{D}_{\text{inv}}, \mathbf{d}^{-1})$
  end for
  $\mathbf{e_p} = \mathbf{J}_\mathbf{p}^\mathsf{T}(\mathbf{b} - \mathbf{f})$ ← $\begin{bmatrix} J_p^T \\ J_X^T \end{bmatrix}(b - f(X)) = \begin{bmatrix} e_p \\ e_X \end{bmatrix}$
  $\mathbf{e_X} = \mathbf{J}_\mathbf{X}^\mathsf{T}(\mathbf{b} - \mathbf{f})$

$J = \begin{bmatrix} J_p & J_X \end{bmatrix}$ $\quad (b - f(p))$ $\quad D^{-1} = \begin{bmatrix} d_1^{-1} & & \\ & \ddots & \\ & & d_M^{-1} \end{bmatrix}$

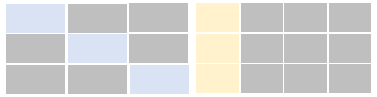


$D^{-1} = \begin{bmatrix} d_1^{-1} & & \\ & \ddots & \\ & & d_M^{-1} \end{bmatrix}$

Algorithm 4 Bundle Adjustment

$\widehat{p} = \begin{bmatrix} \mathbf{p}_1^\mathsf{T} & \cdots & \mathbf{p}_I^\mathsf{T} \end{bmatrix}^\mathsf{T}$ and $\widehat{\mathbf{X}} = \begin{bmatrix} \mathbf{X}_1^\mathsf{T} & \cdots & \mathbf{X}_M^\mathsf{T} \end{bmatrix}$
for iter $= 1 :$ nIters **do**
  Empty $\mathbf{J_p}$, $\mathbf{J_X}$, $\mathbf{b}$, $\mathbf{f}$, $\mathbf{D}_{\text{inv}}$.
  for $i = 1 : M$ **do** ← # of points
    $\mathbf{d} = \mathbf{0}_{3\times3}$
    for $j = 1 : I$ **do** ← # of images
      if the i^{th} point is visible from the j^{th} image **then** ← if visible
        $\mathbf{J}_1 = \mathbf{0}_{2\times7I}$ and $\mathbf{J}_2 = \mathbf{0}_{2\times3M}$
        $\mathbf{J}_1(:, 7(j-1)+1 : 7j) = \frac{\partial f(\mathbf{p}_j, \mathbf{X}_i)}{\partial \mathbf{p}_j}$ ← J_p
        $\mathbf{J}_2(:, 3(i-1)+1 : 3i) = \frac{\partial f(\mathbf{p}_j, \mathbf{X}_i)}{\partial \mathbf{X}_i}$ ← J_X
        $\mathbf{J_p} = \begin{bmatrix} \mathbf{J}_\mathbf{p}^\mathsf{T} & \mathbf{J}_1^\mathsf{T} \end{bmatrix}^\mathsf{T}$ and $\mathbf{J_X} = \begin{bmatrix} \mathbf{J}_\mathbf{X}^\mathsf{T} & \mathbf{J}_2^\mathsf{T} \end{bmatrix}^\mathsf{T}$ ←
        $\mathbf{d} = \mathbf{d} + \frac{\partial f(\mathbf{p}_j, \mathbf{X}_i)}{\partial \mathbf{X}_i}^\mathsf{T} \frac{\partial f(\mathbf{p}_j, \mathbf{X}_i)}{\partial \mathbf{X}_i}$ ← d
        $\mathbf{b} = \begin{bmatrix} \mathbf{b}^\mathsf{T} & \mathbf{u}_{ij}^\mathsf{T} \end{bmatrix}$
        $\mathbf{f} = \begin{bmatrix} \mathbf{f}^\mathsf{T} & \mathbf{x}_{ij}^\mathsf{T} \end{bmatrix}$ where $\mu \begin{bmatrix} \mathbf{x}_{ij} \\ 1 \end{bmatrix} = \mathbf{R}_j \begin{bmatrix} \mathbf{I} & -\mathbf{C}_j \end{bmatrix}$ ← $(b - f(p))$
      end if
    end for
    $\mathbf{d} = \mathbf{d} + \lambda \mathbf{I}$ ← $d = d + \lambda I$
    $\mathbf{D}_{\text{inv}} = \texttt{blkdiag}(\mathbf{D}_{\text{inv}}, \mathbf{d}^{-1})$
  end for
  $\mathbf{e_p} = \mathbf{J}_\mathbf{p}^\mathsf{T}(\mathbf{b} - \mathbf{f})$ ← $\begin{bmatrix} J_p^T \\ J_X^T \end{bmatrix} (b - f(X)) = \begin{bmatrix} e_p \\ e_X \end{bmatrix}$
  $\mathbf{e_X} = \mathbf{J}_\mathbf{X}^\mathsf{T}(\mathbf{b} - \mathbf{f})$
  $\mathbf{A} = \mathbf{J}_\mathbf{p}^\mathsf{T}\mathbf{J_p} + \lambda\mathbf{I}$, $\mathbf{B} = \mathbf{J}_\mathbf{p}^\mathsf{T}\mathbf{J_X}$, $\mathbf{D}^{-1} = \mathbf{D}_{\text{inv}}$ ← $\begin{bmatrix} A & B \\ B^T & D \end{bmatrix} = \begin{bmatrix} J_p^T J_p + \mu I & J_p^T J_X \\ J_X^T J_p & J_X^T J_X + \mu I \end{bmatrix}$

$J = \begin{bmatrix} J_p & J_X \end{bmatrix}$ $\quad (b - f(p))$ $\quad D^{-1} = \begin{bmatrix} d_1^{-1} & & \\ & \ddots & \\ & & d_M^{-1} \end{bmatrix}$

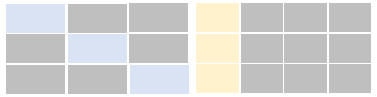


$D^{-1} = \begin{bmatrix} d_1^{-1} & & \\ & \ddots & \\ & & d_M^{-1} \end{bmatrix}$

Algorithm 4 Bundle Adjustment

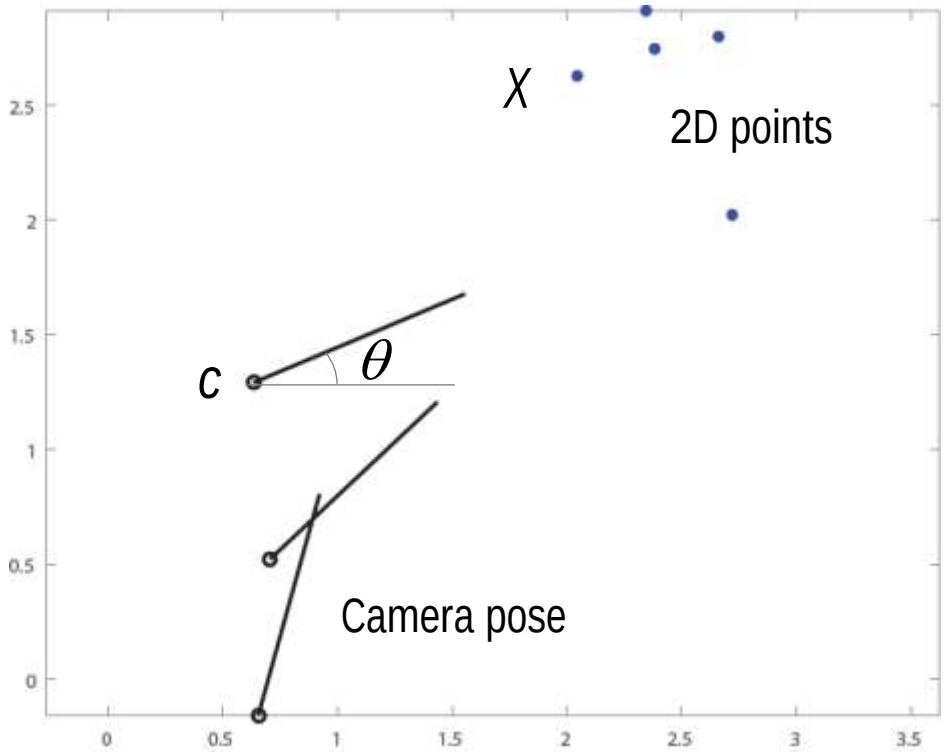
$\hat{p} = \begin{bmatrix} \mathbf{p}_1^\mathsf{T} & \cdots & \mathbf{p}_I^\mathsf{T} \end{bmatrix}^\mathsf{T}$ and $\hat{\mathbf{X}} = \begin{bmatrix} \mathbf{X}_1^\mathsf{T} & \cdots & \mathbf{X}_M^\mathsf{T} \end{bmatrix}$
for iter $= 1 :$ nIters **do**
  Empty $\mathbf{J_p}$, $\mathbf{J_X}$, $\mathbf{b}$, $\mathbf{f}$, $\mathbf{D}_{\text{inv}}$.
  for $i = 1 : M$ **do** ← # of points
    $\mathbf{d} = \mathbf{0}_{3\times3}$
    for $j = 1 : I$ **do** ← # of images
      if the i^{th} point is visible from the j^{th} image **then** ← if visible
        $\mathbf{J}_1 = \mathbf{0}_{2\times7I}$ and $\mathbf{J}_2 = \mathbf{0}_{2\times3M}$
        $\mathbf{J}_1(:, 7(j-1)+1 : 7j) = \frac{\partial f(\mathbf{p}_j, \mathbf{X}_i)}{\partial \mathbf{p}_j}$ ← J_p
        $\mathbf{J}_2(:, 3(i-1)+1 : 3i) = \frac{\partial f(\mathbf{p}_j, \mathbf{X}_i)}{\partial \mathbf{X}_i}$ ← J_X
        $\mathbf{J_p} = \begin{bmatrix} \mathbf{J}_\mathbf{p}^\mathsf{T} & \mathbf{J}_1^\mathsf{T} \end{bmatrix}^\mathsf{T}$ and $\mathbf{J_X} = \begin{bmatrix} \mathbf{J}_\mathbf{X}^\mathsf{T} & \mathbf{J}_2^\mathsf{T} \end{bmatrix}^\mathsf{T}$ ←
        $\mathbf{d} = \mathbf{d} + \frac{\partial f(\mathbf{p}_j, \mathbf{X}_i)}{\partial \mathbf{X}_i}^\mathsf{T} \frac{\partial f(\mathbf{p}_j, \mathbf{X}_i)}{\partial \mathbf{X}_i}$ ← d
        $\mathbf{b} = \begin{bmatrix} \mathbf{b}^\mathsf{T} & \mathbf{u}_{ij}^\mathsf{T} \end{bmatrix}$
        $\mathbf{f} = \begin{bmatrix} \mathbf{f}^\mathsf{T} & \mathbf{x}_{ij}^\mathsf{T} \end{bmatrix}$ where $\mu \begin{bmatrix} \mathbf{x}_{ij} \\ 1 \end{bmatrix} = \mathbf{R}_j \begin{bmatrix} \mathbf{I} & -\mathbf{C}_j \end{bmatrix}$ ← $(b - f(p))$
      end if
    end for
    $\mathbf{d} = \mathbf{d} + \lambda \mathbf{I}$ ← $d = d + \lambda I$
    $\mathbf{D}_{\text{inv}} = \texttt{blkdiag}(\mathbf{D}_{\text{inv}}, \mathbf{d}^{-1})$
  end for
  $\mathbf{e_p} = \mathbf{J}_\mathbf{p}^\mathsf{T}(\mathbf{b} - \mathbf{f})$ ← $\begin{bmatrix} J_p^T \\ J_X^T \end{bmatrix}(b - f(X)) = \begin{bmatrix} e_p \\ e_X \end{bmatrix}$
  $\mathbf{e_X} = \mathbf{J}_\mathbf{X}^\mathsf{T}(\mathbf{b} - \mathbf{f})$
  $\mathbf{A} = \mathbf{J}_\mathbf{p}^\mathsf{T}\mathbf{J_p} + \lambda\mathbf{I}$, $\mathbf{B} = \mathbf{J}_\mathbf{p}^\mathsf{T}\mathbf{J_X}$, $\mathbf{D}^{-1} = \mathbf{D}_{\text{inv}}$ ← $\begin{bmatrix} A & B \\ B^T & D \end{bmatrix} = \begin{bmatrix} J_p^T J_p + \mu I & J_p^T J_X \\ J_X^T J_p & J_X^T J_X + \mu I \end{bmatrix}$
  $\Delta\hat{\mathbf{p}} = (\mathbf{A} - \mathbf{B}\mathbf{D}^{-1}\mathbf{B}^\mathsf{T})^{-1}(\mathbf{e_p} - \mathbf{B}\mathbf{D}^{-1}\mathbf{e_X})$ ← $\Delta p = (A - BD^{-1}B^T)^{-1}(e_p - BD^{-1}e_X)$
  Normalize quaternions.
  $\Delta\hat{\mathbf{X}} = \mathbf{D}^{-1}(\mathbf{e_X} - \mathbf{B}^\mathsf{T}\Delta\hat{\mathbf{p}})$ ← $\Delta X = D^{-1}(e_X - B^T \Delta p)$
end for

$J = \begin{bmatrix} J_p & J_X \end{bmatrix}$ $(b - f(p))$ $D^{-1} = \begin{bmatrix} d_1^{-1} & & \\ & \ddots & \\ & & d_M^{-1} \end{bmatrix}$



$D^{-1} = \begin{bmatrix} d_1^{-1} & & \\ & \ddots & \\ & & d_M^{-1} \end{bmatrix}$

1D Camera Bundle Adjustment

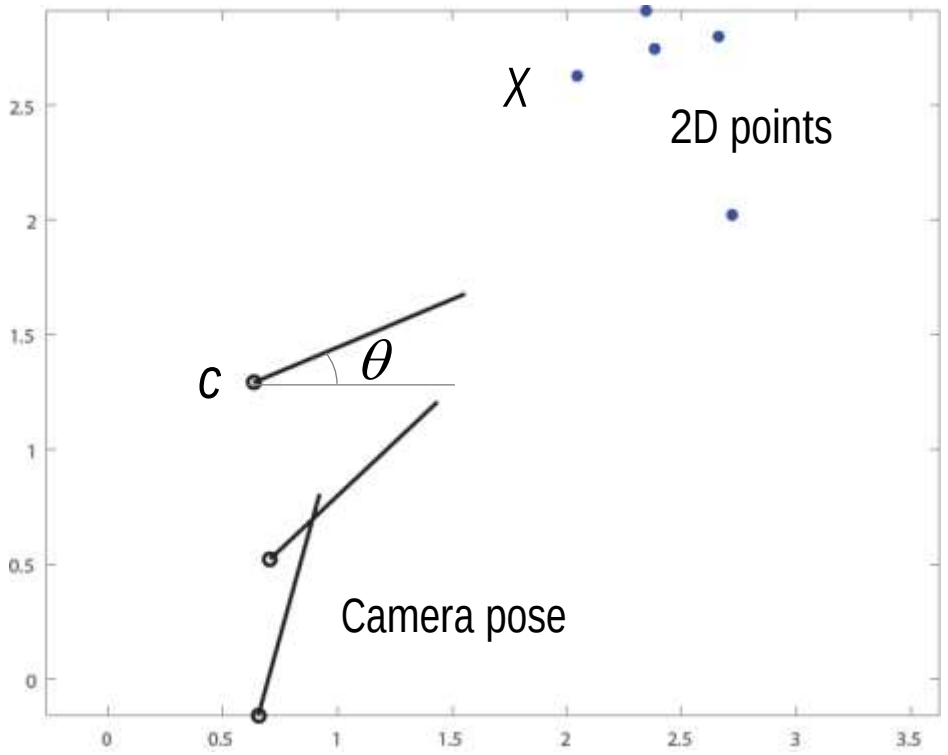


BundleAdjustment1D.m

Data generation

```
nPoints = 5;
nCameras = 3;
X = rand(nPoints, 2) + 2;
c = rand(nCameras,2);
for i = 1 : nCameras
    m = X-ones(nPoints,1)*c(i,:);
    theta = atan2(m(:,2), m(:,1));
    theta = mean(theta);

    C(i).c = c(i,:)';
    theta = theta;
    C(i).R = [sin(theta) -cos(theta); cos(theta) sin(theta)];

    u = C(i).R*[eye(2) -C(i).c] * [X'; ones(1,nPoints)];
    C(i).m = u(1,:)./u(2,:);

    C(i).c = c(i,:)'+0.3*randn(2,1);
    theta = theta + 0.3*randn(1,1);
    C(i).R = [sin(theta) -cos(theta); cos(theta) sin(theta)];
end
X = X + 0.3*randn(size(X));

[C X] = BA(C, X);
```

1D Camera Bundle Adjustment



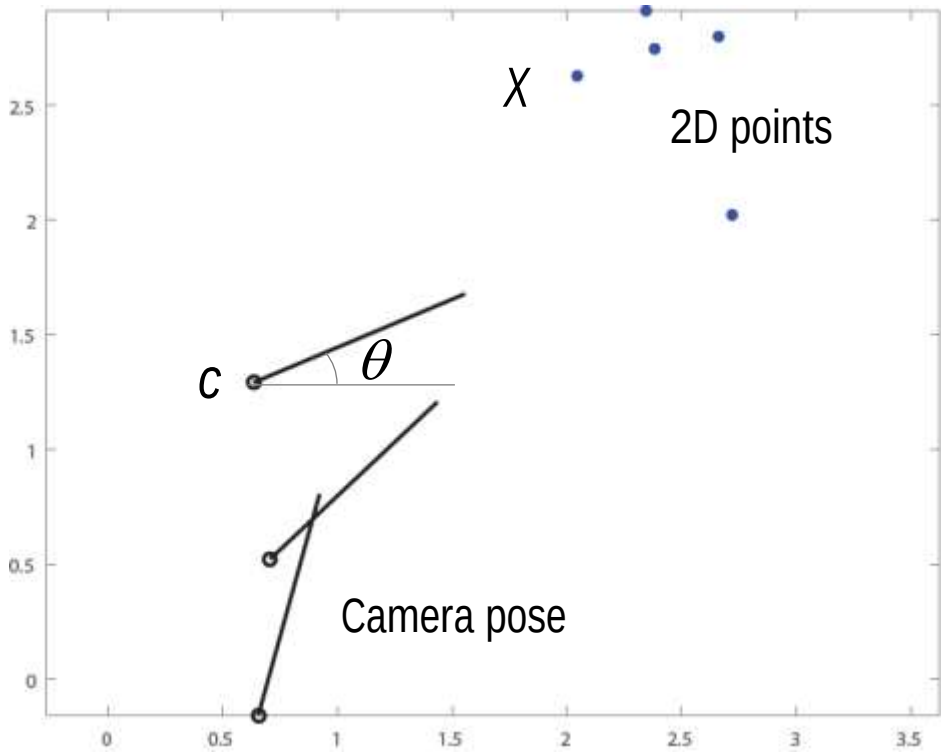
BundleAdjustment1D.m

Data generation

```
nPoints = 5;
nCameras = 3;
X = rand(nPoints, 2) + 2;
c = rand(nCameras,2);
for i = 1 : nCameras
   m = X-ones(nPoints,1)*c(i,:);
   theta = atan2(m(:,2), m(:,1));
   theta = mean(theta);

   C(i).c = c(i,:)';
   theta = theta;
   C(i).R = [sin(theta) -cos(theta); cos(theta) sin(theta)];

   u = C(i).R*[eye(2) -C(i).c] * [X'; ones(1,nPoints)];
   C(i).m = u(1,:)./u(2,:);

   C(i).c = c(i,:)'+0.3*randn(2,1);
   theta = theta + 0.3*randn(1,1);
   C(i).R = [sin(theta) -cos(theta); cos(theta) sin(theta)];
end
X = X + 0.3*randn(size(X));

[C X] = BA(C, X);
```

Ground truth projection

1D Camera Bundle Adjustment



BundleAdjustment1D.m

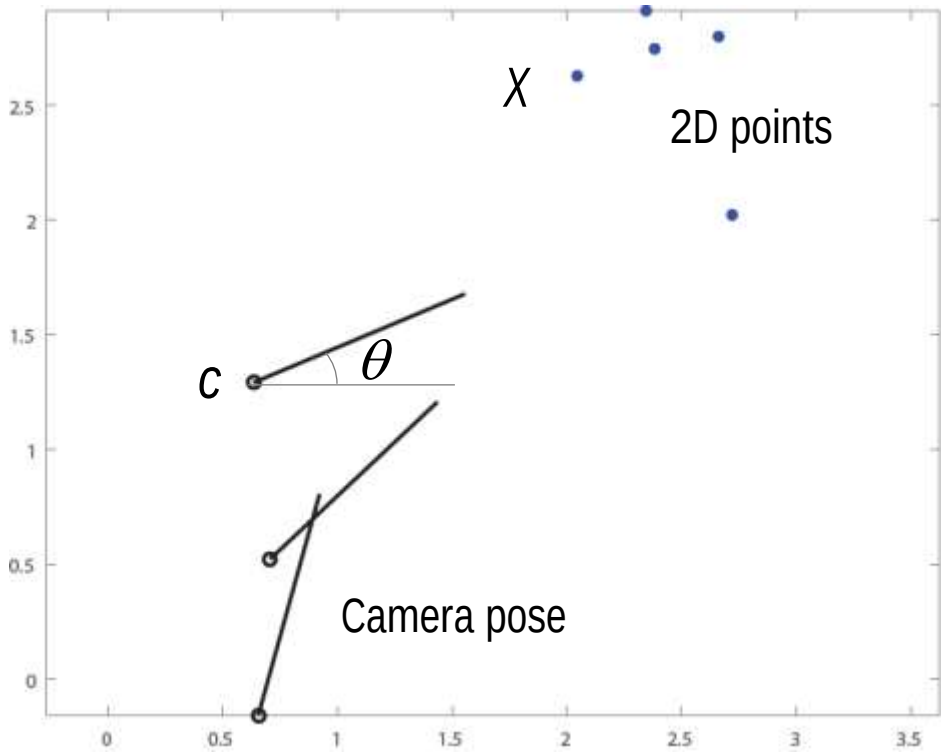
```
nPoints = 5;
nCameras = 3;
X = rand(nPoints, 2) + 2;
c = rand(nCameras,2);
for i = 1 : nCameras
   m = X-ones(nPoints,1)*c(i,:);
   theta = atan2(m(:,2), m(:,1));
   theta = mean(theta);

   C(i).c = c(i,:)';
   theta = theta;
   C(i).R = [sin(theta) -cos(theta); cos(theta) sin(theta)];

   u = C(i).R*[eye(2) -C(i).c] * [X'; ones(1,nPoints)];
   C(i).m = u(1,:)./u(2,:);

   C(i).c = c(i,:)'+0.3*randn(2,1);
   theta = theta + 0.3*randn(1,1);
   C(i).R = [sin(theta) -cos(theta); cos(theta) sin(theta)];
end
X = X + 0.3*randn(size(X));

[C X] = BA(C, X);
```

Data generation

Ground truth projection

Add noise

1D Camera Bundle Adjustment



BundleAdjustment1D.m

```
nPoints = 5;
nCameras = 3;
X = rand(nPoints, 2) + 2;
c = rand(nCameras,2);
for i = 1 : nCameras
   m = X-ones(nPoints,1)*c(i,:);
   theta = atan2(m(:,2), m(:,1));
   theta = mean(theta);

   C(i).c = c(i,:)';
   theta = theta;
   C(i).R = [sin(theta) -cos(theta); cos(theta) sin(theta)];

   u = C(i).R*[eye(2) -C(i).c] * [X'; ones(1,nPoints)];
   C(i).m = u(1,:)./u(2,:);

   C(i).c = c(i,:)'+0.3*randn(2,1);
   theta = theta + 0.3*randn(1,1);
   C(i).R = [sin(theta) -cos(theta); cos(theta) sin(theta)];
end
X = X + 0.3*randn(size(X));

[C X] = BA(C, X);
```

Data generation

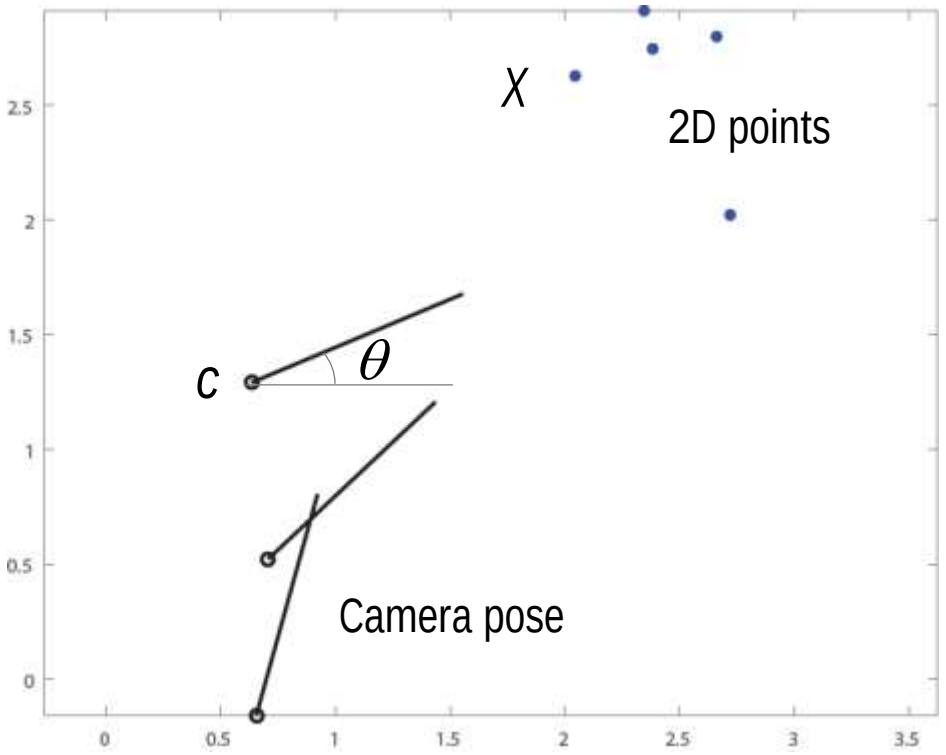
Ground truth projection

Add noise

← 1D bundle adjustment

1D Camera Bundle Adjustment

BundleAdjustment1D.m




```
function [C X] = BA(C, X)

lambda = 0.5;
nIters = 100;

xp = [];
for i = 1 : length(C)
   theta=atan2(C(i).R(2,2), C(i).R(2,1));   xp = [xp; C(i).c; theta];
end

xx = [];
for i = 1 : size(X,1)
   xx = [xx; X(i,:)'];
end
```

Algorithm 4 Bundle Adjustment

$\widehat{p} = \begin{bmatrix} \mathbf{p}_1^\mathsf{T} & \cdots & \mathbf{p}_I^\mathsf{T} \end{bmatrix}^\mathsf{T}$ and $\widehat{\mathbf{X}} = \begin{bmatrix} \mathbf{X}_1^\mathsf{T} & \cdots & \mathbf{X}_M^\mathsf{T} \end{bmatrix}$
for iter $= 1 :$ nIters **do**
  Empty $\mathbf{J_p}$, $\mathbf{J_X}$, $\mathbf{b}$, $\mathbf{f}$, $\mathbf{D}_{\text{inv}}$.
  for $i = 1 : M$ **do**
    $\mathbf{d} = \mathbf{0}_{3\times 3}$
    for $j = 1 : I$ **do**
      if the i^{th} point is visible from the j^{th} image **then**
        $\mathbf{J}_1 = \mathbf{0}_{2\times 7I}$ and $\mathbf{J}_2 = \mathbf{0}_{2\times 3M}$
        $\mathbf{J}_1(:, 7(j-1)+1 : 7j) = \frac{\partial f(\mathbf{p}_j, \mathbf{X}_i)}{\partial \mathbf{p}_j}$
        $\mathbf{J}_2(:, 3(i-1)+1 : 3i) = \frac{\partial f(\mathbf{p}_j, \mathbf{X}_i)}{\partial \mathbf{X}_i}$
        $\mathbf{J_p} = \begin{bmatrix} \mathbf{J}_\mathbf{p}^\mathsf{T} & \mathbf{J}_1^\mathsf{T} \end{bmatrix}^\mathsf{T}$ and $\mathbf{J_X} = \begin{bmatrix} \mathbf{J}_\mathbf{X}^\mathsf{T} & \mathbf{J}_2^\mathsf{T} \end{bmatrix}^\mathsf{T}$
        $\mathbf{d} = \mathbf{d} + \frac{\partial f(\mathbf{p}_j, \mathbf{X}_i)}{\partial \mathbf{X}_i}^\mathsf{T} \frac{\partial f(\mathbf{p}_j, \mathbf{X}_i)}{\partial \mathbf{X}_i}$
        $\mathbf{b} = \begin{bmatrix} \mathbf{b}^\mathsf{T} & \mathbf{u}_{ij}^\mathsf{T} \end{bmatrix}$
        $\mathbf{f} = \begin{bmatrix} \mathbf{f}^\mathsf{T} & \mathbf{x}_{ij}^\mathsf{T} \end{bmatrix}$ where $\mu \begin{bmatrix} \mathbf{x}_{ij} \\ 1 \end{bmatrix} = \mathbf{R}_j \begin{bmatrix} \mathbf{I} & -\mathbf{C}_j \end{bmatrix}$
      end if
    end for
    $\mathbf{d} = \mathbf{d} + \lambda \mathbf{I}$
    $\mathbf{D}_{\text{inv}} = \texttt{blkdiag}(\mathbf{D}_{\text{inv}}, \mathbf{d}^{-1})$
  end for
  $\mathbf{e_p} = \mathbf{J}_\mathbf{p}^\mathsf{T}(\mathbf{b} - \mathbf{f})$
  $\mathbf{e_X} = \mathbf{J}_\mathbf{X}^\mathsf{T}(\mathbf{b} - \mathbf{f})$
  $\mathbf{A} = \mathbf{J}_\mathbf{p}^\mathsf{T}\mathbf{J_p} + \lambda \mathbf{I}$, $\mathbf{B} = \mathbf{J}_\mathbf{p}^\mathsf{T}\mathbf{J_X}$, $\mathbf{D}^{-1} = \mathbf{D}_{\text{inv}}$
  $\Delta\widehat{\mathbf{p}} = (\mathbf{A} - \mathbf{B}\mathbf{D}^{-1}\mathbf{B}^\mathsf{T})^{-1}(\mathbf{e_p} - \mathbf{B}\mathbf{D}^{-1}\mathbf{e_X})$
  Normalize quaternions.
  $\Delta\widehat{\mathbf{X}} = \mathbf{D}^{-1}(\mathbf{e_X} - \mathbf{B}^\mathsf{T}\Delta\widehat{\mathbf{p}})$
end for

BundleAdjustment1D.m

```
for j = 1 : nIters
   Jp = [];    Jx = [];    D_inv = [];    err = [];
   for iPoint = 1 : size(X,1)
      X1 = xx(2*(iPoint-1)+1:2*iPoint); d = zeros(2,2);
      for iC = 1 : length(C)
         c = xp(3*(iC-1)+1:3*(iC-1)+2);
         theta = xp(3*(iC-1)+3);
         R = [sin(theta) -cos(theta); cos(theta) sin(theta)];

         df_dc = JacobianC1D(R, c, X1);
         df_dR = JacobianR1D(R, c, X1)*JacobianQ1D(theta);
         df_dx = JacobianX1D(R, c, X1);

         j1 = zeros(1,3*length(C));  j1(:,3*(iC-1)+1:3*iC) = [df_dc df_dR];
         j2 = zeros(1,2*size(X,1));   j2(:,2*(iPoint-1)+1:2*iPoint) = df_dx;

         Jp = [Jp; j1];       Jx = [Jx; j2];
         d = d + df_dx'*df_dx;

         u = R * [eye(2) -c] * [X1;1];      u = u/u(2);
         u1 = C(iC).m;

         e = [u1(iPoint) - u(1)];
         err = [err; e];
      end
      d = d + lambda*eye(2);       D_inv = blkdiag(D_inv, inv(d));
   end
   ep = Jp' * err;    ex = Jx' * err;
   A = Jp'*Jp + lambda*eye(3*length(C));    B = Jp'*Jx;
   delta_p = inv(A-B*D_inv*B') * (ep-B*D_inv*ex);    delta_x = D_inv * (ex-B'*delta_p);

   xp = xp + delta_p;
   xx = xx + delta_x;
end
```

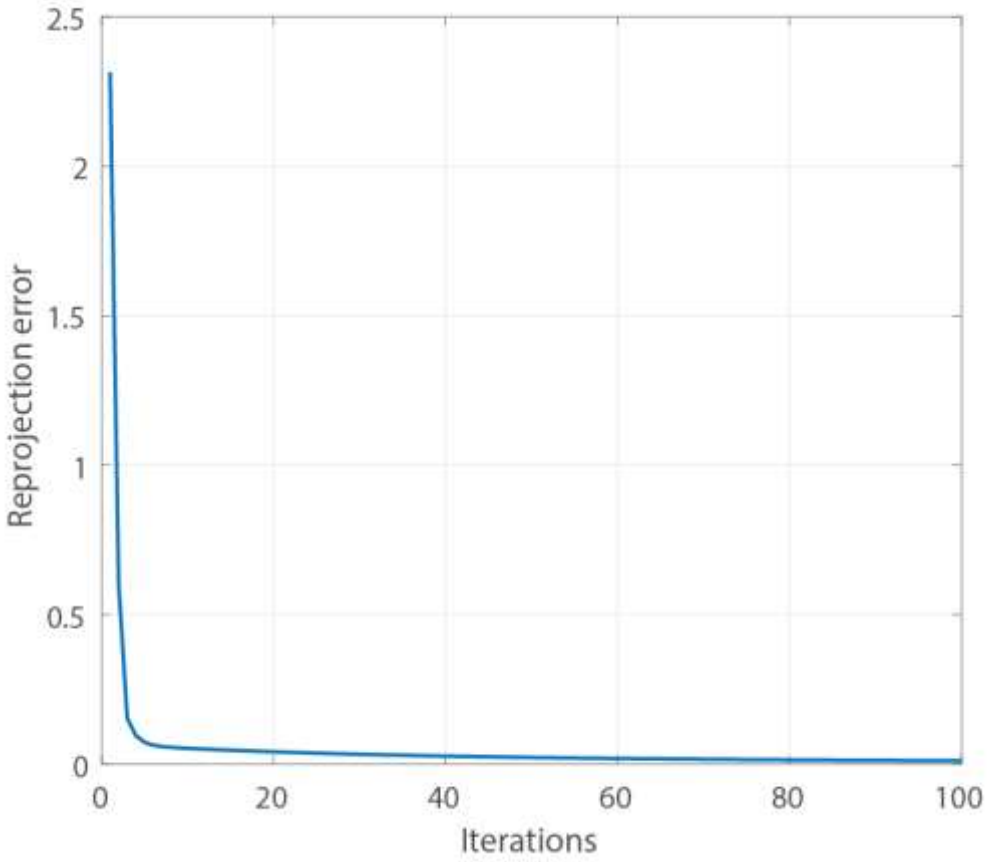
Algorithm 4 Bundle Adjustment

$\widehat{p} = \begin{bmatrix} \mathbf{p}_1^\mathsf{T} & \cdots & \mathbf{p}_I^\mathsf{T} \end{bmatrix}^\mathsf{T}$ and $\widehat{\mathbf{X}} = \begin{bmatrix} \mathbf{X}_1^\mathsf{T} & \cdots & \mathbf{X}_M^\mathsf{T} \end{bmatrix}$
for iter $= 1 :$ nIters **do**
  Empty $\mathbf{J_p}$, $\mathbf{J_X}$, $\mathbf{b}$, $\mathbf{f}$, $\mathbf{D}_{\text{inv}}$.
  for $i = 1 : M$ **do**
    $\mathbf{d} = \mathbf{0}_{3\times3}$
    for $j = 1 : I$ **do**
      if the i^{th} point is visible from the j^{th} image **then**
        $\mathbf{J}_1 = \mathbf{0}_{2\times7I}$ and $\mathbf{J}_2 = \mathbf{0}_{2\times3M}$
        $\mathbf{J}_1(:, 7(j-1)+1 : 7j) = \frac{\partial f(\mathbf{p}_j, \mathbf{X}_i)}{\partial \mathbf{p}_j}$
        $\mathbf{J}_2(:, 3(i-1)+1 : 3i) = \frac{\partial f(\mathbf{p}_j, \mathbf{X}_i)}{\partial \mathbf{X}_i}$
        $\mathbf{J_p} = \begin{bmatrix} \mathbf{J}_\mathbf{p}^\mathsf{T} & \mathbf{J}_1^\mathsf{T} \end{bmatrix}^\mathsf{T}$ and $\mathbf{J_X} = \begin{bmatrix} \mathbf{J}_\mathbf{X}^\mathsf{T} & \mathbf{J}_2^\mathsf{T} \end{bmatrix}^\mathsf{T}$
        $\mathbf{d} = \mathbf{d} + \frac{\partial f(\mathbf{p}_j, \mathbf{X}_i)}{\partial \mathbf{X}_i}^\mathsf{T} \frac{\partial f(\mathbf{p}_j, \mathbf{X}_i)}{\partial \mathbf{X}_i}$
        $\mathbf{b} = \begin{bmatrix} \mathbf{b}^\mathsf{T} & \mathbf{u}_{ij}^\mathsf{T} \end{bmatrix}$
        $\mathbf{f} = \begin{bmatrix} \mathbf{f}^\mathsf{T} & \mathbf{x}_{ij}^\mathsf{T} \end{bmatrix}$ where $\mu \begin{bmatrix} \mathbf{x}_{ij} \\ 1 \end{bmatrix} = \mathbf{R}_j \begin{bmatrix} \mathbf{I} & -\mathbf{C}_j \end{bmatrix}$
      end if
    end for
    $\mathbf{d} = \mathbf{d} + \lambda\mathbf{I}$
    $\mathbf{D}_{\text{inv}} = \texttt{blkdiag}(\mathbf{D}_{\text{inv}}, \mathbf{d}^{-1})$
  end for
  $\mathbf{e_p} = \mathbf{J}_\mathbf{p}^\mathsf{T}(\mathbf{b} - \mathbf{f})$
  $\mathbf{e_X} = \mathbf{J}_\mathbf{X}^\mathsf{T}(\mathbf{b} - \mathbf{f})$
  $\mathbf{A} = \mathbf{J}_\mathbf{p}^\mathsf{T}\mathbf{J_p} + \lambda\mathbf{I}$, $\mathbf{B} = \mathbf{J}_\mathbf{p}^\mathsf{T}\mathbf{J_X}$, $\mathbf{D}^{-1} = \mathbf{D}_{\text{inv}}$
  $\Delta\widehat{\mathbf{p}} = (\mathbf{A} - \mathbf{B}\mathbf{D}^{-1}\mathbf{B}^\mathsf{T})^{-1}(\mathbf{e_p} - \mathbf{B}\mathbf{D}^{-1}\mathbf{e_X})$
  Normalize quaternions.
  $\Delta\widehat{\mathbf{X}} = \mathbf{D}^{-1}(\mathbf{e_X} - \mathbf{B}^\mathsf{T}\Delta\widehat{\mathbf{p}})$
end for

BundleAdjustment1D.m

```
for iC = 1 : length(C)
   C(iC).c = xp(3*(iC-1)+1:3*(iC-1)+2);
   theta = xp(3*(iC-1)+3);
   C(iC).R = [sin(theta) -cos(theta); cos(theta) sin(theta)];
end

for iX = 1 : size(X,1)
   X(iX,:) = xx(2*(iX-1)+1:2*iX);
end
```

Model update



BundleAdjustment1D.m

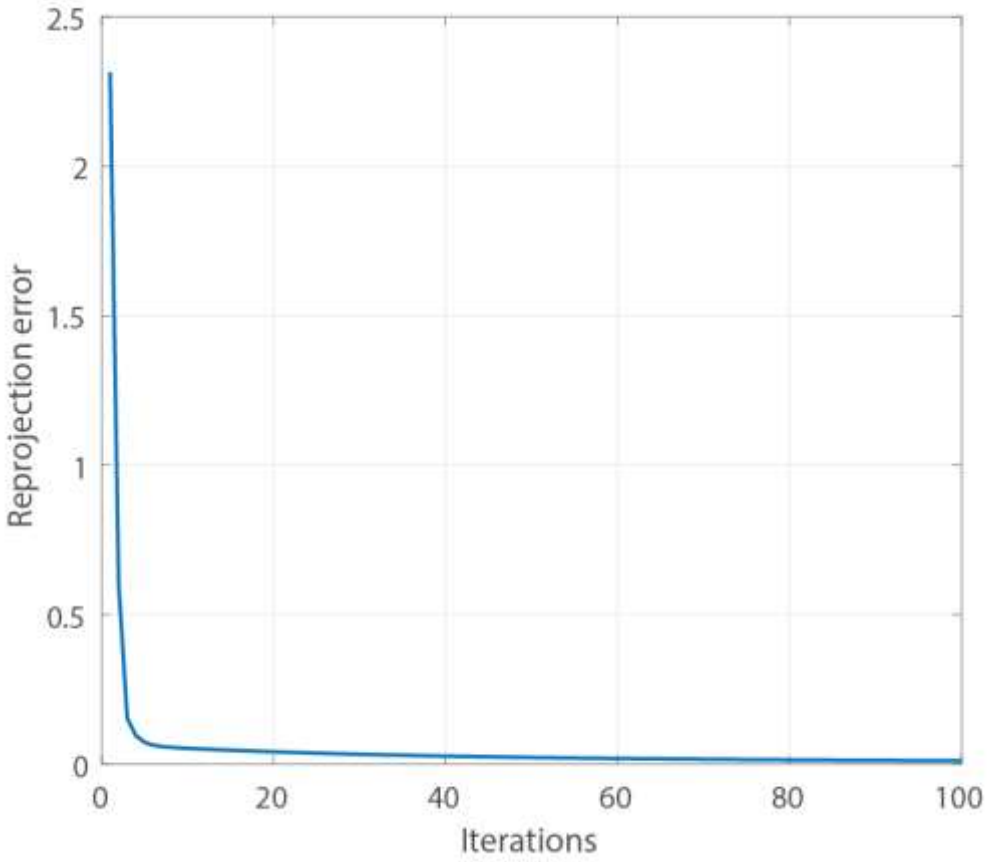
```
for iC = 1 : length(C)
   C(iC).c = xp(3*(iC-1)+1:3*(iC-1)+2);
   theta = xp(3*(iC-1)+3);
   C(iC).R = [sin(theta) -cos(theta); cos(theta) sin(theta)];
end

for iX = 1 : size(X,1)
   X(iX,:) = xx(2*(iX-1)+1:2*iX);
end
```

Model update

mean_delta_X = 0.0265

mean_delta_p = 0.2648

Why camera and point error do not converge to zero while reprojection error converges?



BundleAdjustment1D.m

```
for iC = 1 : length(C)
   C(iC).c = xp(3*(iC-1)+1:3*(iC-1)+2);
   theta = xp(3*(iC-1)+3);
   C(iC).R = [sin(theta) -cos(theta); cos(theta) sin(theta)];
end

for iX = 1 : size(X,1)
   X(iX,:) = xx(2*(iX-1)+1:2*iX);
end
```

Model update

mean_delta_X = 0.0265

mean_delta_p = 0.2648

Why camera and point error do not converge to zero while reprojection error converges?

Because the bundle adjustment is still up to scale and orietation.

SIFT detection
Reprojection

SIFT detection
Reprojection


Measurement
Linear estimate (reproj: 0.199104)
Nonlinear estimate (reproj: 0.119272)


Measurement
Linear estimate (reproj: 0.199104)
Nonlinear estimate (reproj: 0.119272)

Data Capture

- Enough baseline between images
- Share common 3D structure
- Multiple planes

Algorithm 1 Structure from Motion

[Mx, My] = `GetMatches`($\mathcal{I}_1, \cdots, \mathcal{I}_N$)
Normalize coordinate in Mx and My, i.e., $\mathbf{x} = \mathbf{K}^{-1}\mathbf{x}$.
Select two images $\mathcal{I}_{i1}$ and $\mathcal{I}_{i2}$ for the initial pair reconstruction.
[$\mathbf{R}$, $\mathbf{t}$, $\mathbf{X}$] = `CameraPoseEstimation`([Mx(:,$i1$) My(:,$i1$)], [Mx(:,$i2$) My(:,$i2$)])
$\mathcal{P} = \{\mathbf{P}_1, \mathbf{P}_2\}$ where $\mathbf{P}_{i1} = \begin{bmatrix} \mathbf{I}_3 & \mathbf{0} \end{bmatrix}$, $\mathbf{P}_{i2} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \end{bmatrix}$
$\mathcal{R} = \{i1, i2\}$
while $|\mathcal{R}| < N$ **do**
    i = `GetBestFrame`(Mx, My, $\mathcal{R}$);
    [$\mathbf{R}_i$, $\mathbf{t}_i$] = `PnP_RANSAC`([Mx(:,i) My(:,i)], $\mathbf{X}$)
    [$\mathbf{R}_i$, $\mathbf{t}_i$] = `PnP_Nonlinear`($\mathbf{R}_i$ $\mathbf{t}_i$, [Mx(:,i) My(:,i)], $\mathbf{X}$)
    $\mathbf{P}_i = \begin{bmatrix} \mathbf{R}_i & \mathbf{t}_i \end{bmatrix}$
    for f = 1 : $|\mathcal{R}|$ **do**
        $\mathcal{U}$ = `FindUnreconstructedPoints`($\mathbf{X}$, $\mathcal{R}_f$, i, Mx, My)
        for j = 1 : $|\mathcal{U}|$ **do**
            $\mathbf{u}$ = [Mx($\mathcal{U}_j$, i), My($\mathcal{U}_j$, i)] and $\mathbf{v}$ = [Mx($\mathcal{U}_j$, $\mathcal{R}_f$), My($\mathcal{U}_j$, $\mathcal{R}_f$)]
            $\mathbf{x}$ = LinearTriangulation($\mathbf{u}$,P$_i$, $\mathbf{v}$, $\mathbf{P}_{\mathcal{R}_f}$)
            $\mathbf{x}$ = NonlinearTriangulation($\mathbf{x}$, $\mathbf{u}$, $\mathbf{P}_i$, $\mathbf{v}$, $\mathbf{P}_{\mathcal{R}_f}$)
            $\mathbf{X} = \mathbf{X} \cup \mathbf{x}$
        end for
    end for
    $\mathcal{P} = \mathcal{P} \cup \mathbf{P}_i$ and $\mathcal{R} = \mathcal{R} \cup i$.
    [$\mathcal{P}$, $\mathbf{X}$] = `BundleAdjustment`($\mathcal{P}$, $\mathbf{X}$, $\mathcal{R}$, Mx, My)
end while

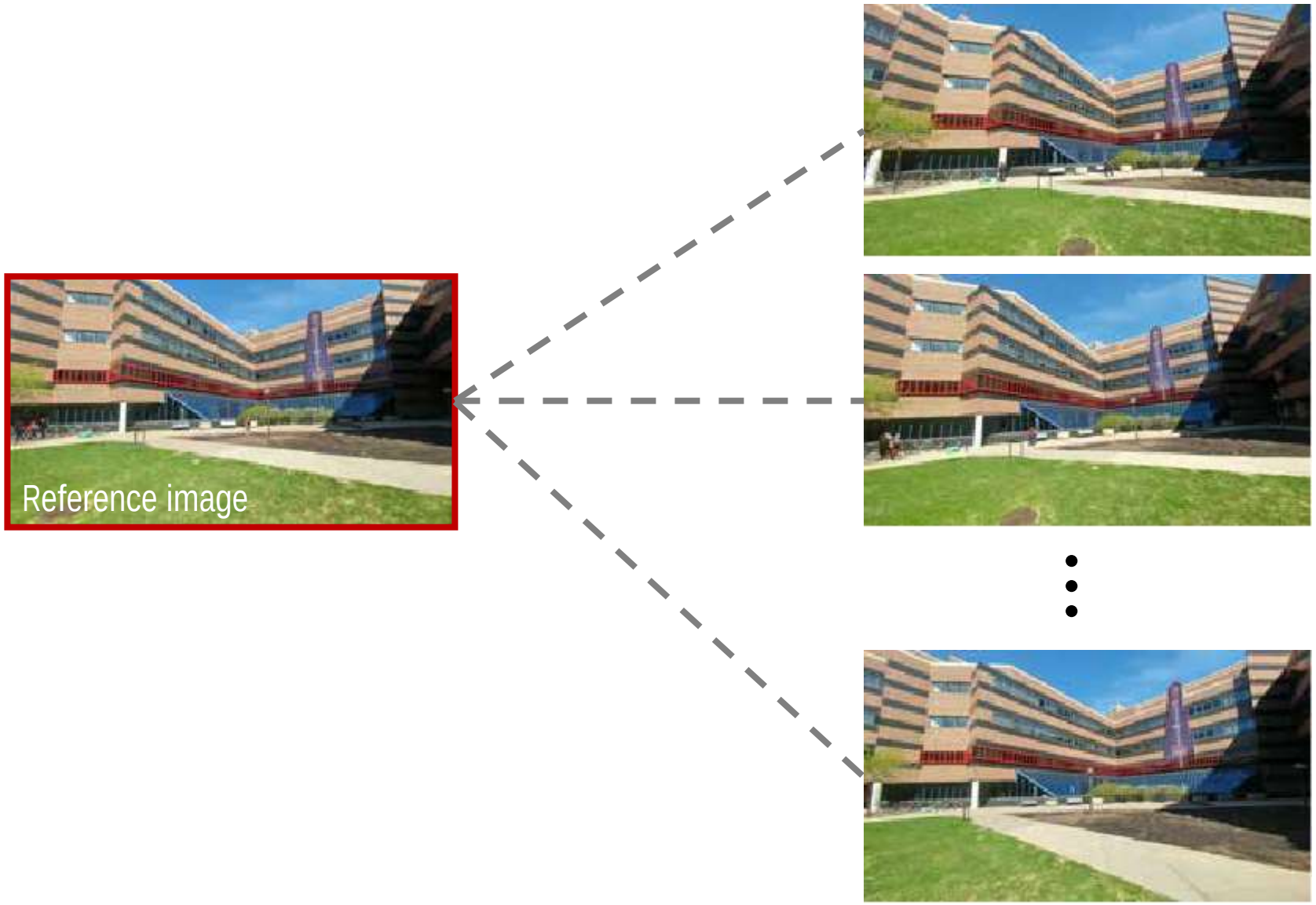
Algorithm 1 Structure from Motion

[Mx, My] = `GetMatches`($\mathcal{I}_1, \cdots, \mathcal{I}_N$)
Normalize coordinate in Mx and My, i.e., $\mathbf{x} = \mathbf{K}^{-1}\mathbf{x}$.
Select two images $\mathcal{I}_{i1}$ and $\mathcal{I}_{i2}$ for the initial pair reconstruction.
[$\mathbf{R}$, $\mathbf{t}$, $\mathbf{X}$] = `CameraPoseEstimation`([Mx(:,$i1$) My(:,$i1$)], [Mx(:,$i2$) My(:,$i2$)])
$\mathcal{P} = \{\mathbf{P}_1, \mathbf{P}_2\}$ where $\mathbf{P}_{i1} = \begin{bmatrix} \mathbf{I}_3 & \mathbf{0} \end{bmatrix}$, $\mathbf{P}_{i2} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \end{bmatrix}$
$\mathcal{R} = \{i1, i2\}$
while $|\mathcal{R}| < N$ **do**
    i = `GetBestFrame`(Mx, My, $\mathcal{R}$);
    [$\mathbf{R}_i$, $\mathbf{t}_i$] = `PnP_RANSAC`([Mx(:,i) My(:,i)], $\mathbf{X}$)
    [$\mathbf{R}_i$, $\mathbf{t}_i$] = `PnP_Nonlinear`($\mathbf{R}_i$ $\mathbf{t}_i$, [Mx(:,i) My(:,i)], $\mathbf{X}$)
    $\mathbf{P}_i = \begin{bmatrix} \mathbf{R}_i & \mathbf{t}_i \end{bmatrix}$
    for f = 1 : $|\mathcal{R}|$ **do**
        $\mathcal{U}$ = `FindUnreconstructedPoints`($\mathbf{X}$, $\mathcal{R}_f$, i, Mx, My)
        for j = 1 : $|\mathcal{U}|$ **do**
            $\mathbf{u}$ = [Mx($\mathcal{U}_j$, i), My($\mathcal{U}_j$, i)] and $\mathbf{v}$ = [Mx($\mathcal{U}_j$, $\mathcal{R}_f$), My($\mathcal{U}_j$, $\mathcal{R}_f$)]
            $\mathbf{x}$ = LinearTriangulation($\mathbf{u}$,P$_i$, $\mathbf{v}$, $\mathbf{P}_{\mathcal{R}_f}$)
            $\mathbf{x}$ = NonlinearTriangulation($\mathbf{x}$, $\mathbf{u}$, $\mathbf{P}_i$, $\mathbf{v}$, $\mathbf{P}_{\mathcal{R}_f}$)
            $\mathbf{X} = \mathbf{X} \cup \mathbf{x}$
        end for
    end for
    $\mathcal{P} = \mathcal{P} \cup \mathbf{P}_i$ and $\mathcal{R} = \mathcal{R} \cup i$.
    [$\mathcal{P}$, $\mathbf{X}$] = `BundleAdjustment`($\mathcal{P}$, $\mathbf{X}$, $\mathcal{R}$, Mx, My)
end while

Reference Image Selection

- An image in the middle may be a good choice of reference image because you can find many matches across all images.

Matching Ref. with other images



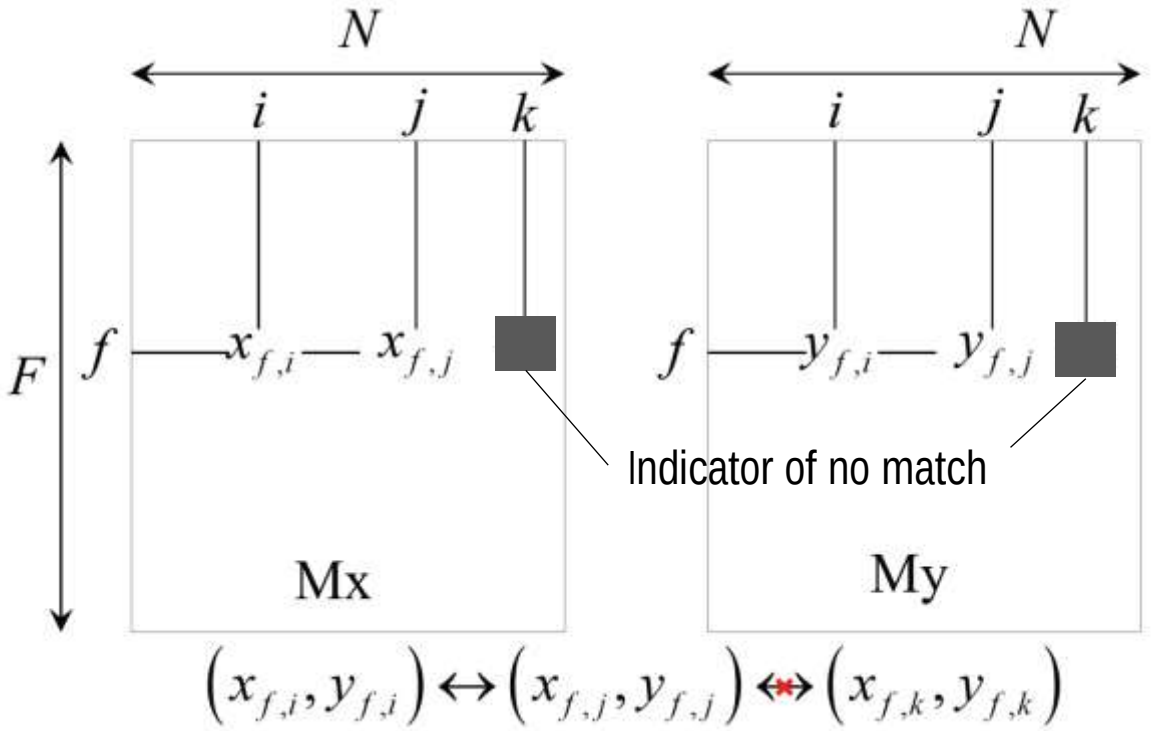
[Mx, My] = GetMatches($\mathcal{I}_1, \cdots, \mathcal{I}_N$)

Mx: F×N matrix storing x coordinate of correspondences

My: F×N matrix storing y coordinate of correspondences

In addition to Mx and My, you may want to have an indicator matrix for visibility.

$\mathcal{I}_i$ $\mathcal{I}_j$



$\hat{u} = K^{-1}u$ $\hat{u}_{f,i}$ $\hat{u}_{f,j}$

Algorithm 1 Structure from Motion

$[\text{Mx}, \text{My}] = \texttt{GetMatches}(\mathcal{I}_1, \cdots, \mathcal{I}_N)$
Normalize coordinate in Mx and My, i.e., $\mathbf{x} = \mathbf{K}^{-1}\mathbf{x}$.
Select two images $\mathcal{I}_{i1}$ and $\mathcal{I}_{i2}$ for the initial pair reconstruction.
$[\mathbf{R}, \mathbf{C}, \mathbf{X}] = \texttt{CameraPoseEstimation}([\text{Mx}(:,i1)\ \text{My}(:,i1)], [\text{Mx}(:,i2)\ \text{My}(:,i2)])$
$\mathcal{P} = \{\mathbf{P}_1, \mathbf{P}_2\}$ where $\mathbf{P}_{i1} = \begin{bmatrix} \mathbf{I}_3 & \mathbf{0} \end{bmatrix}$, $\mathbf{P}_{i2} = \mathbf{R}\begin{bmatrix} \mathbf{I}_3 & -\mathbf{C} \end{bmatrix}$
$\mathcal{R} = \{i1, i2\}$
while $|\mathcal{R}| < N$ **do**
    $i = \texttt{GetBestFrame}(\text{Mx}, \text{My}, \mathcal{R})$;
    $[\mathbf{R}_i, \mathbf{C}_i] = \texttt{PnP\_RANSAC}([\text{Mx}(:,i)\ \text{My}(:,i)], \mathbf{X})$
    $[\mathbf{R}_i, \mathbf{C}_i] = \texttt{PnP\_Nonlinear}(\mathbf{R}_i\ \mathbf{C}_i, [\text{Mx}(:,i)\ \text{My}(:,i)], \mathbf{X})$
    $\mathbf{P}_i = \mathbf{R}_i\begin{bmatrix} \mathbf{I}_3 & -\mathbf{C}_i \end{bmatrix}$
    for f = 1 : $|\mathcal{R}|$ **do**
        $\mathcal{U} = \texttt{FindUnreconstructedPoints}(\mathbf{X}, \mathcal{R}_f, i, \text{Mx}, \text{My})$
        for j = 1 : $|\mathcal{U}|$ **do**
            $\mathbf{u} = [\text{Mx}(\mathcal{U}_j, i), \text{My}(\mathcal{U}_j, i)]$ and $\mathbf{v} = [\text{Mx}(\mathcal{U}_j, \mathcal{R}_f), \text{My}(\mathcal{U}_j, \mathcal{R}_f)]$
            $\mathbf{x} = \texttt{LinearTriangulation}(\mathbf{u}, \mathbf{P}_i, \mathbf{v}, \mathbf{P}_{\mathcal{R}_f})$
            $\mathbf{x} = \texttt{NonlinearTriangulation}(\mathbf{X}, \mathbf{u}, \mathbf{R}_i, \mathbf{C}_i, \mathbf{v}, \mathbf{R}_{\mathcal{R}_f}, \mathbf{C}_{\mathcal{R}_f})$
            $\mathbf{X} = \mathbf{X} \cup \mathbf{x}$
        end for
    end for
    $\mathcal{P} = \mathcal{P} \cup \mathbf{P}_i$ and $\mathcal{R} = \mathcal{R} \cup i$.
    $[\mathcal{P}, \mathbf{X}] = \texttt{BundleAdjustment}(\mathcal{P}, \mathbf{X}, \mathcal{R}, \text{Mx}, \text{My})$
end while

Camera Pose Estimation

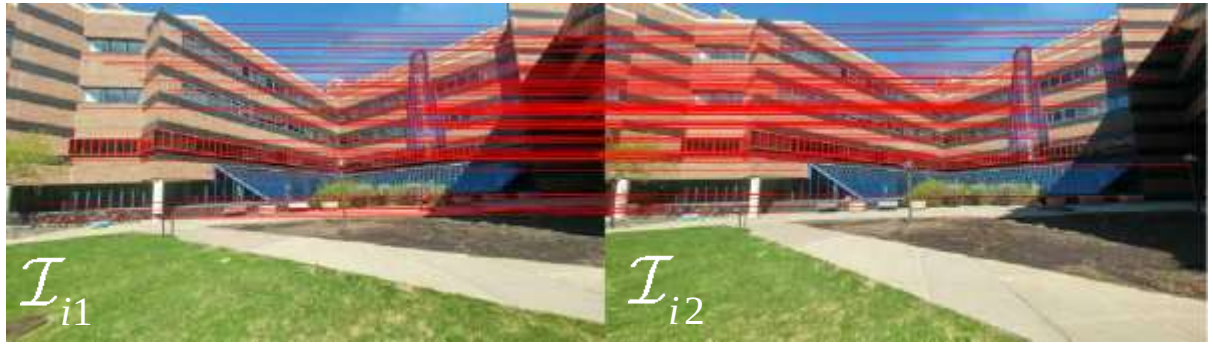
Choose an initial pair of images to compute camera pose

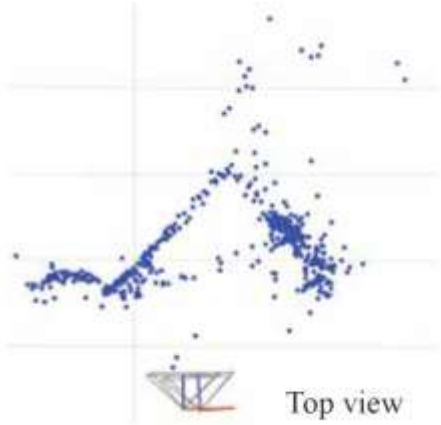
- Have enough number of matches
- Can't be described by a homography, i.e., avoid pure rotation and match through a dominar plane

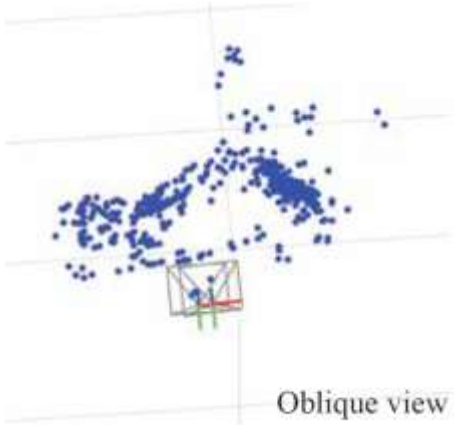
Camera Pose Estimation

$$[\mathbf{R}, \mathbf{t}, \mathbf{X}] = \texttt{CameraPoseEstimation}(\mathbf{u}_1, \mathbf{u}_2)$$

$\mathbf{R}$ and $\mathbf{t}$: the relative transformation of the $i2$ image
$\mathbf{u}_1$ and $\mathbf{u}_2$: 2D-2D correspondences







Algorithm 1 Structure from Motion

$[\text{Mx}, \text{My}] = \texttt{GetMatches}(\mathcal{I}_1, \cdots, \mathcal{I}_N)$
Normalize coordinate in Mx and My, i.e., $\mathbf{x} = \mathbf{K}^{-1}\mathbf{x}$.
Select two images $\mathcal{I}_{i1}$ and $\mathcal{I}_{i2}$ for the initial pair reconstruction.
$[\mathbf{R}, \mathbf{C}, \mathbf{X}] = \texttt{CameraPoseEstimation}([\text{Mx}(:,i1)\ \text{My}(:,i1)], [\text{Mx}(:,i2)\ \text{My}(:,i2)])$
$\mathcal{P} = \{\mathbf{P}_1, \mathbf{P}_2\}$ where $\mathbf{P}_{i1} = \begin{bmatrix} \mathbf{I}_3 & \mathbf{0} \end{bmatrix}$, $\mathbf{P}_{i2} = \mathbf{R}\begin{bmatrix} \mathbf{I}_3 & -\mathbf{C} \end{bmatrix}$
$\mathcal{R} = \{i1, i2\}$
while $|\mathcal{R}| < N$ **do**
    $i = \texttt{GetBestFrame}(\text{Mx}, \text{My}, \mathcal{R})$;
    $[\mathbf{R}_i, \mathbf{C}_i] = \texttt{PnP\_RANSAC}([\text{Mx}(:,i)\ \text{My}(:,i)], \mathbf{X})$
    $[\mathbf{R}_i, \mathbf{C}_i] = \texttt{PnP\_Nonlinear}(\mathbf{R}_i\ \mathbf{C}_i, [\text{Mx}(:,i)\ \text{My}(:,i)], \mathbf{X})$
    $\mathbf{P}_i = \mathbf{R}_i \begin{bmatrix} \mathbf{I}_3 & -\mathbf{C}_i \end{bmatrix}$
    for f = 1 : $|\mathcal{R}|$ **do**
        $\mathcal{U} = \texttt{FindUnreconstructedPoints}(\mathbf{X}, \mathcal{R}_f, i, \text{Mx}, \text{My})$
        for j = 1 : $|\mathcal{U}|$ **do**
            $\mathbf{u} = [\text{Mx}(\mathcal{U}_j, i), \text{My}(\mathcal{U}_j, i)]$ and $\mathbf{v} = [\text{Mx}(\mathcal{U}_j, \mathcal{R}_f), \text{My}(\mathcal{U}_j, \mathcal{R}_f)]$
            $\mathbf{x} = \texttt{LinearTriangulation}(\mathbf{u}, \mathbf{P}_i, \mathbf{v}, \mathbf{P}_{\mathcal{R}_f})$
            $\mathbf{x} = \texttt{NonlinearTriangulation}(\mathbf{X}, \mathbf{u}, \mathbf{R}_i, \mathbf{C}_i, \mathbf{v}, \mathbf{R}_{\mathcal{R}_f}, \mathbf{C}_{\mathcal{R}_f})$
            $\mathbf{X} = \mathbf{X} \cup \mathbf{x}$
        end for
    end for
    $\mathcal{P} = \mathcal{P} \cup \mathbf{P}_i$ and $\mathcal{R} = \mathcal{R} \cup i$.
    $[\mathcal{P}, \mathbf{X}] = \texttt{BundleAdjustment}(\mathcal{P}, \mathbf{X}, \mathcal{R}, \text{Mx}, \text{My})$
end while

Find Next Best Image

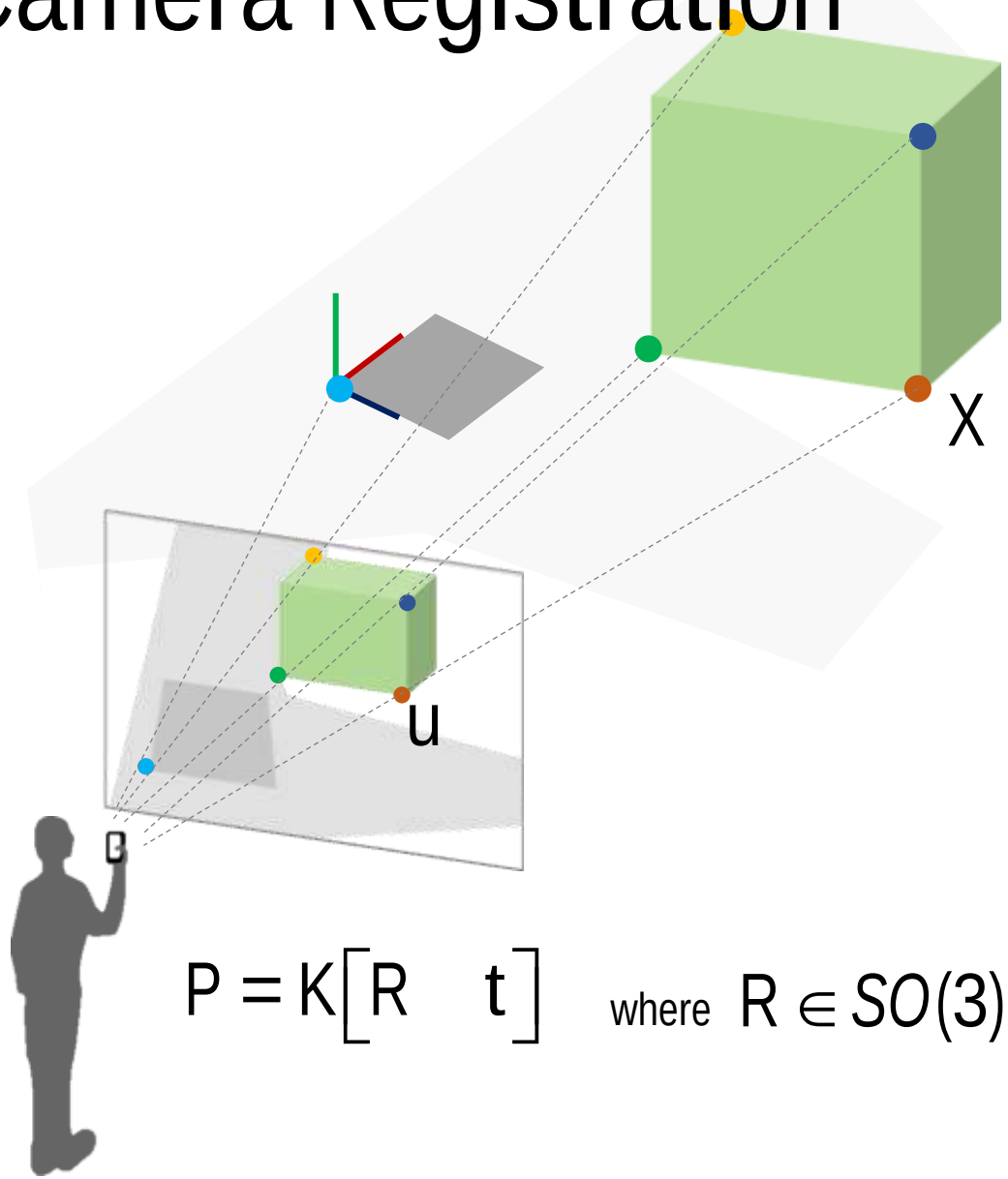
$$i = \texttt{GetBestFrame}(\text{Mx}, \text{My}, \mathcal{R});$$

Reconstructed image index

Find the image that has maximum number of 3D-2D matches

Camera Registration

$$[\mathbf{R}_i, \mathbf{t}_i] = \texttt{PnP\_RANSAC}(\mathbf{u}, \mathbf{X})$$



3D-2D correspondence: $u \leftrightarrow X$

$$u^x = \frac{p_{11}X + p_{12}Y + p_{13}Z + p_{14}}{p_{31}X + p_{32}Y + p_{33}Z + p_{34}} \qquad u^y = \frac{p_{21}X + p_{22}Y + p_{23}Z + p_{24}}{p_{31}X + p_{32}Y + p_{33}Z + p_{34}}$$

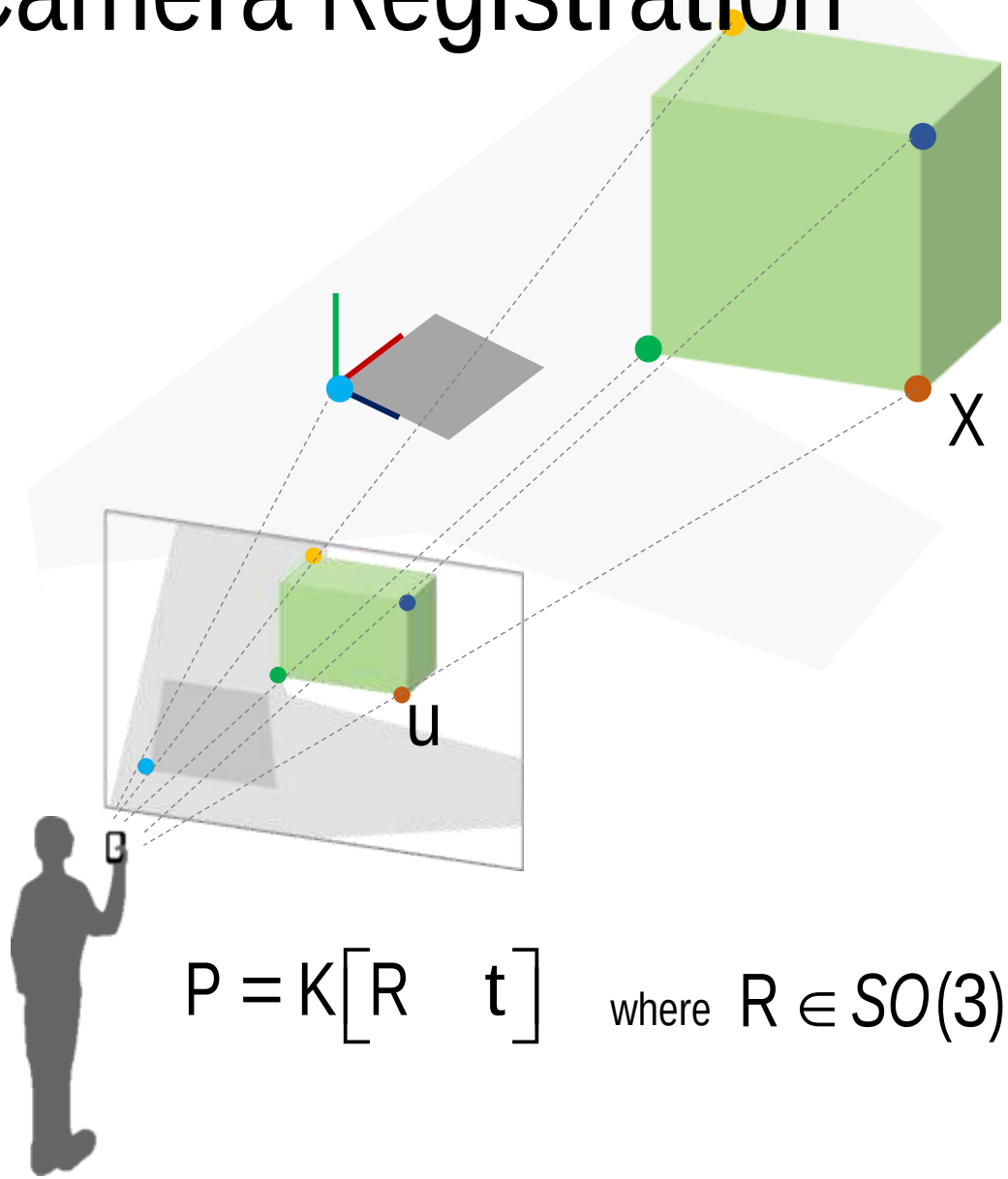
$$\begin{bmatrix} X_1 & Y_1 & Z_1 & 1 & & & & -u_1^x X & -u_1^x Y & -u_1^x Z & -u_1^x \\ & & & & X_1 & Y_1 & Z_1 & 1 & -u_1^y X & -u_1^y Y & -u_1^y Z & -u_1^y \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & & \vdots & \vdots & \vdots & \vdots \\ X_m & Y_m & Z_m & 1 & & & & -u_m^x X & -u_m^x Y & -u_m^x Z & -u_m^x \\ & & & & X_m & Y_m & Z_m & 1 & -u_m^y X & -u_m^y Y & -u_m^y Z & -u_m^y \end{bmatrix} \mathbf{A} \quad \mathbf{x} = \begin{bmatrix} p_{11} \\ p_{12} \\ p_{13} \\ p_{14} \\ p_{21} \\ p_{22} \\ p_{23} \\ p_{24} \\ p_{31} \\ p_{32} \\ p_{33} \\ p_{34} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 0 \end{bmatrix}$$

$2m \times 12$

$$P = K\begin{bmatrix} R & t \end{bmatrix} \quad \text{where} \quad R \in SO(3)$$

Camera Registration

$$[\mathbf{R}_i, \mathbf{t}_i] = \mathrm{PnP\_RANSAC}(\mathbf{u}, \mathbf{X})$$



$$P = K\begin{bmatrix} R & t \end{bmatrix} \quad \text{where} \quad R \in SO(3)$$

If K is given,

$$K\begin{bmatrix} R & t \end{bmatrix} = \gamma\begin{bmatrix} p_1 & p_2 & p_3 & p_4 \end{bmatrix}$$

$$\longrightarrow \gamma R = K^{-1}\begin{bmatrix} p_1 & p_2 & p_3 \end{bmatrix}$$

$$K^{-1}\begin{bmatrix} p_1 & p_2 & p_3 \end{bmatrix} = U\begin{bmatrix} d_{11} & & \\ & d_{22} & \\ & & d_{33} \end{bmatrix}V^T$$

$$\longrightarrow \gamma \approx d_{11}$$

$R = UV^T$: SVD cleanup

$$\longrightarrow t = \frac{K^{-1}p_4}{d_{11}}$$: Translation and scale recovery

Camera Registration

$$[\mathbf{R}_i, \mathbf{t}_i] = \texttt{PnP\_RANSAC}(\mathbf{u}, \mathbf{X})$$

Algorithm 1 PnP RANSAC

$nInliers \leftarrow 0$
for $i = 1 : M$ **do**
    Choose 6 correspondences, $\mathbf{X}_r$ and $\mathbf{w}_r$, randomly from $\mathbf{X}$ and $\mathbf{w}$.
    $[\mathbf{R}_r, \mathbf{t}_r]$ = `LinearPnP`$(\mathbf{X}_r, \mathbf{w}_r, \mathbf{K})$
    Compute the number of inliers, n_r, with respect to $\mathbf{R}_r$, $\mathbf{t}_r$.
    if $n_r > nInliers$ **then**
        $nInliers \leftarrow n_r$
        $\mathbf{R} = \mathbf{R}_r$ and $\mathbf{t} = \mathbf{t}_r$
    end if
end for

Camera Registration

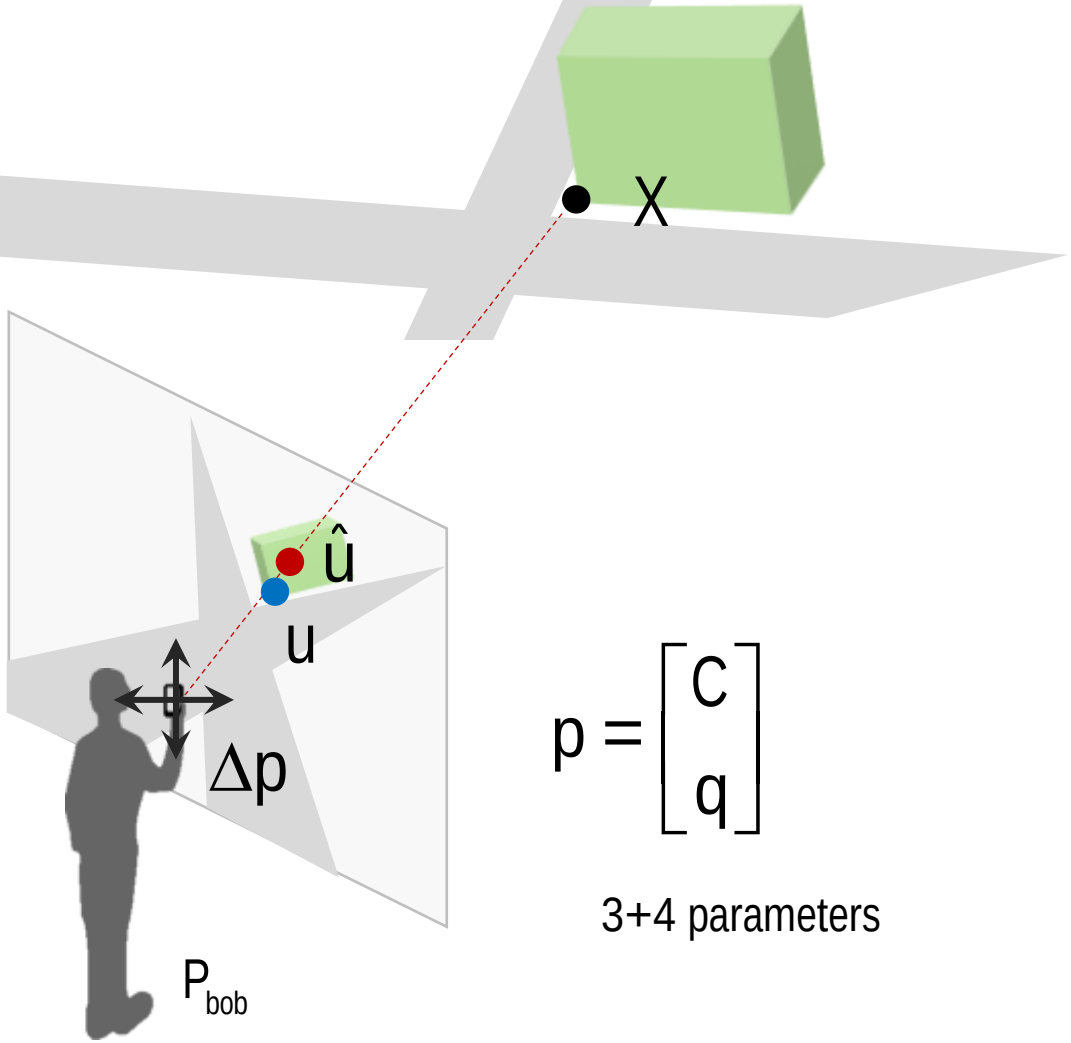
$$[\mathbf{R}_i,\ \mathbf{t}_i] = \texttt{PnP\_Nonlinear}(\mathbf{R}_i\ \mathbf{t}_i,\ \mathbf{u},\ \mathbf{X})$$

Reprojection error:

$$E_{\text{reproj}} = \sum_{j=1}^{n} \left(\frac{u_j}{w_j} - x_j \right)^2 + \left(\frac{v_j}{w_j} - x_j \right)^2$$

Camera Registration

Black: given variables
Red: unknowns



$$p = \begin{bmatrix} C \\ q \end{bmatrix}$$

3+4 parameters

$$E_{geom} = \left(\frac{u}{w} - x\right)^2 + \left(\frac{v}{w} - y\right)^2 \quad \text{where} \quad \begin{bmatrix} u \\ v \\ w \end{bmatrix} = KR(X - C)$$

$$\rightarrow \quad \frac{\partial}{\partial C}\begin{bmatrix} u \\ v \\ w \end{bmatrix} = -KR \qquad \frac{\partial}{\partial q}\begin{bmatrix} u \\ v \\ w \end{bmatrix} = \frac{\partial}{\partial R}\begin{bmatrix} u \\ v \\ w \end{bmatrix}\frac{\partial R}{\partial q}$$

$$\rightarrow \quad \frac{\partial}{\partial p}\begin{bmatrix} u \\ v \\ w \end{bmatrix} = \left[\frac{\partial}{\partial C}\begin{bmatrix} u \\ v \\ w \end{bmatrix} \quad \frac{\partial}{\partial q}\begin{bmatrix} u \\ v \\ w \end{bmatrix}\right]$$

$$f(p) = \begin{bmatrix} \frac{u}{w} \\ \frac{v}{w} \end{bmatrix} \quad \rightarrow \quad \frac{\partial f(p)}{\partial p} = \frac{\partial}{\partial p}\begin{bmatrix} \frac{u}{w} \\ \frac{v}{w} \end{bmatrix} = \begin{bmatrix} \frac{w\frac{\partial u}{\partial p} - u\frac{\partial w}{\partial p}}{w^2} \\ \frac{v\frac{\partial u}{\partial p} - v\frac{\partial w}{\partial p}}{w^2} \end{bmatrix}$$

Camera Registration

$[\mathbf{R}_i, \mathbf{t}_i]$ = PnP_Nonlinear($\mathbf{R}_i$ $\mathbf{t}_i$, $\mathbf{u}$, $\mathbf{X}$)

Algorithm 2 Nonlinear Camera Pose Refinement

1: $\mathbf{p} = [\mathbf{C}^\mathsf{T}\mathbf{q}^\mathsf{T}]^\mathsf{T}$
2: **for** j = 1 : nIters **do**
3: $\mathbf{C} = \mathbf{p}_{1:3}$, $\mathbf{R}$=Quaternion2Rotation($\mathbf{q}$), $\mathbf{q} = \mathbf{p}_{4:7}$
4: Build camera pose Jacobian for all points, $\frac{\partial f(\mathbf{p})_j}{\partial \mathbf{p}} = \left[\frac{\partial f(\mathbf{p})_j}{\partial \mathbf{C}} \quad \frac{\partial f(\mathbf{p})_j}{\partial \mathbf{q}} \right]$.
5: Compute $f(\mathbf{p})$.
6: $\Delta\mathbf{p} = \left(\frac{\partial f(\mathbf{p})}{\partial \mathbf{p}}^\mathsf{T} \frac{\partial f(\mathbf{p})}{\partial \mathbf{p}} + \lambda \mathbf{I} \right)^{-1} \frac{\partial f(\mathbf{p})}{\partial \mathbf{p}}^\mathsf{T} (\mathbf{b} - f(\mathbf{p}))$ using Equation (2).
7: $\mathbf{p} = \mathbf{p} + \Delta\mathbf{p}$
8: Normalize the quaternion scale, $\mathbf{p}_{4:7} = \mathbf{p}_{4:7} / \|\mathbf{p}_{4:7}\|$.
9: **end for**

$$\Delta \mathsf{p} = \left(\frac{\partial f(\mathsf{p})^\mathsf{T}}{\partial \mathsf{p}} \frac{\partial f(\mathsf{p})}{\partial \mathsf{p}} + \lambda \mathsf{I} \right)^{-1} \frac{\partial f(\mathsf{p})^\mathsf{T}}{\partial \mathsf{p}} \left(\mathsf{b} - f(\mathsf{p}) \right)$$

Camera Registration

(a) Reprojection error

(b) Close up

Figure 4: Nonlinear refinement reduces the reprojection error (0.19→0.11).

Algorithm 1 Structure from Motion

[Mx, My] = `GetMatches`($\mathcal{I}_1, \cdots, \mathcal{I}_N$)
Normalize coordinate in Mx and My, i.e., $\mathbf{x} = \mathbf{K}^{-1}\mathbf{x}$.
Select two images $\mathcal{I}_{i1}$ and $\mathcal{I}_{i2}$ for the initial pair reconstruction.
[$\mathbf{R}$, $\mathbf{C}$, $\mathbf{X}$] = `CameraPoseEstimation`([Mx(:,$i1$) My(:,$i1$)], [Mx(:,$i2$) My(:,$i2$)])
$\mathcal{P} = \{\mathbf{P}_1, \mathbf{P}_2\}$ where $\mathbf{P}_{i1} = \begin{bmatrix} \mathbf{I}_3 & \mathbf{0} \end{bmatrix}$, $\mathbf{P}_{i2} = \mathbf{R}\begin{bmatrix} \mathbf{I}_3 & -\mathbf{C} \end{bmatrix}$
$\mathcal{R} = \{i1, i2\}$
while $|\mathcal{R}| < N$ **do**
    i = `GetBestFrame`(Mx, My, $\mathcal{R}$);
    [$\mathbf{R}_i$, $\mathbf{C}_i$] = `PnP_RANSAC`([Mx(:,i) My(:,i)], $\mathbf{X}$)
    [$\mathbf{R}_i$, $\mathbf{C}_i$] = `PnP_Nonlinear`($\mathbf{R}_i$ $\mathbf{C}_i$, [Mx(:,i) My(:,i)], $\mathbf{X}$)
    $\mathbf{P}_i = \mathbf{R}_i \begin{bmatrix} \mathbf{I}_3 & -\mathbf{C}_i \end{bmatrix}$
    for f = 1 : $|\mathcal{R}|$ **do**
        $\mathcal{U}$ = `FindUnreconstructedPoints`($\mathbf{X}$, $\mathcal{R}_f$, i, Mx, My)
        for j = 1 : $|\mathcal{U}|$ **do**
            $\mathbf{u}$ = [Mx($\mathcal{U}_j$, i), My($\mathcal{U}_j$, i)] and $\mathbf{v}$ = [Mx($\mathcal{U}_j$, $\mathcal{R}_f$), My($\mathcal{U}_j$, $\mathcal{R}_f$)]
            $\mathbf{x}$ = `LinearTriangulation`($\mathbf{u}$, $\mathbf{P}_i$, $\mathbf{v}$, $\mathbf{P}_{\mathcal{R}_f}$)
            $\mathbf{x}$ = `NonlinearTriangulation`($\mathbf{X}$, $\mathbf{u}$, $\mathbf{R}_i$, $\mathbf{C}_i$, $\mathbf{v}$, $\mathbf{R}_{\mathcal{R}_f}$, $\mathbf{C}_{\mathcal{R}_f}$)
            $\mathbf{X} = \mathbf{X} \cup \mathbf{x}$
        end for
    end for
    $\mathcal{P} = \mathcal{P} \cup \mathbf{P}_i$ and $\mathcal{R} = \mathcal{R} \cup i$.
    [$\mathcal{P}$, $\mathbf{X}$] = `BundleAdjustment`($\mathcal{P}$, $\mathbf{X}$, $\mathcal{R}$, Mx, My)
end while

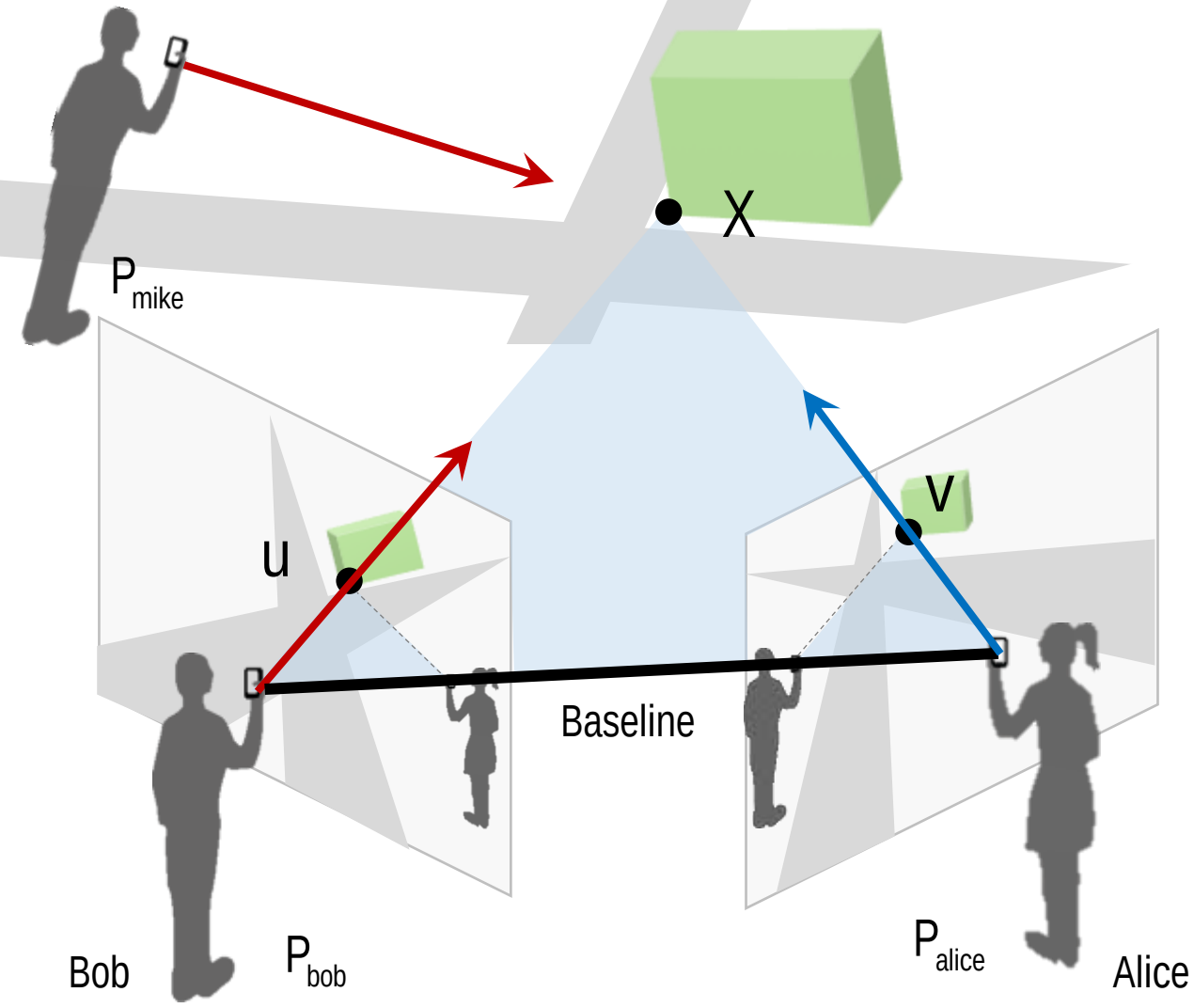
Triangulation of Unreconstructed Points

$$\mathcal{U} = \texttt{FindUnreconstructedPoints}(\mathbf{X}, \mathcal{R}_f, i, \mathrm{Mx}, \mathrm{My})$$

This is a point that is not reconstructed yet while it matches with reconstructed images.

Point Triangulation

$$\mathbf{x} = \mathrm{LinearTriangulation}(\mathbf{u}, P_i, \mathbf{v}, \mathbf{P}_{\mathcal{R}_f})$$



$$\lambda_1 \begin{bmatrix} u \\ 1 \end{bmatrix} = P_{bob} \begin{bmatrix} X \\ 1 \end{bmatrix}$$

Two 3D vectors are parallel.

$$\rightarrow \begin{bmatrix} u \\ 1 \end{bmatrix} \times P_{bob} \begin{bmatrix} X \\ 1 \end{bmatrix} = 0$$

$$\rightarrow \begin{bmatrix} u \\ 1 \end{bmatrix}_{\times} P_{bob} \begin{bmatrix} X \\ 1 \end{bmatrix} = 0$$

$$\begin{bmatrix} v \\ 1 \end{bmatrix}_{\times} P_{alice}$$

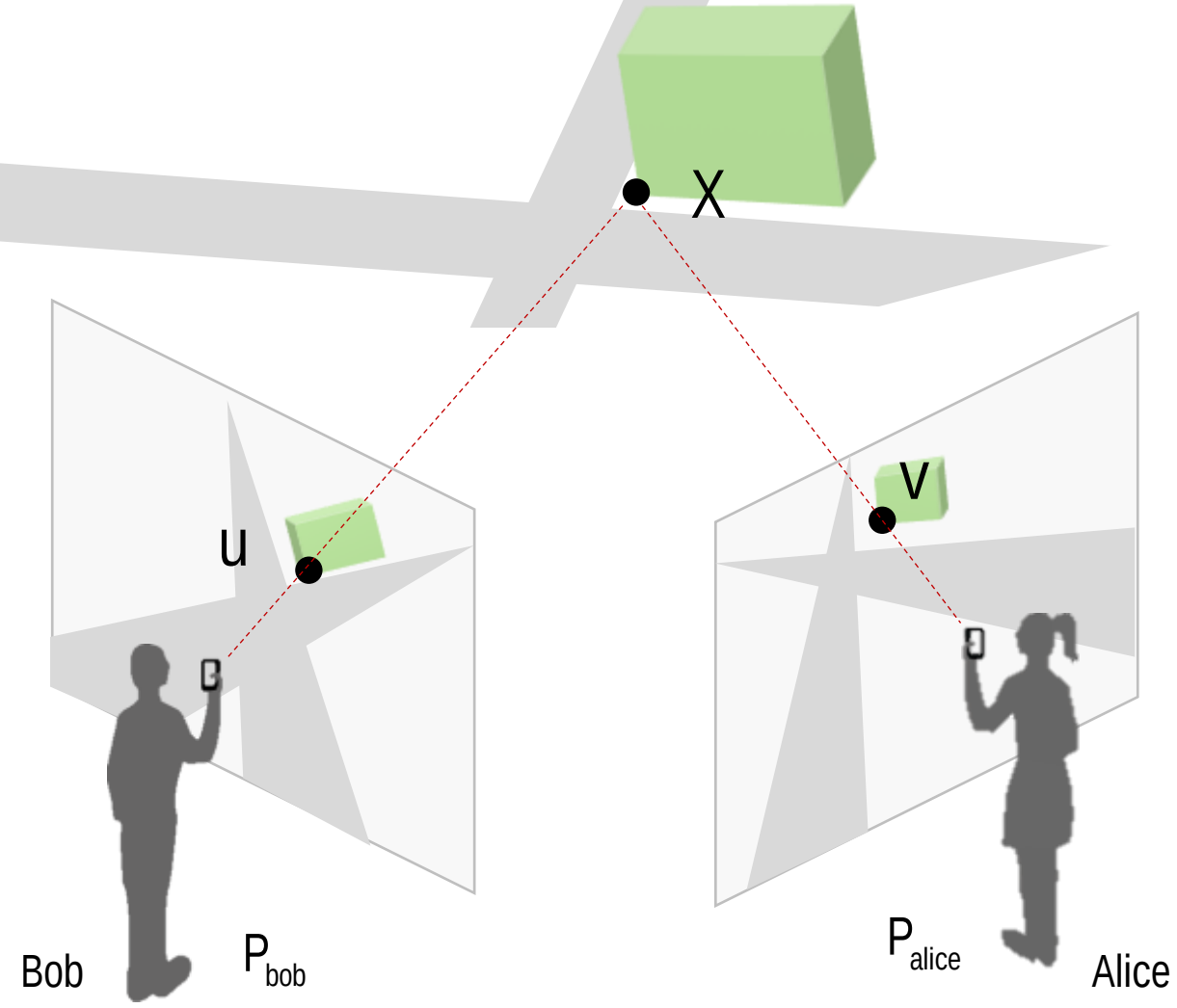
$$\begin{bmatrix} w \\ 1 \end{bmatrix}_{\times} P_{mike}$$

: Knowns

: Unknowns

Nonlinear Point Triangulation

Black: given variables
Red: unknowns



$$E_{geom} = \left\| \begin{bmatrix} u_{bob}/w_{bob} \\ v_{bob}/w_{bob} \\ u_{alice}/w_{alice} \\ v_{alice}/w_{alice} \end{bmatrix} - \begin{bmatrix} x_{bob} \\ y_{bob} \\ x_{alice} \\ y_{alice} \end{bmatrix} \right\|^2$$

$$\frac{\partial f(X)}{\partial X} = \frac{\partial}{\partial X} \begin{bmatrix} \frac{u}{w} \\ \frac{v}{w} \end{bmatrix} = \begin{bmatrix} \frac{w\frac{\partial u}{\partial X} - u\frac{\partial w}{\partial X}}{w^2} \\ \frac{v\frac{\partial u}{\partial X} - v\frac{\partial w}{\partial X}}{w^2} \end{bmatrix}$$

$$\Delta x = \left(\frac{\partial f(x)^T}{\partial x} \frac{\partial f(x)}{\partial x} \right)^{-1} \frac{\partial f(x)^T}{\partial x} \left(b - f(x) \right)$$

Nonlinear Point Triangulation

Algorithm 3 Nonlinear Point Refinement

1: $\mathbf{b} = \begin{bmatrix} \mathbf{u}_1^\mathsf{T} & \mathbf{u}_2^\mathsf{T} \end{bmatrix}^\mathsf{T}$
2: **for** j = 1 : nIters **do**
3: Build point Jacobian, $\frac{\partial f(\mathbf{X})_j}{\partial \mathbf{X}}$.
4: Compute $f(\mathbf{X})$.
5: $\Delta\mathbf{X} = \left(\frac{\partial f(\mathbf{X})}{\partial \mathbf{X}}^\mathsf{T} \frac{\partial f(\mathbf{X})}{\partial \mathbf{X}} + \lambda\mathbf{I} \right)^{-1} \frac{\partial f(\mathbf{X})}{\partial \mathbf{X}}^\mathsf{T} (\mathbf{b} - f(\mathbf{X}))$
6: $\mathbf{X} = \mathbf{X} + \Delta\mathbf{X}$
7: **end for**

$$\frac{\partial f(X)}{\partial X} = \frac{\partial}{\partial X}\begin{bmatrix} \frac{u}{w} \\ \frac{v}{w} \end{bmatrix} = \begin{bmatrix} \frac{w\frac{\partial u}{\partial X} - u\frac{\partial w}{\partial X}}{w^2} \\ \frac{v\frac{\partial u}{\partial X} - v\frac{\partial w}{\partial X}}{w^2} \end{bmatrix}$$

$$\Delta x = \left(\frac{\partial f(x)^\mathsf{T}}{\partial x} \frac{\partial f(x)}{\partial x} \right)^{-1} \frac{\partial f(x)^\mathsf{T}}{\partial x} \left(b - f(x) \right)$$

Damping factor (Levenberg-Marquardt algorithm)

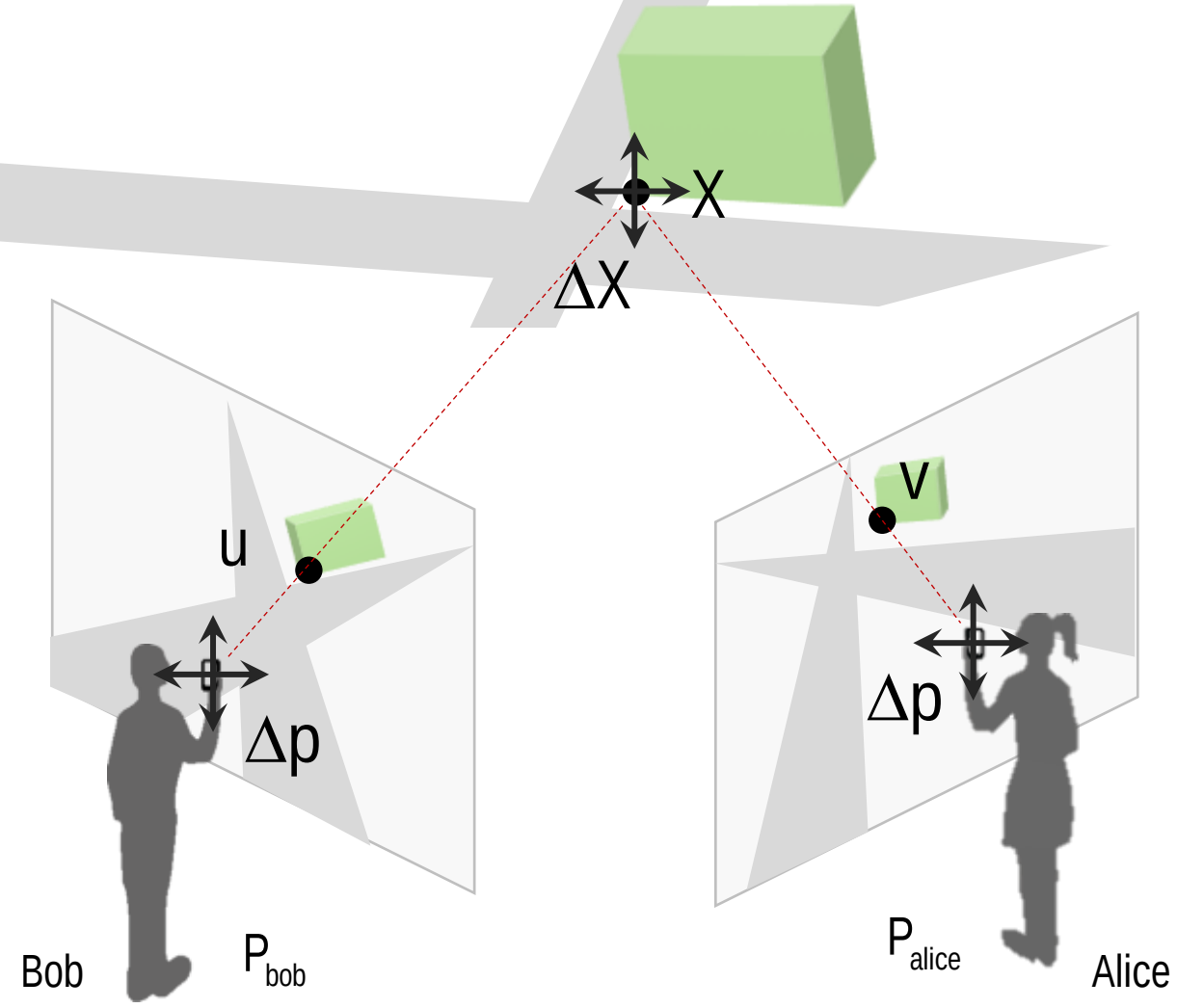
$$\Delta x = \left(\frac{\partial f(x)^\mathsf{T}}{\partial x} \frac{\partial f(x)}{\partial x} + \lambda I \right)^{-1} \frac{\partial f(x)^\mathsf{T}}{\partial x} \left(b - f(x) \right)$$

Algorithm 1 Structure from Motion

$[\text{Mx}, \text{My}] = \texttt{GetMatches}(\mathcal{I}_1, \cdots, \mathcal{I}_N)$
Normalize coordinate in Mx and My, i.e., $\mathbf{x} = \mathbf{K}^{-1}\mathbf{x}$.
Select two images $\mathcal{I}_{i1}$ and $\mathcal{I}_{i2}$ for the initial pair reconstruction.
$[\mathbf{R}, \mathbf{C}, \mathbf{X}] = \texttt{CameraPoseEstimation}([\text{Mx}(:,i1)\ \text{My}(:,i1)], [\text{Mx}(:,i2)\ \text{My}(:,i2)])$
$\mathcal{P} = \{\mathbf{P}_1, \mathbf{P}_2\}$ where $\mathbf{P}_{i1} = \begin{bmatrix} \mathbf{I}_3 & \mathbf{0} \end{bmatrix}$, $\mathbf{P}_{i2} = \mathbf{R}\begin{bmatrix} \mathbf{I}_3 & -\mathbf{C} \end{bmatrix}$
$\mathcal{R} = \{i1, i2\}$
while $|\mathcal{R}| < N$ **do**
    $i = \texttt{GetBestFrame}(\text{Mx}, \text{My}, \mathcal{R})$;
    $[\mathbf{R}_i, \mathbf{C}_i] = \texttt{PnP\_RANSAC}([\text{Mx}(:,i)\ \text{My}(:,i)], \mathbf{X})$
    $[\mathbf{R}_i, \mathbf{C}_i] = \texttt{PnP\_Nonlinear}(\mathbf{R}_i\ \mathbf{C}_i, [\text{Mx}(:,i)\ \text{My}(:,i)], \mathbf{X})$
    $\mathbf{P}_i = \mathbf{R}_i\begin{bmatrix} \mathbf{I}_3 & -\mathbf{C}_i \end{bmatrix}$
    for f = 1 : $|\mathcal{R}|$ **do**
        $\mathcal{U} = \texttt{FindUnreconstructedPoints}(\mathbf{X}, \mathcal{R}_f, i, \text{Mx}, \text{My})$
        for j = 1 : $|\mathcal{U}|$ **do**
            $\mathbf{u} = [\text{Mx}(\mathcal{U}_j, i), \text{My}(\mathcal{U}_j, i)]$ and $\mathbf{v} = [\text{Mx}(\mathcal{U}_j, \mathcal{R}_f), \text{My}(\mathcal{U}_j, \mathcal{R}_f)]$
            $\mathbf{x} = \texttt{LinearTriangulation}(\mathbf{u}, \mathbf{P}_i, \mathbf{v}, \mathbf{P}_{\mathcal{R}_f})$
            $\mathbf{x} = \texttt{NonlinearTriangulation}(\mathbf{X}, \mathbf{u}, \mathbf{R}_i, \mathbf{C}_i, \mathbf{v}, \mathbf{R}_{\mathcal{R}_f}, \mathbf{C}_{\mathcal{R}_f})$
            $\mathbf{X} = \mathbf{X} \cup \mathbf{x}$
        end for
    end for
    $\mathcal{P} = \mathcal{P} \cup \mathbf{P}_i$ and $\mathcal{R} = \mathcal{R} \cup i$.
    $[\mathcal{P}, \mathbf{X}] = \texttt{BundleAdjustment}(\mathcal{P}, \mathbf{X}, \mathcal{R}, \text{Mx}, \text{My})$
end while

Bundle Adjustment

Black: given variables
Red: unknowns



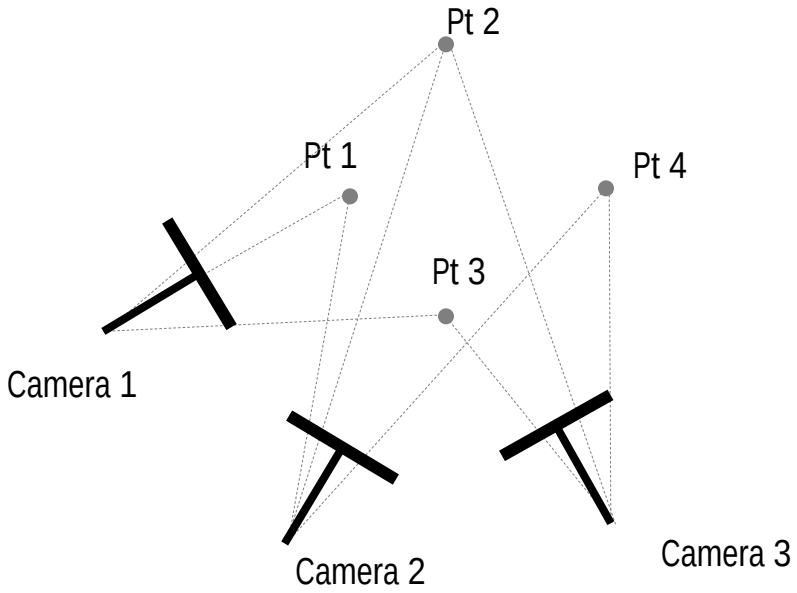
$$E_{\text{geom}} = \left\| \hat{u} - u \right\|^2$$

$$= \left(\frac{P_1 X}{P_3 X} - x \right)^2 + \left(\frac{P_2 X}{P_3 X} - y \right)^2$$

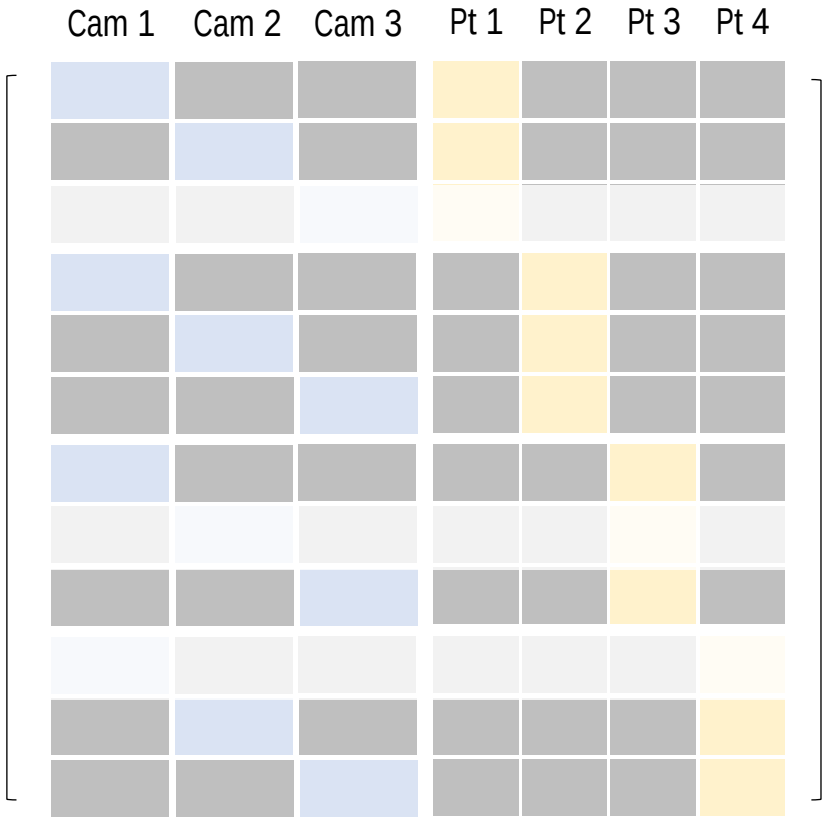
$$f(p, X) = \begin{bmatrix} \frac{u}{w} \\ \frac{v}{w} \end{bmatrix} \longrightarrow \frac{\partial f(p, X)}{\partial p} = \frac{\partial}{\partial p} \begin{bmatrix} \frac{u}{w} \\ \frac{v}{w} \end{bmatrix} = \begin{bmatrix} \frac{w \frac{\partial u}{\partial p} - u \frac{\partial w}{\partial p}}{w^2} \\ \frac{v \frac{\partial u}{\partial p} - v \frac{\partial w}{\partial p}}{w^2} \end{bmatrix}$$

$$\longrightarrow \frac{\partial f(p, X)}{\partial X} = \frac{\partial}{\partial X} \begin{bmatrix} \frac{u}{w} \\ \frac{v}{w} \end{bmatrix} = \begin{bmatrix} \frac{w \frac{\partial u}{\partial X} - u \frac{\partial w}{\partial X}}{w^2} \\ \frac{v \frac{\partial u}{\partial X} - v \frac{\partial w}{\partial X}}{w^2} \end{bmatrix}$$

Visibility Reasoning



J =



# of unknowns: 3x7+4x3

# of projections: 9 (not all points are visible from cameras)

Jacobian Sparsity

$$J^T J = \begin{bmatrix} D & \\ & \end{bmatrix}, \quad D = \begin{bmatrix} d_1 & & \\ & \ddots & \\ & & d_M \end{bmatrix} \longrightarrow D^{-1} = \begin{bmatrix} d_1^{-1} & & \\ & \ddots & \\ & & d_M^{-1} \end{bmatrix}$$

Inversion of block diagonal matrix can be efficiently computed.

$$J^T J = \begin{bmatrix} J_p^T J_p & J_p^T J_X \\ J_X^T J_p & J_X^T J_X \end{bmatrix} = \begin{bmatrix} A & B \\ B^T & D \end{bmatrix}$$

Normal equation:

$$J^T J \begin{bmatrix} \Delta p \\ \Delta X \end{bmatrix} = J^T \left(b - f(X) \right)$$

$$\longrightarrow \begin{bmatrix} A & B \\ B^T & D \end{bmatrix} \begin{bmatrix} \Delta p \\ \Delta X \end{bmatrix} = \begin{bmatrix} e_p \\ e_X \end{bmatrix}$$

$$\longrightarrow \begin{bmatrix} I & -BD^{-1} \\ 0 & I \end{bmatrix} \begin{bmatrix} A & B \\ B^T & D \end{bmatrix} \begin{bmatrix} \Delta p \\ \Delta X \end{bmatrix} = \begin{bmatrix} I & -BD^{-1} \\ 0 & I \end{bmatrix} \begin{bmatrix} e_p \\ e_X \end{bmatrix}$$

$$\longrightarrow \begin{bmatrix} A - BD^{-1}B^T & 0 \\ B^T & D \end{bmatrix} \begin{bmatrix} \Delta p \\ \Delta X \end{bmatrix} = \begin{bmatrix} e_p - BD^{-1}e_X \\ e_X \end{bmatrix}$$

$$\longrightarrow \Delta p = \left(A - BD^{-1}B^T \right)^{-1} \left(e_p - BD^{-1}e_X \right)$$

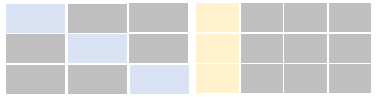
$$\Delta X = D^{-1} \left(e_X - B^T \Delta p \right)$$

Note: $A - BD^{-1}B^T$ is Schur complement of D

Algorithm 4 Bundle Adjustment

$\hat{p} = \begin{bmatrix} \mathbf{p}_1^\mathsf{T} & \cdots & \mathbf{p}_I^\mathsf{T} \end{bmatrix}^\mathsf{T}$ and $\hat{\mathbf{X}} = \begin{bmatrix} \mathbf{X}_1^\mathsf{T} & \cdots & \mathbf{X}_M^\mathsf{T} \end{bmatrix}$
for iter $= 1 :$ nIters **do**
  Empty $\mathbf{J_p}$, $\mathbf{J_X}$, $\mathbf{b}$, $\mathbf{f}$, $\mathbf{D}_{\text{inv}}$.
  for $i = 1 : M$ **do** ← # of points
    $\mathbf{d} = \mathbf{0}_{3\times3}$
    for $j = 1 : I$ **do** ← # of images
      if the i^{th} point is visible from the j^{th} image **then** ← if visible
        $\mathbf{J}_1 = \mathbf{0}_{2\times7I}$ and $\mathbf{J}_2 = \mathbf{0}_{2\times3M}$
        $\mathbf{J}_1(:, 7(j-1)+1 : 7j) = \frac{\partial f(\mathbf{p}_j, \mathbf{X}_i)}{\partial \mathbf{p}_j}$ ← J_p
        $\mathbf{J}_2(:, 3(i-1)+1 : 3i) = \frac{\partial f(\mathbf{p}_j, \mathbf{X}_i)}{\partial \mathbf{X}_i}$ ← J_X
        $\mathbf{J_p} = \begin{bmatrix} \mathbf{J}_\mathbf{p}^\mathsf{T} & \mathbf{J}_1^\mathsf{T} \end{bmatrix}^\mathsf{T}$ and $\mathbf{J_X} = \begin{bmatrix} \mathbf{J}_\mathbf{X}^\mathsf{T} & \mathbf{J}_2^\mathsf{T} \end{bmatrix}^\mathsf{T}$ ←
        $\mathbf{d} = \mathbf{d} + \frac{\partial f(\mathbf{p}_j, \mathbf{X}_i)}{\partial \mathbf{X}_i}^\mathsf{T} \frac{\partial f(\mathbf{p}_j, \mathbf{X}_i)}{\partial \mathbf{X}_i}$ ← d
        $\mathbf{b} = \begin{bmatrix} \mathbf{b}^\mathsf{T} & \mathbf{u}_{ij}^\mathsf{T} \end{bmatrix}$
        $\mathbf{f} = \begin{bmatrix} \mathbf{f}^\mathsf{T} & \mathbf{x}_{ij}^\mathsf{T} \end{bmatrix}$ where $\mu \begin{bmatrix} \mathbf{x}_{ij} \\ 1 \end{bmatrix} = \mathbf{R}_j \begin{bmatrix} \mathbf{I} & -\mathbf{C}_j \end{bmatrix}$ ← $(b - f(p))$
      end if
    end for
    $\mathbf{d} = \mathbf{d} + \lambda \mathbf{I}$ ← $d = d + \lambda I$
    $\mathbf{D}_{\text{inv}} = \texttt{blkdiag}(\mathbf{D}_{\text{inv}}, \mathbf{d}^{-1})$
  end for
  $\mathbf{e_p} = \mathbf{J}_\mathbf{p}^\mathsf{T}(\mathbf{b} - \mathbf{f})$ ← $\begin{bmatrix} J_p^T \\ J_X^T \end{bmatrix}(b - f(X)) = \begin{bmatrix} e_p \\ e_X \end{bmatrix}$
  $\mathbf{e_X} = \mathbf{J}_\mathbf{X}^\mathsf{T}(\mathbf{b} - \mathbf{f})$
  $\mathbf{A} = \mathbf{J}_\mathbf{p}^\mathsf{T}\mathbf{J_p} + \lambda\mathbf{I}$, $\mathbf{B} = \mathbf{J}_\mathbf{p}^\mathsf{T}\mathbf{J_X}$, $\mathbf{D}^{-1} = \mathbf{D}_{\text{inv}}$ ← $\begin{bmatrix} A & B \\ B^T & D \end{bmatrix} = \begin{bmatrix} J_p^T J_p + \mu I & J_p^T J_X \\ J_X^T J_p & J_X^T J_X + \mu I \end{bmatrix}$
  $\Delta\hat{\mathbf{p}} = (\mathbf{A} - \mathbf{B}\mathbf{D}^{-1}\mathbf{B}^\mathsf{T})^{-1}(\mathbf{e_p} - \mathbf{B}\mathbf{D}^{-1}\mathbf{e_X})$ ← $\Delta p = (A - BD^{-1}B^T)^{-1}(e_p - BD^{-1}e_X)$
  Normalize quaternions.
  $\Delta\hat{\mathbf{X}} = \mathbf{D}^{-1}(\mathbf{e_X} - \mathbf{B}^\mathsf{T}\Delta\hat{\mathbf{p}})$ ← $\Delta X = D^{-1}(e_X - B^T \Delta p)$
end for

$J = \begin{bmatrix} J_p & J_X \end{bmatrix}$ $(b - f(p))$ $D^{-1} = \begin{bmatrix} d_1^{-1} & & \\ & \ddots & \\ & & d_M^{-1} \end{bmatrix}$



$D^{-1} = \begin{bmatrix} d_1^{-1} & & \\ & \ddots & \\ & & d_M^{-1} \end{bmatrix}$