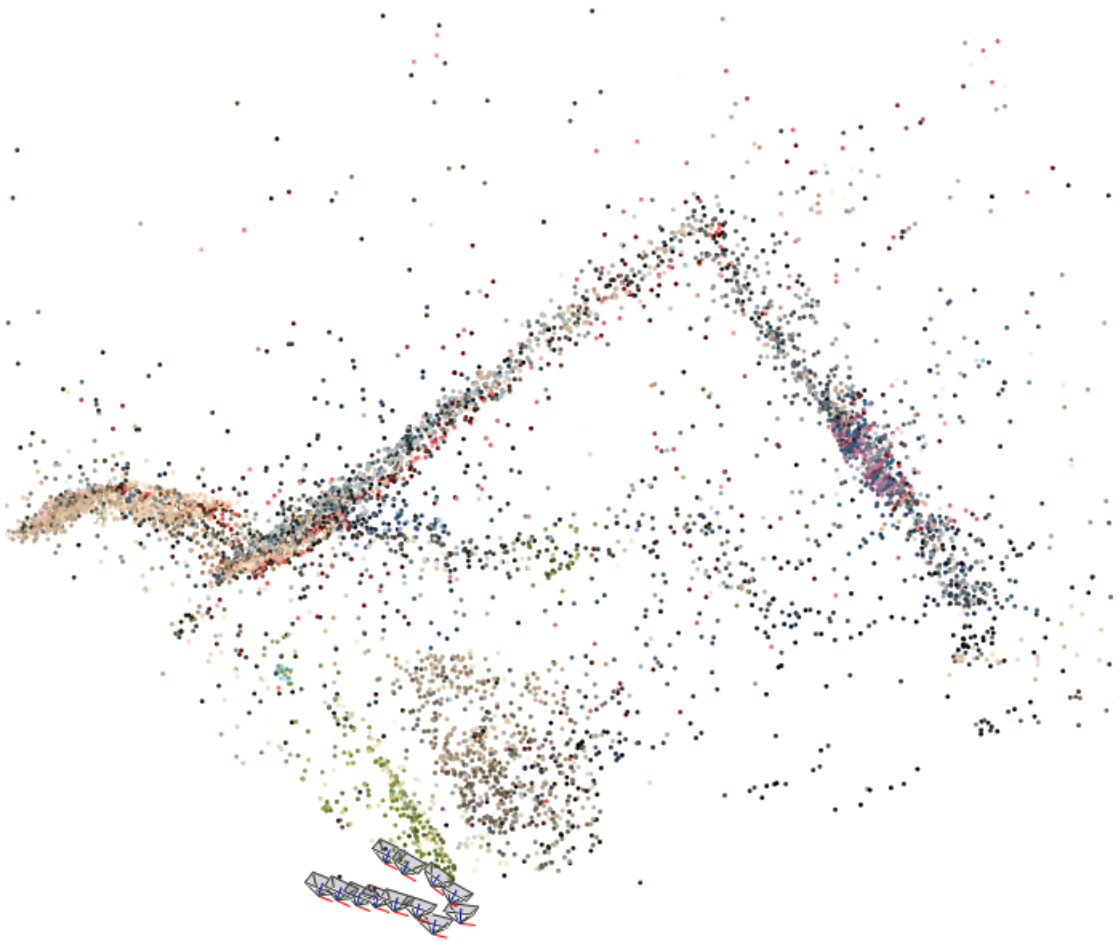


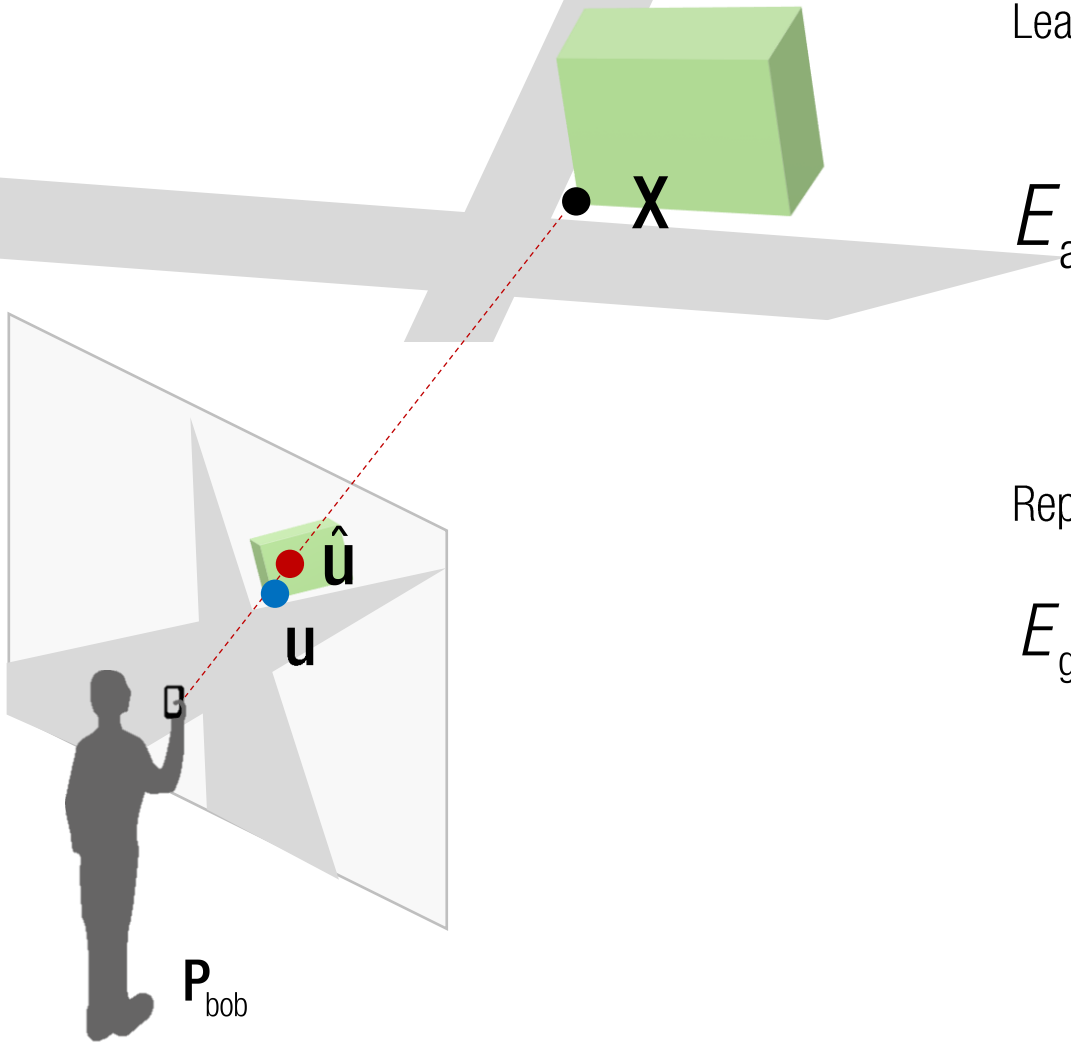
Bundle Adjustment

Announcement

- HW #6 is out
  - Two deadlines: May 3 (Problem 2,3,4,5) and May 10 (Problem 6 with full reconstruction)
  - 5980 students: < 5 image reconstruction
  - 8980 students: <10 image reconstruction

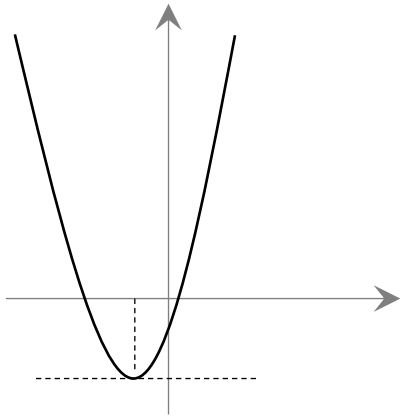
Triangulation Refinement

Algebraic vs. Geometric error



Least squares solution (algebraic error):

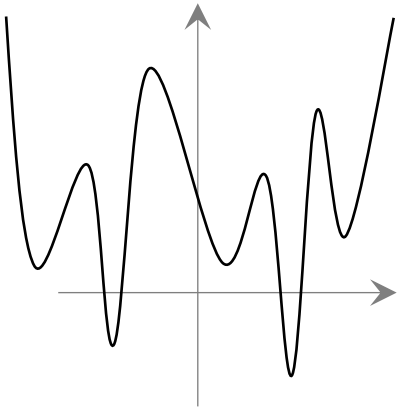
$$E_{\text{alge}} = \| \mathbf{A}\ \mathbf{x} - \mathbf{b} \|^2$$



Reprojection error (geometric error):

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

$$= \left(\frac{\mathbf{P}_1\mathbf{X}}{\mathbf{P}_3\mathbf{X}} - \mathbf{u}_1 \right)^2 + \left(\frac{\mathbf{P}_2\mathbf{X}}{\mathbf{P}_3\mathbf{X}} - \mathbf{u}_2 \right)^2$$

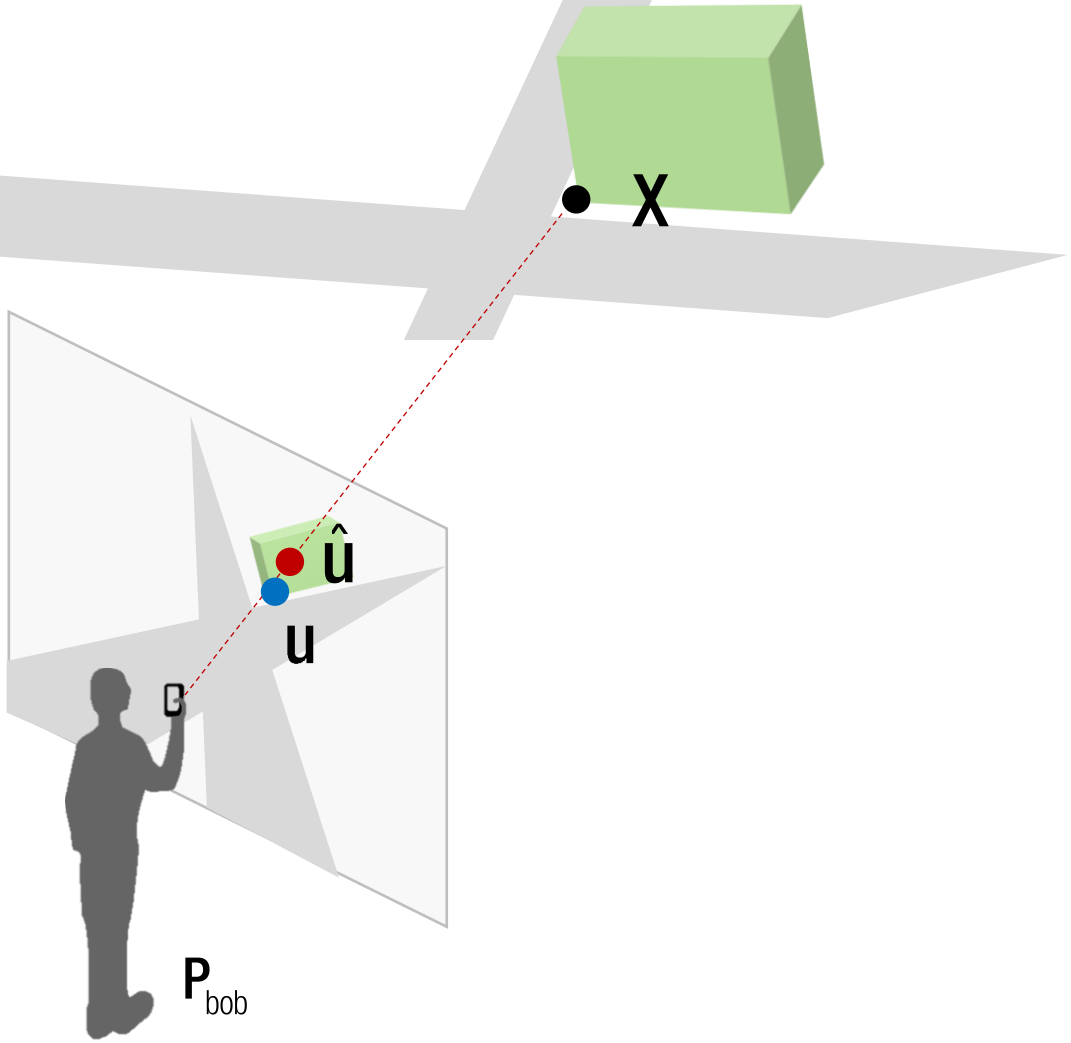


Point Jacobian

Black: given variables
Red: unknowns

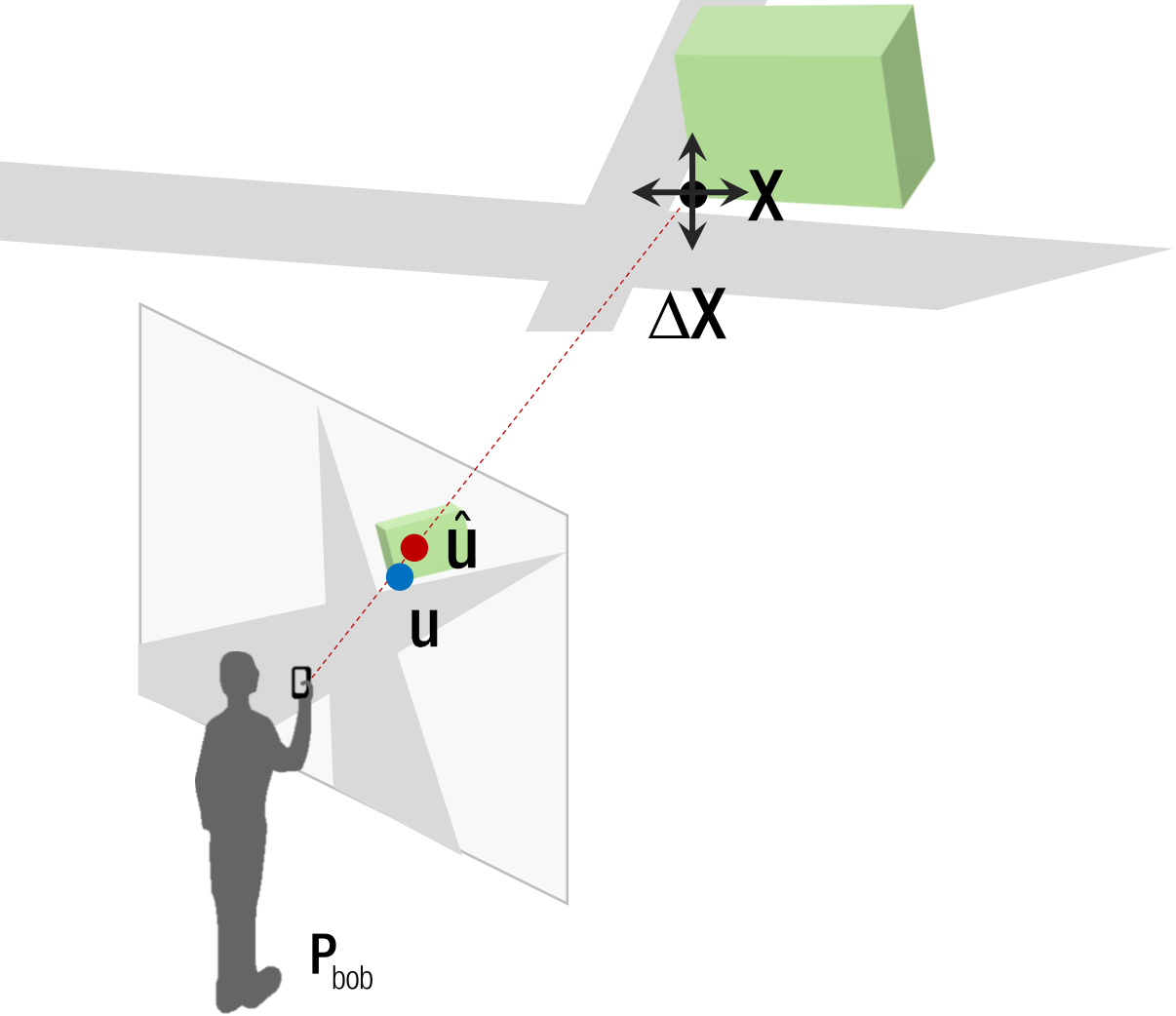
$$E_{\text{geom}} = \left\| \hat{\mathbf{u}} - \mathbf{u} \right\|^2$$

$$= \left(\frac{\mathbf{P}_1\mathbf{X}}{\mathbf{P}_3\mathbf{X}} - x \right)^2 + \left(\frac{\mathbf{P}_2\mathbf{X}}{\mathbf{P}_3\mathbf{X}} - y \right)^2$$



Point Jacobian

Black: given variables
Red: unknowns



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

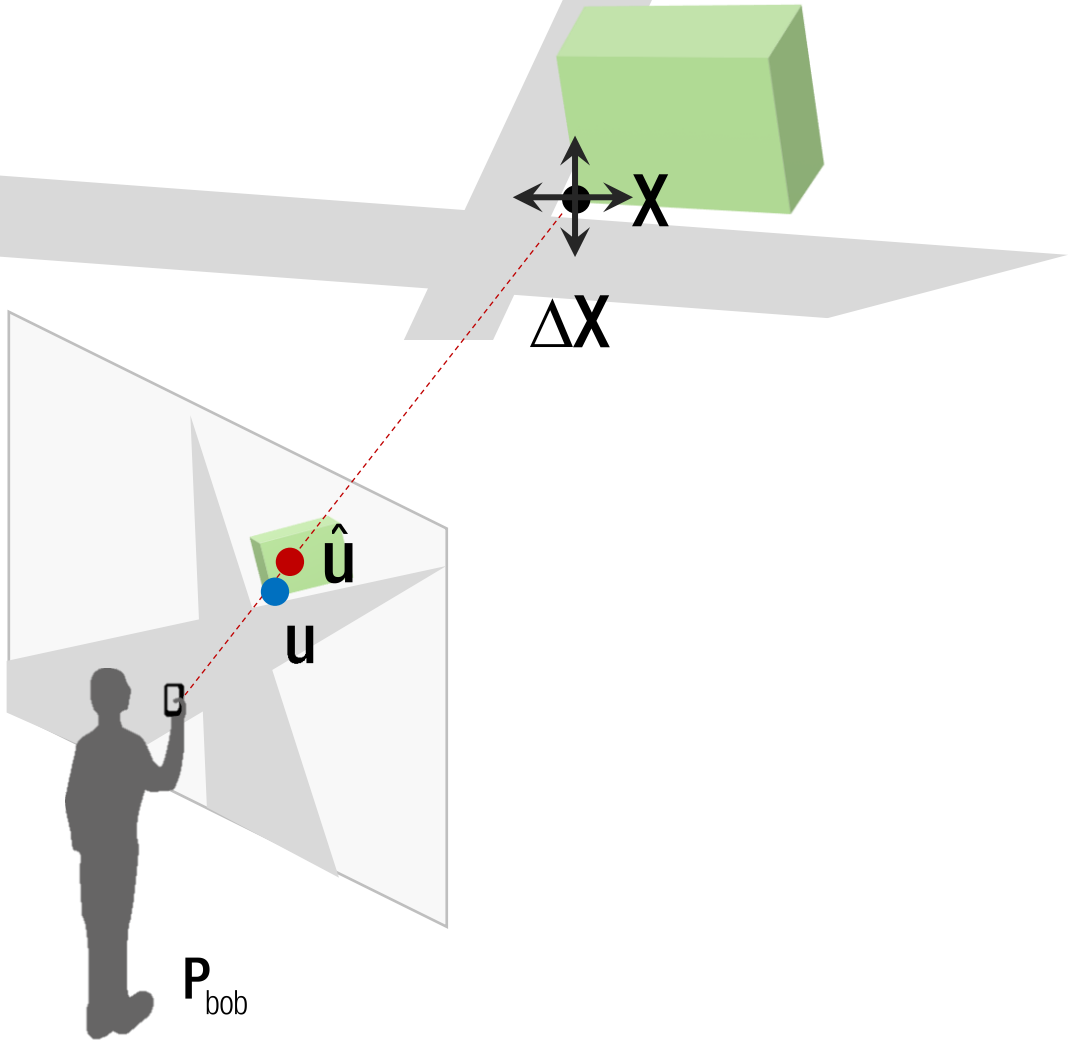
$$= \left(\frac{\mathbf{P}_1\mathbf{X}}{\mathbf{P}_3\mathbf{X}} - x\right)^2 + \left(\frac{\mathbf{P}_2\mathbf{X}}{\mathbf{P}_3\mathbf{X}} - y\right)^2$$

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

Point Jacobian

Black: given variables
Red: unknowns

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

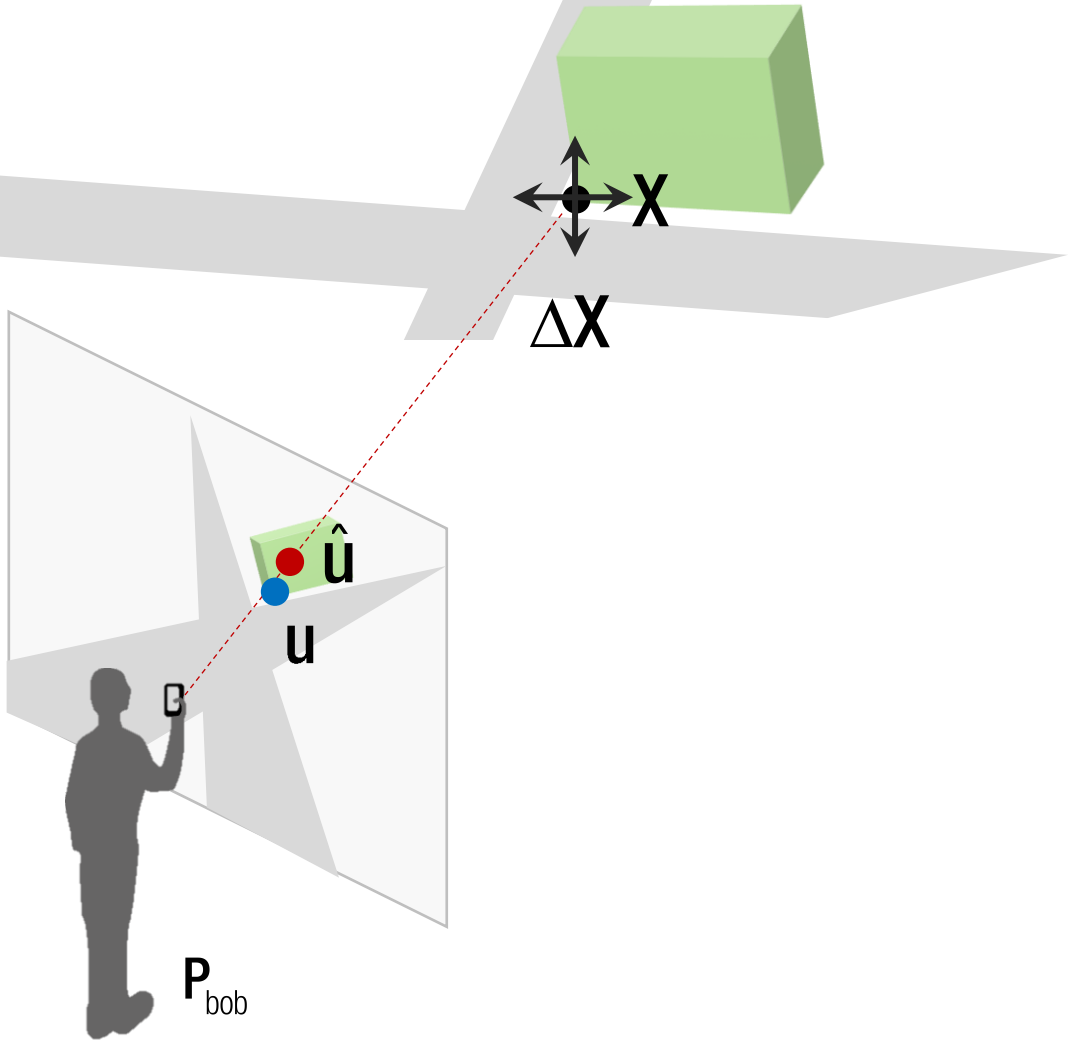


$$\Delta\mathbf{x} = \left(\frac{\partial f(\mathbf{x})}{\partial \mathbf{x}}^{\mathsf{T}} \frac{\partial f(\mathbf{x})}{\partial \mathbf{x}}\right)^{-1} \frac{\partial f(\mathbf{x})}{\partial \mathbf{x}}^{\mathsf{T}} \left(\mathbf{b} - f(\mathbf{x})\right)$$

Point Jacobian

Black: given variables
Red: unknowns

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



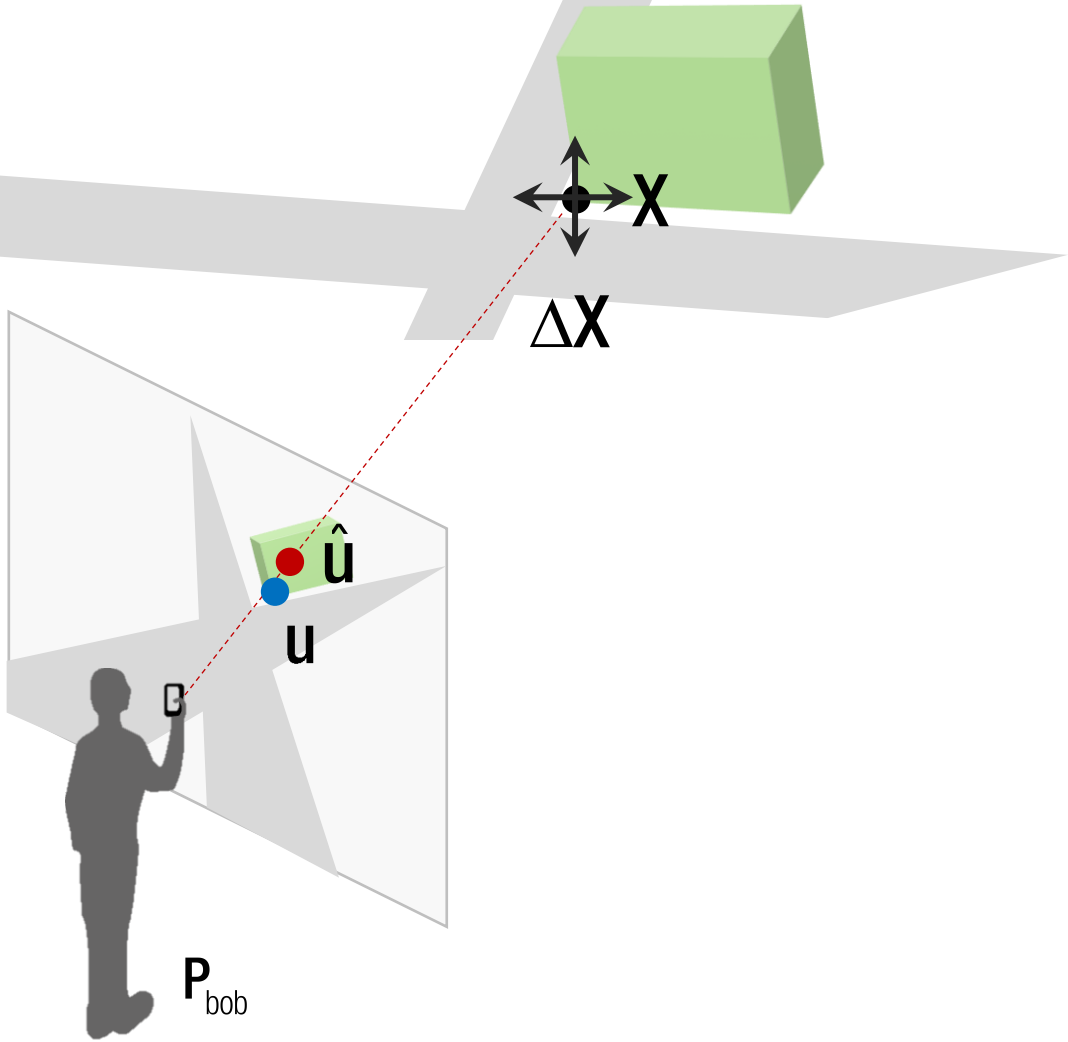
$$f(\mathbf{X}) = \begin{bmatrix} \frac{u}{w} \\ \frac{v}{w} \end{bmatrix}$$

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

Point Jacobian

Black: given variables
Red: unknowns

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

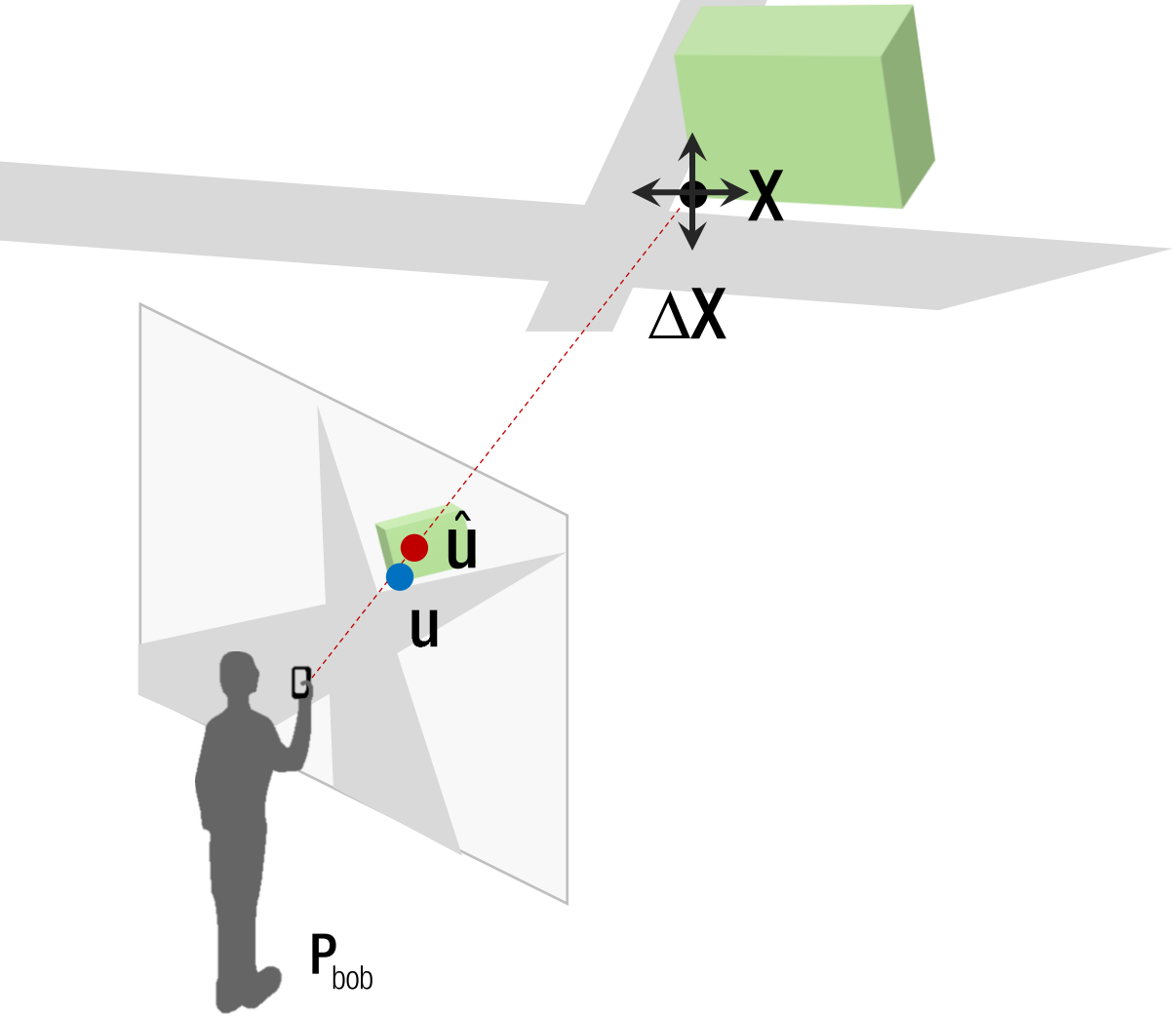


$$f(\mathbf{X}) = \begin{bmatrix} \frac{u}{w} \\ \frac{v}{w} \end{bmatrix} \longrightarrow \frac{\partial f(\mathbf{X})}{\partial \mathbf{X}} = \frac{\partial}{\partial \mathbf{X}} \begin{bmatrix} \frac{u}{w} \\ \frac{v}{w} \end{bmatrix} = \begin{bmatrix} \frac{w\frac{\partial u}{\partial \mathbf{X}}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

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

Point Jacobian

Black: given variables
Red: unknowns



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

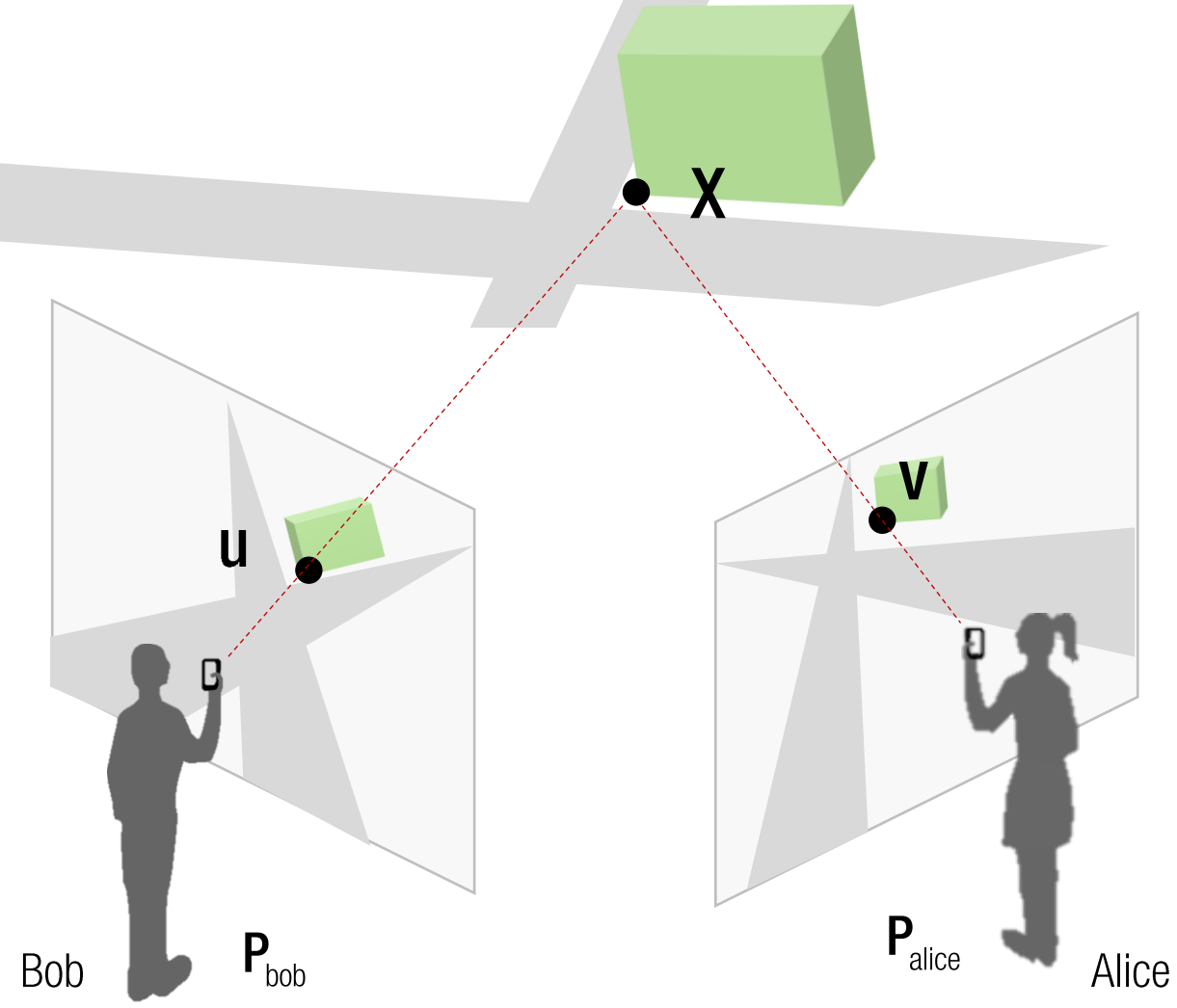
$$\rightarrow \frac{\partial}{\partial \mathbf{X}} \begin{bmatrix} u \\ v \\ w \end{bmatrix} = \mathbf{KR}$$

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

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

Point Jacobian

Black: given variables
Red: unknowns



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

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

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

Algorithm 3 Nonlinear Point Refinement

$\mathbf{b} = \begin{bmatrix} \mathbf{u}_1^\mathsf{T} & \mathbf{u}_2^\mathsf{T} \end{bmatrix}^\mathsf{T}$

for j = 1 : nIters **do**

  Build point Jacobian, $\frac{\partial f(\mathbf{X})_j}{\partial \mathbf{X}}$.

  Compute $f(\mathbf{X})$.

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

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

end for

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

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

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**

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

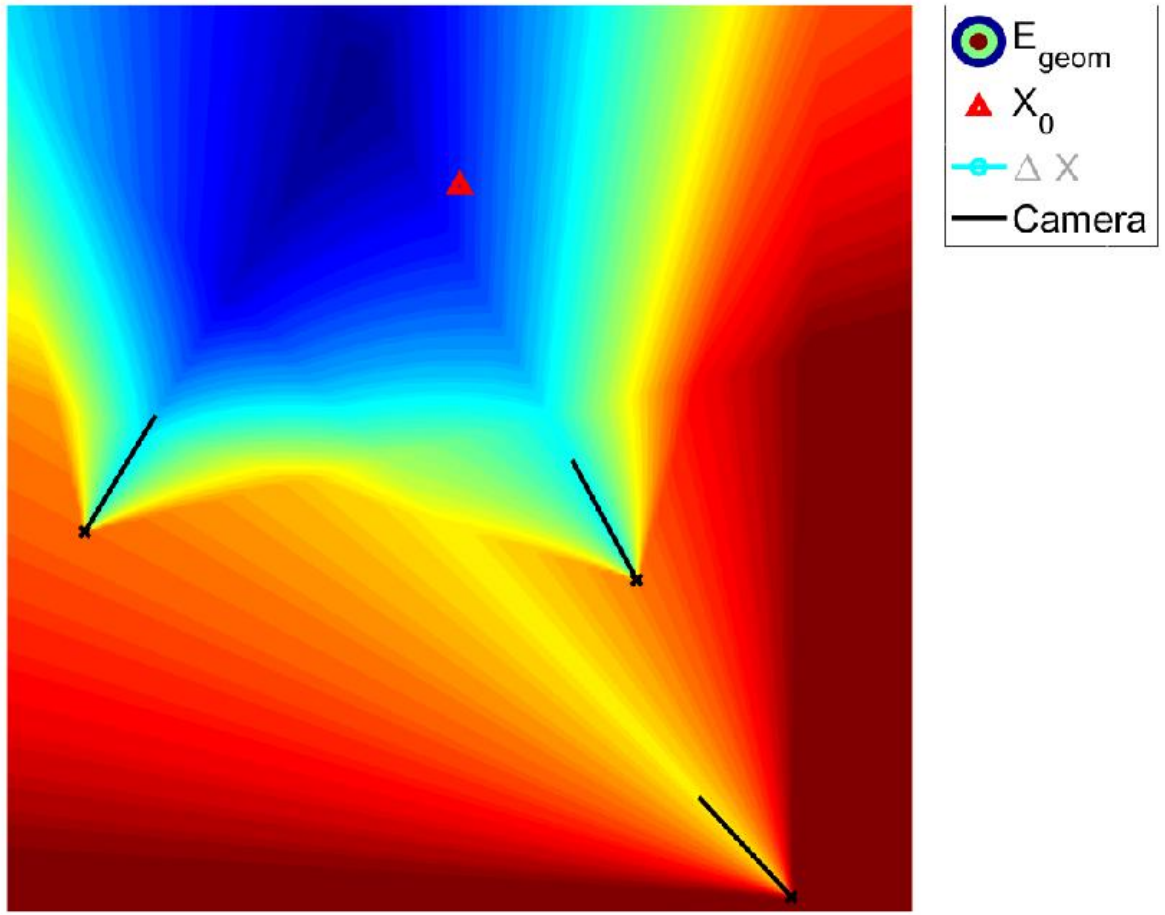
$$\Delta\mathbf{x} = \left(\frac{\partial f(\mathbf{x})}{\partial \mathbf{x}}^\mathsf{T} \frac{\partial f(\mathbf{x})}{\partial \mathbf{x}} \right)^{-1} \frac{\partial f(\mathbf{x})}{\partial \mathbf{x}}^\mathsf{T} (\mathbf{b} - f(\mathbf{x}))$$

Damping factor (Levenberg-Marquardt algorithm)

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

Black: given variables
Red: unknowns

Example: 1D Camera Triangulation

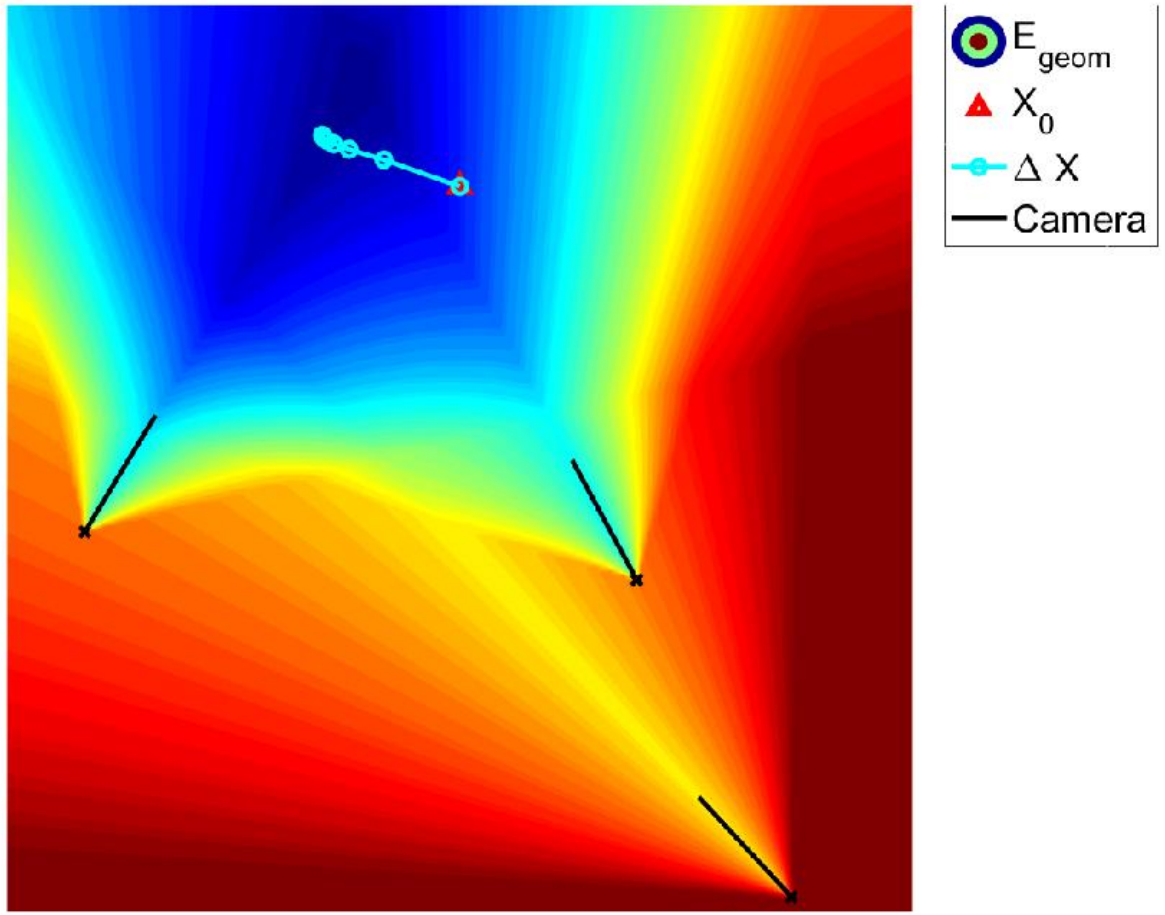



```
function df_dx = JacobianX_1D(K, R,  C,  X)

x = K * R * (X-C);

u = x(1);
w = x(2);

del = K * R;
du_dc = del(1,:);
dw_dc = del(2,:);

df_dx = [(w*du_dc-u*dw_dc)/w^2];
```

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

Black: given variables
Red: unknowns

Example: 1D Camera Triangulation

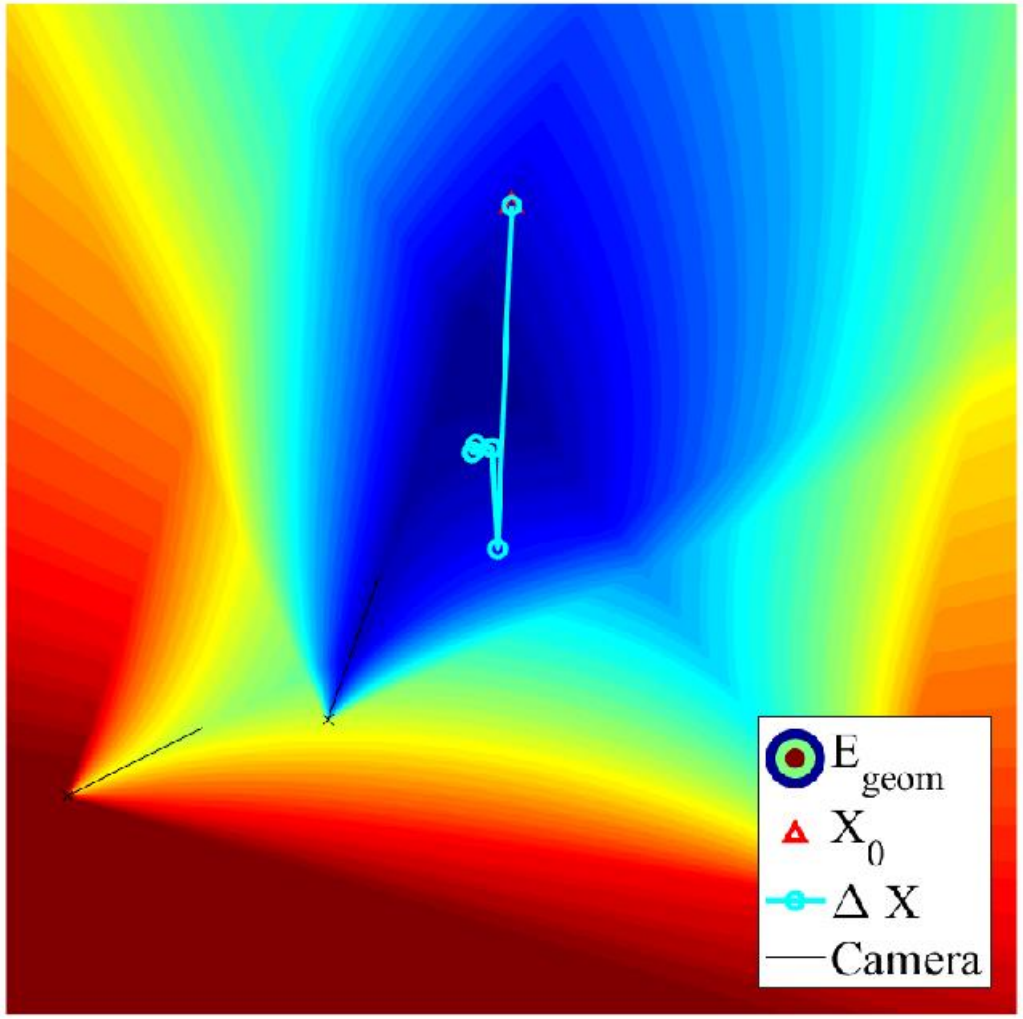


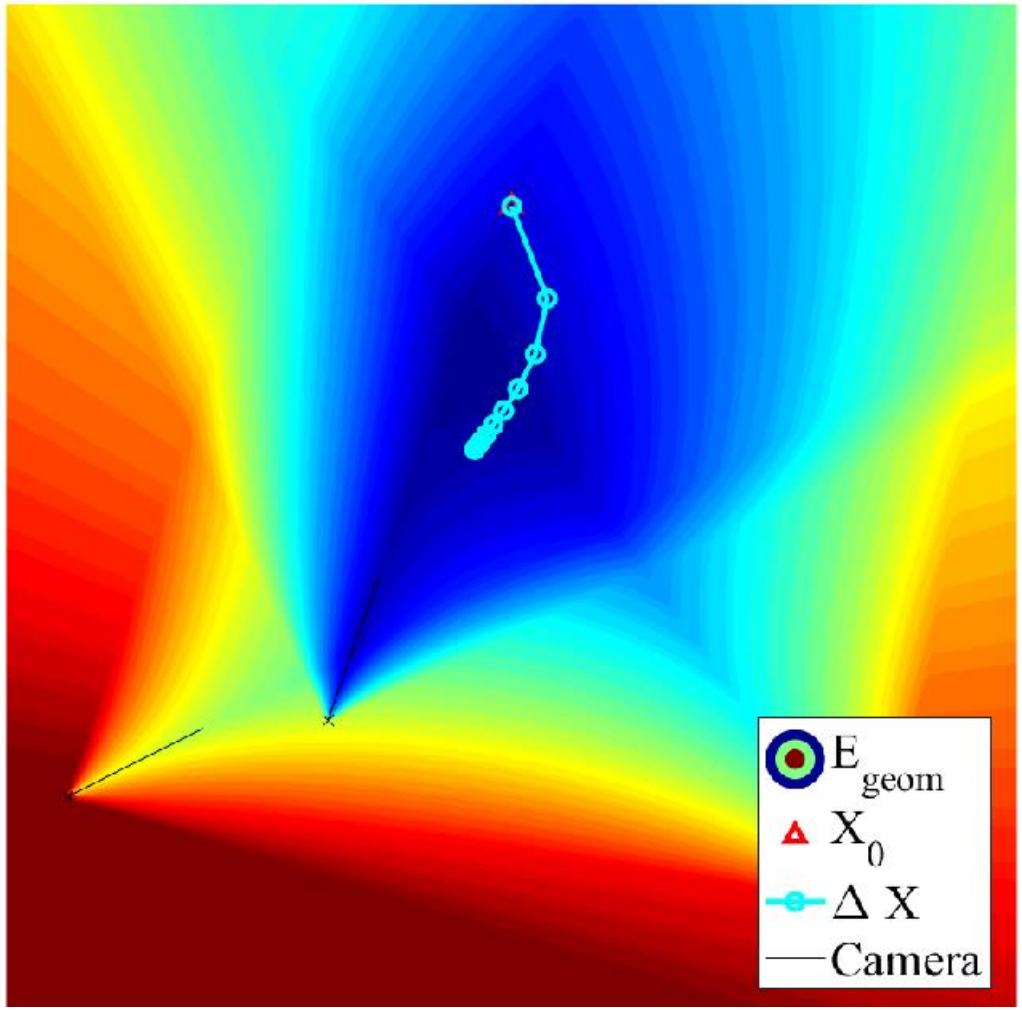
NonlinearTriangulation1D.m

```
for j = 1 : 10
   df_dX = [];
   delta_b = [];
   for i = 1 : size(c,2)
      df_dX = [df_dX; JacobianX(eye(2), R{i}, c(:,i), x)];
      u = R{i} * (x-c(:,i));
      delta_b = [delta_b; -u(1)/u(2)];
   end

   jacobian = df_dX;

   norm(delta_b)

   delta_x =
inv(jacobian'*jacobian+lambda*eye(size(jacobian'*jacobian,1)))*jacobia
n'*delta_b;
   x = x + delta_x;
   X(:,j+1) = x;
end
```

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

Damping Factor





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

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

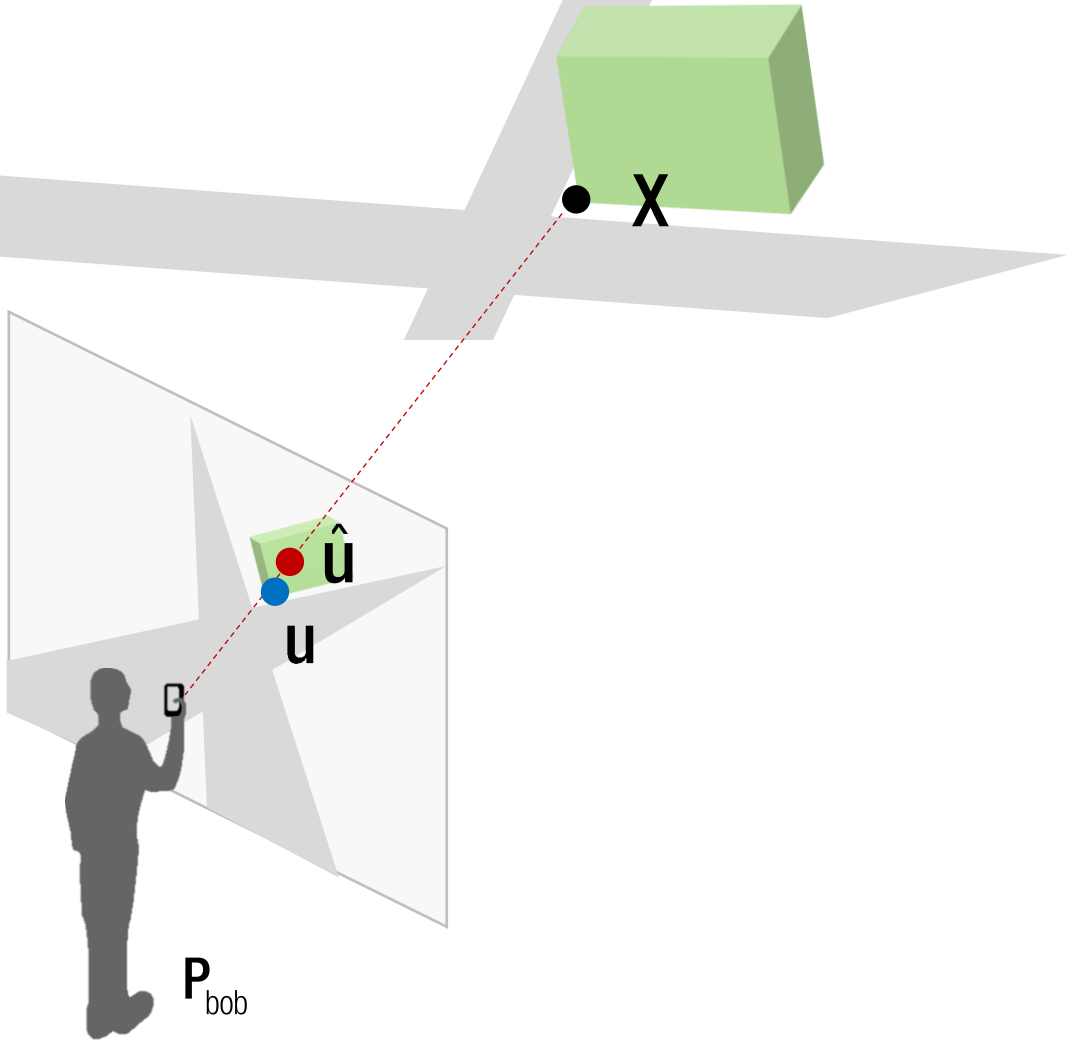
PnP Refinement

Point Jacobian

Black: given variables
Red: unknowns

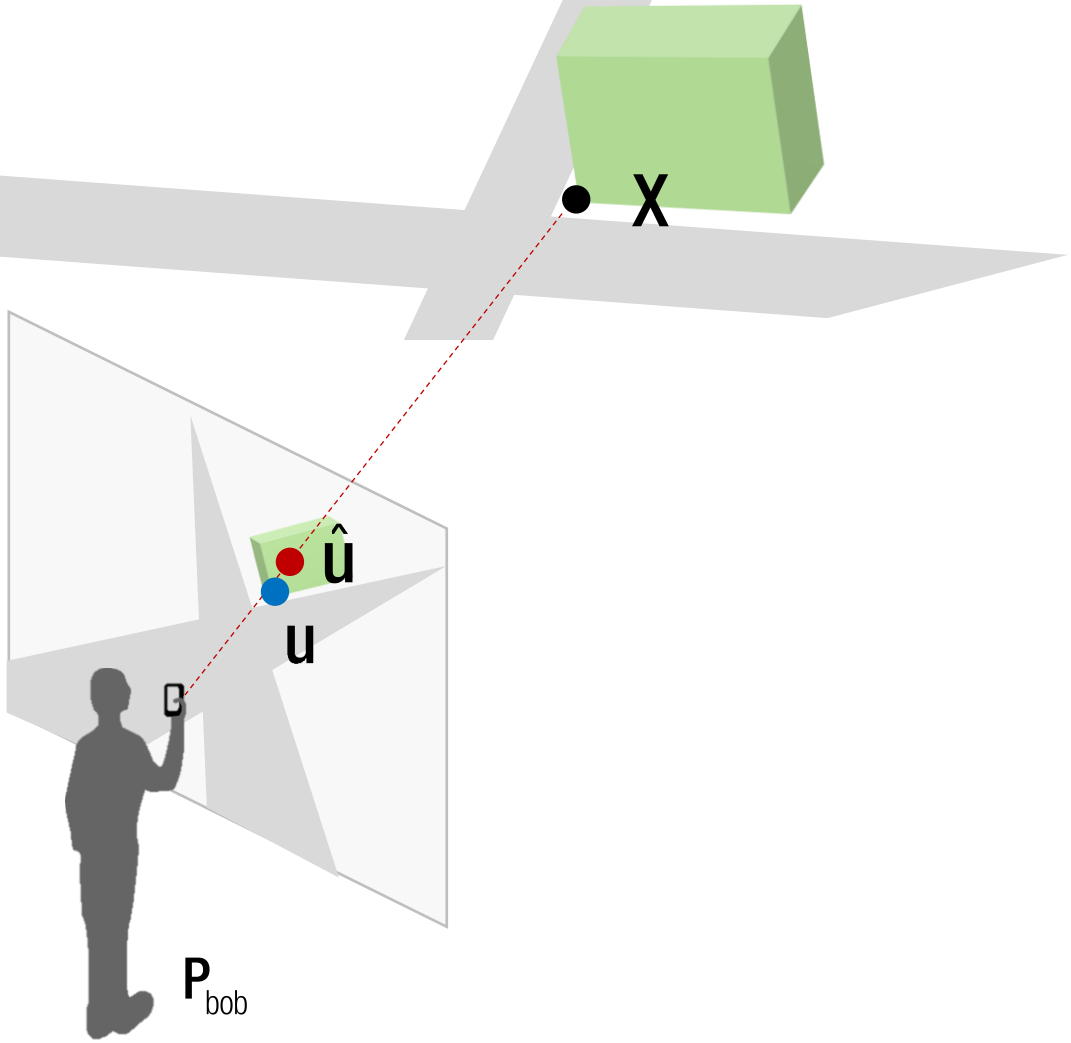
$$E_{\text{geom}} = \left\| \hat{\mathbf{u}} - \mathbf{u} \right\|^2$$

$$= \left(\frac{\mathbf{P}_1\mathbf{X}}{\mathbf{P}_3\mathbf{X}} - x \right)^2 + \left(\frac{\mathbf{P}_2\mathbf{X}}{\mathbf{P}_3\mathbf{X}} - y \right)^2$$



Black: given variables
Red: unknowns

Camera Jacobian

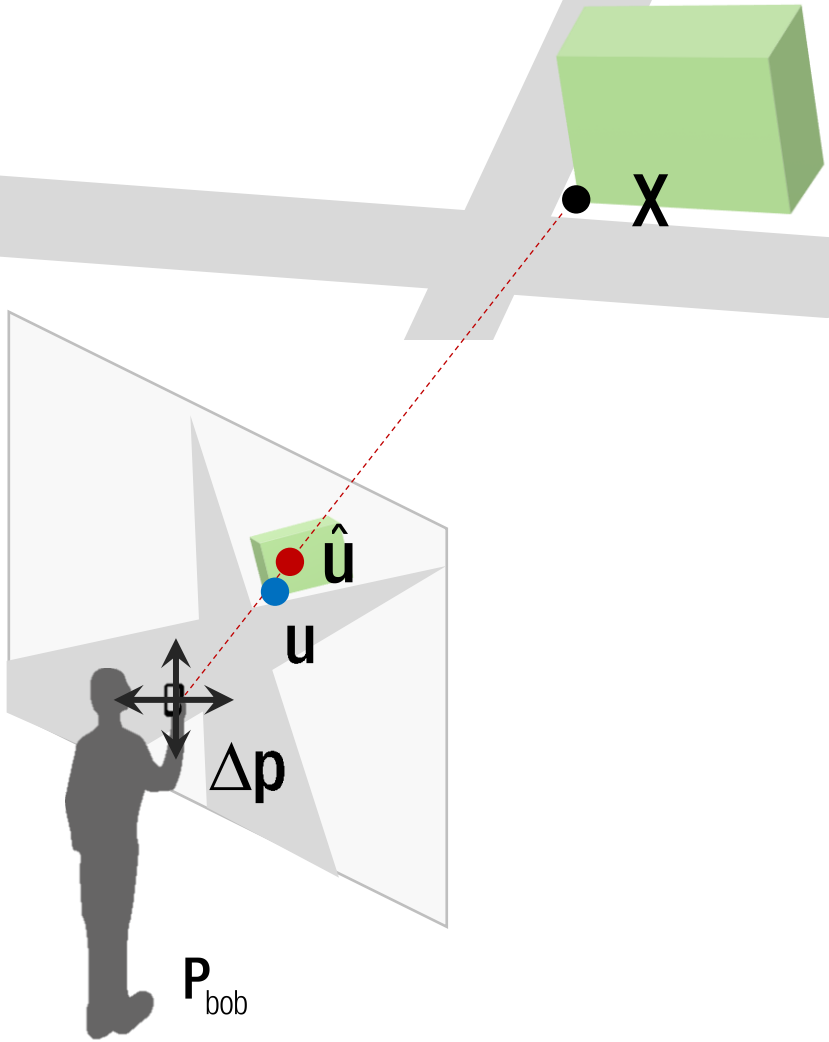


$$E_{\text{geom}} = \|\hat{\mathbf{u}} - \mathbf{u}\|^2$$

$$= \left(\frac{\mathbf{P}_1\mathbf{X}}{\mathbf{P}_3\mathbf{X}} - x\right)^2 + \left(\frac{\mathbf{P}_2\mathbf{X}}{\mathbf{P}_3\mathbf{X}} - y\right)^2$$

Black: given variables
Red: unknowns

Camera Jacobian



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

$$= \left(\frac{\mathbf{P}_1\mathbf{X}}{\mathbf{P}_3\mathbf{X}} - x\right)^2 + \left(\frac{\mathbf{P}_2\mathbf{X}}{\mathbf{P}_3\mathbf{X}} - y\right)^2$$

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

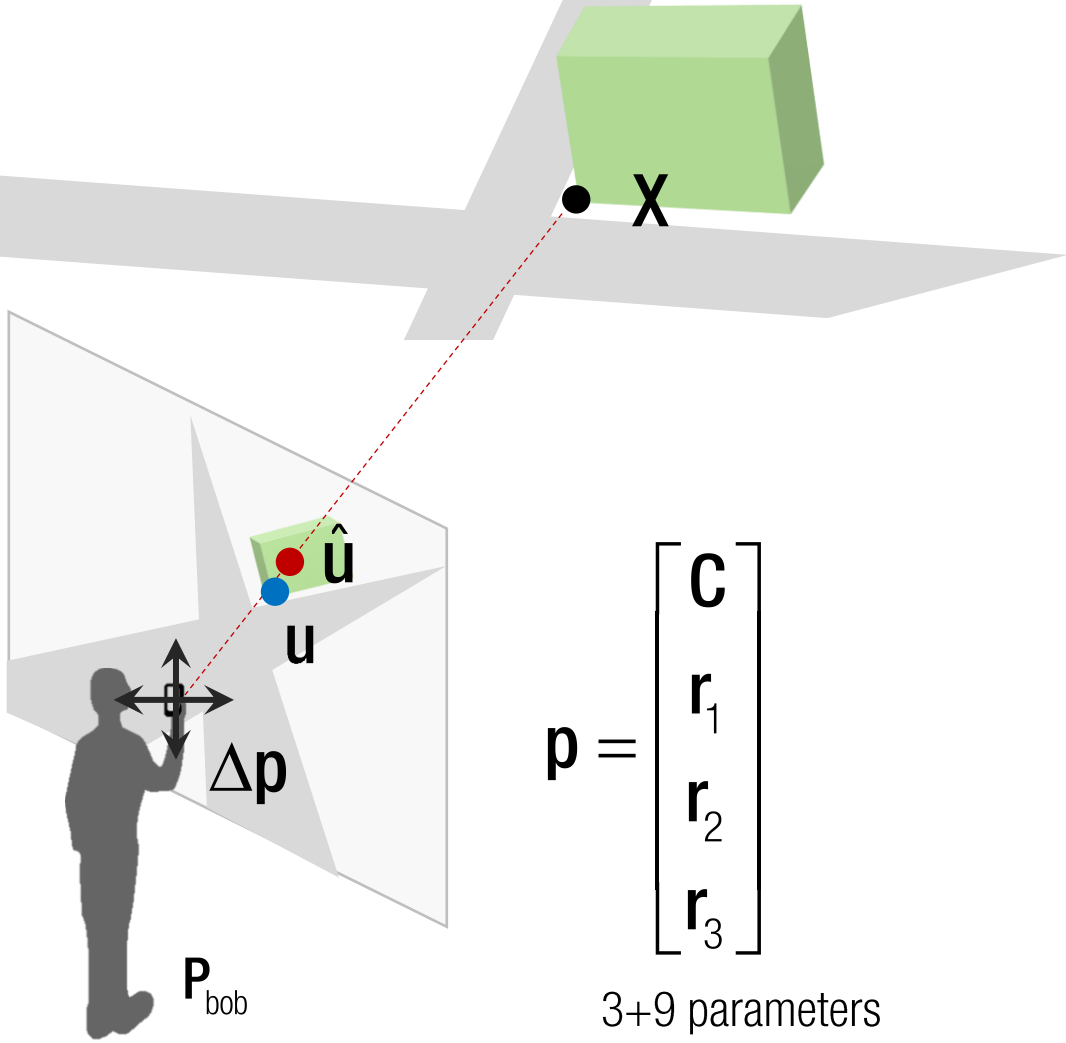
Camera Jacobian

Black: given variables
Red: unknowns

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

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

$$\rightarrow \quad \frac{\partial}{\partial \mathbf{R}} \begin{bmatrix} u \\ v \\ w \end{bmatrix} = \begin{bmatrix} \mathbf{K}_{11}(\mathbf{X} - \mathbf{C}) & \mathbf{0}_{1\times3} & \mathbf{K}_{13}(\mathbf{X} - \mathbf{C}) \\ \mathbf{0}_{1\times3} & \mathbf{K}_{22}(\mathbf{X} - \mathbf{C}) & \mathbf{K}_{23}(\mathbf{X} - \mathbf{C}) \\ \mathbf{0}_{1\times3} & \mathbf{0}_{1\times3} & (\mathbf{X} - \mathbf{C}) \end{bmatrix}$$



$$\mathbf{p} = \begin{bmatrix} \mathbf{C} \\ \mathbf{r}_1 \\ \mathbf{r}_2 \\ \mathbf{r}_3 \end{bmatrix}$$

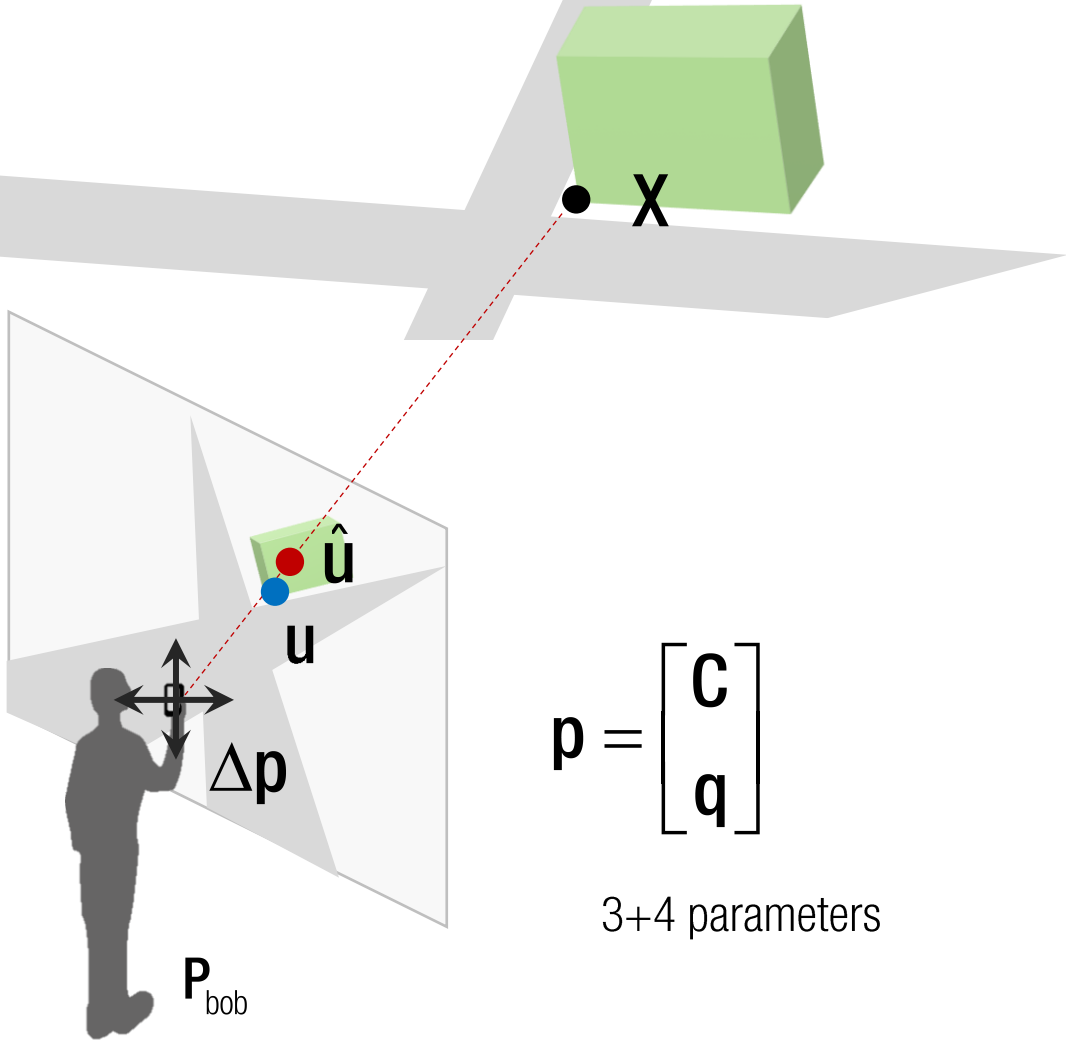
3+9 parameters

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

Camera Jacobian

Black: given variables
Red: unknowns

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



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

3+4 parameters

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

$$\rightarrow \quad \frac{\partial}{\partial \mathbf{R}} \begin{bmatrix} u \\ v \\ w \end{bmatrix} = \begin{bmatrix} \mathbf{K}_{11}(\mathbf{X} - \mathbf{C}) & \mathbf{0}_{1\times3} & \mathbf{K}_{13}(\mathbf{X} - \mathbf{C}) \\ \mathbf{0}_{1\times3} & \mathbf{K}_{22}(\mathbf{X} - \mathbf{C}) & \mathbf{K}_{23}(\mathbf{X} - \mathbf{C}) \\ \mathbf{0}_{1\times3} & \mathbf{0}_{1\times3} & (\mathbf{X} - \mathbf{C}) \end{bmatrix}$$

$$\rightarrow \quad \frac{\partial}{\partial \mathbf{q}} \begin{bmatrix} u \\ v \\ w \end{bmatrix} = \frac{\partial}{\partial \mathbf{R}} \begin{bmatrix} u \\ v \\ w \end{bmatrix} \frac{\partial \mathbf{R}}{\partial \mathbf{q}} \quad : \text{Chain rule}$$

$\overline{\text{Quaternion jacobian}}$

Quaternion Jacobian

$$\mathbf{R} = \begin{bmatrix} 1-2q_y^2-2q_z^2 & 2q_xq_y-2q_zq_w & 2q_xq_z+2q_yq_w \\ 2q_xq_y+2q_zq_w & 1-2q_x^2-2q_z^2 & 2q_yq_z-2q_xq_w \\ 2q_xq_z-2q_yq_w & 2q_yq_z+2q_xq_w & 1-2q_x^2-2q_y^2 \end{bmatrix}$$

where

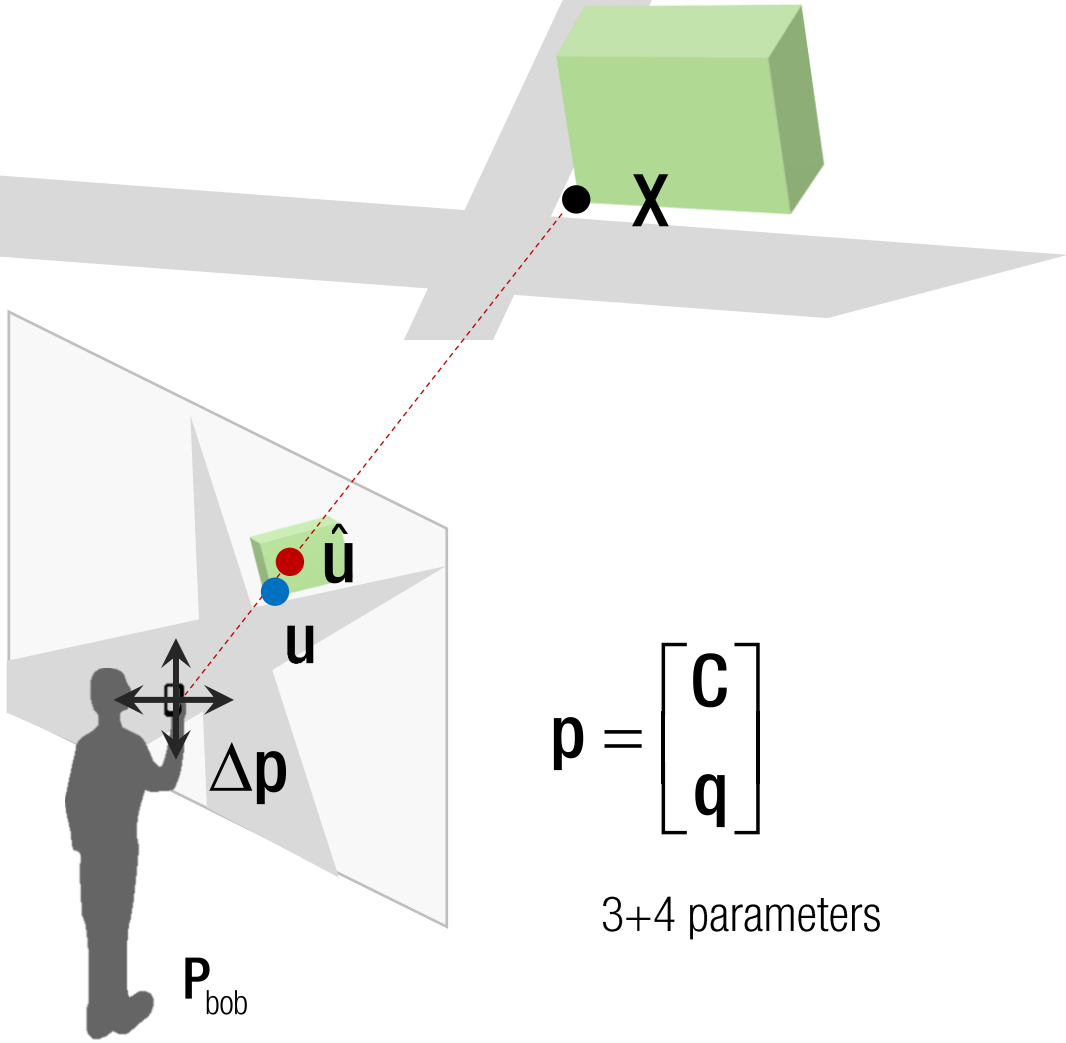
$$\mathbf{q} = \begin{bmatrix} q_w & q_x & q_y & q_z \end{bmatrix}^\top$$

$$\underset{9\text{x}4}{\frac{\partial \mathbf{R}}{\partial \mathbf{q}}} = \begin{bmatrix} 0 & 0 & -4q_y & -4q_z \\ -2q_z & 2q_y & 2q_x & -2q_w \\ 2q_y & 2q_z & 2q_w & 2q_x \\ 2q_z & 2q_y & 2q_x & 2q_w \\ 0 & -4q_x & 0 & -4q_z \\ -2q_x & -2q_w & 2q_z & 2q_y \\ -2q_y & 2q_z & -2q_w & 2q_x \\ 2q_x & 2q_w & 2q_z & 2q_y \\ 0 & -4q_x & -4q_y & 0 \end{bmatrix}$$

Camera Jacobian

Black: given variables
Red: unknowns

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



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

3+4 parameters

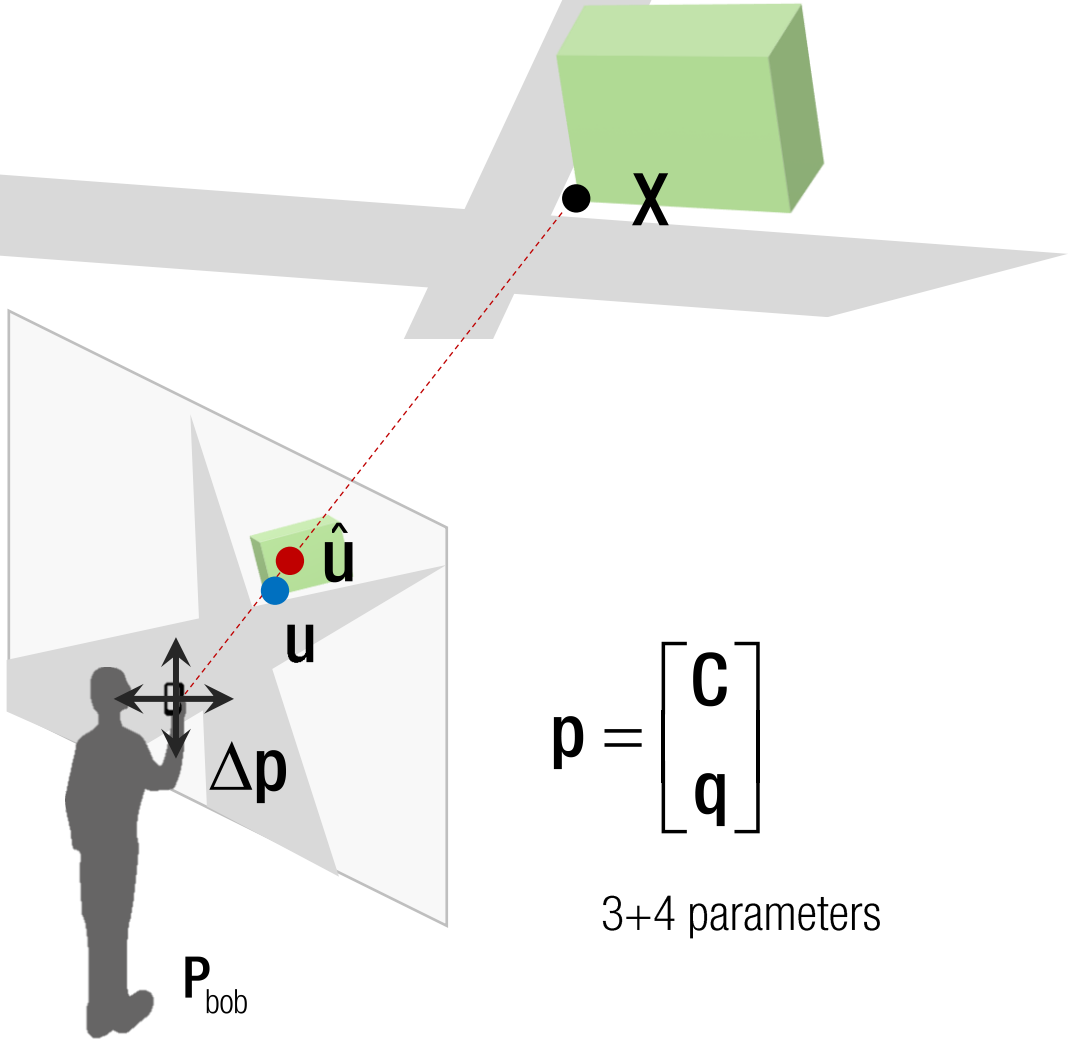
$$f(\mathbf{p}) = \begin{bmatrix} \frac{u}{w} \\ \frac{v}{w} \end{bmatrix} \quad \rightarrow \quad \frac{\partial f(\mathbf{p})}{\partial \mathbf{p}} = \frac{\partial}{\partial \mathbf{p}} \begin{bmatrix} \frac{u}{w} \\ \frac{v}{w} \end{bmatrix} = \begin{bmatrix} \frac{w\frac{\partial u}{\partial \mathbf{p}}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

Camera Jacobian

Black: given variables
Red: unknowns

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

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



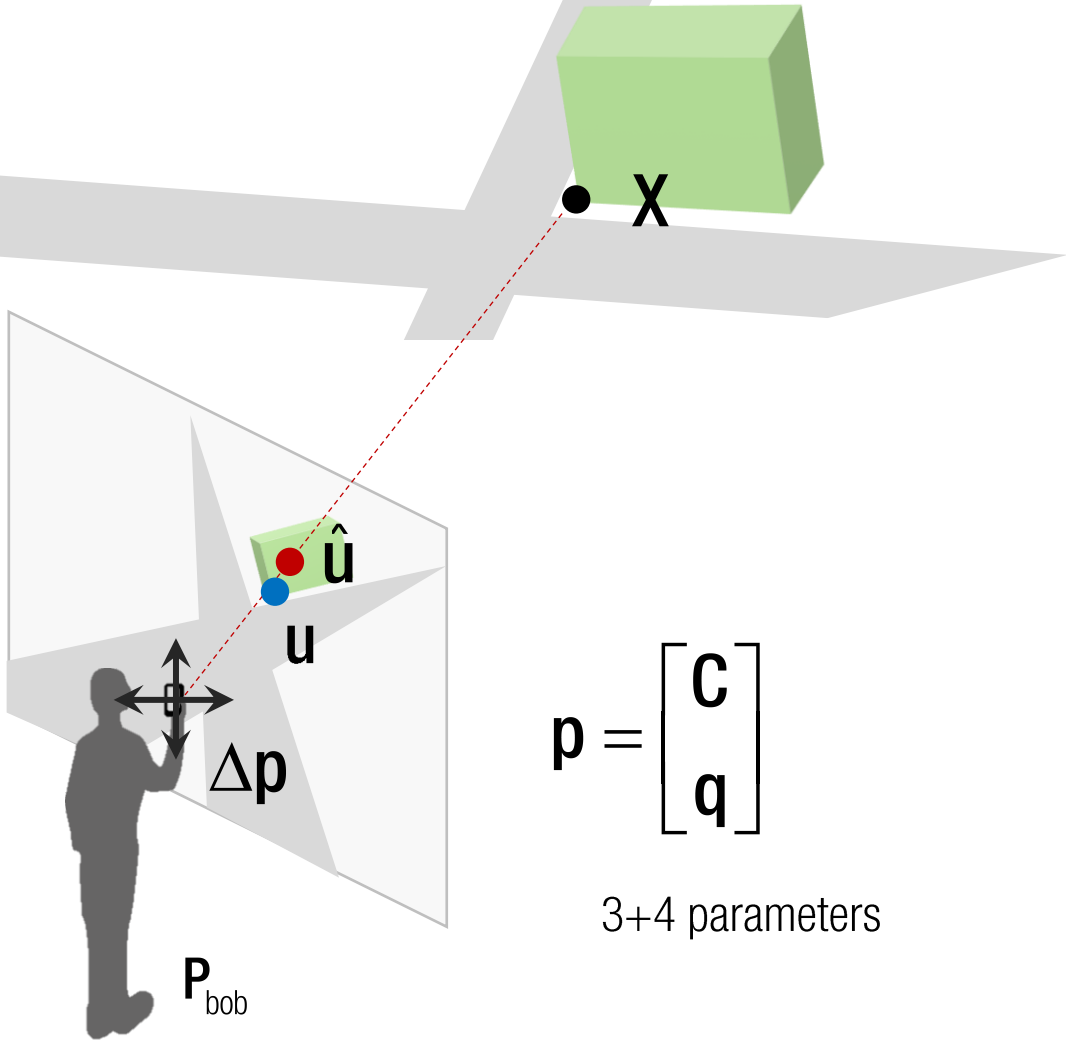
$$\mathbf{p} = \begin{bmatrix} \mathbf{C} \\ \mathbf{q} \end{bmatrix}$$

3+4 parameters

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

Camera Jacobian

Black: given variables
Red: unknowns



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

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

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

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

3+4 parameters

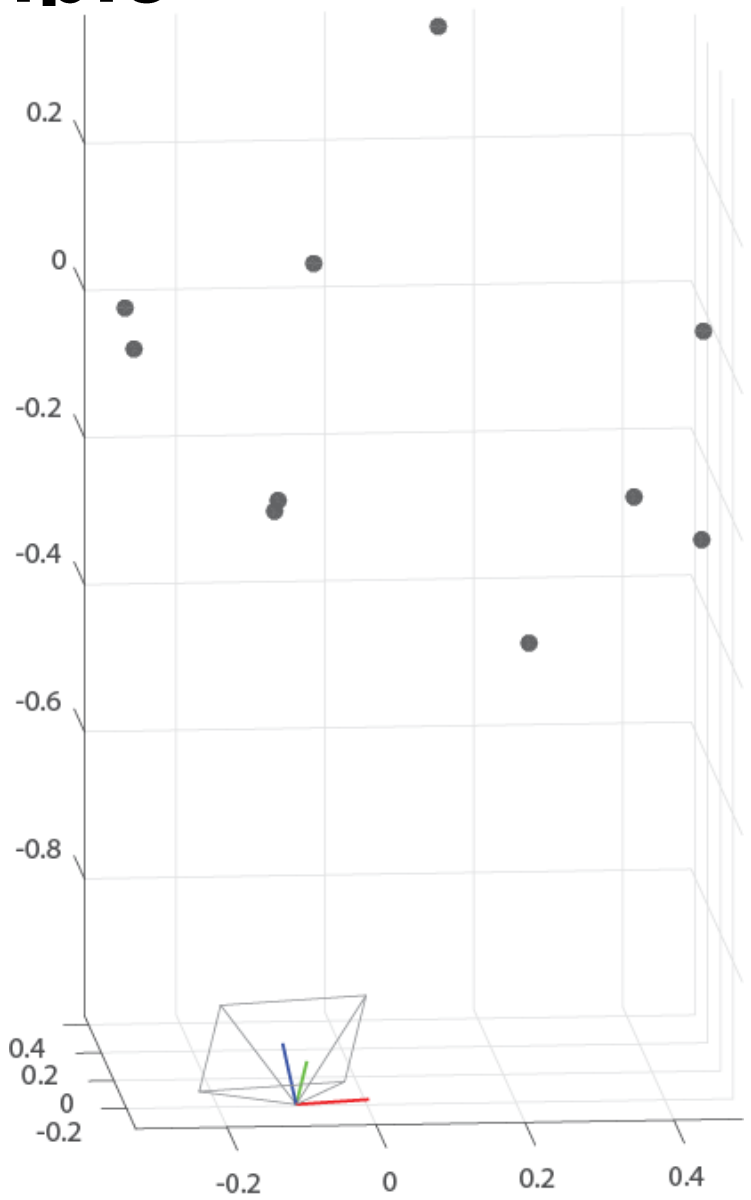
$$f(\mathbf{p}) = \begin{bmatrix} \frac{u}{w} \\ \frac{v}{w} \end{bmatrix} \quad \rightarrow \quad \frac{\partial f(\mathbf{p})}{\partial \mathbf{p}} = \frac{\partial}{\partial \mathbf{p}} \begin{bmatrix} \frac{u}{w} \\ \frac{v}{w} \end{bmatrix} = \begin{bmatrix} \frac{w \frac{\partial u}{\partial \mathbf{p}}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

Algorithm 2 Nonlinear Camera Pose Refinement

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

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

Example

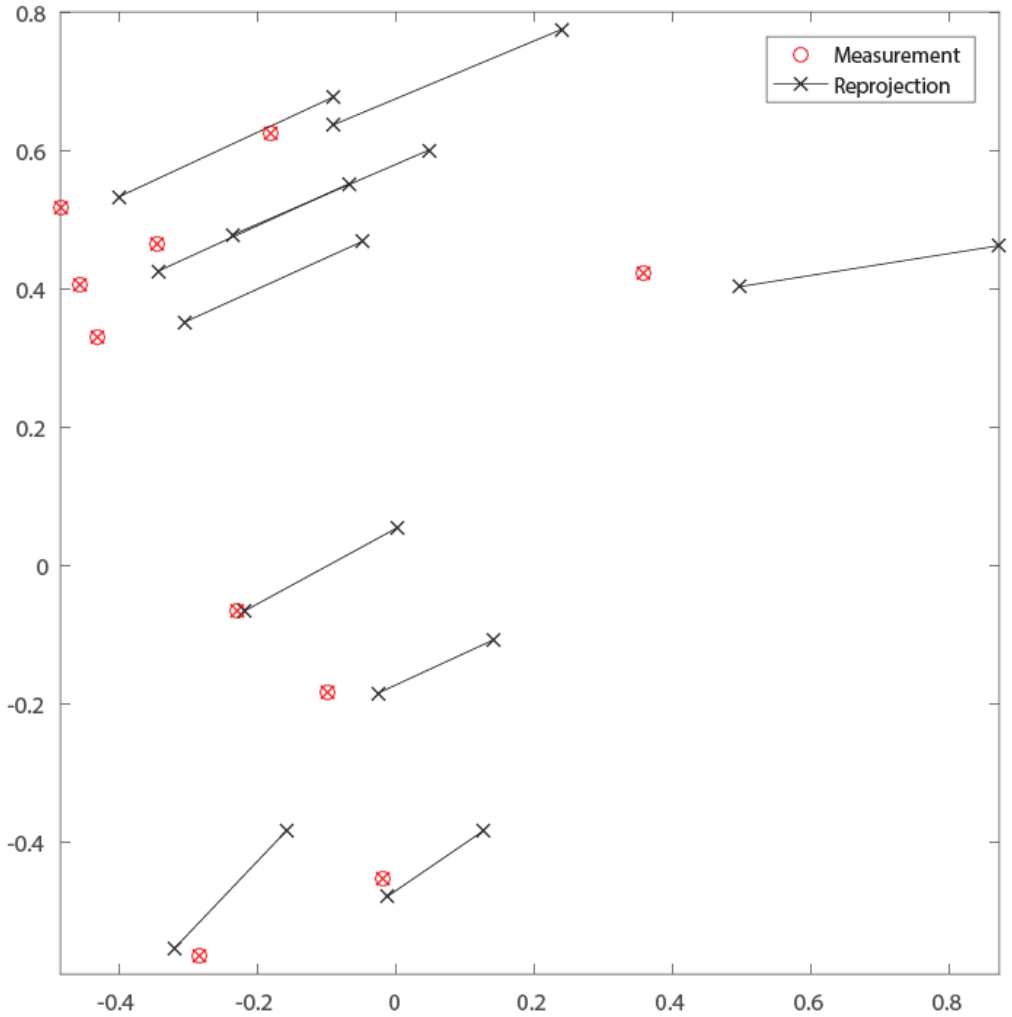



```
u = K*R*[eye(3) -C]*[X'; ones(1,nPoints)];
u = [u(1,:)./u(3,:); u(2,:)./u(3,:)];

x = [C; q];
for j = 1 : 40
    R1 = Quaternion2Rotation(x(4:7));
    C1 = x(1:3);

    df_dc = [];
    df_dR = [];
    for k = 1 : nPoints
        df_dc = [df_dc; JacobianC(K, R1, C1, X(k,:)')];
        df_dR = [df_dR; JacobianR(K, R1, C1, X(k,:)')*JacobianQ(x(4:7))];
    end

    u1 = K*R1*[eye(3) -C1]*[X'; ones(1,nPoints)];
    u1 = [u1(1,:)./u1(3,:); u1(2,:)./u1(3,:)];

    jacobian = [df_dc df_dR];
    delta_b = u(:)-u1(:);

    delta_x = inv(jacobian'*jacobian+lambda*eye(size(jacobian'*jacobian,1))*jacobian'*delta_b
    x = x + delta_x;
    x(4:7) = x(4:7)/norm(x(4:7));
end
```

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

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

Example

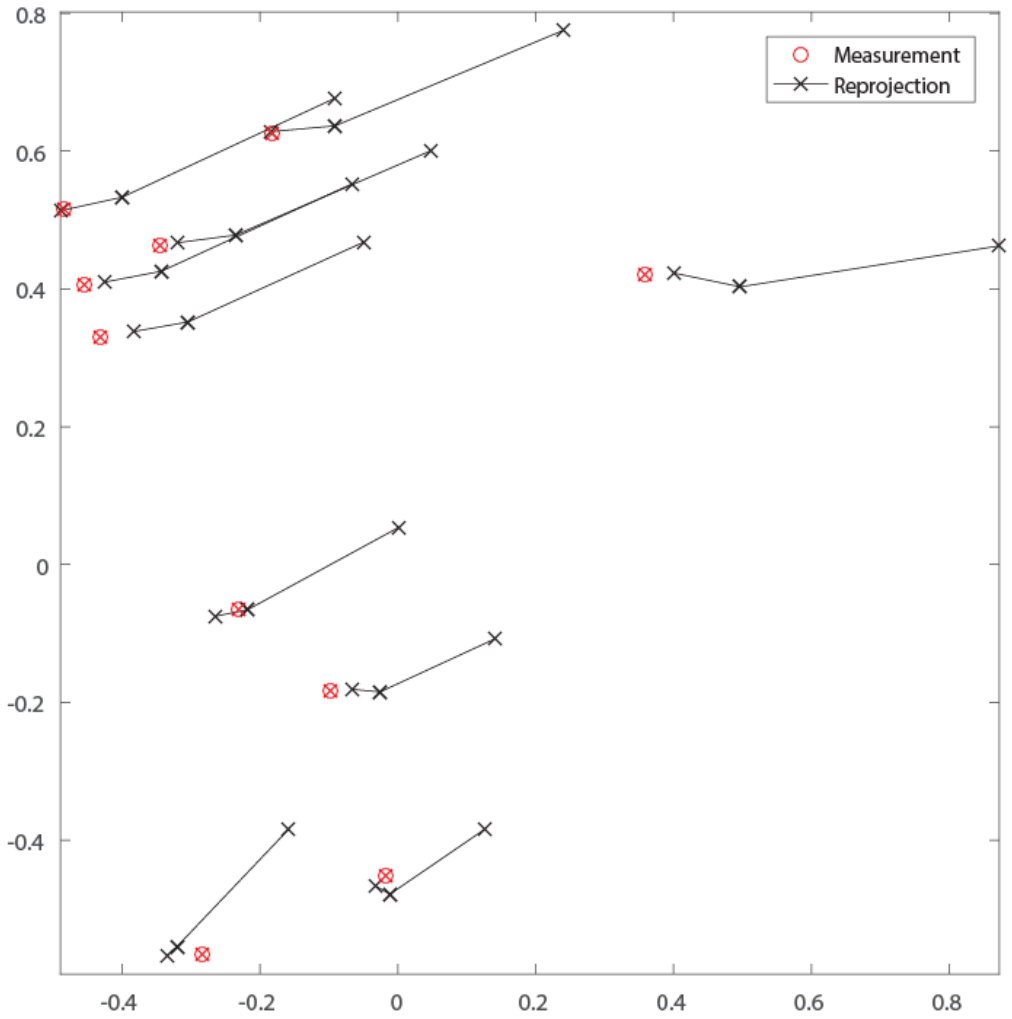



```
u = K*R*[eye(3) -C]*[X'; ones(1,nPoints)];
u = [u(1,:)./u(3,:); u(2,:)./u(3,:)];

x = [C; q];
for j = 1 : 40
    R1 = Quaternion2Rotation(x(4:7));
    C1 = x(1:3);

    df_dc = [];
    df_dR = [];
    for k = 1 : nPoints
        df_dc = [df_dc; JacobianC(K, R1, C1, X(k,:)')];
        df_dR = [df_dR; JacobianR(K, R1, C1, X(k,:)')*JacobianQ(x(4:7))];
    end

    u1 = K*R1*[eye(3) -C1]*[X'; ones(1,nPoints)];
    u1 = [u1(1,:)./u1(3,:); u1(2,:)./u1(3,:)];

    jacobian = [df_dc df_dR];
    delta_b = u(:)-u1(:);

    delta_x = inv(jacobian'*jacobian+lambda*eye(size(jacobian'*jacobian,1))*jacobian'*delta_b
    x = x + delta_x;
    x(4:7) = x(4:7)/norm(x(4:7));
end
```

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

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

Example

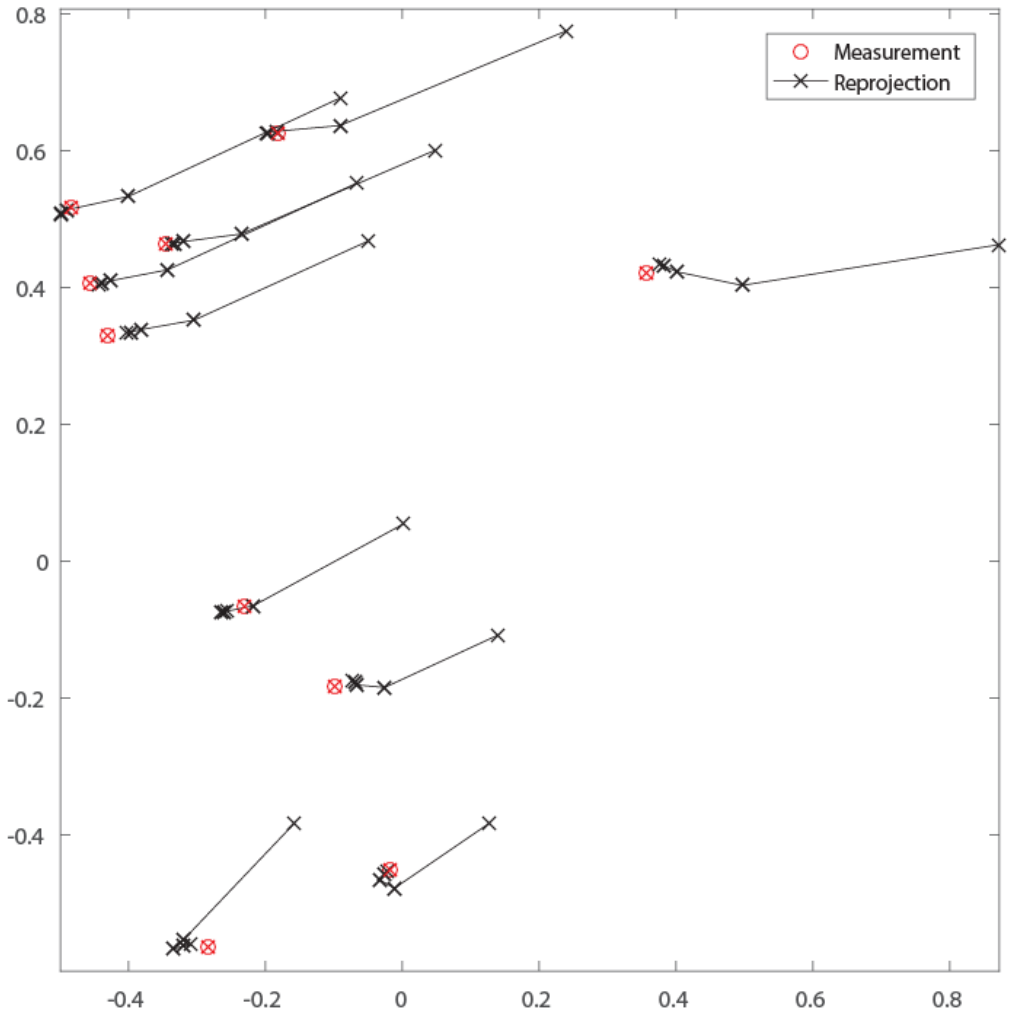



```
u = K*R*[eye(3) -C]*[X'; ones(1,nPoints)];
u = [u(1,:)./u(3,:); u(2,:)./u(3,:)];

x = [C; q];
for j = 1 : 40
    R1 = Quaternion2Rotation(x(4:7));
    C1 = x(1:3);

    df_dc = [];
    df_dR = [];
    for k = 1 : nPoints
        df_dc = [df_dc; JacobianC(K, R1, C1, X(k,:)')];
        df_dR = [df_dR; JacobianR(K, R1, C1, X(k,:)')*JacobianQ(x(4:7))];
    end

    u1 = K*R1*[eye(3) -C1]*[X'; ones(1,nPoints)];
    u1 = [u1(1,:)./u1(3,:); u1(2,:)./u1(3,:)];

    jacobian = [df_dc df_dR];
    delta_b = u(:)-u1(:);

    delta_x = inv(jacobian'*jacobian+lambda*eye(size(jacobian'*jacobian,1))*jacobian'*delta_b
    x = x + delta_x;
    x(4:7) = x(4:7)/norm(x(4:7));
end
```

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

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

Example

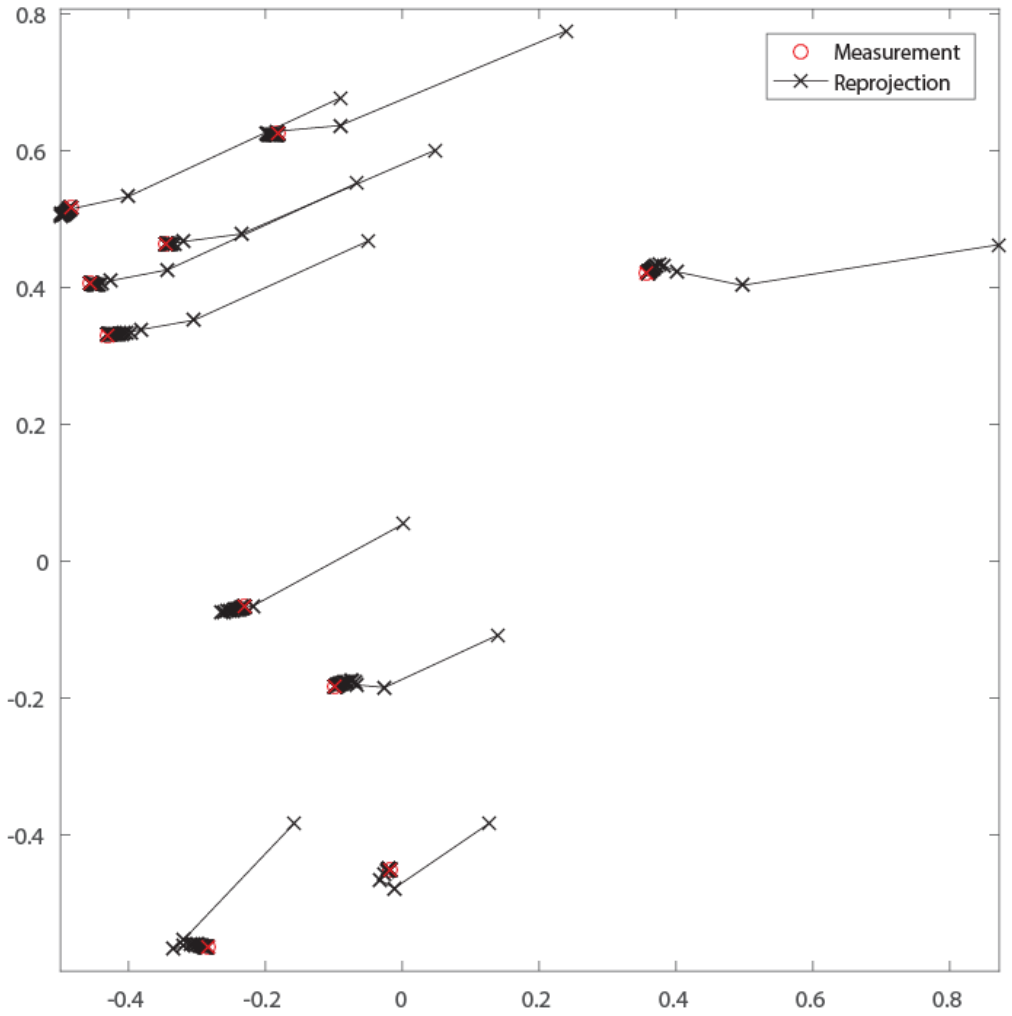



```
u = K*R*[eye(3) -C]*[X'; ones(1,nPoints)];
u = [u(1,:)./u(3,:); u(2,:)./u(3,:)];

x = [C; q];
for j = 1 : 40
    R1 = Quaternion2Rotation(x(4:7));
    C1 = x(1:3);

    df_dc = [];
    df_dR = [];
    for k = 1 : nPoints
        df_dc = [df_dc; JacobianC(K, R1, C1, X(k,:)')];
        df_dR = [df_dR; JacobianR(K, R1, C1, X(k,:)')*JacobianQ(x(4:7))];
    end

    u1 = K*R1*[eye(3) -C1]*[X'; ones(1,nPoints)];
    u1 = [u1(1,:)./u1(3,:); u1(2,:)./u1(3,:)];

    jacobian = [df_dc df_dR];
    delta_b = u(:)-u1(:);

    delta_x = inv(jacobian'*jacobian+lambda*eye(size(jacobian'*jacobian,1))*jacobian'*delta_b
    x = x + delta_x;
    x(4:7) = x(4:7)/norm(x(4:7));
end
```

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

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

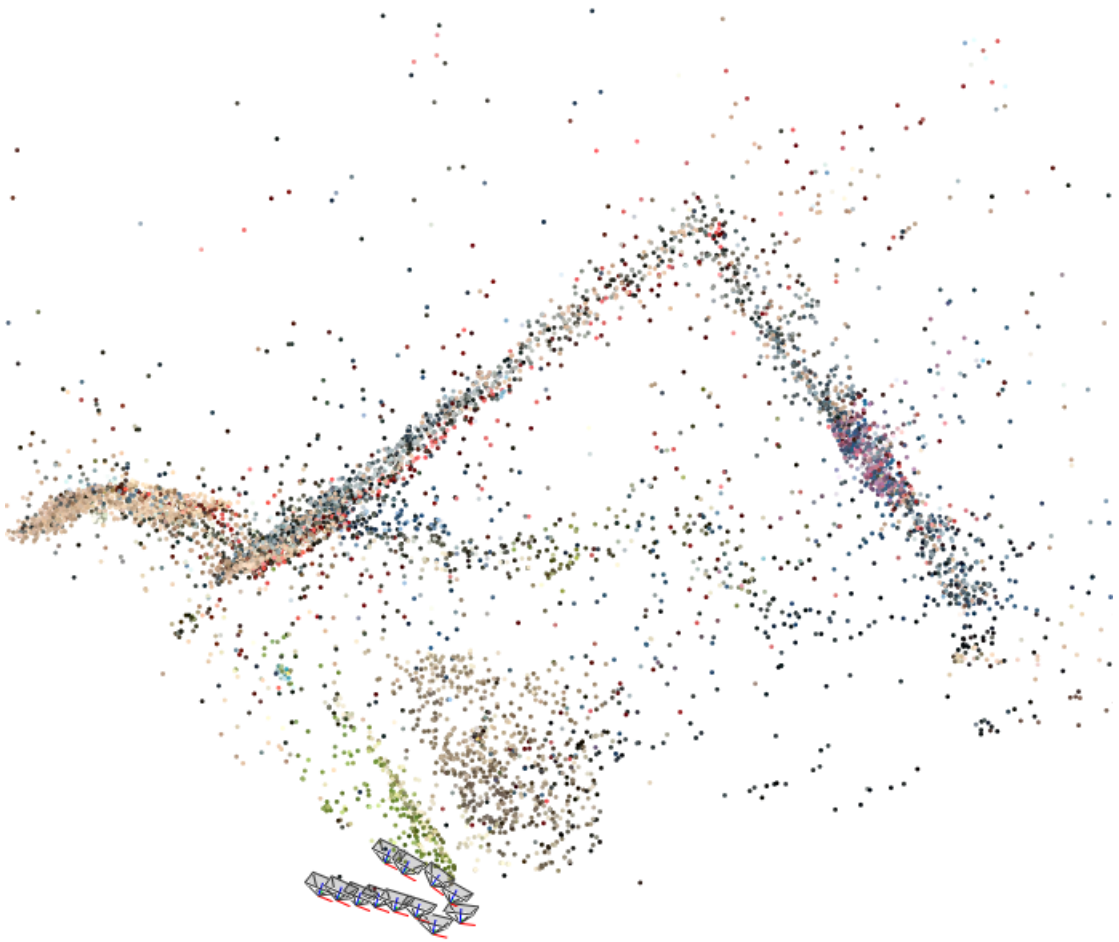
Example




```
u = K*R*[eye(3) -C]*[X'; ones(1,nPoints)];
u = [u(1,:)./u(3,:); u(2,:)./u(3,:)];

x = [C; q];
for j = 1 : 40
    R1 = Quaternion2Rotation(x(4:7));
    C1 = x(1:3);

    df_dc = [];
    df_dR = [];
    for k = 1 : nPoints
        df_dc = [df_dc; JacobianC(K, R1, C1, X(k,:)')];
        df_dR = [df_dR; JacobianR(K, R1, C1, X(k,:)')*JacobianQ(x(4:7))];
    end

    u1 = K*R1*[eye(3) -C1]*[X'; ones(1,nPoints)];
    u1 = [u1(1,:)./u1(3,:); u1(2,:)./u1(3,:)];

    jacobian = [df_dc df_dR];
    delta_b = u(:)-u1(:);

    delta_x = inv(jacobian'*jacobian+lambda*eye(size(jacobian'*jacobian,1))*jacobian'*delta_b
    x = x + delta_x;
    x(4:7) = x(4:7)/norm(x(4:7));
end
```

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

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



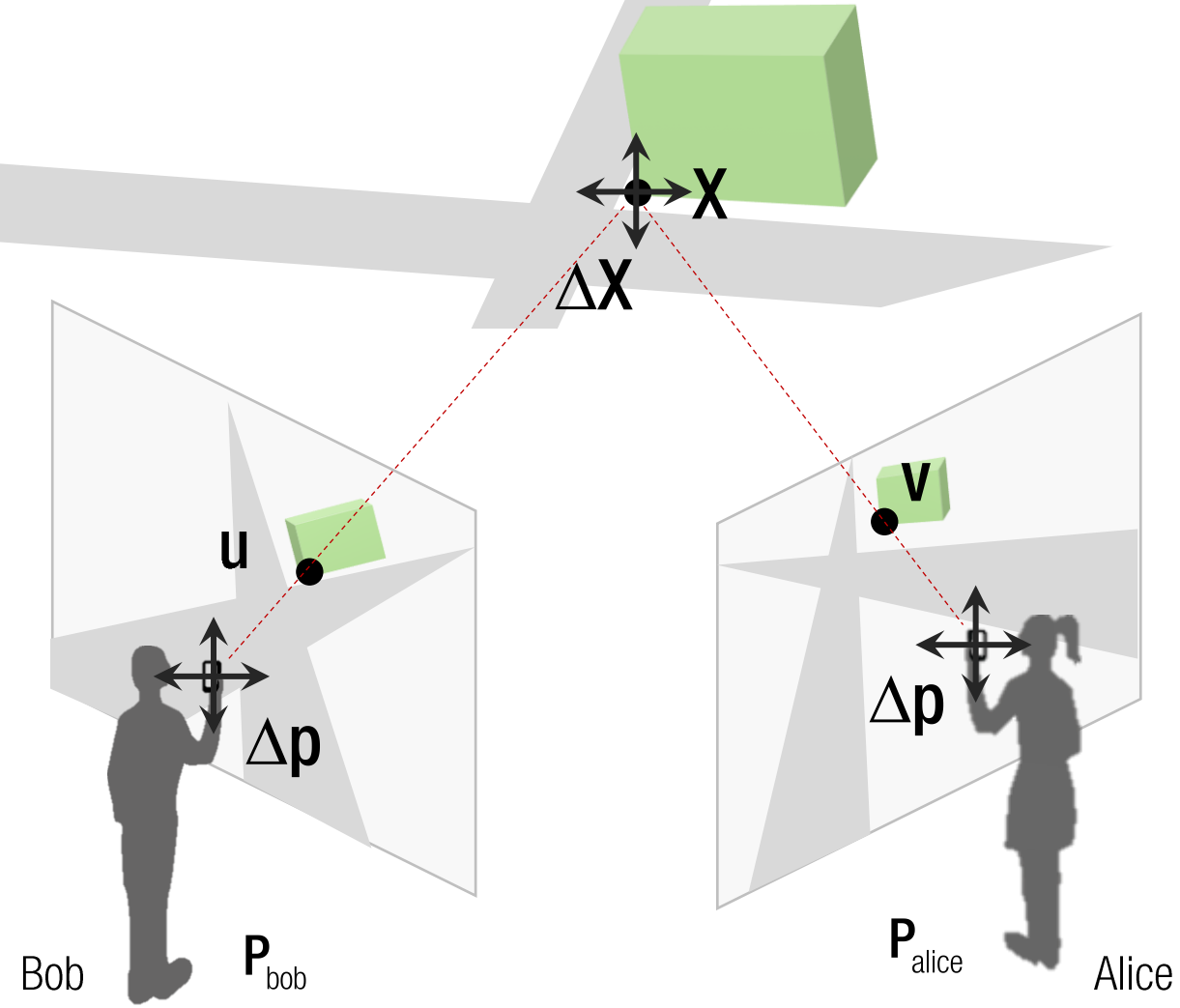
Bundle Adjustment

Black: given variables
Red: unknowns

Camera & Point Jacobian

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

$$= \left(\frac{\mathbf{P}_1\mathbf{X}}{\mathbf{P}_3\mathbf{X}} - x \right)^2 + \left(\frac{\mathbf{P}_2\mathbf{X}}{\mathbf{P}_3\mathbf{X}} - y \right)^2$$

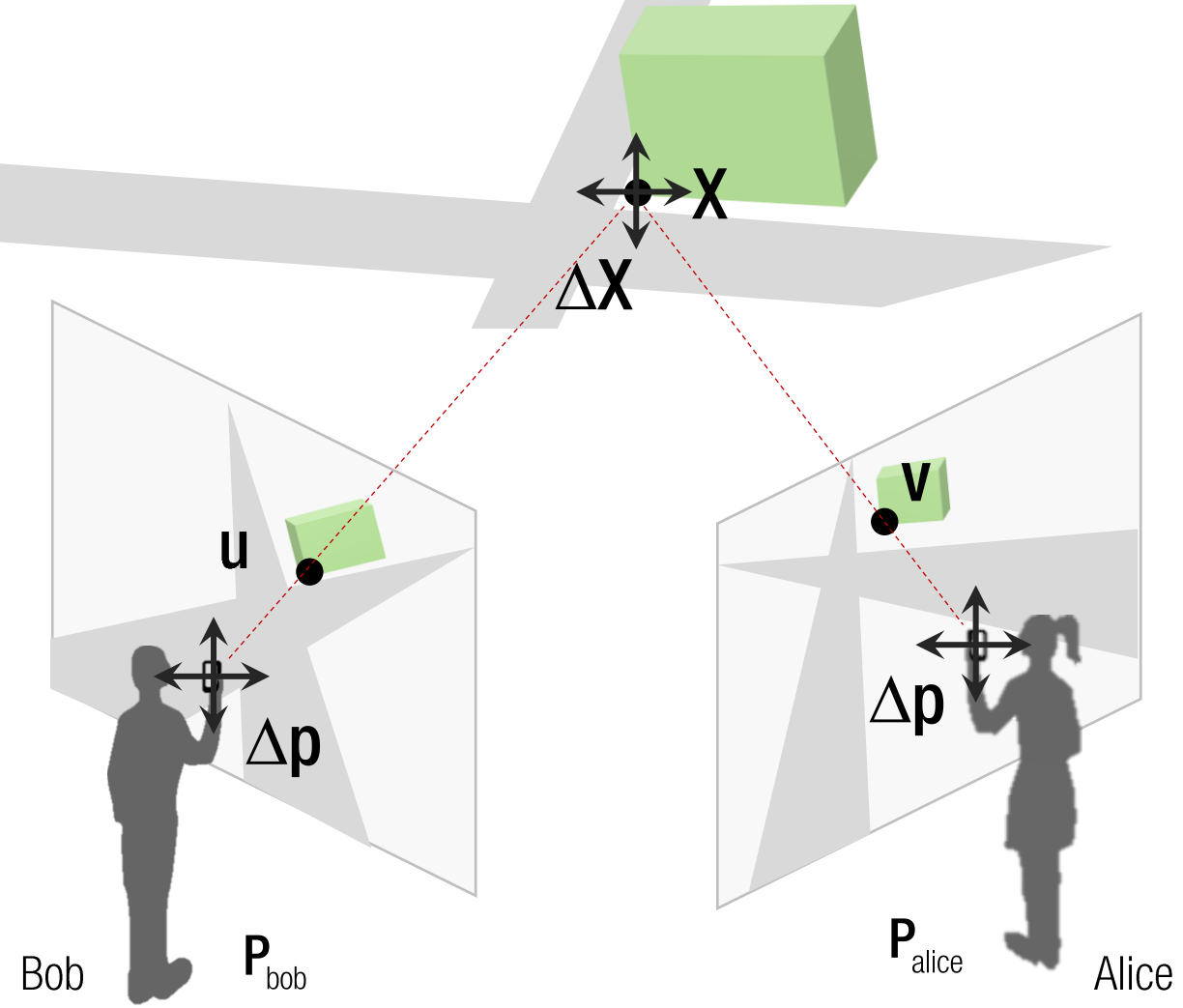


Black: given variables
Red: unknowns

Camera & Point Jacobian

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

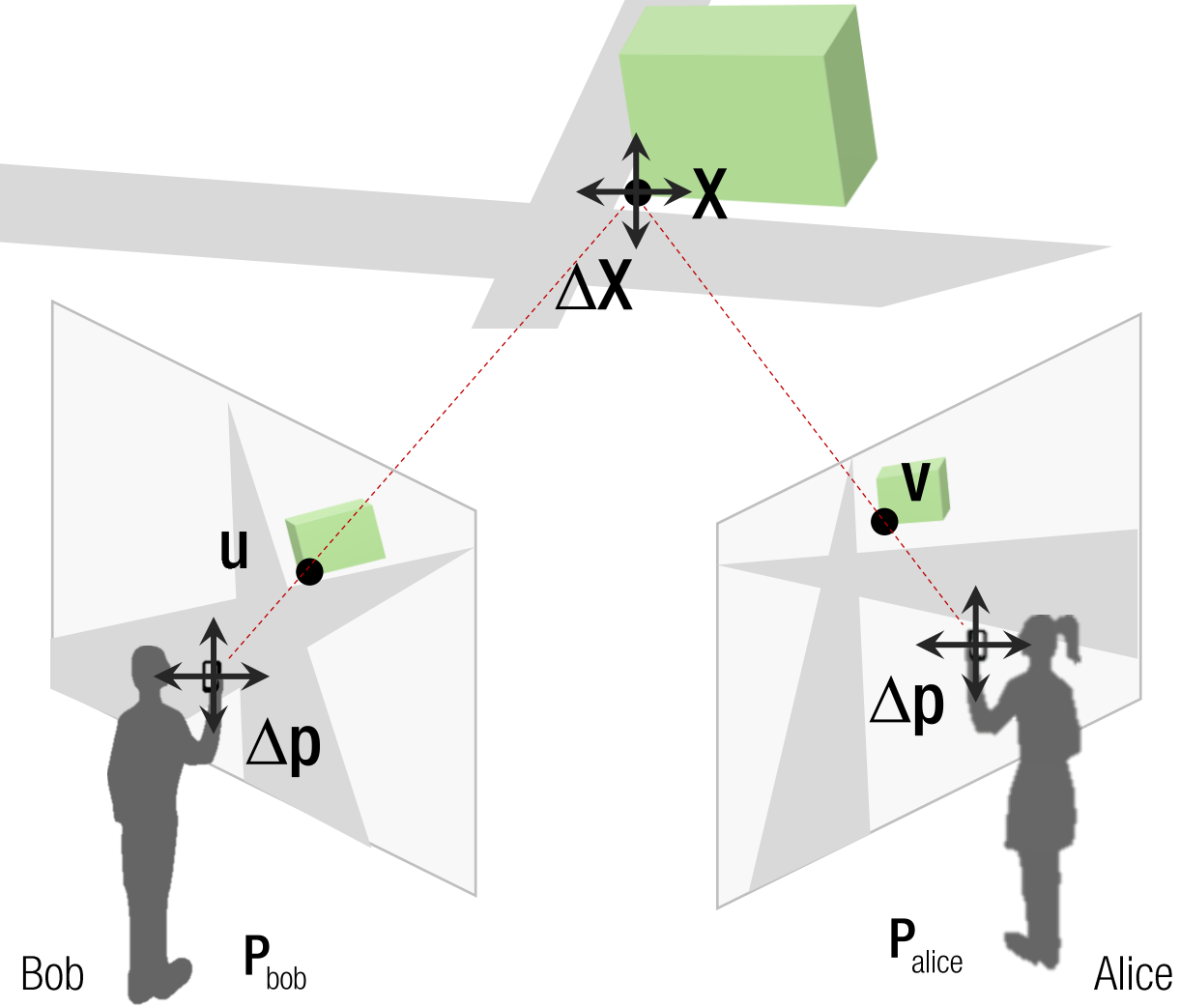
$$= \left(\frac{\mathbf{P}_1\mathbf{X}}{\mathbf{P}_3\mathbf{X}} - x\right)^2 + \left(\frac{\mathbf{P}_2\mathbf{X}}{\mathbf{P}_3\mathbf{X}} - y\right)^2$$



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

Black: given variables
Red: unknowns

Camera & Point Jacobian



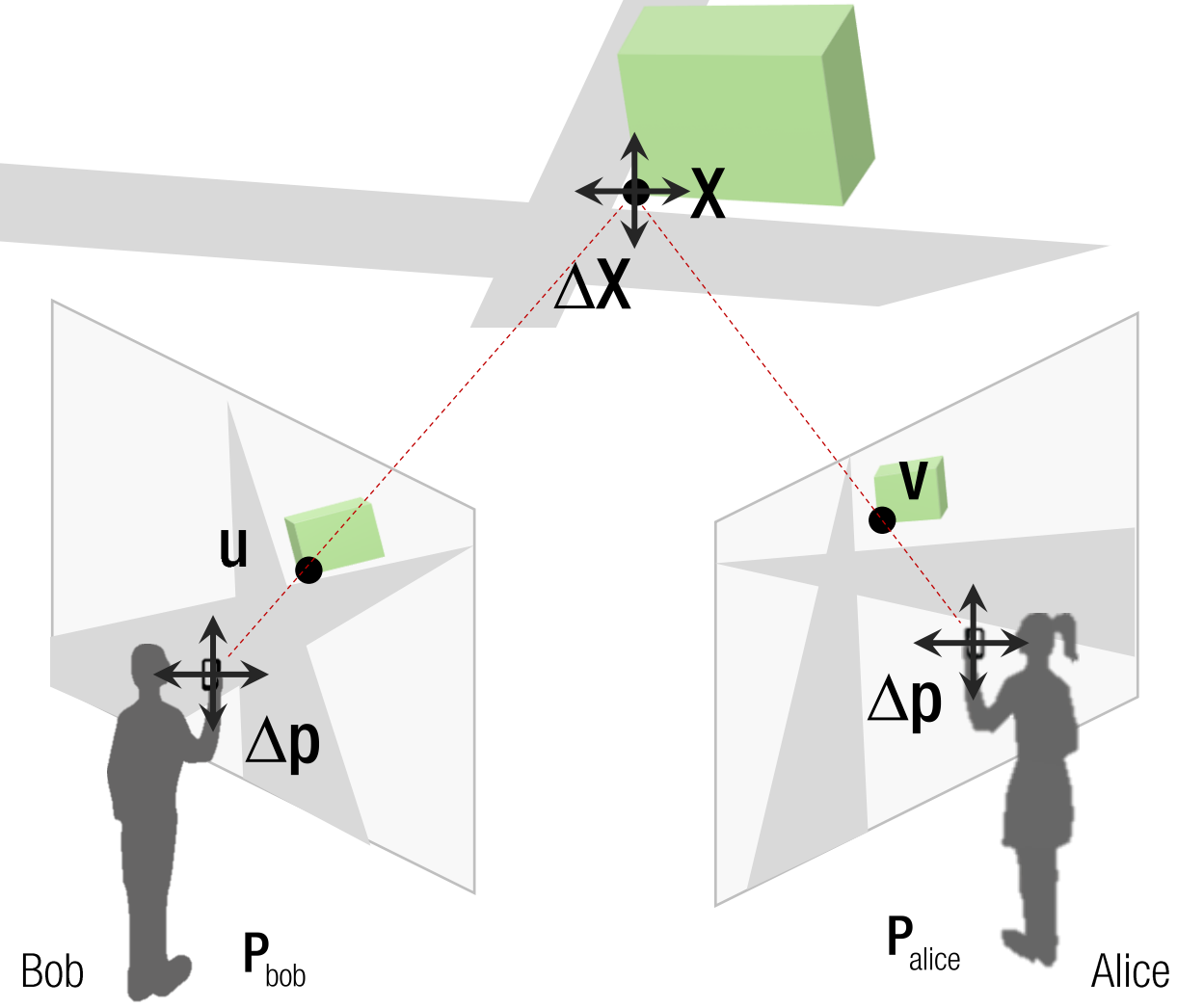
$$E_{\text{geom}} = \left\| \hat{\mathbf{u}} - \mathbf{u} \right\|^2$$

$$= \left(\frac{\mathbf{P}_1\mathbf{X}}{\mathbf{P}_3\mathbf{X}} - x \right)^2 + \left(\frac{\mathbf{P}_2\mathbf{X}}{\mathbf{P}_3\mathbf{X}} - y \right)^2$$

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

Black: given variables
Red: unknowns

Camera & Point Jacobian



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

$$= \left(\frac{\mathbf{P}_1\mathbf{X}}{\mathbf{P}_3\mathbf{X}} - x\right)^2 + \left(\frac{\mathbf{P}_2\mathbf{X}}{\mathbf{P}_3\mathbf{X}} - y\right)^2$$

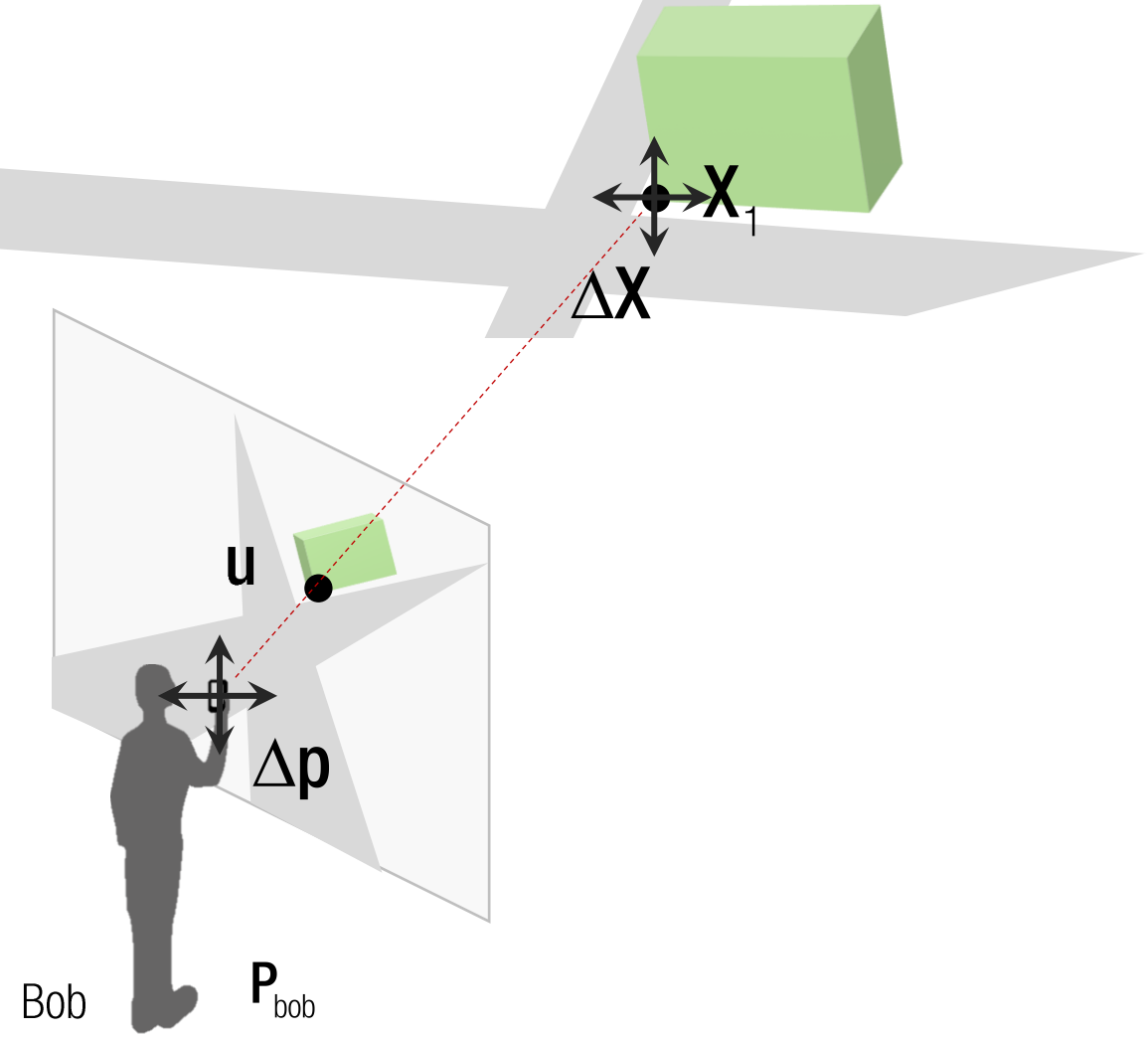
$$f(\mathbf{p},\mathbf{X}) = \begin{bmatrix} \frac{u}{w} \\ \frac{v}{w} \end{bmatrix} \longrightarrow \frac{\partial f(\mathbf{p},\mathbf{X})}{\partial \mathbf{p}} = \frac{\partial}{\partial \mathbf{p}} \begin{bmatrix} \frac{u}{w} \\ \frac{v}{w} \end{bmatrix} = \begin{bmatrix} \frac{w\frac{\partial u}{\partial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

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

Black: given variables
Red: unknowns

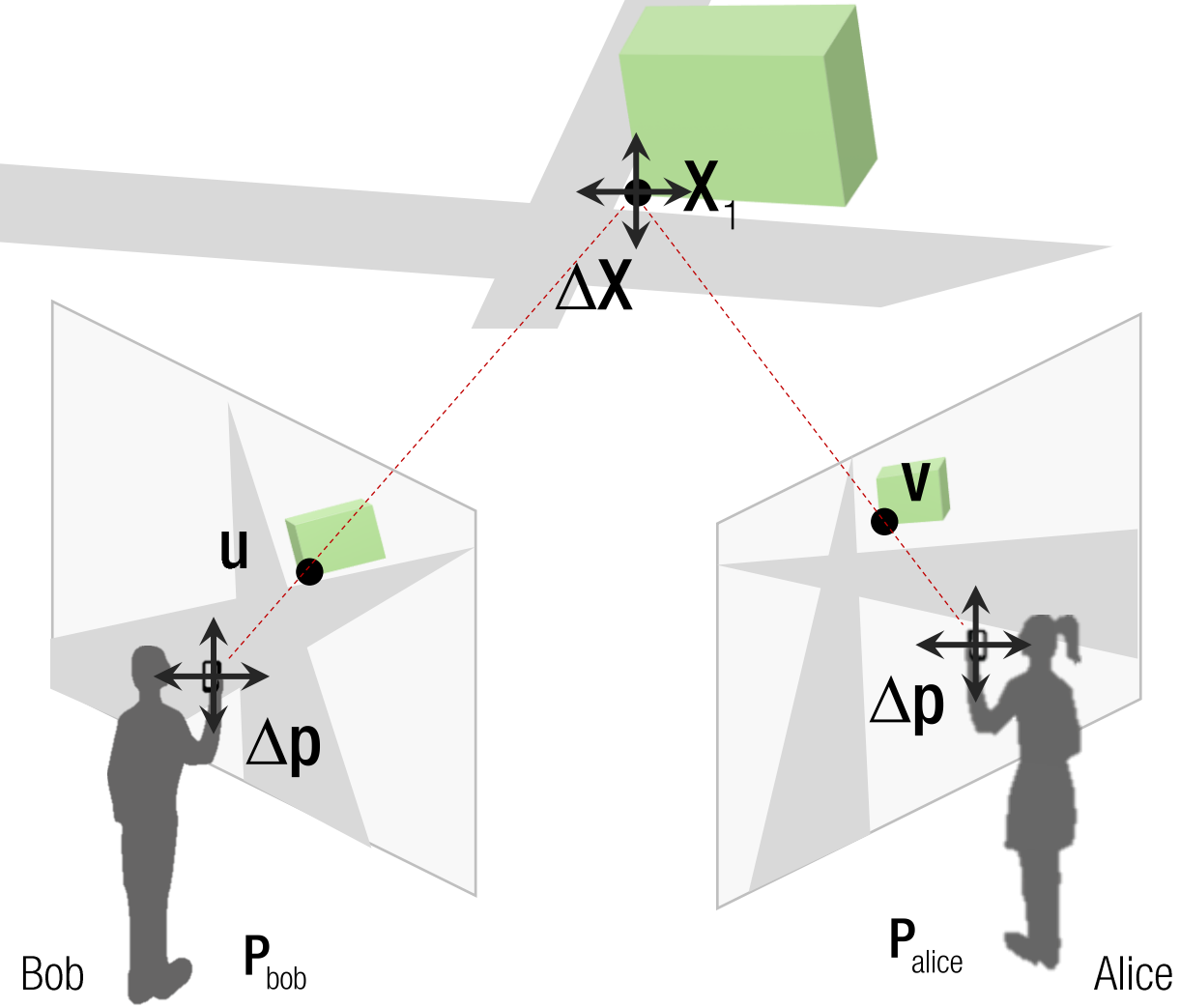
Camera & Point Jacobian

$$\mathbf{J}_{ij} = \left[\frac{\partial f(\mathbf{p}_j, \mathbf{X}_i)}{\partial \mathbf{p}_j} \quad \frac{\partial f(\mathbf{p}_j, \mathbf{X}_i)}{\partial \mathbf{X}_i} \right] = \left[\mathbf{J}_{\mathbf{p}ij} \quad \mathbf{J}_{\mathbf{X}ij} \right]$$



Black: given variables
Red: unknowns

Camera & Point Jacobian

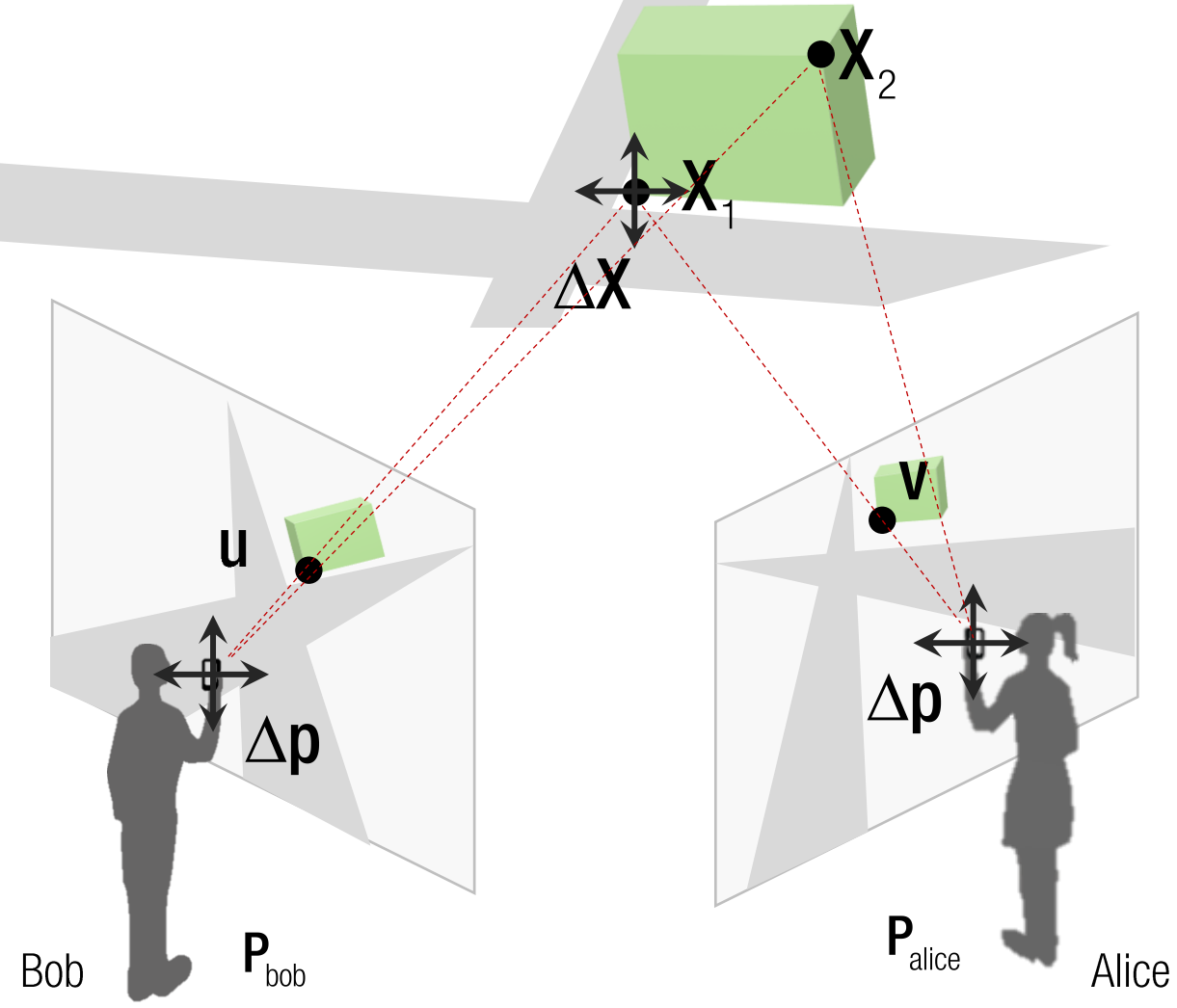


$$\mathbf{J}_{ij} = \left[\frac{\partial f(\mathbf{p}_j, \mathbf{X}_i)}{\partial \mathbf{p}_j} \quad \frac{\partial f(\mathbf{p}_j, \mathbf{X}_i)}{\partial \mathbf{X}_i} \right] = \left[\mathbf{J}_{\mathbf{p}ij} \quad \mathbf{J}_{\mathbf{X}ij} \right]$$

$$\mathbf{J} = \begin{bmatrix} \mathbf{J}_{\mathbf{p}1,bob} & \mathbf{0}_{2\times7} & \mathbf{J}_{\mathbf{X}1,bob} \\ \mathbf{0}_{2\times7} & \mathbf{J}_{\mathbf{p}1,alice} & \mathbf{J}_{\mathbf{X}1,alice} \end{bmatrix}$$

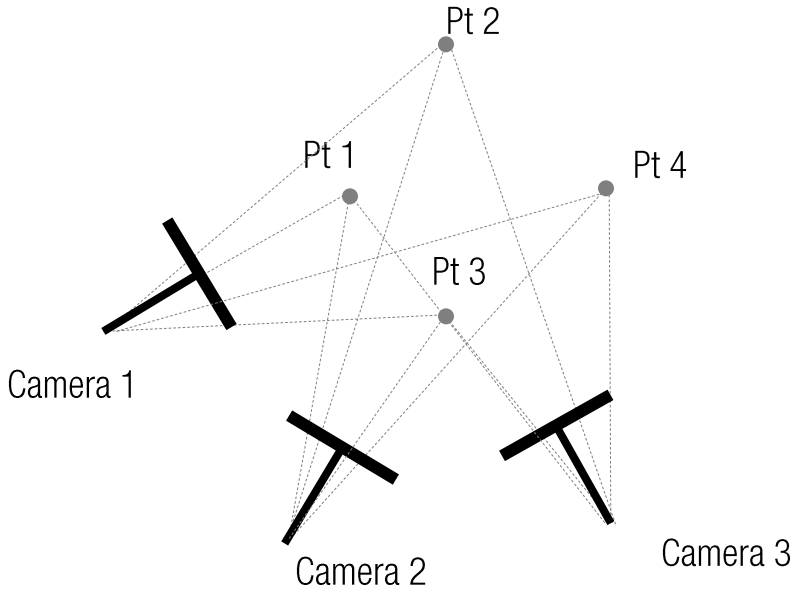
Black: given variables
Red: unknowns

Camera & Point Jacobian

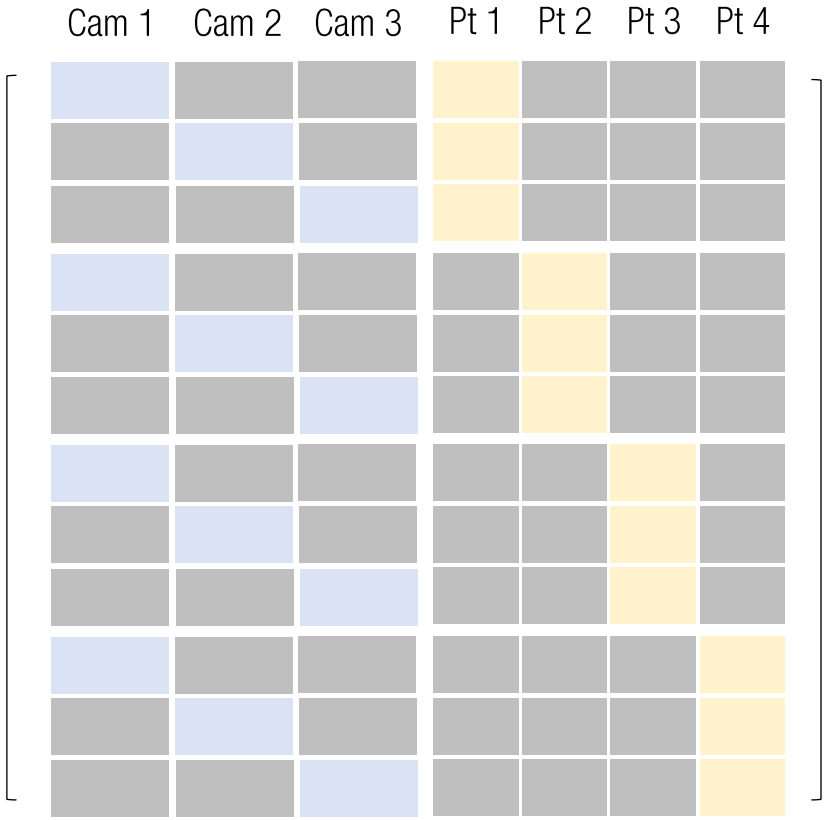


$$\mathbf{J}_{ij} = \left[\frac{\partial f(\mathbf{p}_j, \mathbf{X}_i)}{\partial \mathbf{p}_j} \quad \frac{\partial f(\mathbf{p}_j, \mathbf{X}_i)}{\partial \mathbf{X}_i} \right] = \begin{bmatrix} \mathbf{J}_{\mathbf{p}ij} & \mathbf{J}_{\mathbf{X}ij} \end{bmatrix}$$

$$\mathbf{J} = \begin{bmatrix} \mathbf{J}_{\mathbf{p}1,\text{bob}} & \mathbf{0}_{2\times7} & \mathbf{J}_{\mathbf{X}1,\text{bob}} & \mathbf{0}_{2\times3} \\ \mathbf{0}_{2\times7} & \mathbf{J}_{\mathbf{p}1,\text{alice}} & \mathbf{J}_{\mathbf{X}1,\text{alice}} & \mathbf{0}_{2\times3} \\ \mathbf{J}_{\mathbf{p}2,\text{bob}} & \mathbf{0}_{2\times7} & \mathbf{0}_{2\times3} & \mathbf{J}_{\mathbf{X}2,\text{bob}} \\ \mathbf{0}_{2\times7} & \mathbf{J}_{\mathbf{p}2,\text{alice}} & \mathbf{0}_{2\times3} & \mathbf{J}_{\mathbf{X}2,\text{alice}} \end{bmatrix}$$

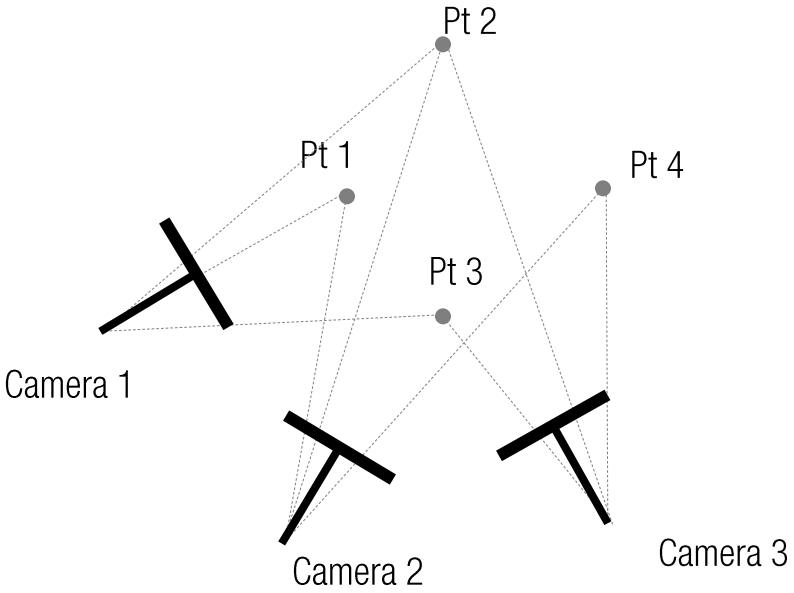


J =

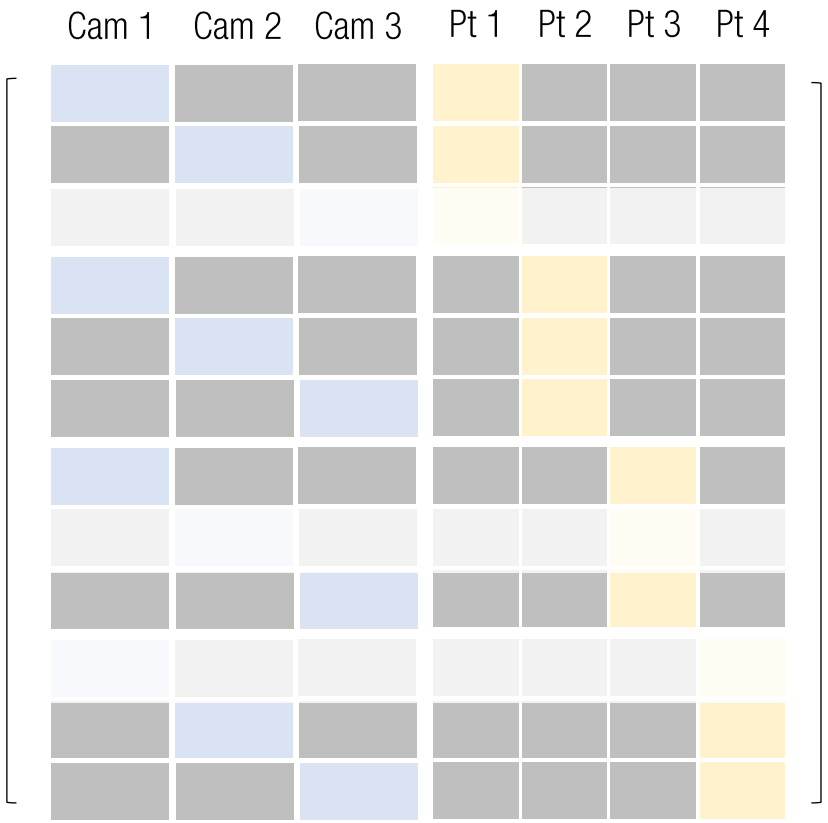


# of unknowns: 3x7+4x3

# of projections: 3x4

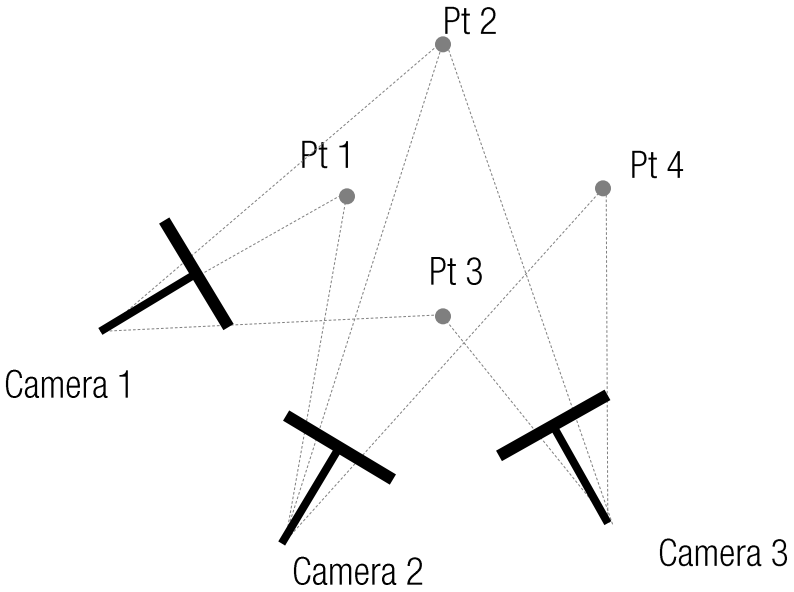


$J =$

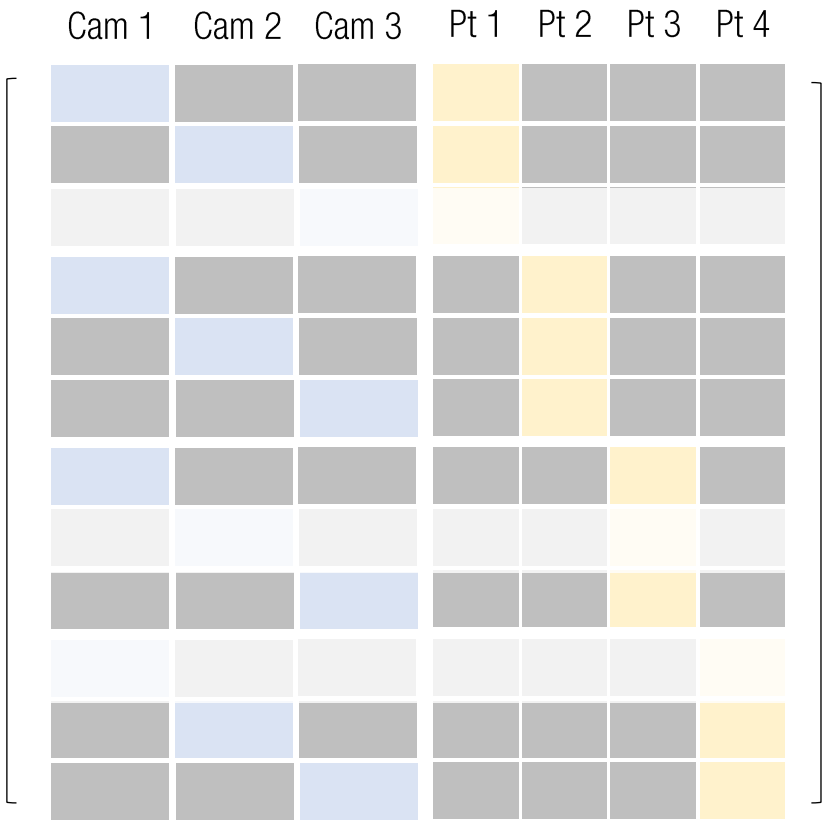


# of unknowns: 3x7+4x3

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



$\mathbf{J} =$

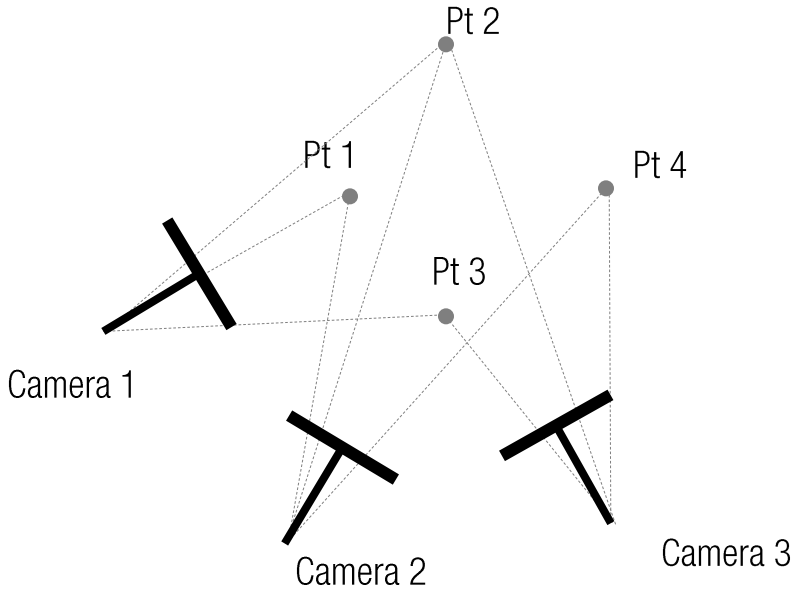


# of unknowns: 3x7+4x3

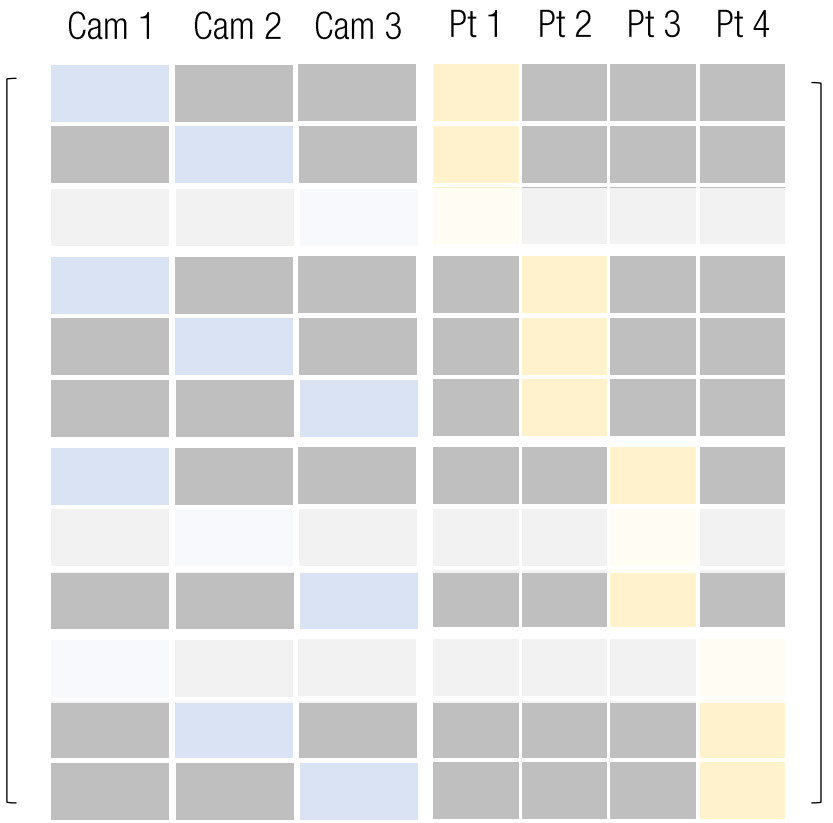
# of projections: 9 (not all points are visible from cameras)

$$\Delta \mathbf{x} = \left(\mathbf{J}^{\top}\mathbf{J}\right)^{-1}\mathbf{J}^{\top}\left(\mathbf{b} - \mathbf{f(x)}\right)$$

size of $\mathbf{J}^{\top}\mathbf{J}$: 24x24



$\mathbf{J} =$



# of unknowns: 3x7+4x3

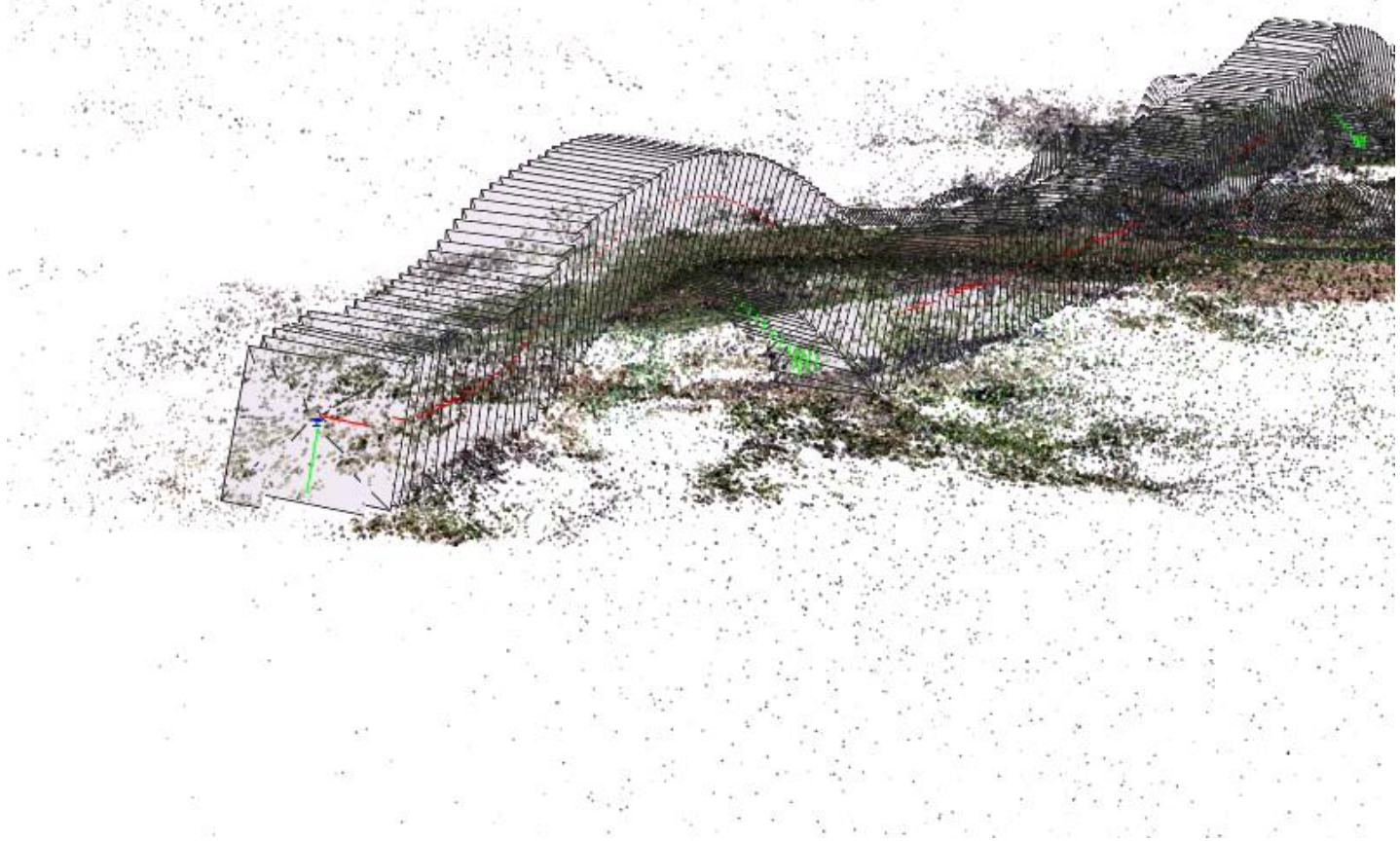
# of projections: 9 (not all points are visible from cameras)

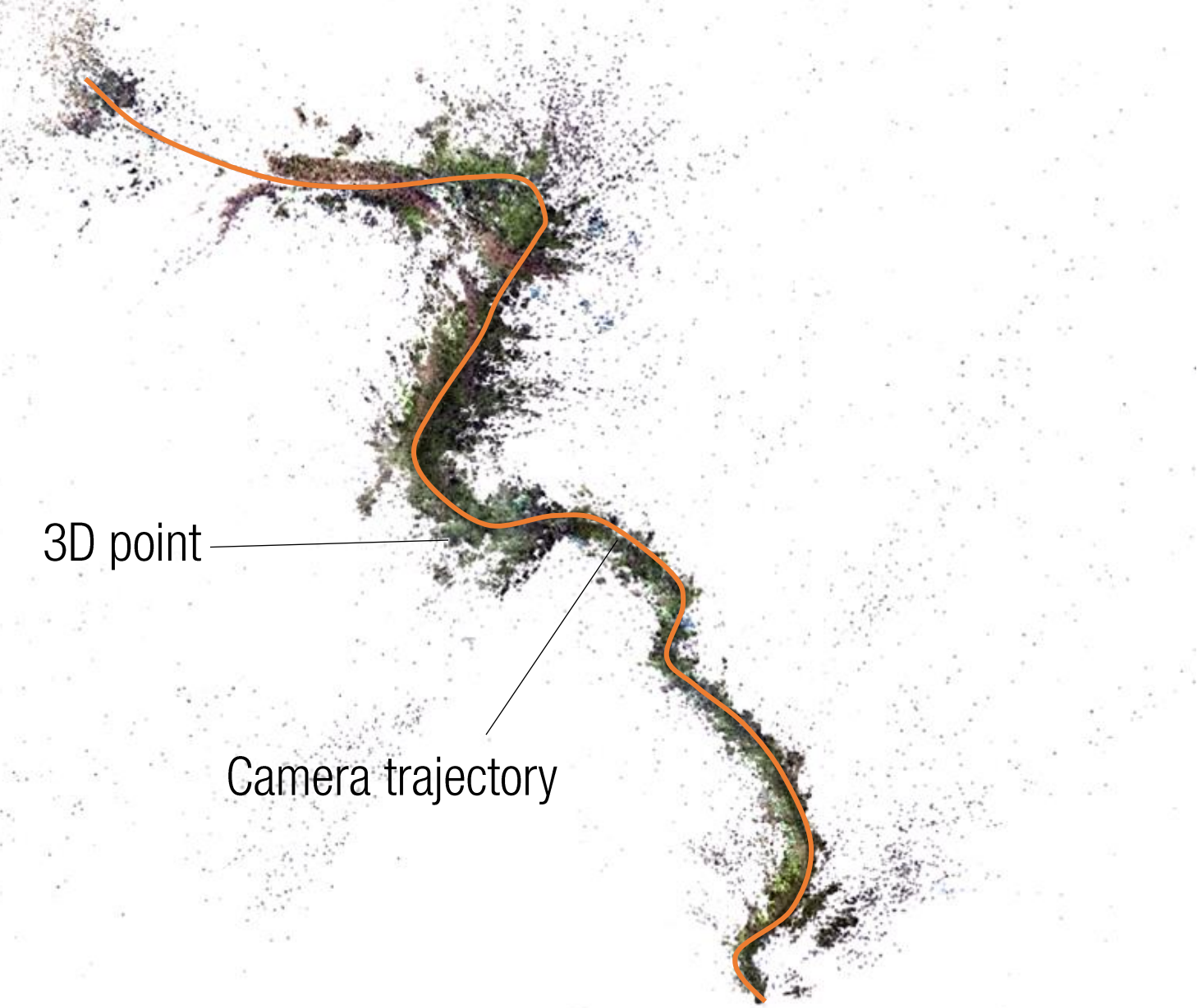
$$\Delta \mathbf{x} = \underline{\left(\mathbf{J}^{\top}\mathbf{J}\right)^{-1}} \mathbf{J}^{\top}\left(\mathbf{b} - \mathbf{f(x)}\right)$$

Main computational bottle neck

size of $\mathbf{J}^{\top}\mathbf{J}$: 24x24


Input: first person video
https://www.youtube.com/watch?v=bKU0Z4bj7Mw

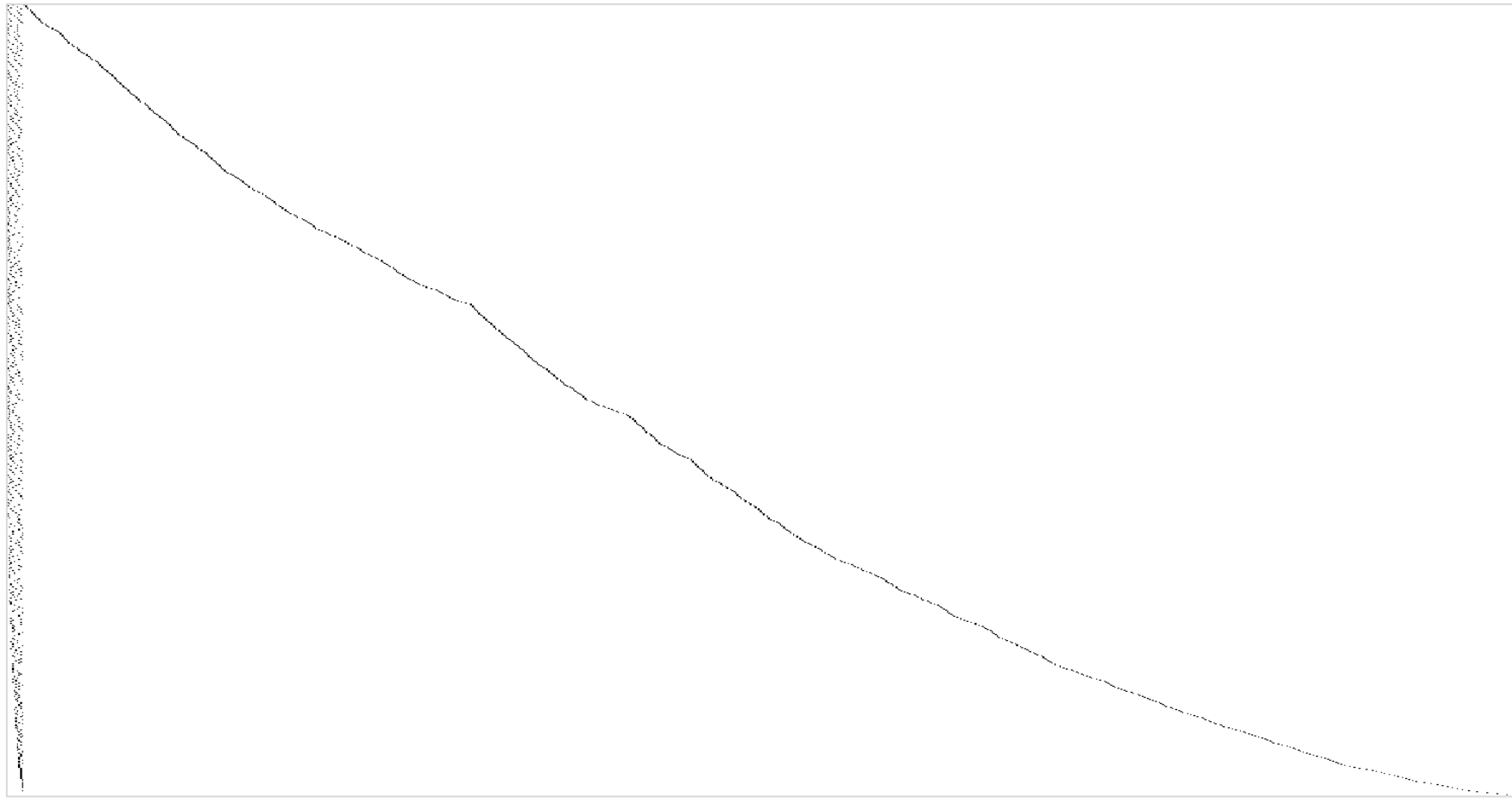




# of cameras: 3009
# of 3D points: 1,298,317

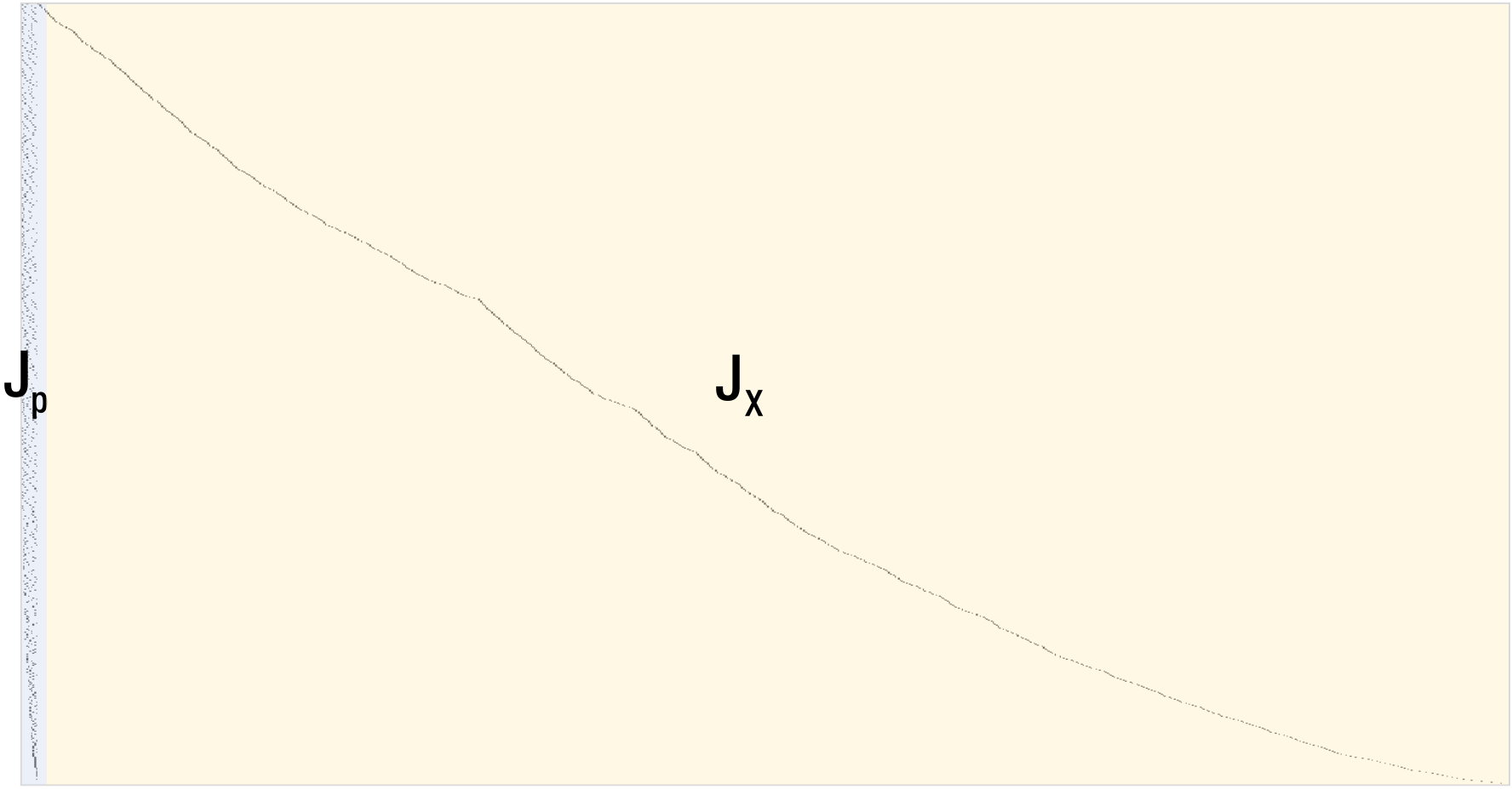
# of unknowns: 7x3009+3x1,298,317
size of $\mathbf{J}^{\top}\mathbf{J}$: 6,167,690x6,167,690

$$J =$$

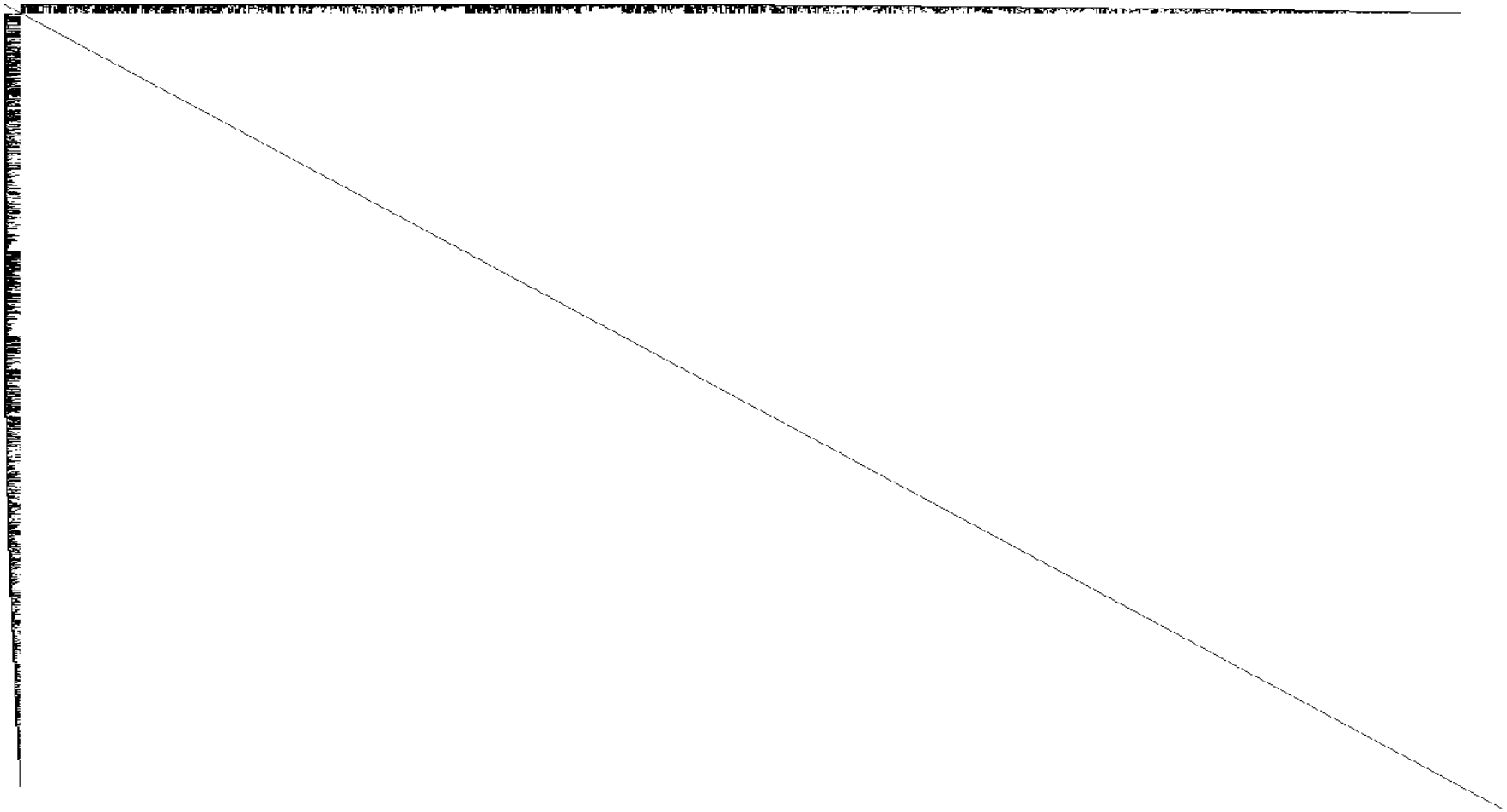


Camera 3D point

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



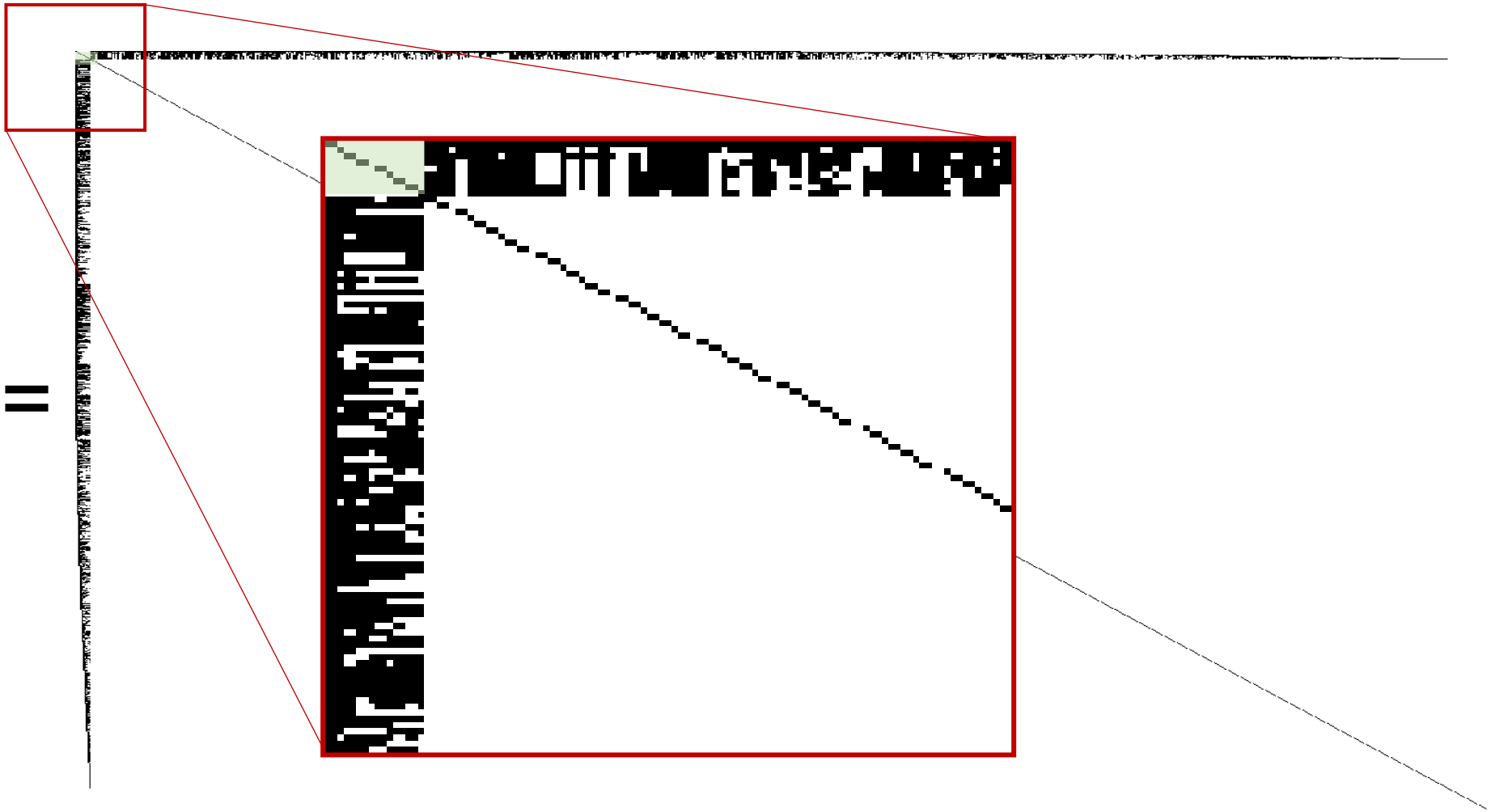
$$J^\top J = \quad$$



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

$$J^\top J = \begin{bmatrix} J_p^\top J_p & J_p^\top J_x \\ J_x^\top J_p & J_x^\top J_x \end{bmatrix}$$

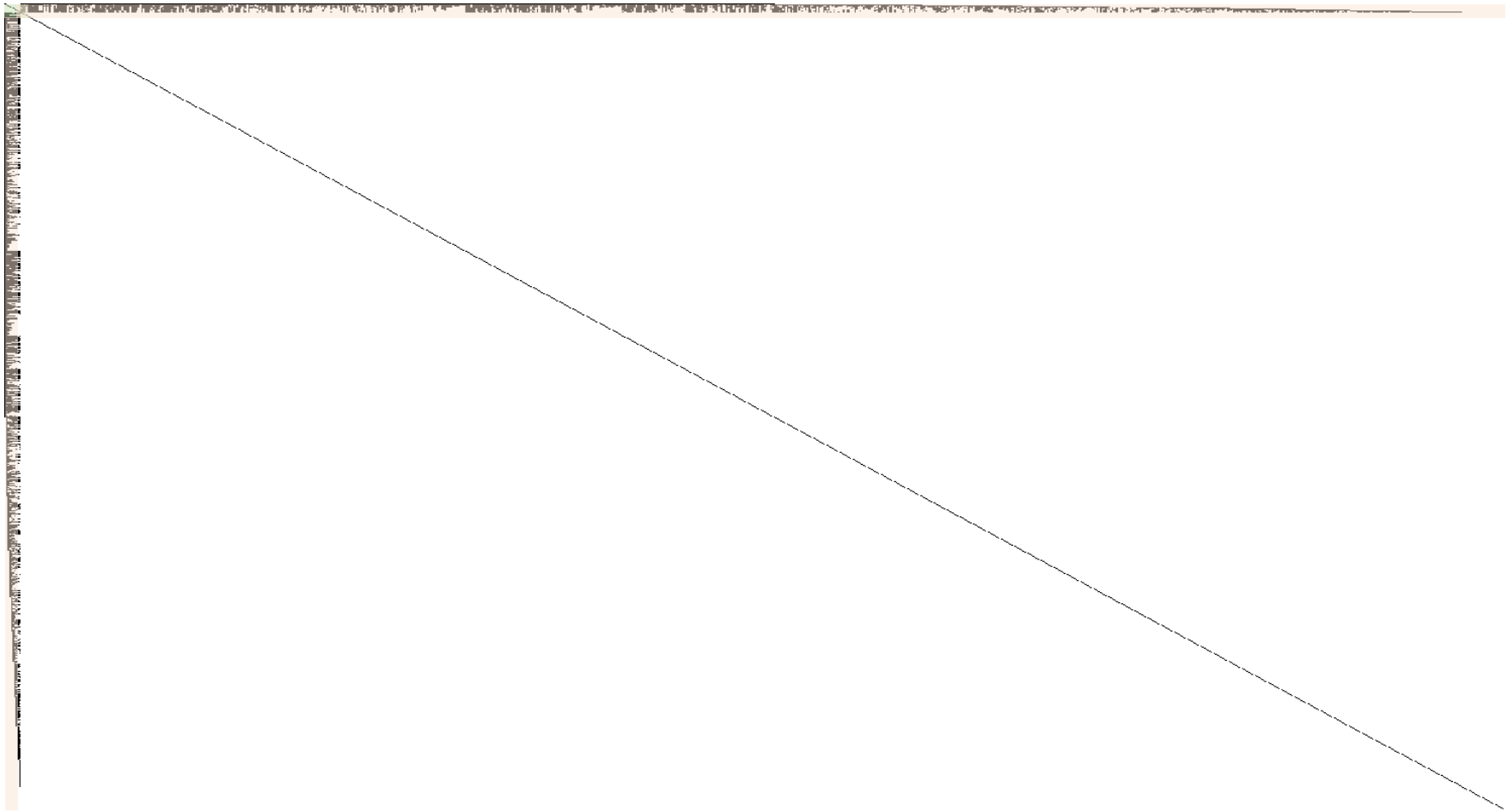
$J^\top J =$



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

$$J^\top J = \begin{bmatrix} J_p^\top J_p & J_p^\top J_x \\ J_x^\top J_p & J_x^\top J_x \end{bmatrix} = \begin{bmatrix} A & B \\ B^\top & D \end{bmatrix}$$

$$J^\top J =$$

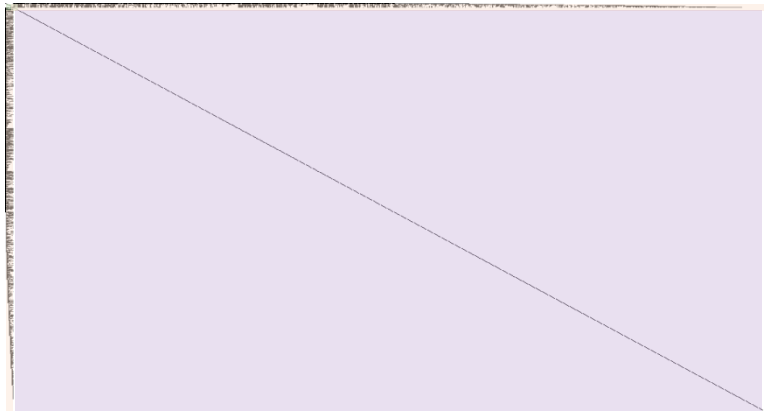


$$J = \begin{bmatrix} J_p & J_x \end{bmatrix}$$

$$J^\top J = \begin{bmatrix} J_p^\top J_p & J_p^\top J_x \\ J_x^\top J_p & J_x^\top J_x \end{bmatrix} = \begin{bmatrix} A & B \\ B^\top & D \end{bmatrix}$$

Normal equation:

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

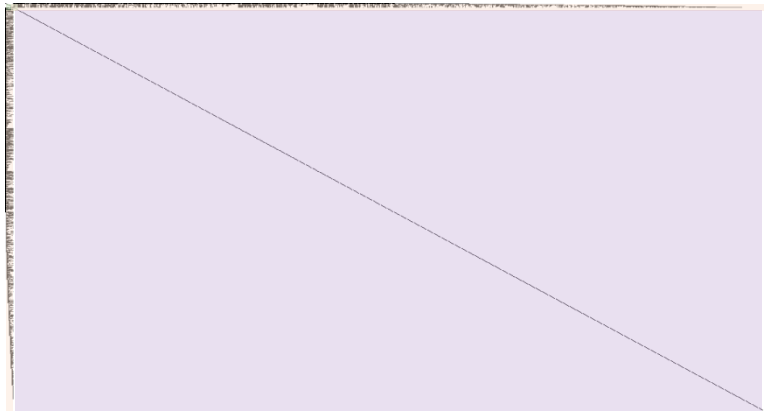
$\mathbf{J}^{\mathsf{T}}\mathbf{J} =$ 

$$\mathbf{J}^{\mathsf{T}}\mathbf{J}=\begin{bmatrix}\mathbf{J}_p^{\mathsf{T}}\mathbf{J}_p & \mathbf{J}_p^{\mathsf{T}}\mathbf{J}_X\\ \mathbf{J}_X^{\mathsf{T}}\mathbf{J}_p & \mathbf{J}_X^{\mathsf{T}}\mathbf{J}_X\end{bmatrix}=\begin{bmatrix}\mathbf{A} & \mathbf{B}\\ \mathbf{B}^{\mathsf{T}} & \mathbf{D}\end{bmatrix}$$

Normal equation:

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

$$\mathbf{J}^{\top}\mathbf{J}=$$

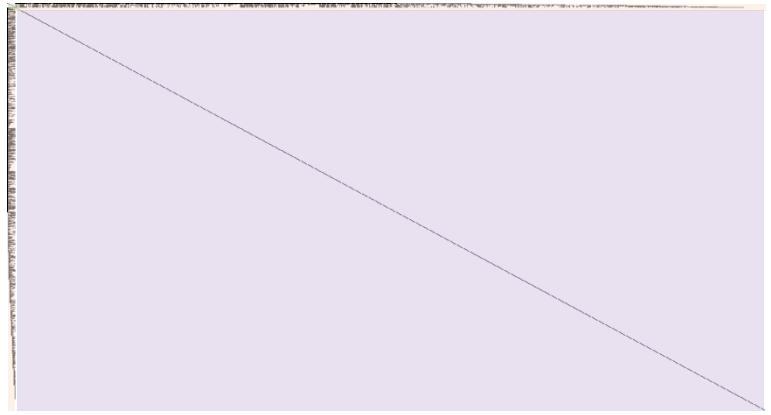


$$\rightarrow\begin{bmatrix}\mathbf{J}_p^{\top}\mathbf{J}_p & \mathbf{J}_p^{\top}\mathbf{J}_X\\ \mathbf{J}_X^{\top}\mathbf{J}_p & \mathbf{J}_X^{\top}\mathbf{J}_X\end{bmatrix}\begin{bmatrix}\Delta\mathbf{p}\\ \Delta\mathbf{X}\end{bmatrix}=\begin{bmatrix}\mathbf{J}_p^{\top}\\ \mathbf{J}_X^{\top}\end{bmatrix}\left(\mathbf{b}-f(\mathbf{X})\right)=\begin{bmatrix}\mathbf{e}_p\\ \mathbf{e}_X\end{bmatrix}$$

$$\mathbf{J}^{\top}\mathbf{J}=\begin{bmatrix}\mathbf{J}_p^{\top}\mathbf{J}_p & \mathbf{J}_p^{\top}\mathbf{J}_X\\ \mathbf{J}_X^{\top}\mathbf{J}_p & \mathbf{J}_X^{\top}\mathbf{J}_X\end{bmatrix}=\begin{bmatrix}\mathbf{A} & \mathbf{B}\\ \mathbf{B}^{\top} & \mathbf{D}\end{bmatrix}$$

Normal equation:

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

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

$$\longrightarrow \begin{bmatrix}\mathbf{J}_p^T\mathbf{J}_p & \mathbf{J}_p^T\mathbf{J}_X\\ \mathbf{J}_X^T\mathbf{J}_p & \mathbf{J}_X^T\mathbf{J}_X\end{bmatrix}\begin{bmatrix}\Delta\mathbf{p}\\ \Delta\mathbf{X}\end{bmatrix} = \begin{bmatrix}\mathbf{J}_p^T\\ \mathbf{J}_X^T\end{bmatrix}\left(\mathbf{b} - f(\mathbf{X})\right) = \begin{bmatrix}\mathbf{e}_p\\ \mathbf{e}_X\end{bmatrix}$$

or

$$\begin{bmatrix}\mathbf{J}_p^T\mathbf{J}_p + \mu\mathbf{I} & \mathbf{J}_p^T\mathbf{J}_X\\ \mathbf{J}_X^T\mathbf{J}_p & \mathbf{J}_X^T\mathbf{J}_X + \mu\mathbf{I}\end{bmatrix}\begin{bmatrix}\Delta\mathbf{p}\\ \Delta\mathbf{X}\end{bmatrix} = \begin{bmatrix}\mathbf{e}_p\\ \mathbf{e}_X\end{bmatrix}$$

or

$$\mathbf{J}^T\mathbf{J} = \begin{bmatrix}\mathbf{J}_p^T\mathbf{J}_p & \mathbf{J}_p^T\mathbf{J}_X\\ \mathbf{J}_X^T\mathbf{J}_p & \mathbf{J}_X^T\mathbf{J}_X\end{bmatrix} = \begin{bmatrix}\mathbf{A} & \mathbf{B}\\ \mathbf{B}^T & \mathbf{D}\end{bmatrix}$$

$$\begin{bmatrix}\mathbf{A} & \mathbf{B}\\ \mathbf{B}^T & \mathbf{D}\end{bmatrix}\begin{bmatrix}\Delta\mathbf{p}\\ \Delta\mathbf{X}\end{bmatrix} = \begin{bmatrix}\mathbf{e}_p\\ \mathbf{e}_X\end{bmatrix}$$

Normal equation:

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

$$\mathbf{J}^{\mathsf{T}}\mathbf{J}=\begin{bmatrix}\mathbf{J}_p^{\mathsf{T}}\mathbf{J}_p & \mathbf{J}_p^{\mathsf{T}}\mathbf{J}_X\\ \mathbf{J}_X^{\mathsf{T}}\mathbf{J}_p & \mathbf{J}_X^{\mathsf{T}}\mathbf{J}_X\end{bmatrix}=\begin{bmatrix}\mathbf{A} & \mathbf{B}\\ \mathbf{B}^{\mathsf{T}} & \mathbf{D}\end{bmatrix}$$

$$\mathbf{J}^{\mathsf{T}}\mathbf{J}=\begin{bmatrix}\mathbf{d}_1 & & \\ & \ddots & \\ & & \mathbf{d}_M\end{bmatrix}=\mathbf{D}\longrightarrow\mathbf{D}^{-1}=\begin{bmatrix}\mathbf{d}_1^{-1} & & \\ & \ddots & \\ & & \mathbf{d}_M^{-1}\end{bmatrix}$$

Inversion of block diagonal matrix can be efficiently computed.

$$\longrightarrow\begin{bmatrix}\mathbf{J}_p^{\mathsf{T}}\mathbf{J}_p & \mathbf{J}_p^{\mathsf{T}}\mathbf{J}_X\\ \mathbf{J}_X^{\mathsf{T}}\mathbf{J}_p & \mathbf{J}_X^{\mathsf{T}}\mathbf{J}_X\end{bmatrix}\begin{bmatrix}\Delta\mathbf{p}\\ \Delta\mathbf{X}\end{bmatrix}=\begin{bmatrix}\mathbf{J}_p^{\mathsf{T}}\\ \mathbf{J}_X^{\mathsf{T}}\end{bmatrix}\left(\mathbf{b}-f(\mathbf{X})\right)=\begin{bmatrix}\mathbf{e}_p\\ \mathbf{e}_X\end{bmatrix}$$

or

$$\begin{bmatrix}\mathbf{J}_p^{\mathsf{T}}\mathbf{J}_p+\mu\mathbf{I} & \mathbf{J}_p^{\mathsf{T}}\mathbf{J}_X\\ \mathbf{J}_X^{\mathsf{T}}\mathbf{J}_p & \mathbf{J}_X^{\mathsf{T}}\mathbf{J}_X+\mu\mathbf{I}\end{bmatrix}\begin{bmatrix}\Delta\mathbf{p}\\ \Delta\mathbf{X}\end{bmatrix}=\begin{bmatrix}\mathbf{e}_p\\ \mathbf{e}_X\end{bmatrix}$$

or

$$\begin{bmatrix}\mathbf{A} & \mathbf{B}\\ \mathbf{B}^{\mathsf{T}} & \mathbf{D}\end{bmatrix}\begin{bmatrix}\Delta\mathbf{p}\\ \Delta\mathbf{X}\end{bmatrix}=\begin{bmatrix}\mathbf{e}_p\\ \mathbf{e}_X\end{bmatrix}$$

Normal equation:

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

$$\rightarrow \begin{bmatrix}\mathbf{A} & \mathbf{B}\\ \mathbf{B}^{\mathsf{T}} & \mathbf{D}\end{bmatrix}\begin{bmatrix}\Delta\mathbf{p}\\ \Delta\mathbf{X}\end{bmatrix} = \begin{bmatrix}\mathbf{e}_{\mathbf{p}}\\ \mathbf{e}_{\mathbf{X}}\end{bmatrix}$$

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

$$\mathbf{D} = \begin{bmatrix}\mathbf{d}_1 & & \\ & \ddots & \\ & & \mathbf{d}_M\end{bmatrix} \rightarrow \mathbf{D}^{-1} = \begin{bmatrix}\mathbf{d}_1^{-1} & & \\ & \ddots & \\ & & \mathbf{d}_M^{-1}\end{bmatrix}$$

Inversion of block diagonal matrix can be efficiently computed.

$$\mathbf{J}^{\mathsf{T}}\mathbf{J} = \begin{bmatrix}\mathbf{J}_{\mathbf{p}}^{\mathsf{T}}\mathbf{J}_{\mathbf{p}} & \mathbf{J}_{\mathbf{p}}^{\mathsf{T}}\mathbf{J}_{\mathbf{X}}\\ \mathbf{J}_{\mathbf{X}}^{\mathsf{T}}\mathbf{J}_{\mathbf{p}} & \mathbf{J}_{\mathbf{X}}^{\mathsf{T}}\mathbf{J}_{\mathbf{X}}\end{bmatrix} = \begin{bmatrix}\mathbf{A} & \mathbf{B}\\ \mathbf{B}^{\mathsf{T}} & \mathbf{D}\end{bmatrix}$$

Normal equation:

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

$$\rightarrow \begin{bmatrix}\mathbf{A} & \mathbf{B}\\ \mathbf{B}^T & \mathbf{D}\end{bmatrix}\begin{bmatrix}\Delta\mathbf{p}\\ \Delta\mathbf{X}\end{bmatrix} = \begin{bmatrix}\mathbf{e}_p\\ \mathbf{e}_X\end{bmatrix}$$

$$\rightarrow \begin{bmatrix}\mathbf{I} & -\mathbf{B}\mathbf{D}^{-1}\\ \mathbf{0} & \mathbf{I}\end{bmatrix}\begin{bmatrix}\mathbf{A} & \mathbf{B}\\ \mathbf{B}^T & \mathbf{D}\end{bmatrix}\begin{bmatrix}\Delta\mathbf{p}\\ \Delta\mathbf{X}\end{bmatrix} = \begin{bmatrix}\mathbf{I} & -\mathbf{B}\mathbf{D}^{-1}\\ \mathbf{0} & \mathbf{I}\end{bmatrix}\begin{bmatrix}\mathbf{e}_p\\ \mathbf{e}_X\end{bmatrix}$$

$$\mathbf{J}^T\mathbf{J} = \begin{bmatrix}\mathbf{J}_p^T\mathbf{J}_p & \mathbf{J}_p^T\mathbf{J}_X\\ \mathbf{J}_X^T\mathbf{J}_p & \mathbf{J}_X^T\mathbf{J}_X\end{bmatrix} = \begin{bmatrix}\mathbf{A} & \mathbf{B}\\ \mathbf{B}^T & \mathbf{D}\end{bmatrix}$$

$$\mathbf{J}^T\mathbf{J} = \begin{bmatrix}\mathbf{D} & \\ & \end{bmatrix}, \quad \mathbf{D} = \begin{bmatrix}\mathbf{d}_1 & & \\ & \ddots & \\ & & \mathbf{d}_M\end{bmatrix} \rightarrow \mathbf{D}^{-1} = \begin{bmatrix}\mathbf{d}_1^{-1} & & \\ & \ddots & \\ & & \mathbf{d}_M^{-1}\end{bmatrix}$$

Inversion of block diagonal matrix can be efficiently computed.

Normal equation:

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

$$\rightarrow \begin{bmatrix}\mathbf{A} & \mathbf{B}\\ \mathbf{B}^T & \mathbf{D}\end{bmatrix}\begin{bmatrix}\Delta\mathbf{p}\\ \Delta\mathbf{X}\end{bmatrix} = \begin{bmatrix}\mathbf{e}_p\\ \mathbf{e}_X\end{bmatrix}$$

$$\rightarrow \begin{bmatrix}\mathbf{I} & -\mathbf{B}\mathbf{D}^{-1}\\ \mathbf{0} & \mathbf{I}\end{bmatrix}\begin{bmatrix}\mathbf{A} & \mathbf{B}\\ \mathbf{B}^T & \mathbf{D}\end{bmatrix}\begin{bmatrix}\Delta\mathbf{p}\\ \Delta\mathbf{X}\end{bmatrix} = \begin{bmatrix}\mathbf{I} & -\mathbf{B}\mathbf{D}^{-1}\\ \mathbf{0} & \mathbf{I}\end{bmatrix}\begin{bmatrix}\mathbf{e}_p\\ \mathbf{e}_X\end{bmatrix}$$

$$\rightarrow \begin{bmatrix}\mathbf{A} - \mathbf{B}\mathbf{D}^{-1}\mathbf{B}^T & \mathbf{0}\\ \mathbf{B}^T & \mathbf{D}\end{bmatrix}\begin{bmatrix}\Delta\mathbf{p}\\ \Delta\mathbf{X}\end{bmatrix} = \begin{bmatrix}\mathbf{e}_p - \mathbf{B}\mathbf{D}^{-1}\mathbf{e}_X\\ \mathbf{e}_X\end{bmatrix}$$

$$\mathbf{J}^T\mathbf{J} = \begin{bmatrix}\mathbf{J}_p^T\mathbf{J}_p & \mathbf{J}_p^T\mathbf{J}_X\\ \mathbf{J}_X^T\mathbf{J}_p & \mathbf{J}_X^T\mathbf{J}_X\end{bmatrix} = \begin{bmatrix}\mathbf{A} & \mathbf{B}\\ \mathbf{B}^T & \mathbf{D}\end{bmatrix}$$

$$\mathbf{J}^T\mathbf{J} = \begin{bmatrix}\mathbf{D} & \\ & \end{bmatrix}, \quad \mathbf{D} = \begin{bmatrix}\mathbf{d}_1 & & \\ & \ddots & \\ & & \mathbf{d}_M\end{bmatrix} \rightarrow \mathbf{D}^{-1} = \begin{bmatrix}\mathbf{d}_1^{-1} & & \\ & \ddots & \\ & & \mathbf{d}_M^{-1}\end{bmatrix}$$

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:

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

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

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

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

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

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

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

$$\mathbf{J} = \begin{bmatrix} \mathbf{J_p} & \mathbf{J_X} \end{bmatrix} \qquad \left(\mathbf{b} - f(\mathbf{p})\right) \qquad \mathbf{D}^{-1} = \begin{bmatrix} \mathbf{d}_1^{-1} & & \\ & \ddots & \\ & & \mathbf{d}_M^{-1} \end{bmatrix}$$

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: $\mathbf{d} = \mathbf{0}_{3\times 3}$
6: **for** $j = 1 : I$ **do** ← # of images
7: **if** the i^{th} point is visible from the j^{th} image **then** ← if visible

$$\mathbf{J} = \begin{bmatrix} \mathbf{J_p} & \mathbf{J_X} \end{bmatrix} \qquad \left(\mathbf{b} - f(\mathbf{p})\right) \qquad \mathbf{D}^{-1} = \begin{bmatrix} \mathbf{d}_1^{-1} & & \\ & \ddots & \\ & & \mathbf{d}_M^{-1} \end{bmatrix}$$

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
    $\mathbf{d} = \mathbf{0}_{3\times 3}$
    for $j = 1 : I$ **do** ← # of images
      if the i^{th} point is visible from the j^{th} image **then** ← if visible
        $\mathbf{J}_1 = \mathbf{0}_{2\times 7I}$ and $\mathbf{J}_2 = \mathbf{0}_{2\times 3M}$
        $\mathbf{J}_1(:, 7(j-1)+1 : 7j) = \frac{\partial f(\mathbf{p}_j, \mathbf{X}_i)}{\partial \mathbf{p}_j}$ ← $\mathbf{J_p}$

$$\mathbf{J} = \begin{bmatrix} \mathbf{J_p} & \mathbf{J_X} \end{bmatrix} \qquad \left(\mathbf{b} - f(\mathbf{p})\right) \qquad \mathbf{D}^{-1} = \begin{bmatrix} \mathbf{d}_1^{-1} & & \\ & \ddots & \\ & & \mathbf{d}_M^{-1} \end{bmatrix}$$

Algorithm 4 Bundle Adjustment

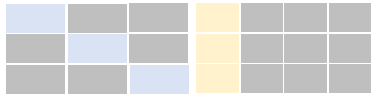
1: $\widehat{p} = \begin{bmatrix} \mathbf{p}_1^\mathsf{T} & \cdots & \mathbf{p}_I^\mathsf{T} \end{bmatrix}^\mathsf{T}$ and $\widehat{\mathbf{X}} = \begin{bmatrix} \mathbf{X}_1^\mathsf{T} & \cdots & \mathbf{X}_M^\mathsf{T} \end{bmatrix}$
2: **for** iter $= 1 :$ nIters **do**
3: Empty $\mathbf{J_p}$, $\mathbf{J_X}$, $\mathbf{b}$, $\mathbf{f}$, $\mathbf{D}_{\text{inv}}$.
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: $\mathbf{J}_1(:, 7(j-1)+1 : 7j) = \frac{\partial f(\mathbf{p}_j, \mathbf{X}_i)}{\partial \mathbf{p}_j}$ ← $\mathbf{J_p}$
10: $\mathbf{J}_2(:, 3(i-1)+1 : 3i) = \frac{\partial f(\mathbf{p}_j, \mathbf{X}_i)}{\partial \mathbf{X}_i}$ ← $\mathbf{J_X}$

$$\mathbf{J} = \begin{bmatrix} \mathbf{J_p} & \mathbf{J_X} \end{bmatrix} \qquad \left(\mathbf{b} - f(\mathbf{p})\right) \qquad \mathbf{D}^{-1} = \begin{bmatrix} \mathbf{d}_1^{-1} & & \\ & \ddots & \\ & & \mathbf{d}_M^{-1} \end{bmatrix}$$

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 $\mathbf{J}_2 = \mathbf{0}_{2\times3M}$
        $\mathbf{J}_1(:, 7(j-1)+1 : 7j) = \frac{\partial f(\mathbf{p}_j, \mathbf{X}_i)}{\partial \mathbf{p}_j}$ ← $\mathbf{J_p}$
        $\mathbf{J}_2(:, 3(i-1)+1 : 3i) = \frac{\partial f(\mathbf{p}_j, \mathbf{X}_i)}{\partial \mathbf{X}_i}$ ← $\mathbf{J_X}$
        $\mathbf{J_p} = \begin{bmatrix} \mathbf{J_p}^\mathsf{T} & \mathbf{J}_1^\mathsf{T} \end{bmatrix}^\mathsf{T}$ and $\mathbf{J_X} = \begin{bmatrix} \mathbf{J_X}^\mathsf{T} & \mathbf{J}_2^\mathsf{T} \end{bmatrix}^\mathsf{T}$

$$\mathbf{J} = \begin{bmatrix} \mathbf{J_p} & \mathbf{J_X} \end{bmatrix} \qquad \left(\mathbf{b} - f(\mathbf{p})\right) \qquad \mathbf{D}^{-1} = \begin{bmatrix} \mathbf{d}_1^{-1} & & \\ & \ddots & \\ & & \mathbf{d}_M^{-1} \end{bmatrix}$$



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

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

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

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

        $\mathbf{J}_1 = \mathbf{0}_{2\times 7I}$ and $\mathbf{J}_2 = \mathbf{0}_{2\times 3M}$

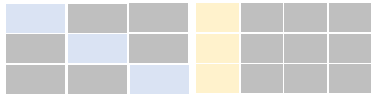
        $\mathbf{J}_1(:, 7(j-1)+1 : 7j) = \frac{\partial f(\mathbf{p}_j, \mathbf{X}_i)}{\partial \mathbf{p}_j}$ ← $\mathbf{J_p}$

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

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

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

$$\mathbf{J} = \begin{bmatrix} \mathbf{J_p} & \mathbf{J_X} \end{bmatrix} \qquad \left(\mathbf{b} - f(\mathbf{p})\right) \qquad \mathbf{D}^{-1} = \begin{bmatrix} \mathbf{d}_1^{-1} & & \\ & \ddots & \\ & & \mathbf{d}_M^{-1} \end{bmatrix}$$



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

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

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

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

        $\mathbf{J}_1 = \mathbf{0}_{2\times 7I}$ and $\mathbf{J}_2 = \mathbf{0}_{2\times 3M}$

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

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

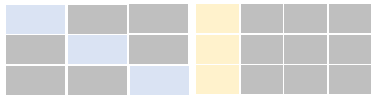
        $\mathbf{J_p} = \begin{bmatrix} \mathbf{J}_\mathbf{p}^\mathsf{T} & \mathbf{J}_1^\mathsf{T} \end{bmatrix}^\mathsf{T}$ and $\mathbf{J_X} = \begin{bmatrix} \mathbf{J}_\mathbf{X}^\mathsf{T} & \mathbf{J}_2^\mathsf{T} \end{bmatrix}^\mathsf{T}$ ←

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

        $\mathbf{b} = \begin{bmatrix} \mathbf{b}^\mathsf{T} & \mathbf{u}_{ij}^\mathsf{T} \end{bmatrix}$

        $\mathbf{f} = \begin{bmatrix} \mathbf{f}^\mathsf{T} & \mathbf{x}_{ij}^\mathsf{T} \end{bmatrix}$ where $\mu \begin{bmatrix} \mathbf{x}_{ij} \\ 1 \end{bmatrix} = \mathbf{R}_j \begin{bmatrix} \mathbf{I} & -\mathbf{C}_j \end{bmatrix}$ ← $\big(\mathbf{b} - f(\mathbf{p})\big)$

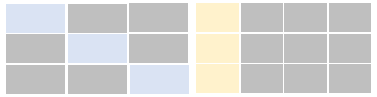
$$\mathbf{J} = \begin{bmatrix} \mathbf{J_p} & \mathbf{J_X} \end{bmatrix} \qquad \big(\mathbf{b} - f(\mathbf{p})\big) \qquad \mathbf{D}^{-1} = \begin{bmatrix} \mathbf{d}_1^{-1} & & \\ & \ddots & \\ & & \mathbf{d}_M^{-1} \end{bmatrix}$$



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

$$\mathbf{J} = \begin{bmatrix} \mathbf{J_p} & \mathbf{J_X} \end{bmatrix} \qquad \mathbf{b} - f(\mathbf{p}) \qquad \mathbf{D}^{-1} = \begin{bmatrix} \mathbf{d}_1^{-1} & & \\ & \ddots & \\ & & \mathbf{d}_M^{-1} \end{bmatrix}$$

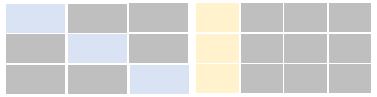


$$\mathbf{D}^{-1} = \begin{bmatrix} \mathbf{d}_1^{-1} & & \\ & \ddots & \\ & & \mathbf{d}_M^{-1} \end{bmatrix}$$

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 $\mathbf{J}_2 = \mathbf{0}_{2\times3M}$
        $\mathbf{J}_1(:, 7(j-1)+1 : 7j) = \frac{\partial f(\mathbf{p}_j, \mathbf{X}_i)}{\partial \mathbf{p}_j}$ ← $\mathbf{J_p}$
        $\mathbf{J}_2(:, 3(i-1)+1 : 3i) = \frac{\partial f(\mathbf{p}_j, \mathbf{X}_i)}{\partial \mathbf{X}_i}$ ← $\mathbf{J_X}$
        $\mathbf{J_p} = \begin{bmatrix} \mathbf{J}_\mathbf{p}^\mathsf{T} & \mathbf{J}_1^\mathsf{T} \end{bmatrix}^\mathsf{T}$ and $\mathbf{J_X} = \begin{bmatrix} \mathbf{J}_\mathbf{X}^\mathsf{T} & \mathbf{J}_2^\mathsf{T} \end{bmatrix}^\mathsf{T}$ ←
        $\mathbf{d} = \mathbf{d} + \frac{\partial f(\mathbf{p}_j, \mathbf{X}_i)}{\partial \mathbf{X}_i}^\mathsf{T} \frac{\partial f(\mathbf{p}_j, \mathbf{X}_i)}{\partial \mathbf{X}_i}$ ← $\mathbf{d}$
        $\mathbf{b} = \begin{bmatrix} \mathbf{b}^\mathsf{T} & \mathbf{u}_{ij}^\mathsf{T} \end{bmatrix}$
        $\mathbf{f} = \begin{bmatrix} \mathbf{f}^\mathsf{T} & \mathbf{x}_{ij}^\mathsf{T} \end{bmatrix}$ where $\mu \begin{bmatrix} \mathbf{x}_{ij} \\ 1 \end{bmatrix} = \mathbf{R}_j \begin{bmatrix} \mathbf{I} & -\mathbf{C}_j \end{bmatrix}$ ← $(\mathbf{b} - f(\mathbf{p}))$
      end if
    end for
    $\mathbf{d} = \mathbf{d} + \lambda \mathbf{I}$ ← $\mathbf{d} = \mathbf{d} + \lambda \mathbf{I}$
    $\mathbf{D}_{\text{inv}} = \texttt{blkdiag}(\mathbf{D}_{\text{inv}}, \mathbf{d}^{-1})$
  end for
  $\mathbf{e_p} = \mathbf{J}_\mathbf{p}^\mathsf{T}(\mathbf{b} - \mathbf{f})$ ← $\begin{bmatrix} \mathbf{J}_\mathbf{p}^\mathsf{T} \\ \mathbf{J}_\mathbf{X}^\mathsf{T} \end{bmatrix} (\mathbf{b} - f(\mathbf{X})) = \begin{bmatrix} \mathbf{e}_\mathbf{p} \\ \mathbf{e}_\mathbf{X} \end{bmatrix}$
  $\mathbf{e_X} = \mathbf{J}_\mathbf{X}^\mathsf{T}(\mathbf{b} - \mathbf{f})$

$\mathbf{J} = \begin{bmatrix} \mathbf{J_p} & \mathbf{J_X} \end{bmatrix}$ $(\mathbf{b} - f(\mathbf{p}))$ $\mathbf{D}^{-1} = \begin{bmatrix} \mathbf{d}_1^{-1} & & \\ & \ddots & \\ & & \mathbf{d}_M^{-1} \end{bmatrix}$

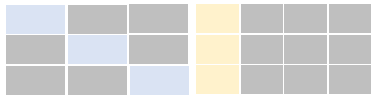


$\mathbf{D}^{-1} = \begin{bmatrix} \mathbf{d}_1^{-1} & & \\ & \ddots & \\ & & \mathbf{d}_M^{-1} \end{bmatrix}$

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

$\mathbf{J} = \begin{bmatrix} \mathbf{J}_\mathbf{p} & \mathbf{J}_\mathbf{X} \end{bmatrix}$ $\quad (\mathbf{b} - f(\mathbf{p}))$ $\quad \mathbf{D}^{-1} = \begin{bmatrix} \mathbf{d}_1^{-1} & & \\ & \ddots & \\ & & \mathbf{d}_M^{-1} \end{bmatrix}$

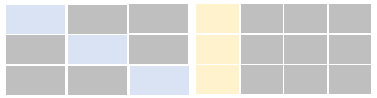


$\mathbf{D}^{-1} = \begin{bmatrix} \mathbf{d}_1^{-1} & & \\ & \ddots & \\ & & \mathbf{d}_M^{-1} \end{bmatrix}$

Algorithm 4 Bundle Adjustment

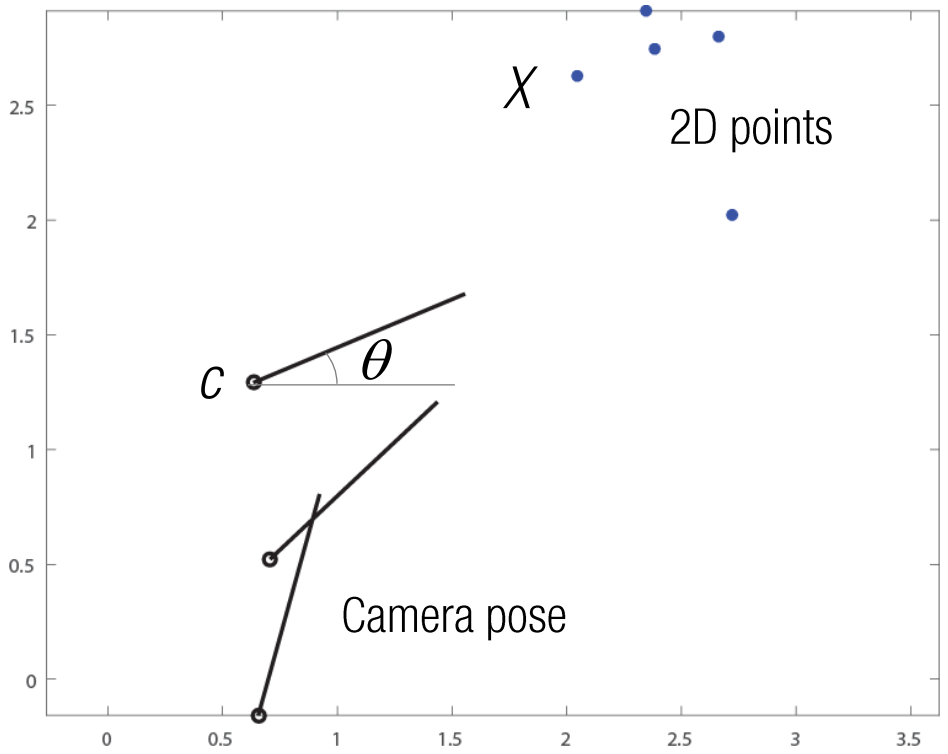
$\widehat{p} = \begin{bmatrix} \mathbf{p}_1^\mathsf{T} & \cdots & \mathbf{p}_I^\mathsf{T} \end{bmatrix}^\mathsf{T}$ and $\widehat{\mathbf{X}} = \begin{bmatrix} \mathbf{X}_1^\mathsf{T} & \cdots & \mathbf{X}_M^\mathsf{T} \end{bmatrix}$
for iter $= 1 :$ nIters **do**
  Empty $\mathbf{J_p}$, $\mathbf{J_X}$,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

$\mathbf{J} = \begin{bmatrix} \mathbf{J}_\mathbf{p} & \mathbf{J}_\mathbf{X} \end{bmatrix}$ $\quad (\mathbf{b} - f(\mathbf{p}))$ $\quad \mathbf{D}^{-1} = \begin{bmatrix} \mathbf{d}_1^{-1} & & \\ & \ddots & \\ & & \mathbf{d}_M^{-1} \end{bmatrix}$



$\mathbf{D}^{-1} = \begin{bmatrix} \mathbf{d}_1^{-1} & & \\ & \ddots & \\ & & \mathbf{d}_M^{-1} \end{bmatrix}$

1D Camera Bundle Adjustment

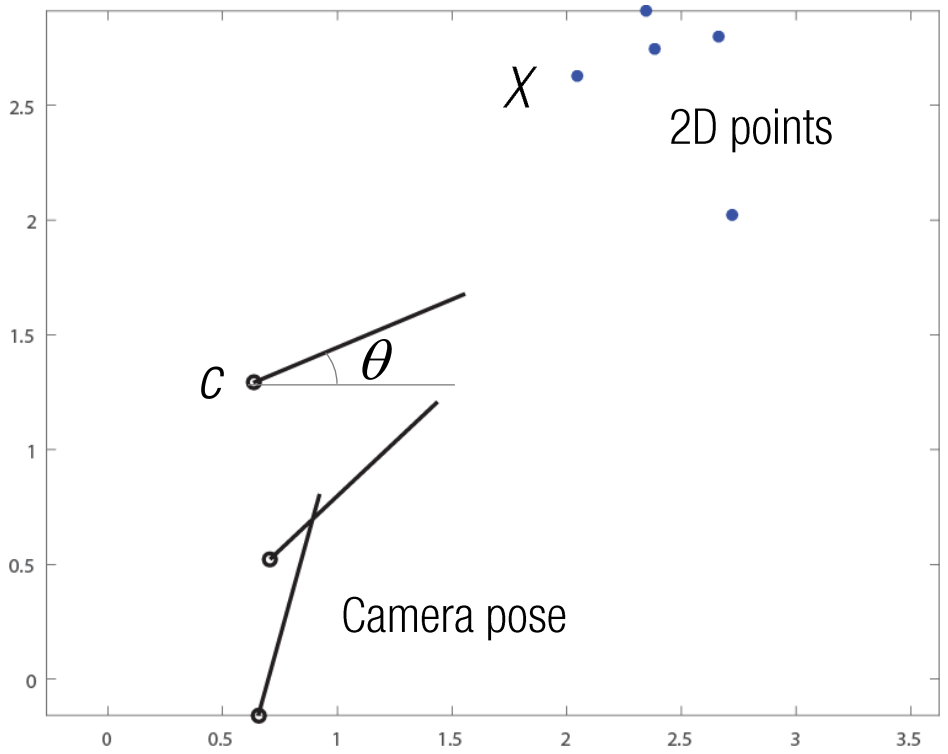


BundleAdjustment1D.m

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



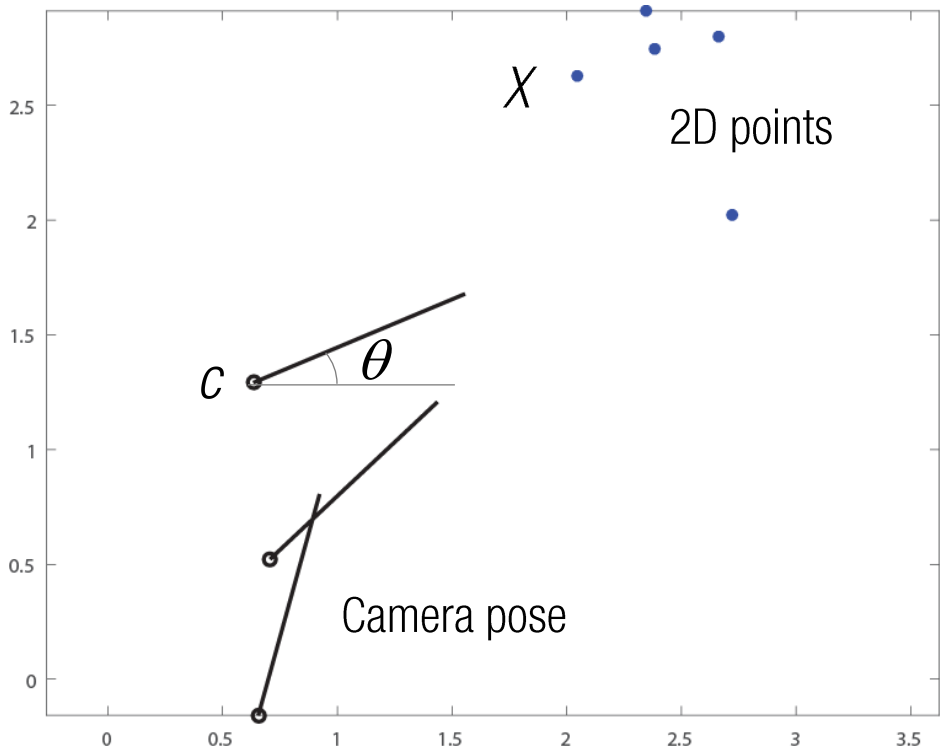
BundleAdjustment1D.m

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

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)];
```

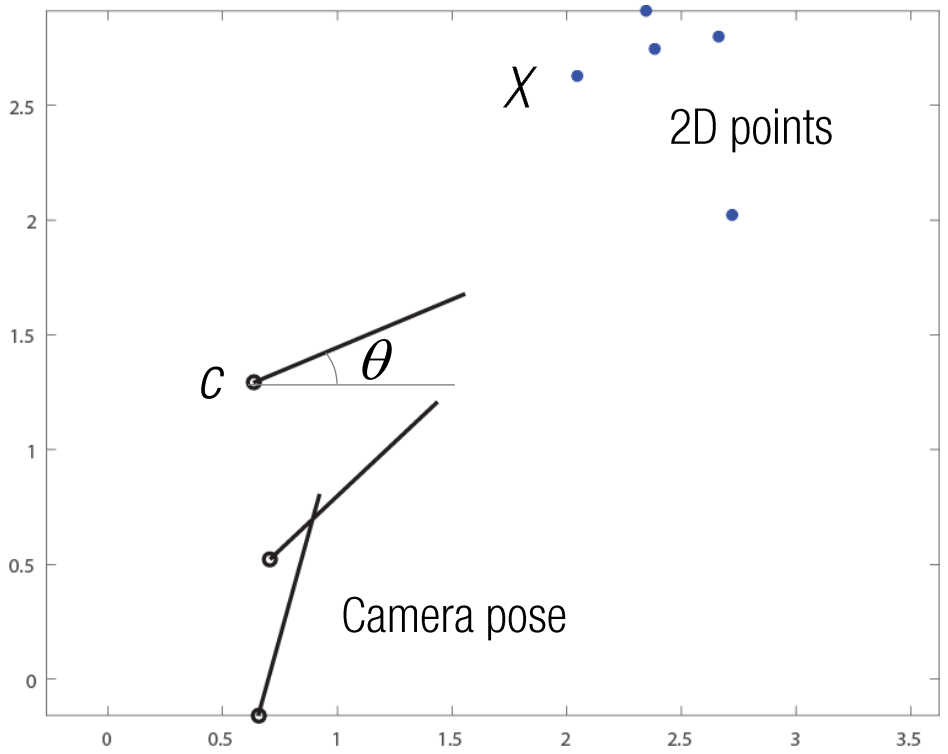
Ground truth projection

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

Add noise

```
   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

```
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

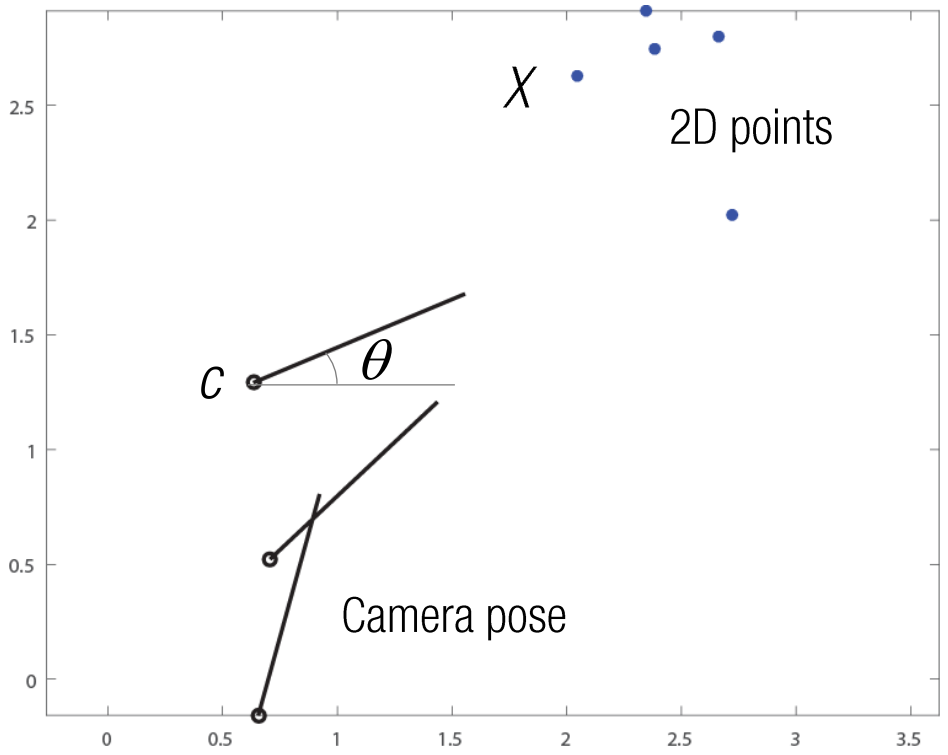
Ground truth projection

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

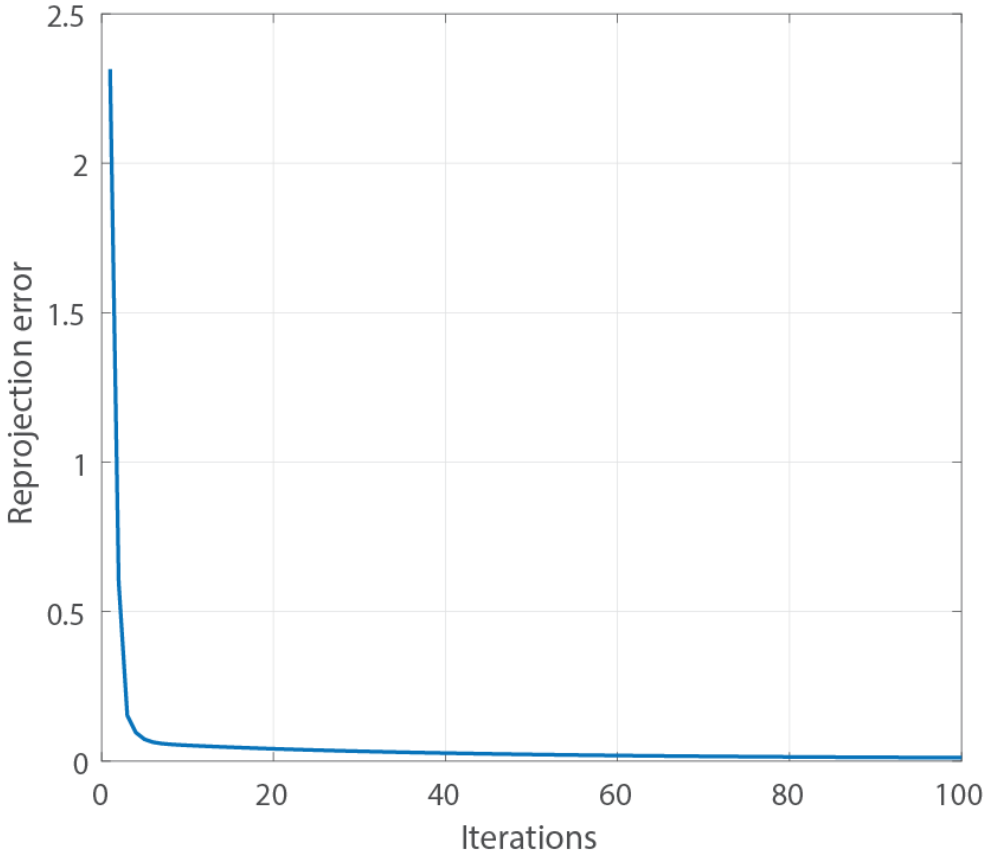
Algorithm 4 Bundle Adjustment

$\widehat{p} = \begin{bmatrix} \mathbf{p}_1^\mathsf{T} & \cdots & \mathbf{p}_I^\mathsf{T} \end{bmatrix}^\mathsf{T}$ 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**
    $\mathbf{d} = \mathbf{0}_{3\times 3}$
    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 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

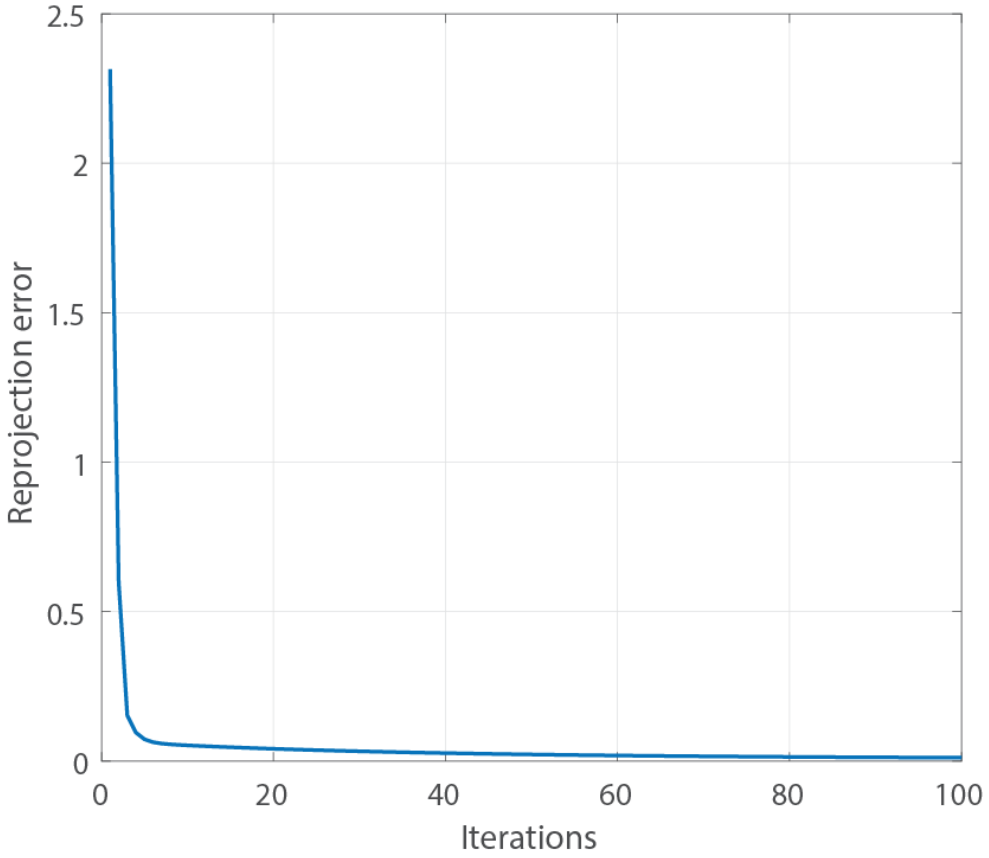
```
for iC = 1 : length(C)
   C(iC).c = xp(3*(iC-1)+1:3*(iC-1)+2);
   theta = xp(3*(iC-1)+3);
   C(iC).R = [sin(theta) -cos(theta); cos(theta) sin(theta)];
end

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)