PS1, Question 2, part A

Derivation of matrix equations for calibrating affine camera: following along Hartley + Zisserman Ch4 and Ch7, start out with the general projective camera model and then simplify a few terms later to make the affine camera approximation.

The equation for a general projective transformation can be written as:

$$\mathbf{x}_i = \mathbf{P}\mathbf{X}_i \qquad or \qquad \mathbf{x}_i = \begin{pmatrix} \mathbf{P}_1^T\mathbf{X}_i \\ \mathbf{P}_2^T\mathbf{X}_i \\ \mathbf{P}_3^T\mathbf{X}_i \end{pmatrix}$$

Where $\mathbf{x}_i$ denotes the image point, $\mathbf{X}_i$ denotes the scene point in world coordinates, and **P** is the 3x4 camera matrix. The terms $\mathbf{P}_k^T$ in the rightmost expression mean the 1x4 vector of elements of the k^{th} row of **P**, and the subscript i denotes that this is for the i^{th} point correspondence we are considering.

We actually find the camera matrix only up to scale. A nice way to account for this is to take the cross product of both sides as follows:

$$\mathbf{x}_i \times \mathbf{x}_i = \mathbf{x}_i \times \mathbf{P}\mathbf{X}_i$$

The left side is zero, so this can all be written as:

$$\mathbf{0} = \mathbf{x}_i \times \mathbf{P}\mathbf{X}_i \qquad or \qquad \mathbf{0} = \mathbf{x}_i \times \begin{pmatrix} \mathbf{P}_1^T\mathbf{X}_i \\ \mathbf{P}_2^T\mathbf{X}_i \\ \mathbf{P}_3^T\mathbf{X}_i \end{pmatrix}$$

Calculating the cross product:

$$\begin{vmatrix} \hat{\imath} & \hat{\jmath} & \hat{k} \\ \mathrm{x}_i & \mathrm{y}_i & \mathrm{w}_i \\ \mathbf{P}_1^T\mathbf{X}_i & \mathbf{P}_2^T\mathbf{X}_i & \mathbf{P}_3^T\mathbf{X}_i \end{vmatrix} = \begin{pmatrix} \mathrm{y}_i\mathbf{P}_3^T\mathbf{X}_i - \mathrm{w}_i\mathbf{P}_2^T\mathbf{X}_i \\ \mathrm{w}_i\mathbf{P}_1^T\mathbf{X}_i - \mathrm{x}_i\mathbf{P}_3^T\mathbf{X}_i \\ \mathrm{x}_i\mathbf{P}_2^T\mathbf{X}_i - \mathrm{y}_i\mathbf{P}_1^T\mathbf{X}_i \end{pmatrix}$$

$$= \begin{pmatrix} \mathbf{0}^T & -\mathrm{w}_i\boldsymbol{X}_i^T & \mathrm{y}_i\boldsymbol{X}_i^T \\ \mathrm{w}_i\boldsymbol{X}_i^T & \mathbf{0}^T & -\mathrm{x}_i\boldsymbol{X}_i^T \\ -\mathrm{y}_i\boldsymbol{X}_i^T & \mathrm{x}_i\boldsymbol{X}_i^T & \mathbf{0}^T \end{pmatrix}_{3\mathrm{x}12} \begin{pmatrix} \mathbf{P}_1 \\ \mathbf{P}_2 \\ \mathbf{P}_3 \end{pmatrix}_{12\mathrm{x}1} = \mathbf{0}_{3\mathrm{x}1}$$

However, the 3 equations obtained by carrying out the matrix multiplication are not linearly independent. The 3rd row can be written up to scale as $\mathrm{x}_i \cdot row1 + \mathrm{y}_i \cdot row2$. So, don't use the third row, and just use 2 equations per point correspondence.

So for the generic projective camera model, for a single point pair correspondence index i, we have:

$$\boxed{\begin{pmatrix} \mathbf{0}^T & -\mathrm{w}_i \boldsymbol{X}_i^T & \mathrm{y}_i \boldsymbol{X}_i^T \\ \mathrm{w}_i \boldsymbol{X}_i^T & \mathbf{0}^T & -\mathrm{x}_i \boldsymbol{X}_i^T \end{pmatrix}_{2\text{x}12} \begin{pmatrix} \mathbf{P}_1 \\ \mathbf{P}_2 \\ \mathbf{P}_3 \end{pmatrix}_{12\text{x}1} = \mathbf{0}_{2\text{x}1}}$$

N such equations can be stacked (each point correspondence $\mathbf{x}_i$ to $\mathbf{X}_i$ contributes 2 equations) to get a 2Nx12 matrix, and the elements of **P** can be solved in a least squares fashion by using SVD, and then the matrix **P** made by reshaping it into a 3x4 matrix.

The boxed equation above can be rewritten as:

$$\begin{pmatrix} \mathbf{0}^T & -\mathrm{w}_i \boldsymbol{X}_i^T \\ \mathrm{w}_i \boldsymbol{X}_i^T & \mathbf{0}^T \end{pmatrix} \begin{pmatrix} \mathbf{P}_1 \\ \mathbf{P}_2 \end{pmatrix} + \begin{pmatrix} \mathrm{y}_i \boldsymbol{X}_i^T \mathbf{P}_3 \\ -\mathrm{x}_i \boldsymbol{X}_i^T \mathbf{P}_3 \end{pmatrix} = \mathbf{0}_{2\text{x}1}$$

Now we can apply simplifications because it's an affine camera.

For an affine camera, $\mathrm{w}_i = 1$, and $\mathbf{P}_3^T = (0,0,0,1)$

$$\begin{pmatrix} \mathbf{0}^T & -\boldsymbol{X}_i^T \\ \boldsymbol{X}_i^T & \mathbf{0}^T \end{pmatrix} \begin{pmatrix} \mathbf{P}_1 \\ \mathbf{P}_2 \end{pmatrix} + \begin{pmatrix} \mathrm{y}_i \\ -\mathrm{x}_i \end{pmatrix} = \mathbf{0}_{2\text{x}1}$$

Then reordering slightly:

$$\begin{pmatrix} \boldsymbol{X}_i^T & \mathbf{0}^T \\ \mathbf{0}^T & \boldsymbol{X}_i^T \end{pmatrix} \begin{pmatrix} \mathbf{P}_1 \\ \mathbf{P}_2 \end{pmatrix} = \begin{pmatrix} \mathrm{x}_i \\ \mathrm{y}_i \end{pmatrix}$$

N such correspondences can be stacked to get a 2Nx8 matrix of constraints:

$$\begin{pmatrix} \boldsymbol{X}_1^T & \mathbf{0}^T \\ \boldsymbol{X}_2^T & \mathbf{0}^T \\ \cdots & \cdots \\ \boldsymbol{X}_N^T & \mathbf{0}^T \\ \mathbf{0}^T & \boldsymbol{X}_1^T \\ \mathbf{0}^T & \boldsymbol{X}_2^T \\ \cdots & \cdots \\ \mathbf{0}^T & \boldsymbol{X}_N^T \end{pmatrix} \begin{pmatrix} \mathbf{P}_1 \\ \mathbf{P}_2 \end{pmatrix} = \begin{pmatrix} \mathrm{x}_1 \\ \mathrm{x}_2 \\ \cdots \\ \mathrm{x}_N \\ \mathrm{y}_1 \\ \mathrm{y}_2 \\ \cdots \\ \mathrm{y}_N \end{pmatrix}$$

This can be solved by matrix inversion using the pseudo-inverse (+ superscript symbol) of the constraint matrix:

$$\boxed{\begin{pmatrix} \mathbf{P}_1 \\ \mathbf{P}_2 \end{pmatrix} = \begin{pmatrix} \boldsymbol{X}_1^T & \mathbf{0}^T \\ \boldsymbol{X}_2^T & \mathbf{0}^T \\ \cdots & \cdots \\ \boldsymbol{X}_N^T & \mathbf{0}^T \\ \mathbf{0}^T & \boldsymbol{X}_1^T \\ \mathbf{0}^T & \boldsymbol{X}_2^T \\ \cdots & \cdots \\ \mathbf{0}^T & \boldsymbol{X}_N^T \end{pmatrix}^{+} \begin{pmatrix} \mathrm{x}_1 \\ \mathrm{x}_2 \\ \cdots \\ \mathrm{x}_N \\ \mathrm{y}_1 \\ \mathrm{y}_2 \\ \cdots \\ \mathrm{y}_N \end{pmatrix}}$$