3D Computer Vision Project Formulas

March 18, 2024

1 ESTIMATION OF ESSENTIAL MATRIX

Essential Matrix is comprised of the translation and Rotation as shown below

$$E = [t]_{\times} R \tag{1.1}$$

Essential matrix can be estimated from the 2D-2D correpondence between two frames, given the camera calibration matrix **K** as

$$X_2^T E X_1 = 0 \tag{1.2}$$

Algorithm 1 Calculate Essential Matrix

$A \leftarrow \text{zeros}(len(points1), 9)$
for $i = 1$ **to** $len(points1)$ **do**
    $(x, y, \_) \leftarrow points1[i]$
    $(x', y', \_) \leftarrow points2[i]$
    $A[i] \leftarrow [x' \times x, x' \times y, x', y' \times x, y' \times y, y', x, y, 1]$
end for
$U, S, V^T \leftarrow \text{svd}(A)$
$E \leftarrow V^T[-1].\text{reshape}(3, 3)$
$U, S, V^T \leftarrow \text{svd}(E)$
$E \leftarrow U \times \text{diag}[1, 1, 0] \times V^T$
$E \leftarrow E/\text{norm}(E)$
return E

1.1 RANSAC Algorithm for Essential matrix estimation

This algorithm is also used for estimating the feature matches between two images.

Algorithm 2 RANSAC for Essential Matrix Estimation

best_E $\leftarrow$ None
best_inliers $\leftarrow$ []
for iter $= 1, 2, \ldots, k_{\max}$ **do**
    $\mathrm{p}_{1_{\text{subset}}}, \mathrm{p}_{2_{\text{subset}}} \leftarrow \text{RandomSubset}(\mathrm{p}_1, \mathrm{p}_2, 8)$
    E $\leftarrow$ CalculateEssentialMatrix($\mathrm{p}_{1_{\text{subset}}}, \mathrm{p}_{2_{\text{subset}}}$)
    inliers $\leftarrow$ []
    for $i = 1$ **to** len(p_1) **do**
        $\epsilon \leftarrow \left|\mathrm{p}_{2_i}^T \cdot \mathrm{E}_{\text{estimate}} \cdot \mathrm{p}_{1_i}\right|$
        if $\epsilon < \tau$ **then**
            inliers.append(i)
        end if
    end for
    if len(inliers) $\geq T$ **then**
        all_inliers $\leftarrow$ inliers
        updated_E $\leftarrow$ CalculateEssentialMatrix([p_{1_i} in all_inliers], [p_{2_i} in all_inliers])
        inliers $\leftarrow$ []
        for $i = 1$ **to** len(p_1) **do**
            $\epsilon \leftarrow \left|\mathrm{p}_{2_i}^T \cdot \text{updated\_E} \cdot \mathrm{p}_{1_i}\right|$
            if $\epsilon < \tau$ **then**
                inliers.append(i)
            end if
        end for
        if len(inliers) $>$ len(best_inliers) **then**
            best_E $\leftarrow$ updated_E
            best_inliers $\leftarrow$ inliers
        end if
    end if
end for
return best_E, best_inliers

2 DECOMPOSITION OF ESSENTIAL MATRIX

In this method we use the concept that Essential Matrix incorporates the translation and rotation matrix as shown in 1.1.

If we consider Initial camera position to be at the origin of world coordinate system then the image of origin in the second camera will be *Epipole*. Since the translation vector is along the baseline, the epipole correspond to translation. we know,

$$\begin{bmatrix} x \\ 1 \end{bmatrix} = P \begin{bmatrix} X \\ 1 \end{bmatrix}$$

Considering normalised points, then t will be epipole in second image because,

$$or, \begin{bmatrix} x \\ 1 \end{bmatrix} = [R\ |t] \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} = t$$

Hence using the property of essential matrix and epipole,

$$t^T E = 0$$

Hence translation being epipole in second image is the left nullspace of essential matrix. Thus from SVD if $E = UDV^T$ where U in term of column vetors be $U = [u_1, u_2, u_3]$ then $t = u_3\ or\ -u_3$ i.e., $U = [u_1, u_2, t]$

$$U \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} V^T = [t]_\times R$$

The $[t]_\times$ operation is converted in terms of U. This cross product result transformation of arbitrary vector into the space perpendicular to t itself. For any vector a new orientation is defined by space $u_1, u_2\ and\ t$ since U is orthogonal matrix. Hence any vector is transformed into this space using U^T and then we remove the t elements so that vector will remain perpendicular to the t. Then we rotate by 90 degree in the space of u_1 and u_2, which is finally transformed back to original space using U matrix. Therefore,

$$U \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} V^T = \left(U \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} U^T \right) R$$

Using SVD for rotation matrix R we have,

$$U \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} V^T = \left(U \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} U^T \right) (UWV^T)$$

$$or, U \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} V^T = U \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} WV^T$$

$$or, \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} W$$

Hence possible solutions for W are

$$or, W = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} or \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}^T$$

Therefore, we have

$$R = U \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} V^T \quad or \quad R = U \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}^T V^T$$

So there are four possible results of the decomposition as

$$R = UWV^T \quad t = u_3$$

$$R = UW^TV^T \quad t = u_3$$

$$R = UWV^T \quad t = -u_3$$

$$R = UW^TV^T \quad t = -u_3$$

If det(R) = -1 then t = -t and R = -R. Here we have four configuration of cameras. This ambiguity is removed by carrying out the triangulation using computed pose and the configuration which gives point in front of camera is the optimal choice.

$$r_3(X - t) > 0$$

This is called **cheirality condition**. Like this we are able to compute pose of camera from 2D correspondence using the concept of epipolar geometry and essential matrix.

3 TRIANGULATION

Given 2D-2D corresponding point between two images of same scene and relative position of second camera with respect to the first one, 3D coordinate of that point can be calculated. This is known as *Triangulation*.
Suppose points x_1 and x_2 represent same 3D point X in two images captured by two different camera having relative rotation and translation represented by matrix R and t
Let

$$P_1 = K_1[I_{3*3}|0_3]$$

then

$$P_2 = K_2[R_{3*3}|t_{3*1}]$$

From projective transformation we have,

$$\lambda \begin{bmatrix} x_1 \\ 1 \end{bmatrix} = P_1 \begin{bmatrix} X \\ 1 \end{bmatrix} \tag{3.1}$$

and,

$$\lambda \begin{bmatrix} x_2 \\ 1 \end{bmatrix} = P_2 \begin{bmatrix} X \\ 1 \end{bmatrix} \tag{3.2}$$

Taking cross product on both sides we have,

$$\begin{bmatrix} x_1 \\ 1 \end{bmatrix} \times P_1 \begin{bmatrix} X \\ 1 \end{bmatrix} = 0 \quad \text{and} \quad \begin{bmatrix} x_2 \\ 1 \end{bmatrix} \times P_2 \begin{bmatrix} X \\ 1 \end{bmatrix} = 0$$

Combining above equations we have,

$$\begin{bmatrix} \begin{bmatrix} x_1 \\ 1 \end{bmatrix} \times P_1 \\ \begin{bmatrix} x_2 \\ 1 \end{bmatrix} \times P_2 \end{bmatrix} \begin{bmatrix} X \\ 1 \end{bmatrix} = 0 \tag{3.3}$$

Equation 3.3 is in the form of

$$Ax = 0$$

This equation is least square problem. This can be simply solved by the use of technique known as *Singular Value Decompostion*

4 POSE FROM 2D-3D CORRESPONDENCE(LINEAR PNP)

Given the corresponding 2D-3D correspondence the pose of camera can be estimated. This method is simple form of perspective n point algorithm also known as **Linear PnP**. For 3D points in world coordinate system obtained after triangulation if we have corresponding 2D keypoints in image present in the camera coordinate system, then using this relation between 2D and 3D points and the concept of camera projection we are able to estimate the pose of the camera in the world coordinate system.

Let $x_i \rightarrow X_i$ be the 2d-3D corresponding points and $P_1 = K_1[I_{3*3}|0_3]$ be the projection matrix of the camera. Then from projective transformation we have,

$$\lambda \begin{bmatrix} x \\ 1 \end{bmatrix} = P \begin{bmatrix} X \\ 1 \end{bmatrix}$$

Taking cross on both sides we have,

$$\begin{bmatrix} x \\ 1 \end{bmatrix} \times P \begin{bmatrix} X \\ 1 \end{bmatrix} = 0$$

$$or, \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \times \begin{bmatrix} P_1 \\ P_2 \\ P_3 \end{bmatrix} \tilde{X} = 0$$

where $\tilde{X}$ is the 3D homogeneous point in four dimension. and P_1, P_2, P_3 are the 3 rows of projection matrix respectively.

$$or, \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \times \begin{bmatrix} P_1\tilde{X} \\ P_2\tilde{X} \\ P_3\tilde{X} \end{bmatrix} = 0$$

$$or, \begin{bmatrix} 0 & -1 & v \\ 1 & 0 & -u \\ -v & u & 0 \end{bmatrix} \begin{bmatrix} P_1\tilde{X} \\ P_2\tilde{X} \\ P_3\tilde{X} \end{bmatrix} = 0$$

$$or, \begin{bmatrix} 0 & -1 & v \\ 1 & 0 & -u \\ -v & u & 0 \end{bmatrix}_{3\times3} \begin{bmatrix} \tilde{X}^T & 0_{1\times4} & 0_{1\times4} \\ 0_{1\times4} & \tilde{X}^T & 0_{1\times4} \\ 0_{1\times4} & 0_{1\times4} & \tilde{X}^T \end{bmatrix}_{3\times12} \begin{bmatrix} P_1^T \\ P_2^T \\ P_3^T \end{bmatrix}_{12\times1} = 0$$

$$or, \begin{bmatrix} 0 & -\tilde{X}^T & v\tilde{X}^T \\ \tilde{X}^T & 0 & -u\tilde{X}^T \\ -v\tilde{X}^T & u\tilde{X}^T & 0 \end{bmatrix}_{3\times12} \begin{bmatrix} P_1^T \\ P_2^T \\ P_3^T \end{bmatrix}_{12\times1} = 0 \tag{4.1}$$

This equation takes form of **least square problem** as $Ax = 0$ which is solved using **Singular Value Decomposition** . Each correspondence gives 2 constraints therefore to compute 12 unknown we require at least 6 point correspondence. Hence we have computed the elements of projection matrix. Now we need to extract the Rotation and Translation from the Projection matrix given the camera matrix K.
We know,

$$P = K[R\ |t]$$

$$or, K^{-1}P = [R\ |t]$$

Hence

$$R = K^{-1}P_{1:3} \quad \text{and} \quad t = K^{-1}P_4$$

Since R must be orthogonal matrix with determinant 1 it must be cleaned up and translation vector must be scaled.

$$R = UDV^T \quad \text{Using SVD}$$

$$R_c = UV^T, \quad t_c = t/D_{1,1} \quad if det(UV^T) = 1$$

$$R_c = -UV^T, \quad t_c = -t/D_{1,1} \quad if det(UV^T) = -1$$

Like this we can estimate the pose of camera given 2D-3D point correspondence. RANSAC algorithm is used to reject the outliers and compute accurate pose of the camera.

Given the camera matrix we can first normalise the 2D points and directly compute the RT matrix.

5 THIRD PERSON (WORLD) PERSPECTIVE

The camera center in world coordinates is denoted as $\{C\}$ and the world coordinate system as $\{W\}$. The transformations are given by:

$$\begin{aligned} C &= -R^{-1}t \\ R^{-1} &= R^{\mathsf{T}} \quad \text{(since } R \text{ is an orthogonal matrix)} \\ P &= K[R|t] \\ &= K[R| - RC] \\ &= KR\left[I_{3\times 3}| - C\right] \end{aligned}$$

where P is the camera projection matrix, R is the rotation matrix, t is the translation vector, K is the camera intrinsic matrix, and C is the camera center seen from the world coordinate system.

6 BUNDLE ADJUSTMENT

6.1 Jacobian Matrix

The Jacobian matrix J is defined as:

$$J = \left[\frac{\partial f(R(q),C,X)}{\partial R} * \frac{\partial R}{\partial q} \quad \frac{\partial f(R(q),C,X)}{\partial C} \quad \frac{\partial f(R(q),C,X)}{\partial X}\right]$$

where the function $f(R(q), C, X)$ is given by:

$$f(R(q), C, X) = \begin{bmatrix} u/w \\ v/w \end{bmatrix}$$

where

$$\begin{aligned} u &= [f_x r_{11} + o_x r_{31}, f_x r_{12} + o_x r_{32}, f_x r_{13} + o_x r_{33}][X - C] \\ v &= [f_y r_{21} + o_y r_{21}, f_y r_{22} + o_y r_{22}, f_y r_{23} + o_y r_{23}][X - C] \\ w &= [r_{31}, r_{32}, r_{33}][X - C] \end{aligned}$$

Here, K $= \begin{bmatrix} f_x & 0 & o_x \\ 0 & f_y & o_y \\ 0 & 0 & 1 \end{bmatrix}$

6.2 Jacobian w.r.t. 3D point (X)

The partial derivative of $f(R(q), C, X)$ with respect to X is:

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

where

$$\frac{\partial u}{\partial X} = [f_x r_{11} + o_x r_{31}, f_x r_{12} + o_x r_{32}, f_x r_{13} + o_x r_{33}]$$
$$\frac{\partial v}{\partial X} = [f_y r_{21} + o_y r_{31}, f_y r_{22} + o_y r_{32}, f_y r_{23} + o_y r_{33}]$$
$$\frac{\partial w}{\partial X} = [r_{31}, r_{32}, r_{33}]$$

6.3 Jacobian w.r.t. camera center (C)

The partial derivative of $f(R(q), C, X)$ with respect to C is:

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

where

$$\frac{\partial u}{\partial C} = -[f_x r_{11} + o_x r_{31}, f_x r_{12} + o_x r_{32}, f_x r_{13} + o_x r_{33}]$$
$$\frac{\partial v}{\partial C} = -[f_y r_{21} + o_y r_{31}, f_y r_{22} + o_y r_{32}, f_y r_{23} + o_y r_{33}]$$
$$\frac{\partial w}{\partial C} = -[r_{31}, r_{32}, r_{33}]$$

6.4 Jacobian w.r.t. Rotation matrix (R)

The partial derivative of $f(R(q), C, X)$ with respect to R is:

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

where

$$\frac{\partial u}{\partial R} = [f_x(X_1 - C_1), 0_{1x3}, o_x(X_3 - C_3)]$$
$$\frac{\partial v}{\partial R} = [0_{1x3}, f_y(X_1 - C_1), o_y(X_3 - C_3)]$$
$$\frac{\partial w}{\partial R} = [0_{1x3}, 0_{1x3}, (X_3 - C_3)]$$

6.5 Rotation Matrix in terms of Quaternion

The rotation matrix R in terms of the quaternion $q = [q_x, q_y, q_z, q_w]^T$ is:

$$R = \begin{bmatrix} 1 - 2q_y^2 - 2q_z^2 & -2q_w q_z + 2q_x q_y & 2q_w q_y + 2q_x q_z \\ 2q_x q_y + 2q_w q_z & 1 - 2q_x^2 - 2q_z^2 & -2q_w q_x + 2q_y q_z \\ 2q_x q_z - 2q_w q_y & 2q_y q_z + 2q_w q_x & 1 - 2q_x^2 - 2q_y^2 \end{bmatrix}$$

The partial derivatives of R with respect to quaternion q are represented as follows:

$$\frac{\partial R}{\partial q} = \begin{bmatrix} \frac{\partial R_{11}}{\partial q} & \frac{\partial R_{12}}{\partial q} & \frac{\partial R_{13}}{\partial q} \\ \frac{\partial R_{21}}{\partial q} & \frac{\partial R_{22}}{\partial q} & \frac{\partial R_{23}}{\partial q} \\ \frac{\partial R_{31}}{\partial q} & \frac{\partial R_{32}}{\partial q} & \frac{\partial R_{33}}{\partial q} \end{bmatrix}$$

where

$$\frac{\partial R_{11}}{\partial q} = \begin{bmatrix} 0 & -4q_y & -4q_z & 0 \end{bmatrix}$$
$$\frac{\partial R_{12}}{\partial q} = \begin{bmatrix} 2q_y & 2q_x & -2q_w & -2q_z \end{bmatrix}$$
$$\frac{\partial R_{13}}{\partial q} = \begin{bmatrix} 2q_z & 2q_w & 2q_x & 2q_y \end{bmatrix}$$
$$\frac{\partial R_{21}}{\partial q} = \begin{bmatrix} 2q_y & 2q_x & 2q_w & 2q_z \end{bmatrix}$$
$$\frac{\partial R_{22}}{\partial q} = \begin{bmatrix} -4q_x & 0 & -4q_z & 0 \end{bmatrix}$$
$$\frac{\partial R_{23}}{\partial q} = \begin{bmatrix} -2q_w & 2q_x & 2q_y & 2q_z \end{bmatrix}$$
$$\frac{\partial R_{31}}{\partial q} = \begin{bmatrix} 2q_z & -2q_w & 2q_x & -2q_y \end{bmatrix}$$
$$\frac{\partial R_{32}}{\partial q} = \begin{bmatrix} 2q_w & 2q_z & 2q_y & 2q_x \end{bmatrix} \quad \frac{\partial R_{33}}{\partial q} = \begin{bmatrix} -4q_x & -4q_y & 0 & 0 \end{bmatrix}$$

6.6 Gauss Newton Method

The update step Δx is given by the equation:

$$\Delta x = (J^T J)^{-1} J^T (b - f(x))$$

Main computational bottleneck is the inverse of Hessian

6.7 Sparse Bundle Adjustment

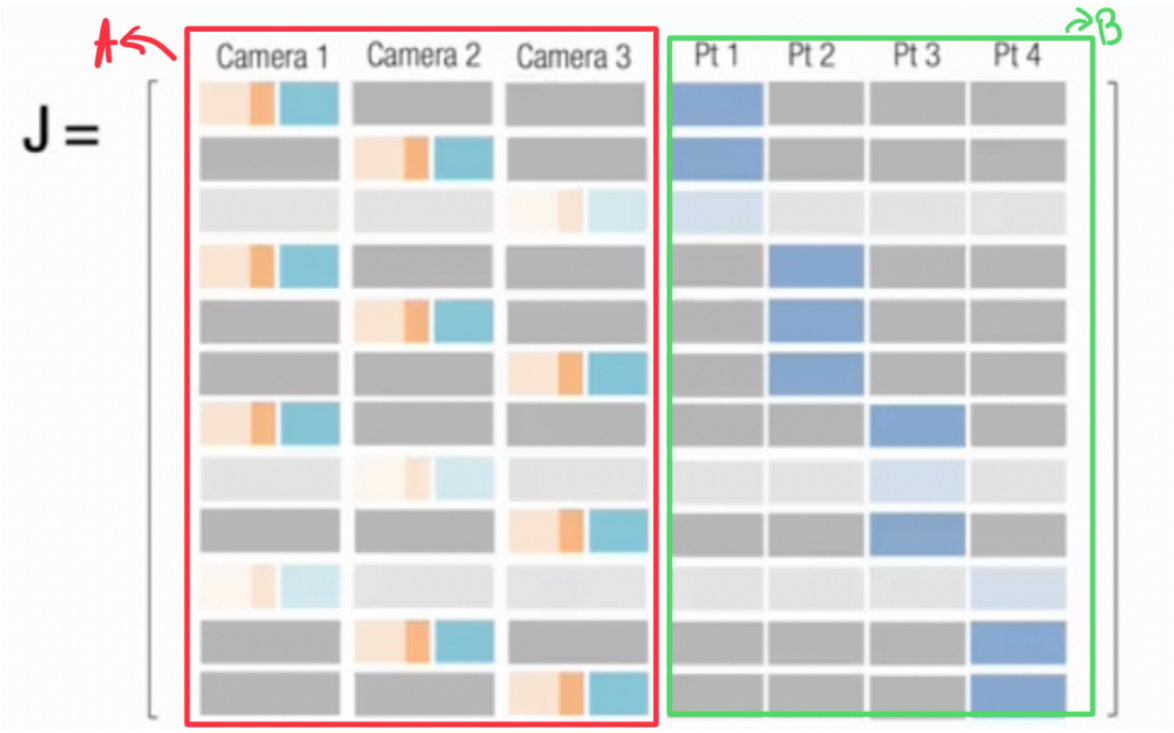

Figure 6.1: Splitting of jacobian matrix

We first split Jacobian matrix into A and B which are derivative with respect to camera parameters and 3d points respectively. as shown in Figure 6.1

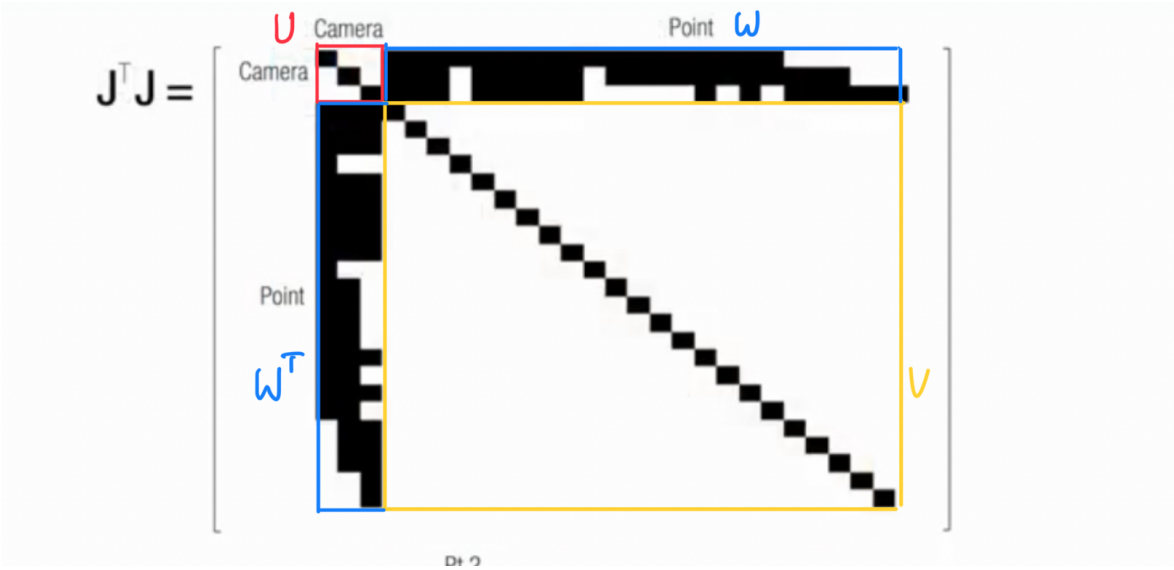

Figure 6.2: Splitting of hessian matrix

Then we compute following section of hessian matrix as shown in 6.2

$$U_j = \sum_i A_{ij}^T A_{ij}$$

$$V_i = \sum_j B_{ij}^T B_{ij}$$

$$\epsilon_{a_j} = \sum_i A_{ij}^T \epsilon_{ij}$$

$$\epsilon_{b_j} = \sum_j B_{ij}^T \epsilon_{ij}$$

$$W_{ij} = A_{ij}^T B_{ij}$$

$$Y_{ij} = W_{ij}^T V_i^{-1}$$

Then we compute,

$$S_{jk} = -\sum_i Y_{ij} W_{ij}^T + U_j \delta_{jk}$$

$$S\Delta x_c = e_j \quad : \text{Reduced system}$$

$$where, e_j = e_{a_j} - \sum_i Y_{ij} e_{b_i}$$

$$\Delta x_p = V_i^{-1}(e_{b_i} - \sum_j W_{ij}^T \Delta x_c) \quad : \text{Back substitution}$$

So we have updates Δx_c and Δx_p for our camera parameters and 3D points.