Camera Pose and 3D Points Estimation

Udit Singh Parihar
uditsinghparihar96@gmail.com

April 20, 2023

1 Problem Statement

We are provided with images of checkerboard captured from 5 unknown camera locations. We are provided the 2D pixel locations of the captured images along with the point indicies on the checkerboard. Checkerboard dimensions are provided in metric units. In the same environment, along checkerboard 3D point cloud is also captued. 3D point cloud has 10 3D points, whose positions are unknown. Our objective is to estimate the 5 camera pose and 10 3D points.

1.1 Problem Setup

Given camera matrix K:

$$K = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \tag{1}$$

Here f_x and f_y are focal lengths equal to 200. And c_x and c_y are principal points equal to 0.

Checkerboard dimensions are 1m × 1m. Checkerboard has 10 rows and 5 columns. Thus, we can calculate the 3D coordinates on the checkerboard, with the origin in the top left corner as shown Figure 1. Since all points lie on the same plane, Z coordinate is 0 for all points. These 3D coordinates on checkerboard along with the 2D pixel coordinates are used to estimate the camera pose using Zhang method. All the camera poses and 3D point cloud location are calculated with respect to the world coordinate system, located at the origin of the checkerboard.

2 Methodology

We first use provided checkerboard 2D pixels and corresponding 3D coordinates to calculate a homography matrix, H, using Zhang method. Then we decompose, H, into rotation, R, and translation, t, using SVD, with a known camera matrix, K. The estimated camera poses, R and t, are used to triangulate the 3D point cloud locations, using the provided 2D pixel coordinates. All the cameras are used to triangulate all the 3D points in the point clouds, as opposed to using only 2 cameras to triangulate a 3D point. Camera poses and 3D point cloud locations are obtained by solving linear equations of the form: $Ax = 0$.

In order to refine the estimated camera poses, R and t, and point cloud 3D locations, we use the solution obtained from linear methods to initialize a bundle adjustment problem and the camera poses and point cloud are jointly optimized.

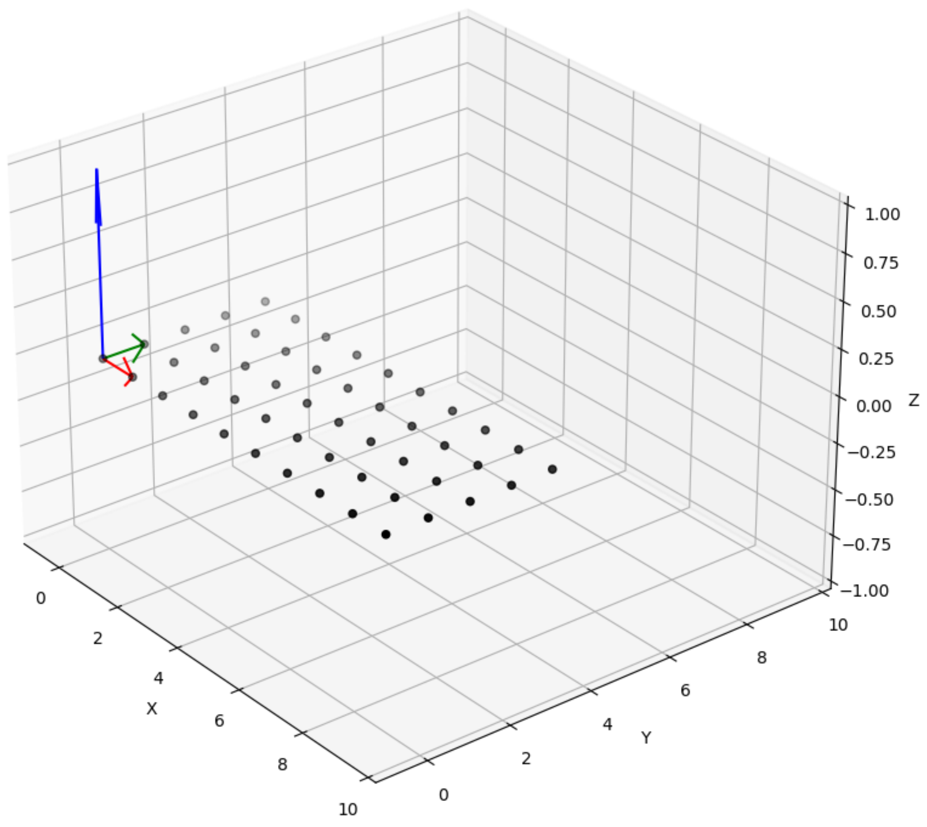

Figure 1: Checkerboard with 3D coordinates in meters

2.1 Homography Estimation

Camera projection equation can be written as:

$$\begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix}_{3\times 1} = \mathbf{K}_{3\times 3} \begin{bmatrix} r_1 & r_2 & r_3 & t \end{bmatrix}_{3\times 4} \begin{bmatrix} X_i \\ Y_i \\ Z_i \\ 1 \end{bmatrix}_{4\times 1} \tag{2}$$

Now for planar checkerboard, $Z_i = 0$. So we can write the camera projection equation as:

$$\begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix}_{3\times 1} = \mathbf{K}_{3\times 3} \begin{bmatrix} r_1 & r_2 & t \end{bmatrix}_{3\times 3} \begin{bmatrix} X_i \\ Y_i \\ 1 \end{bmatrix}_{3\times 1} \tag{3}$$

$$\begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix}_{3\times 1} = \mathbf{H}_{3\times 3} \begin{bmatrix} X_i \\ Y_i \\ 1 \end{bmatrix}_{3\times 1} \tag{4}$$

Let $P_i = \begin{bmatrix} X_i \\ Y_i \\ 1 \end{bmatrix}_{3\times 1}$, then

$$\begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix}_{3\times 1} = \begin{bmatrix} H_1^T \\ H_2^T \\ H_3^T \end{bmatrix}_{3\times 3} \mathbf{P_i} \tag{5}$$

$$x_i = \frac{H_1^T P_i}{H_3^T P_i}, \qquad y_i = \frac{H_2^T P_i}{H_3^T P_i} \tag{6}$$

Taking all terms to the left side:

$$-H_1^T P_i + 0^T + x_i H_3^T P_i = 0 \tag{7}$$
$$0^T - H_2^T P_i + y_i H_3^T P_i = 0 \tag{8}$$

$$\begin{bmatrix} -P_i^T & 0^T & x_i P_i^T \\ 0^T_{1\times 3} & (-P_i^T)_{1\times 3} & (y_i P_i^T)_{1\times 3} \end{bmatrix}_{2I\times 9} \begin{bmatrix} H_1 \\ H_2 \\ H_3 \end{bmatrix}_{9\times 1} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}_{2I\times 1} \tag{9}$$

Where I is the number of 2D-3D correspondences from the checkerboard. In our case, $I = 50$. For each 2D-3D correspondence, we have 2 equations, so we have 100 equations in total. Let $\mathbf{M}$ be the matrix on the left side, then:

$$\mathbf{M}_{100\times 9} \begin{bmatrix} H_1 \\ H_2 \\ H_3 \end{bmatrix}_{9\times 1} = \mathbf{0}_{100\times 1} \tag{10}$$

For obtaining the solution, $\vec{h}$, we can use SVD decomposition of $\mathbf{M}$:

$$\mathbf{M} = \mathbf{U\Sigma V}^T \tag{11}$$
$$\vec{h} = V[:, -1] \tag{12}$$
$$H = \begin{bmatrix} h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 \\ h_7 & h_8 & h_9 \end{bmatrix} \tag{13}$$

Ensuring that the checkerboard lies in front of the camera, then t_z or $H_{3,3}$ should be positive. Therefore, we can enforce this by multiplying H with a sign of $H_{3,3}$.

$$\mathbf{H} = sign(H_{3,3})\mathbf{H} \tag{14}$$

2.2 Reprojection Error from Homography

We can write the projected 2D points as:

$$\begin{bmatrix} x_i^{'} \\ y_i^{'} \\ 1 \end{bmatrix}_{3\times 1} = \mathbf{H}_{3\times 3}\mathbf{P_i}_{3\times 1} \tag{15}$$

Reprojection error with respect to actual 2D pixels x_i and y_i is given by:

$$error = \sum_{i=0}^{49} || \begin{bmatrix} x_i^{'} \\ y_i^{'} \\ 1 \end{bmatrix}_{3\times 1} - \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix}_{3\times 1} ||_2 \tag{16}$$

The above reprojection error for the homography for 5 camera poses is given in the Table 1.

Camera Pose	Reprojection Error
0	1.160
1	1.144
2	1.167
3	1.146
4	1.268
Mean	1.177

Table 1: reprojection Error for Homography

2.3 Camera Pose Estimation

We can obtain corresponding rotation matrix, $\mathbf{R}$, and translation vector, $\mathbf{t}$, from the homography matrix, $\mathbf{H}$, using the following equations:

$$\mathbf{H} = \mathbf{K}\begin{bmatrix}\mathbf{r_1} & \mathbf{r_2} & \mid & \mathbf{t}\end{bmatrix} \tag{17}$$

$$\mathbf{K}^{-1}\mathbf{H} = \begin{bmatrix}\mathbf{r_1} & \mathbf{r_2} & \mid & \mathbf{t}\end{bmatrix} \tag{18}$$

Let, $\mathbf{K}^{-1}\mathbf{H} = \begin{bmatrix}h_1 & h_2 & h_3\end{bmatrix}$, then rotation matrix, $\mathbf{M}$ and translation vector, $\mathbf{t}$, can be obtained as:

$$\mathbf{M} = \begin{bmatrix}\mathbf{m_1} & \mathbf{m_2} & \mathbf{m_3}\end{bmatrix} \tag{19}$$

$$\mathbf{m_1} = \frac{\mathbf{h_1}}{||\mathbf{h_1}||} \tag{20}$$

$$\mathbf{m_1} = \frac{\mathbf{h_2}}{||\mathbf{h_2}||} \tag{21}$$

$$\mathbf{m_3} = \mathbf{m_1} \times \mathbf{m_2} \tag{22}$$

$$\mathbf{t} = \frac{\mathbf{h_3}}{||\mathbf{h_1}||} \tag{23}$$

The scaling factor of translation is the same scaling factor applied to the have unit norm of $\mathbf{m_1}$.

Since the obtained $\mathbf{M}$ is not a rotation matrix, because we haven't enforced any orthogonality constraints between $\mathbf{m_1}$ and $\mathbf{m_2}$. And determinant of $\mathbf{M}$ is also not enforced to be 1. We can obtain a rotation matrix, $\mathbf{R} \in \mathrm{SO}(3)$, by enforcing these constraints.

$$\mathbf{R} = \arg \min_{\mathbf{R} \in \mathrm{SO}(3)} ||\mathbf{R} - \mathbf{M}||_F \tag{24}$$

$$\mathbf{U}, \mathbf{S}, \mathbf{V}^T = \mathrm{SVD}(\mathbf{M}) \tag{25}$$

$$\mathbf{R} = \mathbf{U} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & \det(\mathbf{U}\mathbf{V}^T) \end{bmatrix} \mathbf{V}^T \tag{26}$$

The final solution for 5 camera poses in world frame can be seen in Figure 2.

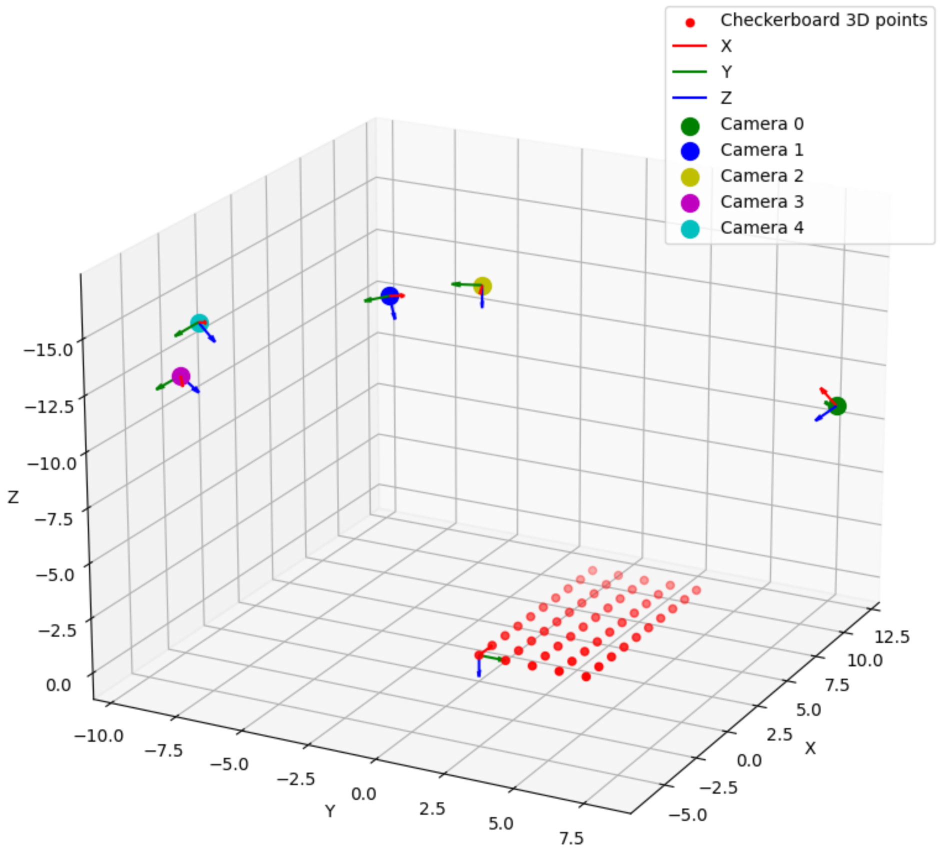

Figure 2: Camera poses from Homogrpahy Decomposition of 2D-3D checkerboard correspondences

2.4 Reprojection error from Rotation and Translation

We can write the projected points $\left[\mathbf{x_i}' \mathbf{y_i}'\right]^T$, in terms of given camera matrix, $\mathbf{K}$ and estimated rotation matrix, $\mathbf{R}$ and translation vector, $\mathbf{t}$, as:

$$\begin{bmatrix} \mathbf{x_i}' \\ \mathbf{y_i}' \end{bmatrix} = \mathbf{K} \begin{bmatrix} \mathbf{R} & \mathbf{t} \end{bmatrix} \begin{bmatrix} \mathbf{X_i} \\ \mathbf{Y_i} \\ \mathbf{Z_i} \\ 1 \end{bmatrix} \tag{27}$$

Here, $\mathbf{Z_i} = 0$ for all the points on the checkerboard. Thus, reprojection error with respect to actual 2D pixels x_i and y_i is given by:

$$error = \sum_{i=0}^{49} || \begin{bmatrix} x_i' \\ y_i' \\ 1 \end{bmatrix}_{3\times 1} - \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix}_{3\times 1} ||_2 \tag{28}$$

The reprojection error for the camera poses $\mathbf{R}$ and $\mathbf{t}$ for 5 camera poses is given in the Table 2.

Camera Pose	Reprojection Error
0	49.883
1	27.429
2	30.867
3	20.680
4	18.578
Mean	29.487

Table 2: Reprojection error from camera poses

2.5 3D Point Cloud Estimation

Now we know the camera poses in world frame, and we are given the 2D pixel coordinates of the points on the point cloud, $\vec{x_i} = \begin{bmatrix} x_i \\ y_i \end{bmatrix}$, we can estimate the 3D point cloud points, $\vec{X_i}$. Let our estimated, projection matrix, $\mathbf{P_i} = \mathbf{K} \begin{bmatrix} \mathbf{R_i} & \mathbf{t_i} \end{bmatrix}$. Then, a single point $\vec{X_1}$ can be triangulated from multiple views as:

$$\begin{bmatrix} u_1 \\ v_1 \\ 1 \end{bmatrix} = \vec{x_1} = \mathbf{P_1} \vec{X_1} \tag{29}$$

$$u_1 = \frac{(\mathbf{P_1}^1)^T \vec{X_1}}{(\mathbf{P_1}^3)^T \vec{X_1}} \tag{30}$$

$$v_1 = \frac{(\mathbf{P_1}^2)^T \vec{X_1}}{(\mathbf{P_1}^3)^T \vec{X_1}} \tag{31}$$

$$\begin{bmatrix} u_1(P_3^1)^T - (P_1^1)^T \\ v_1(P_3^1)^T - (P_2^1)^T \\ u_2(P_3^2)^T - (P_1^2)^T \\ v_2(P_3^2)^T - (P_2^2)^T \\ u_3(P_3^3)^T - (P_1^3)^T \\ v_3(P_3^3)^T - (P_2^3)^T \\ u_4(P_3^4)^T - (P_1^4)^T \\ v_4(P_3^4)^T - (P_2^4)^T \\ u_5(P_3^5)^T - (P_1^5)^T \\ v_5(P_3^5)^T - (P_2^5)^T \end{bmatrix}_{10\times 4} X_1 = 0$$

The solution to this system of equations, $A_{2\times N}\vec{X}_1 = 0$, is given by SVD decomposition of A:

$$\mathbf{U}, \mathbf{S}, \mathbf{V}^T = \mathrm{SVD}(\mathbf{A}) \tag{32}$$

$$\vec{X}_1 = \mathbf{V}[:, -1] \tag{33}$$

$$\vec{X}_1 = \frac{\vec{X}_1}{\vec{X}_1[3]} \tag{34}$$

The final point cloud points, $\vec{X}_i$ along with camera poses and checkerboard points are shown in Figure 3

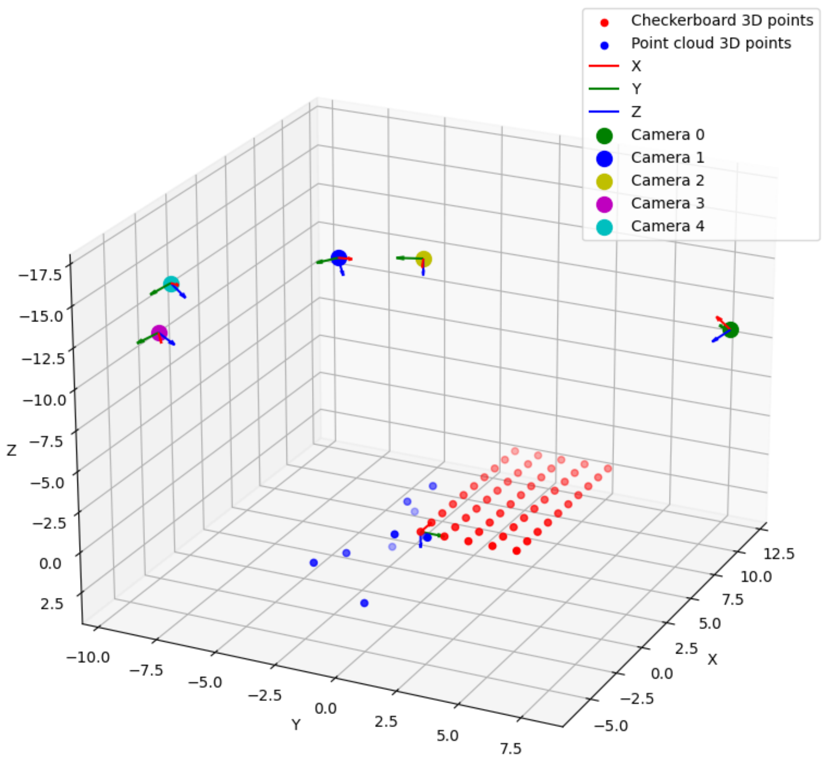

Figure 3: Point cloud points, $\vec{X}_i$, camera poses, $\mathbf{P_i}$, and checkerboard points

Reprojection error for all the points in the point cloud for all the cameras is shown in Table 3.

Camera Pose	Reprojection Error
0	1.222
1	1.110
2	0.816
3	1.636
4	1.545
Mean	1.266

Table 3: Reprojection error of point cloud for all cameras

2.6 Bundle Adjustment

The estimated camera poses and point cloud points are used as initial estimates for bundle adjustment. Bundle adjustment is done to jointly optimized all the unknown parameters, i.e. camera poses and point cloud points. Camera projection equation for a 3D point $\vec{X_i}$, for camera P_j, observing pixel, $\vec{x_{ij}}$, is given by:

$$\begin{bmatrix} u_{ij} \\ v_{ij} \\ 1 \end{bmatrix} = \vec{x_{ij}} = \mathbf{P_j}\vec{X_i} \tag{35}$$

$$\begin{bmatrix} u_{ij} \\ v_{ij} \\ 1 \end{bmatrix} = \begin{bmatrix} P_j^{1\,T} \\ P_j^{2\,T} \\ P_j^{3\,T} \end{bmatrix} \vec{X_i} \tag{36}$$

$$\tag{37}$$

Reprojection error would be:

$$L = \sum_{j=1}^{5}\sum_{i=1}^{10}\left(\left(u_{ij} - \frac{P_j^{1\,T} X_i}{P_j^{3\,T} X_i}\right)^2 + \left(v_{ij} - \frac{P_j^{2\,T} X_i}{P_j^{3\,T} X_i}\right)^2\right) \tag{38}$$

Our objective is:

$$\min_{\vec{X_i}, \mathbf{P_j}} L \tag{39}$$

Since, intrinsic matrix, $\mathbf{K}$ is known, we don't need to optimize over entire $\mathbf{P_j}$, but only over translation vector, $\vec{t_j}$ and rotation matrix, $\mathbf{R_j}$. We can write a rotation matrix, $\mathbf{R}$ in axis angle form, $\vec{w}$ as:

$$\theta = ||\vec{w}|| \tag{40}$$

$$k = \frac{\vec{w}}{||\vec{w}||} \tag{41}$$

$$K = \begin{bmatrix} 0 & -k_3 & k_2 \\ k_3 & 0 & -k_1 \\ -k_2 & k_1 & 0 \end{bmatrix} \tag{42}$$

$$\mathbf{R} = \mathbf{I} + \sin(\theta)\mathbf{K} + (1 - \cos(\theta))\mathbf{K}^2 \tag{43}$$

Thus for a single camera pose, we have 6 unknowns, i.e. 3 for translation vector, $\vec{t_j}$ and 3 for rotation vector, $\vec{w_j}$. For a single point in the point cloud, we have 3 unknowns, X_i, Y_i, Z_i. Thus, we have $6N + 3M$ unknowns, where M is the number of points in the point cloud and N is the number of cameras. The dimension of the residual vector is $2MN$. The Jacobian matrix is $2MN \times (6N + 3M)$. For our case $M = 10$ and $N = 5$, thus the dimension of the Jacobian matrix is 100×60.

2.7 Bundle Adjustment Results

We have used *scipy.optimize.least_squares* function to solve the bundle adjustment problem. It allows to provide the residual function, initial estimates and the sparsity pattern of the Jacobian matrix. It calculates the Jacobian matrix and optimizes the non-linear and non-convex residual. The initial estimates for camera poses and point cloud points are obtained from the previous section. Our initial and final root mean squared residual can be seen in table 4.

Initial RMSE	Final RMSE
1.0155	0.770

Table 4: Root mean squared error of bundle adjustment

Since our initial estimates of camera pose and point cloud points are pretty accurate from Zhang method and multi view triangulation, therefore the bundle adjustment didn't improve the results much, and started pretty close to local minima. The bundle adjustment results can be seen in figure 4.

Values of optimized point cloud points with respect to world coordinate define at checkerboard corner is given in table 5.

Point	X	Y	Z
1	-1.862	1.164	-1.030
2	1.919	-1.140	0.137
3	-3.185	-3.029	0.981
4	0.859	-0.180	0.830
5	-1.914	-0.188	-0.836
6	-1.723	-2.336	1.003
7	-2.938	-1.022	2.984
8	1.237	-0.040	-2.126
9	0.891	-1.618	1.831
10	0.641	-0.841	-1.222

Table 5: Optimized point cloud points

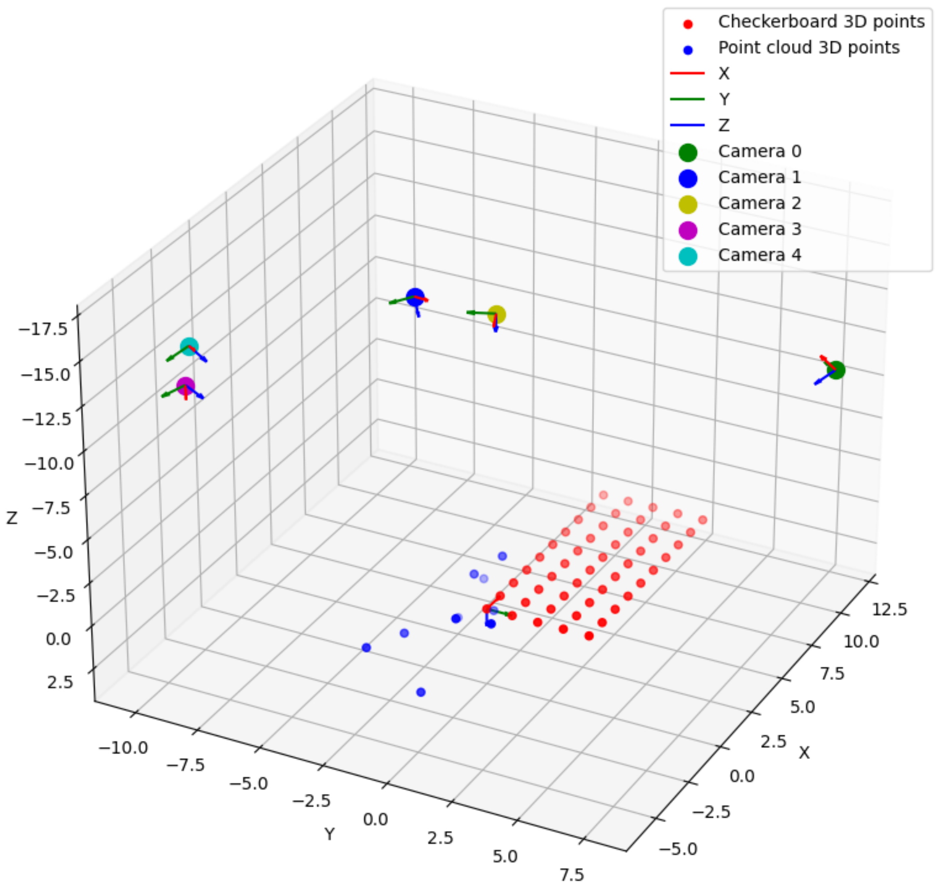

Figure 4: Bundle adjustment results