RBE549 Project 2 - SfM and NeRF

Yaşar İdikut
yidikut@wpi.edu
Using 1 late Day

Harshal Bhat
hbhat@wpi.edu
Using 1 late Day

I. Introduction

A. Data Loader Pipeline and Feature matching

The data loader is responsible for managing images and camera calibration inside the image processing pipeline. We create correspondences between important places in various images by methodically processing feature match files. To ensure accuracy, a homography transformation is used in this step to refine matches. Feature point normalization improves robustness in a variety of image situations. Next, we display the attributes of image pairs as they are depicted in Figure 1.

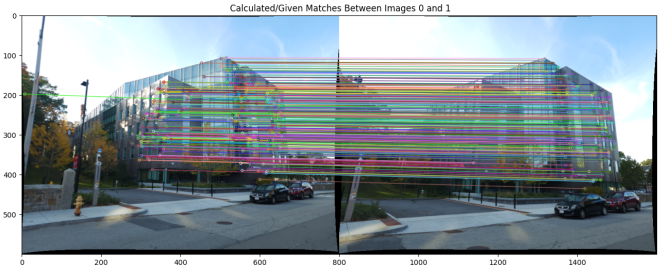

Fig. 1. SIFT Feature matching for images 1 and 2

B. Estimating Fundamental Matrix and Epipolar Constraints

We compute the fundamental matrix using a set of corresponding points. It constructs a matrix from these points, performs Singular Value Decomposition, and enforces a rank-2 constraint on the resulting matrix to obtain the fundamental matrix using the 8-point algorithm. The estimated fundamental matrix is visualized by its epipolar lines in Fig. 2. We can see that corresponding feature points lie on the computed epipolar lines, indicating an accurate result.

We use the Random Sample Consensus (RANSAC) technique to iteratively estimate the fundamental matrix. In each iteration, eight random point pairs are selected, and a candidate fundamental matrix is calculated. The projection error is then determined for each point, and those with errors less than a certain threshold are called inliers. The process is repeated for a predetermined number of iterations, and the fundamental matrix with the highest number of inliers is chosen as the best estimate.

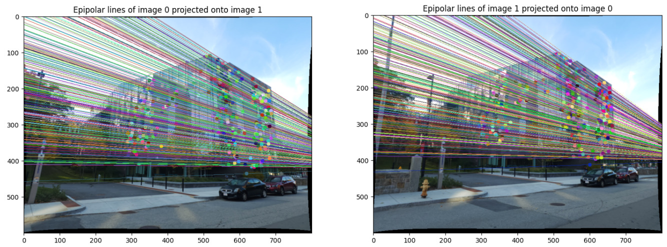
Epipolar lines of image 0 projected onto image 1
Epipolar lines of image 1 projected onto image 0
0
100
200
300
400
500
0
100
200
300
400
500
600
700

Fig. 2. Epipolar lines for images 1 projected onto 2 and images 2 on 1

C. Estimating Essential Matrix

We calculate the Essential Matrix by transforming the given fundamental matrix **bestF** using the camera calibration matrix (K) using the equation shown below

$$E = K^T F K \tag{1}$$

Subsequently, we apply Singular Value Decomposition (SVD) to the essential matrix and enforce a constraint on the singular values to ensure a valid essential matrix.

D. Extracting Camera Pose and Disambiguate the poses

We obtain a set of potential rotation matrices(R) and their corresponding translation vectors(C), representing the potential camera poses associated with the given essential matrix.

$$\text{E = U D } T^T \quad \text{and} \quad \text{W = } \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

1) $C_1 = U(:,3)$ and $R_1 = UWV^T$
2) $C_2 = -U(:,3)$ and $R_2 = UWV^T$
3) $C_3 = U(:,3)$ and $R_3 = UW^TV^T$
4) $C_4 = -U(:,3)$ and $R_4 = UW^TV^T$

If for any configuration, $\det(R_i) = -1$, the camera pose must be corrected, i.e., $C_i = -C_i$ and $R_i = -R_i$.

E. Linear and Non-linear Triangulation

We perform linear triangulation to estimate 3D coordinates of points in the world space using corresponding 2D image coordinates and camera parameters. Given camera intrinsic matrix (K), rotation matrices (R1, R2), translation vectors (C1, C2), and image coordinates (x1, x2) from two views, we construct projection matrices (P1, P2). Then iteratively compute the homogeneous coordinates of the 3D points using the SVD method for A. The resulting 3D coordinates are

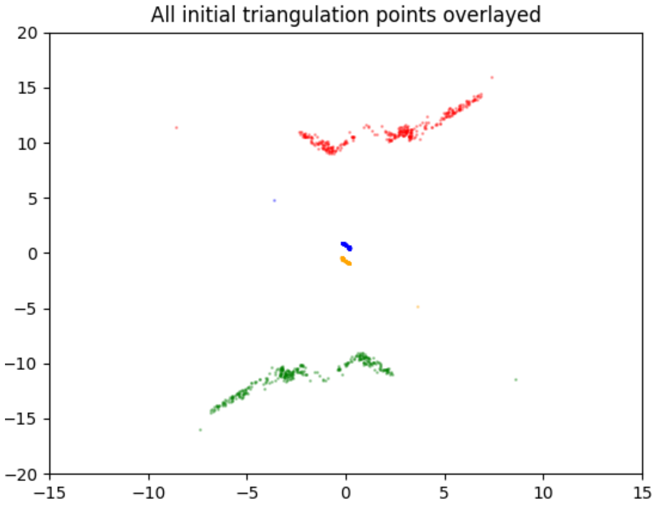

Fig. 3. Initial Triangulation with all 4 Camera Poses

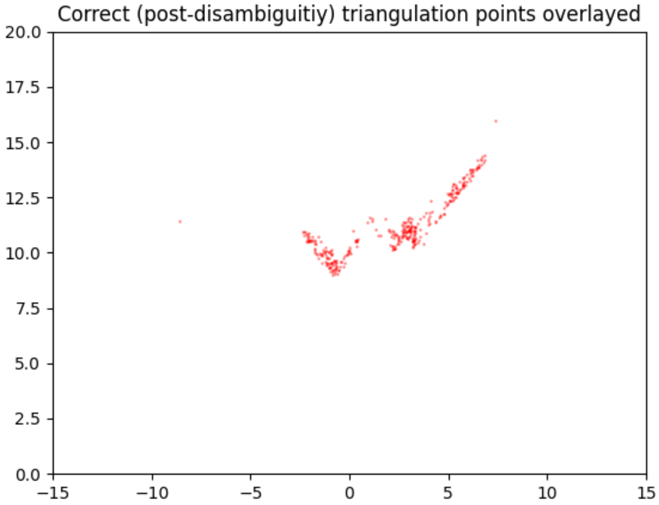

Fig. 4. Post Camera Disambiguity

compiled into an array (X), representing the triangulated points in the world space. We use chirality conditions to rule out the three inappropriate camera poses.

$$A = \begin{bmatrix} y\mathbf{p_3}^T - \mathbf{p_2}^T \\ \mathbf{p_1}^T - x\mathbf{p_3}^T \\ y'\mathbf{p_3'}^T - \mathbf{p_2'}^T \\ \mathbf{p_1'}^T - x'\mathbf{p_3'}^T \end{bmatrix} \tag{2}$$

Non-linear Triangulation is optimizing the triangulated points by minimizing the reprojection error. A slight increase in accuracy through non-linear optimization is shown in Table 1. Even with linear triangulation, the results are already acceptable due to precise camera position estimates and exact point correspondences. Figure 4 shows the comparison for Linear and Non-Linear Triangulation.

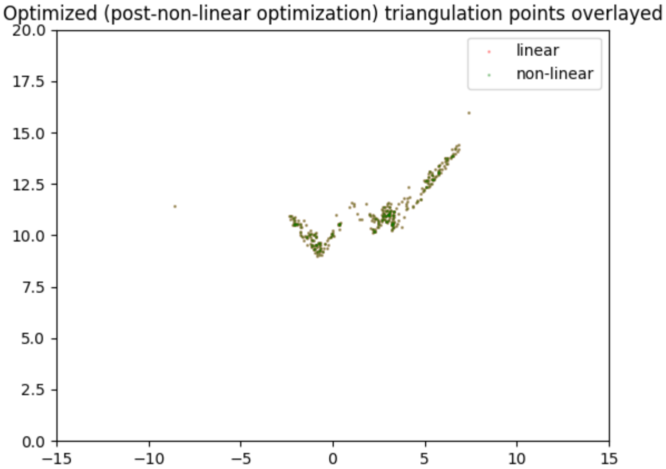

Fig. 5. Linear and Non-Linear Triangulation comparison

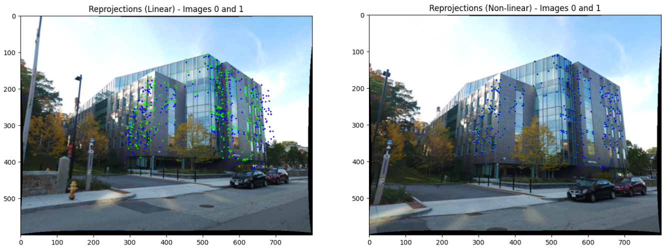
Reprojections (Linear) - Images 0 and 1
Reprojections (Non-linear) - Images 0 and 1
0
100
200
300
400
500
0
100
200
300
400
500
600
700

Fig. 6. Reprojections after Linear and Non-Linear Triangulation

F. Linear and Non-linear Perspective-n-Points

After matching the two first images, the rest of the images are registered one by one. The pipeline for registration is as follows. First, the features between the previous images ($Img_0, ...Img_{i-1}$) and the next image (Img_i) are matched as given in the txt file. We iterate thrice through previous images, feature points, and then matches to construct a global list of all matching feature points for all images and their corresponding world points. The process extracts matching feature points and their 3D correspondences from previous images, appending them to *allmatchingX* and *allmatchingx*. The process aims to establish connections between feature points across multiple views.

We implement the Perspective-n-Point (PnP) algorithm for camera pose estimation from 3D-2D point correspondences. We normalize the 2D image coordinates using the inverse of the camera intrinsic matrix. A matrix A based on the input 3D-2D correspondences as shown in the equation below. Singular Value Decomposition (SVD) is applied to A, and the last column of the resulting matrix V is reshaped to form the projection matrix P. The camera center (C) and rotation matrix (R) are extracted from P, with additional steps taken to ensure orthonormality and correct orientation. The final camera pose, represented by R and C, is returned.

$$A = \begin{bmatrix} X & Y & Z & 1 & 0 & 0 & 0 & -xX & -xY & -xZ & -x \\ 0 & 0 & 0 & 0 & X & Y & Z & 1 & -yX & -yY & -yZ & -y \end{bmatrix} \tag{3}$$

The initial estimate of Linear PnP error is very noisy. We perform 50 iterations for all points using the estimated pose and check how many points have errors below a specified threshold. The pose with the highest number of inliers is retained as the best estimate in our PnP RANSAC algorithm.

Non-linear PnP performs optimization based on the mean reprojection error.

G. Linear and Non-Linear Triangulation

Given the camera pose estimation matrices ($setC_i$ and $setR_i$) and the features on the image coordinate frame, we can use Linear and Non-Linear triangulation to solve for their corresponding world coordinates ($setX_i$). Using this new estimation for 3D world points, we visualize the map again.

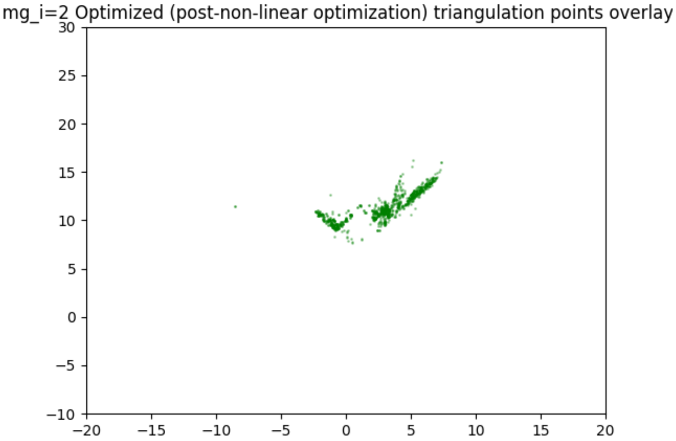

Fig. 7. 3D world point visualization after registering third image (Img_2) without bundle adjustment

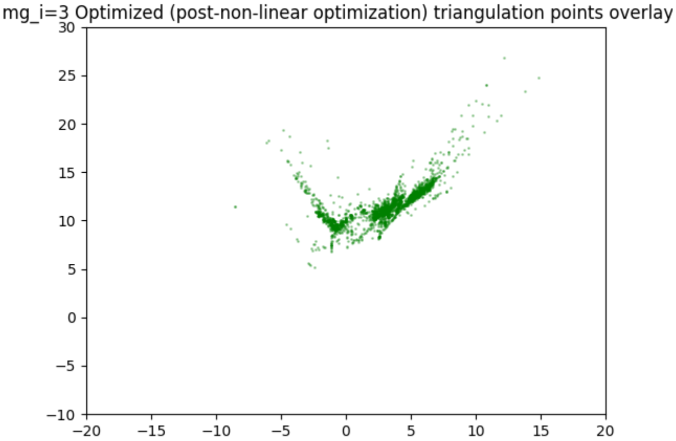

Fig. 8. 3D world point visualization after registering fourth image (Img_3) without bundle adjustment

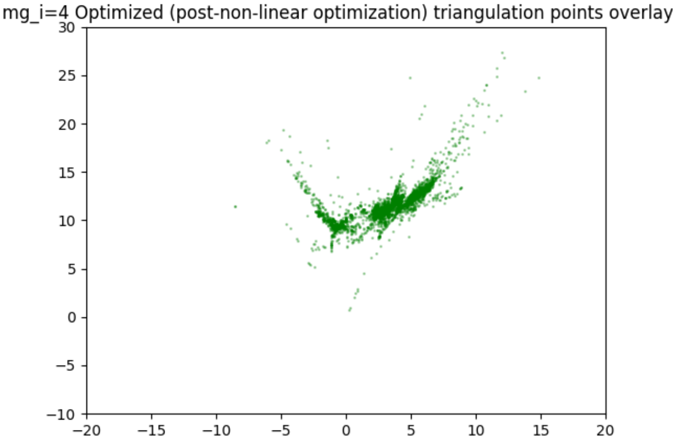

Fig. 9. 3D world point visualization after registering fifth image (Img_4) without bundle adjustment

H. Bundle Adjustment

In bundle adjustment, the algorithm takes in ($setC_i$, $setR_i$), and $setX_i$) and optimizes their values to minimize the reprojection error. As can be seen from the tables, the reprojection error is numerically minimized, however, the 3D points visualizations suggest that they are not minimized for representing the structure. To solve this optimization problem, we use the least_squares method from SciPy. However, without specifying the sparsity matrix, each optimization takes around a minute to calculate. Luckily, we can create a matrix A with boolean values, to denote which input parameters($setC_i$, $setR_i$, and $setX_i$) affect which output values(2D error residuals). We pass this matrix to SciPy and bundle adjustment only takes milliseconds. The mathematical minimization of bundle adjustment can be seen in the following tables.

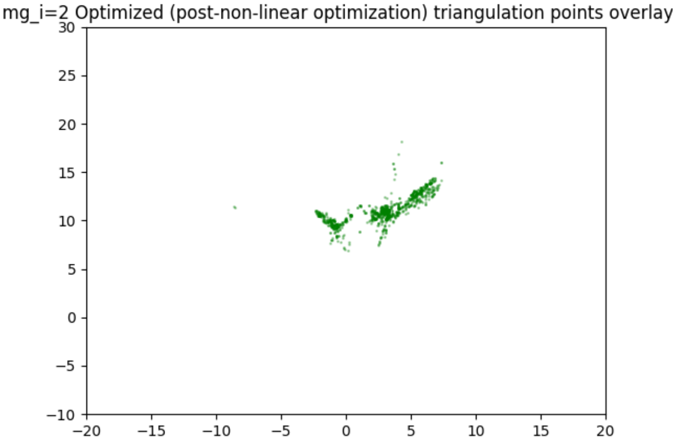

Fig. 10. 3D world point visualization after registering third image (Img_2) with bundle adjustment

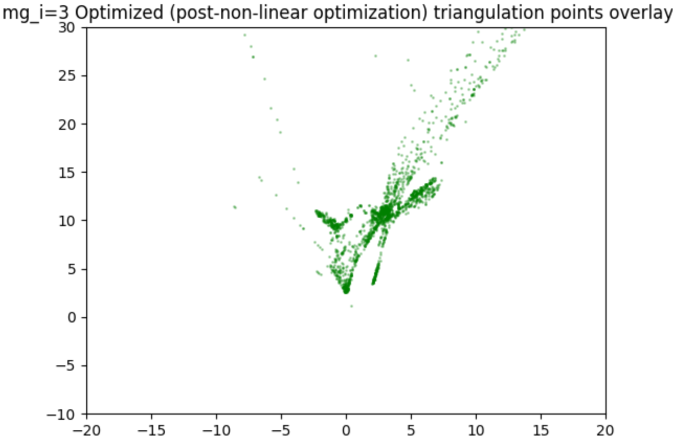

Fig. 11. 3D world point visualization after registering fourth image (Img_3) with bundle adjustment

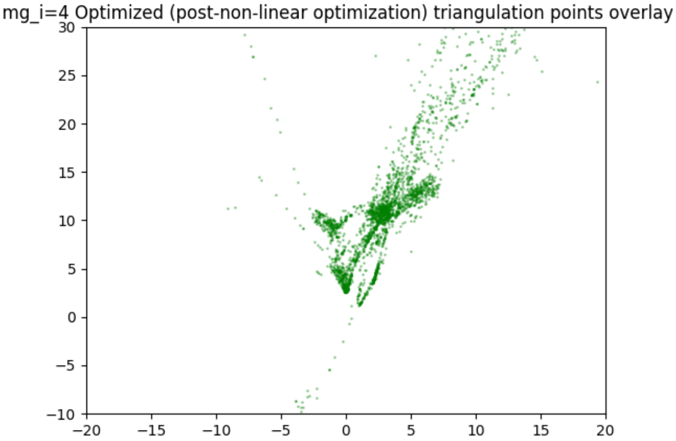

Fig. 12. 3D world point visualization after registering fifth image (Img_4) with bundle adjustment

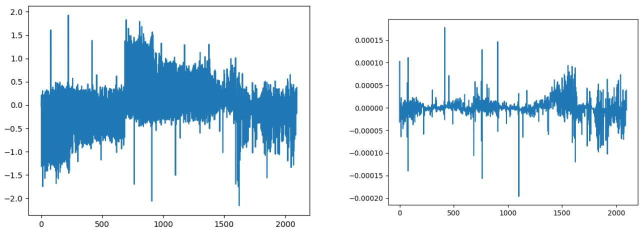

Fig. 13. Before and after residuals for testing the bundle adjustment algorithm

I. *Results*

Stage	Error
Linear Triangulation	1.92
Non-Linear Triangulation	1.91

TABLE I

AVERAGE REPROJECTION ERRORS BETWEEN FRAMES 1 2

Stage	Errors for Image		
	Img_2	Img_3	Img_4
Linear PnP	627.53	1142.73	1272.64
Non-Linear PnP	2.52	98.44	61.02
Non-Linear Triangulation	0.78	7.00	1.11
After Bundle Adjustment	2.33e-10	3.55e-10	4.62e-10

TABLE II

AVERAGE REPROJECTION ERROR IN PIXELS IN THE REGISTRATION PIPELINE FOR EACH NEWLY ADDED IMAGE

J. *Extra Credit*

We used our Pipeline with modification to the feature extraction pipeline and implemented Feature Desctiptor and matching class, that bypasses the need for matching*.txt files. Below are the images for our custom dataset, undistorted images using cv2 functions, and calculated the Camera calibration matrix. The chair is the main object of interest over here. As can be seen, there is a cluster of points in the central portion of the image.

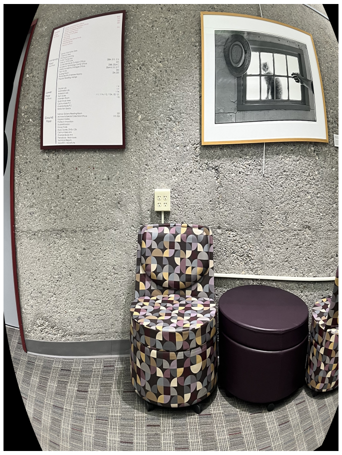

Fig. 14. Sample Image from the Custom Dataset

II. CONCLUSION

We performed Structure from Motion and reconstructed a sparse representation of the Unity Building using the classical approach. We addressed the Perspective-n-Point (PnP) problem, basic matrix estimation, Cheirality requirement for 3D reconstruction, and feature matching. A deeper exploration of 3D scene reconstruction is made possible by Bundle Adjustment, which improved poses and 3D points, and RANSAC, which increased robustness.

REFERENCES

[1] https://scipy-cookbook.readthedocs.io/items/bundle adjustment.html

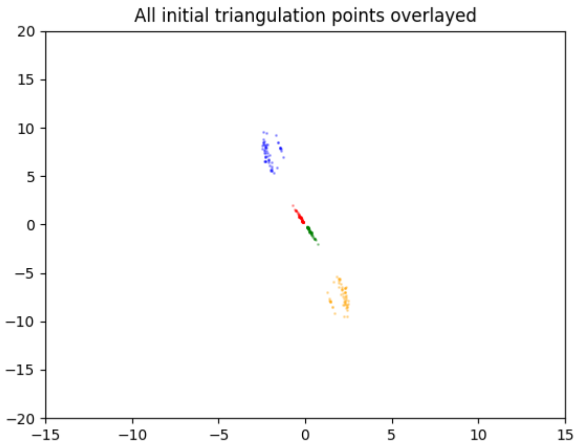

Fig. 15. Triangulation for 4 camera poses for the custom dataset

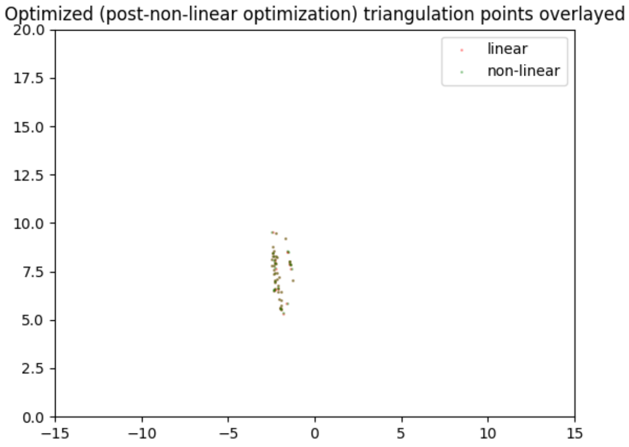

Fig. 16. Linear and Non-Linear Triangulation comparison for the custom dataset

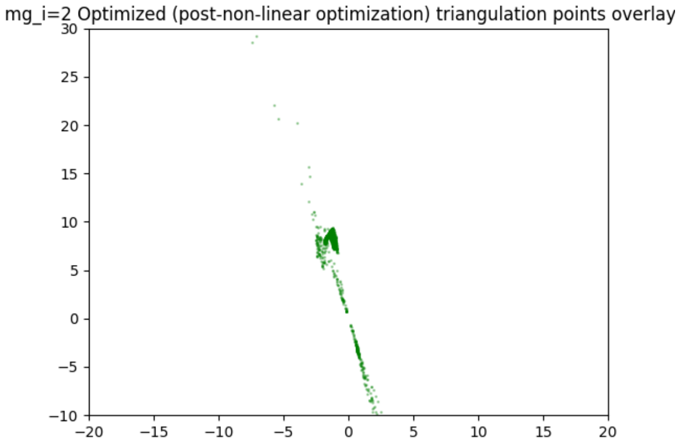

Fig. 17. 3D world point visualization after registering third image (Img_2) with bundle adjustment for the custom dataset

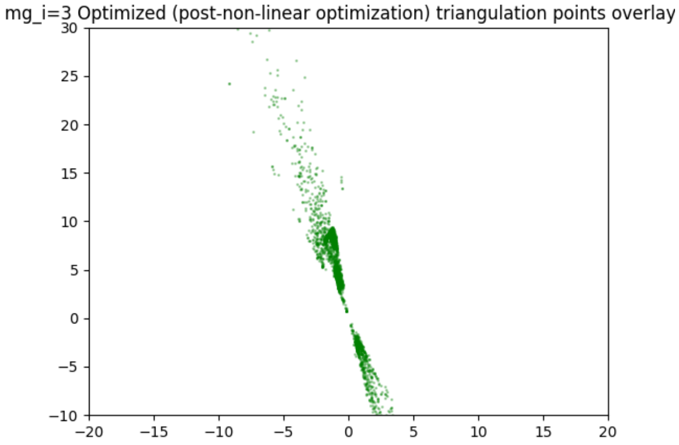

Fig. 18. 3D world point visualization after registering third image (Img_3) with bundle adjustment for the custom dataset