Project 2: Structure from Motion and NeRF

Radha Saraf
Email: rrsaraf@wpi.edu

Sai Ramana Kiran Pinnama Raju
Email: spinnamaraju@wpi.edu

***Abstract*—The aim of this project is to implement an end-to-end pipeline for Structure from Motion (SfM) and explore the same using the modern deep learning approach using Neural Radiance Fields (NeRF)**

I. Phase I: Structure from Motion

A. Visibility matrix

We are given feature correspondences between the different pairs of images in the form of matching files. If any of the available camera poses were denoted by i, then the i^{th} matching file contained the feature correspondences between the i^{th} pose and all the remaining poses in the sequential order. Because of the way this is framed, each entry in all the matching files represents a unique 3D point in the world.

A visibility matrix provides us a way to keep track of the visibility of points in the various camera poses. If the different 3D points were denoted by j, then V_{ij} denoted whether the j^{th} point was visible in the i^{th} camera pose.

The visibility matrix provided a handy way of dealing with the requirements at different stages of the sfM pipeline like feature correspondences, 2d-3d correspondences, and update of triangulated world coordinates.

B. Estimating Fundamental Matrix

Two camera poses in stereo geometry are said to follow an epipolar constraint. Given the projection of a 3D point in one of the camera poses, the corresponding projection in the other pose is constrained to be on a line. The relationship between the two projections is denoted by the Fundamental matrix.

The fundamental matrix presents a homogenous system of linear equations with 9 unknowns. Singular Value Decomposition(SVD) can be used to solve this system. Post solving the system, the rank constraint is enforced by making the last singular value zero and recalculating the fundamental matrix.

C. Refining matches using RANSAC

The feature correspondences from the matching files contained some incorrect matches. RANSAC was used to refine the correspondences with Fundamental matrix providing for the model used.

The figures 1 and 2 show the matches before and after running RANSAC on the feature correspondences.

D. Estimating Essential Matrix from Fundamental Matrix

The Essential matrix is an extension of the Fundamental matrix in that, it extends the relationship from the image view to the camera view. This results in obtaining a physical/geometrical constraint in the relative poses of the two cameras in the stereo setup.

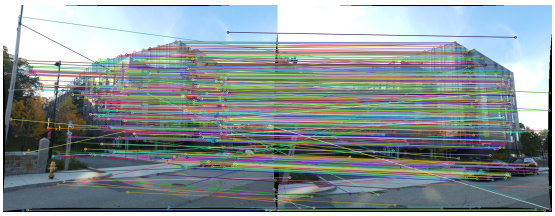

Fig. 1. Feature matches for images 1, 2 before RANSAC

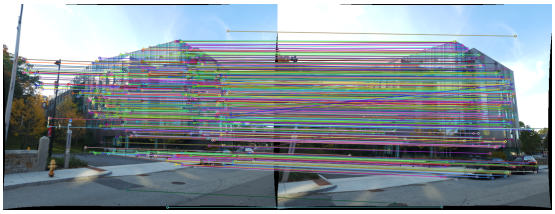

Fig. 2. Feature matches for images 1, 2 after RANSAC

Figure 3 shows the Epipolar lines for images 1 and 2

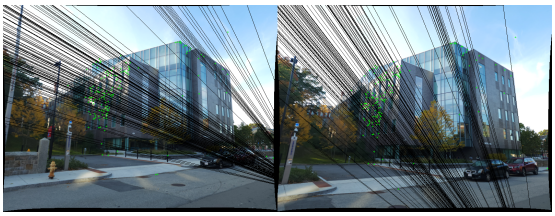

Fig. 3. Epipolar lines for images 1, 2

E. Estimating Camera Pose from Essential Matrix

We decomposed the Essential matrix into rotation and translation matrices using SVD and some clever mathematical tricks described in the problem statement. We also made sure that matrix that was rotation matrix that was derived is an actual rotation matrix by doing SVD cleanup.

F. Linear Triangulation

Linear Triangulation involves triangulating the 3D coordinates of the world points from the features correspondences, the camera matrix and the camera poses. A single 2D-3D correspondence gives us two equations

A single 3D point in the world, and its projections in each of the two images, (image coords) form two lines in 3D space. In an ideal scenario, wherein two different camera poses are capturing the same scene, these lines should intersect. The intersection point is our solution to the triangulation problem. However, if the lines are non-coplanar(which is usually the case), we can't find an exact solution and so we find the best approximation to the solution instead.

From one perspective projection, we can get two equations. We have two such projections(given a pair of corresponding image coordinates), (i.e) a total of 4 equations. But these equations have an unknown scale parameter, s, unique to each projection. So our unknowns add up to 5- (X, Y, Z, s1, s2) which make for an under-determined system that cannot be solved.

To get rid of the scale parameters, we use the fact that a vector's cross product with itself is equal to zero. We create a system of linear equations which is again solved using SVD.

Figure 4 shows the results of linear triangulation using features from images 1 and 2.

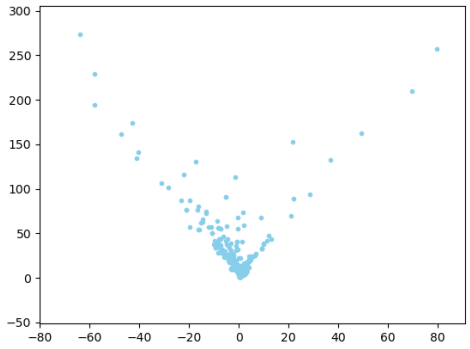

Fig. 4. Linear Triangulation

G. Triangulation check for chierality condition

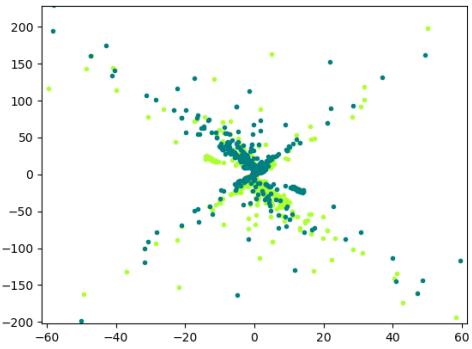

Fig. 5. Disambiguation of camera poses

The Essential matrix presents us with four possible configurations of the rotation matrix and the translation vector. One out of these is correct. The Chierality check helps us distinguish the correct one from the others. It's a simple condition that enforces the fact that points in front of the two camera poses should have positive values for Z coordinates. The Z values are obtained from the triangulation of the feature correspondences for images 1 and 2.

Figure 5 shows the 4 triangulation results for the 4 different configurations obtained from estimating the camera pose from the essential matrix. Only two can be seen in the image because the remaining two are beneath the visible ones.

1) Non Linear Triangulation: After obtaining the correct pose configuration post chierality check, the 3D world coordinates are updated to consider only the inliers from the cheirality condition. These coordinates are further refined with non-linear optimization to reduce the reprojection error using Scipy's minimize() or optimize().

Figure 6 shows the results of non-linear triangulation.

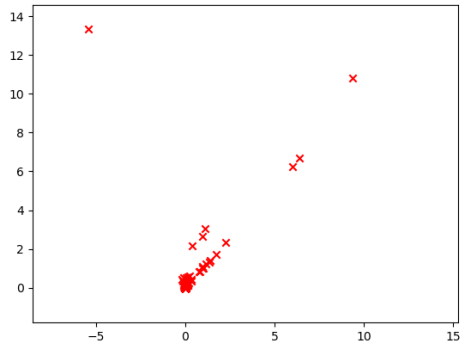

Fig. 6. Non-Linear Triangulation

H. Bundle Adjustment

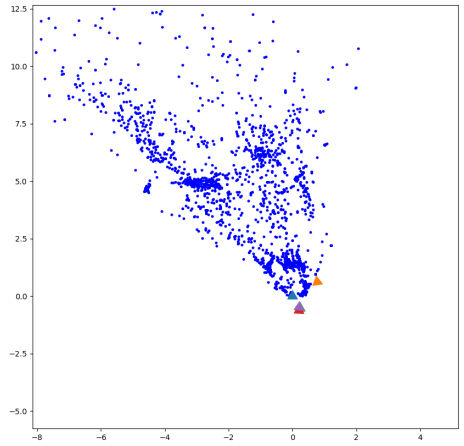

Fig. 7. Bundle Adjustment

Fig. 8. Deep learning output using TPS

Figure 7 shows the results bundle adjustment.

II. Phase II: Neural Radiance Fields (NeRF)

In the deep learning part of this project, we implemented NeRF [2] which creates a 3D render of the entire object of interest with few sparse images. We made the following modifications to simplify the original implementation.

1) Network width with only 64 channels
2) Positional encoding with 16 frequencies
3) implemented only coarse network

Figure 8 shows a sample output of our network after just 40 epochs. GIF of the lego model output is attached along with the project output.

References

[1] https://www.microsoft.com/en-us/research/wp-content/uploads/2016/02/tr98-71.pdf

[2] Mildenhall, Ben, et al. "Nerf: Representing scenes as neural radiance fields for view synthesis." Communications of the ACM 65.1 (2021): 99-106.