P2 - Building Built In Minutes: SfM

Mihir Deshmukh
Robotics Engineering
Worcester Polytechnic Institute
Email: mpdeshmukh@wpi.edu

Ashwin Disa
Robotics Engineering
Worcester Polytechnic Institute
Email: amdisa@wpi.edu

Using 1 LATE day

***Abstract*—In this project assignment, we reconstruct a 3D scene and simultaneously obtain the camera poses of a monocular camera with respect to the given scene also known as Structure from Motion. We create the rigid structure from a set of images with different view points.**

I. Phase 1: Structure from Motion

A. Dataset

We are given with a set of 5 images of Unity Hall at WPI as shown in fig. 1, using a Samsung S22 Ultra's primary camera at f/1.8 aperture, ISO 50 and 1/500 sec shutter speed. The camera is calibrated after resizing using a Radial-Tangential model with 2 radial parameters and one tangential parameter using the MATLAB R2022a's Camera Calibrator Application beforehand. The images provided are already distortion-corrected and resized to $800 \times 600px$.

Fig. 1. Dataset.

B. Feature Matching, estimating fundamental matrix and RANSAC

We are also given matched features. We reject the outliers from the same set using RANSAC. We do so using an estimation of the fundamental matrix, which is based on the epipolar constraint given by $x_i^T F x_i = 0$. F is 3×3 matrix, which we get by solving a homogeneous linear system $Ax = 0$ with nine unknowns applying the Singular Value Decomposition (SVD). We do so by taking eight random points from the pool of matches and estimating a rough F matrix, computing the epipolar constraint, and comparing it with a threshold value. We then estimate the final F matrix using the inliers from RANSAC. We utilize the inliers from cv2 for a more robust result but our implementation also works effectively. Due to noise in the matches, the estimated F matrix can be of rank three i.e. $\sigma_9 \neq 0$. So, to enforce the rank two constraint, the last singular value of the estimated F is set to zero. This also results in all the epipolar lines intersecting at a single point known as the epipole. The epipolar lines for a pair of images are shown in Fig. 2. The epipole is the other camera position in the first image frame; hence, the point is not visible in the image.

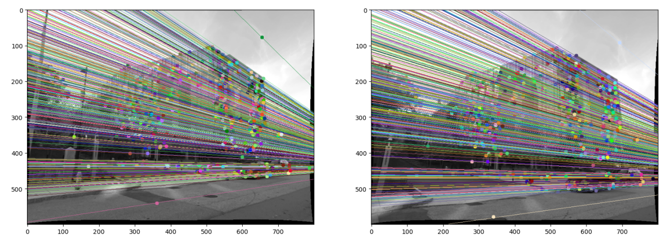


Fig. 2. Epipolar lines for a pair of images.

The final F matrix we got is shown below:

$$F_{final} = \begin{bmatrix} -1.607e-07 & -3.011e-05 & 1.343e-02 \\ 3.266e-05 & 3.202e-06 & -3.486e-02 \\ -1.518e-02 & 3.304e-02 & 1.000e+00 \end{bmatrix}$$

C. Estimate Essential Matrix from Fundamental Matrix

The essential matrix is another 3×3 matrix, but with some additional properties that relate the corresponding points, assuming that the cameras obey the pinhole model. The matrix is given by $E = K^T F K$, where K is the camera intrinsic matrix. E is reconstructed with $(1, 1, 0)$ singular values due to the noise in K given by $E = USV^T$, where S is a diagonal matrix with the given singular values. The estimated Essential matrix is given by:

$$E_{final} = \begin{bmatrix} -9.666e-05 & -5.827e-01 & 1.395e-01 \\ 6.355e-01 & 5.369e-02 & -7.511e-01 \\ -1.880e-01 & 7.960e-01 & 2.791e-02 \end{bmatrix}$$

D. Estimate Camera Pose from Essential Matrix

The camera pose consists of 6 DOF, Rotation (Roll, Pitch, Yaw), and Translation (X, Y, Z) of the camera with respect to the world. The camera pose is estimated from $P = KR[I_{3\times3} - C]$. These four pose configurations can be computed from E matrix where $E = UDV^T$, and $W = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$. The four configurations are $C_1 = U(:, 3)$ and $R_1 = UWV^T$, $C_2 = -U(:, 3)$ and $R_2 = UWV^T$, $C_3 = U(:, 3)$ and $R_3 = UW^TV^T$, $C_4 = -U(:, 3)$ and $R_4 = UW^TV^T$. Also if $det(R) = -1$, the camera pose is corrected i.e $C = -C$ and $R = -R$.

E. Triangulation Check for Cheirality Condition

We found four camera poses by decomposing the Essential matrix. Only one of them is accurate. We triangulate the 3D points given any two camera poses and the matched features and plot all the 3D points as shown in 3. We disambiguate the camera poses by checking the cheirality constraint, which determines if the point is behind the camera. We triangulate the 3D points using linear least squares to check the sign of the depth Z in the camera coordinate system w.r.t. camera center. A 3D point X is in front of the camera $r_3(X - C) > 0$ where r_3 is the third row of the rotation matrix (z-axis of the camera). The pose with the most number of points satisfying the constraint is the true camera pose. Now that we have the camera pose configurations and their linear triangulated points.

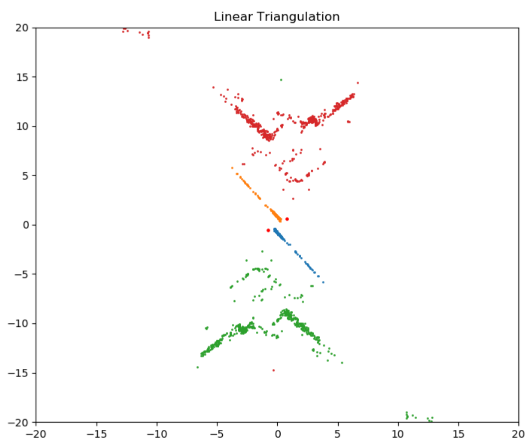


Fig. 3. Triangulated 3D points from 2 camera poses.

To better estimate the triangulated points, we solve a nonlinear minimization problem. The located triangulated points, X, are considered as the initial guess for the problem where we try to minimize the reprojection error. We solve using nonlinear optimization function $scipy.optimize.least\_squares()$. The results are compared with the linear triangulation, which is shown in Fig. 4.

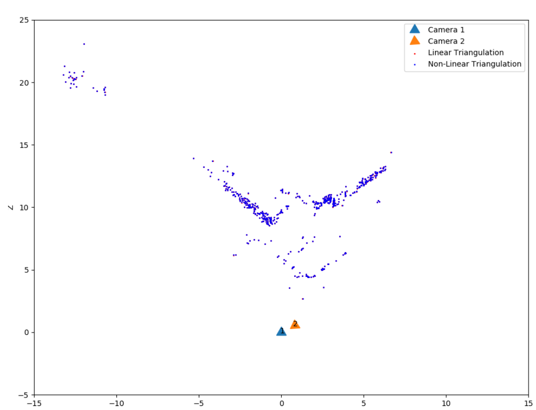


Fig. 4. Linear and nonlinear triangulated 3D points from 2 camera poses.

Fig. 5. Reprojected points after linear triangulation.

Fig. 6. Reprojected points after nonlinear triangulation.

F. Perspective-n-points

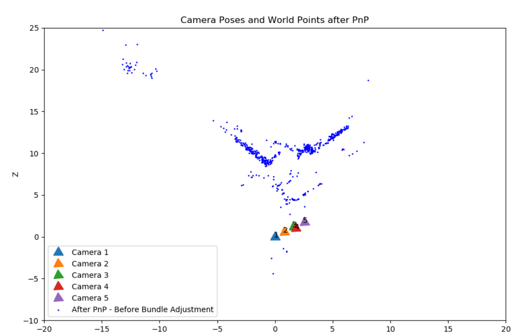


Fig. 7. Camera poses after nonlinear PnP.

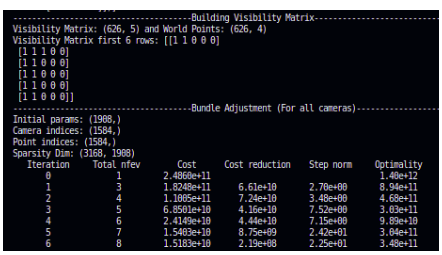
With the world points from nonlinear triangulation, intrinsic parameters, and common points, the camera poses are estimated using LinearPnP. Here, we again get a linear equation and solve it using SVD to get the projection matrix. The rotation and camera poses are extracted from the same matrix. The estimation may not be accurate due to underlying outliers that were removed using PnPRANSAC. The reprojection error is minimized using nonlinear PnP using least squares. The camera poses and all world points are shown in Fig. 7.

The reprojection errors are listed in the following table.

Error	
Linear reprojection	6.8731
Nonlinear reprojection	6.840
Linear PnP (1,3)	777.91
Nonlinear PnP (1,3)	726.17
Linear PnP (1,4)	19.44
Nonlinear PnP (1,4)	7.486
Linear PnP (1,5)	44.71
Nonlinear PnP (1,5)	3.613

G. *Visibility matrix and bundle adjustment*

We create a visibility matrix for the given number of cameras and total world points. The function initializes the matrix with zeros and iterates over each world point and camera to mark the visibility of the world point in each camera. Fig. 8 shows the Visibility matrix size and how each column shows if a world point is visible in each of the camera view. Next, we use bundle adjustment to reduce the projection error simultaneously for all the 5 images using this visibility matrix. Bundle adjustment takes input parameters related to camera poses, 3D points, visibility, and intrinsic matrix and uses least squares optimization to refine the camera poses and 3D points. The function returns the optimized camera poses and 3D points. This is shown in 9, which shows bundle adjustment for all the camera views.



```
--------------------------------------Building Visibility Matrix------------------------
Visibility Matrix: (626, 5) and World Points: (626, 4)
Visibility Matrix first 6 rows: [[1 1 0 0 0]
 [1 1 1 0 0]
 [1 1 0 0 0]
 [1 1 0 0 0]
 [1 1 0 0 0]
 [1 1 0 0 0]]
--------------------------------------Bundle Adjustment (For all cameras)---------------
Initial params: (1908,)
Camera indices: (1584,)
Point indices: (1584,)
Sparsity Dim: (3168, 1908)
   Iteration     Total nfev        Cost      Cost reduction    Step norm     Optimality
       0              1         2.4860e+11                                    1.40e+12
       1              3         1.8248e+11      6.61e+10       2.70e+00       8.94e+11
       2              4         1.1005e+11      7.24e+10       3.48e+00       4.68e+11
       3              5         6.8501e+10      4.16e+10       7.52e+00       3.03e+11
       4              6         2.4149e+10      4.44e+10       7.15e+00       9.89e+10
       5              7         1.5403e+10      8.75e+09       2.42e+01       3.04e+11
       6              8         1.5183e+10      2.19e+08       2.25e+01       3.48e+11
```

Fig. 8. Visibility matrix size and Sample entries.

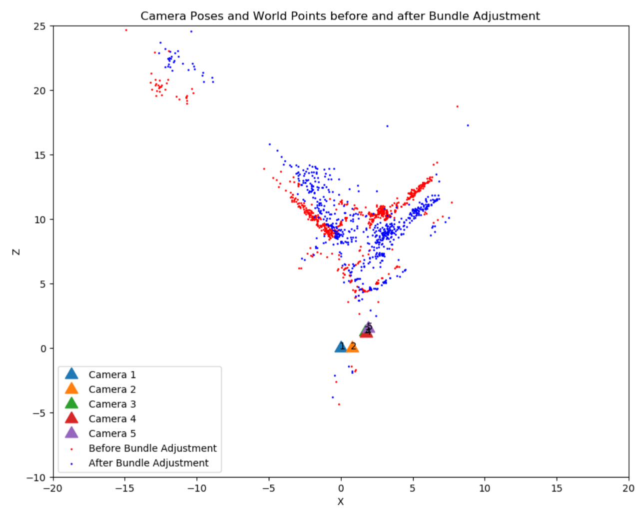


Fig. 9. Camera poses and world points after bundle adjustment.