Project 3
Buildings built in minutes - An SfM Approach

Pavan Mantripragada
James Clark School of Engineering
University of Maryland, College Park
Email: mppavan@umd.edu

Using 1 late day

Vishaal Kanna Sivakumar
James Clark School of Engineering
University of Maryland, College Park
Email: vishaal@umd.edu

Using 1 late day

I. Introduction

The aim of the project is to reconstruct a 3D scene using 2D images of the same scene taken from different perspectives. In this implementation we used 6 images of a building and text files describing correspondences between them, as described in Sec. 5 of [1], to reconstruct 3D world coordinates of the features and relative camera poses.

II. Estimating Fundamental Matrix

Given corresponding 2D image points between two images, a Fundamental matrix 'F' for the image pair is estimated and the inlier feature points are identified using RANSAC. First, from the given set of correspondences 8 random points are chosen and a F matrix is computed using eight-point algorithm [2]. For the computed F we identify inlier point-pairs by the condition $|\mathbf{x}_i'^{\mathbf{T}}\mathbf{F}\mathbf{x}_i| < \tau$ where, τ is appropriately chosen threshold. This process is repeated for a certain no. of iterations and the F matrix which has highest no. of inliers is chosen. In all further steps only the final inliers are used to compute the structure of the scene.

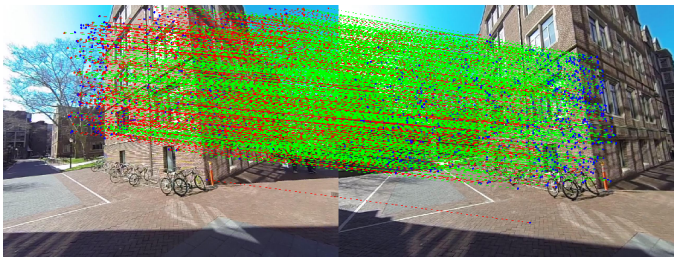

Fig. 1: Outlier rejection from given correspondences after RANSAC. The inliers are represented as green lines and the outliers are in red.

III. Essential Matrix From Fundamental Matrix

The essential matrix is computed for every image pair using the above set of F matrices and the given camera calibration matrix K

$$E = K^T F K$$

After the above computation, we perform singular value decomposition of E and enforce the singular values of E to be [1, 1, 0].

IV. Estimate Camera Pose From Essential Matrix

Since, we assume that the first camera is located at the origin of world and looking towards positive Z direction. We need to register all other cameras w.r.t the first camera frame. This is done by decomposing our E matrix into 4 sets of 3D euclidean transforms as mentioned in Sec. 3.4 [1].

$$C_1 = U(:,3) \quad R_1 = UWV^T$$

$$C_2 = -U(:,3) \quad R_2 = UWV^T$$

$$C_3 = U(:,3) \quad R_3 = UW^TV^T$$

$$C_4 = -U(:,3) \quad R_4 = UW^TV^T$$

Where, $\mathbf{E} = UDV^T$ and $W = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

The ambiguity in this pose is resolved by checking the depth positivity condition in Sec. V-B.

V. Triangulation

To obtain 3D world points of the correspondences we need to triangulate them both algebraically and geometrically. This is performed in sections V-A, V-C along with disambiguation of camera poses.

A. Linear Triangulation

Given two camera poses $[R_1, t_1]$, $[R_2, t_2]$ and camera matrix K we can find their corresponding projection matrices by

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

With the help of these two matrices we can project our unknown 3D world points into the corresponding 2D image points (both in homogeneous form).

$$x = P_1 X \quad x' = P_2 X$$

rearranging the above equations will give us a set 4 homogeneous equation which can be solved for obtaining 3D world points in homogeneous form.

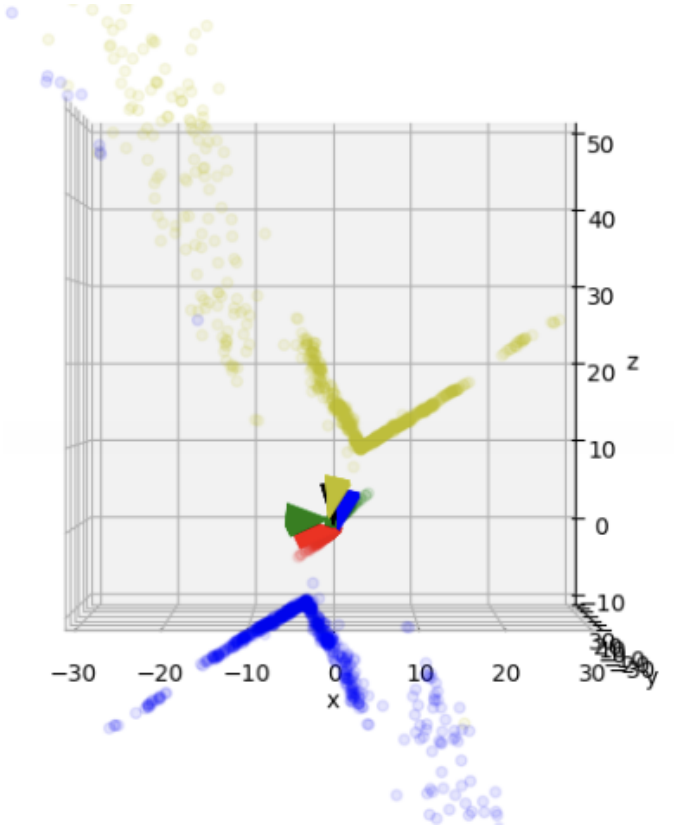

Fig. 2: 3D points obtained from triangulation

B. Cheirality Check

During the registration of second camera frame w.r.t first camera we had obtained a set of 4 poses out of which only one represents the true camera pose. In order to identify that camera pose, we triangulate correspondences using all four poses and choose the pose in which maximum no. of 3D points are in front of both the cameras i.e., $r_3(X - C) > 0$ and $x_3 > 0$.

C. Non-Linear Triangulation

The 3D points obtained from the linear triangulation, along with the camera poses after disambiguation are used to refine the coordinates of the points by minimizing the reprojection error. The reprojection error is computed by measuring error between measurement and projected 3D point as given:

$$\sum_{j=1,2} \left(u^j - \frac{P_1^{jT} \widetilde{X}}{P_3^{jT} X} \right)^2 + \left(v^j - \frac{P_2^{jT} \widetilde{X}}{P_3^{jT} X} \right)^2 \tag{1}$$

The initial guess of the solution is obtained from linear triangulation and the optimization is performed using scipy.optimize.least_squares with the above cost function.

VI. Perspective-n-Points

A. PnP RANSAC

From a set of 3D points in the world, the corresponding 2D projections in an image and the intrinsic parameter of the camera, the 6 DOF camera pose can be estimated using linear least squares. In order reject outliers from the set of 3D-2D correspondences, RANSAC is used over Linear PnP to get the best set of camera poses.

B. Linear Camera Pose Estimation

Given 2D-3D correspondences, and the intrinsic paramter K, we estimate the camera pose using linear least squares. 2D points are normalized by the intrinsic parameter to isolate camera parameters. To enforce orthogonality of the rotation matrix, the rotation matrix is corrected by $R = UV^T$ where $R = UDV^T$ and if the corrected rotation has 1 determinant, we assign $R = R$.

C. Non-Linear PnP

Camera poses from Linear PnP are refined such that it minimizes reprojection error as done in non-linear triangulation.

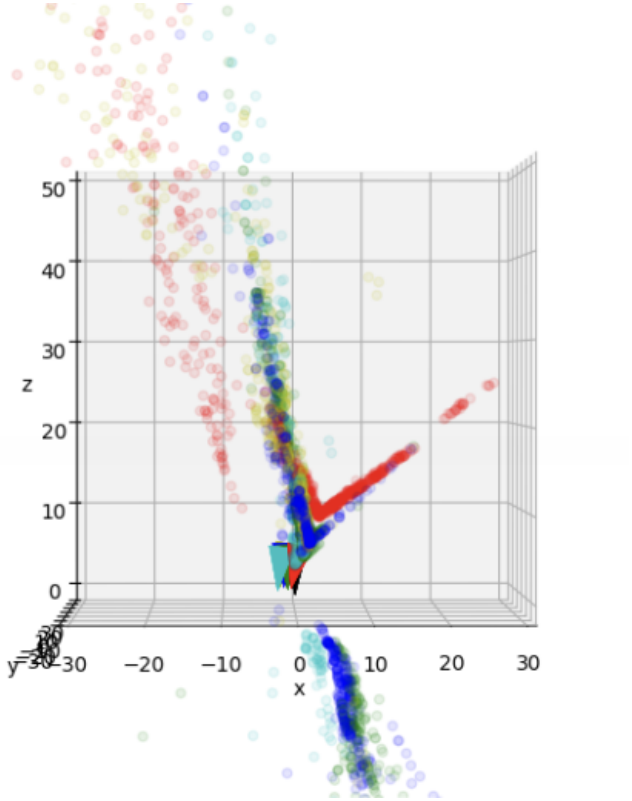

Fig. 3: Camera poses and the 3D points after non-linear PnP

VII. Bundle Adjustment

A. Visibility Matrix

We have created a binary matrix V of shape $I \times J$ where I is no. of features in all images that are being adjusted and J is no. of images. This is created with the help of mask matrices, which we maintain to keep track of inliers in each image and their corresponding world coordinates.

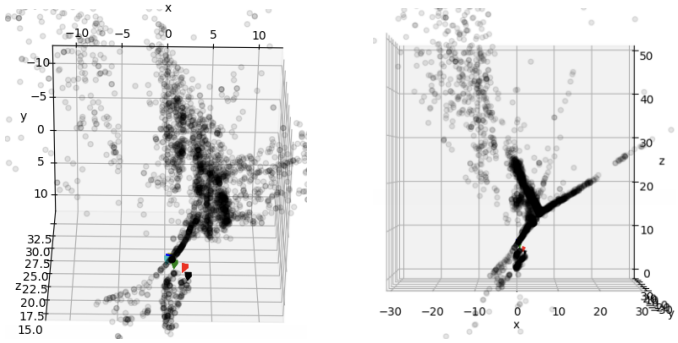

Fig. 4: Camera poses and the 3D points after bundle adjustment.

B. Bundle Adjustment

After registering every new camera, we refine the camera poses and 3D world points simultaneously by minimising the re-projection error. This error term is highly non-linear. So, we have used scipy.least_squares optimizer to minimize the following objective function.

$$\sum_{i=1}^{I}\sum_{j=1}^{J} V_{ij}\left(\left(u^j - \frac{P_1^{jT}\tilde{X}}{P_3^{jT}\tilde{X}}\right)^2 + \left(v^j - \frac{P_2^{jT}\tilde{X}}{P_3^{jT}\tilde{X}}\right)^2\right)$$

VIII. Results

We tabulate the reprojection errors we obtain after each step in the SFM pipeline. The error is minimized after bundle adjustment for each pair of image. Our implementation of PnP does not yield good results in comparison to camera poses obtained from the Essential matrix between the first two images. We believe that the results would improve on tuning the parameters used in PnP RANSAC. For visualization, we have used the code from [3]

Re-Projection Errors

Image	Non-linear PnP/triangulation	Bundle Adjustment
1	10.2	2.3
2	12.2	4.1
3	253.4	101.9
4	126.6	128.9
5	64.2	20.3
6	162.2	70.3

References

[1] https://cmsc733.github.io/2022/proj/p3/
[2] https://en.wikipedia.org/wiki/Eight-point_algorithm
[3] https://github.com/demul/extrinsic2pyramid