From drone footage video to 3D reconstruction via point clouds

Yixian Li
Stanford Department of Computer Science
353 Jane Stanford Way, Stanford, CA
S031820@stanford.edu

Abstract

The final report regards the project of constructing 3D point cloud models from the drone footage video, or an uncalibrated image sequence, where a model of interest is filmed from different angles. The report briefly introduces the overall frame of the problem, the plan to approach the problem, the dataset, and the expected results. Next, a comprehensive review of background and previous work is provided. Then, the technical approach is introduced that involves feature detecting, matching, fundamental matrix approximation, structure from motion, and camera calibration. Afterwards, the experiments from the above approach are presented via a sparse 3D point cloud model, with a comparison to a pre-built pipeline of multi-image 3D sparse reconstruction. Finally, a conclusion is drawn from the approach and experiments, with further work presented as the same time.

1. Introduction

The use of consumer drones is growing, and so is consumer cameras. We have seen great footage of famous sites taken from on-drone consumer camera. In this report, we want to take a step further once we obtain these great videos. This involves the use of modern technique of computer vision to reconstruct the scene we are interested. This project strives to provide a pipeline for people to rebuild a 3D point cloud model from their own footage video.

The next following sections introduces the problem statement, the dataset that we use, the software package, and the expected results and evaluations.

1.1. Problem Statement

We are given a short video clip taken from an on-drone consumer camera, where the drone rotates and translates around the subject we are interested to reconstruct. The expected outcome from the problem is a 3D sparse point cloud model of the interested object reconstructed via key points from the scene.

1.2. Dataset

The dataset of this project is a video clip (29 seconds, 50 frames/second, 1920p * 1080p) of the multi-views of a church building taken from a drone camera [1]. From the video we extracted the first 20 frames by retrieving one frame every one second. See fig.1, fig.2 for illustration.

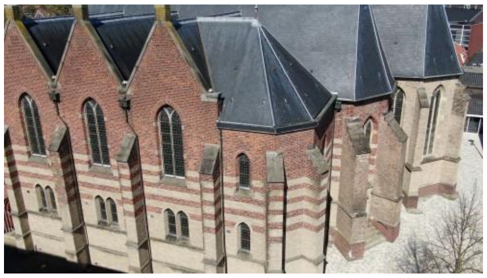


Fig.1 Frame number 600 from the original video.

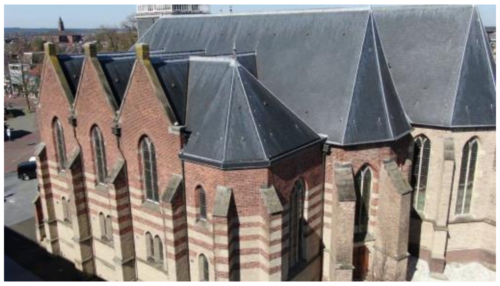


Fig.2 Frame number 800 from the original video.

For this project, we specifically picked nine from all frames that have been retrieved. The frames picked have an appropriately large baseline, so the reconstruction would not be too inaccurate when performing the triangulation (see section 3.3).

1.3. Expected Results and Evaluations

The expected results for this project would be presented via sparse 3D point cloud. Without an accurate ground truth model, we want to evaluate our model qualitatively, which typically involves three quality metrics:

1. There is a clear indication of a cluster of points that correspond to the front surface of the landmark we want to reconstruct.
2. There is a clear indication of a segment of points that correspond to the edges and corners of the building.
3. There are few points that does not correspond to any part of the building.

Evaluations are compared between an uncalibrated multi-view reconstruction case and a calibrated multi-view reconstruction case. Then, a well-developed 3D realization and visualization software, VisualSFM developed by C. Wu [2,3], is utilized to compare our results to the result from a well-built pipeline. The package detail is presented in section 1.4.

1.4. Available Packages

The environment we are using is the latest Python 3.8 version. Meanwhile, we will be utilizing OpenCV-python for feature selection and matching portion our 3d visualizing process [4].

Besides, we will be using VisualSFM, a developed 3d point cloud visualization software [2,3]. The purpose of using the software is to verify the completeness and correctness of our own solution.

2. Review of Previous Work

There are various sources we take advantage of in building our own pipeline of 3D reconstruction. These sources are presented in the sections below.

2.1. Pollefeys et al.'s Paper (Pipeline)

The pipeline of the project is followed mainly from the paper written by Pollefeys et al. [5]. The paper suggests a complete pipeline from an uncalibrated image sequence to a 3D reconstruction model. Our project follows the procedure described in section 3 and 4 in the paper, which involves the building of the projective reconstruction and the metric reconstruction. The paper provides great insight into various aspects of the project as well. This includes the use of RANSAC algorithm to estimate the fundamental matrix [6], the finding of projection matrix, and camera-self calibration process.

2.2. Various Resources (References)

Other resources are for reference use. This includes some details that are presented in Pollefey et al.'s paper but are not explicitly explained. This typically includes work from various journal papers. We will talk through those in the following technical approach sections.

3. Technical Approach

In this section, we want to elaborate on the technical approach that is conducted or ready to be conducted to effectively solve our problem. The problem is broken down into the following steps.

3.1. Feature Detection and Matching

The first step of the reconstruction process is feature matching. In this project, we used the SIFT (scaled-invariant feature transform) algorithm introduced by Lawn [7]. The SIFT algorithm is useful in detecting edge and blob features between images. An example of feature detection via SIFT is shown in Fig.4.

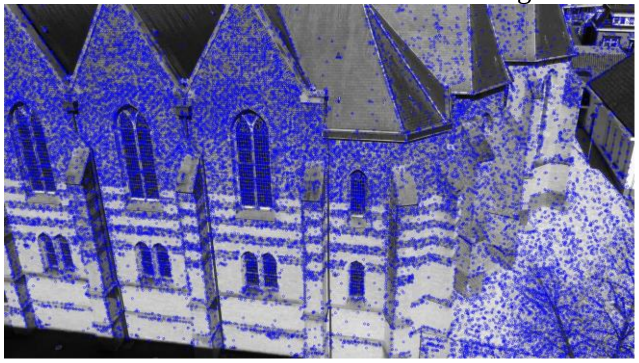


Fig 4. SIFT feature detection in the baseline image.

After finding the features, we apply the Brute-force matching with applied ratio method to find the good (confident) correspondences [7,8]. Fig.3 shows the SIFT matching between the two base frames of the same scene we chose.

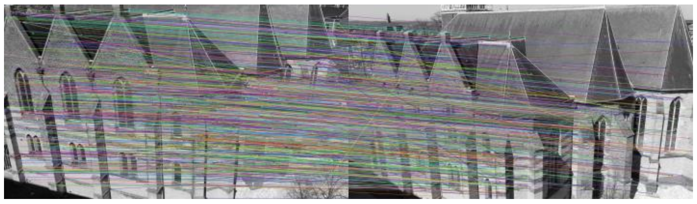


Fig.3 Feature matching of two frames we picked (Fig.1 and Fig.2) by brute-force SIFT matching.

From the SIFT algorithm, we obtained a total of 1092 “confident” correspondences. However, because our interested model subjects to noises and several repetitive structures, the correspondences need to be further filtered to be more accurate. This process is introduced in section 3.2.

3.2. Estimating Fundamental Matrix

This step of the reconstruction process is to find the fundamental matrix relating the two images. The traditional eight-point algorithm would be used in conjunction with the RANSAC algorithm [6]. An example of this algorithm adapted from Pollefeys et al.’s paper is as followed [5]:

```
• Repeat numSample times:
   - randomly select 8 matches from sample
   - compute F using the eight-point algorithm
   - count the number of samples (inliers) that satisfies a distance threshold: p2.T* F * p1 <= threshold.
  - update F if this iteration yields the most inliers.
• Check if numInlier >= 95%, if not, increase threshold value, and run the algorithm again.
• Use all the inliers to recalculate F.
```

Table 1. RANSAC algorithm used to estimate the fundamental matrix, adapted from Pollefey et al.’s paper.

The output of the algorithm not only gives a more accurate fundamental matrix estimation but also gives reliable inliers that we would use for triangulation in the further steps.

3.3. Structure from Motion

This section introduces SFM, from which we determine the projective matrix of a set of cameras. The procedure is described in section 3.3.1 and 3.3.2 below.

3.3.1 SFM of a pair of images

From section 3.2, we have determined a 3*3 fundamental matrix between two cameras. Denote this matrix as F12. We will calculate the epipole of the second camera e’ via SVD (singular value decomposition) of F12.T (transpose of F12). The equation is given by [9]:

$$F^T{}_{12} \cdot e' = 0$$

Since F12 is rank-deficient, we could just take the last column from v, and normalize it, set this value to be e’.

From in-class lecture, we understand that given a pair of cameras. If we want to obtain their projection matrices P1 and P2 to triangulate points in 3D, we want to set the first camera matrix to be conic [5,9].

$$P_1 = [I^{3*3} \quad 0^{3*1}]$$

The other projection matrix is given as an expression of e’ we just calculated [5]:

$$P_2 = [[e'_x]F_{12} \quad e']$$

where e’x is the 3*3 cross matrix formulated from e’.

With the two projection matrices, we could triangulate any matched correspondence to a point in 3D (X) by solving the equations below via SVD [9]:

$$x_1 = P_1 * X$$
$$x_2 = P_2 * X$$

where x1, x2 are the homogeneous 2d corresponding points in each image.

3.3.2 Adding views - SFM of multiple images

At this point, we have already obtained a 3D point cluster with two views. The next step is to apply more frames into our model to adjust our existed 3D points. The Pollefey et al.’s paper provides us with some insight on this [5].

Once a new view is applied, we applied the method in section 3.1 and 3.2 between the initial view and the new view. The new correspondences can be further categorized into:

1. Correspondences that has already been established and converted into a 3D point.

2. New correspondences that has not yet been established and converted into a 3D point.

* Before doing triangulation in the following steps, we calculate the new projection matrix based on the initial projection matrix from the new fundamental matrix.

3.3.2.1 Point adjustment

If correspondences fall into category 1, we will re-estimate the 3D point using the new projection matrix. We record the new 3D coordinate value.

After finding all 3D coordinate value, we will use RANSAC algorithm again to differentiate inliers from outliers by setting up a threshold of a unit ball distance. We then take the average of those inliers and relocate the new 3D points.

3.3.2.2 Point addition

If correspondences fall into category 2, we will establish new points based on section 3.3.1. Once new points have been determined, record the new 3D coordinates and their corresponding match.

3.4. Camera Self-calibration

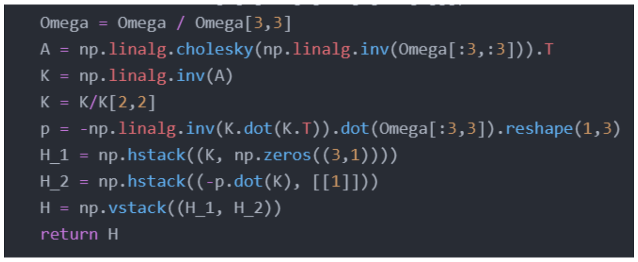
The camera self-calibration is used to restrict the ambiguity occurred in our projective model. Our method follows the method described by Pollefey et al. [5]. The process involves an initial calibration via a linear method and a refined calibration via a non-linear least-square minimization problem. Our algorithm implements the initial calibration portion of the camera calibration.

Assuming square pixels, same focal length along x and y, and centered principal point (where we enforce it by centering our data), given three or more projection matrices, we could form a system of equation described as [5]:

$$P_k^{(1)}\Omega P_k^{(1)^T} = P_k^{(2)}\Omega P_k^{(2)^T}$$
$$P_k^{(1)}\Omega P_k^{(2)^T} = 0$$
$$P_k^{(2)}\Omega P_k^{(3)^T} = 0$$
$$P_k^{(2)}\Omega P_k^{(3)^T} = 0$$

Where k >= 3, Ω is 4*4 symmetric matrix with 10 unknowns.

After Ω is solved via SVD, we could decompose it using Cholesky decomposition [10]. As suggested in the book "Multiple Views Geometry in Computer Vision" by R. Hartley and A. Zisserman [11], the rest of the algorithm could be formulated as:



```
    Omega = Omega / Omega[3,3]
    A = np.linalg.cholesky(np.linalg.inv(Omega[:3,:3])).T
    K = np.linalg.inv(A)
    K = K/K[2,2]
    p = -np.linalg.inv(K.dot(K.T)).dot(Omega[:3,3]).reshape(1,3)
    H_1 = np.hstack((K, np.zeros((3,1))))
    H_2 = np.hstack((-p.dot(K), [[1]]))
    H = np.vstack((H_1, H_2))
    return H
```

Fig 4. Algorithm that calculates H from Ω.

We then found H and could correct our 3D points by the in inverse of H:

$$P_{new} = H^{-1}P$$

3.5. 3D Point Cloud Visualization

The visualization is done by making the 3D scatter plot via matplotlib. A more detailed and comprehensive analysis will be drawn from the comparison of the output from the prebuilt pipeline model out from VisualSFM [2,3]. Once the sparse point cloud model has been developed, we will analyze the different aspects of the model presented in 1.3. From there, the final evaluation of the model could be made.

4. Experiment

This section presents the results of the project, which includes the plots of the 3D sparse reconstruction model from both the uncalibrated views and calibrated views. Also, we will also present the resulting plot from VisualSFM to further evaluate our calibrated views case. The details of those results are given in the following sections:

4.1. Uncalibrated views

The result of the uncalibrated view case is shown in the fig. 5.

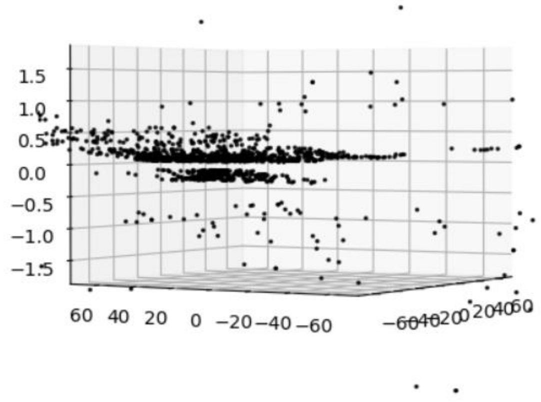


fig. 5 Uncalibrated Three Views by our own model.

The result is as expected for the uncalibrated case (run before section 3.4). From the graph, we could see points are clustered at nearly the center of the z-axis, with few other points scattered around. According to our metric, we believe that most of the points stayed at the same plane, which indicates the front surface of the building. However, we do not see clear indication of corners and edges under such ambiguity.

4.2. Calibrated views (Own model)

The result of the calibrated view case is shown in the fig. 6.

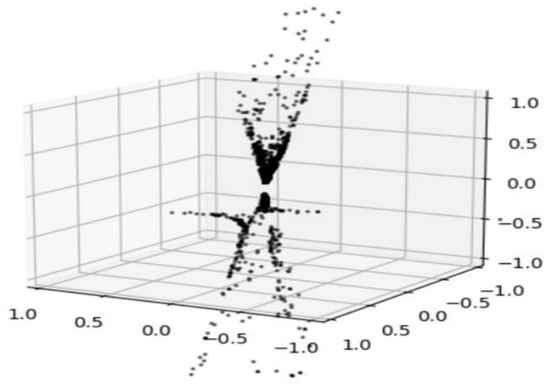


fig. 6 Calibrated Three Views by our own model.

The result is not as expected for the calibrated case. From the point cloud we obtained, we could clearly visualize that the model did not correspond to our scene. There are no clear indication of the front plane, corners, or edges. The shape of the resulting point clouds looks symmetric to the z=0 plane and is cone-like. A comparison to the outcome run by VisualSFM is presented in the following section [2,3].

4.3. Calibrated views (VisualSFM [2,3])

The result of the output from running VisualSFM software is shown in fig 6 and fig 7.

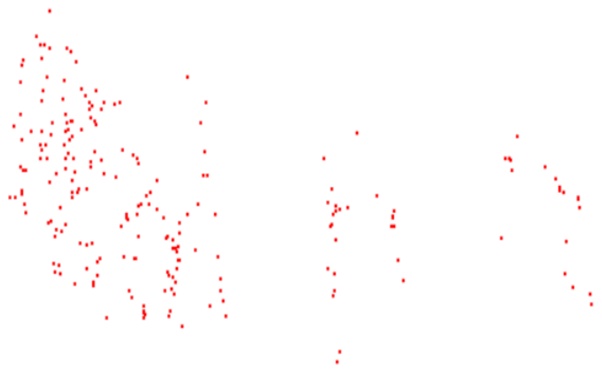


fig. 6 Calibrated Three Views by VisualSFM model.

As we could see in fig.6, the 3d sparse reconstruction from same three views differ a lot from the reconstruction we performed. It is shown clearly that most points are clustered on the left-hand side of the plot, which corresponds to the front surface. The few points on the right corresponds to the side surface. There is no clear indication of edges and corners given the reconstruction from three views.

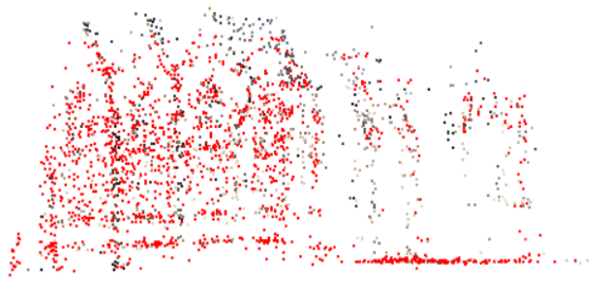


Fig.7 Calibrated Nine Views by VisualSFM model.

When more views are added, the sparse reconstruction becomes more and more representative of the scene. From this point cloud model, we visualized that not only the front plane and side plane has been clearly segmented, but we could also see there are clear indications of corners and edges that characterize the shape of the building we are interested.

As our construction does not successfully reconstruct a 3D sparse point cloud model, a further investigation the method we chose and the code we developed should be launched. The further steps we will take are presented in section 5 of the report.

5. Conclusion

In this section, we reflect on some of the issues that arise in the project that does not contribute a successful reconstruction. Given the problem, we will suggest some improvements that could be further developed into future work that we will work upon.

5.1. Problem and Improvement

There are multiple areas that could be improved. First, there might be too many outliers from SIFT and brute-force matching, which leads to the wrong estimate of fundamental matrix and usable inliers, given we are using the RANSAC algorithm. An improvement to this is to

Second, the calculation of homograph H is not good enough in the camera self-calibration process. This leads to further investigation into the codes developed and the method we adapted. This typically includes the additional implementation of the non-linear optimization process to adjust our H matrix.

Third, the point correspondences are not sufficient. We need to use more image pairs to get more correspondences, and thus filtering out false positive from reconstructed model.

5.2. Future Work

Future work should be focusing on fixing the pre-exist issue described in section 5.1 first. From there, we could adjust the 3D points further by implementing bundle adjustment algorithm after the self-calibration process. After determining a reasonable sparse reconstruction model, we could work on creating a dense 3d reconstruction model from more matches from the images by searching more correspondences through epipolar lines.

References

[1] P. Krabbe. *Drone Footage of a Church Building Exterior*. Available from: https://www.pexels.com/video/drone-footage-of-a-church-building-exterior-4072222/

[2] Changchang Wu, "VisualSFM: A Visual Structure from Motion System", http://ccwu.me/vsfm/, 2011

[3] Changchang Wu, Sameer Agarwal, Brian Curless, and Steven M. Seitz, "Multicore Bundle Adjustment", CVPR 2011

[4] OpenCV_python interface. Available from https://pypi.org/project/opencv-python/

[5] M. Pollefeys et al. “Metric 3D Surface Reconstruction from uncalibrated Image Sequences”. K.U.Leuven, EXAT-PSI, Kard. Mercierlaan 94, B-3001 Heverlee, Belgium. Nov. 15, 1998

[6] Martin A. Fischler & Robert C. Bolles (June 1981). "Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography". Comm. ACM. 24 (6): 381–395. doi:10.1145/358669.358692. S2CID 972888

[7] Lowe, David G. (1999). "Object recognition from local scale-invariant features". *Proceedings of the International Conference on Computer Vision*. 2. pp. 1150–1157. doi:10.1109/ICCV.1999.790410.

[8] Open-Source Computer Vision. *Feature Matching*. Available from: https://docs.opencv.org/master/dc/dc3/tutorial_py_matcher.html

[9] S. Savarese and J. Bohg. Multi-view Geometry. Available from: https://web.stanford.edu/class/cs231a/lectures/lecture7_SFM.pdf

[10] Dereniowski, Dariusz; Kubale, Marek (2004). "Cholesky Factorization of Matrices in Parallel and Ranking of Graphs". *5th International Conference on Parallel Processing and Applied Mathematics*. Lecture Notes on Computer Science. 3019. Springer-Verlag. pp. 985–992. doi:10.1007/978-3-540-24669-5_127. ISBN 978-3-540-21946-0. Archived from the original on 2011-07-16.

[11] Hartley, R., and Zisserman, A. (2003). Multiple View Geometry in Computer Vision: Vol. 2nd ed. Cambridge University Press.

Link to Github

https://github.com/Yixian-work/cs231a-project

(Access granted to the project mentor)