Evaluation of an OpenCV Implementation of Structure from Motion on Open Source Data

Ali Alouache[1], Qinghe Wu[2]

[1] Centre des Techniques Spatiales, Agence Spatiale Algérienne
Arzew, Algérie
[2] Automation School, Beijing Institute of Technology
Beijing, China
alouache15@yahoo.com, qinghew@bit.edu.cn

Abstract. Structure from Motion (SfM) is the technology of recovering the 3D model from multiple 2D views. It has received a great attention from computer vision community to construct large scale 3D models whose spatial resolution of comparable quality to LiDAR. Nowadays, aerial SfM is becoming even more important due to the rapid growth of low cost commercial UAV and small satellite market. This paper presents the evaluation of an OpenCV implementation of incremental SfM approach on open source data. The results of 3D construction obtained by OpenCV are compared to Visual SfM program in terms of precision, density of features and spatial resolution.

Keywords: 3D Reconstruction, Structure from Motion, OpenCV.

1 Introduction

Light detection and ranging (LiDAR) is very useful 3D data source that is commonly adopted to construct 3D models of the world [1], [2], [3], [4]. An increasing number of applications in the field of geography and environmental science require 3D model as input source such as remote sensing [5], [6], [7], topographic mapping [8], [9], and geographic information systems (GIS) [10], [11]. LiDAR sends out pulses of laser light and measures the exact time it takes for these pulses to return as they bounce from the ground. Then through measuring the timing and intensity of the returning pulses, it can provide readings of the terrain and of points on the ground.

The 3D map generated from LiDAR provides elevation information, which can be colorized based on either elevation or intensity to aid interpretation. The major advantages of LiDAR are the accuracy in terms of spatial resolution, and its capability of capturing 3D data in the day as well as in the night time. In addition, LiDAR performs well when it used to capture 3D data over terrains that contain power lines and dense vegetation [12], [13]. However, a visual inspection of the results shows that the 3D

point cloud constructed by LiDAR in urban areas is difficult to interpret, and this may be explained due to the absence of the color in the point cloud which has a strong relevance to object recognition. Moreover, LiDAR is high cost because in addition to the laser sensor that captures 3D data, some other sensors are usually required such as high precision satellite positioning system (GNSS) and inertial measurement unit (IMU) in order to determine the position of LiDAR in space.

Recently, the Structure from Motion (SfM) technology has received a great attention from the researchers of computer vision community to construct large scale 3D models whose spatial resolution of comparable quality to Lidar [14], [15], [16], [17], [18], [19]. Nowadays, aerial SfM is becoming even more important due to the availability of low cost commercial UAV and small satellite market. However, the images should captured at the same height and contain at least 50% overlap in order to make SfM more suitable for 3D mapping. The output of SfM is a 3D model that contains not only elevation/height information, but also texture, shape, and color for every point on the map, which enables easier interpretation of the resulting 3D point cloud. Moreover, using low cost RGB cameras for capturing the images, then processing these images based on SfM will reduce the cost of 3D construction compared to Lidar [18].

Structure from Motion is similar to visual simultaneous localization and mapping (visual SLAM) in robotics [20]. The work [21] demonstrated that visual SLAM can be considered as a special case of SfM. However, both visual SLAM and SfM require computer vision algorithms for 3D mapping by using RGB cameras [22]. While LiDAR requires knowledge about how to exploit 3D data, and construct a surface model based on geometry post processing algorithms. Hence, the choice between LiDAR, visual SLAM or SfM depends on many factors such as the desired 3D output, the considered application, the available datasets, and the equipment to be used for 3D mapping.

There have been many different implementations of SfM pipelines in the literature. This paper searched for the most effective pipelines with publicly available source code that could allow customization of the pipeline itself. Examples of the available pipelines include the followings. Bundler [23] is a 3D reconstruction software that provides all of the executable version and source code. COLMAP [24] is a SfM and Multi-View Stereo (MVS) pipeline with a graphical and command-line interface. OpenMVG [25], is a SfM library well documented and accessible, including all the dependent libraries for simple installation. VisualSFM [26] is GUI application for 3D construction from a set of uncalibrated images based on SfM and MVS software. Theia [27] is a computer vision library aimed at providing efficient and reliable algorithms for SfM.

This paper presents the evaluation of an OpenCV implementation of incremental SfM approach [28] on open source data. Efficient algorithms are used in this implementation such as ORB with brute force for features detection and matching. The rejection of outliers based on RANSAC to obtain a robust estimation. Moreover, Google Ceres solver is embedded in this code to further optimize the 3D model by minimizing the reprojection errors given by Levenberg–Marquardt algorithm.

To the best of our knowledge this the only OpenCV implementation of incremental SfM that is available as open source code in the internet. However, it is a monocular SfM approach which considers that the images are captured by the same camera. Therefore, the contributions of this paper are two folds. First, the monocular SfM approach

of [28] is extended to multi view SfM which makes it possible to construct 3D point cloud using images taken from different cameras that is more advantageous than a monocular camera. Second, a comparison is made between the characteristics of the OpenCV implementation of SfM and Visual SfM program.

The remainder of this paper is structured as follows. Section 2 describes the algorithm of incremental SfM. Section 3 presents the OpenCV implementation and the experimental results. Section 4 presents the comparison made between the OpenCV and Visual SfM program. Section 5 presents the conclusions and the future works.

2 Algorithmic Description

There are two approaches of incremental SfM. The first approach involves performing iteratively the two view SfM algorithm over multiple 2D views. After computing the intrinsic and extrinsics parameters for the first two views. The next step would be to get the 3D points of the two views using the projection matrix. Upon getting the initial estimate of the 3D points, these points have to be triangulated. Triangulation is performed iteratively over every image pair in a similar fashion. However, one of the main drawbacks of this approach is that the 3D reconstruction is up to scale. Which means that the motion obtained between the two views is going to have an arbitrary unit of measurement, that is, it is not in centimeters or inches but simply a given unit of scale. Thus, the reconstructed cameras will be one unit of scale distance apart. This has a big implications when extending the two views SfM to multiple views, as each pair of cameras will have their own units of scale, rather than a common one.

The second approach is based on using the Perspective-n-Point (PnP) algorithm [29] for multi view reconstruction. PnP takes care about the scale issue that existed in the first approach. However, it is observed in some cases that PnP algorithm don't give a reasonable projection matrix. Therefore, PnP is used in conjuction with RANSAC algorithm which is more robust. After getting the projection matrices of the first image pair, a triangulation is performed to get the initial 3D points. Now a baseline structure is computed, the projection matrix of the next view is calculated by using the 3D points that correspond to the 2D points of the new frame which is in turn used as input to PnP RANSAC module. Then, the 3D reconstruction is performed by triangulation and the process is repeated iteratively to get the projection matrix of successive views. However, this approach involves bookkeeping which means one needs to keep a track of the 3D points reconstructed using the previous two frames that correspond to the 2D points in the new frame. For each point in the 3D model, a vector denoting the 2D points is stored, then features matching is used to get a matching pair.

The incremental SfM approach presented in this paper is based on the robust PnP RANSAC module. The flowchart of fig. 1 shows the architecture of the incremental SfM approach proposed in this paper. It mainly introduces an additional operation compared to the monocular approach of [28], that is extract EXIF data. This operation is needed only for initialization at the beginning of the 3D reconstruction, because after that all the parameters are refined based on bundle adjustment.

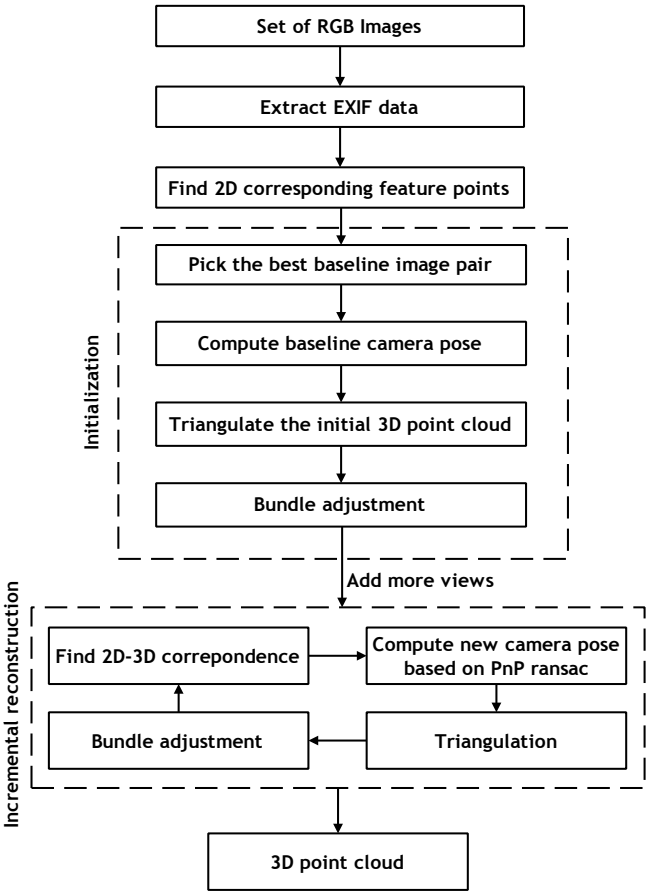


Fig. 1. Flowchart of the proposed incremental SfM approach

The flowchart of fig.1 cover the following four phases.

1. Extraction of EXIF data.
2. Finding 2D corresponding features points.
3. Initialization.
4. Incremental reconstruction.

A. Phase 1: extraction of EXIF data

EXIF information that is stored in the header of each image is parsed to get the camera focal length information for each input ith image. Then the focal length is used to construct the internal calibration matrix that is further optimized based on bundle adjustment.

B. Phase 2: finding 2D corresponding feature points

The purpose of this phase is to detect the 2D locations of the feature points in all the images, then estimating their corresponding points in all the possible image pairs using RANSAC. The algorithm is described below.

Algorithm of finding corresponding features
Input: set of n RGB images
For each i^{th} image
1. Apply the robust features detector.
End for
For each i^{th} and j^{th} image pair
2. Match the features points.
3. Compute the fundamental matrix (F_{ij}).
4. Estimate again the 2D features based on RANSAC.
End for
Output: fundamental matrices, corresponding feature points.

C. Phase 3: initialization

In this phase it is desired to pick the best baseline image pair in order to construct an initial 3D model. Indeed, if the baseline is small it is clear that it is not possible to determine the depth of the scene.

The accuracy of the initial baseline structure will determine the quality of the 3D model. Which means if the initialization is bad then the quality of the final 3D model will be low. Otherwise, if the initialization is accurate, then the overall 3D model will have a good accuracy. The initialization algorithm is given as follows.

Initialization Algorithm
Input: corresponding 2D feature points, focal lengths, baseline=false
For the image pair (Im_i, Im_j) with large number of matching features
1. Compute the Homography matrix H_{ij} using RANSAC
If the number of inliers is below the threshold *then* baseline=true
2. Set the projection matrix of the first camera is $M_1 = [I\ 0]$ where I denotes the identity matrix.
3. Compute the essential matrix E_{ij}.
4. Decompose E_{ij} based on SVD to get M_2.
5. Reconstruct the 3D points based on triangulation.
6. Refine the reconstruction of the 3D model based on bundle adjustment.
7. Get the colors of the 3D points.
8. Store the 3D point cloud #1.
Else
9. Increment i and j
10. *Check if the number of matches is enough then baseline is true*
11. Back to step 3
12. Otherwise, initialization is failed
End if
End for
Output: 3D point cloud #1, M_1, M_2.

D. Phase 4: incremental reconstruction

After getting the initial baseline structure, the following algorithm is carried out to construct the 3D model from multiple views based on the robust PnP RANSAC module.

Algorithm of incremental reconstruction

Input: M_1, M_2, 3D point cloud #1, 2D features points of the remaining images, focal lengths.

For each registered i^{th} image

1. Find 2D-3D correspondence points between the 2D corresponding feature points of the i^{th} image and the 3D point cloud #1.
2. Compute the projection matrix of the camera pose M_i based on PnP ransac.
3. Reconstruct the 3D points based on triangulation according to the initial baseline structure and the registered images.
4. Refine the reconstruction of the 3D model based on bundle adjustment.
5. Get the colors of the 3D points.

End For

6. Increment i
7. Back to step 1

Output: 3D point cloud, RGB data, camera pose.

3 Implementation and Experimental Results

A. Implementation based on OpenCV

OpenCV provides a range of feature detectors, descriptor extractors, and matchers. Hence ORB (Oriented Binary Robust Independent Elementary Features) is used to get the location of the feature points and their respective descriptors. ORB may be preferred over traditional 2D features such as the Speeded-Up Robust Features (SURF) or Scale Invariant Feature Transform (SIFT) because it is unencumbered with intellectual property and shown to be faster to detect, compute, and match.

After detecting feature points based on ORB, brute force binary matcher is used to get the matching, which simply matches two feature sets by comparing each feature in the first set to each feature in the second set.

Bundle adjustment is performed by Google Ceres solver [30] that is embedded in the code. Ceres Solver is an open source C++ library for modeling and solving large, complicated optimization problems. It can be used to solve Non-linear Least Squares problems with bounds constraints and general unconstrained optimization problems. It is a mature, feature rich, and performant library that has been used in production at Google since 2010.

Point cloud library (PCL) is used for real time visualization of the 3D reconstruction. After compiling the code, an executable file is generated in order to run SfM process. The point cloud that arises from the images is saved to PLY files, which can be opened in most 3D editing software.

B. Results of 3D construction

This section demonstrates the evaluation of the OpenCV implementaion of SfM compared with Visual SfM program.

Visual SfM saves on computation time by using GPU-accelerated SIFT in feature tracking to locate keypoints. In addition to its speed, Visual SfM has an excellent graphical user interface (GUI) that allows it to be operated easily. Moreover, it can be integrated with multi view stereo programs to produce dense 3D models. For more details about how to compile Visual SfM and use its GUI interface to generate sparse and dense 3D models the reader is referred to [31].

However, note that any other SfM software can be used as well for comparing the results of 3D reconstruction with Open CV.

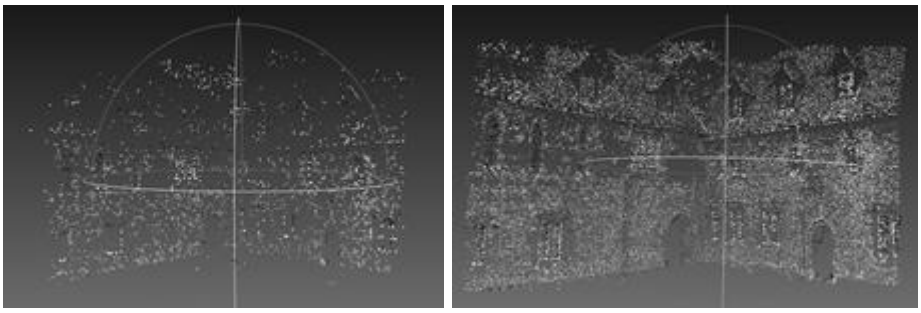
Consider as dataset the 3 images depicted in fig. 2(a) that are captured for façade of Merton College. The open source data is downloaded from the following link [32]. The ground truth model which includes 3D points, lines and cameras are shown in fig. 2(b). The software that is used for visualizing the (.wrl) model is view3dscene.

The results of 3D reconstruction based on Visual SfM and OpenCV are depicted in fig. 3. The software used for visualizing the 3D model is Meshlab.

The 3D model constructed based on Visual SfM shown in fig. 3(a) contains 2983 points, which is very sparse and leaves many holes on the surface. Whereas the 3D model constructed based on OpenCV shown in fig. 3(b) contains 24482 points, which is dense and most of the objects appear clearly on the surface of the model.

(a) (b)

Fig. 2. Image dataset and the ground truth 3D model



(a) (b)

Fig. 3. 3D construction based on Visual SfM and OpenCV

4 Comparison

The comparison that is made between the OpenCV and Visual SfM program is summarized in table 1. The following points are noted.

- There are some algorithms that are not available about the Visual SfM that are denoted by (n/A) in Table 1.
- Extraction of feature points in the images is the most important part of SfM process, and consists the main difference between the approaches of table 1. Based on the comparative results it is noticed that SIFT algorithm used by Visual SfM is accurate and robust but time costs even though it can be used with GPU. Whereas ORB of OpenCV that is fast and more efficient than SIFT.
- Even though Visual SfM utilizes GPU-accelerated SIFT in feature tracking and multi core bundle adjustment to save on computation time, but the constructed 3D model is very sparse. In most cases the 3D point cloud generated based on Visual SfM is refined to a finer resolution using Multi-View Stereo (MVS) programs. However, the MVS software are computationally expensive in terms of processing time and memory resources because they are based on dense matching algorithms.
- The comparative results shown in fig. 3, it is concluded that OpenCV implementation of SfM is effective to satisfy the near real time requirements in terms of computation time, precision and spatial resolution of the 3D model.

Table 1. Comparison between Visual SfM and OpenCV

Characteristics	Visual SfM	OpenCV
Features Extraction	SIFT	ORB
Features Matching	Sequential preemptive	Brute force
Robust Estimation	RANSAC	RANSAC
Triangulation	n/A	DLT
Image Registration	n/A	PnP RANSAC
Bundle Adjustment	Multicore BA	Ceres
Visualization	OpenGL	PCL

5 Conclusions

This paper presents the evaluation of an OpenCV implementation of incremental SfM approach using open source data. The comparative results with Visual SfM program demonstrate that the proposed SfM approach is capable to produce high quality 3D point clouds in terms of precision and spatial resolution.

In the future works the following tasks are suggested. Developing the proposed SfM approach for very large scale 3D reconstruction. Using parallel computing methods based on GPU processors to accelerate SfM process. Investigation of visual SLAM and 3D reconstruction by considering no prerecorded dataset.

References

1. Wang, R., Peethambaran, J., Chen, D.: LiDAR Point Clouds to 3-D Urban Models: A Review. IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing. Vol. 11, no. 2, pp. 606–627 (2018).
2. Kühner, T., Kümmerle, J.: Large-Scale Volumetric Scene Reconstruction using LiDAR. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA). pp. 6261-6267(2020).
3. You, R.J., Lin, B.: A Quality Prediction Method for Building Model Reconstruction Using LiDAR Data and Topographic Maps. IEEE Transactions on Geoscience and Remote Sensing. Vol. 49, no. 9, pp. 3471–3480 (2011).
4. Orthuber, E., Avbelj, J.: 3D Building Reconstruction from LiDAR Point Clouds by Adaptive Dual Contouring. ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences (2015).
5. Shao, Z., Yang, N., Xiao, X., Zhang, L., Peng, Z.: A Multiview Dense Point Cloud Generation Algorithm Based on Low–Altitude Remote Sensing Images. Remote Sensing, MDPI. Vol. 8, no. 5, pp. 381 (2016).
6. Hu, T., Sun, X., Su, Y., Guan, H., Sun, Q., Kelly, M., Guo, Q.: Development and Performance Evaluation of a Very Low-Cost UAV-LiDAR System for Forestry Applications. Remote Sensing. Vol. 13, no. 1, pp. 77 (2020).
7. Obanawa H., Shibata H.: Applications of UAV Remote Sensing to Topographic and Vegetation Surveys. In: Unmanned Aerial Vehicle: Applications in Agriculture and Environment. pp. 131–142 (2020).
8. James, M. R., Robson, S., Smith, M.: 3-D uncertainty-based topographic change detection with structure-from-motion photogrammetry: precision maps for ground control and directly georeferenced surveys. Earth Surface Processes and Landforms. Vol. 42, no. 12, pp. 1769–1788 (2017b).
9. Stott, E., Williams, R.D., Hoey, T.B.: Ground Control Point Distribution for Accurate Kilometre-Scale Topographic Mapping Using an RTK-GNSS Unmanned Aerial Vehicle and SfM Photogrammetry. Drones. pp. 4-55 (2020).
10. Biljecki, F., Sindram, M.: Estimating Building Age with 3D GIS. ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences. Vol. IV-4/W5 (2017).
11. Kulawiak, M., Kulawiak, M., Lubniewski, Z.: Integration, Processing and Dissemination of LiDAR Data in a 3D Web-GIS. ISPRS International Journal of Geo-Information. Vol. 8, no. 3, pp. 144 (2019).
12. Ahmad, J., Malik, A.S., Xia, L., Ashikin, N.: Vegetation encroachment monitoring for transmission lines right-of-ways: a survey. Electric Power Systems Research. Vol. 95, pp. 339–352 (2013).
13. Shi, Z., Lin, Y., Li, H.: Extraction of urban power lines and potential hazard analysis from mobile laser scanning point clouds. International Journal of Remote Sensing. Vol. 41, no. 9, pp. 3411-3428 (2020).
14. Olsson, C., Enqvist, O.: Stable Structure from Motion for Unordered Image Collections. In: Proceedings of the 17th Scandinavian conference on Image analysis. pp. 524-535 (2011).
15. Moulon, P., Monasse, P., Marlet, R.: Global Fusion of Relative Motions for Robust Accurate and Scalable Structure from Motion. In: Proceedings of the IEEE Conference on Computer Vision. pp: 3248-3255 (2013).
16. Dzenan, L., Jasmin, V., Haris, B.: Framework for automated reconstruction of 3D model from multiple 2D aerial images. In: Proceedings of the International Symposium ELMAR. pp. 173-176 (2017).

17. Al Khalil, O.: Structure from motion (SfM) photogrammetry as alternative to laser scanning for 3D modelling of historical monuments. Open Science Journal. Vol. 5, no. 2 (2020).
18. Wallace, L., Lucieer, A., Malenovský, Z., Turner, D., Vopěnka, P.: Assessment of Forest Structure Using Two UAV Techniques: A Comparison of Airborne Laser Scanning and Structure from Motion (SfM) Point Clouds. Forests. Vol. 7, no. 3, pp. 62 (2016).
19. Alouache, A., Yao, X., Wu, Q.: Creating Textured 3D Models from Image Collections using Open Source Software. International Journal of Computer Applications. Vol. 163, no. 9, pp. 14-19 (2017).
20. Saputra, M.R.U., Markham, A., Trigoni, N.: Visual SLAM and Structure from Motion in Dynamic Environments: A Survey. ACM Computing Surveys (CSUR). Vol. 51, no. 2, pp. 37 (2018).
21. Özyeşil, O., Voroninski, V., Basri, R., Singer, A.: A Survey of Structure from Motion. Acta Numerica. Vol. **26**, pp. 305-364 (2017).
22. Ham, H., Wesley, J., Hendra, H.: Computer Vision Based 3D Reconstruction: A Review. International Journal of Electrical and Computer Engineering. Vol. 9, no. 4, pp. 2394-2402 (2019).
23. Snavely, N., Seitz, S.M.; Szeliski, R.: Modeling the World from Internet Photo Collections. International Journal of Computer Vision. pp:189-210 (2008).
24. Schönberger, J.L., Frahm, J.M.: Structure-from-Motion Revisited. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 4104-4113 (2016).
25. Moulon P., Monasse P., Perrot R., Marlet R.: OpenMVG: Open Multiple View Geometry. In: Proceedings of the International Workshop on Reproducible Research in Pattern Recognition. pp. 60-74 (2016).
26. Wu, C.: VisualSFM: A Visual Structure from Motion System (2011). http://www.cs.washington.edu/homes/ccwu/vsfm/.
27. Sweeney, C.: Theia Multiview Geometry Library: Tutorial & Reference. http://theia-sfm.org.
28. Baggio, D. L., Emami, S., Escriva D., Ievgen M. K., Saragih J., Shilkrot R.: Mastering OpenCV 3. Packt Publishing (2017).
29. Lepetit, V., Moreno-Noguer, F., Fua, P.: EPnP: An Accurate o(n) Solution to the PnP Problem. International Journal of Computer Vision. Vol. 81, no. 2, pp. 155-166 (2009).
30. Agarwal, S., Mierle, K.: Ceres solver (2012). http://ceres-solver.org.
31. Torres, J.C., Arroyo, G., Romo, C., De Haro, J.: 3D Digitization using Structure from Motion". In: Proceeding of the Spanish Computer Graphics Conference (2012).
32. https://www.robots.ox.ac.uk/~vgg/data/mview.