ResearchGate

See discussions, stats, and author profiles for this publication at: https://www.researchgate.net/publication/312561280

Open source stack for Structure from Motion 3D reconstruction: A geometric overview

Conference Paper · August 2016

DOI: 10.1109/INAES.2016.7821933

CITATIONS
6

READS
682

2 authors:

Djurdjani Djurdjani
Universitas Gadjah Mada
8 PUBLICATIONS **12** CITATIONS
SEE PROFILE

Dany Laksono
Universitas Gadjah Mada
15 PUBLICATIONS **49** CITATIONS
SEE PROFILE

Some of the authors of this publication are also working on these related projects:

NodeJS Fullstack WebGIS View project

All content following this page was uploaded by Dany Laksono on 20 March 2019.

The user has requested enhancement of the downloaded file.

Open Source Stack for Structure from Motion 3D Reconstruction: A Geometric Overview

Djurdjani
Department of Geodetic Engineering
Faculty of Engineering, Gadjah Mada University (UGM)
Yogyakarta, Indonesia

Dany Laksono
Department of Geodetic Engineering
Faculty of Engineering, Gadjah Mada University (UGM)
Yogyakarta, Indonesia

***Abstract*— Given a series of overlapped images, it is possible to reconstruct camera position and orientation, as well as 3D coordinates of object in the images. Such method is known as Stucture from Motion (SfM), which is currently implemented in various commercial and open source software. The first softwarse commonly don't provide Sfm algorithm. While the latter software are commonly developed partially. This research aims to perform a SfM reconstruction using a stack of open source SfM software, with each software serves different purposes in producing 3D models from images. A python script were employed to produce seamless integration of the software. Geometric evaluations were performed to the result of SfM reconstruction on three datasets obtained using mobile handheld camera. The deliverables of the stack were compared to result of commercial software using cloud-to-cloud comparison to obtain geometric closeness of both the results. Ground truth measurement were conducted to assess the positional consistency in SfM reconstruction on different dataset. The test showed that 3D models resulted from the open source stack perform on pair with the result of Agisoft Photoscan as indicated by the small RMSE value obtained from cloud-to-cloud comparison (0.019818 m and 0.350701 m for two datasets, respectively). The result of ground-truth evaluation indicates large error possibly due to the homogeneity in the dataset used, as indicated by the camera pose graph.**

Keywords*— *3D Reconstruction, Structure from Motion, Open Source

I. INTRODUCTION

The Structure from Motion (SfM) algorithm could be defined as a method to obtain 3D coordinates of points located in an object (the 'structure') as well as the camera position and orientation at the time the images are captured (the 'motion')[1]. Since SfM relies on solving non-linear mathematical models, this method has not been largely implemented in the past due to limitations in computational capabilities. However, rapid development of hardware and software technologies in the last decades has prooved that SfM algorithm has capabilites for producing point-clouds, mesh surfaces and 3D models efficiently due to its simplified workflow [2]. The automatic workflow of SfM become a challenging issues in photogrammetric field where geometric accuracy is a pivotal objective. On the other hand, SfM algorithm offers automatic 3D modelling workflow with lower cost implementation compared to other methods, such as land surveying [3]. Recent advances shows combinations in the field of photogrammetry and computer vision [4].

One of the adaptations is utilization of Structure from Motion (SfM) algorithm to obtain accurate 3D models of topographic and non-topographic objects. Adaptations of SfM algorithm for topographic or non-topographic 3D modeling can be traced in photogrammetry softwares which implement the algorithm. Examples of software (commercials and open source) implementing SfM algorithm are demonstrated in [5]. While providing users with easy-to-use interface, the SfM algorithm behind reconstruction process in commercial software are usually not made available to the public. This might be beneficial for end-users which main purpose is to produce 3D models from images. However, researchers who wants to employ different variables for the algorithm would prefer to use open source software. While different licenses could be applied, most open source software would make the algorithm and source code available to the public, thus provides the users with customizable parameters for Structure from Motion reconstruction.

Discussion on the implementation of Structure from Motion as an open source software can be found in the works of [6]. As part of a research, open source software implementing SfM algorithm often employs state-of-the-art development in SfM method. For example, the Global Structure from Motion algorithms [7] were developed instead of the widely used incremental SfM method. Open Source implementation of Global SfM were available as open source softwares called OpenMVG [8]. Open source SfM softwares often serve particular aspect of SfM pipeline (e.g. sparse reconstruction), thus several libraries need to be employed to produce a 3D model using SfM algorithm.

This research combine three different open source libraries i.e. OpenMVG [8], MVE [9] [10] and MVS-Texturing [11] in to an open source stack using python script. Each of the three libraries serves different purposes in SfM pipeline. 3D models resulted from the stack were evaluated against ground truth measurement (which conducted using Total Station Measurement) and commercial software using cloud-to-cloud method.

978-1-5090-0741-7/16/$31.00 ©2016 IEEE

II. LITERATURE REVIEW

A. Structure from Motion

Although similarly related to photogrammetry in that both method extract 3D information from 2D images, the algorithm of SfM differs from the latter. While photogrammetry focuses on the derivation of geometric information based on carefully measured position of acquisition device (i.e. camera), Structure from Motion uses geometric relationship in stereo vision and artificial intelligence to estimate the camera pose (position and orientation) and 3D positions of the object in question [12].

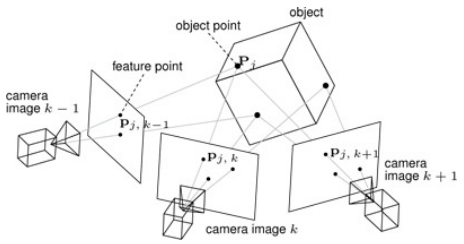

Fig. 1. SfM Scene Configuration [13]

In the case of a monocular camera, the three views (Figure 1) are produced by moving the camera in three different position (k-1), k, and (k+1). If the point P in object projected as Pj,k-1, Pj,k, and Pj,k+1 in each image frame respectively, the geometry of the scene can be reconstructed using the relationship of Essential matrix and Fundamental matrix. When more cameras are used, the equation could be expanded and solved as linear problem as shown in [14] or non-linear problem as stated in [15].

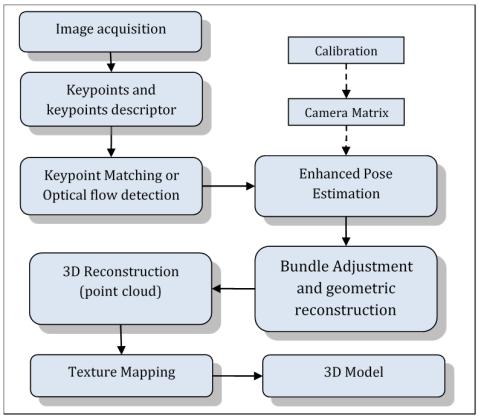

Fig. 2. Structure from Motion Pipeline

There are various SfM algorithm to solve a scene of SfM problems. Discussion on three main methods used to estimate position and orientation of the camera prior to solving SfM problems could be found in [16]: a) Sequential/incremental method, b) Factorization Method and c) Global Method. The Global method are used for its faster solution and less processing resources needed [7], thus it is suitable for solving SfM problems on-the-fly. Global Method in this paper refers to [7], and was implemented using open source SfM library called OpenMVG.

B. SfM Software Implementation

Among the implementations of SfM are software and libraries that provide SfM reconstruction. According to its license of use, there are four categories of SfM software/library [6]:

a. Commercial software, which obliged user to pay for the software in order to use the SfM reconstruction features. Examples of software in this category are Agisoft Photoscan, Photomodeller Scanner, Pix4D Mapper, and so on.
b. Sharewares, where most of the features are available for use, but at some point user are obliged to pay (for additional features or time). An example of this software is Autodesk 123D catch
c. Free software. Users of this software are not obliged to pay any fees for using the SfM reconstructions. However, the user will not be able to modify the algorithm behind the software. Examples of these software are VisualSfM and CMPMVS
d. Open Source software. Of all the software above, this category is the least strictly licensed software. The user are free to use the software and modify the source code as they want. Examples of this software are OpenMVG, MVE and Theia [17].

Open source software usually developed as libraries which serve particular purposes from SfM pipeline. Users need to combine the libraries to perform complete SfM pipeline, as shown in [18]. The developed software architecture chain for SfM reconstruction use free and/or open source software, i.e. Bundler, CMVS/PMVS, Poisson Surface Reconstruction, and Meshlab.

C. Geometric Evaluation

Internal evaluation and external evaluation are often used to evaluate the robustness of SfM products. The formula for evaluating correctness of epipolar geometry EG are given as follow [19]:

$$E_R = \frac{1}{M}\sum_{p=1}^{M}\hat{P}_{missing}(p) = \frac{1}{M}\sum_{p=1}^{M}\frac{1}{N}\sum_{i=1}^{N}\hat{P}_{missing}(p,i) \quad (1)$$

With N being the total number of images and M is the total number of 3D points reconstructed from the view. $\hat{P}_{missing}(p,i)$ is the probability that the feature descriptor (e.g. SIFT) does not match its image projection in i, with $\hat{P}_{missing}(p)$ being its average in all views. Since SfM depends on the correctness of feature detection and matching, evaluating the erroneous detected features against the whole scene would give estimated robustness of a scene. The internal evaluation can be expressed as a connected visibility graph or pairwise matrix which shows correspondences between each camera in a scene.

External evaluation is performed by comparing result of SfM reconstruction with ground truth data. Iterative Closest Point (ICP) is commonly used for this particular purpose. The so-called ICP algorithm performs transformation of 'source' point cloud to 'target' and iteratively refines the transformation until minimal error is achieved [20]. ICP can be used to register two point clouds from different sources, e.g. from SfM

reconstruction and terrestrial laser scanner [3]. The method to employ cloud-to-cloud method for comparison of SfM reconstruction against ground truth data are found in researches such as [21]. Implementation of ICP for cloud-to-cloud comparison can be found on software such as CloudCompare [22], which is used in this research.

III. METHODOLOGY

A. Research Tools and Datasets

These three datasets below are used to test the result of online Structure from Motion web service:

a. Tree dataset, consist of 91 images of a tree in front of Department of Geodetic Engineering's building.

b. Dragon dataset, consist of 81 photos of a Javanese Dragon statue.

c. UGM Central building dataset, consist of 35 photos of the southern part of UGM Central office building.

Each of the dataset were obtained using mobile handset, i.e. Samsung Galaxy Grand 2 Duos. Each of the photo has image dimensions of 3264 x 2448 pixels with 72 dot per inch image resolution and 24 bit sRGB color depth. Different ISO speed and Exposure Time were obtained for all images in the three datasets due to the automated settings of the handheld device's camera, although the focal length remains the same for each image (i.e. 3 mm). Of the three datasets, two datasets were used to perform cloud-to-cloud evaluation, while one dataset were employed for ground-truth evaluation.

Fig. 3. Excerpt from the three datasets used for evaluation: Dragon, Tree and UGM Building

Research tools can be divided into two categories, which are hardware and software. Hardware used in this research were as follows:

a. Samsung Galaxy Grand 2, with Samsung 8 MP (megapixel) rear camera (Quadcore 1.2GHz Cortex-A7 and Adreno 305, 1.5Mb). Used to obtain all the datasets.

b. Leica Reflectorless Total Station TS02-5 (serial number: 1338740), used to obtain ground truth data on UGM dataset

Software used for SfM processing are various open source SfM libraries which serves different purposes for SfM reconstruction pipeline, which are:

a. OpenMVG v0.9 '*Corydoras sterbai*' [23]. This libraries are used for Global SfM reconstruction. Deliverables are structure (sparse and dense, colorized point-cloud) and motion (camera path).

b. MVE - Multiview Environment [9]. This software consist of libraries for conducting sparse and dense SfM reconstruction, generate depth maps and producing mesh from sparse point cloud.

c. FSSR - Floating-Scale Surface Reconstruction [10]. This software consist of libraries used for noise reduction and surface reconstruction from dense point cloud. As of October 2015, the libraries in this software were merged with MVE.

d. MVS-Texturing [11]. This library used state-of-the-art techniques for texturing reconstructed surface. Deliverables are 3D model of textured object in OBJ format.

B. Development of SfM Stack

Given that each of the SfM software used in this research is an independent libraries which serves different aspects of Structure from Motion, an underlying framework needs to be used to assure that each libraries works seamlessly. A script written in Python were developed to handle the task of integrating all the libraries. Different libraries from openMVG, MVE, FSSR and MVS-Texturing sit on top of this script and perform corresponding operations to chain the input and output of each libraries to produce final 3D models.

The python script develops an integration of Global Structure from Motion pipeline, point-cloud densification, multi-view meshing and cleaning and texture draping in order to seamlessly integrate various SfM libraries into the desired output. The python script translates default and advanced parameters from the client and setup the SfM reconstruction accordingly.

The open source stack combines different input and output from each libraries which have different parameter settings. In order to maintain the use of different parameters, the Python script were designed to be able to read customized parameters from a *settings.json* file. For example, in its default settings Open source stack uses estimated focal camera based on the following formula [7]:

$$\text{Focal}_{pix} = \frac{max(w_{\text{pix}}, h_{\text{pix}}) * \text{focal}_{\text{mm}}}{\text{ccdw}_{\text{mm}}} \quad (2)$$

where:

focal_{pix} : Focal length from EXIF data in pixel

focal_{mm} : Focal length in mm

$w_{\text{pix}}, h_{\text{pix}}$: Image size in pixel (width*height)

ccdw_{mm} : Sensor width size (CCD) of the camera

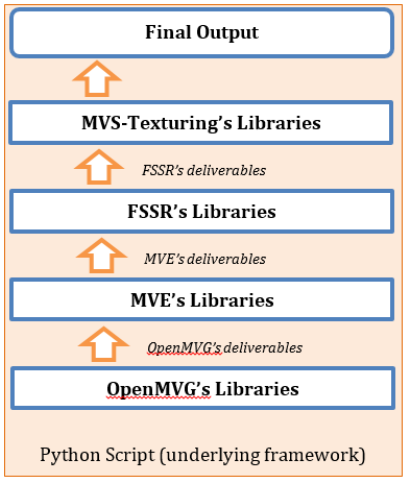

Fig. 4. The Open Source SfM Stack

Fig. 5. 3D Models, result of three datasets

According to the camera properties, the focal length of Samsung Galaxy Grand 2 camera is 2.93 mm, the image size (width and length) is 3264x2448 and the sensor size is approximately 3x2.3 mm. Thus, according to the formula above, the focal length of Samsung Galaxy Grand 2 is 3187.84 pixels, which was used during the acquisition of the datasets.

C. *Evaluations of 3D Models*

Cloud-to-cloud comparison of 3D models resulted from the open source stack were performed using CloudCompare, an open source software for comparing point-clouds [22]. The datasets used for the comparison were Dragon and Tree dataset. For the comparison, the same images from both the dataset were run through commercial software Agisoft Photoscan. The 3D models from the open source stack would be used as the 'model', while the Agisoft Photoscan's 3D models would be used as the 'reference'.

CloudCompare was also used in order to register and obtain coordinates of sample points from 3D models of UGM Main building dataset run through Open source stack. The obtained coordinates were then compared with the result of ground-truth measurement conducted using Total Station to find RMS Error of the measured coordinates with respect to coordinates from CloudCompare.

IV. RESULT AND DISCUSSION

Evaluations were conducted to test the results of the reconstruction session. The results of Open source stack reconstruction was a set of textured 3D models for each dataset used in this thesis, i.e. Dragon dataset, Tree Dataset and UGM Dataset. Figure 5 shows the result of 3D reconstruction using Dargon dataset (top left), Tree dataset (top right) and UGM Dataset (bottom). The result of Dragon and Tree dataset are visually identical to the object in question, and both results possessed little to no noise in the images. Different result can be observed in UGM dataset which shows erroneous result of the 3D modeling.

The results of Open source stack reconstruction were evaluated using three kinds of evaluation: Internal evaluation, which evaluates the internal reliability of each reconstruction project; and the external comparison, consist of cloud-to-cloud comparison, which employs Iterative Closest-Point (ICP) algorithm to assess two point clouds from commercial software and Open source stack and Ground-truth evaluation, which was performed by comparing coordinates of sample points obtained from Total Station measurement and coordinates extracted from the result of Open source stack reconstruction.

TABLE I. RESULT OF INTERNAL EVALUATION

Dataset	No. of reconstructed camera pose	RMSE (meter, unscaled)	Max. Residual Value
Dragon	81 from 81	0.90439	5.44819
Tree	91 from 91	0.607263	3.99226
UGM Building	67 from 68	0.845978	16.0597

The internal evaluations consist of observation on the residuals and outlier test and the observation of RMSE (Root Mean Square Error) value in the dataset. The small value of RMS Error indicates the geometrical closeness between the observed coordinates of points in all the iterations of views (i.e. each of the estimated camera pose), and thus indicates consistency of the observation. However, the value did not necessarily means that the reconstructed structure were accurate. It can be observed that while the RMS Error (in relative point-cloud unit) yields relatively small number (i.e. 0.845978), the UGM dataset have large numbers of maximum residual value (16.0597) compared to that of Naga dataset (5.44819) and Tree dataset (3.99226). This indicates the existence of outliers and erroneous computation of camera pose in the scene.

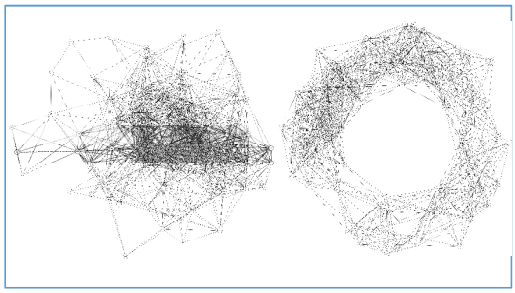

Fig. 6. Camera pose graph of Dragon (*left*) and Tree (*right*) datasets

Cause of the erroneous observations can be detected from the camera pose and view graph. The graph expressed how an image is connected or shared views with other images. Figure 6 shows camera pose or visibility graph of Dragon dataset, Tree dataset and UGM Building dataset. The visibility graph of Dragon and Tree dataset shows the connected views as expected. The connected images (shown in the graph as nodes) were consistent with the relative position of the camera during data acquisition. Meanwhile, the visibility graph for UGM Building datasets in Figure 7 shows inconsistencies with the camera position during data acquisition. The graph shows two groups connected to each other, while in reality each of the group is a collection of images capturing different wings of the building. It is possible that the images were connected during feature detection due to the similarity of the feature.

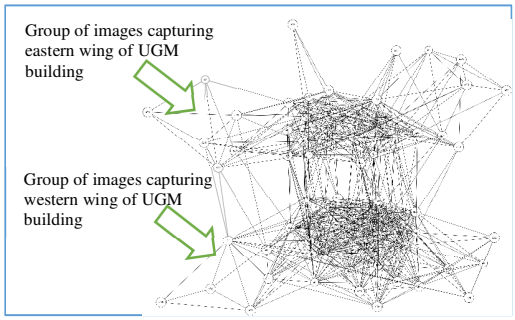

Fig. 7. Camera pose graph of UGM datasets

Cloud-to-cloud evaluation were conducted to both the Dragon and Tree dataset. The evaluation were conducted using CloudCompare on the two pairs of datasets: 3D models from Agisoft Photoscan and 3D models resulted from Open source stack. The result of cloud-to-cloud comparison of Naga dataset are shown in Figure 8.

The color represent the distance of each point in 'model' point cloud (Dragon 3D model from Open source stack) to the closest point in 'reference' point cloud (Dragon 3D model from Agisoft Photoscan). The color space used to represent the point cloud distance is blue-green-yellow-red, where blue color indicates the closest distance (close to zero or identical point cloud's position), and vice versa. The same result is obtained for the Tree Dataset. It can be concluded that both the tested datasets (Dragon and Tree dataset from the open source stack projects) and the reference datasets (Dragon and Tree dataset from Agisoft projects) were geometrically similar.

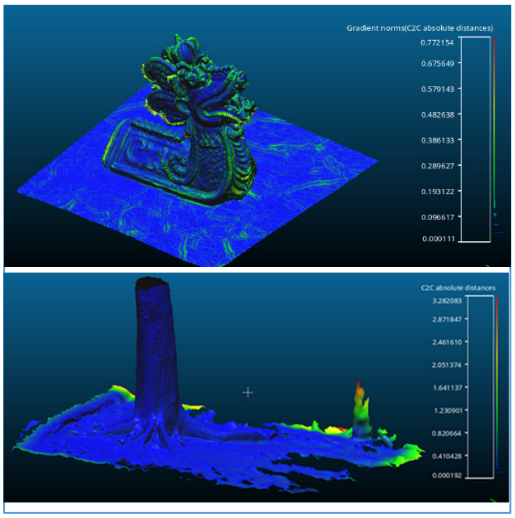

Fig. 8. C2C Evaluation of two datasets (*Top: Dragon, Bottom: Tree*)

The cloud-to-cloud (C2C) comparison conducted using CloudCompare calculates distances between the tested dataset and the reference dataset. Among the value obtained from the test are Mean Distance and Standard Deviation, which express the closeness of both model and reference dataset. The value for mean distances (in relative point cloud unit) of Dragon dataset and Tree dataset are 0.028808 and 0.176421 in relative point cloud unit, respectively.

TABLE II. C2C EVALUATION – MEAN AND STANDARD DEVIATION

	Mean Distance (meter, unscaled)	Standard Deviation (meter, unscaled)
Dragon Dataset	0.028808	0.019818
Tree Dataset	0.176421	0.350701

This result indicates that the 3D models (from Open source stack and Agisoft Photoscan) are close to each other. Another indication that the results from Open source stack perform on par with the result from Agisoft Photoscan is the value of their standard deviations, which yield 0.019818 and 0.350701 relative point cloud unit for Dragon dataset and Tree dataset, respectively.

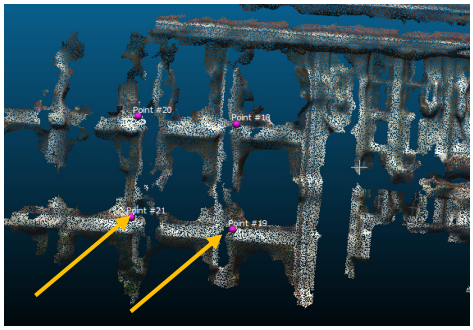

Fig. 9. Two points in UGM Dataset with largest discrepancies (shown with arrows)

Ground-truth evaluation were conducted by comparing set of coordinates measured using Total Station and coordinates retrieved from the 3D models of Open source stack project using CloudCompare. The result of this comparison showed quite large value of Total Root Mean Square (RMS) Error. The calculated value are 6.4733 m for RMS_X; 2.1322 m for RMS_Y; and 1.1879 m for RMS_Z. Thus, the accumulated error or RMS_{TOTAL} are 6.9182 m. The largest RMS Error were observed in point #19 and #21, which yielded RMS value of 12.36 m and 36.53 m, respectively. This values could be considered as outlier compared to the result obtained in other points. The large error were obtained due to inaccuracies of the 3D models and the failure to generate reconstruction on the eastern wing of UGM Dataset.

V. Conclusion

We presented a comparison of Structure from Motion reconstruction using open source software and commercial software. An open source stack consisted of three different software, i.e. OpenMVG, MVE and MVS-Texturing, were seamlessly integrated using a python script. The deliverables of this open source stack were compared to the result of the same dataset processed using Agisoft Photoscan. Result of the cloud-to-cloud assessment suggests that the 3D models resulted from open source stack are geometrically close to the same 3D models from commercial software. However this research failed to demonstrate the spatial accuracy of 3D model from the open source stack compared to ground truth data obtained using Total Station. The large discrepancies of some sample points in the UGM dataset are caused by failure in camera pose orientation during initial sparse reconstruction as indicated in camera pose graph. Therefore, future research on this topic should consider using a more controlled environment for the purpose of accuracy assessment using ground truth data.

VI. Acknowledgment

The authors of this paper would like to thank Department of Geodetic Engineering, Faculty of Engineering, Gadjah Mada University for providing financial support for the research. We would also thank Pierre Moulon and Simon Fuhrmann for their support and help on OpenMVG and MVE during the research.

VII. References

[1] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision*. Cambridge university press, 2003.

[2] R. Klette and R. Reulke, "Modeling 3D scenes: Paradigm shifts in photogrammetry, remote sensing and computer vision," CITR, The University of Auckland, New Zealand, 2005.

[3] M. J. Westoby, J. Brasington, N. F. Glasser, M. J. Hambrey, and J. M. Reynolds, "'Structure-from-Motion' photogrammetry: A low-cost, effective tool for geoscience applications," *Geomorphology*, vol. 179, pp. 300–314, Dec. 2012.

[4] W. Förstner, "Computer vision and photogrammetry–mutual questions: geometry, statistics and cognition," *Bild. Sci. Swed. Soc. Photogramm. Remote Sens.*, pp. 151–164, 2002.

[5] M. Pierrot-Deseilligny, L. D. Luca, and F. Remondino, "Automated Image-Based Procedures for Accurate Artifacts 3D Modeling and Orthoimage Generation," *Geoinformatics FCE CTU*, vol. 6, no. 0, pp. 291–299, Dec. 2011.

[6] F. Remondino, S. Del Pizzo, T. P. Kersten, and S. Troisi, "Low-cost and open-source solutions for automated image orientation–A critical overview," in *Progress in cultural heritage preservation*, Springer, 2012, pp. 40–54.

[7] P. Moulon, P. Monasse, and R. Marlet, "Global fusion of relative motions for robust, accurate and scalable structure from motion," in *Computer Vision (ICCV), 2013 IEEE International Conference on*, 2013, pp. 3248–3255.

[8] P. Moulon, P. Monasse, and R. Marlet, *OpenMVG (open Multiple View Geometry)*. 2012.

[9] S. Fuhrmann, F. Langguth, and M. Goesele, "MVE-A Multi-View Reconstruction Environment," in *Eurographics Workshop on Graphics and Cultural Heritage*, 2014, pp. 11–18.

[10] S. Fuhrmann and M. Goesele, "Floating scale surface reconstruction," *ACM Trans. Graph. TOG*, vol. 33, no. 4, p. 46, 2014.

[11] M. Waechter, N. Moehrle, and M. Goesele, "Let There Be Color! Large-Scale Texturing of 3D Reconstructions," in *Computer Vision–ECCV 2014*, Springer, 2014, pp. 836–850.

[12] H. Wang, S. Shen, and X. Lu, "Comparison of the camera calibration between photogrammetry and computer vision," in *2012 International Conference on System Science and Engineering (ICSSE)*, 2012, pp. 358–362.

[13] T. Thormählen, N. Hasler, M. Wand, and H.-P. Seidel, "Registration of sub-sequence and multi-camera reconstructions for camera motion estimation," *J. Virtual Real. Broadcast.*, vol. 7, no. 2, pp. 1–10, 2010.

[14] I. Thomas and E. Simoncelli, "Linear structure from motion," *NSF Sci. Technol. Cent. Res. Cogn. Sci.*, no. IRCS Report 94–26, 1994.

[15] R. Szeliski and S. B. Kang, "Recovering 3D shape and motion from image streams using nonlinear least squares," *J. Vis. Commun. Image Represent.*, vol. 5, no. 1, pp. 10–28, 1994.

[16] N. Jiang, Z. Cui, and P. Tan, "A global linear method for camera pose registration," in *Computer Vision (ICCV), 2013 IEEE International Conference on*, 2013, pp. 481–488.

[17] C. Sweeney, T. Hollerer, and M. Turk, "Theia: A Fast and Scalable Structure-from-Motion Library," in *Proceedings of the 23rd Annual ACM Conference on Multimedia Conference*, 2015, pp. 693–696.

[18] A. J. Rossi, H. Rhody, C. Salvaggio, and D. J. Walvoord, "Abstracted workflow framework with a structure from motion application," in *Image Processing Workshop (WNYIPW), 2012 Western New York*, 2012, pp. 9–12.

[19] N. Jiang, P. Tan, and L.-F. Cheong, "Seeing double without confusion: Structure-from-motion in highly ambiguous scenes," in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, 2012, pp. 1458–1465.

[20] P. J. Besl and N. D. McKay, "Method for registration of 3-D shapes," in *Robotics-DL tentative*, 1992, pp. 586–606.

[21] K. Thoeni, A. Giacomini, R. Murtagh, and E. Kniest, "A comparison of multi-view 3D reconstruction of a rock wall using several cameras and a laser scanner," in *Proceedings of ISPRS Technical Commission V Symposium, Riva del Garda, Italy*, 2014, pp. 23–25.

[22] D. Girardeau-Montaut, "CloudCompare-Open Source project," *OpenSource Proj.*, 2011.

[23] P. Moulon, P. Monasse, and R. Marlet, "Adaptive Structure from Motion with a contrario model estimation," in *Computer Vision–ACCV 2012*, Springer, 2013, pp. 257–270.

View publication stats