Incremental structure-less SfM using Trifocal Tensors

Chuqiao Li, Wenhao Xu, Yiming Zhao, Zilin Fang
ETH Zurich
chuqli,wenhxu,yimzhao,zifang@student.ethz.ch

Abstract

In this project, we will use structure from motion method to get camera poses and reconstruct three-dimensional point clouds using two-dimensional image sequences. In order to reduce the accumulated errors, we will explore a structure-less approach where we avoid triangulating 3D points and instead perform a minimization problem using the trifocal constraints similar to [6]. This is implemented by optimizing the trilinear constraints error from multiple triplets of different camera views. As the reconstruction grows, conventional bundle adjustment (BA) by triangulating 3D points, which has to go through all data, becomes increasingly time-consuming, slowing down the reconstruction. This new method can speed up the process since it divides pose calculation and 3D points reconstruction into two parts and it can increase the accuracy of camera poses at the same time.

1. Introduction

Conventional incremental Structure-from-Motion pipelines work by alternating between estimating the pose of a new image and triangulating new 3D points. To avoid the accumulation of errors it is necessary to perform local optimization, called Bundle Adjustment, frequently. However, as the reconstruction proceeding, each iteration have to go through more and more data. Thus we use trilinear constraints for BA and combine incremental and global SfM to make the whole process more efficient and accurate.

1.1. Structure from motion

Structure from Motion (SfM) [7] is an imaging technique for estimating parameters of three-dimensional structures and of camera motion from features of two-dimensional image sequences that may be coupled with local motion signals. It is studied in the fields of computer vision and visual perception. Assuming the feature position errors are zero-mean Gaussian, by minimizing the sum of squares of residuals we can get the maximum likelihood estimation. In computer vision literature, this process is called bundle adjustment. [6] Structure from motion methods generally can be categorized as incremental or global based on the initialization of BA . Incremental SfM initialize cameras one by one, whereas global SfM initialize cameras all together and have better performance in efficiency and accuracy. [12]

Structure from motion technique represents a major advancement in the field of photogrammetry for geoscience applications. In paper [11], structure from motion is implemented to solve camera pose and scene geometry simultaneously using a highly redundant bundle adjustment based on the matching features between partially overlapping images. By comparing results obtained by terrestrial laser technique, it reveals that decimetre-scale vertical accuracy can be achieved using structure from motion even for sites with complex topography and a range of land-covers. Structure from motion is implemented in remote or rugged environment where terrestrial laser scanning technique has limitations. It is presented in paper [9] an evalution of structure from motion photogrammetry and point-cloud-based techniques for capturing the geometry and volume of large wood(LW) in the field.

Structure from motion is also often used to estimate situations as well as planning and maintenance efforts and costs. structure from motion gives an in-invasive way for the structure of heritage without direct interaction. [2]

1.2. Bundle adjustment

Bundle adjustment is the problem of refining a visual reconstruction to produce jointly optimal 3D structure and estimating camera parameters (camera pose and/or calibration). Optimal means the method minimize some cost function that quantifies the model fitting error to estimate the camera parameters. Jointly means that the solution is simultaneously optimal with respect to both structure and camera variations.[10]

Bundle adjustment is almost always used both as the last step of every feature-based 3D reconstruction algorithm, and sometimes also as an intermediate steps, for example, during tracking. Many of BA algorithms need to go through all data at each step, which slows down the BA process. However, structure less approaches are alternative to these methods, which can speed up the process and at the same time give satisfactory estimations. For example, Global Epipolar Adjustment (GEA) is one of the structure-less methods which use error based on the linear constraint on the epipolar constraints for each pair of images. Another example is GTA, which uses error based on the trilinear constraints for each triplets of images. Incremental light adjustment (iBLA) is also an example, which uses the combination of epipolar constraints and a variant of the trifocal constraints.

2. Related Works

In this section we introduce works that is relevant to our topic. In subsection 2.1 we further explain and formulize bundle adjustment problem, in subsection 2.2 we introduce the traditional structure-less bundle adjustment method of Global Epipolar Adjustment (GEA). In subsection 2.3, we briefly introduce the trifocal tensors that are used in this project.

2.1. Formulation of Bundle Adjustment Problem

The question of Bundle adjustment(BA) desires to solve the problem during feature-based 3D reconstruction algorithm. During a typical 3D reconstruction process, the reconstruction error of two consecutive frames will gradually add up, therefore to compensate this,bundle adjustment is oftem used at the end of the 3D reconstruction algorithm to minimize the re-projection error between the observed location(which is the pixel value) and re-projected points from our predicted 3D point location.

The bundle adjustment is formulated as follows, given an initial set of camera positions P_i, and our predicted 3D point coordinate $\mathbf{X}_p$, and their corresponding pixel values on the i-th camera plane x_{ip}. We want to minimize the re-projection error e

$$e = \sum_i \sum_p \|x_{ip} - \frac{1}{\lambda_{ip}} P_i \mathbf{X}_p\|^2 \tag{1}$$

The error can be expressed as the sum of squares of a large number of nonlinear, real-valued functions, therefore various non-linear least square optimization algorithms can be adopted to solve bundle adjustment. For instance, we can use gradient descent or Newton methods, one of the most predominant method used in BA is the LM (Levenberg-Marquardt) algorithm [5], which is a variation of Newton-Gaussian method.

2.2. Global Epipolar Adjustment

As is seen, the error is optimized over all camera positions and image points, therefore traditional bundle adjustment methods take significantly long time to run. Therefore, researchers have proposed structure less approaches to simplify BA optimization and speed up the process. One of the widely used method is Global Epipolar Adjustment(cite GTA 17), where epipolar constraints are used and only the camera posed are optimized rather than all the corresponding feature points. The epipolar constraint between camera pose P_i and P_j , and a world point p, is shown as follows:

$$x_{ip}^T E_{ij} x_{jp} = x_{ip}^T R_i \left[\frac{T_j - T_i}{\|T_j - T_i\|} \right]_\times R_j^T x_{jp} = 0 \tag{2}$$

Where x_{ip} and x_{jp} are the projections of p on Camera i and j, respectively, whereas R_i, R_j and T_i, T_j represent rotation matrices and centers of camera of i and j. The essential matrix is denoted as E_{ij}.$[\cdot]_\times$ represent the skew symmetric matrix of cross product.

Generally, the error to optimize for GEA is

$$e = \sum_{i,j} \sum_p (x_{ip}^T E_{ij} x_{jp})^2 \tag{3}$$

The format $x_{ip}^T E_{ij} x_{jp}$ can be rewritten as the format $K_{ijp} e_{ij}$, where e_{ij} is the flattened vector for essential matrix E_{ij}, while K_{ijp} is the Jacobian of the epipolar constraint with respect to e_{ij}. As a result, the full cost function can be rewritten as:

$$e = \sum_{i \neq j} e_{ij}^T K_{ij}^T K_{ij} e_{ij} \tag{4}$$

where K_{ij} is the Jacobian matrix of all correspondences with respect to camera i and j. It can be observed that the Jacobian matrix K_{ij} is not relevant to the camera parameters of i and j, therefore it can be computed in advance and reused in all the optimization steps. Moreover, we can compute that the dimension of K_{ij} is $n_{ij} \times 9$ where n_{ij} is the number of correspondences between frame i and j. Intuitively $n_{ij} \gg 9$, therefore, we can do QR decomposition to K_{ij} and reduce it to a 9×9 matrix $\tilde{K}_{ij}$which yields $e_{ij}^T K_{ij}^T K_{ij} e_{ij} = e_{ij}^T \tilde{K}_{ij}^T \tilde{K}_{ij} e_{ij}$. By using these methods, the calculation time for bundle adjustment can be reduced sharply.

2.3. Trifocal Tensor

In epipolar geometry, we use epipolar constraints to compute ralative positions of two cameras, while according to [6], by using triplets of cameras instead of pairs, faster convergence can be achieved and degenerate cases can be handled, and thus giving out better performance. In this section we briefly introduce the notion of trifocal tensors.

The trifocal tensor depicts the relative positions of three cameras within a triplet set. In comparison with the fundamental matrix that depicts the relation between two camera poses, the trifocal tensor has the shape of $3 \times 3 \times 3$. Among all 27 elements with in the trifocal tensor, only 18 of them are independent.

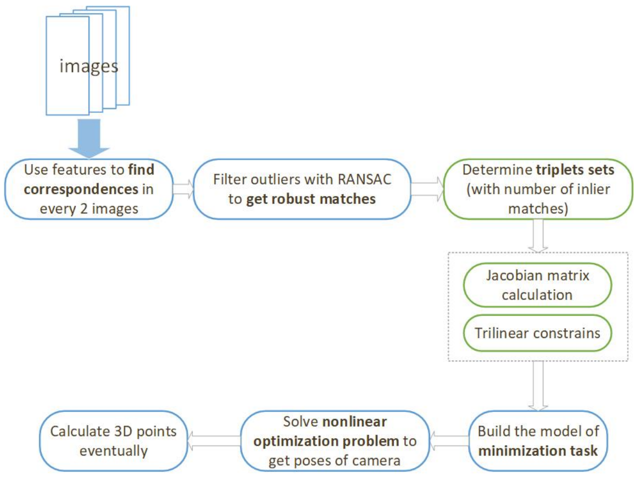
images
Use features to find correspondences in every 2 images
Filter outliers with RANSAC to get robust matches
Determine triplets sets (with number of inlier matches)
Jacobian matrix calculation
Trilinear constrains
Build the model of minimization task
Solve nonlinear optimization problem to get poses of camera
Calculate 3D points eventually

Figure 1. Roadmap of the image processing pipeline

The trifocal tensor T can be viewed as a combination of three 3×3 matrices, which we denote as T_1, T_2, T_3. Assume that the three projection matrices are $P_1 = [I|0], P_2 = [R_2|t_2], P_3 = [R_3|t_3]$, the trifocal tensors are defined as:

$$\begin{aligned} T_1 &= R_{21}t_3^T - t_2R_{31}^T \\ T_2 &= R_{22}t_3^T - t_2R_{32}^T \\ T_3 &= R_{23}t_3^T - t_2R_{33}^T \end{aligned} \tag{5}$$

The trilinear constraints of the corresponding triplet points (x, x', x'') can be expressed as

$$[x']_\times(\sum_i x_iT_i)[x'']_\times = \mathbf{0}_{3\times3}. \tag{6}$$

3. Methodology

In this section we describe the method used in our project. The overall algorithm takes in a sequence of image frames as input and generates a 3D point cloud as output. The pipeline is consist of two major parts, the first one conducts feature point extraction, while the second part put all the point correspondences into an optimization problem and estimate extrinsic parameters of all cameras. We then put those estimated parameters into Colmap, [8] which generates a point cloud for the 3D object.

This section is structured as follows: in Section 3.1 we will give an overview of the whole pipeline. While in Section 3.2 we explain the derivation and some details of the optimization problem based on trifocal tensors and trilinear constraints. Besides, in Section 3.3 we will present a more detailed algorithm about building this minimization model.

3.1. Overview of the Pipeline

The general pipeline we implemented is shown in Fig. 1. The first part includes using colmap to find correspondences in every two image pairs using typical methods such as SIFT [4]. The extracted correspondences are not necessarily robust and will contain a lot of outliers, therefore a RANSAC [1] is conducted to get only the robust matches. This part can be achieved directly in Colmap application and we only need to export all the data from the database it provides. After which we move into the second part, which is computed in MATLAB, we use a method that is based on [6] called Global Trifocal Adjustment(GTA), where we determine triplet sets based on the number of inliners, and then we formulate an optimization problem based on trilinear constraints where all the camera poses are treated as variables, and conduct LM (Levenberg-Marquardt) algorithm to find an optimal solution. Based on the predicted external parameters for each camera, we can eventually calculate a 3D point cloud.

3.2. Optimization Based on Trifocal Tensor

3.2.1 Using Jacobian Matrix to Represent trilinear constraints

Similar with the GEA problem, the Global Trifocal Adjustment method uses the extended version of epipolar constraint. The trilinear constraints for points in a triplet camera set can be expressed as (6). By vectorizing the trifocal tensor T as a 27×1 vector $\mathbf{t}$, we can compute the gradient of the trilinear constraint with respect to $\mathbf{t}$, which forms a Jacobian matrix A.

The equation system for all the points in (6) can be expressed by equation

$$A\mathbf{t} = 0 \tag{7}$$

where A has shape n_{ijk}, which means the number of correspondences in triplet camera set $\{$i,j,k$\}$. We can acquire the trifocal tensor $\mathbf{t}$ by solving the linear equations in (7).

3.2.2 Regularization and Reducing Dimension for Jacobian Matrix

The process to find feature correspondences can introduce noise into the system, and besides, although the tensor can satisfy the equation (7), it does not definitely mean that the camera poses are correct. Therefore instead of directly solving equation (7) which regards $\mathbf{t}$ as unknown, we treat the camera matrix $[R_i|t_i]$ as unknown and use equation (5) to compute vector $\mathbf{t}$. This way, we can guarantee that all the trifocal tensors during our iteration are valid and corresponds to three camera matrices. We use a regularization term which prevents the center of three cameras from converging. The final optimization problem is defined as follows:

$$\mathbf{P} = \arg \min_{(P_1, P_2, \ldots, P_n)} \sum_{(i,j,k) \in Q} W_{ijk}^2 \|A_{ijk} \mathbf{t}_{ijk}(P_i, P_j, P_k)\|^2 \tag{8}$$

$$W_{ijk} = \frac{1}{\|C_i - C_j\|} + \frac{1}{\|C_i - C_k\|} + \frac{1}{\|C_k - C_j\|} \tag{9}$$

We use C_i, C_j, C_k to denote the centers of the three cameras and P_i, P_j, P_k to represent camera matrices for camera i,j,k, respectively. If we let $P_i = [I|0], P_j = [R_2|t_2], P_k = [R_3|t_3]$, then the three centers are $0, -R_2t_2, -R_3t_3$, respectively.

The computational cost for the optimization can be further reduce when we consider the matrix A_{ijk} with shape $n_{ijk} \times 27$, the error $\|A_{ijk}\mathbf{t}_{ijk}\|^2 = \mathbf{t}_{ijk}^T A_{ijk}^T A_{ijk} \mathbf{t}_{ijk}$, by doing QR decomposition $A_{ijk} = QR$, the error term can be re-expressed as:

$$\begin{aligned} e &= \mathbf{t}_{ijk}^T A_{ijk}^T A_{ijk} \mathbf{t}_{ijk} = \mathbf{t}_{ijk}^T R^T Q^T Q R \mathbf{t}_{ijk} \\ &= \mathbf{t}_{ijk}^T R^T R \mathbf{t}_{ijk} = \mathbf{t}_{ijk}^T [R_1^T \ 0] \begin{bmatrix} R_1 \\ 0 \end{bmatrix} \mathbf{t}_{ijk} \\ &= \mathbf{t}_{ijk}^T R_1^T R_1 \mathbf{t}_{ijk} \end{aligned} \tag{10}$$

Therefore we can reduce the matrix A_{ijk} to matrix R_1, which is 27×27. Since LM algorithm requires to compute the inverse of the estimated Hessian matrix, the optimization problem with reduced matrix can be computed much faster.

3.2.3 Initial Value Computation

As we are trying to find the optimal for an optimization problem, the selection of initial position can significantly influence final performance. To obtain stability, we need close estimations for camera We decided to use essential matrix estimation in MATLAB to compute three estimations of essential matrices. As essential matrix is defined as $E = [t]_\times R$, we can compute R and t by using SVD decomposition of matrix E according to [3]. And it is already a function in MATLAB which we can use directly.

3.3. Build the Model for Optimization

3.3.1 Define the Triplets

As the minimization problem is based on a set of triplets, it can influence the optimization process significantly. Considering both efficiency and robustness, this set should contain a appropriate number of triplets which can indicate the constraints well. Therefore, we propose the following ideas to create a triplets set (Q set):

1. Determine two lower bound for inlier matches in two images and the minimum for intersections of three images to create basic Q set;

2. Q set can be iterated more than one time to add in more constraints which can contribute to the robustness of the algorithm;

3. Merge all iteration results and select final Q set.

3.3.2 Initialization and Optimization Method

Once we obtain the Q set for nonlinear optimization, there are generally three options to build the detailed model. One simple idea is to initialize all the camera at the beginning with essential matrix and then optimize them all, but the accumulated errors are unavoidable in this situation, which means it can not be used. The second method is to determine the camera poses triplet by triplet, that is, former cameras which are not in current triplet are fixed and not used for optimizing. While the third method optimizes all triplets before at every iteration.

Obviously, the third method is more reasonable since it considers all camera poses till now in every step and is more like a global idea. And it can also update or tune former cameras if they are included in the constraints for latter cameras. Therefore, the algorithm can be more robust and accurate.

In the implementation, we initialize first three cameras in the problem list at the beginning and start the optimization. Then, we initialize a batch of camera's poses each time and form them into triples according to previously generated Q-set and put them into optimization. After each optimization we repeat this step again to add more cameras. Those batch of cameras is selected along the Q-list. It is also significant, that the pose of each camera of the Q-list will be only initialized once during whole optimization process. It means that the camera will be initialized when this camera presents first time in a triple and this triple is added into optimization. After initialization, we will just use the result from last optimization for this camera but not initialize it again, even if we add another triple, which contain this camera.

But in the following procedures, camera positions are not optimized in every process, instead, we determine the number of steps automatically by the size of triplets sets.

4. Experiment and Result

We have experimented our methods on eight different datasets, where cameras are either placed in a circle or lined up in a straight line. The data typically contains 20-60 images, while some of them consists of more than 100, the number of frames for each dataset is shown in the bracket after the names. We conduct such experiments in order to test the robustness of our method. All the experiments are compared against the traditional Bundle Adjustment method provided in Colmap. The quantitative results are shown in Table 1. As we can see, the method that we used in this project performs consistently better in both rotation error and posision error on all eight datasets, which is a strong demonstration of the robustness of the GTA method we used. And we should point out, to obtain good and stable results, limits in the creation of Q set should be determined carefully.

Dataset	Method	Rotation error	Position error
De Guerre(35)	BA	2.77932	0.02144
	GTA	**2.13466**	**0.00533**
Fountain(14)	BA	1.12113	0.00290
	GTA	**0.06708**	**0.00080**
Fort.C.gate(27)	BA	0.13802	0.00331
	GTA	**0.020735**	**0.00183**
GustavIIAdolf(57)	BA	0.87351	0.00200
	GTA	**0.05270**	**0.00140**
Nijo(19)	BA	3.29930	0.01151
	GTA	**0.27376**	**0.00288**
NikolaiI(98)	BA	0.92055	0.00750
	GTA	**0.05628**	**0.00295**
Park gate(34)	BA	0.63952	0.00514
	GTA	**0.16354**	**0.00470**
UrbanII(96)	BA	0.88450	0.01469
	GTA	**0.18220**	**0.00200**

Table 1. Comparison of the two methods on different datasets

Followings are the three optimization results of the camera positions. The cameras in green is the results of our calculation and cameras in red are the ground truth. As we can see, they are pretty much coincide.

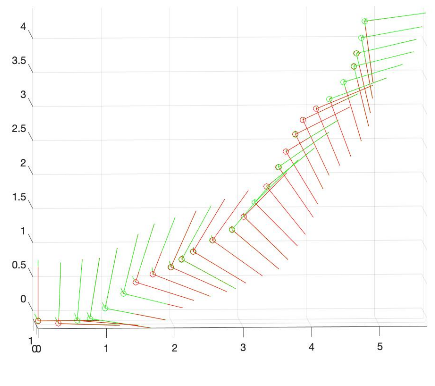

Figure 2. Camera position results of Fort.C.gate

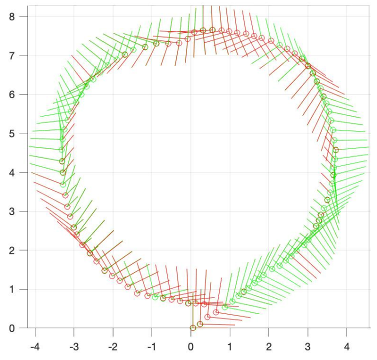

Figure 3. Camera position results of NikolaiI

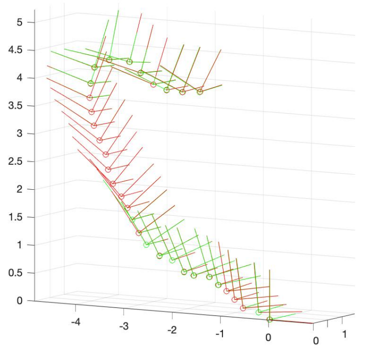

Figure 4. Camera position results of Park Gate

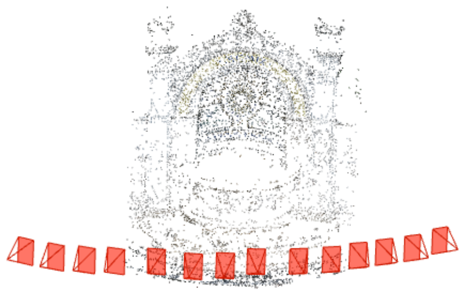

Figure 5. 3D point cloud of fountain

Figure 5 shows the reconstructed 3D points by Colmap with the known camera poses by our optimization methods. However, we should indicate that the poses results are quite unstable since feature points under this situation are far more than our experiments with former datasets and they are also really different. Therefore, the two limits of our proposed methods should be picked carefully, which means we did many experiments to obtain an appropriate pose result. We still cannot solve it till now and it remains a problem.

5. Conclusion

In this project we aim to improve the accuracy and speed of Bundle Adjustment by using trifocal tensor and LM algorithm for optimization based on the GTA methods mentioned in paper [6]. As is shown in the result section, our performance is better than Bundle Adjustment in both the estimation of rotation and projection matrices. Yet since our code is written in MATLAB, we failed to compare our speed results with Bundle adjustment. In future research, we plan to implement the full code in C++ as to compare our performance with respect to computational speed.

5.1. Statement of the Work

In this project, feature extraction and matching are executed by Colmap directly and in the second optimization part, we use external codes to calculate Jacobian matrix and trifocal tensors. The optimization model is built by ourselves, including the determination of triplets set, the strategy of switch between initialization and optimization and function for minimization problem. Besides, we also write some codes to export the data from the database provided by Colmap.

Wenhao built the first structure for our model and did the first attempt for optimization, such as, selecting specific Q set for testing; Chuqiao tested all Jacobian matrix and tensor calculation and built the first usable incremental model, she also did most of the experiments for our final model and calculate the error of bundle adjustment for the comparison. Yiming and Zilin ran the Colmap and exported data together at the begining. Yiming determined the final strategy for the creation of Q set and number of steps. Zilin structured the whole codes and finished the global optimization. Zilin and Yiming also adapted the codes for 3D points reconstruction. Besides, our supervisor Dr. Viktor and Zhaopeng helped us a lot to fix the bugs in the final optimization.

References

[1] O. Chum, J. Matas, and J. Kittler. Locally optimized ransac. In *Joint Pattern Recognition Symposium*, pages 236–243. Springer, 2003.

[2] G. Guidi, J.-A. Beraldin, and C. Atzeni. High-accuracy 3d modeling of cultural heritage: the digitizing of donatello's" maddalena". *IEEE Transactions on image processing*, 13(3):370–380, 2004.

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

[4] G. Lowe. Sift-the scale invariant feature transform. *Int. J*, 2:91–110, 2004.

[5] J. J. Moré. The levenberg-marquardt algorithm: implementation and theory. In *Numerical analysis*, pages 105–116. Springer, 1978.

[6] P. Persson and K. Åström. Global trifocal adjustment. In *Scandinavian Conference on Image Analysis*, pages 287–298. Springer, 2019.

[7] J. L. Schonberger and J.-M. Frahm. Structure-from-motion revisited. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4104–4113, 2016.

[8] J. L. Schönberger and J.-M. Frahm. Structure-from-motion revisited. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[9] G. Spreitzer, J. Tunnicliffe, and H. Friedrich. Using structure from motion photogrammetry to assess large wood (lw) accumulations in the field. *Geomorphology*, 346:106851, 2019.

[10] B. Triggs, P. F. McLauchlan, R. I. Hartley, and A. W. Fitzgibbon. Bundle adjustment—a modern synthesis. In *International workshop on vision algorithms*, pages 298–372. Springer, 1999.

[11] 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*, 179:300–314, 2012.

[12] C. Wu. Towards linear-time incremental structure from motion. In *2013 International Conference on 3D Vision-3DV 2013*, pages 127–134. IEEE, 2013.