A Survey of Large Scale Structure from Motion Using Bundle Adjustment

Mahebub Aalam Khatri

Abstract

Structure from Motion (SfM) refers to the process of obtaining the 3D structure of a scene using a collection of 2D images while simultaneously extracting the internal and external parameters of the cameras which took those images. This survey briefly explores factorization methods for SfM before moving on to a more thorough description of the modern SfM pipeline which utilizes a parameter optimization technique known as Bundle Adjustment (BA). The challenges and current research trends for many steps of the SfM pipeline are discussed. The survey concludes with an exploration of different applications of SfM in domains including robotics, large scale reconstruction from internet images, and geology, discussing the particular challenges they face and the constraints they utilize.

1 Introduction

Determining the structure of a scene is a vital task for several computer vision applications. For example, scene reconstruction methods have been used in geosciences for topographic surveying [6, 62], preservation of cultural heritage artifacts [23], environment reconstruction for robotics and augmented reality [38], and is the basis for view synthesis methods like NeRF, and Gaussian Splatting [30, 39]. While many families of methods for scene reconstruction from 2D RGB images exist including shape from shading, shape from defocus, and

more recently shape from learning [55, 61], among the most successful family of methods is known as *Structure from Motion* (SfM). In SfM, the 3D positions of *stationary* scene points are simultaneously reconstructed from a collection of 2D images along with the poses of the cameras which captured the images. SfM is similar to other related and sometimes synonymous methods such as Multiview Stereo (MVS), Photogrammetry, and Simultaneous Localization and Mapping (SLAM) which can often be seen as specific cases of SfM.

In this survey, we will briefly review factorization methods in §2. Then, we will review the basics of the optimization approach used in many modern SfM pipelines known as *Bundle Adjustment* (BA) in §3. Next, §4 will discuss several of the other steps that make up the larger SfM pipeline found in modern SfM systems. Following this discussion, we will focus on how BA has been used for large-scale reconstruction tasks using unordered collections of hundred to thousands of internet images of famous landmarks in §5. The survey will conclude with a review of other specific SfM applications in §6 including the use of SfM in SLAM for robotics applications and the use of SfM in geographic surveying.

2 Factorization Methods

Factorization methods were originally introduced in [56, 57] in the early 90s and feature an elegant single-iteration global optimization formulation for SfM. Notably, these methods assume that features have already been matched across all images and assume that internal camera parameters are already known – i.e., they assume the calibration matrices for the cameras are given. It is due to these limiting assumptions and their single-iteration formulation that factorization methods have lost the spotlight in recent literature. Compared to more modern techniques, factorization methods lack scalability, generalizability, and resilience to noise. However, they are still useful for creating estimates of scene structure and camera poses that can then used to initialize BA and are much faster for smaller scenes where the number of images and features are low. In the remainder of this section, we will

review the basics behind factorization and its extensions to non-orthographic projections.

2.1 Factorization Under Orthography

The original formulation in [56] for factorization utilized orthographic projection, assuming that variations in the depth of objects in the scene are small compared to the relative distance of the scene from the camera. In other words, the magnification for all points in the scene from all images is constant and the 2D locations of the 3D points are parallel projections onto each image. Under this assumption, the u_{fp}, v_{fp} coordinates in image f of the projection of a point p in the scene P_p can be expressed as

$$u_{fp} = \hat{i}_f^T (P_p - C_f)$$
$$v_{fp} = \hat{j}_f^T (P_p - C_f)$$

where $\hat{i}_f, \hat{j}_f$ represent the unit axes of image f from the origin of the image plane in world coordinates (effectively identifying the orientation of the camera where the third axis k_f can be determined by the cross product $\hat{i}_f \times \hat{j}_f$) and C_f is the position of the camera which captured image f. Then, we can define the set of scene points $\{P_p\}$ where $p = 1, ..., N$ and a set of camera positions $\{C_f\}$ and orientations $\{(\hat{i}_f, \hat{j}_f)\}$ where $f = 1, ..., F$. At this point, $\{(u_{fp}, v_{fp})\}$ are known to us while $\{P_p\}$, $\{C_f\}$, and $\{(\hat{i}_f, \hat{j}_f)\}$ are unknown.

We can remove the unknowns $\{C_f\}$ by setting the origin of the world coordinates at the centroid of all scene points. The registered image coordinates $\{(\tilde{u}_{fp}, \tilde{v}_{fp})\}$ are then:

$$\tilde{u}_{fp} = u_{fp} - \bar{u}_f$$
$$\tilde{v}_{fp} = v_{fp} - \bar{v}_f$$

where

$$\bar{u}_f = \frac{1}{N}\sum_{p=1}^{N} u_{fp} = \frac{1}{N}\sum_{p=1}^{N} \hat{i}_f^T(P_p - C_f) = \frac{1}{N}\hat{i}_f^T \sum_{p=1}^{N} P_p - \frac{1}{N}\sum_{p=1}^{N} \hat{i}_f^T C_f = -\hat{i}_f^T C_f$$

since $\sum_{p=1}^{N} P_p = 0$ because we are assuming the origin O is at the center of all $\{P_p\}$ and $\hat{i}_f^T C_f$ is a constant. Likewise,

$$\bar{v}_f = -\hat{j}_f^T C_f.$$

This finally means that we can express the registered image coordinates as simply:

$$\begin{bmatrix} \tilde{u}_{fp} \\ \tilde{v}_{fp} \end{bmatrix} = \begin{bmatrix} \hat{i}_f^T \\ \hat{j}_f^T \end{bmatrix} P_p.$$

Now, we can construct an *observation matrix* W:

$$\left[\begin{array}{ccc} \tilde{u}_{11} & \cdots & \tilde{u}_{1N} \\ \vdots & \ddots & \vdots \\ \tilde{u}_{F1} & \cdots & \tilde{u}_{FN} \\ \hline \tilde{v}_{11} & \cdots & \tilde{v}_{1N} \\ \vdots & \ddots & \vdots \\ \tilde{v}_{F1} & \cdots & \tilde{v}_{FN} \end{array}\right] = \left[\begin{array}{c} \hat{i}_1^T \\ \vdots \\ \hat{i}_F^T \\ \hline \hat{j}_1^T \\ \vdots \\ \hat{j}_F^T \end{array}\right] \begin{bmatrix} P_1 & \cdots & P_N \end{bmatrix}$$

$$W_{2F\times N} = R_{2F\times 3}\, S_{3\times N}$$

where R is the matrix containing all the camera orientations, and S is the matrix containing all scene coordinates. Given the sizes of matrices R and S, we know that the rank of the observation matrix W must be less than or equal to 3. However, since W is likely to contain

noise, we can create a rank-3 approximation by doing singular value decomposition (SVD) on the observation matrix and taking the leading three singular values:

$$W_{2F\times N} = U_{2F\times 3}\ \Sigma_{3\times 3}\ V^T_{3\times N}.$$

To actually factorize the SVD into our matrices R and S, we can define

$$W = RS = U(\Sigma)^{1/2}Q \mid Q^{-1}(\Sigma)^{1/2}V^T$$

where Q is an arbitrary 3×3 invertible matrix subject to the orthonormality constraints

$$\hat{i}_f^T Q Q^T \hat{i}_f = 1$$
$$\hat{j}_f^T Q Q^T \hat{j}_f = 1$$
$$\hat{i}_f^T Q Q^T \hat{j}_f = 0.$$

2.2 Extension to Perspective Projection

In 1996, [53] proposed an extension of the factorization technique to various other types of projections. Notably, they introduced a method for perspective projection where they begin with an equation describing the projection of a 4D point $\mathbf{P}_b$ in homogeneous coordinates being mapped to a 3D point in homogeneous coordinates via a 3×4 projection matrix C_a which occurs up to an unknown scale factor λ_{ab} called the projective depth:

$$\lambda_{ab}\mathbf{p}_{ab} = C_a\mathbf{P}_b,\ a = 1, \ldots, n,\ b = 1 \ldots, m.$$

Similar to [56], these projective measurements are collected into a $3n \times m$ observation matrix:

$$W \equiv \begin{bmatrix} \lambda_{11}\mathbf{p}_{11} & \cdots & \lambda_{1m}\mathbf{p}_{1m} \\ \vdots & \ddots & \vdots \\ \lambda_{n1}\mathbf{p}_{n1} & \cdots & \lambda_{nm}\mathbf{p}_{nm} \end{bmatrix} = \begin{bmatrix} C_1 \\ \vdots \\ C_n \end{bmatrix} \begin{bmatrix} \mathbf{P}_1 & \cdots & \mathbf{P}_m \end{bmatrix}.$$

And like the orthographic case, this observation matrix can now be decomposed using SVD but now up to rank 4. Since we do not have direct access to the projective depths, [53] considers the system of linear equations for camera pairs (i, j) and point p given by

$$F_{ij}\mathbf{p}_{jp}\lambda_{pj} = (\mathbf{e}_{ij} \wedge \mathbf{p}_{ip})\lambda_{ip}$$

where F_{ij} is the fundamental matrix and $\mathbf{e}_{ij}$ is the epipole. Note here that the fundamental matrix satisfies $E_{ij} = K_i^T F_{ij} K_j$ where K_i is the i'th camera's calibration matrix and E_{ij} is the essential matrix for the pair of cameras (i, j) [46]. Solving for these linear equations, the projective depths are estimated up to a global scale and substituted back into the factorization of the observation matrix into its camera and structure components.

3 Bundle Adjustment

Most modern approaches for SfM utilize a technique known as *Bundle Adjustment* (BA) for the joint optimization of 3D structure, camera motion (pose), and possibly also intrinsic camera parameters (i.e., calibration parameters). The authors of [58] which provide a review of modern BA techniques provide a concise description for BA as "a large sparse geometric parameter estimation problem" which is solved using non-linear least squares methods. Essentially, BA techniques aim to minimize what is known as the *reprojection error*. Starting off with some estimate of the scene parameters (which include camera poses, point locations, and potentially intrinsic camera parameters) computed with some other, simpler technique,

BA is the process of iteratively refining the estimates of those parameters where the loss function is defined as the difference in the position between the predicted locations of image features according to the model in each image and the actual locations of those same features within the images. More precisely, for an individual 3D feature $\mathbf{X}_j$ imaged by cameras with extrinsic and intrinsic camera parameters defined by $\mathbf{C}_i$ satisfies some predictive model $\pi_i(\mathbf{X}_j)$ which provides the estimated location of the point on the image plane of camera i. Then, given the true measurement of the point on the image plane x_{ij}, the feature prediction error for point $\mathbf{X}_j$ in camera i is

$$\Delta x_{ij}(\mathbf{C}_i, \mathbf{X}_j) \equiv x_{ij} - \pi_i(\mathbf{X}_j).$$

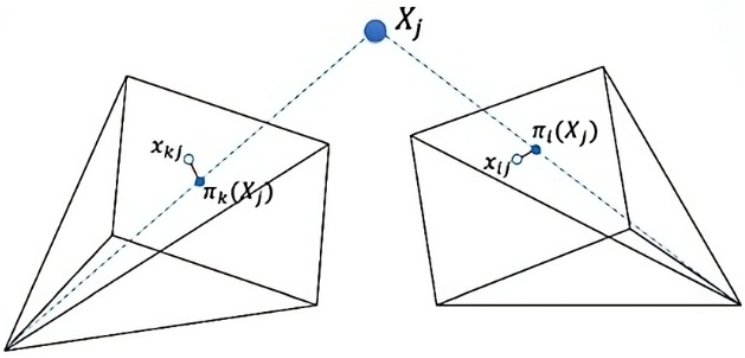

Figure 1: Bundle Adjustment aims to minimize the reprojection error – the difference between the predicted locations according to the model parameters and the actual observed locations of scene points. Figure courtesy of [65].

The feature prediction errors are then fed into some pre-determined cost function $f(x)$ for the set of current parameters x. A common choice for cost function is the sum of least squares over the reprojection errors (i.e., the L2 norm) [3]. However, [58] argue that robust, statistically-based error metrics like Maximum Likelihood (ML) and maximum a posteriori (MAP) which better handle outliers are better choices. In terms of algorithms for actually solving the non-linear least squares problem, three classes of numerical optimization techniques exist. These include second-order Newton style methods that have fast iterative convergence rates but a relatively high cost per iteration, first-order methods such as simple

gradient descent, and finally, sequential methods that begin with a smaller subset of images and incorporate more observations one-by-one instead of solving globally for the whole system [46]. In recent literature, as the focus on SfM has shifted from well structured data sets to larger, unordered data sets, the Levenberg-Marquardt algorithm has gained popularity for its ability to handle more dense connectivity graphs (see §4.2 for more information on connectivity graphs) [3, 49, 58].

3.1 The Levenberg-Marquardt Algorithm

The Levenberg-Marquardt algorithm (LM), which was originally introduced in [40], is the most popular algorithm for solving non-linear least squares in BA. The main objective of LM is to minimize the sum of squares of residuals between observed and predicted values. It iteratively adjusts the parameters of a nonlinear model to minimize the discrepancy between the model predictions and the actual data. At each iteration, the algorithm computes an update to the parameter vector by solving a modified linear least squares problem. This modification includes a damping term that adapts dynamically based on the progress of the optimization. The damping parameter controls the trade-off between the gradient descent direction and the Gauss-Newton direction, allowing the algorithm to navigate efficiently through the parameter space while ensuring convergence stability. LM combines the advantages of both the gradient descent method and the Gauss-Newton method. It is robust, efficient, and suitable for solving non-linear optimization problems with a large number of parameters. Additionally, it can handle ill-conditioned problems and is less sensitive to the choice of initial parameters compared to some other optimization techniques.

3.2 The Schur Complement Trick

For most cases, the recommended method for solving the LM algorithm is to use QR factorization. However, the special sparsity structure of BA enables the use of a method known

as the *Schur complement trick* [3]. In the LM algorithm, at each iteration, a linear system of equations involving the Hessian matrix is solved which represents the second-order derivatives of the objective function with respect to the parameters. This Hessian matrix for large-scale BA problems that have many parameters can be very sparse. The Schur complement trick exploits the sparsity structure of the Hessian matrix to decompose it into smaller, more manageable blocks. Specifically, it splits the Hessian matrix into four submatrices: the top-left block corresponding to the Gauss-Newton approximation, the bottom-right block representing the damping term, and the remaining off-diagonal blocks.

By applying the Schur complement formula, the inverse of the Hessian matrix can be expressed in terms of the inverses of the above-mentioned smaller blocks. This decomposition allows for more efficient computations, as it reduces the computational complexity of solving the linear system. In the LM algorithm, the Schur complement trick is typically used to compute the update direction of the parameters. Instead of directly inverting the entire Hessian matrix, the update direction can be computed by solving a smaller linear system involving only the Gauss-Newton approximation and the damping term. This reduces the computational cost and improves the efficiency of the optimization process. For a more complete mathematical description of the LM algorithm and the use of the Schur complement trick, the reader is directed to §2.1 and §2.2 of [3].

4 The Larger SfM Pipeline

The BA optimization process introduced in §3 is actually often the last step in a much larger SfM pipeline which encapsulates several other areas of active research. The overall pipeline can be more accurately described as consisting of three main steps: 1) feature extraction and matching, 2) camera parameter estimation, and 3) scene reconstruction, with BA being a member of this last step. In this section, we will cover some of these steps in more detail and explain the challenges and current research being done to improve the performance and

efficiency of SfM methods.

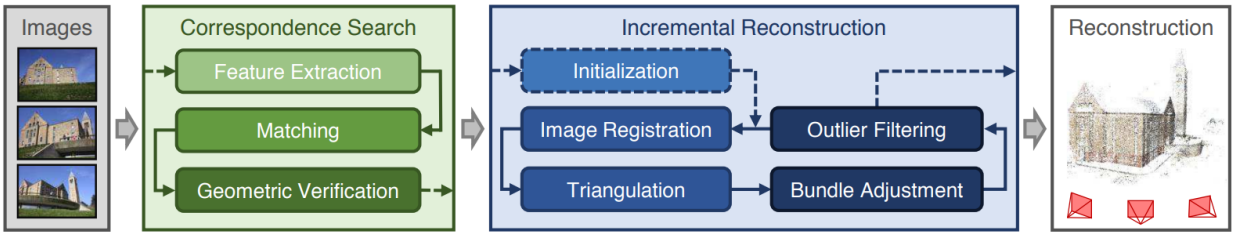

Figure 2: An example SfM pipeline. In particular, this figure represents the incremental SfM pipeline as shown in [49] in which the incremental reconstruction loop is repeated until convergence.

4.1 Feature Extraction and Matching

The first step in *any* SfM pipeline is the determination of features in images and the matching of those features between the images. The most popular and traditional technique for feature extraction is known as the *Scale Invariant Feature Transform* (SIFT) first introduced in [35, 36]. SIFT has been used as the primary feature extraction method used in a variety of popular SfM publications and packages including [9, 51, 64].

The SIFT algorithm begins by constructing a scale-space representation of the input image using Gaussian blurring and down-sampling, which helps in detecting features across different scales. It then subtracts consecutive scale space Gaussians from each other to create *Difference of Gaussian* (DoG) images (see figure 3). Local extrema in these DoG images are then identified as potential keypoints. These keypoints are then adjusted or pruned by a process known as localization where the algorithm computes the precise location and scale of each keypoint based on a Taylor series expansion of the scale-space function. During the localization process, candidates that have low contrast or are poorly localized along an edge are removed.

Once keypoints are localized, SIFT assigns an orientation to each keypoint to achieve invariance to image rotation. This is done by computing gradient magnitudes and orientations in the local neighborhood of each keypoint and selecting the dominant orientation.

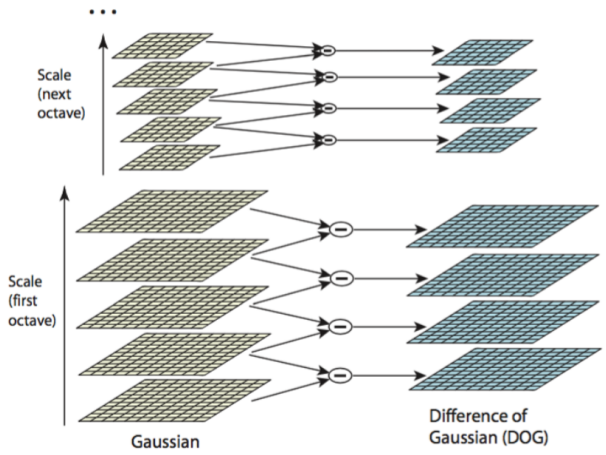

Figure 3: The scale space representation used in SIFT courtesy of [35]. The initial image is repeatedly convolved with Gaussians to produce the scale space of images. Consecutive Gaussian images are subtracted from one another to create the DoG images. Each consecutive octave is down-sampled by a factor of 2.

Keypoints are then described using a histogram of gradient orientations in their vicinity, resulting in a highly distinctive feature vector known as the SIFT descriptor (see figure 4).

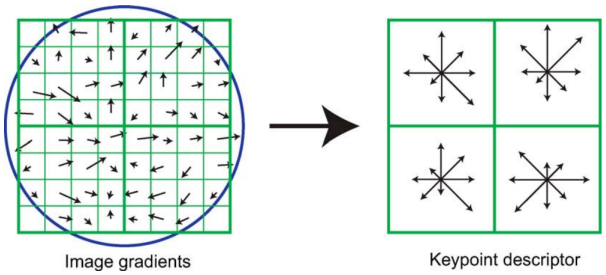

Figure 4: A keypoint descriptor is formed by computing gradient magnitude and orientation at sample points surrounding the keypoint. These values are weighted by a Gaussian window and aggregated into orientation histograms over 4x4 subregions. The length of arrows in the histograms represents the sum of gradient magnitudes in each direction within the region. Courtesy of [35].

One of the key advantages of SIFT is its robustness to variations in illumination, viewpoint changes, and partial occlusions due to its scale and rotation invariance properties. These properties make it particularly well-suited for applications like SfM, where robust feature detection and matching across multiple images are essential for accurate scene reconstruction.

While SIFT generally has good performance, its computational complexity and memory

requirements have led to various techniques which aim to improve its efficiency including simple optimizations by parallelizing the algorithm using the GPU [27]. Additionally, similar feature descriptor creation techniques like *Histogram of Gradients* (HoG) and *Speeded Up Robust Features* (SURF) have since been released which compete with SIFT in accuracy and beat it in speed [4, 11]. HoG captures the local gradient information in an image by dividing it into small cells and computing histograms of gradient orientations within each cell. These histograms are then normalized to enhance robustness against changes in illumination and contrast. The HoG descriptor effectively captures the local edge and gradient patterns, providing a compact representation of image structure. SURF on the other hand is similar to SIFT but utilizes integral images and Haar wavelets to create the feature descriptors.

More recent literature on detecting and creating feature descriptors has focused on learning image descriptors without neural networks using methods like Linear Discriminant Analysis (LDA) and Powell minimization [5], and other literature utilizing neural networks to enhance feature detection. For example, it has been shown that convolutional neural networks (CNNs) which can learn image features outperform SIFT on feature detection and even feature matching [14]. Other literature has incorporated CNNs into the larger SfM pipeline to optimize the *featuremetric* error of keypoints based on dense features learned by a CNN to achieve sub-pixel accuracy on keypoint localization and boast results with close-to-LiDAR levels of accuracy [32].

4.2 Feature Tracks and Connectivity Graphs

Typically, for larger scale SfM, we can expect that the number of scene points is several orders of magnitude higher than the number of images in the SfM instance. For example, in some literature, the data sets used have on the order of 10^3 images with 10^6 3D structure points [19] while in others, the numbers are in the massive order of 10^6 images with 10^9 3D feature points [2]. Therefore, a key challenge in SfM is to efficiently process the data. A common strategy is to initialize BA with a small subset of the images and then iteratively

add features from additional images. However, this requires the creation of *feature tracks* which are a collection of images that share the same features, and *connectivity graphs* that indicate which images share features. For structured datasets (e.g., images that have been captured from a moving truck), we expect the connectivity graph to be very sparse and feature a band-diagonal structure which can be exploited. However, for unstructured data sets like community photo collections of, say, famous tourist locations where many images are taken from similar locations of the same objects, we can expect the connectivity graph to be far more dense (see figure 5) [3].

(a) Structured - 6375 photos (b) Unstructured - 4585 photos

Figure 5: (a) A sparse connectivity graph for a structured data set taken from a moving truck. (b) A comparatively dense connectivity graph for an unstructured data set taken from a community photo collection. Figure courtesy of [3].

The creation of the connectivity graphs in themselves can be extremely computationally expensive. Even in the simpler case of 10^3 images each with an order of 10^4 features, a brute force comparison of image features for matching would require 10^7 feature descriptor comparisons. By some accounts, image matching can occupy *half* of the computational cost of the entire SfM pipeline and inaccurate matches can cause failures in the reconstruction [7]. Solutions to increase the efficiency of determining connectivity graphs include incorporating Bag of Words (BoW) methods which create histograms of the feature descriptors and compare those histograms instead of the feature descriptors themselves. One popular addition to BoW methods is *Total Recall* which was proposed in [8] which utilizes *query expansion* with BoW. This effectively adds other relevant visual features to the BoW search criteria, enabling more robust feature matching. Other approaches create vocabulary trees structured as Kd-trees to improve the lookup of matching images features [44]. In production libraries

for SfM, the preferred method for image matching is to use a cascade hashing approach as described in [7]. Cascade hashing creates hash tables with extremely fast lookups that can be created without needing to train on the data. This process can be seen as similar to the construction of Kd-trees since they utilize multiple hashing stages which iteratively refine the hash codes. This method has been shown to be up to around 300 times faster than brute force methods and up to 10 times faster than Kd-tree methods [7].

4.3 Geometric Verification

Since feature matching is solely based on appearance, there is no guarantee that corresponding features actually map to the same point in the scene. Therefore, many SfM pipelines incorporate a geometric verification step which validates image matches using projective geometry. In particular, a homography matrix $\mathbf{H}$ can be calculated using a simple least squares approach which describes a pure transformation consisting of a translation and rotation from the image coordinates of one frame to another. If there exists a valid homography transformation which maps a sufficient number of features between two matching candidate images, then they are geometrically verified [49]. To search for such a homography matrix, a technique for filtering noisy correspondences is used known as *Random Sample Consensus* (RANSAC) [15]. As the name suggests, RANSAC begins by, 1) choosing s random samples. Then, 2) it uses those s samples to do a least-squares fit for the homography matrix. Note that s is often chosen to be the minimum number of samples needed to fit the homography matrix. Once the least squares fit is computed, 3) the model is applied to all correspondence points and the number of inliers m are determined which lie within some predefined threshold. Next, steps 1 to 3 are repeated for n iterations. Finally, the model with the highest number of inliers is chosen and the model is optionally re-fit to those inliers.

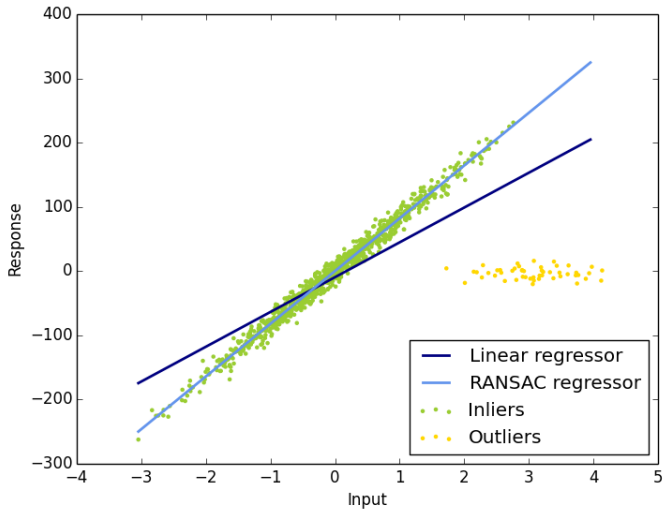

Figure 6: Random Sample Consensus (RANSAC) is utilized in the geometric verification step to increase the estimated model's redundancy to noise. Compared to linear least squares fit, RANSAC chooses a model that maximizes the number of inliers.

4.4 Initialization

Initialization for BA is important since it is non-convex. Consequently, poor initialization can cause the optimization process to get stuck in a local minima. The particular method chosen for initialization is generally dependent on the particular structure of a given SfM instance. For example, [49] aims to create a general SfM pipeline so they initialize with a carefully selected two-view reconstruction. To improve robustness, accuracy, and performance, the seed location is chosen to be at a dense location in the scene graph – i.e., in a location with many overlapping camera views. However, if lower run-times are desired, a sparser location can be chosen which in turn will mean more iterations of the BA optimization will be sparse. The rest of the scene is then reconstructed by sequentially incorporating more images, either one-by-one or in small groups.

On the other hand, [52] suggest the creation of a *skeletal set* in a similar vein to [49] of initializing with a small subset of the images then sequentially adding more images. However, instead of just two views, the skeletal set method allows an initial reconstruction with an accuracy that approximates the complete set by choosing views from the connectivity graph

that effectively maximize the number of features per image. The remaining images can then be incorporated by estimating their pose relative to the existing detected features in the initial reconstruction and a final BA step can be used to improve the accuracy of the reconstruction.

The main idea behind the construction of the skeletal set is to identify a directed measure of proximity between cameras in the image graph given by the translational uncertainty in each pairwise reconstruction of cameras [52]. The proximity measure is quantified by the trace of the covariance matrix of the Schur complement of the camera pair. Given this measure of proximity, the goal is to identify the smallest subset of cameras such that the length of the shortest path (in terms of total uncertainty of the path) between any pair of cameras is at most some pre-determined stretch factor t times longer than their distance in the original image graph. Note that the image graph mentioned here is similar to the connectivity graphs mentioned above, except that the edges are now weighted directed edges with the weights set by the trace of the covariance matrices as mentioned earlier. When edges in this graph are removed to create the skeletal set, the lengths of the shortest paths between two nodes can increase (i.e., the sum of the weighted edges of the shortest path) and the goal is to remove edges such that the increase in this path length is minimized. Unfortunately, this process is NP-complete, so a two-step heuristic is used to approximate the skeletal set. This method has been shown to significantly increase the overall efficiency and accuracy of SfM reconstructions compared to pre-existing SfM methods that were applied on the entire set of images. For more details on skeletal sets and their construction, the reader is directed to [52].

4.5 Image Registration and Triangulation

As mentioned, since many SfM pipelines utilize sequential reconstruction techniques, a challenge that arises is registering new images to the current model. This problem is also known as the *Perspective-n-Point* (PnP) problem and is solved by utilizing feature correspondences

to existing triangulated points from images that have already been registered. Lepetit et. al. [31] propose a non-iterative solution to the PnP problem called EPnP with a time complexity of $O(n)$ (compared to other techniques which had a complexity of $O(n^5)$ or even $O(n^8)$). Their method, which is applicable to all PnP problems where $n \geq 4$, expresses the n 3D points as a weighted sum of just four *virtual* control points. Then, the problem effectively reduces to estimating the coordinates of the control points in the camera referential. This can be done in $O(n)$ time by expressing the control point coordinates as the weighted sum of eigenvectors of a twelve by twelve matrix and solving a few quadratic equations to choose the right weights. Other methods such as [20] utilize RANSAC (see §4.4) and a minimal pose solver to make pose estimates from outlier-contaminated 2D-3D correspondences.

Once a new image is registered to the existing points, new scene points can be triangulated as long as at least one other registered image in the set from a different viewpoint contains those scene points. Multiple methods for optimal triangulation exist. For example, [29] propose a robust multi-view L_2 triangulation approach based on optimal inlier selection and 3D structure refinement. However, for production systems, [49] propose a novel method for handling outlier contaminated feature tracks. They argue that for sparsely matched image collections, transitive correspondences can boost triangulation performance. Most matching techniques favor images that highly overlap and are visually similar in appearance, leading to correspondences with a small baseline. Larger baselines can be achieved by utilizing transitivity which in turn can enable better triangulation. Triangulation is the last step in a typical SfM pipeline before iterative BA is performed to optimize the initial estimates created by the steps up to this point (see §3).

5 BA with Large Unordered Collections

A significant portion of BA literature has focused on expanding BA methods to large scale, unordered collections of internet images of famous tourist locations and even entire cities [2,

Figure 7: Four different views of the city of Dubrovnik with the corresponding view of the reconstruction. This reconstruction consisted of 4,585 images and 2,662,981 3D points with 11,839,682 observed features. Figure courtesy of [2].

3, 17, 19, 22, 43, 47]. Such large scale reconstructions have several commercial applications in entertainment for reconstruction of cities used in movies, digital mapping for services like Google Maps (including interactive reconstructions of popular tourist destinations), urban planning, and training in simulation for civil services that benefit from a virtual urban recreation like fire and police departments [43].

These sorts of data sets pose several unique challenges due to the sheer volume, heterogeneity, and unstructured nature of the data. Firstly, the scale of such datasets can be enormous, making traditional BA techniques computationally prohibitive. Additionally, internet images often vary widely in terms of quality, resolution, viewpoint, lighting conditions, and occlusions, leading to inconsistent feature detection and matching. Furthermore, internet images may lack metadata or accurate camera calibration information, complicating the initialization and optimization stages of BA. Moreover, the unordered nature of the image collection poses difficulties in establishing reliable correspondences across images, as traditional sequential approaches for BA may not be applicable. Consequently, robust feature matching, outlier rejection, and efficient optimization algorithms tailored to handle large-scale, unordered datasets are necessary to overcome these challenges and achieve accurate scene reconstructions. Many of the extensions to traditional BA methods discussed in §3 and §4 were specifically created to handle the exacerbated difficulties of large-scale unordered

data. For instance, many large scale SfM packages utilize the BoW and query expansion methods for feature matching and use parallelized variants of SIFT on GPUs [8, 27].

6 Other SfM Applications

To wrap up this survey, we wish to introduce a few related SfM applications along with their unique challenges and the specific constraints those applications can utilize.

6.1 Visual SLAM

The Visual Simultaneous Localization and Mapping (SLAM) problem in robotics involves the *real-time* estimation of a robot's trajectory and map of its environment using visual data obtained from its cameras. This task is essential for autonomous navigation as it allows robots to localize themselves within unknown environments while simultaneously building a map of their surroundings. It can also be useful in AR applications which require knowledge of the user's surroundings to overlay virtual objects. The visual SLAM problem has been studied extensively by the robotics community, particularly as a cheaper alternative to other SLAM methods such as using RGB-D cameras that incorporate time-of-flight sensors or Light Detection and Ranging (LiDAR) [12, 18, 38, 54].

Like the SfM pipeline in computer vision, Visual SLAM involves several, similar steps. First, feature detection and tracking are performed using methods like SIFT mentioned above to identify distinctive points or landmarks in the visual data and track their movements across frames. Next, correspondences between these features are established to estimate the relative motion between consecutive frames. Once the relative motion is estimated, the robot's pose is updated using BA. One distinction between SLAM and large-scale SfM is that the connectivity graph for SLAM image features will necessarily be more sparse and structured since images are captured from a single moving source. Additionally, SLAM does not require the computation of intrinsic camera parameters since those parameters are often

given or can be separately determined.

Despite similarities to the SfM pipeline, Visual SLAM introduces additional challenges due to the real-time nature of robotics applications and the need to handle sensor noise, uncertainties in motion estimation, and dynamic environments. Therefore, Visual SLAM algorithms often employ techniques such as Kalman filtering, particle filtering, or optimization-based methods to robustly estimate the robot's pose and map while accounting for these challenges [38].

6.2 SfM in Geosciences

Many geoscience applications require the collection of topographic data such as terrain mapping, geomorphological terrain analysis, monitoring of geological features, and digital elevation modelling. Traditional methods for acquiring topographic data such as terrestrial laser scanning (TLS) or LiDAR are often prohibitively expensive and are not conducive to remote or otherwise hostile locations. For these reasons, SfM has been proposed as a low-cost, portable tool for acquiring topographic data [6, 16, 62].

Similar to the application of SfM in visual SLAM, images for geoscience applications are often taken by just a few ground-based cameras capturing panoramic views or by a relatively cheap drone or Unmanned Aerial Vehicle (UAV) both of which can take highly structured images of a target landscape. Also, like in SLAM, the intrinsic camera parameters are usually known or can be determined. Another constraint that can be used for geoscience SfM is GPS localization, i.e., a fairly precise GPS location is often known for captured images which can help increase the accuracy of reconstructions. Finally, for accessible landscapes, clearly identifiable control points with precisely known positions can be used to constrain the reconstruction. Often these control points are physically placed markers on the landscape and can be easily distinguished by having an identifiable color compared to the rest of the landscape (see figure 8)[62].

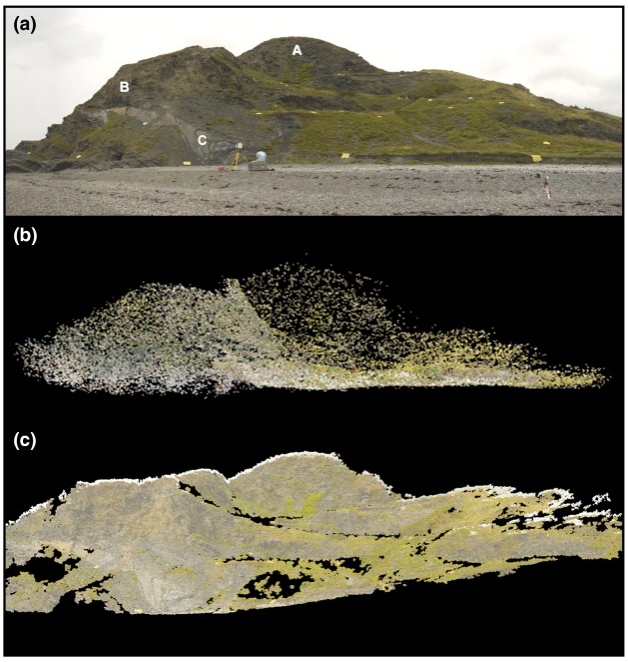

Figure 8: Views of *Constitution Hill* showcasing the use of SfM in geoscience. a) A panoramic image of the survey location with clearly identifiable yellow control points. b) A sparse point cloud reconstruction of the scene. c) A dense point cloud reconstruction. Figure courtesy of [62].

7 Conclusion

In this survey, we briefly reviewed the factorization method for structure from motion and its extensions to projective projections, explained the limitations of factorization methods, then introduced bundle adjustment, a non-linear parameter estimation problem which is the primary optimization technique used in modern SfM pipelines. Then, we reviewed the key steps in the SfM pipeline and provided brief descriptions of the common techniques used in each step and some of the ongoing research being done to improve the performance or efficiency of those steps. Finally, we discussed specific applications of SfM in large scale reconstructions from unordered collections, a technique similar to SfM known as visual SLAM which is used in robotics and augmented reality applications, and SfM for topographic measurements in geoscience applications.

Extrapolating from the current state of research in SfM, we predict that there are two main areas in which SfM will continue to grow: speed and accuracy. The general SfM

pipeline has been established for nearly two decades and the research since has focused on generalizing SfM to handle larger and more complex data sets [2, 3, 19, 26] or to creating ever finer reconstructions [10, 32]. Therefore a prediction that this will continue to be the trend is fairly obvious.

Besides the obvious expected improvements in speed and accuracy, another area in which we expect future research to be conducted is in task-specific SfM. By this, we are referring instances where specific constraints can be utilized to simplify the SfM problem. This is opposed to the current trend of ever-increasing generalizability that has been the focus of recent research (e.g., [49]). For example, in augmented reality or virtual reality applications, the camera motion is often known from other, non-visual sources such as Inertial Measurement Units (IMUs) which can constrain SfM to real-time depth estimation.

References

[1] Sameer Agarwal, Noah Snavely, and Steven M Seitz. "Fast algorithms for L_∞ problems in multiview geometry". In: *2008 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE. 2008, pp. 1–8.

[2] Sameer Agarwal et al. "Building rome in a day". In: *Communications of the ACM* 54.10 (2011), pp. 105–112.

[3] Sameer Agarwal et al. "Bundle adjustment in the large". In: *Computer Vision–ECCV 2010: 11th European Conference on Computer Vision, Heraklion, Crete, Greece, September 5-11, 2010, Proceedings, Part II 11*. Springer. 2010, pp. 29–42.

[4] Herbert Bay et al. "Speeded-up robust features (SURF)". In: *Computer vision and image understanding* 110.3 (2008), pp. 346–359.

[5] Matthew Brown, Gang Hua, and Simon Winder. "Discriminative learning of local image descriptors". In: *IEEE transactions on pattern analysis and machine intelligence* 33.1 (2010), pp. 43–57.

[6] Jonathan L Carrivick, Mark W Smith, and Duncan J Quincey. *Structure from Motion in the Geosciences*. John Wiley & Sons, 2016.

[7] Jian Cheng et al. "Fast and accurate image matching with cascade hashing for 3d reconstruction". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2014, pp. 1–8.

[8] Ondrej Chum et al. "Total recall: Automatic query expansion with a generative feature model for object retrieval". In: *2007 IEEE 11th International Conference on Computer Vision*. IEEE. 2007, pp. 1–8.

[9] David Crandall et al. "Discrete-continuous optimization for large-scale structure from motion". In: *CVPR 2011*. IEEE. 2011, pp. 3001–3008.

[10] Hainan Cui et al. "HSfM: Hybrid structure-from-motion". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 1212–1221.

[11] Navneet Dalal and Bill Triggs. "Histograms of oriented gradients for human detection". In: *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*. Vol. 1. Ieee. 2005, pp. 886–893.

[12] Andrew J Davison et al. "MonoSLAM: Real-time single camera SLAM". In: *IEEE transactions on pattern analysis and machine intelligence* 29.6 (2007), pp. 1052–1067.

[13] Frank Dellaert et al. "Structure from motion without correspondence". In: *Proceedings IEEE Conference on Computer Vision and Pattern Recognition. CVPR 2000 (Cat. No. PR00662)*. Vol. 2. IEEE. 2000, pp. 557–564.

[14] Philipp Fischer, Alexey Dosovitskiy, and Thomas Brox. "Descriptor matching with convolutional neural networks: a comparison to sift". In: *arXiv preprint arXiv:1405.5769* (2014).

[15] Martin A Fischler and Robert C Bolles. "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography". In: *Communications of the ACM* 24.6 (1981), pp. 381–395.

[16] Mark A Fonstad et al. "Topographic structure from motion: a new development in photogrammetric measurement". In: *Earth surface processes and Landforms* 38.4 (2013), pp. 421–430.

[17] Jan-Michael Frahm et al. "Building rome on a cloudless day". In: *Computer Vision–ECCV 2010: 11th European Conference on Computer Vision, Heraklion, Crete, Greece, September 5-11, 2010, Proceedings, Part IV 11*. Springer. 2010, pp. 368–381.

[18] Jorge Fuentes-Pacheco, José Ruiz-Ascencio, and Juan Manuel Rendón-Mancha. "Visual simultaneous localization and mapping: a survey". In: *Artificial intelligence review* 43 (2015), pp. 55–81.

[19] Yasutaka Furukawa et al. "Towards internet-scale multi-view stereo". In: *2010 IEEE computer society conference on computer vision and pattern recognition*. IEEE. 2010, pp. 1434–1441.

[20] Xiao-Shan Gao et al. "Complete solution classification for the perspective-three-point problem". In: *IEEE transactions on pattern analysis and machine intelligence* 25.8 (2003), pp. 930–943.

[21] Steffen Gauglitz, Tobias Höllerer, and Matthew Turk. "Evaluation of interest point detectors and feature descriptors for visual tracking". In: *International journal of computer vision* 94 (2011), pp. 335–360.

[22] Michael Goesele et al. "Multi-view stereo for community photo collections". In: *2007 IEEE 11th International Conference on Computer Vision*. IEEE. 2007, pp. 1–8.

[23] Gabriele Guidi, J-A Beraldin, and Carlo Atzeni. "High-accuracy 3D modeling of cultural heritage: the digitizing of Donatello's" Maddalena"". In: *IEEE Transactions on image processing* 13.3 (2004), pp. 370–380.

[24] Richard Hartley and Andrew Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003.

[25] Richard I Hartley. "In defense of the eight-point algorithm". In: *IEEE Transactions on pattern analysis and machine intelligence* 19.6 (1997), pp. 580–593.

[26] Michal Havlena, Akihiko Torii, and Tomáš Pajdla. "Efficient structure from motion by graph optimization". In: *Computer Vision–ECCV 2010: 11th European Conference on Computer Vision, Heraklion, Crete, Greece, September 5-11, 2010, Proceedings, Part II 11*. Springer. 2010, pp. 100–113.

[27] S Heymann et al. "SIFT implementation and optimization for general-purpose GPU". In: (2007).

[28] Takeo Kanade and Daniel D Morris. "Factorization methods for structure from motion". In: *Philosophical Transactions of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences* 356.1740 (1998), pp. 1153–1173.

[29] Lai Kang, Lingda Wu, and Yee-Hong Yang. "Robust multi-view L2 triangulation via optimal inlier selection and 3D structure refinement". In: *Pattern Recognition* 47.9 (2014), pp. 2974–2992.

[30] Bernhard Kerbl et al. “3D Gaussian Splatting for Real-Time Radiance Field Rendering”. In: *ACM Transactions on Graphics* 42.4 (2023).

[31] Vincent Lepetit, Francesc Moreno-Noguer, and Pascal Fua. “EPnP: An accurate O(n) solution to the PnP problem”. In: *International journal of computer vision* 81 (2009), pp. 155–166.

[32] Philipp Lindenberger et al. “Pixel-perfect structure-from-motion with featuremetric refinement”. In: *Proceedings of the IEEE/CVF international conference on computer vision*. 2021, pp. 5987–5997.

[33] Sikang Liu et al. “Efficient SfM for Large-Scale UAV Images Based on Graph-Indexed BoW and Parallel-Constructed BA Optimization”. In: *Remote Sensing* 14.21 (2022).

[34] H Christopher Longuet-Higgins. “A computer algorithm for reconstructing a scene from two projections”. In: *Nature* 293.5828 (1981), pp. 133–135.

[35] David G Lowe. “Distinctive image features from scale-invariant keypoints”. In: *International journal of computer vision* 60 (2004), pp. 91–110.

[36] David G Lowe. “Object recognition from local scale-invariant features”. In: *Proceedings of the seventh IEEE international conference on computer vision*. Vol. 2. Ieee. 1999, pp. 1150–1157.

[37] Yi Ma, Jana Košecká, and Shankar Sastry. “Optimization criteria and geometric algorithms for motion and structure estimation”. In: *International Journal of Computer Vision* 44 (2001), pp. 219–249.

[38] Andréa Macario Barros et al. “A comprehensive survey of visual slam algorithms”. In: *Robotics* 11.1 (2022), p. 24.

[39] Ben Mildenhall et al. “Nerf: Representing scenes as neural radiance fields for view synthesis”. In: *Communications of the ACM* 65.1 (2021), pp. 99–106.

[40] Jorge J Moré. “The Levenberg-Marquardt algorithm: implementation and theory”. In: *Numerical analysis: proceedings of the biennial Conference*. Springer. 1977, pp. 105–116.

[41] Etienne Mouragnon et al. “Generic and real-time structure from motion using local bundle adjustment”. In: *Image and Vision Computing* 27.8 (2009), pp. 1178–1193.

[42] Raul Mur-Artal, Jose Maria Martinez Montiel, and Juan D Tardos. “ORB-SLAM: a versatile and accurate monocular SLAM system”. In: *IEEE transactions on robotics* 31.5 (2015), pp. 1147–1163.

[43] Przemyslaw Musialski et al. “A survey of urban reconstruction”. In: *Computer graphics forum*. Vol. 32. 6. Wiley Online Library. 2013, pp. 146–177.

[44] David Nister and Henrik Stewenius. “Scalable recognition with a vocabulary tree”. In: *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR’06)*. Vol. 2. Ieee. 2006, pp. 2161–2168.

[45] David Nistér. “An efficient solution to the five-point relative pose problem”. In: *IEEE transactions on pattern analysis and machine intelligence* 26.6 (2004), pp. 756–770.

[46] Onur Özyeşil et al. “A survey of structure from motion*.” In: *Acta Numerica* 26 (2017), pp. 305–364.

[47] Marc Pollefeys et al. “Detailed real-time urban 3d reconstruction from video”. In: *International Journal of Computer Vision* 78 (2008), pp. 143–167.

[48] Rahul Raguram, Jan-Michael Frahm, and Marc Pollefeys. “A comparative analysis of RANSAC techniques leading to adaptive real-time random sample consensus”. In: *Computer Vision–ECCV 2008: 10th European Conference on Computer Vision, Marseille, France, October 12-18, 2008, Proceedings, Part II 10*. Springer. 2008, pp. 500–513.

[49] Johannes L Schonberger and Jan-Michael Frahm. “Structure-from-motion revisited”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 4104–4113.

[50] Sudipta N. Sinha, Drew Steedly, and Richard Szeliski. “A Multi-stage Linear Approach to Structure from Motion”. In: *Trends and Topics in Computer Vision*. Ed. by Kiriakos N. Kutulakos. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 267–281.

[51] Noah Snavely, Steven M Seitz, and Richard Szeliski. “Photo tourism: exploring photo collections in 3D”. In: *ACM siggraph 2006 papers*. 2006, pp. 835–846.

[52] Noah Snavely, Steven M. Seitz, and Richard Szeliski. “Skeletal graphs for efficient structure from motion”. In: *2008 IEEE Conference on Computer Vision and Pattern Recognition*. 2008, pp. 1–8.

[53] Peter Sturm and Bill Triggs. “A factorization based algorithm for multi-image projective structure and motion”. In: *Computer Vision—ECCV’96: 4th European Conference on Computer Vision Cambridge, UK, April 15–18, 1996 Proceedings Volume II 4*. Springer. 1996, pp. 709–720.

[54] Takafumi Taketomi, Hideaki Uchiyama, and Sei Ikeda. “Visual SLAM algorithms: A survey from 2010 to 2016”. In: *IPSJ Transactions on Computer Vision and Applications* 9.1 (2017), pp. 1–11.

[55] Zachary Teed and Jia Deng. “Deepv2d: Video to depth with differentiable structure from motion”. In: *arXiv preprint arXiv:1812.04605* (2018).

[56] Carlo Tomasi and Takeo Kanade. “Shape and motion from image streams under orthography: a factorization method”. In: *International journal of computer vision* 9 (1992), pp. 137–154.

[57] Carlo Tomasi and Takeo Kanade. “Shape and motion without depth”. In: *Proceedings of the DARPA Image Understanding Workshop*. 1990, p. 258.

[58] Bill Triggs et al. “Bundle adjustment—a modern synthesis”. In: *Vision Algorithms: Theory and Practice: International Workshop on Vision Algorithms Corfu, Greece, September 21–22, 1999 Proceedings*. Springer. 2000, pp. 298–372.

[59] Tinne Tuytelaars and Krystian Mikolajczyk. “Local Invariant Feature Detectors: A Survey”. In: *Foundations and Trends in Computer Graphics and Vision* 3.3 (2008), pp. 177–280.

[60] Shimon Ullman. "The interpretation of structure from motion". In: *Proceedings of the Royal Society of London. Series B. Biological Sciences* 203.1153 (1979), pp. 405–426.

[61] Sudheendra Vijayanarasimhan et al. *SfM-Net: Learning of Structure and Motion from Video*. 2017.

[62] Matthew J Westoby et al. "'Structure-from-Motion'photogrammetry: A low-cost, effective tool for geoscience applications". In: *Geomorphology* 179 (2012), pp. 300–314.

[63] Changchang Wu. "Towards linear-time incremental structure from motion". In: *2013 International Conference on 3D Vision-3DV 2013*. IEEE. 2013, pp. 127–134.

[64] Changchang Wu et al. "VisualSFM: A visual structure from motion system". In: (2011).

[65] Yang Yu et al. "Multi-view 2D–3D alignment with hybrid bundle adjustment for visual metrology". In: *The Visual Computer* 38 (Apr. 2022).

[66] Zhengyou Zhang. "Determining the Epipolar Geometry and its Uncertainty: A Review". In: *International Journal of Computer Vision* 27 (2 1998).