A Deep Learning Method for Frame Selection in Videos for Structure from Motion Pipelines

Francesco Banterle[1], Rui Gong[2], Massimiliano Corsini[1], Fabio Ganovelli[1], Luc Van Gool[1], and Paolo Cignoni[2]

[1]ISTI-CNR, Italy
[2]ETH Zürich, Switzerland

Abstract

Structure-from-Motion (SfM) using the frames of a video sequence can be a challenging task because there is a lot of redundant information, the computational time increases quadratically with the number of frames, there would be low-quality images (e.g., blurred frames) that can decrease the final quality of the reconstruction, etc. To overcome all these issues, we present a novel deep-learning architecture that is meant for speeding up SfM by selecting frames using predicted subsampling frequency. This architecture is general and can learn/distill the knowledge of any algorithm for selecting frames from a video for generating high-quality reconstructions. One key advantage is that we can run our architecture in real-time saving computations while keeping high-quality results.

Keywords: Structure from Motion, Deep Learning, Point-Cloud Generation, Video Processing

1 Introduction

The last decade has seen an outburst of methods and software (both free and commercial) for computing 3D reconstructions from sets of images using the so-called Structure-from-Motion or SfM [25] paradigm. Since video sequences became easier to acquire, a straightforward pipeline consists of simply feeding all the frames of a video to the SfM algorithms. However, SfM software is specialized to produce high-quality results for a collection of sparse images, and it turns out it performs poorly when videos are employed. There are two reasons for this. One is simply that computational memory and time requirements grow quadratically with the number of images. The second reason is that frames from a video make a poorly conditioned dataset for SfM reconstruction. By design, these methods need a set

of images that overlap enough so that every point of the scene is seen by three or more images and that there is enough parallax among these images so that the computation of the position of the point in 3D can be accurate. If we selected all the frames from a video, we would generally have a highly redundant dataset where the overlap is too much and the parallax between images with matching features is too little. One effective approach is to take a subset of those frames and run SfM over it. The challenge consists in choosing an optimal subset, a task that an experienced practitioner could do reasonably well, provided that enough time is given.

We observed that there is not a specific optimal subset of frames as much as many possible equivalent choices of subsets. For example, if we consider a video shot from a camera moving at a constant speed, the set of even frames $(0, 2, 4, \ldots)$ and the set of odd frames $(1, 3, 5, \ldots)$ would produce essentially the same final result. Therefore, we looked for ways to encode the notion of a set of subsets and choose to do it in terms of *local frequency*, that is, the sampling factor to use for selecting frames from a given segment of the video. We designed a deep-based framework that takes as input a video and calculates, for each segment of the video, how many frames to regularly sample from that segment in the reconstruction algorithm. In the training phase, we fed the network with a large number of possible subsets of frames of videos. For each subset, we ran an SfM reconstruction, computing the loss function in terms of accuracy of the reconstruction w.r.t to the ground truth. In the online phase, given the input video to the network, the network returns, for each consecutive segment of frames, the sampling factor to use to form the set to input to the SfM reconstruction. To summarize, our contribution is: (i) a novel architecture for distilling any algorithm for selecting high-quality sets of frames for 3D reconstruction; (ii) the evaluation of this architecture using a greedy method; and (iii) a dataset of videos for 3D reconstruction using SfM pipelines.

2 Related Work

Related Work on SfM using DL. Our work is related to the works on exploiting the deep learning techniques to improve the accuracy and the speed of the structure-from-motion (SfM) pipeline. The SfM task [7, 24] aims at reconstructing the 3D structure of the scene or objects from multiple 2D images. There are two key aspects for solving the SfM problem, camera motion extraction, and structure estimation. To improve the whole SfM performance, the traditional SfM works have explored well on improving both the camera motion and structure estimation and the connection between the camera motion and the structure estimation. For example, some works [4] focus on improving the robustness and the effectiveness of the low-level feature extraction and matching to improve both the camera motion

and structure estimation performance. Some other works [24, 1, 6] pay more attention to optimizing the camera motion and the structure estimation at the same time, by exploiting the connection between the motion and structure. The traditional SfM methods have already made great progress in the past decades and proven the effectiveness for the real applications on different tasks such as the scene reconstruction [1, 9, 17, 18, 30], and the 3D object modeling [20]. However, there still exist problems that need to be further solved such as time efficiency. Recently, deep learning techniques have made a great impact in the field of computer vision and show an advantage in accuracy and efficiency[16].More recently, more and more works are exploring to exploit the deep learning techniques to help improve the SfM task [27, 14, 28] performance on efficiency and accuracy. When applied to the SfM task, the advantage of the deep learning-based techniques is proven on the efficiency [28], compared with the traditional SfM methods. However, the disadvantage is also found out for the low robustness and accuracy under varying environment [29], due to the high reliability of deep learning on the image data distribution which makes the deep model hard to be generalized to different settings. To take the advantage of the deep-based techniques on the efficiency and the advantage of the traditional SfM techniques on robustness and accuracy, our work exploits the deep learning-based module to preprocess the video frames to select well-conditioned sets of frames w.r.t. to feed to the traditional SfM pipeline to speed up the reconstruction preserving the accuracy.

Related Work on Frame Selection for SfM. Our work is related to the works on frame selection for SfM works. One important aspect, which limits the time efficiency and accuracy for the traditional SfM pipeline, is the redundant and low-quality frames feed into the pipeline [22]. To improve the efficiency and accuracy of the traditional SfM pipeline, more and more works explore to select the keyframes from the input videos, to reduce the noise for the optimization and reduce the computational load. Some works [26, 23] select the keyframes in the process of the optimization such as the bundle adjustment to remove the noise for the matrix decomposition. However, these works still need to process all the frames during the feature extraction and matching stage, which still cost much time on the redundant and low-quality frames. Besides, some works [22, 21] propose to preprocess the input videos and select the keyframes according to the quality of the frame, which is more time-efficient. However, these works can only remove the low-quality frames according to prior knowledge but still cannot reduce the redundancy. Our work also follows the keyframe selection for preprocessing the input video. However, different from the previous works, our module learns to sample the keyframes from the video segments adaptively and automatically, to only sample the frames which are essential for the final reconstruction to highly reduce the redundancy and improve the efficiency.

Related Work on Distilling Knowledge. Our work is also related to works on knowledge distillation. Knowledge distillation is an important topic in deep learning [10, 12]; it enables to have more efficient networks while transferring knowledge from a *teacher* network to a smaller *student* one; which is typically a simplified model. This approach can be applied to computationally cumbersome algorithms such as perceptual metrics [2, 3]. We share a similar philosophy with these works on knowledge transferring and distillation. However, our work does not transfer and distill knowledge between the different teacher and student networks. More specifically, we want to distill the frames which are key for the final 3D reconstruction, to reduce the redundancy and improve time efficiency. To our knowledge, this work is the first one to propose a general framework for distilling any frame selection algorithm into a generic deep learning-based architecture.

3 Method

Our goal is to estimate the frequency, f, for sampling frames to generate high-quality 3D models from videos without manual intervention to determine the frames to be selecting. This sets users free from inspecting thousands of frames during the 3D reconstruction step.

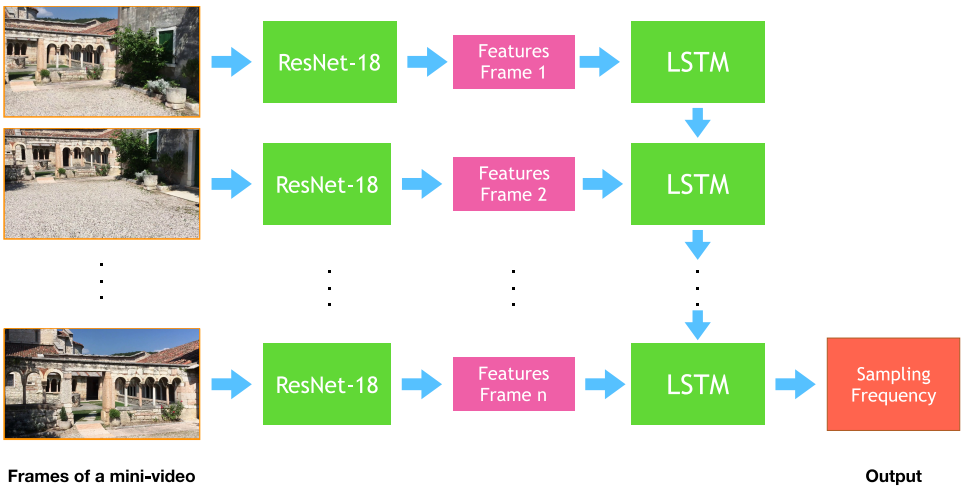


Figure 1: Our architecture for predicting the sampling frequency of frames for SfM.

As for the model architecture, we combined a convolutional neural network (CNN) [15] with a long-short term memory (LSTM) network [11] (see Fig. 1). Donahue et al. [5] showed that this strategy is successful for video recognition and description. The former network projects each frame of a sequence into a latent space obtaining an array of features. From such an

array, we can predict the frequency value by feeding them into the latter network. An LSTM is typically a natural choice for modeling time-series data such as the temporal evaluation of the extracted features as in this case. We used an LSTM with 2 hidden layers and 128 hidden nodes; we determined these hyper-parameters during early experimentation.

As CNN, we adopted a ResNet [8] because such a model has provided high-quality results in image classification and was successfully used in other techniques involving computer vision/imaging tasks. Since we have to process many frames, we employed ResNet-18, which has a good trade-off between quality and speed. This is extremely important when evaluating video sequences with a lot of frames; see Section 4.3. Since we are not interested in classification, we removed the last layers and we replaced them with three dense layers with 512 neurons each outputting a feature vector of 300 values; a value found out during pilot experiments.

3.1 Dataset

Acquisition. We acquired videos from different devices (e.g., drones, smartphones, etc.) at full HD resolution (1920×1080), following the classic photogrammetry guidelines. In total, we captured 11 videos for training and validation that we divided into 1,356 segments of 30-frame each; we chose this length for fitting into the video memory during training. For testing, we acquired other 7 videos that we divided into 403 segments of 30-frame each, and these videos were completely separated from the training and evaluation datasets. Note that the camera speed is not constant but varies along the video sequences and also within the same video sequence. In addition, the frame rate of these sequences was either at 30fps or 60fps.

Ground Truth. From these videos, we computed the ground-truth frequency of frames to be selected. To achieve this, we used a greedy approach. We created 3D reconstructions of each video at different regular sampling rates, selecting the sampling rate that was maximizing the number of generated sparse 3D points. To have manageable computations (i.e., under 2-hour per subsequence), we divided each video into 60-frame subsequences and we applied our greedy algorithm to each. For 3D reconstructions, we used OpenMVG [19]; a popular open-source package for SfM. Note that blurred frames and static frames were removed [22] before running 3D reconstructions.

Augmentation. We performed data augmented to enlarge this initial dataset. We applied three types of rotations (90°, 180°, and 270°) and horizontal flipping (also applied to each rotated image). We thus apply a total of six transformations for each video, obtaining $7,112$ videos (including the original ones). This means a total of $213,360$ frames to be processed.

3.2 Frames Encoding

During early experiments, we noted that the performance of our architecture depends on the content of the video, and this is undesirable. We assessed this problem by testing videos without motion (both camera and people/objects). We create a motionless video using a single frame of a segment that had a high-frequency ground-truth value. Since this video is motionless, we would expect to have a zero or close to zero frequency. However, the model was predicting a high frequency for sampling frames. This means that only the visual content of the frame itself was used by the network for the frequency prediction and not the motion. To overcome this issue, we decided to encode segments differently to force the model to extract motion-related features. In this encoding, we convert a sequence n frames long (in our case, $n = 30$) into a $n - 1$ sequence in which the new frames of this encoding, f_d, are computed as

$$f_d(i) = f(i + 1) - f(i), \tag{1}$$

where $f(i)$ is the i-th frame. We call this encoding for our segments *differential encoding*.

4 Results

We trained our model on a Linux machine (Ubuntu 18.04) equipped with an Intel CPU Core i7-7800X (3.50 GHz) with 64 GB of memory and an NVIDIA GeForce GTX 1080 GPU with 8 GB of memory. We used PyTorch 1.3.1 as the deep-learning framework for implementing our architecture.

We trained the network using mini-batch stochastic gradient descent and the Adam optimizer [13] with the learning rate set to $5 \cdot 10^{-5}$. We left the rest of the parameters set to their default values; i.e., $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\epsilon = 10^{-8}$. Since we were estimating a quantity; i.e., the subsampling frequency, we defined the loss function to be the *Mean Square Error* (MSE) between the predicted frequency and the ground-truth. We employed a pre-trained ResNet-18 available in PyTorch, and we kept the weights of the convolutional part frozen. We set the batch size of our network to 8, which is the largest value that can be set with the GPU memory available. The training set is shuffled whenever an epoch is completed to diminish the impact of order-based biases during training. We set the maximum number of epochs to 600 for an approximate training of 7 days and 8 hours.

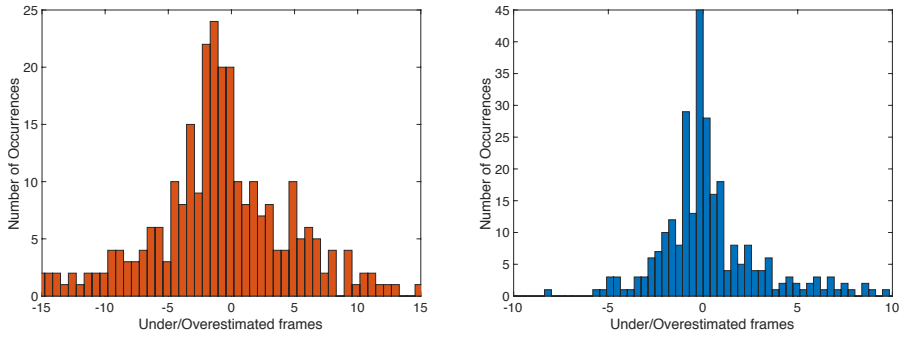
4.1 Ablation Study

We trained our architecture with and without our differential encoding. Figure 3 displays the training curves; i.e., the evolution of the loss as evaluated

on the training and validation. When we train our model using the differential encoding, the model reaches the converge faster than without it. In addition, as can be noticed from the plots.

4.2 Quality of Learning

In order to assess the quality of learning; i.e., estimating how-well the network distilled the ground-truth algorithm, we decided to plot the error (in frames) between the ground-truth and the values estimated by the network for the test dataset. Figure 2 plots the histograms of errors for the test data with respect to the classic encoding (a) and the differential one (b).



(a) Without Differential Encoding. (b) With Differential Encoding.

Figure 2: The histograms of the error distribution of under/overestimated frames for 30-frame sequences for standard encoding (a) and differential encoding (b).

The predictions our model produces are particularly accurate using the differential encoding; see Figure 2(b), as witnessed by a narrower distribution of test errors around 0 and a lower presence of outliers than using the classic encoding; see Figure 2(a). In addition, the differential encoding has the advantage to detect static scenes because there is no change in the temporal derivative.

4.3 Timings

Network Performance. Regarding the timings of our architecture, we measure its performance running all segments in the training set. On average, the computational time required by our network for estimating the frequency for 30 frames is about 122ms. This means we need only 7.32 seconds to process one minute of video.

3D Reconstruction. As a further step to show the advantage of our architecture, we applied it to the test videos in our dataset. Then, we used the selected frames as input images for COLMAP [24]. Note that COLMAP is not the same SfM software used in the training (i.e., OpenMVG). This

is done on purpose to test and show the robustness of our approach. For comparison, we ran COLMAP using all frames of the input video; Table 1 shows the results of this comparison.

Video	**Time Full** (min.)	**Time Our** (min.)	$\|V\|$ **Full**	$\|V\|$ **Our**	Reproj. Err. **Full**	Reproj. Err. **Our**
Street1	8	51(s.)	14,096	6,409	0.781	0.724
Street2	178	18	109,747	68,939	0.934	0.826
Valp.2	571	68	316,897	202,134	0.740	0.663
River	999	64	84,900	59,287	0.853	0.781
Fountain	1,046	14	68,715	28,623	1.272	1.151
Square	2,145	18	310,659	158,766	0.782	0.713
Apulia	2,175	2	63,923	16,811	0.844	0.629

Table 1: A comparison of computational time, number of generated vertices ($|V|$), and reprojection error (point-to-point) between using all frames of a videos (**full**) and frames selected using our architecture (**our**). Our solution can deliver a 3D model in a fraction of time compared to all frames with less reprojection error.

From this table, we can notice that the use of our architecture can save a lot of computational time compared to the use of all frames of an input video. For example, we can achieve on average two orders of magnitude speed up with a peak of three orders. Regarding the numbers of generated vertices, $|V|$, our approach can have on overage a 49% reduction in the number of points with a peak of 73%. Although this seems to be a large reduction, our methodology has the advantage to provide to the users results in a matter of minutes or a few hours instead of days without the need to inspect and select manually thousands of frames. Finally, Table 1 elicits that our method has a lower reprojection error than using all frames; i.e., less than 11% on average.

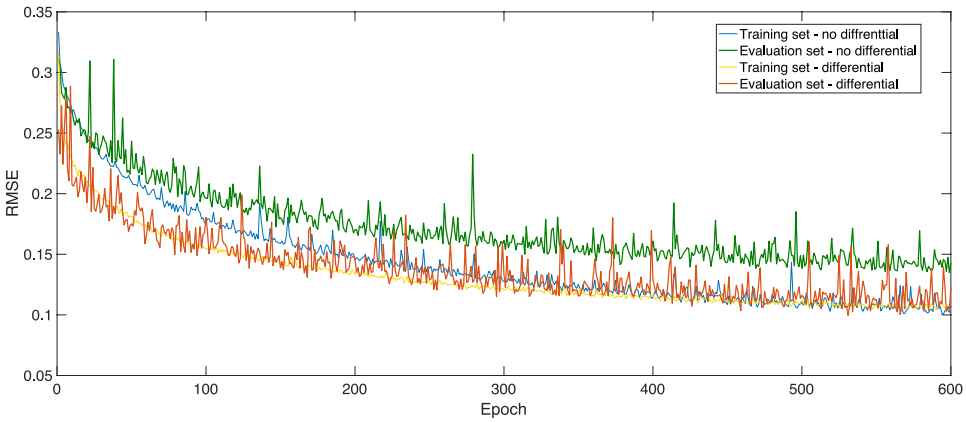


Figure 3: A comparison between the training plots of our architecture with and without differential encoding; see Section 3.2.

5 Conclusions and Future Works

In this work, we have presented a novel architecture for transferring the knowledge of an algorithm for selecting high-quality frames for SfM reconstruction. This architecture mimics the original frame selection algorithm, and it has the great advantage that it works in real-time and can be plugged into deep-learning-based pipelines for SfM. We have shown that 3D reconstructions using our method can reduce greatly their computational costs while keeping high-quality reconstructions. Another advantage is that the proposed method is independent of the track length since we worked at the frame-segments level of constant size.

The main limitation of our architecture is that it underestimates frames, as shown in Fig. 2. While overestimation is not an issue because it does not affect the final reconstruction quality, missing frames may reduce the number of generated vertices in the final point-cloud due to a too large baseline. In future work, we would like to solve this underestimation of frames to maintain a high vertices throughput.

Acknowledgments. This work was funded by the European Union's Horizon 2020 research and innovation programme under grant agreement No 820434 (ENCORE).

References

[1] Sameer Agarwal, Yasutaka Furukawa, Noah Snavely, Ian Simon, Brian Curless, Steven M Seitz, and Richard Szeliski. Building rome in a day. *Communications of the ACM*, 54(10):105–112, 2011.

[2] Alessandro Artusi, Francesco Banterle, Fabio Carrara, and Alejandro Moreo. Efficient evaluation of image quality via deep-learning approximation of perceptual metrics. *IEEE Trans. Image Process.*, 29:1843–1855, 2020.

[3] Francesco Banterle, Alessandro Artusi, Alejandro Moreo, and Fabio Carrara. NoR-VDPNet: A No-Reference High Dynamic Range Quality Metric Trained On Hdr-Vdp 2. In *2020 IEEE International Conference on Image Processing (ICIP)*, pages 126–130, 2020.

[4] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. Surf: Speeded up robust features. In *European conference on computer vision (ECCV)*, pages 404–417. Springer, 2006.

[5] Jeff Donahue, Lisa Anne Hendricks, Marcus Rohrbach, Subhashini Venugopalan, Sergio Guadarrama, Kate Saenko, and Trevor Darrell. Long-term recurrent convolutional networks for visual recognition and

description. *IEEE Trans. Pattern Anal. Mach. Intell.*, 39(4):677–691, April 2017.

[6] Jakob Engel, Thomas Schöps, and Daniel Cremers. Lsd-slam: Large-scale direct monocular slam. In *European conference on computer vision*, pages 834–849. Springer, 2014.

[7] Yasutaka Furukawa, Brian Curless, Steven M Seitz, and Richard Szeliski. Towards internet-scale multi-view stereo. In *2010 IEEE computer society conference on Computer Vision and Pattern Recognition*, pages 1434–1441. IEEE, 2010.

[8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE CVPR 2016, Las Vegas, USA*, pages 770–778, 2016.

[9] Jared Heinly, Johannes L Schonberger, Enrique Dunn, and Jan-Michael Frahm. Reconstructing the world* in six days*(as captured by the yahoo 100 million image dataset). In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3287–3295, 2015.

[10] Geoffrey Hinton, Oriol Vinyals, and Jeffrey Dean. Distilling the knowledge in a neural network. In *NIPS Deep Learning and Representation Learning Workshop*, 2015.

[11] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.

[12] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.

[13] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *Proc. of the 3rd International Conference on Learning Representations (ICLR)*, 2014.

[14] Maria Klodt and Andrea Vedaldi. Supervising the new with the old: Learning SFM from SFM. In *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part X*, volume 11214 of *Lecture Notes in Computer Science*, pages 713–728. Springer, 2018.

[15] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems (NIPS 2012)*, volume 25, pages 1106–1114, 2012.

[16] Yann LeCun, Yoshua Bengio, and Geoffrey E. Hinton. Deep learning. *Nat.*, 521(7553):436–444, 2015.

[17] Ludovic Magerand and Alessio Del Bue. Practical projective structure from motion (p2sfm). In *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, pages 39–47. IEEE Computer Society, 2017.

[18] Simone Milani and Enrico Tronca. Improving three-dimensional reconstruction of buildings from web-harvested images using forensic clues. *J. Electronic Imaging*, 26(1):11009, 2017.

[19] Pierre Moulon, Pascal Monasse, Romuald Perrot, and Renaud Marlet. OpenMVG: Open Multiple View Geometry. In *1st Workshop on Reproducible Research in Pattern Recognition*, volume 10214 of *LNCS*, pages 60–74. Springer, Dec 2016.

[20] Kemal E Ozden, Konrad Schindler, and Luc Van Gool. Multibody structure-from-motion in practice. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(6):1134–1141, 2010.

[21] M-G Park and K-J Yoon. Optimal key-frame selection for video-based structure-from-motion. *Electronics letters*, 47(25):1367–1369, 2011.

[22] Gaia Pavoni, Matteo Dellepiane, Marco Callieri, and Roberto Scopigno. Automatic Selection of Video Frames for Path Regularization and 3D Reconstruction. In Chiara Eva Catalano and Livio De Luca, editors, *Eurographics Workshop on Graphics and Cultural Heritage*. The Eurographics Association, 2016.

[23] Benjamin Resch, Hendrik Lensch, Oliver Wang, Marc Pollefeys, and Alexander Sorkine-Hornung. Scalable structure from motion for densely sampled videos. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3936–3944, 2015.

[24] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[25] Noah Snavely, Steven M. Seitz, and Richard Szeliski. Photo tourism: Exploring photo collections in 3d. *ACM Trans. Graph.*, 25(3):835–846, July 2006.

[26] Thorsten Thormählen, Hellward Broszio, and Axel Weissenfeld. Keyframe selection for camera motion and structure estimation from multiple views. In *European Conference on Computer Vision*, pages 523–535. Springer, 2004.

[27] Sudheendra Vijayanarasimhan, Susanna Ricco, Cordelia Schmid, Rahul Sukthankar, and Katerina Fragkiadaki. Sfm-net: Learning of structure and motion from video. *arXiv preprint arXiv:1704.07804*, 2017.

[28] Xingkui Wei, Yinda Zhang, Zhuwen Li, Yanwei Fu, and Xiangyang Xue. Deepsfm: Structure from motion via deep bundle adjustment. *arXiv preprint arXiv:1912.09697*, 2019.

[29] Tinghui Zhou, Matthew Brown, Noah Snavely, and David G. Lowe. Unsupervised learning of depth and ego-motion from video. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[30] Siyu Zhu, Runze Zhang, Lei Zhou, Tianwei Shen, Tian Fang, Ping Tan, and Long Quan. Very large-scale global sfm by distributed motion averaging. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4568–4577, 2018.