PixStabNet: FAST MULTI-SCALE DEEP ONLINE VIDEO STABILIZATION WITH PIXEL-BASED WARPING

Yu-Ta Chen[1] *Kuan-Wei Tseng*[1] *Yao-Chih Lee*[2] *Chun-Yu Chen*[1] *Yi-Ping Hung*[1]

[1]National Taiwan University [2]Academia Sinica, Taiwan

ABSTRACT

Online video stabilizaton is increasingly needed for real-time applications such as live streaming, drone remote control, and video communication. We propose a multi-scale convolutional neural network (PixStabNet) which stabilizes video in real time without using future frames. Instead of calculating a global homography or multiple homographies, we estimate a pixel-based warping map to make the transformation of each pixel to achieve more precise modelling. In addition, we propose well-designed loss functions along with a two-stage training scheme to enhance network robustness. The quantitative result shows that our method outperforms other learning-based online methods in terms of stability with excellent geometric and temporal consistency. Moreover, to the best of our knowledge, the proposed algorithm is the most efficient approach for video stabilization. The models and results are available at: https://yu-ta-chen.github.io/PixStabNet.

***Index Terms*—** Video Stabilization, Multi-Scale Architecture, Pixel-Based Warping, Real-Time Processing

1. INTRODUCTION

Video captured with hand-held cameras usually contains undesirable shaky content, making it difficult to watch. To remove jitters and generate stable video that can be viewed comfortably, many software video stabilization alogrithms have been proposed. Offline methods [1, 2, 3, 4] are designed to stabilize videos that have been fully recorded. It usually requires a considerable amount of time to achieve an optimal result. Online methods [5, 6, 7, 8] are designed to stabilize videos in real-time without using future frames. Although there are hardware solutions such as Optical Image Stabilization (OIS) and Electronic Image Stabilization (EIS), they might not be available or reliable on low-end devices.

Recently, with advances in deep learning, convolutional neural networks (CNNs) have been widely used in computer vision tasks. To use CNN to stabilize a video in real time, a pre-trained network is necessary. StabNet, the pioneer work proposed by Wang et al. [6] predicts a set of mesh-grid transformations. Xu et al. [9] use spatial transformer networks

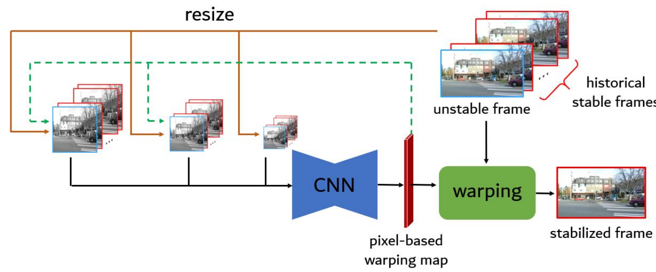

Fig. 1. Pipeline of proposed method.

(STNs) [10] to predict affine transformations. However, these two methods do not account for depth variation because of the transformations they use. Besides, they are not robust since they use historical ground-truth frames as training input but historical stabilized frames as testing input, which can result in severely distorted and tilted output videos. PW-StableNet [7] consider depth variation by generating pixel-based warping maps instead, which however incurs a delay of at least 15 frames due to the use of 15 future frames as network input. To solve these problems, in this work, we propose a novel deep learning approach for online video stabilization.

The pipeline of our method is shown in Fig. 1. First, we propose a multi-scale CNN that directly predicts transformations for each incoming unstable frame. This multi-scale approach resembles a coarse-to-fine optimization strategy. Second, the proposed network is an encoder-decoder architecture which estimates pixel-based warping maps to stabilize frames. We train our network on a public dataset with well-designed loss functions. Moreover, a two-stage training scheme is proposed to enhance the robustness of our network.

We evaluate our approach on the NUS dataset [1]. Experimental results demonstrate that the proposed model produce more stable video than other state-of-the art online methods. Moreover, the main advantage of our approach is that it runs on an NVIDIA RTX 2080Ti graphics card at a real-time speed of 54.6 FPS, which is the most efficient approach for software video stabilization.

2. RELATED WORK

In this section, we briefly complement some relevant works on video stabilization frameworks. We do not discuss tradi-

This work is partially supported by MediaTek Inc., and Ministry of Science and Technology in Taiwan (MOST 109-2221-E-002-207-MY3)

tional offline methods [1, 3, 11, 12] since these method rely on estimating and smoothing camera trajectory via non-linear optimization, which is too slow for real-time applications. In recent years, many offline stabilization methods based on deep learning have been proposed. Yu and Ramamoorthi [2] treat the CNN as an optimizer and propose objective functions based on optical flow to optimize input videos. They also proposed inferring per-pixel warping fields from the optical flow fields of the input video [13]. However, the pre-stabilization stage in their framwork, which requires either KLT [14] or SURF [15] for feature tracking, is time-consuming. Furthermore, learning-based optical-flow estimation such as standard FlowNet2 [16] takes 123 ms per frame, making it impossible for real-time performance.

Online video stabilization is difficult to implement as it must be executed real-time. Therefore, there are few traditional online video stabilization methods. The pioneer work [8] applies low-pass filters to smooth model parameters. Liu et al. [5] place a regular 2D mesh on the video frame to yield a motion vector at each vertex. As for learning-based methods, as mentioned in previous section, such as StabNet [6], predicts a mesh-based transformation to warp unstable frames into stable frames. Although some of these methods show strong and efficient results, they do not solve errors caused by depth variation because they only estimate a global homography or several homographies based on meshes to warp shaky frames. To address this problem, PW-StableNet [7] estimates a pixel-based warping map. However, it requires 15 future frames as input, resulting in a fixed-delay.

3. PROPOSED METHOD

We propose an encoder-decoder architecture to generate pixel-based warping maps instead of one global homography, or mesh-based transformations with multiple homographies to stabilize an unstable video. To process an unstable frame I_t, the input of our network is concatenated by I_t and a sequence of consecutive historical stable frames $S_t =< s_{t-\omega}, s_{t-\omega+1}, \ldots, s_{t-1} >$ for time-stamp t and window size w. The output of the network is a warping map T_t with the same size as I_t containing channels T_t^x and T_t^y. For each pixel (x', y') of the stabilized frame $\hat{I}_t$, the corresponding pixel (x, y) on I_t is obtained from T_t^x and T_t^y.

3.1. Multi-Scale Approach

As shown in Fig. 2, we use a Siamese network, which has two branches that share the same parameters to enhance the temporal consistency of successive stabilized frames $\hat{I}_{t-1} = T_{t-1}(I_{t-1})$ and $\hat{I}_t = T_t(I_t)$ for training. For testing, only one branch is needed to stabilize a video. The input consists of the unstable current frame I_t and a sequence of consecutive historical stable frames S_t. Inspired by Nah et al. [17], we adopt a coarse-to-fine optimization strategy. Each branch is a

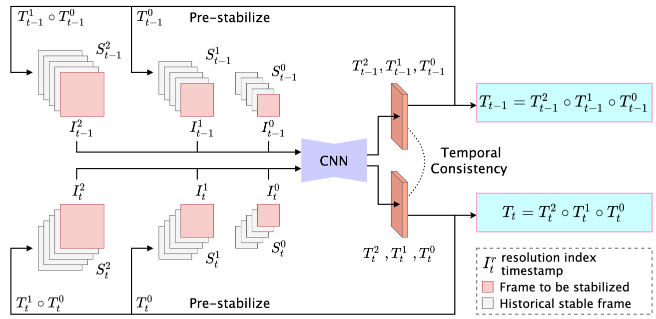

Fig. 2. Proposed multi-scale network approach. I is the frame to be stabilized, S is the sequence of consecutive historical stable frames, T is the warping map, and $\circ$ is the warping map conbination operator.

multi-scale architecture, and the input is resized to three resolution stacks (S_t^d, I_t^d), where $d \in \{0, 1, 2\}$ is the resolution index, and larger numbers represent higher resolutions. The network starts off at the coarsest level to estimate rough transformations. To deliver coarser level output to finer levels, we use the warping map predicted by the coarser level network to pre-stabilize the finer level unstable frame before feeding into the network, after which the network processes the finer level stack for further optimization. Finally, we combine the warping maps of each level to yield the final warping map T_t.

3.2. Network Architecture

The details of our network are shown in Fig. 3. The network is similar to U-Net [18], which uses an encoder-decoder architecture. The encoder is composed of multiple convolution layers for feature extraction, and the decoder is composed of multiple upconvolution layers and convolution layers for pixel-wise motion prediction. Feature maps of the same channel and same size in the encoder and the decoder are connected through skip-connections. Moreover, we add an additional STN module to the last layer of the encoder to predict a warping map F_0. The last layer of the decoder is used to generate a pixel-based warping map F_1 to fine-tune the motion of each pixel. The first two scales only use F_0 to estimate the warping map T_t^0 and T_t^1 to move unstable frames to an approximate location in 2D space, and the warping map of the last scale T_t^2 is combined with F_0 and F_1 to generate a per-pixel warping field.

3.3. Loss Function

The loss function consists of three terms: stability loss $\mathcal{L}_{stab}$, distortion-reducing loss $\mathcal{L}_{dr}$, and temporal loss $\mathcal{L}_{temp}$. We calculate the loss for each scale d and define the total loss function as,

$$\mathcal{L} = \sum_{d \in \{0,1,2\}} 2^d \mathcal{L}^d \tag{1}$$

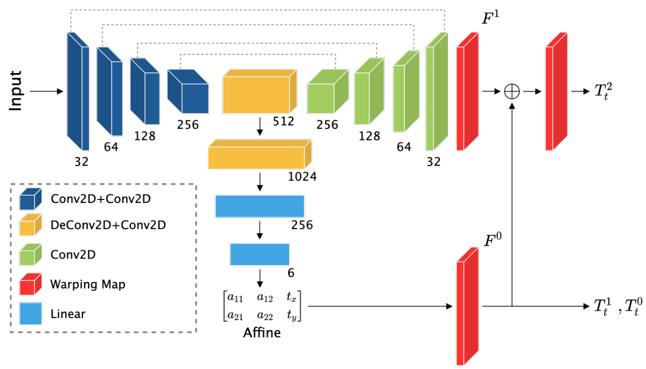

Fig. 3. Proposed convolution neural network architecture

$$\mathcal{L}^d = \gamma\,\mathcal{L}_{temp} + \sum_{k\in\{t-1,t\}} \alpha\,\mathcal{L}_{stab} + \beta\,\mathcal{L}_{dr} \tag{2}$$

For each scale d, the input of each term is the frame or warping map in the corresponding scale.

3.3.1. Stability Loss.

The stability loss is used to drive the stabilized frame to the ground-truth stable frames. It is defined as combination of multi-scale photometric loss $\mathcal{L}_{photo}$, modified from [19] under the inspiration of [20], and perceptual loss $\mathcal{L}_{VGG}$ in [21].

$$\mathcal{L}_{stab}(\hat{I}_t, \tilde{I}_t) = w_0\,\mathcal{L}_{photo}(\hat{I}_t, \tilde{I}_t) + \mathcal{L}_{VGG}(\hat{I}_t, \tilde{I}_t), \tag{3}$$

In photometric loss, we merge mean absolute error (MAE) and structural similarity (SSIM) to evaluate how the stabilized frame $\hat{I}_t$ aligns with the ground-truth stable frame $\tilde{I}_t$. This, however, does not work if there are many pixels in the low-texture region or that are far from the ground truth. To account for this, we shrink $\hat{I}_t$ and $\tilde{I}_t$ to the same size as the three resolutions to compute the gradients from larger spatial regions. Thus, the photometric loss is formulated as,

$$\mathcal{L}_{photo}(\hat{I}_t, \tilde{I}_t) = \sum_{r=0}^{2} \lambda_0 |\hat{I}_t^r - \tilde{I}_t^r| + \lambda_1 \mathrm{DSSIM}(\hat{I}_t^r, \tilde{I}_t^r) \tag{4}$$

Since pixel-wise loss functions do not capture perceptual differences such as high texture. Therefore, we use perceptual loss $\mathcal{L}_{VGG}$, for which we compute the mean squared error (MSE) of $\hat{I}_t$ and $\tilde{I}_t$ in the feature spaces of VGG16 [21].

$$\mathcal{L}_{VGG}(\hat{I}_t, \tilde{I}_t) = \mathrm{MSE}\left(\mathrm{VGG16}(\hat{I}_t), \mathrm{VGG16}(\tilde{I}_t)\right) \tag{5}$$

3.3.2. Distortion-Reducing Loss

Since the pixel-based warping map can result in severe distortion in the stabilized frame, distortion-reducing loss is proposed to prevent visual artifacts. The loss constrains the per-pixel warping field to approximate a linear warping field. We first downscale the warping map T_t to 16×16 to remove the noise, and then upscale it to its original size, after which we align T_t to the denoising warping map T'. We define the loss function as

$$\mathcal{L}_{dr}(T_t', T_t) = \|T_t' - T_t\| \tag{6}$$

Note that only the finest level in our multi-scale approach generates a pixel-based warping map, $\mathcal{L}_{dr}$ places the constraint on the maximum resolution warping map T_t^2.

3.3.3. Temporal Loss

To ensure temporal smoothness in adjacent stabilized frames, we utilize a temporal loss function to retain low-frequency motion in the input video. At each time t, I_{t-1} and I_t are fed to the network, generating two successive stabilized frames, $\hat{I}_{t-1}$ and $\hat{I}_t$. We define the temporal loss $\mathcal{L}_{temp}$ as the photometric error between $\hat{I}_t$ and $\omega(\hat{I}_{t-1})$, where $\omega(\cdot)$ is a function that warps the stable frame $\hat{I}_{t-1}$ to $\hat{I}_t$ according to pre-computed optical flow estimated by FlowNet2 [16].

$$\mathcal{L}_{temp} = w_0\,\mathcal{L}_{photo}\left(\hat{I}_t, \omega(\hat{I}_{t-1})\right) \tag{7}$$

3.4. Training

The model is trained on the DeepStab dataset [6], which contains 61 pairs of synchronized unstable and stable videos. We split the videos into 45 training pairs, 8 validation pairs, and 8 testing pairs.

3.4.1. Two-stage Training Scheme

Video artifacts occur in methods [6, 9] which use historical stable frames as input. These approaches only stack as training inputs ground-truth stable frames which are well-shaped and have smooth camera trajectories. However, as no ground-truth frames are available during testing, inputs are replaced by historical stabilized frames. The resultant differences in quality between training and testing input can lead to severe distortion and incorrect tilt in the output video. An example of these problems is shown in Fig. 4.

To solve this problem, we propose a two-stage training approach. In the first stage, we take historical ground-truth frames as training input, which allows the network to converge quickly. Also, we exert some black borders by a randomly sampled homography on the historical ground-truth frames. In the second stage, we replace the historical ground-truth frames with the stabilized frames to simulate the test situation, contributing to more robust stabilization results.

3.4.2. Implementation Details

For pre-processing, we convert the frames to grayscale images and normalize the pixel values to between -1 and 1 before feeding them to the network. We define the scale levels of the proposed multi-scale approach: 64×64, 128×128,

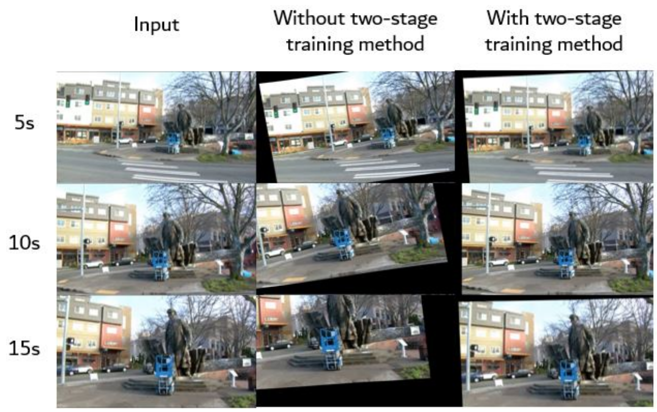

Fig. 4. Stabilization w/o two-stage training scheme. The output video tilt increasingly due to propogation of uncertainty if the network is trained merely on ground-truth.

and 256×256, corresponding to resolution index 0, 1, and 2, respectively. We used ADAM for optimization with $\beta_1 = 0.9$ and $\beta_2 = 0.999$. The hyper-parameters are $\alpha = 200$, $\beta = 0.1$, $\gamma = 200$, $\lambda_0 = 0.15$, and $\lambda_1 = 0.85$. w_0 is the reciprocal of number of pixels of the input frame. In the two-stage training strategy, the network was trained for 10 epochs in the first stage and 30 epochs in the second stage. The initial learning rate was set to 2×10^{-5} and divided by 10 every 10 epochs starting from the second stage. During the training process, considering that most videos play at 30 FPS, we set the length of historical stable frames w to 30. In testing, we repeated the first frame w times and placed these at the head of the video.

4. EXPERIMENTS

We compare the proposed approach (PixStabNet) with several state-of-the-art learning-based online stabilization alogorithms, including StabNet [6], and PWStableNet [7], on NUS dataset [1]. It contains six categories, a total of 144 collected videos. The experiments were conducted on an Intel i7-8700 CPU and an NVIDIA RTX 2080Ti graphics card.

The quantitative evaluation, with metric modified from [1], and runtime comparison are shown in Table 1. We use the non-cropping ratio, the non-distortion value, and the stability score to evaluate stabilization methods. The non-cropping ratio (C) evaluates the remaining area after processing. For each stabilized frame s_t, a homography H_t is first estimated from s_t to the unstable frame u_t, after which H_t projects s_t to u_t. The non-cropping ratio is then defined as the ratio of the remaining area in the stabilized frame $H_t(s_t)$ between the area of the unstable frame u_t. The non-distortion value (D) measures the degree of stabilized frame distortion. At each time t, we estimate a homography as in non-cropping ratio. The distortion is arised from anisotropic scaling of H_t, which can be extracted from SVD decomposition of H_t. The stability score (S) is computed as the average signal-to-noise ratio (SNR) of

Category	Metric	StabNet	PWStableNet	PixStabNet
Regular	C	0.54	0.78	0.61
	D	0.82	0.97	0.93
	S	**13.85**	10.68	13.72
Parallax	C	0.45	0.76	0.51
	D	0.71	0.93	0.90
	S	**15.07**	12.30	14.92
Crowd	C	0.41	0.77	0.43
	D	0.65	0.94	0.88
	S	17.58	16.24	**17.95**
Running	C	0.41	0.68	0.45
	D	0.77	0.92	0.92
	S	12.88	11.03	**13.11**
Quick Rotation	C	0.39	0.75	0.44
	D	0.67	0.91	0.89
	S	18.13	18.16	**18.82**
Zooming	C	0.47	0.79	0.50
	D	0.71	0.93	0.87
	S	17.25	15.02	**17.54**
Runtime (FPS)		8.5	42.4	**54.6**

Table 1. Quantitative and runtime comparison (*PWStableNet is semi-online with 15 frames fixed delay.)

the entire sequence. We first estimate the optical flow [22] between successive frames, and then divide the flow maps into 4×4 grids and average each grid to obtain the 2D local motions, which are converted into two 1D temporal signals for frequency domain analysis. The lowest 5% of the frequency components are taken as the signal, and the rest are treated as the noise (the DC component is excluded).

The result shows that our network produces more stable and less distorted results than StabNet. Although the videos produced by PWStableNet are less distorted, they still show severe shaking. Morever, the proposed method is the fastest online method which *do not* use any future frame. Thus, we conclude that the proposed approach produces quantitatively better results than other learning-based online methods.

5. CONCLUSIONS

We propose a learning-based method to solve problems with online video stabilization. The contributions of our work are threefold. First, we utilize a multi-scale network architecture to generalize spatially consistent camera motion characteristics. Second, it is a true online method that can operate in real-time (54.6 FPS) without any use of future frames. Last but not least, we propose versatile loss functions with two-stage training scheme to obtain high geometric and temporal consistency. Experimental results show that the proposed algorithm surpasses other learning-based online methods in terms of stability with high shape preservation. Moreover, the proposed approach has the highest processing speed of all state-of-the-art methods.

6. REFERENCES

[1] Shuaicheng Liu, Lu Yuan, Ping Tan, and Jian Sun, "Bundled camera paths for video stabilization," *ACM Transactions on Graphics (TOG)*, vol. 32, no. 4, pp. 1–10, 2013.

[2] Jiyang Yu and Ravi Ramamoorthi, "Robust video stabilization by optimization in CNN weight space," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 3800–3808.

[3] Matthias Grundmann, Vivek Kwatra, and Irfan Essa, "Auto-directed video stabilization with robust L1 optimal camera paths," in *CVPR 2011*, 2011, pp. 225–232.

[4] Jinsoo Choi and In So Kweon, "Deep iterative frame interpolation for full-frame video stabilization," *ACM Transactions on Graphics (TOG)*, vol. 39, no. 1, pp. 1–9, 2020.

[5] Shuaicheng Liu, Ping Tan, Lu Yuan, Jian Sun, and Bing Zeng, "Meshflow: Minimum latency online video stabilization," in *European Conference on Computer Vision*. Springer, 2016, pp. 800–815.

[6] Miao Wang, Guo-Ye Yang, Jin-Kun Lin, Song-Hai Zhang, Ariel Shamir, Shao-Ping Lu, and Shi-Min Hu, "Deep online video stabilization with multi-grid warping transformation learning," *IEEE Transactions on Image Processing*, vol. 28, no. 5, pp. 2283–2292, 2018.

[7] Minda Zhao and Qiang Ling, "Pwstablenet: Learning pixel-wise warping maps for video stabilization," *IEEE Transactions on Image Processing*, vol. 29, pp. 3582–3595, 2020.

[8] Hung-Chang Chang, Shang-Hong Lai, and Kuang-Rong Lu, "A robust real-time video stabilization algorithm," *Journal of Visual Communication and Image Representation*, vol. 17, no. 3, pp. 659–673, 2006.

[9] Sen-Zhe Xu, Jun Hu, Miao Wang, Tai-Jiang Mu, and Shi-Min Hu, "Deep video stabilization using adversarial networks," in *Computer Graphics Forum*. Wiley Online Library, 2018, vol. 37, pp. 267–276.

[10] Max Jaderberg, Karen Simonyan, Andrew Zisserman, et al., "Spatial transformer networks," in *Advances in Neural Information Processing Systems*, 2015, pp. 2017–2025.

[11] Feng Liu, Michael Gleicher, Hailin Jin, and Aseem Agarwala, "Content-preserving warps for 3D video stabilization," *ACM Transactions on Graphics (TOG)*, vol. 28, no. 3, pp. 1–9, 2009.

[12] Shuaicheng Liu, Lu Yuan, Ping Tan, and Jian Sun, "Steadyflow: Spatially smooth optical flow for video stabilization," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 4209–4216.

[13] Jiyang Yu and Ravi Ramamoorthi, "Learning video stabilization using optical flow," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 8159–8167.

[14] Jianbo Shi and Tomasi, "Good features to track," in *IEEE Conference on Computer Vision and Pattern Recognition*, 1994, pp. 593–600.

[15] Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool, "Speeded-up robust features (surf)," *Computer vision and image understanding*, vol. 110, no. 3, pp. 346–359, 2008.

[16] Eddy Ilg, Nikolaus Mayer, Tonmoy Saikia, Margret Keuper, Alexey Dosovitskiy, and Thomas Brox, "Flownet 2.0: Evolution of optical flow estimation with deep networks," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 2462–2470.

[17] Seungjun Nah, Tae Hyun Kim, and Kyoung Mu Lee, "Deep multi-scale convolutional neural network for dynamic scene deblurring," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 3883–3891.

[18] Olaf Ronneberger, Philipp Fischer, and Thomas Brox, "U-Net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, 2015, pp. 234–241.

[19] Hang Zhao, Orazio Gallo, Iuri Frosio, and Jan Kautz, "Loss functions for image restoration with neural networks," *IEEE Transactions on Computational Imaging*, vol. 3, no. 1, pp. 47–57, 2016.

[20] Tinghui Zhou, Matthew Brown, Noah Snavely, and David G Lowe, "Unsupervised learning of depth and ego-motion from video," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 1851–1858.

[21] Karen Simonyan and Andrew Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[22] Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz, "PWC-Net: CNNs for optical flow using pyramid, warping, and cost volume," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 8934–8943.