Non-linear Motion Estimation for Video Frame Interpolation using Space-time Convolutions

Saikat Dutta Arulkumar Subramaniam Anurag Mittal
Indian Institute of Technology Madras
Chennai, India
saikat.dutta779@gmail.com {aruls,amittal}@cse.iitm.ac.in

Abstract

Video frame interpolation aims to synthesize one or multiple frames between two consecutive frames in a video. It has a wide range of applications including slow-motion video generation, video compression and developing video codecs. Some older works tackled this problem by assuming per-pixel linear motion between video frames. However, objects often follow a non-linear motion pattern in the real domain and some recent methods attempt to model per-pixel motion by non-linear models (e.g., quadratic). A quadratic model can also be inaccurate, especially in the case of motion discontinuities over time (i.e. sudden jerks) and occlusions, where some of the flow information may be invalid or inaccurate. In our paper, we propose to approximate the per-pixel motion using a space-time convolution network that is able to adaptively select the motion model to be used. Specifically, we are able to softly switch between a linear and a quadratic model. Towards this end, we use an end-to-end 3D CNN encoder-decoder architecture over bidirectional optical flows and occlusion maps to estimate the non-linear motion model of each pixel. Further, a motion refinement module is employed to refine the non-linear motion and the interpolated frames are estimated by a simple warping of the neighboring frames with the estimated per-pixel motion. We show that our method outperforms state-of-the-art algorithms on four datasets.

1. Introduction

Video frame interpolation (VFI) is a significant video enhancement problem which aims to synthesize one or more visually coherent frames between two consecutive frames in a video, i.e., to up-scale the number of video frames. Efficient VFI algorithms can play a major in a video compression-decompression framework by simply dropping frames in the compression stage and reconstructing those frames seamlessly in the decompression stage [35, 29]. In addition, VFI finds its usage in numerous video-based applications such as slow-motion video generation (e.g., in sports and TV commercials), generating short videos from GIF images [33], novel view synthesis [6] and medical imaging [15].

VFI methods use various temporal cues to aid in frame interpolation. *Optical flow based approaches* [11, 19, 32] predominantly use 2D optical flow [18, 13, 31, 37, 30, 26] to warp the neighboring frames and estimate the interpolated frame. However, estimating accurate optical flow is a hard problem, especially when it involves large motion, illumination variations and motion blur. Alternatively, *phase-based approaches* estimate per-pixel phase [21, 20, 39] to aid frame interpolation. *kernel-based methods* strive to estimate per-pixel kernels to blend patches from neighborhood frames [24, 25, 17]. Different from these conventional methods, recent deep approaches use multiple frames to capture complex motion dynamics between frames. For instance, Choi et al. [4] utilize three frames and their bi-directional optical flows to generate the intermediate flows and use warping to estimate the final interpolated frame. Chi et al. [3] use a pyramid style network with cubic modeling to produce seven intermediate frames. Xu et al. [36] use four frames to model a quadratic motion between frames and determine the quadratic motion parameters by an analytical solution involving optical flow. Following this paradigm, in our method, we propose to use four frames and estimate non-linear (quadratic) motion model similar to [36]. However, we show that using a powerful 3D CNN to estimate the motion parameters instead of an analytical solution significantly performs better (ref. Section 3).

In our work, first we compute bi-directional flow and occlusion maps from four neighboring frames and predict a non-linear flow model with the help of a 3D CNN. In this regard, we formulate a novel 3D CNN architecture namely "GridNet-3D" inspired from [38] for efficient multi-scale feature aggregation. Further, the predicted non-linear flow model is used as coefficients in a quadratic formulation of inter-frame motion. The idea is that such an approach can adaptively select between linear and quadratic models by estimating suitable values for the coefficients. Intermedi-

ate backward flows are produced through flow reversal and motion refinement. Finally, two neighboring frames are warped and combined using a blending mask to synthesize the interpolated frame. Our algorithm demonstrates state-of-the-art performance over existing approaches on multiple datasets.

The main contributions of our work are summarized as follows:

- We introduce a novel frame interpolation algorithm that utilizes both flow and occlusion maps between four input frames to estimate an automatically adaptable pixel-wise non-linear motion model to interpolate the frames.
- We propose a parameter and runtime-efficient 3D CNN named "GridNet-3D" to aggregate multi-scale features efficiently.
- Through a set of comprehensive experiments on four publicly available datasets (Vimeo, DAVIS, HD and GoPro), we demonstrate that our method achieves state-of-the-art performance.

2. Space-time convolution network for non-linear motion estimation

Determining the motion trajectory of pixels is essential to determine the transition of pixel values from one frame to the next. Traditional methods use optical flow to achieve this goal with the assumption of brightness constancy and velocity smoothness constraint and use a linear model for interpolation. While some methods recently have used a quadratic model for flow estimation with improved results, such a model is not applicable in certain scenarios such as motion discontinuities and occlusions. In this work, we opt to use a 3D CNN encoder-decoder architecture to estimate per-pixel non-linear motion that can easily switch between a linear and quadratic model. Specifically, the 3D CNN takes a set of bi-directional optical flows and occlusion maps between consecutive video frames $\{I_{-1}, I_0, I_1, I_2\}$ to estimate the non-linear motion model that is utilized by other modules to predict an interpolated frame I_t, where $t \in (0, 1)$. i.e., the output frame I_t needs to be coherent in terms of appearance and motion between I_0 and I_1.

An overview of our framework is shown in Figure 1. The framework consists of the five modules namely: 1) Non-linear motion estimation (NME) module, 2) Backward flow estimation (BFE) module, 3) Motion refinement (MR) module, 4) Blending mask estimation (BME) module, and 5) Frame synthesis. The details of each module are described in the following sections.

2.1. Non-linear motion estimation (NME) module

Recent methods attempt to overcome linear motion assumption by modeling a non-linear motion. Xu et al. [36] proposed to model a quadratic motion model in terms of time t. i.e., with an assumption that pixel motion follows a quadratic motion of form $\alpha t + \beta t^2$. They estimate the motion model parameters α, β by an analytical formula derived using per-pixel optical flow. However, such a quadratic assumption cannot be applied to the pixels involving unreliable optical flow estimates (*e.g.* occluded pixels). Using such unreliable optical flow estimates may lead to inaccurate intermediate flow estimation and may end up with erroneous interpolation results. Instead of directly estimating quadratic motion parameters from optical flow, we attempt to estimate α, β through a 3D CNN model.

To learn suitable α and β in the non-linear motion model, given the input frames $\{I_{-1}, I_0, I_1, I_2\}$, we first estimate bi-directional flow and occlusion maps between neighboring frames using a pre-trained *PWCNet-Bi-Occ* network [9].

The bi-directional optical flows $\{F_{i\to(i+1)}, F_{(i+1)\to i}\}_{i=-1}^{2}$ and occlusion maps $\{O_{i\to(i+1)}, O_{(i+1)\to i}\}_{i=-1}^{2}$ are arranged in temporal order and results in a 5D tensor of size $B \times 6 \times \#\text{frames} \times H \times W$. Here B, H, W denote batch size, height and width respectively, and the 6 channels belong to bi-directional optical flows and occlusion maps. This tensor is passed through a 3D CNN model to estimate a representation of dimension $B \times 4 \times 2 \times H \times W$. The temporal dimension of 2 corresponds to $t = 0$ and $t = 1$. In each temporal slice, we predict two coefficient maps α and β, each of 2-dimensions. We refer these coefficients α, β as the flow representation. Now the per-pixel non-linear motion $F_{0\to t}$ of frame I_0 towards the interpolated frame I_t is given by:

$$F_{0\to t} = \alpha_0 \times t + \beta_0 \times t^2 \tag{1}$$

Similarly, $F_{1\to t}$ is given by:

$$F_{1\to t} = \alpha_1 \times (1-t) + \beta_1 \times (1-t)^2 \tag{2}$$

Estimating the coefficients $\alpha_0, \beta_0, \alpha_1$ and β_1 through a neural network instead of an analytical solution [36] offers the following advantages: 1) The network can flexibly choose between linear and non-linear motion. For pixels to follow a linear motion, the network may predict $\beta = 0$; 2) Unlike [36], learned estimates of α's, β's are better equipped to handle occlusion by utilizing the occlusion maps, 3) Having access to large temporal receptive field of 4 frames, the non-linear motion coefficients estimated through a 3D CNN can determine more accurate motion than [36] which rely on optical flow to estimate the coefficients. Figure 2 shows the pipeline of the non-linear motion estimation module.

Network specification: We formulate the NME module to predict α, β with two crucial design choices in mind: 1) to capture spatiotemporal features and 2) to incorporate multi-scale features efficiently. 3D CNN networks are the natural choices to capture spatiotemporal features among video frames. However, the existing architectures

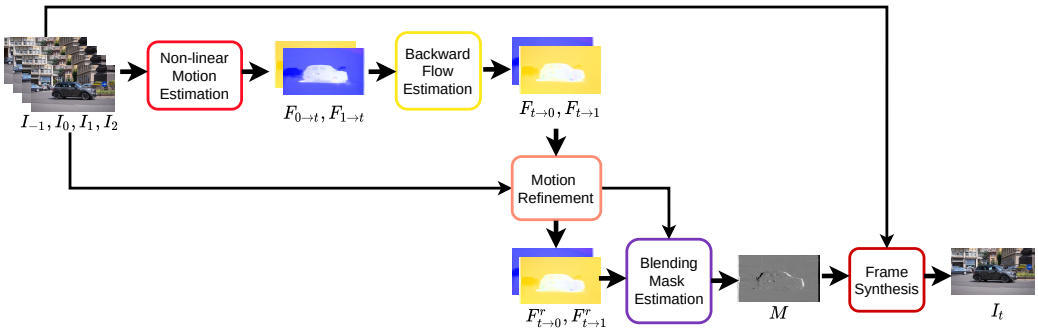

Figure 1. Overview of our interpolation algorithm.

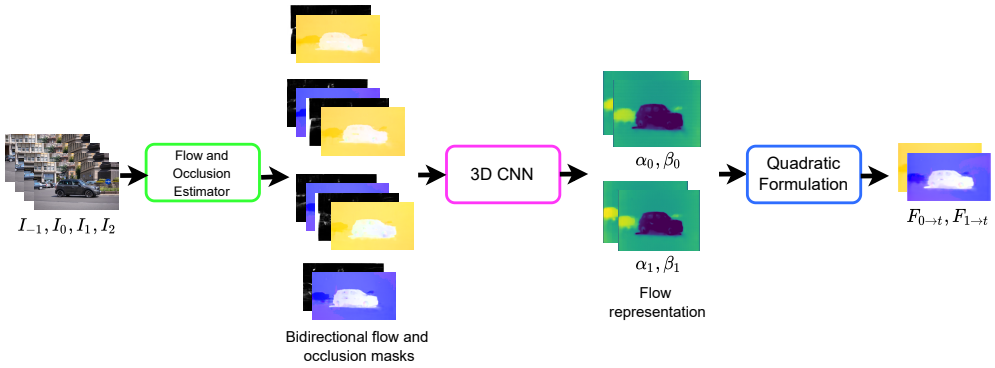

Figure 2. Non-linear motion estimation module.

for pixel-wise tasks (*e.g.*, UNet-3D [14]) adopt a single-stream Encoder-Decoder style architecture that aggregates multi-scale features by the process of sequential downsampling and skip-connection which may result in information loss [7]. Inspired by the success of GridNet [8, 23] in efficiently incorporating multi-resolution features, we formulate a novel 3D version of GridNet namely "GridNet-3D" by replacing its 2D convolutional filters with 3D convolutional filters. Additional details about the architecture is mentioned in the supplementary material.

2.2. Backward flow estimation (BFE) module

The non-linear motions ($F_{0\to t}, F_{1\to t}$) estimated in NME module are forward intermediate flows. To make use of backward warping operation [12] on the frames I_0 and I_1, we require the backward intermediate flows ($F_{t\to 0}$ and $F_{t\to 1}$) to be determined. To achieve this, we use a differentiable flow reversal layer proposed by [36] to obtain $F_{t\to 0}$ and $F_{t\to 1}$ from $F_{0\to t}$ and $F_{1\to t}$ respectively. Backward flow at a pixel position $\mathbf{x}$ is formulated as weighted average of forward flows of all pixels $\mathbf{p}$ that fall into neighborhood of pixel $\mathbf{x}$. $F_{t\to 0}$ at pixel position $\mathbf{x} = (x, y)$ is given by,

$$F_{t\to 0}(\mathbf{x}) = \frac{\sum_{\mathbf{p}+F_{0\to t}(\mathbf{p})\in N(\mathbf{x})} w(\mathbf{x}, \mathbf{p}+F_{0\to t}(\mathbf{p}))(-F_{0\to t}(\mathbf{p}))}{\sum_{\mathbf{p}+F_{0\to t}(\mathbf{p})+F_{0\to t}(\mathbf{p}))\in N(\mathbf{x})} w(\mathbf{x}, \mathbf{p})} \tag{3}$$

where $N(\mathbf{x})$ denotes a 2×2 neighborhood around $\mathbf{x}$ and $w(.,.)$ is a weighting function given by, $w(\mathbf{a}, \mathbf{b}) = e^{-||\mathbf{a}-\mathbf{b}||_2^2}$. Following similar procedure in Equation 3, $F_{t\to 1}$ is computed from $F_{1\to t}$.

2.3. Motion refinement (MR) module

To further refine the estimated backward flows ($F_{t\to 0}$ and $F_{t\to 1}$), we use a learning based motion refinement approach [36]. To this end, the refinement network takes concatenated source frames, warped frames and flow maps as input and applies a fully convolutional network to generate per-pixel offset $(\Delta x, \Delta y)$ and residuals $(r(x, y))$.

Refined optical flow, $F^r_{t\to 0}$ at pixel (x, y) is given by:

$$F^r_{t\to 0}(x, y) = F_{t\to 0}(x + \Delta x, y + \Delta y) + r(x, y) \tag{4}$$

$F_{t\to 1}$ is refined in a similar manner to obtain $F^r_{t\to 1}$. We choose GridNet-2D [8, 23] as the motion refinement network due to its superior performance.

2.4. Blending mask estimation (BME) module

The refined backward motions $F^r_{t\to 0}$ and $F^r_{t\to 1}$ are used to warp images I_0 and I_1 to yield two estimates I_{t0}, I_{t1} for interpolated frame I_t. We use a learnable CNN that takes input as the stack of warped frames and intermediate feature maps from previous step to output a soft blending mask M. The BME module consists of three convolutional layers followed by a sigmoid activation function [36] to generate the mask M.

2.5. Frame synthesis

We linearly blend the warped frame using blending mask [13] computed from the BME module. The final interpolated frame I_t is given by:

$$\hat{I}_t = \frac{(1-t) \times M \odot bw(I_0, F^r_{t\to 0}) + t \times (1-M) \odot bw(I_1, F^r_{t\to 1})}{(1-t) \times M + t \times (1-M)} \tag{5}$$

where $bw(.,.)$ denotes the backward warping function.

Table 1. Quantitative comparison with state-of-the-art methods. Best and second best scores are in red and blue respectively.

Method	Input frames	Vimeo Septuplet		DAVIS		HD		GoPro		Params (M)	Runtime (s)
		PSNR	**SSIM**	**PSNR**	**SSIM**	**PSNR**	**SSIM**	**PSNR**	**SSIM**		
SepConv [25]	2	33.04	0.9334	25.38	0.7428	30.24	0.8784	26.88	0.8166	21.6	0.024
SuperSloMo [13]	2	33.46	0.9423	25.84	0.7765	30.37	0.8834	27.31	0.8367	39.61	0.025
CAIN [5]	2	31.70	0.9106	24.89	0.7235	29.22	0.8523	26.81	0.8076	42.78	0.02
BMBC[1][26]	2	31.34	0.9054	23.50	0.6697	-	-	24.62	0.7399	11.0	0.41
Tridirectional [4]	3	32.73	0.9331	25.24	0.7476	29.84	0.8692	26.80	0.8180	10.40	0.19
QVI [36]	4	34.50	0.9521	27.36	0.8298	30.92	0.8971	28.80	0.8781	29.22	0.10
FLAVR [14]	4	33.56	0.9372	25.74	0.7589	29.96	0.8758	27.76	0.8436	42.06	0.20
Ours	4	34.99	0.9544	27.53	0.8281	31.49	0.9000	29.08	0.8826	20.92	0.32

[1] BMBC encountered out-of-memory error when tested on HD dataset.

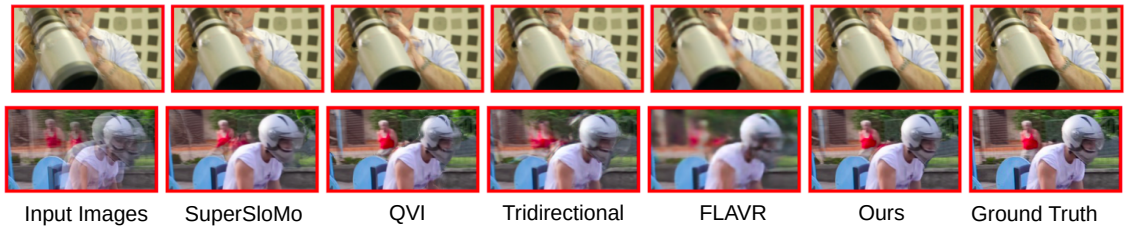

Figure 3. Qualitative comparison of our method with other state-of-the-art algorithms.

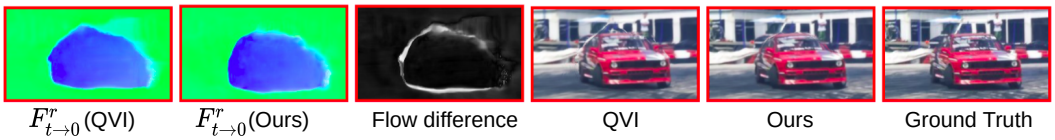

Figure 4. Intermediate flow visualization between QVI and our approach.

3. Datasets, Experiments, Results

Datasets: We have used four datasets of different image resolutions in our experiments: Vimeo Septuplet [37], DAVIS [28], HD [2] and GoPro [22]. We use training subset of Vimeo Septuplet dataset for training and evaluate the model on other datasets without fine-tuning.

Training Details: We develop our models using the Pytorch [27] framework. During training, we optimize the network using Adam optimizer [16] with the following hyper-parameters: batch size = 64, $\beta_1 = 0.9$ and $\beta_2 = 0.999$, input frame size = random crop of 256×256. The learning rate is initially set to 2×10^{-4} and is divided by a factor of 10 when the loss plateaus. The *PWCNet-Bi-Occ* network [9] is fixed until the learning rate reaches the value 2×10^{-6} and then, it is fine-tuned with the whole network. The model takes around 16 epochs to converge.

Objective Functions: Following prior work [13], we use Reconstruction loss ($\mathcal{L}_r$), Perceptual loss ($\mathcal{L}_p$), Warping loss ($\mathcal{L}_w$) and Smoothness loss ($\mathcal{L}_s$) to train our model. Our final loss is a linear combination of all the loss functions mentioned above.

$$\mathcal{L} = \lambda_r \mathcal{L}_r + \lambda_p \mathcal{L}_p + \lambda_w \mathcal{L}_w + \lambda_s \mathcal{L}_s \tag{6}$$

We choose $\lambda_r = 204$, $\lambda_p = 0.005$, $\lambda_w = 102$ and $\lambda_s = 1$. When the model is trained with low learning rate at later phase, we set λ_w and λ_s to 0.

Comparison with state-of-the-arts: We compare our model with multiple state-of-the-art methods: TOFlow [37], Sepconv-$\mathcal{L}_1$ [25], SuperSloMo [13], CAIN [5], BMBC [26], QVI [36], Tridirectional [4] and FLAVR [14]. We train these models on Vimeo-Septuplet train set with same learning rate schedule and batch size as ours for fair comparison. We use unofficial repositories of SuperSloMo [1] and Sepconv [10] to train the corresponding models. Please note, official pretrained models of other methods might produce different results due to difference in training data and training settings. During evaluation, Peak Signal-to-Noise ratio (PSNR) and Structural Similarity (SSIM) [34] are used as evaluation metric to compare performances. Quantitative comparisons with state-of-the-art methods on Vimeo, DAVIS, HD and GoPro datasets are shown in Table 1. Number of parameters and average runtime to produce a frame of resolution 256×448 on NVIDIA 1080Ti GPU for each model is also reported.

Our method achieves best PSNR and SSIM scores in Vimeo, HD and GoPro datasets. Our method performs best in PSNR and second best in SSIM metric on DAVIS dataset. Qualitative comparison with other methods is shown in Figure 3.

Intermediate flow visualizations: We visualize the backward flow $F^r_{t\to 0}$ estimated by QVI [36] and our approach in Figure 4. We notice that erroneous results in QVI's [36] interpolated frame is caused by incorrect estimation of the backward flow. However, our method remedies this by accurately estimating the backward flow as visualized in the absolute flow difference map in Figure 4.

4. Conclusion

In this paper, we presented a 3D CNN based frame interpolation algorithm in which the bi-directional flow and occlusion maps between neighboring frames are passed as input to a 3D CNN to predict per-pixel non-linear motion. This makes our network flexible to choose between linear and quadratic motion models instead of a fixed motion model as used in prior work. Our method achieves state-of-the-art results in multiple datasets. Since flow and occlusion estimates from *PWCNet-Bi-Occ* are often not accurate and hence can create a performance bottleneck in interpolation task, further research can explore whether inclusion of RGB frames as input to 3D CNN can improve the performance. Finally, flow representation estimation for cubic modeling can also be investigated in future.

Acknowledgements: This work is supported by grants from PM's fellowship for Doctoral Research (SERB, India) & Google PhD Fellowship to Arulkumar Subramaniam.

References

[1] avinashpaliwal. Super-SloMo. https://github.com/avinashpaliwal/Super-SloMo. 4

[2] Wenbo Bao, Wei-Sheng Lai, Xiaoyun Zhang, Zhiyong Gao, and Ming-Hsuan Yang. Memc-net: Motion estimation and motion compensation driven neural network for video interpolation and enhancement. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2019. 4

[3] Zhixiang Chi, Rasoul Mohammadi Nasiri, Zheng Liu, Juwei Lu, Jin Tang, and Konstantinos N Plataniotis. All at once: Temporally adaptive multi-frame interpolation with advanced motion modeling. *arXiv preprint arXiv:2007.11762*, 2020. 1

[4] Jinsoo Choi, Jaesik Park, and In So Kweon. High-quality Frame Interpolation via Tridirectional Inference. *Winter Conference on Applications of Computer Vision (WACV)*, 2021. 1, 4

[5] Myungsub Choi, Heewon Kim, Bohyung Han, Ning Xu, and Kyoung Mu Lee. Channel attention is all you need for video frame interpolation. In *AAAI*, 2020. 4

[6] John Flynn, Ivan Neulander, James Philbin, and Noah Snavely. Deepstereo: Learning to predict new views from the world's imagery. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5515–5524, 2016. 1

[7] Damien Fourure, Rémi Emonet, Elisa Fromont, Damien Muselet, Alain Tremeau, and Christian Wolf. Residual conv-deconv grid network for semantic segmentation. *arXiv preprint arXiv:1707.07958*, 2017. 3

[8] Damien Fourure, Rémi Emonet, Élisa Fromont, Damien Muselet, Alain Trémeau, and Christian Wolf. Residual conv-deconv grid network for semantic segmentation. *arXiv preprint arXiv:1707.07958*, 2017. 3

[9] Junhwa Hur and Stefan Roth. Iterative residual refinement for joint optical flow and occlusion estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5754–5763, 2019. 2, 4

[10] HyeongminLEE. pytorch-sepconv. https://github.com/HyeongminLEE/pytorch-sepconv. 4

[11] 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 *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2462–2470, 2017. 1

[12] Max Jaderberg, Karen Simonyan, Andrew Zisserman, and Koray Kavukcuoglu. Spatial transformer networks. In *Proceedings of the 28th International Conference on Neural Information Processing Systems-Volume 2*, pages 2017–2025, 2015. 3

[13] Huaizu Jiang, Deqing Sun, Varun Jampani, Ming-Hsuan Yang, Erik Learned-Miller, and Jan Kautz. Super slomo: High quality estimation of multiple intermediate frames for video interpolation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9000–9008, 2018. 1, 3, 4

[14] Tarun Kalluri, Deepak Pathak, Manmohan Chandraker, and Du Tran. Flavr: Flow-agnostic video representations for fast frame interpolation. *arXiv preprint arXiv:2012.08512*, 2020. 3, 4

[15] Alexandros Karargyris and Nikolaos Bourbakis. Three-dimensional reconstruction of the digestive wall in capsule endoscopy videos using elastic video interpolation. *IEEE transactions on Medical Imaging*, 30(4):957–971, 2010. 1

[16] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 4

[17] Hyeongmin Lee, Taeoh Kim, Tae-young Chung, Daehyun Pak, Yuseok Ban, and Sangyoun Lee. Adacof: adaptive collaboration of flows for video frame interpolation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5316–5325, 2020. 1

[18] Ziwei Liu, Raymond A Yeh, Xiaoou Tang, Yiming Liu, and Aseem Agarwala. Video frame synthesis using deep voxel flow. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4463–4471, 2017. 1

[19] Simon Meister, Junhwa Hur, and Stefan Roth. Unflow: Unsupervised learning of optical flow with a bidirectional census loss. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018. 1

[20] Simone Meyer, Abdelaziz Djelouah, Brian McWilliams, Alexander Sorkine-Hornung, Markus Gross, and Christopher Schroers. Phasenet for video frame interpolation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 498–507, 2018. 1

[21] Simone Meyer, Oliver Wang, Henning Zimmer, Max Grosse, and Alexander Sorkine-Hornung. Phase-based frame interpolation for video. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1410–1418, 2015. 1

[22] Seungjun Nah, Tae Hyun Kim, and Kyoung Mu Lee. Deep multi-scale convolutional neural network for dynamic scene deblurring. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3883–3891, 2017. 4

[23] Simon Niklaus and Feng Liu. Context-aware synthesis for video frame interpolation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1701–1710, 2018. 3

[24] Simon Niklaus, Long Mai, and Feng Liu. Video frame interpolation via adaptive convolution. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 670–679, 2017. 1

[25] Simon Niklaus, Long Mai, and Feng Liu. Video frame interpolation via adaptive separable convolution. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 261–270, 2017. 1, 4

[26] Junheum Park, Keunsoo Ko, Chul Lee, and Chang Su Kim. Bmbc: Bilateral motion estimation with bilateral cost volume for video interpolation. In *16th European Conference on Computer Vision, ECCV 2020*, pages 109–125. Springer Science and Business Media Deutschland GmbH, 2020. 1, 4

[27] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison,

Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019. 4

[28] Jordi Pont-Tuset, Federico Perazzi, Sergi Caelles, Pablo Arbeláez, Alex Sorkine-Hornung, and Luc Van Gool. The 2017 davis challenge on video object segmentation. *arXiv preprint arXiv:1704.00675*, 2017. 4

[29] Reza Pourreza and Taco Cohen. Extending neural p-frame codecs for b-frame coding. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6680–6689, 2021. 1

[30] Anurag Ranjan and Michael J Black. Optical flow estimation using a spatial pyramid network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4161–4170, 2017. 1

[31] 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*, pages 234–241. Springer, 2015. 1

[32] Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz. Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8934–8943, 2018. 1

[33] Yang Wang, Haibin Huang, Chuan Wang, Tong He, Jue Wang, and Minh Hoai. Gif2video: Color dequantization and temporal interpolation of gif images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1419–1428, 2019. 1

[34] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004. 4

[35] Chao-Yuan Wu, Nayan Singhal, and Philipp Krahenbuhl. Video compression through image interpolation. In *Proceedings of the European conference on computer vision (ECCV)*, pages 416–431, 2018. 1

[36] Xiangyu Xu, Li Siyao, Wenxiu Sun, Qian Yin, and Ming-Hsuan Yang. Quadratic video interpolation. In *NeurIPS*, 2019. 1, 2, 3, 4

[37] Tianfan Xue, Baian Chen, Jiajun Wu, Donglai Wei, and William T Freeman. Video enhancement with task-oriented flow. *International Journal of Computer Vision*, 127(8):1106–1125, 2019. 1, 4

[38] YUE Yuanchen, CAI Yunfei, and WANG Dongsheng. Gridnet-3d: A novel real-time 3d object detection algorithm based on point cloud. *Chinese Journal of Electronics*, 30(5):931–939, 2021. 1

[39] Lunan Zhou, Yaowu Chen, Xiang Tian, and Rongxin Jiang. Frame interpolation using phase and amplitude feature pyramids. In *2019 IEEE International Conference on Image Processing (ICIP)*, pages 4190–4194. IEEE, 2019. 1