MVFI-Net: Motion-aware Video Frame Interpolation Network

Xuhu Lin[1], Lili Zhao[1,2], Xi Liu[1], and Jianwen Chen[1]✉

[1] University of Electronic Science and Technology of China, Chengdu 611731, China
chenjianwen@uestc.edu.cn
[2] China Mobile Research Institute, Beijing 100032, China
zhaoliliyjy@chinamobile.com

(a) SepConv++[32] (b) GDConvNet[40] (c) MVFI-Net (Ours)

Fig. 1. Results of frame interpolation on UHD videos with large motion. The interpolated frame by using our proposed MVFI-Net is sharper with more texture details than those generated by the state-of-the-art kernel-based methods, *e.g.*, SepConv++ [32] and GDConvNet [40].

Abstract. Video frame interpolation (VFI) is to synthesize the intermediate frame given successive frames. Most existing learning-based VFI methods generate each target pixel by using the warping operation with either one predicted kernel or flow, or both. However, their performances are often degraded due to the issues on the limited direction and scope of the reference regions, especially encountering complex motions. In this paper, we propose a novel motion-aware VFI network (MVFI-Net) to address these issues. One of the key novelties of our method lies in the newly developed warping operation, *i.e.*, *motion-aware convolution* (MAC). By predicting multiple extensible temporal motion vectors (MVs) and filter kernels for each target pixel, the direction and scope could be enlarged simultaneously. Besides, we first attempt to incorporate the pyramid structure into the kernel-based VFI, which can decompose large motions into smaller scales to improve the prediction efficiency. The quantitative and qualitative experimental results have demonstrated the proposed method delivers the state-of-the-art performance on the diverse benchmarks with various resolutions. Our codes are available at https://github.com/MediaLabVFI/MVFI-Net

Keywords: Video Frame Interpolation · Video Processing

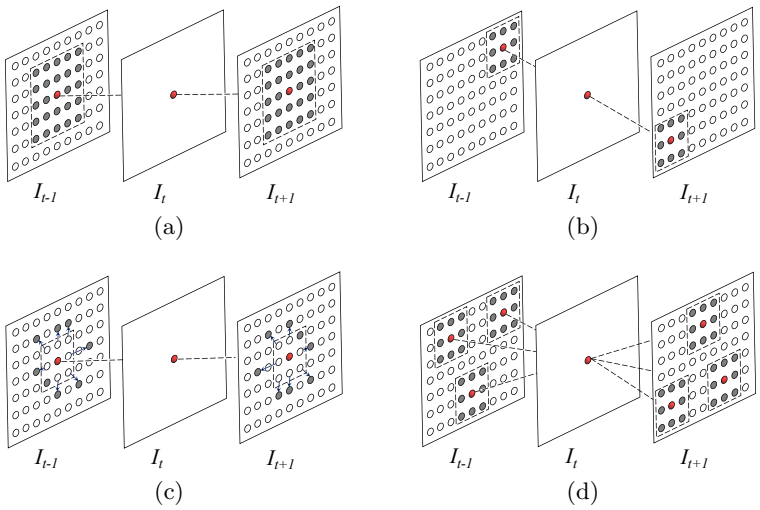


Fig. 2. Comparisons between the mainstream methods and our work. (a) The kernel-based method (*e.g.*, SepConv [31]); (b) The method based on the kernel and flow (*e.g.*, SDC-Net [37]); (c) DSepConv [7]; (d) The proposed MAC.

1 Introduction

Video frame interpolation (VFI) has been greatly demanded in many applications such as frame-rate up conversion [4,5], slow-motion generation [19,35,39] and video compression [10,16], etc. In recent years, deep learning shows its strong capacity in a series of low-level tasks, some learning-based VFI methods have been subsequently developed [40,31,37,2,7,33,29,21,41,34], which can be viewed as a two-stage process: 1) motion estimation (ME), *i.e.*, for the output pixels, finding their reference regions in reference frames; 2) motion compensation (MC), *i.e.*, synthesizing the output pixels based on ME.

Among the existing methods, one popular strategy is to conduct the optical flow estimation as ME (*e.g.*, [2,33,29,41,34]). However, the optical flow estimator will impose much computational complexity into the VFI algorithms, and the quality of the interpolated frame largely depends on the quality of flows [46]. Besides, the flow-based methods more consider the most related pixel in reference frame, while ignoring the influence of surrounding pixels, which could result in the limited direction issues.

Instead of using optical flows, some kernel-based VFI methods are developed (*e.g.*, [40,31,37,7,21]), which can be viewed as a convolution process. SepConv [31] estimated a separable kernel for each location, and then convolved these kernels with the corresponding reference regions to predict the output pixels. However, it can not deal with the motions beyond the kernel size since its scope is fixed and limited, as demonstrated in Fig. 2(a). In this case, SDC-Net [37] proposed a spatially-displaced convolution which predicted one flow and

one kernel for each location, as shown in Fig. 2(b). Then, the scope could be enlarged based on flow information, but the direction is still limited. Recently, DSepConv [7] utilized a separably deformable convolution to expand the direction of the reference regions [seeing Fig. 2(c)]. The work [21] provided the version without weight sharing as an independent warping operation namely AdaCoF. Nevertheless, the issue of limited scope is exposed again, especially for videos with a large number of complex motions.

Based on the above-mentioned discussions, there are two issues existing in VFI: 1) the direction and scope of reference regions are limited and hard to be loosed simultaneously; 2) kernel-based methods are non-robust for complex motions. To tackle the aforementioned issues, we propose a novel VFI method namely motion-aware VFI network (MVFI-Net).

For issue 1), we develop a novel warping technique called motion-aware convolution (MAC), which is one of the key novelties. In specific, multiple temporally extensible motion vectors (MVs) and corresponding spatially-varying kernels are predicted for each target pixel. Then the target pixel is calculated by convolving kernels with selected regions. The rationale behind this design is that during the motion estimation via the network, it may be possible to search MVs in a very small range and later synthesize the current pixel only based on the adjacent pixels. Obviously, the performance would be largely degraded while dealing with complex motion. In this case, we propose a motion-aware extension mechanism to adaptively extend the temporal MVs for a wider search range, and improve the efficiency of VFI. As illustrated in Fig. 2(d), compared to the previous works, MAC can explore more directions and larger scopes, which has the potential to overcome both limitations.

For issue 2), it has been known that many optical flow-based approaches have used the feature pyramid network (FPN) as the feature extractor to get multi-scale feature maps, which can be warped by internally scaling the flow in different sizes. This operation can decompose large motions into a smaller scale, which facilitates more accurate motion estimation. Thus, it is highly desirable to bring FPN into kernel-based VFI methods. However, for warping multi-scale feature maps, kernels with various sizes are required. This means that large memory is demanded, which is impractical. To solve this problem, we propose a two-stage warping strategy to warp reference frames and multi-scale features, while exploiting a frame synthesis network to mix these information for generating high-quality frame. Experimental results show that our design can improve the robustness and performance of VFI with complex motion.

Our contributions can be summarized as follows: 1) a novel warping technique MAC is proposed to simultaneously alleviate the issues of limited direction and scope of reference regions; 2) We propose a two-stage warping strategy to firstly integrate the pyramid structure into the kernel-based VFI method; 3) The proposed method delivers the state-of-the-art results on several benchmarks with various resolutions.

2 Related Work

In this section, we briefly review the works about VFI. The VFI methods can be mainly divided into three categories: 1) the phase-based methods (*e.g.*, [27,26]), 2) the optical flow-based methods (*e.g.*, [2,33,29,41,34,28,23,22,3,9]), and 3) the kernel-based methods (*e.g.*, [32,40,31,7,21,8]).

Phase-based VFI. It is commonly-known that images can be transformed to the frequency field by Fourier Transform, and this operation has been used into VFI by [27,26]. Specifically, they utilized a Phase-Net to conduct the motion estimation by the linear combinations of wavelets, which achieved competitive inference time. However, it is difficult for this category of methods to process the large motion on the high frequency components, usually imposing artifacts and pixel disappearance.

Optical flow-based VFI. In recent years, the deep learning-based optical flow estimation [36,13,17,44,45] has delivered impressive quality on motion estimation. Therefore, the optical flow estimation has been used in many subsequently developed VFI approaches (*e.g.*, [19,2,33,29,34,3,15]). For example, in [29], the interpolated frame I_t is synthesized by forward warping the input consecutive frames (I_0 and I_1) with their features, under the guidance of the estimated optical flows $t \cdot F_{0\to1}$ and $(1-t) \cdot F_{1\to0}$ via softmax splatting and a frame synthesis network. Instead of using the forward-warping method, another trend in this research have exploited the backward-warping strategy (*e.g.*, [19,2,33,3]). It is well-known that for these backward warping-based methods, the flows from the target frame to the bi-directional reference frames, denoted as $F_{t\to0}$ and $F_{t\to1}$, are unavailable. To address this issue, the algorithms presented in [19,3] approximated $F_{t\to0}$ and $F_{t\to1}$ by linearly combining bi-directional optical flows based on the target time step t. However, such motion estimation is under a basic assumption that the motion is linear and symmetric. Therefore, it can not model complex motions in real-world videos, and any initial errors imposed by flow estimation would be inevitably propagated to the subsequent processing procedures. More recently, the work proposed in [15] directly predicted the intermediate flow via a teacher-student net architecture without any prior bi-directional flow estimation. For the asymmetric motions in real-world videos, a bi-directional correlation volume algorithm [34] is proposed to modify intermediate flows, respectively, which achieves the state-of-the-art performance.

Kernel-based VFI. Consider that incredibly extra cost would be introduced for predicting prior optical flow, some works (*e.g.*, [32,31,7,21,8,30]) attempt to directly synthesize the pixel by spatially-varying convolution. For example, the work in [30] synthesized a target pixel by predicting kernels with the size of 41×41 for two reference frames, and then convolved them with the reference pixels. However, a huge amount of memory is required to store the kernels with such large size. In this case, [32,31] decomposed the convolutional kernels into two one-dimensional vectors, and then used outer product to obtain the final kernel. These two methods save much memory but can not handle videos with large motions beyond the limited kernel size. To address this issue, an added kernel is estimated for each flow to collect local appearance information [37]. Moreover,

inspired by deformable convolution, [40,8] predicted weight-sharing offsets for each element of kernels, and thus irregular motions could be described. In [21], the degree of freedom of square kernel is further improved, while it is still limited by reference scopes. Obviously, compared to optical flow-based VFI approaches, the existing kernel-based methods have drawbacks in motion estimation due to lack of prior motion guidance. Moreover, it is hard to introduce the pyramid structure into kernel-based VFI methods. This is because unlike the optical flow, if kernels are down-sampled, their weights would change completely, and there is no one-to-one correspondence between pixels. Fortunately, our work can fill up this gap.

3 Method

In this section, we first present the problem statement, and then we give an overview of our MVFI-Net. Later, more details of each module will be provided, respectively.

3.1 Problem Statement

VFI is to synthesize the temporally-consistent middle frame I_1 between two consecutive frames I_0 and I_2. An essential step is to find a transformation function $\mathcal{T}(\cdot)$ to warp reference frames based on the motion estimation (ME) results $\{\theta_0, \theta_2\}$. Therefore, the procedure of VFI can be formulated as

$$I_1 = \mathcal{T}_{\theta_0}(I_0) + \mathcal{T}_{\theta_2}(I_2). \tag{1}$$

However, it is commonly-known that some undesirable artifacts could be yielded during the aforementioned linear combination, when the target pixel is only visible in one of reference frames. This phenomenon is called as the occlusion issue. In this case, we define a soft mask [21,8] $M \in [0,1]^{H \times W}$ to tackle this problem, where $[H, W]$ is the target frame size. Then, Eq. (1) can be modified as

$$I_1 = M \odot \mathcal{T}_{\theta_0}(I_0) + (J - M) \odot \mathcal{T}_{\theta_2}(I_2), \tag{2}$$

where $\odot$ is element-wise multiplication, and $J \in R^{H \times W}$ is a matrix where each element is equal to one.

3.2 Overall Architecture

The pipeline of the proposed MVFI-Net is shown in Fig. 3, which is mainly composed of four modules: a motion estimation network MENet ($\mathcal{U}$), a novel warping technique motion-aware convolution MAC ($\mathcal{M}$), a context-pyramid feature extractor ($\mathcal{C}$) and a frame synthesis network ($\mathcal{G}$). In specific, $\mathcal{U}$ first takes two reference frames I_0 and I_2 as inputs to predict temporal MVs $\{F_{10}, F_{12}\}$, kernel weights $\{K_{10}, K_{12}\}$ and the aforementioned mask M. Concurrently, the weight-sharing $\mathcal{C}$ extracts multi-scale feature maps of $\{I_0, I_2\}$, *i.e.*, $\{c_0^0, c_0^1, c_0^2\}$ for I_0

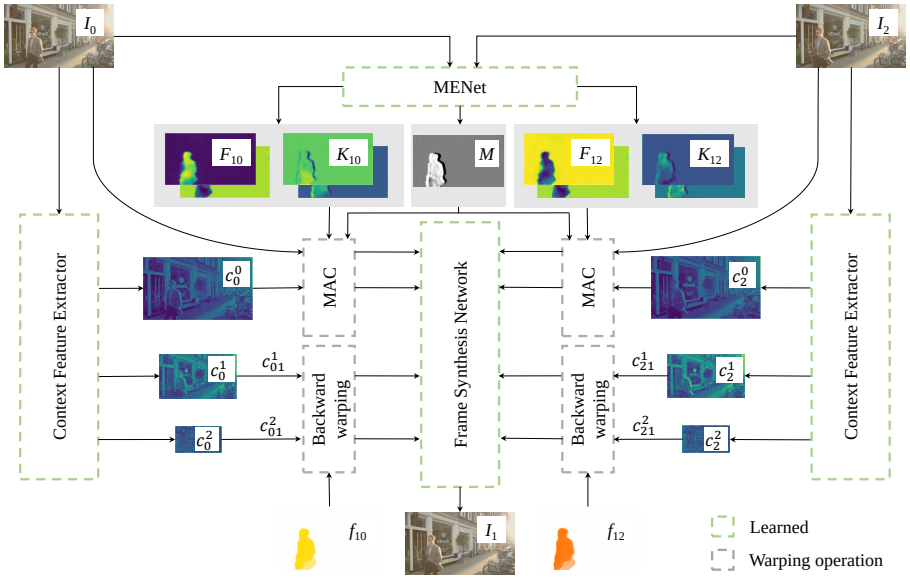


Fig. 3. The overview of our proposed MVFI-Net. The network takes two consecutive frames I_0 and I_2 as inputs, and finally generates the middle frame I_1. Note that green dotted line box represents the parameters of this part will be updated by gradient descent methods, while the gray one means non-parametric warping operation.

and $\{c_2^0, c_2^1, c_2^2\}$ for I_2. Then, the proposed two-stage warping strategy is used to align these feature maps and reference frames with time step 1. Particularly, $\mathcal{M}$ is adopted to warp $\{I_0, I_2\}$ and feature maps at the first pyramid layer $\{c_0^0, c_2^0\}$. Next, instead of predicting multi-scale kernels, the flow information $\{f_{10}, f_{12}\}$, called as *sumflow*, is calculated by weighted summation of $\{F_{10}, F_{12}\}$ along the channel axis, respectively. Later, lower-resolution feature maps $\{c_0^1, c_2^1, c_0^2, c_2^2\}$ are backward warped to the middle ones $\{c_{01}^1, c_{21}^1, c_{01}^2, c_{21}^2\}$ with the guidance of $\{f_{10}, f_{12}\}$ which are internally scaled to the corresponding size. Finally, the interpolated frame I_1 will be synthesized by $\mathcal{G}$. More details will be analyzed next.

3.3 Motion Estimation Network

As illustrated in Fig. 4, MENet ($\mathcal{U}$) is designed based on the U-Net[38] architecture, followed by nine parallel sub-nets, which are used to predict five elements: the bi-directional temporal MVs ($F_{10} = [u_{10}, v_{10}]$, $F_{12} = [u_{12}, v_{12}]$), filter kernels ($K_{10} = [k_{10}^u, k_{10}^v]$, $K_{12} = [k_{12}^u, k_{12}^v]$) and the soft mask M, where u and v represent the horizontal and vertical direction, respectively. Each prediction process $\mathcal{X}$ can be formulated by

$$\mathcal{X} = \mathcal{U}(Cat[I_0, I_2]), \tag{3}$$

where $Cat[\cdot]$ is the concatenation operation along the channel axis.

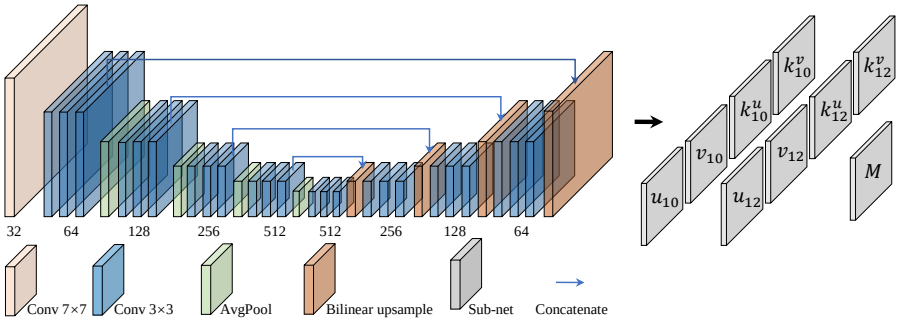


Fig. 4. Our designed MENet. Notably, each sub-net (gray box) consists of three 3×3 convolution layers and one bilinear up-sampling layer. Note that the weights across sub-nets are not sharing. Therefore, temporal MVs, filter kernels and the soft mask are predicted independently.

3.4 Motion-aware Convolution

Let $\bar{\bar{I}}$ be the target frame and I be the reference frame. Most kernel-based VFI methods (*e.g.*, [32,31]) assume that $\mathcal{T}(\cdot)$ is conducted by using the spatially-varying convolution, which is described as

$$\bar{\bar{I}}(x,y) = \sum K(x,y) \odot P_I(x,y), \tag{4}$$

where $P_I(x,y)$ is the patch centered at (x,y) in I, and the sum ($\sum$) represents the summation of all elements in Hadamard product.

Besides, $K(x,y)$ is a 2D filter kernel obtained by the outer product of two 1D vectors, which is computed as

$$K(x,y) = k^v (k^u)^T \tag{5}$$

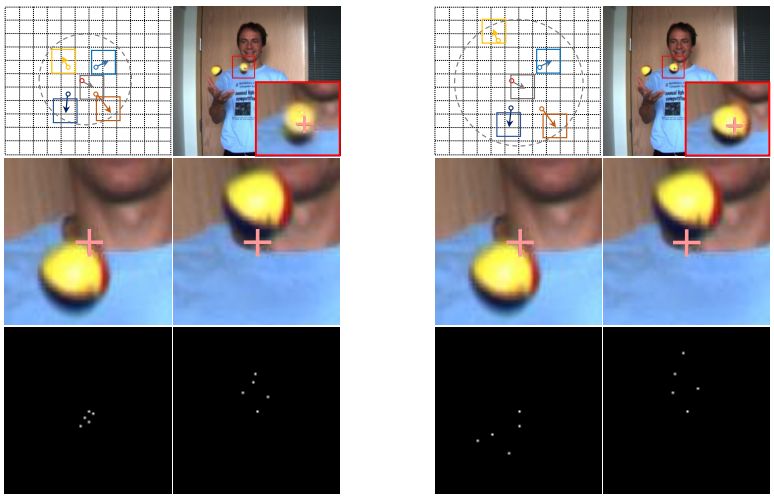
However, for these methods, the direction and scope of reference regions are limited, and thus they fail to handle large motion beyond the kernel size. In this case, we attempt to predict multiple extensible temporal MVs for each location, and the corresponding Eq. (4) can be modified as

$$\bar{\bar{I}}(x,y) = \sum_{l=0}^{L-1} \sum K^l(x,y) \odot P_I(x + u^l(x,y) + d_u(l), y + v^l(x,y) + d_v(l)), \tag{6}$$

where $u^l(x,y)$ and $v^l(x,y)$ are horizontal and vertical components of the l^{th} temporal MV, respectively, and L denotes the amount of temporal MVs. $d_u(l)$ and $d_v(l)$ are offset biases, which are adaptively calculated by our proposed motion-aware extension mechanism (MAEM) [see Fig. 5(b) top]. $d_u(l)$ can be formulated as

$$d_u(l) = l \cdot sign(u^l(x,y)), \|u^l(x,y)\| \geq \gamma, \tag{7}$$

where $sign(\cdot)$ is the signal function, while γ is the preset threshold and we empirically set $\gamma = 1$. Note that $d_v(l)$ can be calculated in the same way. The motivation behind it is that we find the traditional dilation mechanism [see Fig. 5(a) top] fails to handle irregular motions due to its fixed dilation coefficient.



(a) (b)

Fig. 5. Visualization of the effect of using the traditional dilation mechanism (TDM) and our MAEM on the search range. The first row displays the toy examples and qualitiative results given by two methods, and the second row depicts the co-location patches of two reference frames, where the pink cross denotes the co-location position of the target pixel. The third row describes the endpoint of each temporal MV from start point. (a) The result by using TDM. (b) The result by replacing TDM with our MAEM. It can be seen that our MAEM can extend the search range effectively.

To address this issue, we propose motion-aware extension mechanism (MAEM), which is able to adaptively extend temporal MV by modifying its start position on the basis of initial prediction. As shown in Fig. 5, our method can capture more accurate MVs and deliver the higher quality frame.

3.5 Multi-scale Feature Aggregation

Context-pyramid Feature Extractor. Consider that the features of frames in VFI tasks are different from those in classification tasks. Motivated by [29], we optimize a feature extractor ($\mathcal{C}$) from scratch, rather than using the existing pre-trained model (*e.g.,* VGG [42] and Resnet [14]). As depicted in Fig. 6, the features can be obtained by

$$c_t^s = \mathcal{C}(I_t)^s, \tag{8}$$

where s is the scale of pyramid, and t is the time step of reference frames.

As discussed previously, extremely large memory is required while predicting different kernels (*e.g.,* $H \times W$, $\frac{H}{2} \times \frac{W}{2}$, $\frac{H}{4} \times \frac{W}{4}$) for each scale feature map, which is impractical for applications. To address this issue, we only apply MAC ($\mathcal{M}$) for c_0^0 and c_2^0 which have the same resolutions as the target frame. Then, the

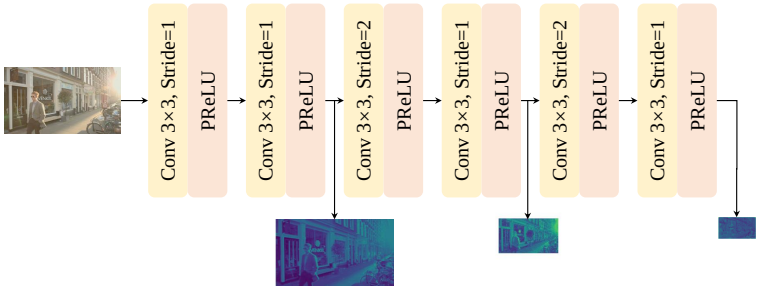


Fig. 6. The architecture of the weight-sharing context feature extractor.

warped feature map of the first layer is obtained by

$$c_1^0 = M \odot \mathcal{M}_{F_{10};K_{10}}(c_0^0) + (J - M) \odot \mathcal{M}_{F_{12};K_{12}}(c_2^0). \tag{9}$$

For c_t^1 and c_t^2, we first calculate the *sumflow* (f_{10} and f_{12}) through the predicted temporal MVs as

$$f_{10} = Cat[\sum_{l}^{L-1} u_{10}^l(x,y) + d_{u_{10}}(l), \sum_{l}^{L-1} v_{10}^l(x,y) + d_{v_{10}}(l)]. \tag{10}$$

f_{12} can be computed by the same way.

Then the backward warping operation [18] is used for the temporal alignment, which is described as

$$\begin{aligned} c_{01}^s &= backwarp((f_{10})^{\downarrow s}, c_0^s), \\ c_{21}^s &= backwarp((f_{12})^{\downarrow s}, c_2^s), \end{aligned} \tag{11}$$

where $\downarrow s$ denotes that sumflow has been downsampled to the same size with s^{th} feature map.

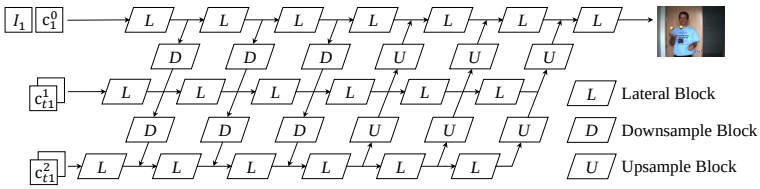


Fig. 7. The structure of the frame synthesis network.

Frame Synthesis Network. The modified version of Grid-Net [28] is used as our frame synthesis network. As depicted in Fig. 7, the inputs of the network are warped frame I_1 and feature maps $\{c_1^0, c_{01}^1, c_{21}^1, c_{01}^2, c_{21}^2\}$, and the output is the final interpolated frame.

4 Experiments

4.1 Datasets and Implementation Details

We select four benchmarks with various video resolutions for comparison, which are Vimeo90K [47] (448 × 256), UCF101 [43,24] (256 × 256), Middlebury [1] (640 × 480) and SNU-FILM [11] (1280 × 720). We next describe our training details for MVFI-Net.

Loss Functions. To evaluate the difference between the interpolated frame I_1 and its ground truth I_{gt}, we combine Charbonnier penalty function [6] with the gradient loss [25], which can facilitate generating a sharper frame.

$$\mathcal{L}_d = \lambda_1 \mathcal{L}_{char} + \lambda_2 \mathcal{L}_{gdl}, \tag{12}$$

where we empirically set $\lambda_1 = 1$, $\lambda_2 = 1$.

Training Strategy. We train the MVFI-Net for 150 epochs on the Vimeo90K training triplets, and use AdaMax [20] optimization with $\beta_1 = 0.9$, $\beta_2 = 0.999$, where the initial learning rate is set as 0.001. Note that the learning rate will be decreased by a factor of 0.5 when the validation loss does not decrease for five epochs.

Table 1. Quantitative comparisons on three benchmarks. We also calculate the inference time and MACs on Middlebury [1] 'Urban' set. All methods are tested on one NVIDIA 2080Ti GPU. For a fair comparison, each method is only trained on Vimeo90K triplets [47] by taking only two frames as reference.

	Vimeo90K[47]		UCF101[43,24]		Middlebury[1]			Runtime	MACs
	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	IE	(seconds)	(T)
SepConv-$\mathcal{L}_1$[31]	33.86	0.974	34.96	0.966	35.79	0.978	2.25	0.051	0.19
DAIN[2]	34.70	0.979	35.00	0.968	36.70	0.982	2.07	0.130	5.79
CAIN[11]	34.76	0.976	34.98	0.968	35.11	0.974	2.73	0.041	0.42
AdaCoF[21]	34.35	0.974	34.90	0.968	35.72	0.978	2.26	**0.034**	0.37
BMBC[33]	35.06	0.979	35.16	0.968	36.79	0.982	2.06	0.774	2.50
SepConv++[32]	34.83	0.977	35.27	0.968	37.28	0.984	1.96	0.110	0.14
CDFI[12]	35.17	0.978	35.21	0.967	37.14	0.983	2.01	0.221	0.26
EDSC-$\mathcal{L}_C$[8]	34.86	0.977	35.17	0.968	36.76	0.982	2.03	0.046	**0.08**
X-VFI[41]	35.07	0.977	35.08	0.968	36.71	0.982	2.05	0.097	0.14
GDConvNet[40]	34.99	0.975	35.16	0.968	35.42	0.978	2.33	1.277	0.87
MVFI-Net$_S$ (ours)	35.71	0.980	35.30	0.968	**37.51**	0.984	**1.93**	0.087	0.35
MVFI-Net$_L$ (ours)	**35.83**	**0.981**	**35.33**	**0.969**	37.48	**0.984**	1.94	0.189	0.49

4.2 Comparison with the State-of-the-Arts

To prove the effectiveness of our proposed algorithm, we compare our method with other competitive works, including SepConv [31], DSepConv [7], DAIN [2], CAIN [11], AdaCoF [21], BMBC [33], SepConv++ [32], CDFI [12], EDSC [8], X-VFI [41] and GDConvNet [40]. For evalutation metrics, we measure the performance of VFI methods in terms of Peak Signal-to-Noise Ratio (PSNR) and

Structural Similarity (SSIM). Additionally, we also calculate the widely-used Interpolation Error (IE) on Middlebury-Other set for evaluation. Note that the code of all compared methods are publicly available.

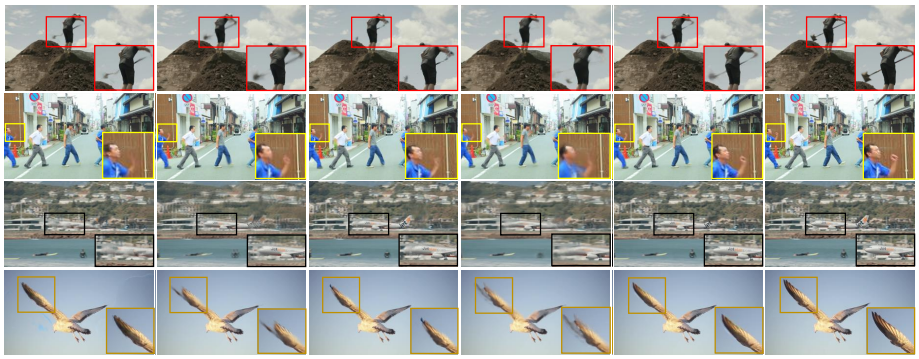
Table 2. Quantitative results of the current competitive kernel-based methods on four settings of SNU-FILM [11].

	Easy		Medium		Hard		Extreme	
	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
SepConv-$\mathcal{L}_1$[31]	39.68	0.990	35.06	0.977	29.39	0.928	24.32	0.856
DSepConv[7]	39.94	0.990	35.30	0.977	29.50	0.925	24.33	0.850
AdaCoF[21]	39.80	0.990	35.05	0.976	29.46	0.925	24.31	0.852
EDSC-$\mathcal{L}_C$[8]	40.01	0.990	35.36	0.978	29.59	0.927	24.38	0.851
CDFI[12]	40.12	0.991	35.51	0.978	29.73	0.928	24.53	0.848
MVFI-Net$_L$ (ours)	**40.17**	**0.991**	**35.57**	**0.979**	**29.86**	**0.932**	**24.62**	**0.862**

Quantitative Results. We provide two versions of MVFI-Net for comparison, which have different model sizes. Formally, MVFI-Net$_S$ represents five temporal MVs and corresponding 11×11 kernels are predicted, while MVFI-Net$_L$ possesses eleven temporal MVs and the same kernel size. As shown in Table 1, our method is far superior to others on diverse benchmarks in terms of PSNR and SSIM. Compared to the state-of-the-art kernel-based method SepConv++ [32], our lightweight model MVFI-Net$_S$ improves the PSNR by **0.88** dB on Vimeo90K with **1.5**× faster inference speed. Although the fastest VFI approach AdaCoF [21] is **2.3**× faster than us, MVFI-Net$_S$ gives **1.36** dB improvement on Vimeo90K. Note that the gains are distinct on the other two benchmarks while the MACs are similar. Compared to optical flow-based approaches, we comprehensively provide improvements either objective qualities or inference speed. This result supports that kernel-based approaches could yield impressive results without prior flow estimation.

To further prove that MVFI-Net is more robust for complex motions, we compare it with current competitive kernel-based algorithms on SNU-FILM [11] which is divided into four settings according to motions. From Table 2, it can be obviously found that our proposed MVFI-Net$_L$ delivers better performance on all settings. Generally, it is difficult to retain the structure and shape of objects in the interpolated frame when large motion exists, which imposes artifacts and blurriness with lower SSIM. Nevertheless, MVFI-Net$_L$ partly fixes this defect and supply an impressive result.

Qualitative Results. To verify the subjective quality, we also visually compare MVFI-Net with high-performance VFI approaches. As illustrated in Fig. 8, it can be seen that there are no overshoot artifacts in the frames generated by our method, while others fail. For example, in the third row and fourth row, our method clearly keep the texture details of the airplane head and the wing of the bird, while other methods impose serious artifacts and blur background.



(a) DAIN (b) SepConv++ (c) X-VFI (d) GDConvNet (e) MVFI-Net (f) GT

Fig. 8. Qualitative comparisons with DAIN [2], SepConv++ [32], XVFI [41] and GDConvNet [40] on the test set of Vimeo90K[47] and SNU-FILM[11] extreme setting, including the large motion and the occlusion issue.

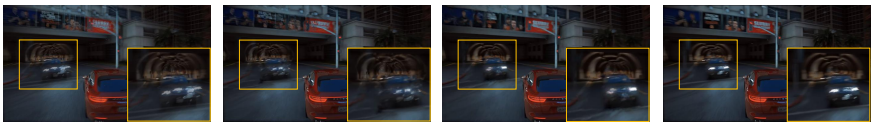
4.3 Ablation Study

In this section, we first conduct ablation studies to demonstrate the effect of our proposed motion-aware extension mechanism (MAEM) and the improvements of introducing the pyramid structure. Then, we design a series of experiments to explore the effectiveness of different amounts of temporal MVs and different kernel sizes. Finally, we attempt to transfer our method to AdaCoF [21], in which multiple flows are predicted for each target pixel, to analyze whether the performance can be improved by our algorithm.

Table 3. Ablation studies of the motion-aware extension mechanism and pyramid architecture.

			Middlebury[1]			Vimeo90K[47]		Extreme[11]	
	MAEM	Pyramid structure	PSNR	SSIM	IE	PSNR	SSIM	PSNR	SSIM
1	×	×	36.57	0.981	2.06	34.81	0.976	24.31	0.851
2	✓	×	36.84	0.983	2.02	34.83	0.976	24.39	0.852
3	✓	✓	**37.51**	**0.984**	**1.93**	**35.71**	**0.980**	**24.46**	**0.860**

MAEM and Pyramid Structure. For a fair comparison, each group is retrained under the same condition without any prior information. We first remove the pyramid structure, and then the final interpolated frame is synthesized by Eq. (2) where $\mathcal{T}(\cdot)$ is MAC. Next, we continue to remove MAEM, temporal MVs are no longer extended according to the initial prediction. From Table 3, it can be observed that our proposed MAEM, which is non-parametric and non-extra inference time cost, provides stable improvements on each benchmark in terms of two evaluation metrics. Besides, it can be demonstrated that the pyra-



(a) Overlayed (b) Baseline (c) Baseline+MAC (d) Baseline+MAC+FPN

Fig. 9. Qualitative comparisons with Baseline, Baseline+MAC and Baseline+MAC+FPN. It can be seen that the frame quality is gradually enhanced by our design.

mid structure is vital for VFI, since large motions could be decomposed into the smaller scale that can be easier predicted and captured by the network. It can be seen that there is nearly **1** dB gain on Vimeo90K and Middlebury, while supplying higher structure similarity. We also illustrate a visual result for intuitive comparison in Fig. 9.

Table 4. Ablation studies of the amounts of temporal motion vectors.

	Middlebury[1]			Vimeo90K[47]		Extreme[11]	
	PSNR	SSIM	IE	PSNR	SSIM	PSNR	SSIM
$L = 1$	36.78	0.983	2.01	35.34	0.979	24.37	0.857
$L = 5$	**37.51**	0.984	**1.93**	35.71	0.980	24.46	0.860
$L = 11$	37.48	**0.984**	1.94	**35.83**	**0.981**	**24.62**	**0.862**

Amount of Temporal MVs. As discussed above, more reference pixels are required for complex motions. Therefore, the amount of predicted temporal MVs would directly influence the performance. To verify it, we first set a fixed kernel size $N = 11$, and let the number of temporal MVs be $L \in \{1, 5, 11\}$. As displayed in Table 4, increasing temporal MVs helps the network explore more related regions, and thus the middle frame with a higher quality is synthesized. It should be noted that for Middlebury [1], the model with five temporal MVs is a little better than that with eleven temporal MVs. This can be explained by the fact that the videos in Middlebury usually have small motion. This means that the motion difference between consecutive frames is relatively small. Therefore, redundant temporal MVs may incur dispensable appearance information, leading to undesirable artifacts.

Kernel Size. It has been demonstrated that the quality of the interpolated frame is closely relevant to the size of the adaptive kernel [31]. To explore the effectiveness of the kernel size, we train several models by using the kernels with different sizes. Similar to the above experiments, we fix the number of temporal MVs $L = 5$ and modify the kernel size $N \in \{1, 5, 11\}$. Note that $N = 11$ means the 11×11 kernel is predicted for each target pixel. From Table 5, it can be observed that a larger kernel can facilitate generating a better interpolated result. However, when the kernel becomes larger (*i.e.*, $N = 11$), there is no significant

Table 5. Ablation studies of kernel size

	Middlebury[1]			Vimeo90K[47]		Extreme[11]	
	PSNR	SSIM	IE	PSNR	SSIM	PSNR	SSIM
$N = 1$	36.82	0.983	2.07	35.45	0.979	24.53	0.860
$N = 5$	37.08	0.984	1.98	35.68	0.980	**24.59**	**0.861**
$N = 11$	**37.51**	**0.984**	**1.93**	**35.71**	**0.980**	24.46	0.860

improvements. It is because our proposed MAEM has facilitated capturing motions accurately, while larger kernels have little effect and may impose repetitive local information.

Table 6. Ablation study on transfering our method to AdaCoF [21]

	Middlebury[1]			Vimeo90K[47]		Extreme[11]	
	PSNR	SSIM	IE	PSNR	SSIM	PSNR	SSIM
AdaCoF [21]	35.72	0.978	2.26	34.35	0.974	24.31	0.852
AdaCoF-*ours*	**36.23**	**0.981**	**2.22**	**35.19**	**0.979**	**24.55**	**0.860**

Transferability. In AdaCoF [21], multiple flows are predicted for each target pixel, which is similar to our algorithm. Moreover, they introduce a fixed dilation coefficient to expand the initial point of flow for a wider searching range. As demonstrated above, our MAEM could facilitate more accurate motion estimation and the pyramid structure is vital for VFI. Therefore, we attempt to transfer MAEM and the two-stage warping strategy into AdaCoF to analyze their effectiveness. Table 6 illustrates that our proposed method significantly provides stable gains for AdaCoF on the basis of each benchmark.

5 Conclusion

In this paper, we propose a novel VFI network namely MVFI-Net. There are two novelties: (1) we design an efficient warping technique MAC, where multiple extensible temporal MVs and corresponding filter kernels are predicted for each target pixel, which enlarges the direction and scope of reference region simultaneously; (2) we firstly integrate the pyramid structure into the kernel-based VFI approach, which can decompose complex motions to a smaller scale, improving the efficiency of searching for temporal MVs. Extensive simulations conducted on various datasets have demonstrated that our proposed MVFI-Net is able to consistently deliver the state-of-the-art results in terms of the objective quality and human perception.

References

1. Baker, S., Scharstein, D., Lewis, J., Roth, S., Black, M.J., Szeliski, R.: A database and evaluation methodology for optical flow. In: Proceedings of the International Journal of Computer Vision. pp. 1–8 (2007)
2. Bao, W., Lai, W.S., Ma, C., Zhang, X., Gao, Z., Yang, M.H.: Depth-aware video frame interpolation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 3698–3707 (2019)
3. Bao, W., Lai, W.S., Zhang, X., Gao, Z., Yang, M.H.: Memc-net: Motion estimation and motion compensation driven neural network for video interpolation and enhancement. IEEE Transactions on Pattern Analysis and Machine Intelligence pp. 933–948 (2019)
4. Bao, W., Zhang, X., Chen, L., Ding, L., Gao, Z.: High-order model and dynamic filtering for frame rate up-conversion. IEEE Transactions on Image Processing **27**, 3813–3826 (2018)
5. Castagno, R., Haavisto, P., Ramponi, G.: A method for motion adaptive frame rate up-conversion. IEEE Transactions on Circuits and Systems for Video Technology **6**, 436–446 (1996)
6. Charbonnier, P., Blanc-Feraud, L., Aubert, G., Barlaud, M.: Two deterministic half-quadratic regularization algorithms for computed imaging. In: Proceedings of 1st International Conference on Image Processing. pp. 168–172 (1994)
7. Cheng, X., Chen, Z.: Video frame interpolation via deformable separable convolution. In: Proceedings of the AAAI Conference on Artificial Intelligence. pp. 10607–10614 (2020)
8. Cheng, X., Chen, Z.: Multiple video frame interpolation via enhanced deformable separable convolution. IEEE Transactions on Pattern Analysis and Machine Intelligence pp. 1–1 (2021)
9. Chi, Z., Mohammadi Nasiri, R., Liu, Z., Lu, J., Tang, J., Plataniotis, K.N.: All at once: Temporally adaptive multi-frame interpolation with advanced motion modeling. In: Proceedings of the European Conference on Computer Vision. pp. 107–123 (2020)
10. Choi, H., Bajić, I.V.: Deep frame prediction for video coding. IEEE Transactions on Circuits and Systems for Video Technology **30**, 1843–1855 (2020)
11. Choi, M., Kim, H., Han, B., Xu, N., Lee, K.M.: Channel attention is all you need for video frame interpolation. In: Proceedings of the AAAI Conference on Artificial Intelligence. pp. 10663–10671 (2020)
12. Ding, T., Liang, L., Zhu, Z., Zharkov, I.: Cdfi: Compression-driven network design for frame interpolation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 7997–8007 (2021)
13. Dosovitskiy, A., Fischer, P., Ilg, E., Hausser, P., Hazirbas, C., Golkov, V., Van Der Smagt, P., Cremers, D., Brox, T.: Flownet: Learning optical flow with convolutional networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 2758–2766 (2015)
14. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016)
15. Huang, Z., Zhang, T., Heng, W., Shi, B., Zhou, S.: Rife: Real-time intermediate flow estimation for video frame interpolation. arXiv preprint arXiv:2011.06294 (2020)
16. Huo, S., Liu, D., Li, B., Ma, S., Wu, F., Gao, W.: Deep network-based frame extrapolation with reference frame alignment. IEEE Transactions on Circuits and Systems for Video Technology **31**, 1178–1192 (2021)

17. Ilg, E., Mayer, N., Saikia, T., Keuper, M., Dosovitskiy, A., Brox, T.: Flownet 2.0: Evolution of optical flow estimation with deep networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 2462–2470 (2017)
18. Jaderberg, M., Simonyan, K., Zisserman, A.: Spatial transformer networks. In: Proceedings of the Advances in Neural Information Processing Systems. pp. 2017–2025 (2015)
19. Jiang, H., Sun, D., Jampani, V., Yang, M.H., Learned-Miller, E., Kautz, J.: Super slomo: High quality estimation of multiple intermediate frames for video interpolation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 3813–3826 (2018)
20. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
21. Lee, H., Kim, T., Chung, T.y., Pak, D., Ban, Y., Lee, S.: Adacof: Adaptive collaboration of flows for video frame interpolation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 5315–5324 (2020)
22. Li, H., Yuan, Y., Wang, Q.: Video frame interpolation via residue refinement. In: IEEE International Conference on Acoustics, Speech and Signal Processing. pp. 2613–2617 (2020)
23. Liu, Y.L., Liao, Y.T., Lin, Y.Y., Chuang, Y.Y.: Deep video frame interpolation using cyclic frame generation. In: Proceedings of the AAAI Conference on Artificial Intelligence (AAAI). pp. 8794–8802 (2019)
24. Liu, Z., Yeh, R.A., Tang, X., Liu, Y., Agarwala, A.: Video frame synthesis using deep voxel flow. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 4473–4481 (2017)
25. Mathieu, M., Couprie, C., LeCun, Y.: Deep multi-scale video prediction beyond mean square error. In: arXiv preprint arXiv:1511.05440. pp. 1–14 (2016)
26. Meyer, S., Djelouah, A., McWilliams, B., Sorkine-Hornung, A., Gross, M., Schroers, C.: Phasenet for video frame interpolation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 498–507 (2018)
27. Meyer, S., Wang, O., Zimmer, H., Grosse, M., Sorkine-Hornung, A.: Phase-based frame interpolation for video. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 1410–1418 (2015)
28. Niklaus, S., Liu, F.: Context-aware synthesis for video frame interpolation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 1701–1710 (2018)
29. Niklaus, S., Liu, F.: Softmax splatting for video frame interpolation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 5436–5445 (2020)
30. Niklaus, S., Mai, L., Liu, F.: Video frame interpolation via adaptive convolution. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 670–679 (2017)
31. Niklaus, S., Mai, L., Liu, F.: Video frame interpolation via adaptive separable convolution. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 261–270 (2017)
32. Niklaus, S., Mai, L., Wang, O.: Revisiting adaptive convolutions for video frame interpolation. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. pp. 1098–1108 (2021)
33. Park, J., Ko, K., Lee, C., Kim, C.S.: Bmbc: Bilateral motion estimation with bilateral cost volume for video interpolation. In: Proceedings of the European Conference on Computer Vision. pp. 109–125 (2020)

34. Park, J., Lee, C., Kim, C.S.: Asymmetric bilateral motion estimation for video frame interpolation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 14519–14528 (2021)
35. Peleg, T., Szekely, P., Sabo, D., Sendik, O.: Im-net for high resolution video frame interpolation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 2393–2402 (2019)
36. Ranjan, A., Black, M.J.: Optical flow estimation using a spatial pyramid network. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 4161–4170 (2017)
37. Reda, F.A., Liu, G., Shih, K.J., Kirby, R., Barker, J., Tarjan, D., Tao, A., Catanzaro, B.: Sdc-net: Video prediction using spatially-displaced convolution. In: Proceedings of the European Conference on Computer Vision. pp. 718–733 (2018)
38. Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation. In: International Conference on Medical Image Computing and Computer-assisted Intervention. pp. 234–241 (2015)
39. Shen, W., Bao, W., Zhai, G., Chen, L., Min, X., Gao, Z.: Blurry video frame interpolation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 5113–5122 (2020)
40. Shi, Z., Liu, X., Shi, K., Dai, L., Chen, J.: Video frame interpolation via generalized deformable convolution. IEEE Transactions on Multimedia **20**, 426–436 (2022)
41. Sim, H., Oh, J., Kim, M.: Xvfi: extreme video frame interpolation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 14469–14478 (2021)
42. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)
43. Soomro, K., Zamir, A.R., Shah, M.: Ucf101: A dataset of 101 human actions classes from videos in the wild. arXiv preprint arXiv:1212.0402 (2012)
44. Sun, D., Yang, X., Liu, M.Y., Kautz, J.: Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 8934–8943 (2018)
45. Teed, Z., Deng, J.: Raft: Recurrent all-pairs field transforms for optical flow. In: Proceedings of the European conference on computer vision. pp. 402–419 (2020)
46. Wu, Z., Zhang, K., Xuan, H., Yang, J., Yan, Y.: Dapc-net: Deformable alignment and pyramid context completion networks for video inpainting. IEEE Signal Processing Letters **28**, 1145–1149 (2021)
47. Xue, T., Chen, B., Wu, J., Wei, D., Freeman, W.T.: Video enhancement with task-oriented flow. In: Proceedings of the International Journal of Computer Vision. pp. 1106–1128 (2019)