Received August 17, 2019, accepted August 28, 2019, date of publication September 11, 2019,
date of current version September 24, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2940510

Multi-Frame Pyramid Refinement Network for Video Frame Interpolation

HAOXIAN ZHANG[1,2], **RONGGANG WANG**[1,2], **AND YANG ZHAO**[2,3]

[1]School of Electronic and Computer Engineering, Shenzhen Graduate School, Peking University, Shenzhen 518055, China
[2]Peng Cheng Laboratory, Shenzhen 518055, China
[3]School of Computer and Information, Hefei University of Technology, Hefei 230009, China

Corresponding author: Ronggang Wang (rgwang@pkusz.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61672063 and Grant 61972129, in part by the Shenzhen Research Projects under Grant JCYJ20160506172227337 and Grant GGFW2017041215130858, and in part by the Shenzhen Key Laboratory for Intelligent Multimedia and Virtual Reality under Grant ZDSY201703031405467.

ABSTRACT Video frame interpolation aims at synthesizing new video frames in-between existing frames to generate higher frame rate video. Current methods usually use two adjacent frames to generate intermediate frames, but sometimes fail to handle challenges like large motion, occlusion, and motion blur. This paper proposes a multi-frame pyramid refinement network to effectively use spatio-temporal information contained in multiple frames (more than two). There are three technical contributions in the proposed network. First, a special coarse-to-fine framework is proposed to refine optical flows in-between multiple frames with residual flows at each pyramid level. Therefore, large motion and occlusion can be effectively estimated. Second, a 3D U-net feature extractor is used to excavate spatio-temporal context and restore texture, which tend to disappear at course pyramid levels. Third, a multi-step perceptual loss is adopted to preserve more details in intermediate frame. It is worth mentioning that our approach can be easily extended to multi-frame interpolation. Our network is trained end-to-end using more than 80K collected frame groups (25 frames per group). Experimental results on several independent datasets show that our approach can effectively handle challenging cases, and perform consistently better than other state-of-the-art methods.

INDEX TERMS Video frame interpolation, multiple frames, spatio-temporal information, optical flow, coarse-to-fine framework, deep learning.

I. INTRODUCTION

Video frame interpolation is a classic computer vision task and has a wide range of applications, such as novel view synthesis, frame rate up-conversion [1], and slow-motion video generation [2]. The traditional solution to these problems is to first estimate optical flow between frames, and then synthesize intermediate frames along with optical flow [3]. The performance of these approaches heavily depends on the quality of optical flow. In recent years, deep learning has been gradually applied to optical flow estimation and video frame interpolation tasks.

Existing CNN-based methods usually use two adjacent frames with strong correlation to generate intermediate frames [2], [4]–[9]. However, these methods sometimes fail to produce satisfactory results in some challenging cases, such as large motion, occlusion, motion blur, motion boundary, and texture-less area. Since object motion in more than two consecutive frames is generally continuous, spatio-temporal information contained in these frames can be used to further handle the above challenges. Then, how to make full use of these spatio-temporal information within these multiple frames?

A straightforward way is to train a neural network to directly generate video frames [10]–[13]. For example, Beyond MSE [12] takes four frames as input and directly hallucinate pixel values for video prediction. However, these methods fail to detect motion and result in blurry frames. While synthesizing intermediate frames through sampling pixels from source frames can effectively avoid blurring and produce high quality intermediate frames [2], [4]–[8]. Thus, we focus on applying this type of methods to multi-frame information excavation. This type of CNN-based methods need an intermediate motion estimation step before

The associate editor coordinating the review of this manuscript and approving it for publication was Gangyi Jiang.

frame interpolation. The more precise the motion estimation, the better results can be obtained. Recently, the combination of domain knowledge of optical flow, such as feature extraction and coarse-to-fine strategy, with deep learning [14], [15] has made great progresses, which gives inspiration to frame interpolation.

Inspired by recent advances in video frame interpolation and optical flow estimation, a multi-frame pyramid refinement network (MPRN) for high-quality video frame interpolation is presented in this paper. The proposed approach mainly contains three technical contributions.

First, a multi-frame pyramid refinement (MPR) framework is proposed to effectively excavate the motion and occlusion information in-between multiple frames, through predicting residual flows to refine optical flows from coarse to fine. At the top level of the pyramid, the motion can reasonably assumed to be locally linear and small around the low-resolution frames. Thus, collinear optical flows can be predicted to symmetrically sample pixels from multiple downsampled frames to synthesize intermediate frames in low resolution. The assumption above can provide a good base for the lower pyramid levels and help facilitate fast convergence. At lower levels of the pyramid, residual flows are predicted to update the upsampled flows from the previous pyramid level to gradually solve nonlinear flows in-between multiple frames. Intermediate frames are synthesized through asymmetrically sampling pixels from multiple frames along these flows in each resolution. Occlusion masks at each pyramid level are also predicted to handle occlusions. A multi-scale method is also mentioned in DVF [4], while their method uses CNNs to directly integrate motion information from different scales to the output flow. Our MPR framework can iteratively utilize motion information from previous scales to better estimate large motions. Moreover, our framework can utilize spatio-temporal information in-between more frames to address challenges like occlusion and non-linear motion.

Second, our network employs a 3D U-net [16] to extract spatio-temporal features from multiple frames. These features are downsampled by average pooling and passed to each pyramid level. It can improve robustness of our network by restoring more texture information, which easily disappear at course pyramid levels.

Third, different from original perceptual loss [17]–[19], a multi-step perceptual loss is adopted to further improve the subjective and objective quality of the interpolated frame. It is worth mentioning that our approach can be extended to multi-frame interpolation, by directly changing the channel size of last layer at each pyramid level to predict multiple optical flows for different time locations.

Our network is trained end-to-end using more than 80K collected frame groups (25 frames per group). Experimental results verify each part of our network and our network outperforms state-of-art approaches [1], [4], [7], [20]–[22] on several independent datasets, including Middlebury [3], UCF101 [23] and Thumos15 [24] test data (high-resolution videos). Note that our MPR framework with only two frames as input still achieves state-of-the-art results, especially in high-resolution videos.

II. RELATED WORK

Traditional frame interpolation approaches have two steps: optical flow estimation, and frame interpolation [3], [25], [26]. The quality of frame interpolation depends heavily on the accuracy of optical flow. In recent years, despite the great progresses in optical flow estimation [14], [15], [27]–[33], there are still some difficulties, such as obvious occlusion, motion boundary, large motion and motion blur. Mahajan *et al.* [34] compute paths in the input frames and copy pixel gradients along them to the interpolated frame, and then synthesize the intermediate frame via Poisson reconstruction. Meyer *et al.* [1] develop a phase-based approach for interpolation by propagating phase information across oriented multi-scale pyramid levels, but it is easy to loss high-frequency details in case of large motion.

As deep learning has achieved significant success in many computer vision tasks [35], [36], more researchers begin to focus on using deep learning to interpolate high quality video frames. There are a number of papers that use CNNs to directly generate intermediate frames. However, it is hard to generate high-quality intermediate frames in this way. For example, Long *et al.* [11] train a CNN that takes two consecutive original frames as input and outputs an intermediate frame which tends to be blurry.

Synthesizing intermediate frames through sampling pixels from source frames can effectively avoid blurring. Zhou *et al.* [37] develop an method that employs a convolutional neural network to estimate appearance flow and then warp input pixels to create a novel view. The deep voxel flow (DVF) approach [4] samples coherent regions of pixels from existing frames according to the voxel flow, but it sometimes still produces unsatisfactory results due to inaccuracies in voxel flow estimation. Niklaus *et al.* [6], [7] employ a CNN to learn a spatially-adaptive convolution kernel for each pixel and then convolve them with input frames to generate an intermediate frame. But their methods are limited by the size of adaptive kernels and can not handle large motion. Bao *et al.* [22] recently integrate the flow-based and kernel-based approaches into an end-to-end network to inherit the benefit from both sides. Niklaus and Liu [5] also propose a context-aware frame synthesis approach. This method estimates bidirectional flow to warp not only the input frames but also their pixel-wise contextual information, and uses them to interpolate an intermediate frame. Jiang *et al.* [2] first use a U-Net to estimate the bidirectional flow and linearly fuse two flow fields to approximate the intermediate flow fields. They then use another U-Net to refine the approximated flow fields for interpolation at multiple time steps. Ahn *et al.* [38] propose a hybrid task-based CNN for fast and accurate frame interpolation of 4K videos. Liu *et al.* [39] propose a cycle

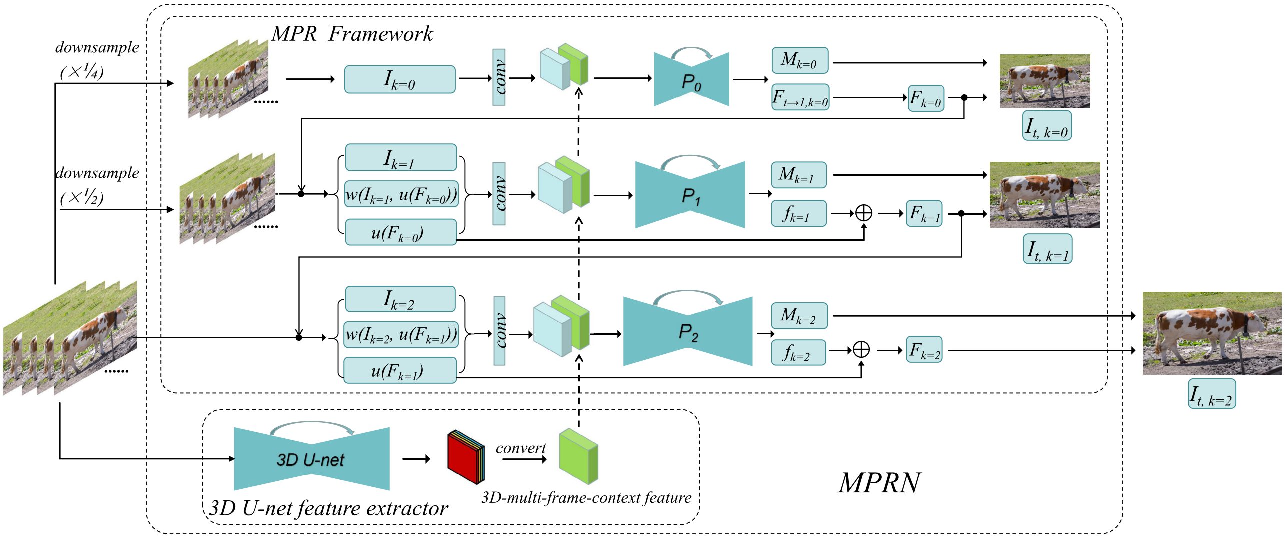

FIGURE 1. Overview of the MPRN, which contains a 3-level MPR framework and a 3D U-net feature extractor. The P_0 take 4 (or more) smallest frames as input and compute optical flow to synthesis the intermediate frame at the lowest resolution. Residual flows are predicted at lower levels of the pyramid to refine upsampled optical flow from coarse levels and output the intermediate frame at higher resolution. 3D-multi-frame-context feature extracted by a 3D U-net is downsampled by average pooling and passed to each level of MPR framework.

consistency loss to reconstruct input frames with synthesized frames, which makes synthesized frames more reliable. Deng *et al.* [40] introduce a novel self-reproducing mechanism to further substantially improve the consistency and performance of video frame interpolation.

Deep learning is also used in recent optical flow algorithms. Dosovitskiy *et al.* [28] develop two network architectures: FlowNetS and FlowNetC, which show the feasibility of estimating optical flow from raw frames using a U-Net [41] architecture. Wang *et al.* [33] presents a semantic-guided interpolation scheme (SemFlow) to handle motion boundaries and occlusions in large displacement optical flow. Recently, the combination of classical principles of optical flow with the network architecture [14], [15] achieves better results and requires less computation. Inspired by this, our network use pyramid refinement strategy to estimate motion for frame interpolation.

III. PROPOSED APPROACH

Our proposed multi-frame pyramid refinement network (MPRN) is summarized in Figure 1. Given four input frames $I = \{I_0, I_1, I_2, I_3\}$, our goal is to synthesize the intermediate frame I_t at the temporal arbitrary location t in-between I_1 and I_2. t, $(t \in (0, 1))$ refers to the time interval between the I_1 and the I_2 with the I_1 as the starting position. Let us assume $F = \{F_{t\to 0}, F_{t\to 1}, F_{t\to 2}, F_{t\to 3}\}$ to represent the predicted optical flows from I_t to I_0 ,I_1, I_2 and I_3, respectively. The intermediate frame I_t can be synthesized through warping four frames along these flow and fusing them as follows.

$$I_t = \frac{1}{4}\sum_{i=0}^{3} w(I_i, F_{t\to i}), \tag{1}$$

where $w(\cdot,\cdot)$ denotes a backward warping function, which can be implemented using bilinear interpolation [4], [8], [14] and it is differentiable. Occlusion often results in artifacts in the warped frames. To address this issue, some interpolation algorithms [2], [4], [8] estimate occlusion masks and only use pixels that are not occluded in interpolation. Hence, occlusion masks are also utilized in this paper. Let $M = \{M_{t\leftarrow 0}, M_{t\leftarrow 1}, M_{t\leftarrow 2}, M_{t\leftarrow 3}\}$ denote occlusion masks of the four warped frames. The intermediate frame I_t then can be described as follows.

$$I_t = \sum_{i=0}^{3} M_{t\leftarrow i} \otimes w(I_i, F_{t\to i}), \tag{2}$$

where $\sum_{i=0}^{3} M_{t\leftarrow i}(p) = 1$, $M_{t\leftarrow i}(p) \in [0, 1]$, and $\otimes$ denotes element-wise multiplication.

A. MPR FRAMEWORK

As illustrated in Figure 1, the multi-frame pyramid refinement (MPR) framework is a coarse-to-fine structure. It adopts the U-Net architecture [28], [41] at each pyramid level with separate weights to learn residual flow $f = \{f_{t\to 0}, f_{t\to 1}, f_{t\to 2}, f_{t\to 3}\}$ which propagates to higher resolution layers of the pyramid until high-quality optical flows are obtained at full resolution. In this way, motion can be accurately estimated, which is crucial for frame interpolation.

Let $u(\cdot)$ be the upsampling function using bilinear interpolation. Let the MPR framework has $d + 1$ levels. I_k denotes low-resolution frames downsampled 2^{d-k} times from raw input frames I at the k-th pyramid level. Let $\{P_0..P_k\}$ denotes U-nets at the different levels of pyramid. M_k, F_k and f_k denote occlusion masks M, optical flow F and residual flow f, respectively. Note that the M_k, F_k and f_k have the same resolution with I_k.

At the top level of the pyramid ($k = 0$), the P_0 takes four downsampled frames $I_{k=0}$ as input, and outputs an optical flow $F_{t\to 1,k=0}$ and four occlusion masks $M_{k=0}$. The other three collinear optical flows in $F_{k=0}$ at the top pyramid level can be described as follows.

$$\begin{aligned} F_{t\to 0,k=0} &= F_{t\to 1,k=0} \times (1+t)/t \\ F_{t\to 2,k=0} &= -F_{t\to 1,k=0} \times (1-t)/t \\ F_{t\to 3,k=0} &= -F_{t\to 1,k=0} \times (2-t)/t \end{aligned} \tag{3}$$

By downsampling original frames to sufficient low resolution, the motion of objects correspondingly becomes small and smooth. Optical flow can be reasonably assumed to be locally linear and temporally symmetric around the in-between four downsampled frames $I_{k=0}$, which can provide a good base for the lower levels of the pyramid and facilitate fast convergence of the network. Hence, the intermediate frame $I_{t,k=0}$ can be calculated through symmetrically sampling pixels from $I_{k=0}$ according to Equation (2), as shown in Figure 2 (*Left*).

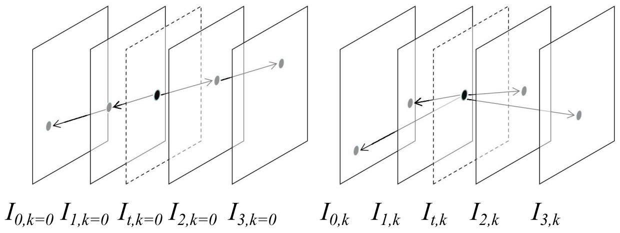

FIGURE 2. Two different interpolation ways. *Left*: Interpolation with symmetrically sampling pixels. *Right*: Interpolation with asymmetrically sampling pixels.

At the k-th level of the pyramid, different from current methods [2], [4], the MPR framework outputs residual flow f to further refine optical flow (rather than using CNN to output refined flow directly). We upsample flows from the previous pyramid level to get upsampled flows $u(F_{k-1})$, and use them to warp downsampled frames I_k to get warped frames $w(I_k, u(F_{k-1}))$. The P_k takes I_k, $u(F_{k-1})$ and $w(I_k, u(F_{k-1}))$ as input, outputs four residual flows f_k and four occlusion masks M_k. F_k at the k-th pyramid level can be described as follows.

$$F_k = u(F_{k-1}) + f_k \tag{4}$$

The assumption that optical flow is locally linear in-between four frames is hard to be correct when the resolution of the frame increases. Hence four residual flows are predicted at the k-th pyramid to update optical flows from the previous level of the pyramid. The intermediate frame $I_{t,k}$ is then calculated through asymmetrically sampling pixels from I_k according to Equation (2), as shown in Figure 2 (*Right*). By refining optical flows with residual flows at each pyramid level, large motion can be accurately estimated. Moreover, by taking four frames as input, our MPR framework can also effectively use spatio-temporal correlation between successive frames to better handle challenges, such as non-linear motion and occlusion.

In our experiments, we use a 3-level ($d = 2$) pyramid, as shown in Figure 1. But note that our framework can use more levels of pyramid, and take arbitrary even number of frames as input. Considering that the correlation between frames will decrease as the distance between frames increases, four frames are used in this paper by balancing the motion continuity and more motion information. MPR framework with only two frames as input can also achieve state-of-the-art results, especially in high resolution videos.

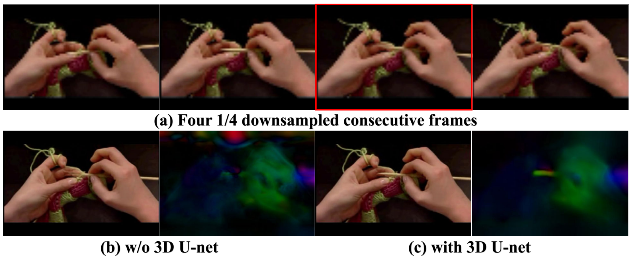

FIGURE 3. (a) Four 1/4 downsampled consecutive frames with different degrees of texture loss are used for the coarsest layer of a 3-level MPR framework. (b) and (c) are the final interpolation result and the flow from the target time position to the frame marked in red box.

B. 3D U-NET FEATURE EXTRACTOR

More spatio-temporal information are contained in four consecutive frames. 3-Dimensional convolutional networks [42], [43] is well-suited for spatio-temporal feature learning on videos. Thus, we use a 3D U-net [16] feature extractor to extract spatio-temporal feature within multiple consecutive frames, called 3D multi-frame-context feature. The 3D-multi-frame-context feature can further help MPR framework to estimate motion and restore texture. As shown in Figure 3, texture information easily disappears at course pyramid levels due to downsampling operations, especially the small or thin objects that move quickly. This has a serious impact on the MPR framework for solving optical flow, because the quality of each layer's optical flow depends on the the result of previous layers. However, the texture lost in consecutive frames is inconsistent because of the change of video. Therefore, the texture can be complemented by each other in consecutive frames, reducing the risk that texture disappears simultaneously in two adjacent frames. 3D U-net can excavate this spatio-temporal context to restore texture better, which makes it an important complement to the MPR framework. In Figure 3, the needle can be clearly seen in only one downsampled frame. But the MPR framework with 3D U-net can better solve optical flow from the target time position to the frame marked in red box, which loss the texture of the needle at course pyramid levels, and generate sharper result.

Our extractor takes a $4 \times h \times w \times 3$ (*depth* × *height* × *width* × *channel*)volume as input and outputs a $4 \times h \times w \times 8$ feature map. And then the output is converted to a $h \times w \times 32$

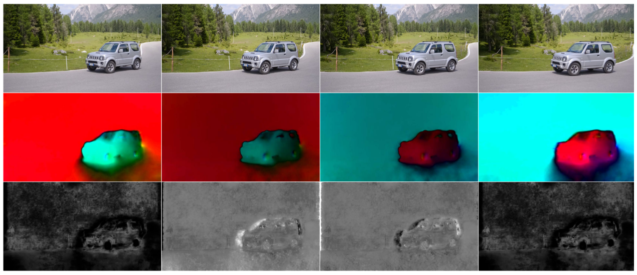

FIGURE 4. The first row are the input four frames and the second row shows predicted flows from $t = 0.5$ to the four frames. The occlusion masks are shown in the third row, the white area indicate such pixels contribute more to the synthesized frame.

3D-multi-frame-context feature through splitting the output in the depth dimension and stacking them in the channel dimension. As shown in Figure 1, the 3D-multi-frame-context feature is downsampled by average pooling and stacked with the first layer of U-net at the each level of MPR framework. Note that the pooling layers of the 3D U-net are carefully designed with the intention of not to merge the temporal signal too early. The detailed configuration of the network is described in the APPENDIX.

Visual results of final predicted flows and occlusion masks are shown in Figure 4. Our network can effectively learn additional information and predict optical flows in-between multiple frames. Occlusion mask can better handle motion boundaries and tend to sample from frames with strong spatio-temporal correlation. This indicates that nonlinear motion information between multi-frames can be helpful to refine interpolated frame from two frames. However, sampling from two frames is not always the best choice. Video frames always change over time, and the process of this change can be represented by multiple frames. This is the reason that the proposed method utilizes more neighbor frames although the most adjacent two frames have stronger spatio-temporal correlation.

C. LOSS FUNCTION

Various loss functions are considered to measure the difference between the interpolated frame $\widehat{I}_{t,k}$ and its ground truth $I_{t,k}$ at each level of the pyramid framework, to make sure residual flows have been learned at each level. Among these, we adopt a multi-step perceptual loss, which is helpful for subjective and objective quality. Let the network has $d + 1$ level. Our total loss function is a linear combination of three terms:

$$l = \lambda_r l_r + \lambda_c l_c + \lambda_s l_s \tag{5}$$

Reconstruction loss l_r [2], [4], [5] is a color-based loss function, where pixel values are normalized into the range $[-1, 1]$. The l_r models reconstruction of intermediate frame at each pyramid level:

$$l_r = \sum_{k=0}^{d} \alpha_k \cdot \left\| I_{t,k} - \widehat{I}_{t,k} \right\|_1 \tag{6}$$

Multi-step perceptual loss l_c is a feature-based loss that measures perceptual difference for pyramid framework. Usually, perceptual loss [17] utilizes the feature maps or response from pretrained network (VGG16, VGG19) [44] to extract features of final result and ground truth. More visually pleasing results can be obtained by minimizing their differences. Different from original perceptual loss, feature maps from different layers of pretrained VGG19 are used at different level of pyramid. Generally, the low-level layers of CNN tend to learn low-level semantics from frames, like corners, edges and color conjunctions. As the number of layers increases, CNN will gradually learn advanced semantic information, like texture and complex feature [45]. In our pyramid framework, the result at each level depends on the result of its previous level. At top pyramid levels, we pay more attention to motion estimation, so low level layers of pretrained VGG19 are utilized to extract local feature to better capture motion. At the bottom pyramid levels we prefer the quality of interpolation results, so high level layers of VGG19 are adopted to preserve more details and refine visual results. Multi-step perceptual loss makes our network gradually obtain higher quality. Let φ_k denotes the feature maps of an ImageNet pretrained VGG19 model used in the k level, multi-step perceptual loss l_c is define as follows.

$$l_c = \sum_{k=0}^{d} \beta_k \cdot \left\| \varphi_k(I_{t,k}) - \varphi_k(\widehat{I}_{t,k}) \right\|_2 \tag{7}$$

We will discuss and compare impact of multi-step perceptual loss and original perceptual loss in Section IV-B.

Smoothness loss l_s [2], [4]. We add spatial coherence regularizations to encourage neighboring pixels to have similar flow values, which is helpful to reduce artifacts:

$$l_s = \sum_{k=0}^{d} (\gamma_k \cdot \sum_{i=0}^{3} \left\| \nabla F_{t\rightarrow i,k} \right\|_1) \tag{8}$$

Parameters sensitivity analysis. The pyramid framework parameters $(\alpha_k, \beta_k, \gamma_k)$, should be set larger as the resolution of the pyramid layer increases.

For the parameters of the reconstruction loss (α_k) and multi-step perceptual loss (β_k), there are two main reasons. First, the last layer of the network is responsible for outputting the final result, which should be given larger weight. Second, as the resolution increases, it is more difficult to solve high quality optical flow, while the smaller resolution layer is easier to converge. Therefore, smaller parameter can be used for small resolution layers, while larger resolution layers should use larger parameter. Moreover, the results of each layer of the pyramid depend on the previous layers. This setting can effectively avoids the subsequent layers from being difficult to converge due to frequent changes in the output of the previous layers.

For the parameters of the smoothness term (γ_k), its purpose is to control the smoothness of the optical flow at each pyramid level. Small-resolution images are generally rough, and their optical flow is more discontinuous. Therefore, it is not suitable for assigning strong smoothing restrictions to the optical flow of small-resolution images. As the resolution of the image increases, we can gradually enhance the smoothing restrictions, and utilize the temporal correlation of the optical flow to make the network converge faster. This also helps to solve the optical flow in the flat region of the final result.

The parameters λ_r, λ_c and λ_s control the contribution between the three loss terms. The λ_s should be two orders of magnitude smaller than the other two parameters, otherwise it will cause most pixels to have the same flow values. The reconstruction loss and multi-step perceptual loss, as the main penalty, are equally important, which is demonstrated in Section IV-B.

In our experiments, we use a 3-level pyramid framework. So φ_0 are $conv2\_2$ features, φ_1 are $conv3\_2$ features and φ_2 are $conv4\_3$ features. And the weights are empirically set to be $\lambda_r = 1$, $\lambda_c = 0.8$, $\lambda_s = 0.025$, $\alpha_0 = 0.5$, $\alpha_1 = 0.5$, $\alpha_2 = 1$, $\beta_0 = 0.5$, $\beta_1 = 0.5$, $\beta_2 = 1$, $\gamma_0 = 0.1$, $\gamma_1 = 0.25$, $\gamma_2 = 0.5$ using validation set.

IV. EXPERIMENTS

A. TRAINING

1) TRAINING DATASET

For training, we collect 240-fps videos with a resolution of 1280×720 from YouTube and hand-held cameras [46], which contain various scenes. And then videos are split into frame groups (25 frames per frame group), among them the 1^{st}, 9^{th}, 17^{th} and 25^{th} frame serve as input, while the $8^{th} \sim 16^{th}$ frames serve as ground truth. The input of each frame group is a 30fps frame sequence. This dataset allows us to train network to interpolate frames at 7 time steps in-between two frames. Following Niklaus *et al.* [6], [7], we extract patches with a size of 480 × 480 from these frame groups and only select patches with useful information. So we calculate optical flow between 9^{th} and 17^{th} frame using DIS flow [47] to drop samples with no or little motion. In order to acquire high-quality sample, we use Laplace operator to discard bury samples and compute the entropy of patches in each sample to discard samples with little texture. Moreover, color histogram between patches is used to detect shot boundaries. Finally, 85,000 samples (frame groups) are selected. Among them, 7500 samples are used for validation and 7500 samples are used for ablation experiments.

We also include various types of data augmentation during training. The samples are cropped to patches with size of 352 × 352. Each patch is then randomly flipped vertically or horizontally, and its temporal order is randomly swapped.

2) IMPLEMENTATION DETAILS

To train our network, we initialize the weights using Gaussian distribution with standard deviation of 0.01. The network is solved via Adam [48] optimization with $\beta_1 = 0.9$, $\beta_2 = 0.999$, and the batch size is set to 8. The learning rate is initialized to be 0.0001 and decreased by a factor of 10 every 15 epochs. Batch normalization [49] is adopted for accelerating convergence. We train our model on an NVIDIA Tesla V100 GPU card, which takes about 1.5 days to converge.

We train our network on our dataset and evaluate our network on several independent datasets, including Middlebury benchmark [3], UCF101 [23] and Thumos15 [24] test data (high resolution videos). In order to compare our method with representative state-of-the-art methods, we train our network to interpolate intermediate frame at $t = 0.5$ temporal location in comparative experiments. We also verify effectiveness of multi-frame intermediate of our network. Note that a 3-level pyramid framework is used in experiments. In section IV-C, our approach is compared with the state-of-the-art methods published on Middlebury benchmark. For all these methods, we use the code or trained models from the original papers. To further verify the impact of different components in our network, a number of ablation experiments are performed in section IV-B. PSNR, SSIM [50] and the interpolation error (IE) [3] are used to evaluate the quality of interpolated video frame.

B. ABLATION EXPERIMENTS

In this section, we perform ablation studies to analyze our network. For these experiments, our testing dataset and two quality assessments of PSNR and SSIM are used.

1) IMPACT OF DIFFERENT COMPONENTS OF MULTI-FRAME PYRAMID REFINEMENT (MPR) FRAMEWORK

In this part, we train four variants of multi-frame pyramid refinement framework (MPR framework): whole MPR framework, MPR framework with 2 frame as input (MPR-2 frames), MPR framework without warped frames as input at each level of pyramid, and multi-frame pyramid framework (MP framework) which directly predicts optical flow rather than residual flow at each level of pyramid. Above models do not use feature extractor and perceptual loss.

TABLE 1. **Ablation study of MPR framework.**

	PSNR	SSIM
MPR-2 frames	30.90	0.894
MPR w/o warped pic	31.68	0.913
MP	30.67	0.900
MPR	**31.86**	**0.916**

As reported in Table 1, with the same number of frames as input, MPR outperforms the MP by 1.19dB. This verifies that residual flow prediction at each resolution can effectively excavate multi-frame information. Comparing MPR-2 frames and MPR, we can find that the use of multiple frames brings a large performance gain, which indicates that multiple frames facilitates the frame interpolation tasks. Note that our framework which only takes two frames as input (MPR-2 frames) can still obtain high-quality images, outperforming the MP by 0.23dB. Moreover, the performance degradation of MPR framework without warped frames as input at each pyramid

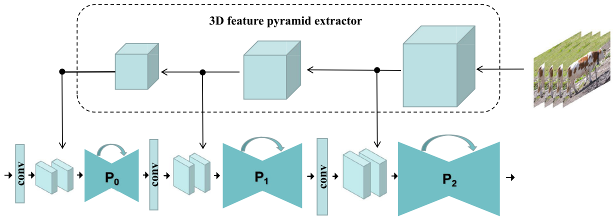

FIGURE 5. Illustration of 3D feature pyramid extractor.

level demonstrates that it is crucial for each level of pyramid to take sufficient information as input to predict residual flows.

2) EFFECTIVENESS OF 3D-MULTI-FRAME-CONTEXT FEATURE

To examine the effectiveness of 3D-multi-frame-context feature, we compare MPR with MPR using 3D-multi-frame-context feature extracted by 3D U-net feature extractor. In addition, we also test the performance of feature pyramid extractor mentioned in [15]. However, their method uses the same 2D Siamese CNN to generate feature for each frame. We therefore construct feature pyramid extractor using 3D ConvNets, called 3D feature pyramid extractor, to generate a 3-level pyramid of feature representations for four input frames, as shown in Figure 5. The output at the each pyramid level would be split in the depth dimension and warped by optical flow [15]. And then it is converted to features like 3D-multi-frame-context feature and passed to the same level of MPR. Here we use MPR as the baseline model. The above models do not use perceptual loss.

TABLE 2. Impact of different feature extractors.

	PSNR	SSIM
MPR	31.86	0.916
MPR+3D-pyramid	31.91	**0.917**
MPR+3D U-net	**31.95**	**0.917**

We can observe from Table 2 that both 3D U-net feature extractor and 3D feature pyramid extractor can provide performance gain. And 3D U-net feature extractor performs better. Although 3D-multi-frame-context feature only improves the interpolation performance slightly in PSNR, it can effectively help to improve the robustness of interpolation in some difficult cases by providing more spatial and temporal context information, as shown in Figure 6.

3) IMPACT OF MULTI-STEP PERCEPTUAL LOSS

We consider three different loss functions to train our frame synthesis neural network, as detailed in Section III-C. The previous two ablation experiments only use reconstruction loss l_r and smoothness loss l_s. Here we add original perceptual loss $l_{c\_original}$ and multi-step perceptual loss l_c to network respectively and discuss their impacts on subjective and objective quality. MPR+3D U-net is used as the baseline model here. As reported in Table 3, the objective quality of the baseline model is similar to the baseline model with original perceptual loss (*conv*4_3 features of VGG19 are used here). However, we find that the perceptual loss tends to produce visually better results, as shown in Figure 7. Multi-step perceptual loss can improve the subjective and objective quality at the same time and preserve more details of the interpolated frames compared to original perceptual loss.

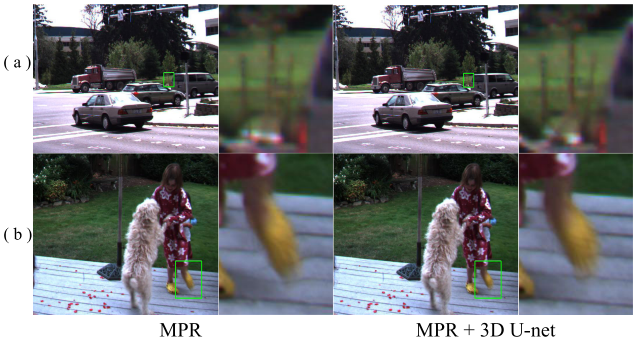

FIGURE 6. Examples for the effectiveness of 3D-multi-frame-context feature. (a) Small or thin objects that move quickly or suddenly appear in the next frame easily disappear at coarse pyramid levels, making it hard to capture their motion. (b) The edge region of a moving blurred object are usually severely distorted in predictions. 3D-multi-frame-context feature can help to restore these texture information.

TABLE 3. Impact of different perceptual loss.

	PSNR	SSIM
MPR+3D U-net	31.95	**0.917**
MPR+3D U-net+$l_{c\_original}$	31.97	0.915
MPR+3D U-net+l_c (MPRN)	**32.13**	**0.917**

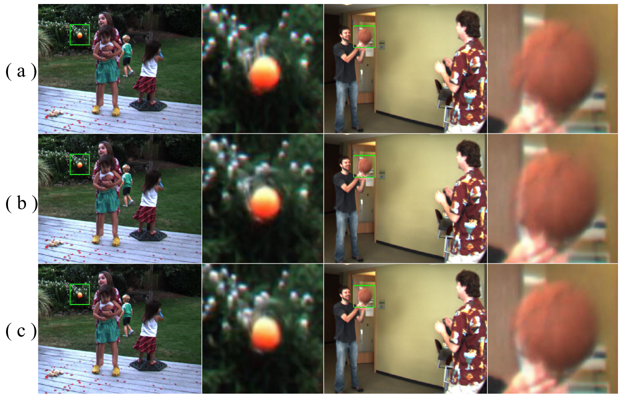

FIGURE 7. Examples of using different perceptual loss functions. (a) MPR + 3D U-net w/o any perceptual loss. (b) MPR + 3D U-net with original perceptual loss. (c) MPR + 3D U-net with multi-step perceptual loss.

4) IMPACT OF THE NUMBER OF PYRAMID LAYER

We retrain MPRN with 2-4 layers on the same train data. As reported in Table 4, the performance of MPRN is increasing as the number of pyramid layers increases. However, the

TABLE 4. Impact of the number of pyramid level.

	PSNR	SSIM
MPRN(2 layers)	30.88	0.895
MPRN(3 layers)	32.13	0.917
MPRN(4 layers)	**32.55**	**0.918**

gain from the increase in the number of pyramid layers gradually reduces. MPRN (3 layers) outperforms the MPRN (2 layers) by 1.25dB while MPRN (4 layers) only improved 0.42dB further. This indicates MPRN with three layers is sufficient for our dataset which is made from 720P video frames. When faced with larger resolution cases, we recommend more pyramid layers.

5) IMPACT OF THE NUMBER OF INPUT FRAMES UNDER DIFFERENT FRAME RATE

In general, the motion between multiple frames of high frame rate video is small, while it is large for low frame rate video. Especially in areas with complex motion, there are significant differences between multiple frames of low frame rate video. In this part, we re-create train data and valid data with 41 consecutive frames per group, respectively. Our new dataset is still 240fps. And then we extract the sequences of 30fps from new train data to retrain MPRN with six frames as input, and test two version MPRN on 30fps and 60fps sequences extracted from new valid data with the same groundtruth. As reported in Table 5, for the 30fps sequence, the correlation between frames that are far apart is low, which makes it difficult to further improve performance significantly. For the 60fps sequence, more frames with strong correlation can effectively improve the interpolation quality. Therefore, we recommend to use more frames for high frame rate video, and use four frames for videos with frame rate under 30fps by balancing the motion continuity and more motion information.

TABLE 5. Impact of the number of input frames under different frame rate.

	Test on 30fps sequence		Test on 60fps sequence	
	PSNR	SSIM	PSNR	SSIM
MPRN-4 frames	32.14	0.917	33.40	0.920
MPRN-6 frames	32.23	0.917	33.87	0.925

6) EFFECTIVENESS OF MULTI-FRAME VIDEO INTERPOLATION

Our network can be easily extended to multi-frame interpolation, by directly changing the channel size of last layer at each pyramid level to predict multiple optical flows for different t-location. We trained a multi-frame interpolation network to interpolate three frames at a time, and compared it with our single-frame interpolation network, which interpolates three frames iteratively. As reported in Table 6, predicting a set of in-between frames together can enforce the network to generate temporally coherent sequences, and avoid error propagation caused by iterative interpolation.

TABLE 6. Effectiveness of multi-frame video interpolation.

	AVERAGE		t=0.25		t=0.50		t=0.75	
	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
Single frame	32.68	**0.924**	32.95	0.927	32.13	**0.917**	32.97	0.928
Multiple frame	**32.81**	0.923	**33.14**	**0.929**	**32.15**	0.911	**33.15**	**0.929**

C. COMPARISON WITH STATE-OF-THE-ART METHODS

In this section, we compare our approach with state-of-the-art methods including MDP-Flow2 [21], DeepFlow [20], phasebased interpolation approach from Meyer *et al.* [1], separable adaptive convolution (SepConv) [7], deep voxel flow (DVF) [4] and MEMC-Net [22]. MDP-Flow2 and DeepFlow are optical flow algorithms which rank the top in the Middlebury interpolation benchmark. For these optical flow methods, we apply the interpolation algorithm presented in [3]. SepConv, DVF and MEMC-Net [22] are recent CNN-based approaches, which produce excellent interpolation results as well. A multi-scale DVF is also proposed in [4]. In order to compare different multi-scale methods, we take multi-scale DVF and MPR-2 frames model (without 3D multi-frame-context feature and multi-step perceptual loss) into consideration. Pretrained models of DVF and multi-scale DVF are not publicly available, so they are retrained using our training dataset. In addition to PSNR and SSIM, the interpolation error (IE) [3], which is defined as root-mean-square difference between the ground-truth and the prediction, is calculated to compare our approach with other state-of-the-art methods.

1) MIDDLEBURY BENCHMARK

Since the interpolation category of the Middlebury optical flow benchmark is typically used for assessing frame interpolation methods, we submit our frame interpolation results of eight sequences to its website. The IE scores from Middlebury [3] interpolation section are shown in Table 7. Our method performs well in the whole frame (all), discontinuous motion regions (disc) and textureless regions (unt), in which it is hard to predict motion. Particularly, the Teddy sequence only provides two frames as input, so we duplicate frames and pass them into our network. In spite of this, we still achieve satisfactory interpolation result. Both of these validate the robustness and generalization ability of our approach. It also validates our hypothesis that two frames with strong correlation are able to generate a reasonable result, while four frames can further improve the performance. Our approach performs well and is stable in various situations while the performance of other methods fluctuate significantly in some special regions. For example, SepConv does not perform well in textureless regions of the Teddy sequence.

TABLE 7. Evaluation on the Middlebury benchmark. disc.: Regions with discontinuous motion. unt.: Textureless regions.

	AVERAGE			Mequon			Schefflera			Urban			Teddy			Backyard			Basketball			Dumptruck			Evergreen		
	all	disc.	unt.	all	disc.	unt.	all	disc.	unt.	all	disc.	unt.	all	disc.	unt.	all	disc.	unt.	all	disc.	unt.	all	disc.	unt.	all	disc.	unt.
Ours(MPRN)	**5.38**	**8.64**	**1.98**	2.53	**4.43**	1.21	3.78	**4.97**	1.57	**3.39**	5.49	**1.28**	**5.04**	**6.58**	3.19	**9.53**	**11.9**	**3.31**	**5.25**	**9.92**	2.22	**6.87**	**15.5**	**1.49**	6.72	10.4	**1.60**
MDP-Flow2	5.83	9.69	2.15	2.89	5.38	1.19	**3.47**	5.07	**1.26**	3.66	6.10	2.48	5.20	7.48	**3.14**	10.2	12.8	3.61	6.13	11.8	2.31	7.36	16.8	**1.49**	7.75	12.1	1.69
SepConv	5.61	8.74	2.33	**2.52**	4.83	**1.11**	3.56	5.04	1.90	4.17	**4.15**	2.86	5.41	6.81	3.88	10.2	12.8	3.37	5.47	10.4	**2.21**	6.88	15.6	1.72	**6.63**	**10.3**	1.62
DeepFlow	5.96	9.78	2.04	2.98	5.67	1.22	3.88	5.78	1.52	3.62	5.93	1.34	5.39	7.20	3.17	11.0	13.9	3.63	5.91	11.3	2.29	7.14	16.3	**1.49**	7.80	12.2	1.70

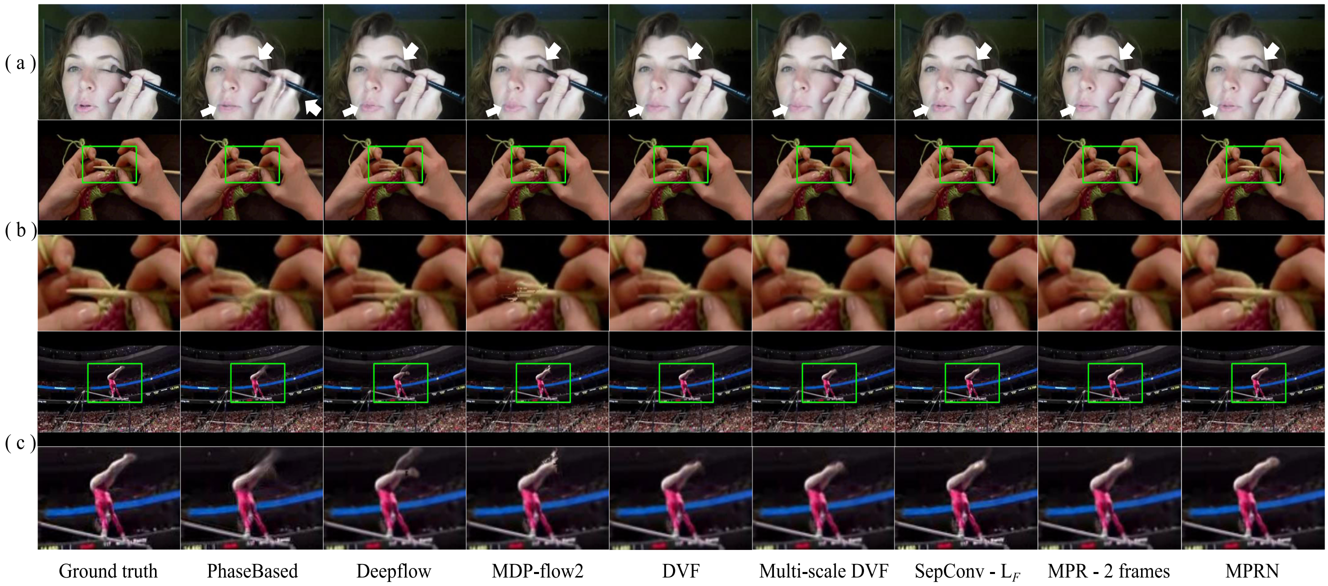

FIGURE 8. Sample interpolation results from UCF101: Our MPRN produces less artifacts around the brush and the mouse in (a). Small or thin objects that move quickly can effectively detected by our MPRN in (b). Row (c) shows our MPRN can better restore details of legs of the gymnast. Please see supplementary material for video results.

TABLE 8. Interpolation results comparison on the UCF101 dataset.

	PSNR	SSIM	IE
MDP-Flow2 [21]	34.49	0.959	6.37
DeepFlow [20]	34.40	0.957	6.44
Phase-Based [1]	33.65	0.946	6.83
SepConv-L_F [7]	34.62	0.959	6.30
DVF [4]	34.13	0.956	6.35
Multi-scale DVF [4]	34.36	0.957	6.26
MEMC-Net [22]	34.70	**0.963**	**5.95**
Ours (MPR-2 frames)	34.49	0.959	6.18
Ours (MPRN)	**34.84**	0.961	5.98

TABLE 9. Interpolation results comparison on the Thumos15 dataset.

	PSNR	SSIM	IE
DeepFlow [20]	33.65	0.946	7.67
Phase-Based [1]	32.77	0.927	8.42
SepConv-L_F [7]	33.73	0.940	7.79
DVF [4]	33.46	0.937	8.03
Multi-scale DVF [4]	33.64	0.938	7.78
MEMC-Net [22]	33.96	0.948	6.99
Ours (MPR-2 frames)	34.19	0.948	7.23
Ours (MPRN)	**34.71**	**0.951**	**6.75**

MDP-Flow2 and Deepflow perform poorly in regions with discontinuous motion of the Evergreen sequence. Furthermore, according to the feedback from the Middlebury benchmark organizer, our interpolation results outperform most published methods on the Middlebury benchmark website.

2) UCF101

Videos from UCF101 are in low resolution and relatively easy to interpolate intermediate frames. So we select samples with obvious motion using DIS optical flow [47], and evaluate these methods by computing PSNR, SSIM and IE. The quantitative results are shown in Table 8. Our MPRN has similar performance to MEMC-Net [22] on UCF101, consistently outperforming other both non-neural [1], [20], [21] and CNN-based approaches [4], [7]. Moreover, the MPR-2 frames model also achieves high performance with just two frames as input. Sample interpolation results from UCF101 can be found at Figure 8.

3) Thumos15

To compare our approach with other methods in high-resolution videos, we select videos with a resolution of 1280 × 720 from Thumos15 test data, which contains videos in various resolutions. These high-resolution videos contain

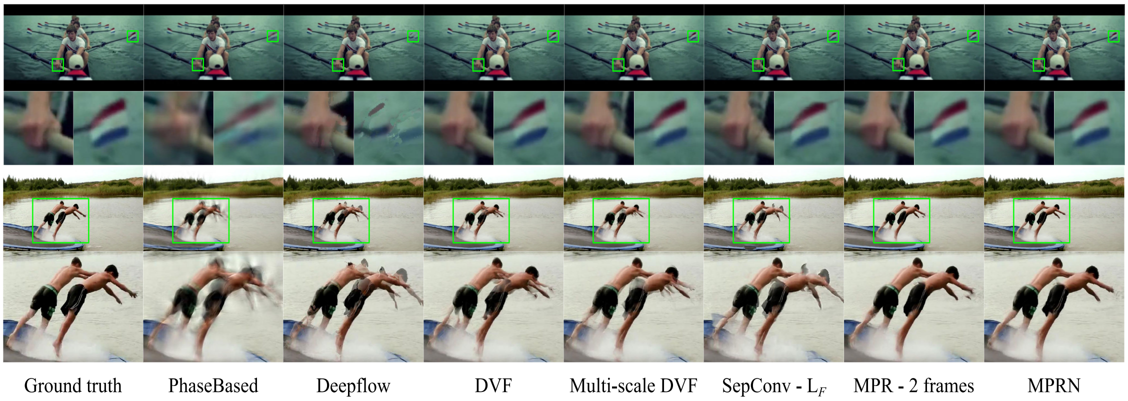

FIGURE 9. Sample interpolation results from Thumos15 (720P videos): Top two rows show our models can better handle occlusions, motion and motion edges in high-resolution videos. Bottom two rows show a challenging example with large motion. Both our models can effectively deal with this situation while MPRN perform better. Please see supplementary material for video results.

TABLE 10. Impact of different train data.

	Test on UCF101			Test on Thumos15		
	PSNR	SSIM	IE	PSNR	SSIM	IE
MPRN (trained on UCF101)	34.90	0.961	5.82	33.65	0.938	7.78
MPRN (trained on our dataset)	34.84	0.961	5.98	34.71	0.951	6.75

TABLE 11. Comparisons with other methods on parameter and runtime.

	#Parameters (million)	Runtime (seconds)
MPRN	11.7	0.15
MPRN w/o 3D U-net	10.3	0.09
SepConv [7]	21.6	0.15
MEMC-net [22]	70.3	0.10
Slomo [2]	39.6	0.14
MPRN-3 farme	11.7	0.17
Slomo-3 farme	39.6	0.40

a variety of situations, such as small and large movement, motion blur, global motion, and occlusion. Experiments here do not contain MDP-Flow2 as it is too time consuming to compute optical flow between two high-resolution frames. MDP-Flow2 is not very suited for high-resolution video frame interpolation. As reported in Table 9, both of our models, MPRN and MPR-2 frames, outperform other state-of-art methods. Compared with other frame interpolation approaches, our MPR framework is better in handling challenging large-motion and occlusion, which makes our networks to be more advantageous in high-resolution frame interpolation tasks. Qualitative comparisons are shown in Figure 9.

4) IMPACT OF DIFFERENT TRAIN DATA

To discuss the impact of different train data, we retrain our network on the UCF101 dataset and then test it on the UCF101 dataset as well as Thumos15 dataset. As reported in Table 10, MPRN trained on our dataset perform well on both datasets, while the performance of MPRN trained on UCF101 is not satisfactory on Thumos15. We can observe that the MPRN trained on UCF101 does not handle high resolution videos well, because the motion of objects in UCF101 is generally small. Therefore, it is essential to make a dataset that covers vivid natural and different amplitude motions for this task.

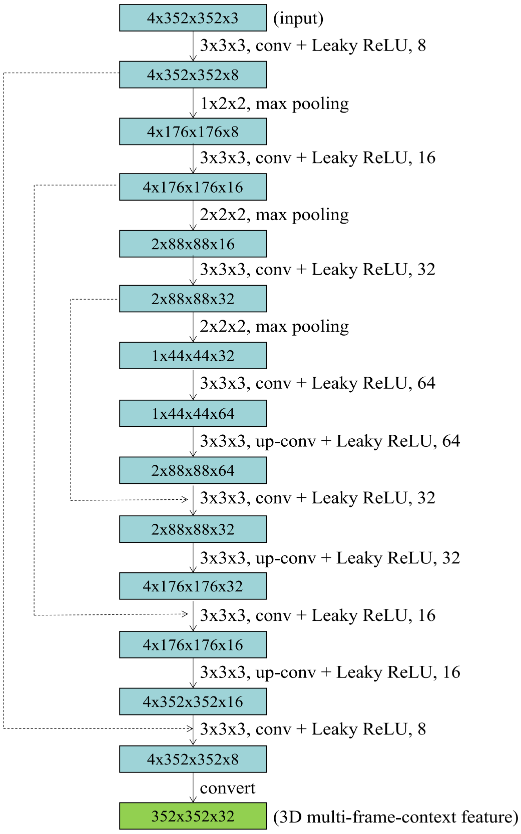
4x352x352x3
(input)
3x3x3, conv + Leaky ReLU, 8
4x352x352x8
1x2x2, max pooling
4x176x176x8
3x3x3, conv + Leaky ReLU, 16
4x176x176x16
2x2x2, max pooling
2x88x88x16
3x3x3, conv + Leaky ReLU, 32
2x88x88x32
2x2x2, max pooling
1x44x44x32
3x3x3, conv + Leaky ReLU, 64
1x44x44x64
3x3x3, up-conv + Leaky ReLU, 64
2x88x88x64
3x3x3, conv + Leaky ReLU, 32
2x88x88x32
3x3x3, up-conv + Leaky ReLU, 32
4x176x176x32
3x3x3, conv + Leaky ReLU, 16
4x176x176x16
3x3x3, up-conv + Leaky ReLU, 16
4x352x352x16
3x3x3, conv + Leaky ReLU, 8
4x352x352x8
convert
352x352x32
(3D multi-frame-context feature)

FIGURE 10. Illustration of the 3D U-net feature extractor. Blue boxes represent feature maps with size of ***depth*** × ***height*** × ***width*** × ***channel***. Dotted lines represent skip connection. 3D convolution and pooling kernels have size of ***d*** × ***k*** × ***k***, where ***d*** is kernel temporal depth and ***k*** is kernel spatial size. The last number in each convolutional layer denotes the number of kernels.

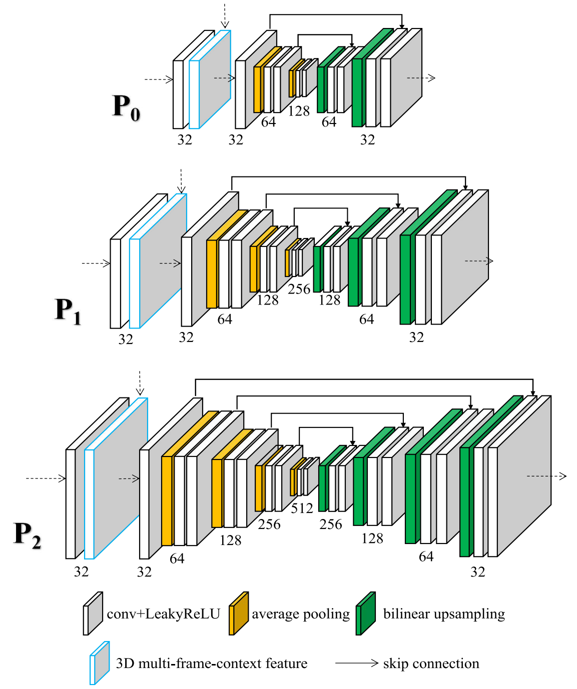

FIGURE 11. Illustration of each pyramid level in a 3-level pyramid framework. 7 × 7 convolutional kernels are used in the first two and the last two hierarchies at each level. For layers in the rest of each level, 5 × 5 kernels are used.

5) COMPUTATIONAL EFFICIENCY

The size of U-net at each pyramid level is adjusted according to the resolution. And the channel size and number of layers of 3D U-net is carefully designed to control the amount of parameters in an acceptable range, as shown in Figure 10 and Figure 11. Our approach is implemented with unoptimized Tensorflow code. We list the number of model parameters and execution time (test on a 640 × 480 image with a Tesla V100 GPU) of each method in Table 11. Compared with representative state-of-the-art methods, the MPRN can achieve better performance with similar execution time and fewer parameters, which is 46% less than SepConv [7] and 83% less than MEMC-Net [22]. When faced with low complexity requirements, MPRN without 3D U-net can furhter use fewer parameters and run faster than other method. It is worth mentioning that our multi-frame interpolation model can interpolate 3 frames at a time in only 0.17s, which is more efficient with 0.056s per frame.

V. CONCLUSION AND DISCUSSION

In this paper, we propose a multi-frame pyramid refinement network to excavate information in-between multiple frames for video frame interpolation. First, we propose a multi-frame pyramid refinement (MPR) framework which utilizes coarse-to-fine refinement framework to make full use of the correlations among multiple consecutive frames. Second, 3D-multi-frame-context features extracted from 3D U-net feature extractor are embedded in the MPR framework to improve the performance and robustness of interpolation. Third, we adopt a multi-step perceptual loss to further improve the subjective and objective quality. As demonstrated in our experiments, the proposed method can produce high-quality video frame interpolation results and outperform state-of-the-art methods.

It has been shown in recent research on image synthesis that a proper adversarial loss can help to produce high quality visual results. In the future, we think it is promising to use it to further improve the quality of frame interpolation.

We highly recommend the reviewers to check the slow motion videos generated by MPRN in the supplementary material.

APPENDIX

NETWORK DETAILS

Figure 10 shows details of the 3D U-net feature extractor used in our experiment. Figure 11 shows detailed configuration of each pyramid level in a 3-level pyramid framework.

REFERENCES

[1] S. Meyer, O. Wang, H. Zimmer, M. Grosse, and A. Sorkine-Hornung, "Phase-based frame interpolation for video," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 1410–1418.

[2] H. Jiang, D. Sun, V. Jampani, M.-H. Yang, E. Learned-Miller, and J. Kautz, "Super SloMo: High quality estimation of multiple intermediate frames for video interpolation," Nov. 2017, *arXiv:1712.00080*. [Online]. Available: https://arxiv.org/abs/1712.00080

[3] S. Baker, D. Scharstein, J. P. Lewis, S. Roth, M. J. Black, and R. Szeliski, "A database and evaluation methodology for optical flow," *Int. J. Comput. Vis.*, vol. 92, no. 1, pp. 1–31, 2011.

[4] Z. Liu, R. A. Yeh, X. Tang, Y. Liu, and A. Agarwala, "Video frame synthesis using deep voxel flow," in *Proc. ICCV*, Oct. 2017, pp. 4463–4471.

[5] S. Niklaus and F. Liu, "Context-aware synthesis for video frame interpolation," May 2018, *arXiv:1803.10967*. [Online]. Available: https://arxiv.org/abs/1803.10967

[6] S. Niklaus, L. Mai, and F. Liu, "Video frame interpolation via adaptive convolution," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, vol. 1, no. 2, pp. 670–679.

[7] S. Niklaus, L. Mai, and F. Liu, "Video frame interpolation via adaptive separable convolution," Aug. 2017, *arXiv:1708.01692*. [Online]. Available: https://arxiv.org/abs/1708.01692

[8] T. Xue, B. Chen, J. Wu, D. Wei, and W. T. Freeman, "Video enhancement with task-oriented flow," Nov. 2017, *arXiv:1711.09078*. [Online]. Available: https://arxiv.org/abs/1711.09078

[9] M. Hannemose, J. N. Jensen, G. Einarsson, J. Wilm, A. B. Dahl, and J. R. Frisvad, "Video frame interpolation via cyclic fine-tuning and asymmetric reverse flow," in *Scandinavian Conference on Image Analysis*. Sweden, Europe: Springer, 2019, pp. 311–323.

[10] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 2672–2680.

[11] G. Long, L. Kneip, J. M. Alvarez, H. Li, X. Zhang, and Q. Yu, "Learning image matching by simply watching video," in *Proc. Eur. Conf. Comput. Vis.* Amsterdam, The Netherlands: Springer, 2016, pp. 434–450.

[12] M. Mathieu, C. Couprie, and Y. LeCun, "Deep multi-scale video prediction beyond mean square error," Nov. 2015, *arXiv:1511.05440*. [Online]. Available: https://arxiv.org/abs/1511.05440

[13] N. Srivastava, E. Mansimov, and R. Salakhutdinov, "Unsupervised learning of video representations using LSTMs," in *Proc. Int. Conf. Mach. Learn.*, Jul. 2015, pp. 843–852.

[14] A. Ranjan and M. J. Black, "Optical flow estimation using a spatial pyramid network," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, vol. 2, Jul. 2017, pp. 4161–4170.

[15] D. Sun, X. Yang, M.-Y. Liu, and J. Kautz, "PWC-Net: CNNs for optical flow using pyramid, warping, and cost volume," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 8934–8943.

[16] Ö. Çiçek, A. Abdulkadir, S. S. Lienkamp, T. Brox, and O. Ronneberger, "3D U-Net: Learning dense volumetric segmentation from sparse annotation," in *Proc. Int. Conf. Med. Image Comput. Comput.-Assist. Intervent.* Athens, Greece: Springer, 2016, pp. 424–432.

[17] L. A. Gatys, A. S. Ecker, and M. Bethge, "Image style transfer using convolutional neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 2414–2423.

[18] J. Johnson, A. Alahi, and L. Fei-Fei, "Perceptual losses for real-time style transfer and super-resolution," in *Proc. Eur. Conf. Comput. Vis.* Amsterdam, The Netherlands: Springer, 2016, pp. 694–711.

[19] C. Li and M. Wand, "Precomputed real-time texture synthesis with Markovian generative adversarial networks," in *Proc. Eur. Conf. Comput. Vis.* Amsterdam, The Netherlands: Springer, 2016, pp. 702–716.

[20] P. Weinzaepfel, J. Revaud, Z. Harchaoui, and C. Schmid, "DeepFlow: Large displacement optical flow with deep matching," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2013, pp. 1385–1392.

[21] L. Xu, J. Jia, and Y. Matsushita, "Motion detail preserving optical flow estimation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 9, pp. 1744–1757, Sep. 2012.

[22] W. Bao, W.-S. Lai, X. Zhang, Z. Gao, and M.-H. Yang, "MEMC-Net: Motion estimation and motion compensation driven neural network for video interpolation and enhancement," Oct. 2018, *arXiv:1810.08768*. [Online]. Available: https://arxiv.org/abs/1810.08768

[23] K. Soomro, A. R. Zamir, and M. Shah, "UCF101: A dataset of 101 human actions classes from videos in the wild," Dec. 2012, *arXiv:1212.0402*. [Online]. Available: https://arxiv.org/abs/1212.0402

[24] H. Idrees, A. R. Zamir, Y. Jiang, A. Gorban, I. Laptev, R. Sukthankar, and M. Shah, "The THUMOS challenge on action recognition for videos 'in the wild,'" *Comput. Vis. Image Understand.*, vol. 155, pp. 1–23, 2017.

[25] M. Werlberger, T. Pock, M. Unger, and H. Bischof, "Optical flow guided TV-L^1 video interpolation and restoration," in *Proc. Int. Workshop Energy Minimization Methods Comput. Vis. Pattern Recognit.* Colorado Springs, CO, USA: Springer, 2011, pp. 273–286.

[26] Z. Yu, H. Li, Z. Wang, Z. Hu, and C. W. Chen, "Multi-level video frame interpolation: Exploiting the interaction among different levels," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 23, no. 7, pp. 1235–1248, Jul. 2013.

[27] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black, "A naturalistic open source movie for optical flow evaluation," in *Proc. Eur. Conf. Comput. Vis.* Florence, Italy: Springer, 2012, pp. 611–625.

[28] A. Dosovitskiy, P. Fischer, E. Ilg, P. Hausser, C. Hazirbas, V. Golkov, P. Van Der Smagt, D. Cremers, and T. Brox, "Flownet: Learning optical flow with convolutional networks," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2015, pp. 2758–2766.

[29] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, and T. Brox, "Flownet 2.0: Evolution of optical flow estimation with deep networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, vol. 2, Jul. 2017, pp. 2462–2470.

[30] J. Janai, F. Guney, J. Wulff, M. J. Black, and A. Geiger, "Slow flow: Exploiting high-speed cameras for accurate and diverse optical flow reference data," in *Proc. CVPR*, vol. 2, Jul. 2017, pp. 3597–3607.

[31] M. Menze and A. Geiger, "Object scene flow for autonomous vehicles," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2015, pp. 3061–3070.

[32] J. Revaud, P. Weinzaepfel, Z. Harchaoui, and C. Schmid, "Epicflow: Edge-preserving interpolation of correspondences for optical flow," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2015, pp. 1164–1172.

[33] X. Wang, D. Zhu, Y. Liu, X. Ye, J. Li, and X. Zhang, "Semflow: Semantic-driven interpolation for large displacement optical flow," *IEEE Access*, vol. 7, pp. 51589–51597, 2019.

[34] D. Mahajan, F.-C. Huang, W. Matusik, R. Ramamoorthi, and P. Belhumeur, "Moving gradients: A path-based method for plausible image interpolation," *ACM Trans. Graph.*, vol. 28, no. 3, Aug. 2009, Art. no. 42.

[35] Z. Cui, Z. Li, S. Wu, X.-Y. Zhang, and L. Wang, "Dressing as a whole: Outfit compatibility learning based on node-wise graph neural networks," in *Proc. World Wide Web Conf.*, May 2019, pp. 307–317.

[36] X.-Y. Zhang, H. Shi, C. Li, K. Zheng, X. Zhu, and L. Duan, "Learning transferable self-attentive representations for action recognition in untrimmed videos with weak supervision," in *Proc. 33rd AAAI Conf. Artif. Intell.*, Feb. 2019, pp. 1–8.

[37] T. Zhou, S. Tulsiani, W. Sun, J. Malik, and A. A. Efros, "View synthesis by appearance flow," in *Proc. Eur. Conf. Comput. Vis.* Amsterdam, The Netherlands: Springer, 2016, pp. 286–301.

[38] H.-E. Ahn, J. Jeong, and J. W. Kim, "A fast 4K video frame interpolation using a hybrid task-based convolutional neural network," *Symmetry*, vol. 11, no. 5, p. 619, 2019.

[39] Y.-L. Liu, Y.-T. Liao, Y.-Y. Lin, and Y.-Y. Chuang, "Deep video frame interpolation using cyclic frame generation," in *Proc. AAAI Conf. Artif. Intell.*, 2019, pp. 1–9.

[40] J. Deng, H. Yu, Z. Wang, X. Wang, and T. Huang, "Self-reproducing video frame interpolation," in *Proc. IEEE Conf. Multimedia Inf. Process. Retr. (MIPR)*, Mar. 2019, pp. 193–198.

[41] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional networks for biomedical image segmentation," in *Proc. Int. Conf. Med. Image Comput. Comput.-Assist. Intervent.* Munich, Germany: Springer, 2015, pp. 234–241.

[42] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, "Learning spatiotemporal features with 3d convolutional networks," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2015, pp. 4489–4497.

[43] X. Zhu, Z. Li, X.-Y. Zhang, C. Li, Y. Liu, and Z. Xue, "Residual invertible spatio-temporal network for video super-resolution," in *Proc. AAAI Conf. Artif. Intell.*, vol. 33, Jul. 2019, pp. 5981–5988.

[44] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," Sep. 2014, *arXiv:1409.1556*. [Online]. Available: https://arxiv.org/abs/1409.1556

[45] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *Proc. Eur. Conf. Comput. Vis.* Zürich, Switzerland: Springer, 2014, pp. 818–833.

[46] S. Su, M. Delbracio, J. Wang, G. Sapiro, W. Heidrich, and O. Wang, "Deep video deblurring for hand-held cameras," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, vol. 2, Jul. 2017, pp. 237–246.

[47] T. Kroeger, R. Timofte, D. Dai, and L. Van Gool, "Fast optical flow using dense inverse search," in *Proc. Eur. Conf. Comput. Vis.* Amsterdam, The Netherlands: Springer, 2016, pp. 471–488.

[48] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," Dec. 2014, *arXiv:1412.6980*. [Online]. Available: https://arxiv.org/abs/1412.6980

[49] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," Feb. 2015, *arXiv:1502.03167*. [Online]. Available: https://arxiv.org/abs/1502.03167

[50] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Trans. Image Process.*, vol. 13, no. 4, pp. 600–612, Apr. 2004.

HAOXIAN ZHANG received the B.S. degree in electronic information engineering from Central South University, China, in 2017. He is currently pursuing the M.S. degree in computer engineering with Peking University. His research interests include image processing, motion estimation, and computer vision.

RONGGANG WANG received the Ph.D. degree in computer engineering from the Institute of Computing Technology, Chinese Academy of Sciences, in 2006. He is currently a Professor in computer engineering with Peking University. His research interests include video coding techniques, 3D reconstruction, and image enhancement.

YANG ZHAO received the B.E. and Ph.D. degrees from the Department of Automation, University of Science and Technology of China, in 2008 and 2013, respectively. Since September 2013, he has been a Postdoctoral Fellow with the School of Electronic and Computer Engineering, Shenzhen Graduate School, Peking University, China. He is currently a Research Associate Professor with the School of Computer and Information, Hefei University of Technology. His research interests include image processing and pattern recognition.