Facial Video Frame Interpolation Combining Symmetric and Asymmetric Motions

Jintae Kim, Junheum Park, Whan Choi, and Chang-Su Kim
School of Electrical Engineering, Korea University, Seoul, Korea
E-mail: jtkim@mcl.korea.ac.kr, jhpark@mcl.korea.ac.kr, hwanc@mcl.korea.ac.kr, changsukim@korea.ac.kr

***Abstract*—We propose a facial video frame interpolation algorithm, which combines symmetric and asymmetric motions to complement each other. We generate multi-scale contextual features and intermediate frame candidates using both kinds of motion vectors. Then, through a frame synthesis block, we refine those intermediate frame candidates, by exploiting the contextual features, to produce the final output. Extensive experimental results demonstrate that the proposed algorithm provides state-of-the-art performance by combining both symmetric and asymmetric motion information effectively.**

I. Introduction

Video frame interpolation (VFI) is a low-level vision task to synthesize intermediate frames between successive real frames [1]–[3]. It increases the temporal resolution (*i.e.* the frame rate) of a video sequence, so it is useful in various applications such as slow-motion video [4], [5], video quality enhancement [6], and frame recovery in video streaming [7]. Recently, lots of attempts have been made to develop VFI algorithms based on convolutional neural networks, which can be classified into two categories: flow-based and kernel-based ones.

Flow-based algorithms use optical flow to interpolate intermediate frames. Optical flow estimation is a classical problem [8]; effective flow estimation methods have been proposed recently [9], [10]. Based on these optical flow estimators, many flow-based VFI algorithms have been developed. Niklaus and Liu [11] used forward warping and exploited contextual features to interpolate frames. Forward warping, however, incurs hole and occlusion problems. Hence, most VFI algorithms [3], [5], [12], [13] use backward warping, which does not cause holes or collided regions. Instead, backward warping needs bilateral motion vectors, representing the motion from an intermediate frame to real frames. Bao *et al.* [12] proposed a depth-aware VFI algorithm, which estimates bilateral motion vectors based on depth information. Park *et al.* [3] employed symmetric bilateral motion vectors based on the linear motion constraint. However, these flow-based algorithms rely heavily on optical flow. Therefore, they tend to yield poor interpolation results when optical flow is estimated unreliably.

Kernel-based algorithms convolve dynamic kernels with input frames for VFI. Niklaus *et al.* [14] proposed an adaptive convolution scheme, which selects two regions in input frames adaptively and estimates 2D kernels to process those selected regions for the interpolation. Niklaus *et al.* [4] also proposed estimating such 2D kernels more efficiently. However, these algorithms [4], [14] need large memory to estimate even small motions; they cannot estimate large motions greater than a predefined kernel size. Therefore, Lee *et al.* [15] introduced adaptive deformable convolution, called AdaCoF, to estimate larger motions than [4], [14] do, but using the same kernel size. These kernel-based algorithms, however, do not enforce any constraint on bilateral motion vectors, which are highly correlated to each other.

In this paper, we propose a kernel-based VFI algorithm combining symmetric and asymmetric motions. The proposed algorithm uses motion information more explicitly than the conventional kernel-based algorithms [4], [14], [15]. Whereas the conventional algorithms perform motion estimation and frame synthesis jointly, the proposed algorithm divides the VFI task explicitly into symmetric motion estimation, asymmetric motion estimation, and frame synthesis. First, we estimate symmetric and asymmetric motion fields from input frames and warp the input frames and contextual features using the motion fields. Then, we combine the warped frames and the warped feature pyramids, referred to as intermediate frame candidates and feature pyramid candidates, and pass them through the frame synthesis block to yield the final intermediate frame.

Recently, various researches have been carried out on facial data, such as facial landmark detection [16], [17], face super-resolution [18], image generation [19], [20], and age estimation [21], [22]. Also, the usage of facial video has increased rapidly [23]–[25]. Such usage in telemedicine, video conferencing, education, and online meetings demands high frame rates to convey the speakers' intention faithfully. Facial video, however, has not been the main interest of the VFI research, although the importance of facial VFI has risen. For this reason, we focus on facial VFI in this work.

II. Proposed Algorithm

Fig. 1 is an overview of the proposed algorithm. Given two adjacent frames I_0 and I_1, it generates an intermediate frame $I_{0.5}$. The proposed algorithm consists of five submodules: context extraction, feature extraction, asymmetric motion, symmetric motion, and frame synthesis blocks. Each motion block estimates a motion field and kernel weights and generates two intermediate frame candidates through warping. Each motion block also generates two feature pyramid candidates, which are composed of warped multi-scale contextual features. Then, the frame synthesis block combines all intermediate frame candidates and feature pyramid candidates to construct

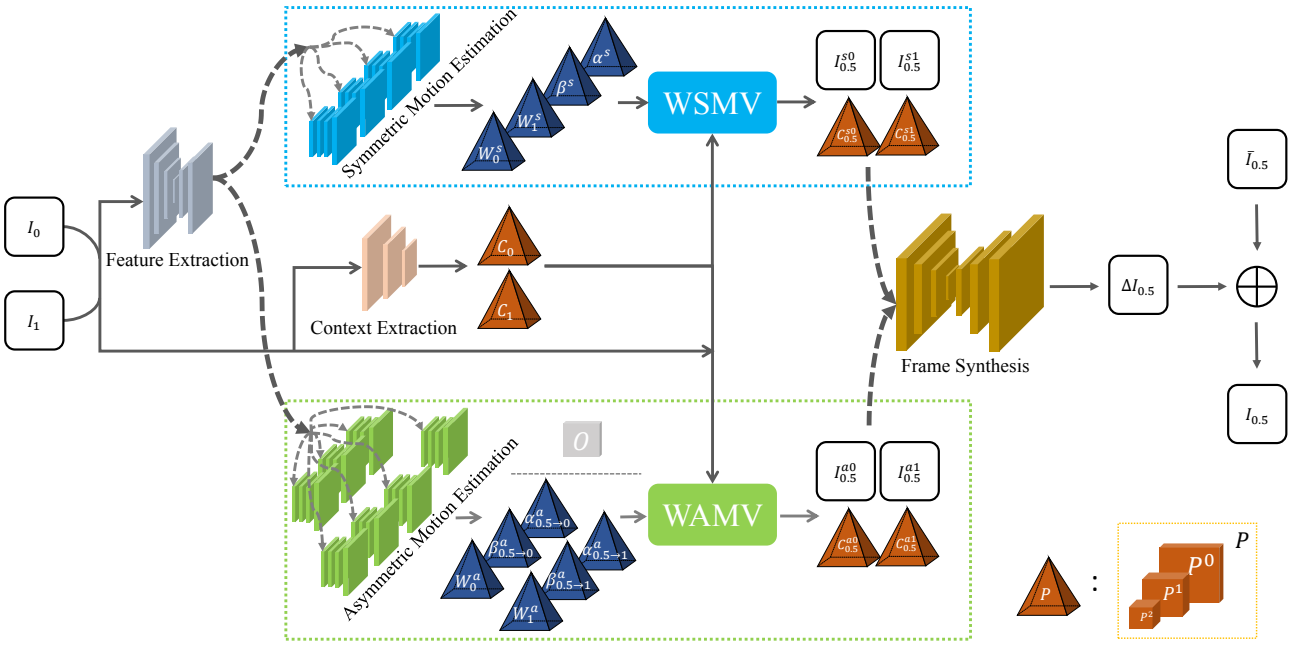


Fig. 1. An overview of the proposed algorithm: Adjacent input frames pass through the context extraction block, resulting in feature pyramids. The input frames are also processed by the feature extraction block and then the two motion blocks, respectively, to yield two kinds of intermediate frame candidates through warping. Finally, all the intermediate frame candidates are combined by the frame synthesis block to generate the final intermediate frame.

a residual frame $\Delta I_{0.5}$. Finally, $I_{0.5}$ is obtained by adding $\Delta I_{0.5}$ to $\bar{I}_{0.5}$, which is obtained from the two asymmetric intermediate frame candidates in an occlusion-aware manner. Let us describe the details of each block subsequently.

A. Context Extraction Block

We use 3×3 convolutional layers to extract contextual features from input frames. To exploit multi-scale contextual information, we generate feature pyramids. Specifically, we employ three convolutional layers to yield 8, 16, and 32 contextual features, respectively, and perform downsampling after the second and last convolutional layers. For the downsampling, we choose dilated convolution instead of pooling to obtain high-quality features [26]. We generate two feature pyramids C_0 and C_1 by processing the two input frames I_0 and I_1, respectively.

B. Feature Extraction Block

For feature extraction, we use the U-Net [27], which consists of an encoder and a decoder with skip connections. In the encoder, we use 3×3 convolutional layers and average pooling layers for downsampling. In the decoder, we also use 3×3 convolutional layers but bilinear interpolation for upsampling. The feature extraction block takes two input frames and generates features of 64 channels and of half the spatial resolution in both width and height, which are then used for both asymmetric and symmetric motion estimation.

C. Asymmetric Motion Block

In this block, the asymmetric motion estimation is first performed, and the warping via asymmetric motion vectors (WAMV) is then conducted.

Asymmetric motion estimation: We use seven subnetworks to estimate seven kinds of parameters – two asymmetric motion fields $(\alpha^a_{0.5\to0}, \beta^a_{0.5\to0})$ and $(\alpha^a_{0.5\to1}, \beta^a_{0.5\to1})$, two kernel weights W^a_0 and W^a_1, and an occlusion map O. Each subnetwork has four convolutional layers and one upsampling layer. The subnetworks for the kernel weights perform softmax activation to normalize each sum of weights to 1. The subnetwork for the occlusion map adopts sigmoid activation to normalize the range to $[0, 1]$. For all subnetworks, except for the one for the occlusion map, we perform 2×2 average pooling and 4×4 average pooling to yield multi-scale output. Then, we divide the half-resolution motion fields by 2 and the quarter-resolution motion fields by 4 for scaling.

WAMV: The WAMV layer is based on AdaCoF [15]. However, whereas AdaCoF considers a single-scale image only, WAMV considers multi-scale features, as well as the single-scale image. In AdaCoF, the intermediate frame candidate is given by

$$I_{0.5}(i,j) = \sum_{k=0}^{F-1}\sum_{l=0}^{F-1} W(i,j,k,l) \times I(i+dk+\alpha(i,j,k,l), j+dl+\beta(i,j,k,l)) \quad (1)$$

where F is the kernel size and d is the dilation term. We extend Eq. (1) to multi-scale features. For example, a feature pyramid candidate is given by

$$C^{a0,m}_{0.5}(i,j) = \sum_{k=0}^{F-1}\sum_{l=0}^{F-1} W^{a,m}_0(i,j,k,l) \times C^m_0(i+k+\alpha^{a,m}_{0.5\to0}(i,j,k,l), j+l+\beta^{a,m}_{0.5\to0}(i,j,k,l)) \quad (2)$$

where the multi-scale pyramidal level is $m \in \{1,2,3\}$. We

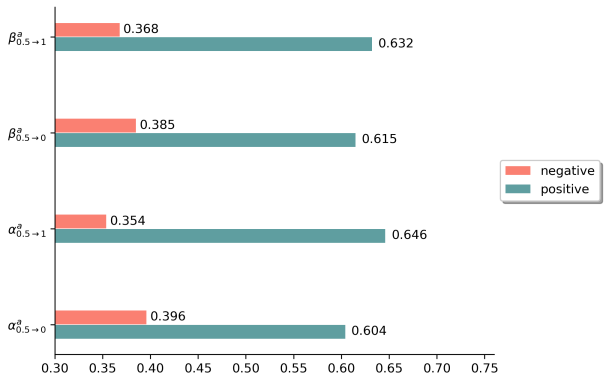


Fig. 2. Empirical probabilities that α and β in Eq. (2) have positive or negative values. We test the pre-trained model of AdaCoF [15] on the Vimeo90K test dataset [6]. We randomly choose 10,000 motion vectors from each motion field ($\alpha^a_{0.5\to0}$, $\beta^a_{0.5\to0}$) or ($\alpha^a_{0.5\to1}$, $\beta^a_{0.5\to1}$), and observe the distribution of signs. Note that the motion vectors tend to have more positive components than negative ones.

fix the dilation term to 1. The spatial resolution of C^m is $\frac{H}{2^{m-1}} \times \frac{W}{2^{m-1}}$, while the resolution of C is $H \times W$.

Eq. (2) represents the feature pyramid candidate warped from C_0. Similarly, WAMV obtains the other feature pyramid candidate $C^{a1}_{0.5}$ from C_1. Also, WAMV generates two intermediate frame candidates $I^{a0}_{0.5}$ and $I^{a1}_{0.5}$. In the cases of intermediate frame candidates, we consider only the finest level $m = 1$.

D. Symmetric Motion Block

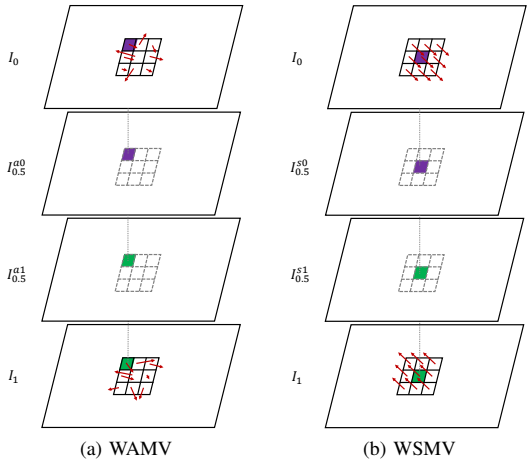
Fig. 2 demonstrates the problem of the WAMV layer. Note that the motion vectors in WAMV tend to have more positive components, which is not physically correct. To overcome this issue, we introduce the warping via symmetric motion vectors (WSMV) layer.

In this symmetric motion block, the motion field from $I_{0.5}$ to I_1 is assumed to be symmetric with that from $I_{0.5}$ to I_0. Instead of WAMV, this block uses WSMV. Fig. 3 illustrates the difference between WSMV and WAMV. Note that, compared with WAMV, WSMV enforces linear motion trajectories [3] between I_0 and I_1 using symmetric motion vectors.

Symmetric motion estimation: We employ four subnetworks to estimate four kinds of parameters – a motion field (α^s, β^s) and two kernel weights W^s_0 and W^s_1. Similarly to the asymmetric motion estimation, the subnetworks for the kernel weights perform softmax activation. Also, each subnetwork conduct 2×2 and 4×4 average pooling for multi-scale processing. For scaling, we divide the half-resolution motion field and the quarter-resolution motion field by two and four, respectively. We then define symmetric motion fields as

$$\begin{aligned} \alpha^s_{0.5\to0} &= \alpha^s, \quad \beta^s_{0.5\to0} = \beta^s, \\ \alpha^s_{0.5\to1} &= -\alpha^s, \quad \beta^s_{0.5\to1} = -\beta^s, \end{aligned} \tag{3}$$

satisfying the symmetry $\alpha^s_{0.5\to1} = -\alpha^s_{0.5\to0}$ and $\beta^s_{0.5\to1} = -\beta^s_{0.5\to0}$.



(a) WAMV (b) WSMV

Fig. 3. Comparison of WAMV with WSMV. Red arrows depict motion vectors. WAMV layer uses $2 \times F^2$ motion vectors for each pixel. In contrast, WSMV uses one pair of symmetric motion vectors for each pixel.

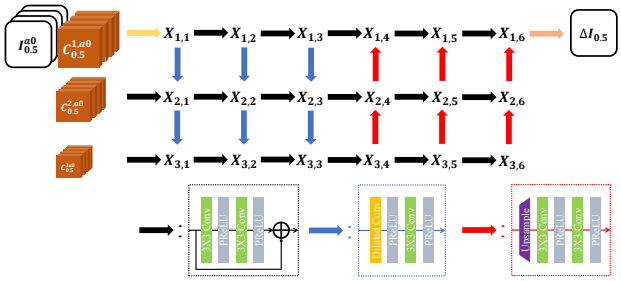


Fig. 4. The network structure of the frame synthesis block. Black arrows are residual units that do not change the resolution. Blue arrows are downsampling units including one dilated convolutional layer with $d = 2$ and one ordinary convolutional layer. Red arrows are upsampling units composed of bilinear interpolation and two convolutional layers.

WSMV: For example, the feature pyramid candidate $C^{s0}_{0.5}$ is given by

$$\begin{aligned} C^{s0,m}_{0.5}(i,j) = \sum_{k=-f}^{f} \sum_{l=-f}^{f} W^{s,m}_0(i,j,k,l) \times \\ C^m_0(i+k+\alpha^{s,m}_{0.5\to0}(i,j), j+l+\beta^{s,m}_{0.5\to0}(i,j)) \end{aligned} \tag{4}$$

where the pyramid level is $m \in \{1,2,3\}$ and $f = \frac{F-1}{2}$. Similarly to WAMV, the WSMV layer generates two feature pyramid candidates $C^{s0}_{0.5}$, $C^{s1}_{0.5}$ and two intermediate frame candidates $I^{s0}_{0.5}$, $I^{s1}_{0.5}$. For the intermediate frame candidates, we fix $m = 1$.

E. Frame Synthesis Block

We use the GridNet [29] for frame synthesis. Fig. 4 shows the network structure of the frame synthesis block, where $X_{i,j}$ represents the feature map in the ith row and jth column. Feature maps in different rows have different resolutions and channels. The three rows have the full-resolution with 32 channels, the half-resolution with 64 channels, and the quarter-resolution with 96 channels, respectively. The frame synthesis block accepts four intermediate frame candidates and four feature pyramid candidates to yield a residual frame $\Delta I_{0.5}$.

TABLE I
COMPARISON OF THE PROPOSED ALGORITHM WITH CONVENTIONAL ALGORITHMS ON THE 300VW DATASET ACCORDING TO THE DIFFICULTY LEVELS.

300VW	Whole		Very Easy		Easy		Normal		Hard	
	PSNR(↑)	SSIM(↑)	PSNR(↑)	SSIM(↑)	PSNR(↑)	SSIM(↑)	PSNR(↑)	SSIM(↑)	PSNR(↑)	SSIM(↑)
SepConv-$\mathcal{L}_1$ [4]	37.85	0.9810	42.54	0.9914	39.90	0.9872	35.65	0.9747	33.50	0.9718
SepConv-$\mathcal{L}_f$	37.44	0.9796	42.00	0.9906	39.53	0.9860	35.32	0.9731	33.05	0.9692
CyclicGen [28]	37.67	0.9799	42.38	0.9910	39.93	0.9867	35.48	0.9739	32.98	0.9688
DAIN [12]	37.97	0.9801	42.49	0.9911	40.09	0.9855	35.77	0.9730	33.73	0.9725
AdaCoF [15]	37.91	0.9811	42.54	0.9912	40.07	0.9872	35.70	0.9750	33.48	0.9717
BMBC [3]	38.12	0.9816	42.88	0.9914	40.24	0.9875	35.82	0.9752	**33.80**	**0.9731**
Proposed	**38.22**	**0.9820**	**42.91**	**0.9918**	**40.44**	**0.9881**	**35.95**	**0.9760**	33.75	0.9728

Also, we obtain $\bar{I}_{0.5}$ by

$$\bar{I}_{0.5} = O \odot I^{a0}_{0.5} + (1 - O) \odot I^{a1}_{0.5} \tag{5}$$

where $\odot$ represents the Hadamard product, and O is the occlusion map from the asymmetric motion block. Then, we reconstruct the final intermediate frame by

$$I_{0.5} = \bar{I}_{0.5} + \Delta I_{0.5}. \tag{6}$$

III. EXPERIMENTS

A. Datasets

We use the YouTube-8M dataset [30] for training. We select the comedy, academic awards, and news program classes to focus on facial videos. We download 7,327 videos and form each triplet by choosing three adjacent frames. We select less than ten triplets from each video and construct 30.6K triplets in total. All frames have a spatial resolution of 640×360. We also collect 284 YouTube videos containing interviews and music videos and choose ten triplets per video to make 2,840 triplets for the validation set.

For the test, we use the 300VW dataset [31]. We choose 44 videos with the 640×360 resolution at 24 frames per second. Then, we classify these videos into four difficulty levels according to the amounts of movements: very easy, easy, normal, and hard. We extract all the frames from the video and skip every other frame. Then, we synthesize those skipped frames and compare them with the ground-truth.

B. Training Details

The training is done by combining the reconstruction loss and the smoothness loss [5]. We use the reconstruction loss between a ground-truth frame $I^{GT}_{0.5}$ and an output frame $I_{0.5}$, given by

$$\mathcal{L}_r = \rho(I^{GT}_{0.5} - I_{0.5}) \tag{7}$$

where $\rho(x) = (x^2 + \epsilon^2)^{\frac{1}{2}}$ with $\epsilon = 0.001$ is Charbonnier penalty function [32]. We also employ the smoothness loss to encourage neighboring pixels to have similar motion vectors, which is given by

$$\begin{aligned}\mathcal{L}_s =& \rho(\nabla(W_0 \odot \alpha_{0.5\to0})) + \rho(\nabla(W_1 \odot \alpha_{0.5\to1})) \\ &+ \rho(\nabla(W_0 \odot \beta_{0.5\to0})) + \rho(\nabla(W_1 \odot \beta_{0.5\to1}))\end{aligned} \tag{8}$$

where ∇ is the gradient operator and $\odot$ denotes the Hadamard product. Then the total loss are defined as a weighted sum

$$\mathcal{L}_{\text{total}} = \mathcal{L}_r + \lambda \mathcal{L}_s \tag{9}$$

where λ = 0.01.

We pre-train the asymmetric motion block and the symmetric motion block, respectively, for 0.2M iterations to minimize $\mathcal{L}_{\text{total}}$ in Eq. (9). We use the Adamax optimizer with a batch size of 8 for the pre-training. During the pre-training of the asymmetric motion block, the WAMV layer generates intermediate candidates and yields an output frame via Eq. (5). In contrast, the symmetric motion block normally does not produce an occlusion map, but we estimate an occlusion map during its pre-training. Then, the symmetric motion block generates an intermediate frame candidate

$$I_{0.5} = O^s \odot I^{s0}_{0.5} + (1 - O^s) \odot I^{s1}_{0.5} \tag{10}$$

where O^s represents the occlusion map from the symmetric motion block during the pre-training. Finally, we fine-tune the overall network using $\mathcal{L}_{\text{total}}$ with $\lambda = 0$. The Adamax optimizer is also used with a batch size of 8 for 0.2M iterations.

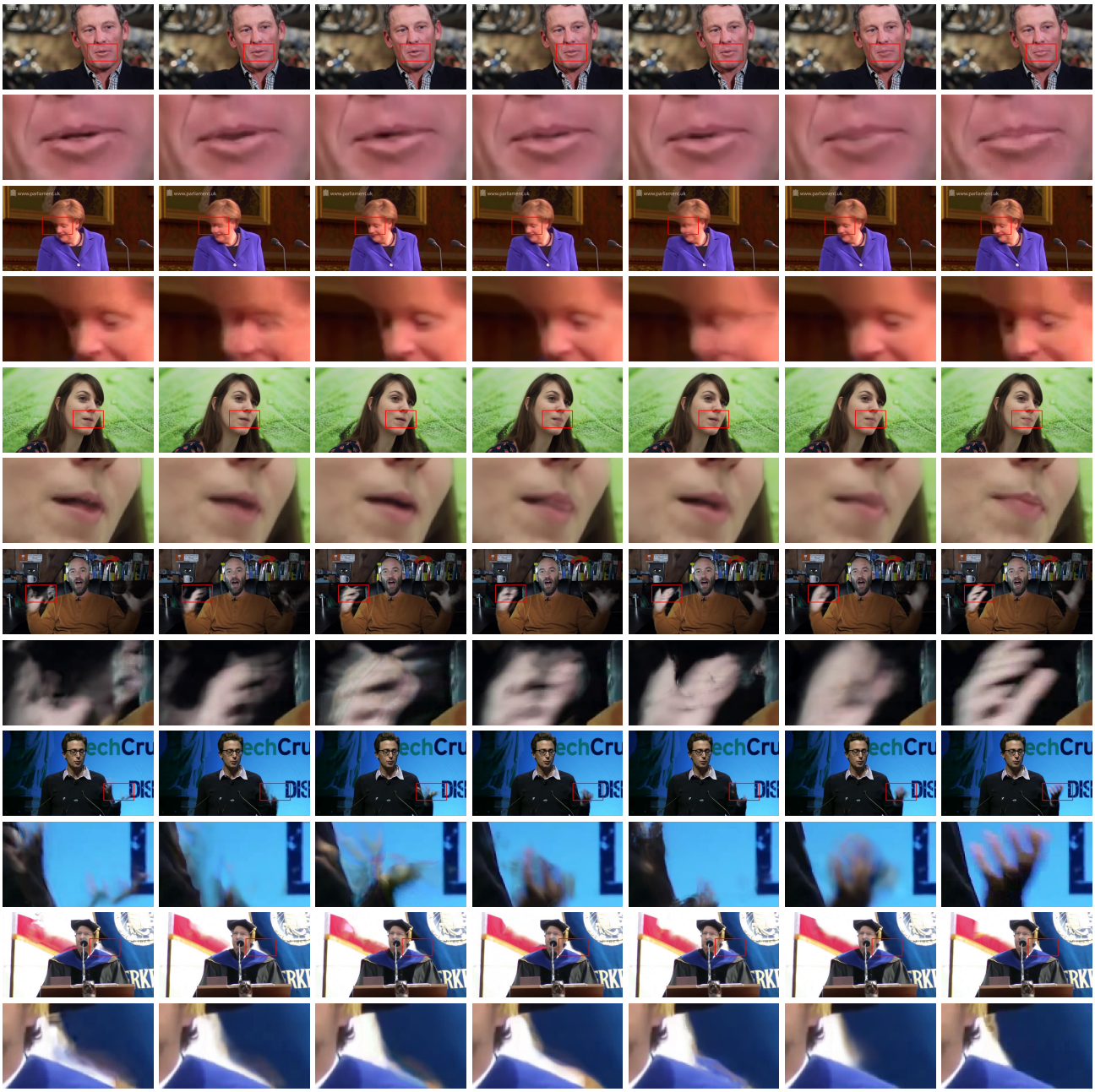
C. Comparison with the State-of-the-Arts

We compare the proposed algorithm with conventional state-of-the-art ones: SepConv [4], CyclicGen [28], DAIN [12], AdaCoF [15] and BMBC [3]. The evaluation metrics are the peak signal-to-noise ratio (PSNR) and the structural similarity index measure (SSIM). Table I lists the average PSNR and SSIM scores of each algorithm. Except for the 'hard' category, the proposed algorithm provides the best PSNR and SSIM results. In the 'hard' category, it ranks second after BMBC. Note that the performance of the proposed algorithm is at least 0.25dB higher than the other kernel-based algorithms SepConv and AdaCoF. Moreover, the proposed algorithm even outperforms the flow-based algorithms DAIN and BMBC in most cases.

Fig. 5 show qualitative comparison results. The top six rows compare facial parts, while the bottom six rows do non-facial parts. In the facial parts, especially on the eyes and mouths, the symmetric motion vectors represent the movements more faithfully than the asymmetric ones do. On the other hand, the non-facial parts have larger motions, so the asymmetric motion vectors are more effective in these cases. It is observed that the proposed algorithm provides reliable interpolation results in both facial and non-facial parts, by combining symmetric and asymmetric motions.

D. Ablation Study

We analyze the contribution of $\bar{I}_{0.5}$ in Eq. (5) to the frame synthesis. To this end, we compare the proposed algorithm with two modified versions: proposed-AS and proposed-S.



(a) Sepconv-$\mathcal{L}_f$ (b) CyclicGen (c) DAIN (d) AdaCoF (e) BMBC (f) Proposed (g) Ground Truth

Fig. 5. Qualitative comparison of VFI results.

- Proposed-AS: We modify the subnetwork that estimates the occlusion map in the asymmetric motion block to consider four intermediate candidates. Then we set $\bar{I}_{0.5}$ as follows.

$$\bar{I}_{0.5} = O_0^{as} \times I_{0.5}^{a0} + O_1^{as} \times I_{0.5}^{a1} + O_2^{as} \times I_{0.5}^{s0} + O_3^{as} \times I_{0.5}^{s1}. \quad (11)$$

- Proposed-S: We generate $\bar{I}_{0.5}$ using Eq. (10) instead of Eq. (5).

Table II compares the results on the whole test set. In Eq. (11), there are too many occlusion weights to be estimated, so they cannot be reliably estimated. Thus, Proposed-AS degrades the performance significantly. Also, by comparing the proposed algorithm with Proposed-S, we see that the asymmetric initial estimate in Eq. (5) is more effective than the symmetric one in Eq. (10).

Fig. 6 shows qualitative results. The proposed algorithm and Proposed-S show similar results in the facial part where symmetric motions are more reliable. In the non-facial part,

TABLE II
ABLATION STUDY. THE RESULTS ON THE WHOLE 300VW DATASET ARE REPORTED.

	PSNR(↑)	SSIM(↑)
Proposed-AS	37.73	0.9804
Proposed-S	38.12	0.9817
Proposed	**38.22**	**0.9820**

(a) Proposed-AS (b) Proposed-S (c) Proposed (d) Ground Truth

Fig. 6. Examples of interpolated frames in the ablation study.

however, the proposed algorithm outperforms both Proposed-AS and Proposed-S.

IV. CONCLUSIONS

We proposed a facial video frame interpolation algorithm, which exploits both symmetric and asymmetric motion information. We designed two motion estimation blocks: one for symmetric motion and the other for asymmetric motion. We then generated multi-scale contextual features and intermediate frame candidates using both symmetric and asymmetric motion fields. Then, through a frame synthesis block, we refined those intermediate frame candidates, by exploiting the contextual features, to produce the final interpolation output. Experimental results demonstrated that the proposed algorithm outperforms conventional algorithms, by combining both symmetric and asymmetric motion information effectively.

V. ACKNOWLEDGMENT

This work was supported partly by Samsung Electronics Co., Ltd. (No. IO201214-08156-01) and partly by the National Research Foundation of Korea (NRF) grants funded by the Korea government (MSIT) (No. NRF-2018R1A2B3003896 and No. NRF-2021R1A4A1031864).

REFERENCES

[1] B.-D. Choi, J.-W. Han, C.-S. Kim, and S.-J. Ko, "Motion-compensated frame interpolation using bilateral motion estimation and adaptive overlapped block motion compensation," *IEEE Trans. Circuit Syst. Video Technol.*, vol. 17, no. 4, pp. 407–416, 2007.

[2] J. Park, C. Lee, and C.-S. Kim, "Deep learning approach to video frame rate up-conversion using bilateral motion estimation," in *APSIPA*, 2019.

[3] J. Park, K. Ko, C. Lee, and C.-S. Kim, "BMBC: Bilateral motion estimation with bilateral cost volume for video interpolation," in *ECCV*, 2020.

[4] S. Niklaus, L. Mai, and F. Liu, "Video frame interpolation via adaptive separable convolution," in *ICCV*, 2017.

[5] Z. Liu, R. A. Yeh, X. Tang, Y. Liu, and A. Agarwala, "Video frame synthesis using deep voxel flow," in *ICCV*, 2017.

[6] T. Xue, B. Chen, J. Wu, D. Wei, and W. T. Freeman, "Video enhancement with task-oriented flow," *International Journal of Computer Vision*, vol. 127, no. 8, pp. 1106–1125, 2019.

[7] M. Usman, X. He, K.-M. Lam, M. Xu, S. M. M. Bokhari, and J. Chen, "Frame interpolation for cloud-based mobile video streaming," *IEEE Transactions on Multimedia*, vol. 18, no. 5, pp. 831–839, 2016.

[8] B. D. Lucas, T. Kanade *et al.*, "An iterative image registration technique with an application to stereo vision," in *IJCAI*, 1981.

[9] 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 *CVPR*, 2017.

[10] D. Sun, X. Yang, M.-Y. Liu, and J. Kautz, "PWC-Net: CNNs for optical flow using pyramid, warping, and cost volume," in *CVPR*, 2018.

[11] S. Niklaus and F. Liu, "Context-aware synthesis for video frame interpolation," in *CVPR*, 2018.

[12] W. Bao, W.-S. Lai, C. Ma, X. Zhang, Z. Gao, and M.-H. Yang, "Depth-aware video frame interpolation," in *CVPR*, 2019.

[13] J. Park, C. Lee, and C.-S. Kim, "Asymmetric bilateral motion estimation for video frame interpolation," in *ICCV*, 2021.

[14] S. Niklaus, L. Mai, and F. Liu, "Video frame interpolation via adaptive convolution," in *CVPR*, 2017.

[15] H. Lee, T. Kim, T.-Y. Chung, D. Pak, Y. Ban, and S. Lee, "AdaCoF: Adaptive collaboration of flows for video frame interpolation," in *CVPR*, 2020.

[16] Z. Xu, B. Li, M. Geng, Y. Yuan, and G. Yu, "Anchorface: An anchor-based facial landmark detector across large poses," in *AAAI*, 2021.

[17] X. Dong and Y. Yang, "Teacher supervises students how to learn from partially labeled images for facial landmark detection," in *ICCV*, 2019.

[18] C. Ma, Z. Jiang, Y. Rao, J. Lu, and J. Zhou, "Deep face super-resolution with iterative collaboration between attentive recovery and landmark estimation," in *CVPR*, 2020.

[19] X. Ma, X. Kong, S. Zhang, and E. Hovy, "Macow: Masked convolutional generative flow," in *NeurIPS*, 2019.

[20] J. Menick and N. Kalchbrenner, "Generating high fidelity images with subscale pixel networks and multidimensional upscaling," in *ICIP*, 2019.

[21] K. Lim, N.-H. Shin, Y.-Y. Lee, and C.-S. Kim, "Order learning and its application to age estimation," in *ICLR*, 2019.

[22] S.-H. Lee and C.-S. Kim, "Deep repulsive clustering of ordered data based on order-identity decomposition," in *ICLR*, 2020.

[23] D. Jiménez-Rodríguez, A. Santillán García, J. Montoro Robles, M. d. M. Rodríguez Salvador, F. J. Muñóz Ronda, and O. Arrogante, "Increase in video consultations during the covid-19 pandemic: healthcare professionals' perceptions about their implementation and adequate management," *International journal of environmental research and public health*, vol. 17, no. 14, p. 5112, 2020.

[24] J. R. Wlodarczyk, E. M. Wolfswinkel, and J. N. Carey, "Coronavirus 2019 video conferencing: preserving resident education with online meeting platforms," *Plastic and Reconstructive Surgery*, vol. 146, no. 1, pp. 110e–111e, 2020.

[25] D. M. Mann, J. Chen, R. Chunara, P. A. Testa, and O. Nov, "Covid-19 transforms health care through telemedicine: evidence from the field," *Journal of the American Medical Informatics Association*, 2020.

[26] Y. Li, X. Zhang, and D. Chen, "Csrnet: Dilated convolutional neural networks for understanding the highly congested scenes," in *CVPR*, 2018.

[27] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *MICCAI*, 2015.

[28] Y.-L. Liu, Y.-T. Liao, Y.-Y. Lin, and Y.-Y. Chuang, "Deep video frame interpolation using cyclic frame generation," in *AAAI*, 2019.

[29] D. Fourure, R. Emonet, E. Fromont, D. Muselet, A. Tremeau, and C. Wolf, "Residual conv-deconv grid network for semantic segmentation," in *BMVC*, 2017.

[30] S. Abu-El-Haija, N. Kothari, J. Lee, P. Natsev, G. Toderici, B. Varadarajan, and S. Vijayanarasimhan, "Youtube-8m: A large-scale video classification benchmark," *arXiv preprint arXiv:1609.08675*, 2016.

[31] G. Tzimiropoulos, "Project-out cascaded regression with an application to face alignment," in *CVPR*, 2015.

[32] P. Charbonnier, L. Blanc-Féraud, G. Aubert, and M. Barlaud, "Deterministic edge-preserving regularization in computed imaging," *IEEE Transactions on image processing*, vol. 6, no. 2, pp. 298–311, 1997.