Supplemental Materials on

BMBC: Bilateral Motion Estimation with Bilateral Cost Volume for Video Interpolation

Junheum Park[1], Keunsoo Ko[1], Chul Lee[2], and Chang-Su Kim[1]

[1] School of Electrical Engineering, Korea University, Seoul, Korea
[2] Department of Multimedia Engineering, Dongguk University, Seoul, Korea

S-1 Network Configurations

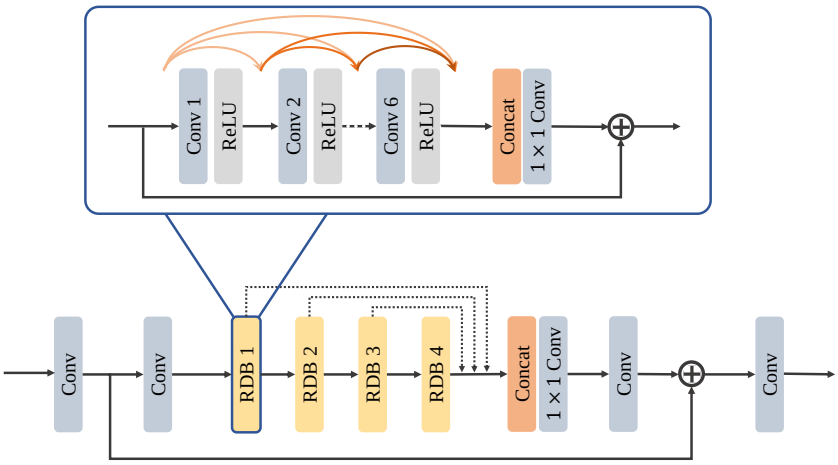


Fig. S-1. Illustration of dynamic filter generation network.

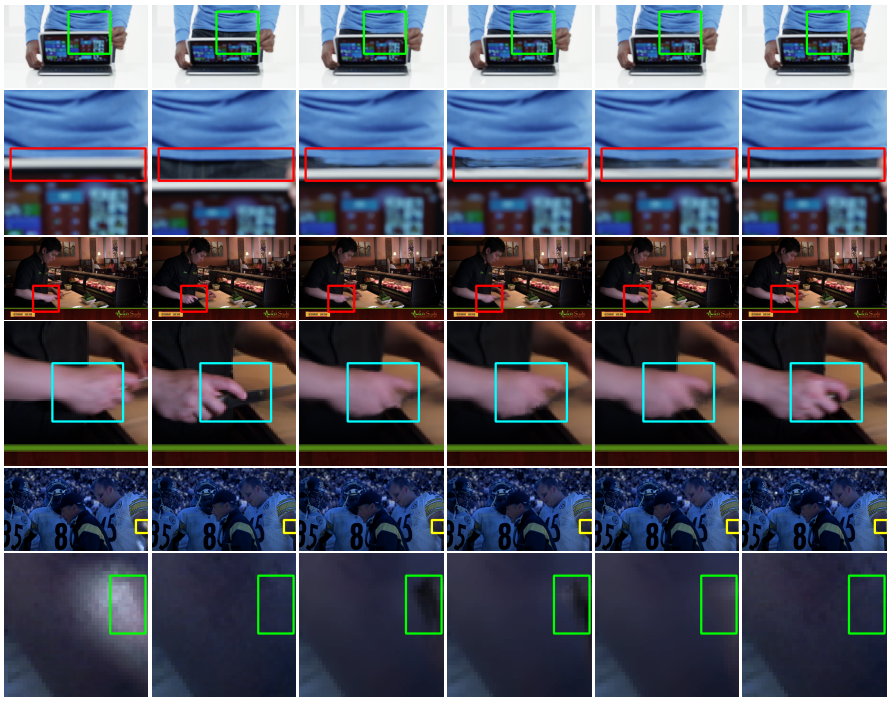
Fig. S-1 shows the structure of the proposed dynamic filter generation network, which is used in the frame synthesis block in Fig. 6 in the main paper. We employ the super-resolution network RDN [34] as the basis of this dynamic filter generation network. RDN combines residual networks with dense networks to exploit hierarchical features and achieve efficient training. In the original RDN, there is an up-sampling layer for super-resolution. However, since an interpolated frame has the same spatial resolution as input frames in this work, we remove the up-sampling layer. We use 3×3 convolution in all convolutional layers. There are three hyper-parameters in the dynamic filter generation network: the number D of residual dense blocks (RDBs), the number C of convolutional layers per RDB, and the growth rate G, which is the number of filters at each convolutional layer in an RDB. In this work, since intermediate candidates are already motion-compensated, the network requires a relatively small receptive field. Thus, we set $D = 4$, $C = 6$, and $G = 16$.

S-2 Ablation Study on Dynamic Filter Generation Network

In Table 5 in the main paper, we analyze how different input settings affect the PSNR performances of the dynamic filter generation network. Here, we qualitatively compare interpolated frames using the three settings using 5×5 kernels in Table 5, which are denoted by

- DF-None: Only intermediate candidates are used.
- DF-Input: In addition to the above candidates, input frames are used.
- DF-All: In addition to DF-Input, six warped context maps are used.

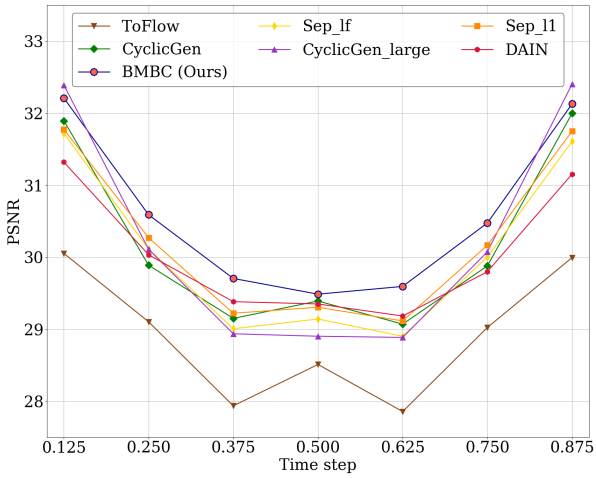
Fig. S-2 compares interpolated regions, which are challenging due to fast motion, object deformation, and occlusion. We see that, as more input information is used, the dynamic filter generation network yields more reliable filters and leads to more faithful interpolation results.



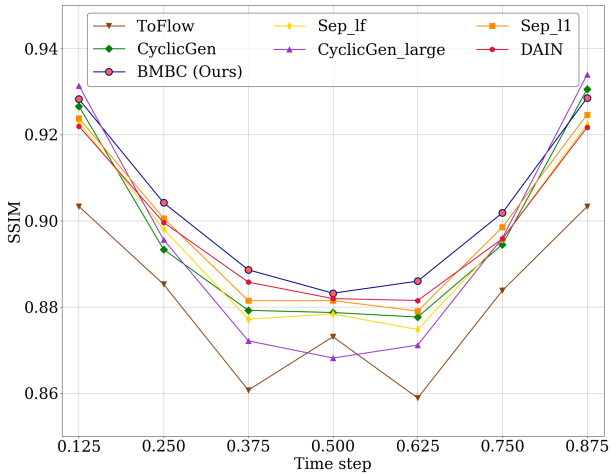
(a) (b) (c) (d) (e) (f)

Fig. S-2. Comparison of interpolated frames: the ground-truth 1st frame I_1 (a), the ground-truth 3rd frame I_3 (b), interpolated 2nd frames I_2, which are obtained by DF-None (c), DF-Inputs (d), and DF-All (e), and the ground-truth 2nd frame (f).

S-3 Interpolation at Arbitrary Time Instances



(a) PSNR



(b) SSIM

Fig. S-3. Comparisons with state-of-the-art algorithms at different time steps.

Fig. S-3 compares the average PSNR and SSIM performances at each time step on the Adobe240-fps dataset [29]. Note that $\times 2$, $\times 4$, and $\times 8$ interpolation results correspond to the results at $T = 0.5$, $T \in \{0.25, 0.5, 0.75\}$, and $T \in \{0.125, 0.25, 0.375, 0.5, 0.625, 0.75, 0.875\}$, respectively. As mentioned in the main paper, only DAIN [2] and the proposed algorithm can provide interpolation results at arbitrary time steps. The other conventoinal algorithms perform $\times 2$ interpolation recursively to obtain these results. The proposed algorithm yields

the best performances at each time step T, except that it yields slightly worse results than the CyclicGen large model [16] at $T = 0.125$ and 0.875. Note that CyclicGen interpolates frames quite well when the sequence does not contain fast motions. In other words, it yields good results when a frame to be interpolated is similar to one of the input frames. Therefore, at $T = 0.125$ and 0.875, CycleGen yields reliable interpolation results, which are similar to I_0 and I_1, respectively.

Fig. S-4 compares interpolation error maps. Brighter pixels indicate larger differences from the ground-truth. In general, the proposed algorithm yields the least amount of errors.

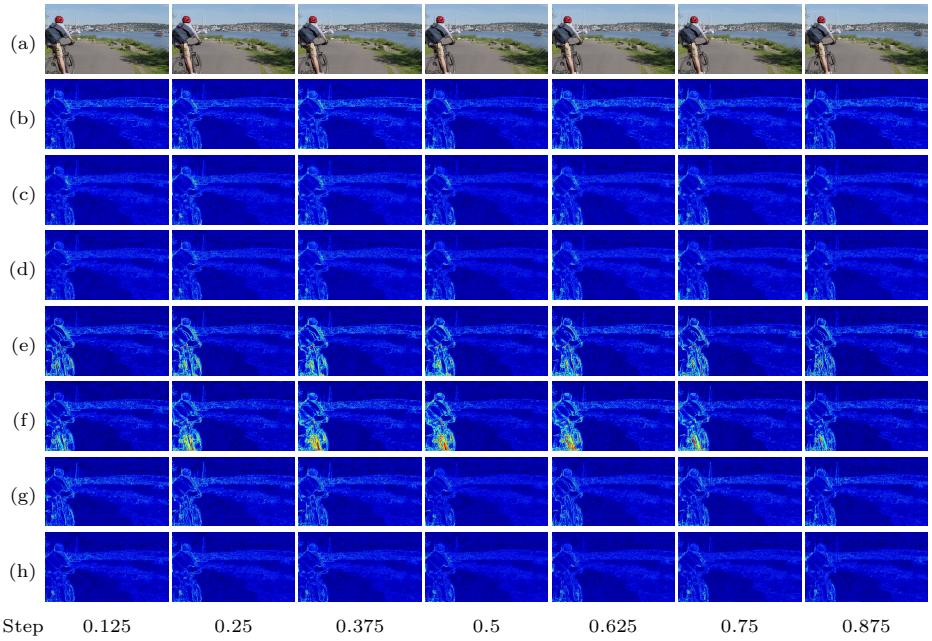


Fig. S-4. Comparison of error maps at different time steps: (a) Ground-truth, (b) ToFlow [32], (c) SepConv-L_f [22], (d) SepConv-L_1 [22], (e) CyclicGen [16], (f) CyclicGen_large [16], (g) DAIN [2], and (h) BMBC (Ours).

Fig. S-5 shows interpolated results at different time steps. The proposed algorithm generates less distortions than the conventional algorithms especially on the fast rotating wheel.

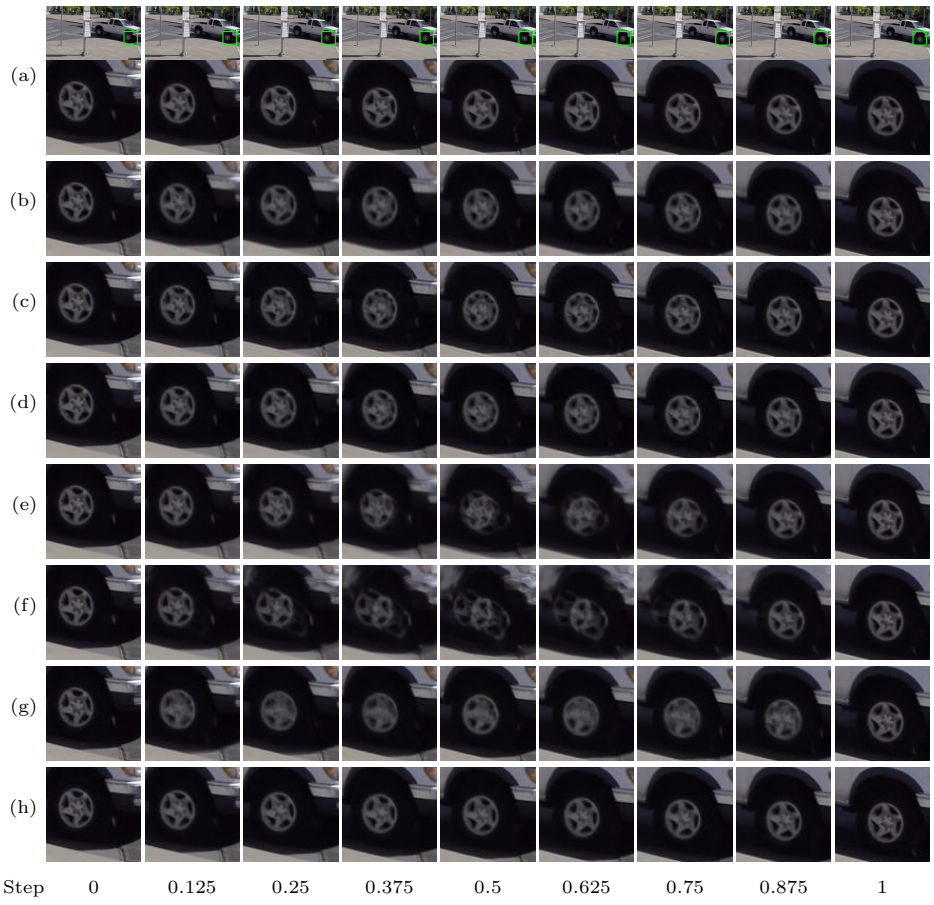
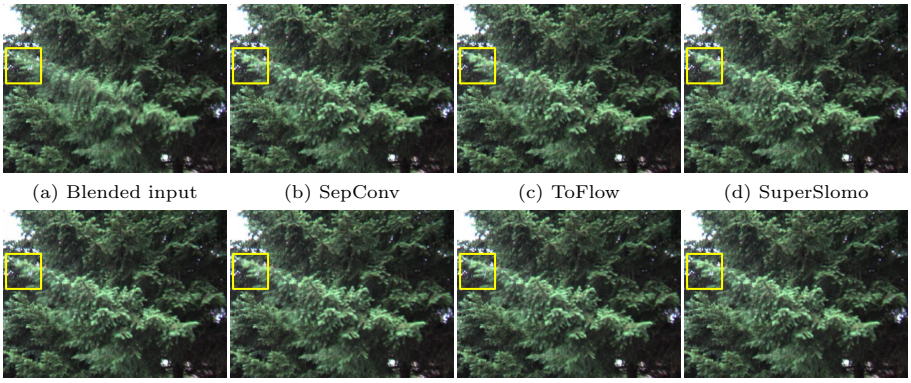


Fig. S-5. Comparison of interpolation results at different time steps: (a) Ground-truth, (b) ToFlow [32], (c) SepConv-L_f [22], (d) SepConv-L_1 [22], (e) CyclicGen [16], (f) CyclicGen_large [16], (g) DAIN [2], and (h) BMBC (Ours).

S-4 More Experimental Results

We provide more comparative results on the Middlebury [1], UCF101 [28], and Vimeo90K [32] datasets.

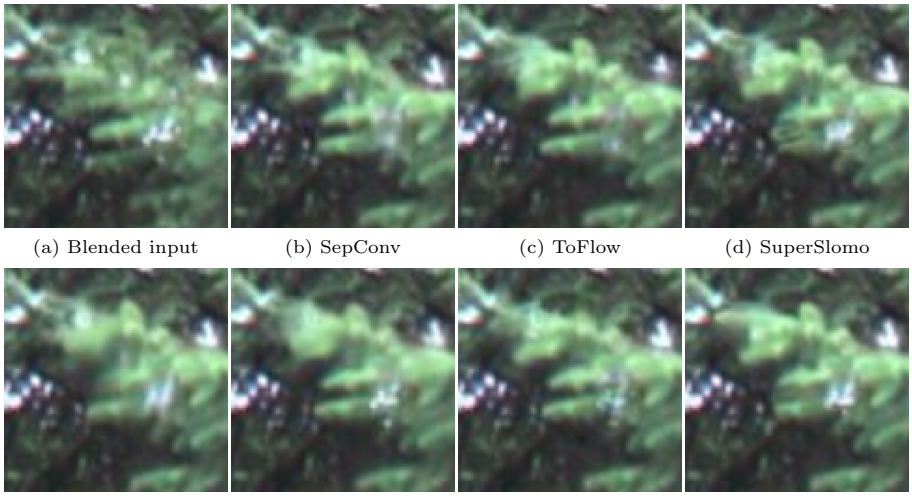
S-4.1 Middlebury



(a) Blended input (b) SepConv (c) ToFlow (d) SuperSlomo

(e) CtxSyn (f) MEMC-Net* (g) DAIN (h) BMBC (Ours)

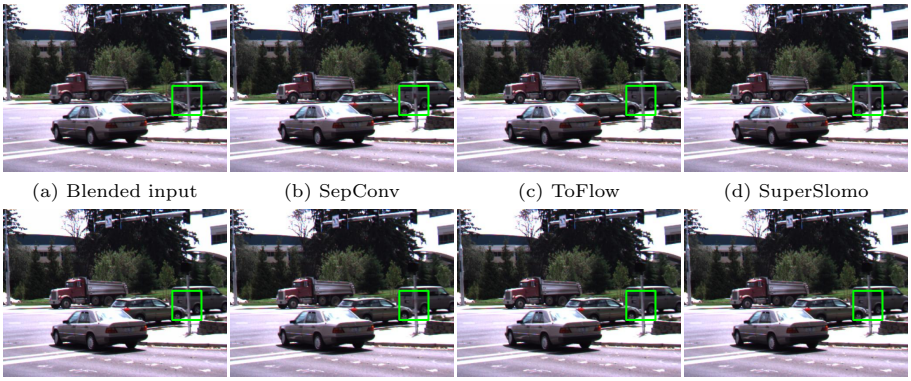
Fig. S-6. Comparison of interpolated frames on the "Evergreen" sequence: (a) Blended input, (b) SepConv [22], (c) ToFlow [32], (d) SuperSlomo [14], (e) CtxSyn [20], (f) MEMC-Net* [3], (g) DAIN [2], and (h) BMBC (Ours).



(a) Blended input (b) SepConv (c) ToFlow (d) SuperSlomo

(e) CtxSyn (f) MEMC-Net* (g) DAIN (h) BMBC (Ours)

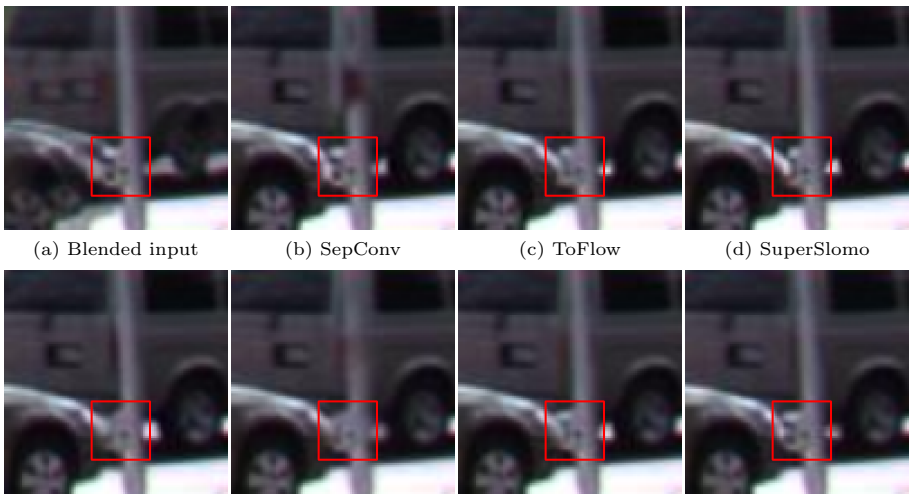
Fig. S-7. Zoomed results of the yellow square region in Fig. S-6.



(a) Blended input (b) SepConv (c) ToFlow (d) SuperSlomo

(e) CtxSyn (f) MEMC-Net* (g) DAIN (h) BMBC (Ours)

Fig. S-8. Comparison of interpolated frames on the "Dumptruck" sequence: (a) Blended input, (b) SepConv [22], (c) ToFlow [32], (d) SuperSlomo [14], (e) CtxSyn [20], (f) MEMC-Net* [3], (g) DAIN [2], and (h) BMBC (Ours).



(a) Blended input (b) SepConv (c) ToFlow (d) SuperSlomo

(e) CtxSyn (f) MEMC-Net* (g) DAIN (h) BMBC (Ours)

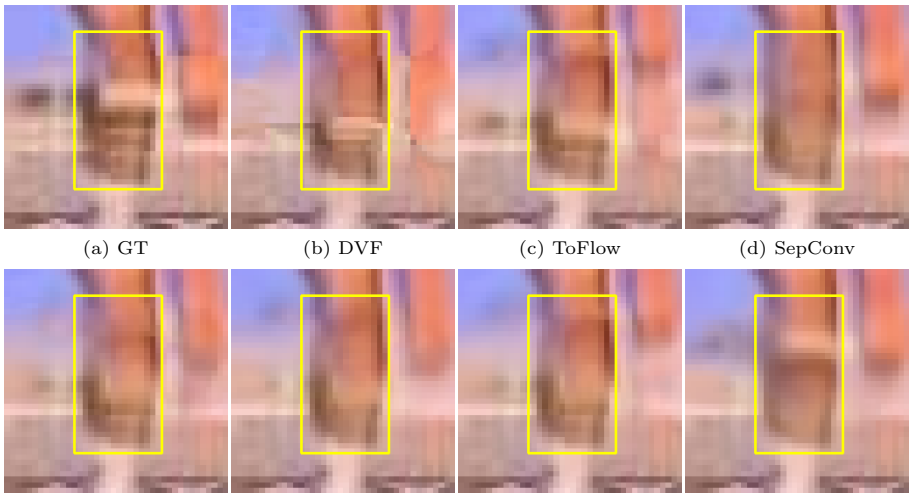
Fig. S-9. Zoomed results of the green square region in Fig. S-8. The differences between algorithms are easily observable within red squares.

S-4.2 UCF101

(a) GT (b) DVF (c) ToFlow (d) SepConv

(e) CyclicGen (f) MEMC-Net* (g) DAIN (h) BMBC (Ours)

Fig. S-10. Visual comparison on the UCF101 dataset: (a) Ground-truth, (b) DVF [17], (c) ToFlow [32], (d) SepConv-L_1 [22], (e) CyclicGen [16], (f) MEMC-Net* [3], (g) DAIN [2], and (h) BMBC (Ours).



(a) GT (b) DVF (c) ToFlow (d) SepConv

(e) CyclicGen (f) MEMC-Net* (g) DAIN (h) BMBC (Ours)

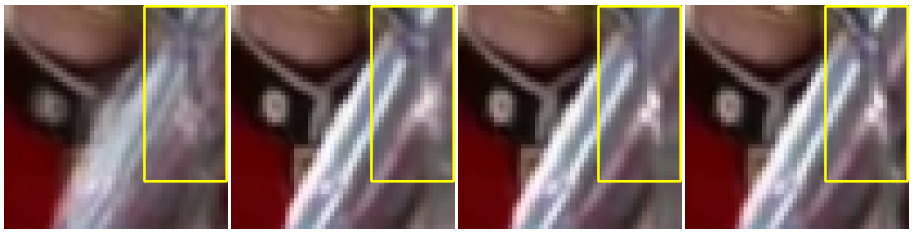
Fig. S-11. Zoomed results of the green square region in Fig. S-10. The proposed algorithm produces less distortions than the conventional algorithms around the fast moving bar in the yellow box.

(a) GT (b) DVF (c) ToFlow (d) SepConv

(e) CyclicGen (f) MEMC-Net* (g) DAIN (h) BMBC (Ours)

Fig. S-12. Visual comparison on the UCF101 dataset: (a) Ground-truth, (b) DVF [17], (c) ToFlow [32], (d) SepConv-L_1 [22], (e) CyclicGen [16], (f) MEMC-Net* [3], (g) DAIN [2], and (h) BMBC (Ours).

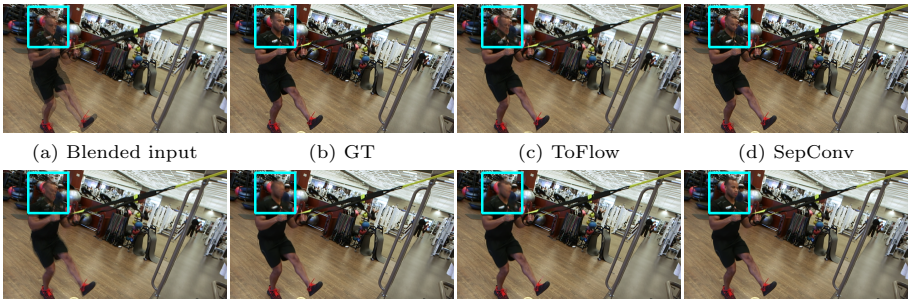
(a) GT (b) DVF (c) ToFlow (d) SepConv



(e) CyclicGen (f) MEMC-Net* (g) DAIN (h) BMBC (Ours)

Fig. S-13. Zoomed results of the green square region in Fig. S-12. The proposed algorithm faithfully reconstructs details of the tuba in the yellow box.

S-4.3 Vimeo90K



(a) Blended input (b) GT (c) ToFlow (d) SepConv

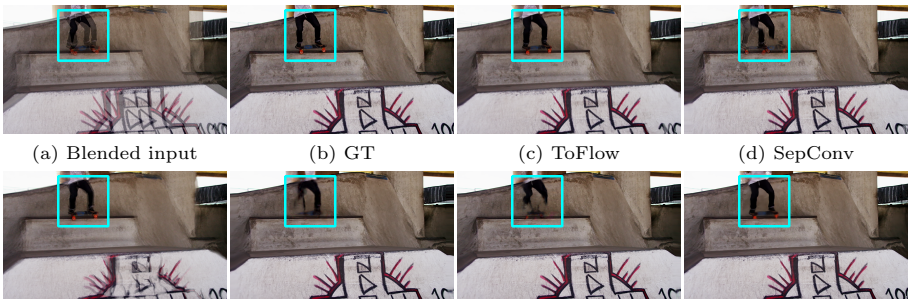
(e) CyclicGen (f) MEMC-Net* (g) DAIN (h) BMBC (Ours)

Fig. S-14. Visual comparison on the Vimeo90K dataset: (a) Blended input, (b) ground-truth, (c) ToFlow [32], (d) SepConv-L_f [22], (e) CyclicGen [16], (f) MEMC-Net* [3], (g) DAIN [2], and (h) BMBC (Ours).

(a) Blended input (b) GT (c) ToFlow (d) SepConv

(e) CyclicGen (f) MEMC-Net* (g) DAIN (h) BMBC (Ours)

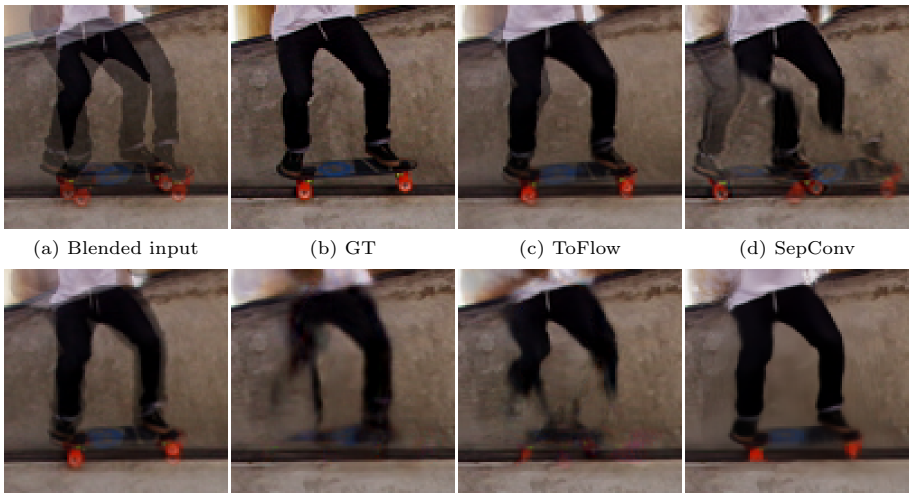
Fig. S-15. Zoomed results of the sky blue square region in Fig. S-14.



(a) Blended input (b) GT (c) ToFlow (d) SepConv

(e) CyclicGen (f) MEMC-Net* (g) DAIN (h) BMBC (Ours)

Fig. S-16. Visual comparison on the Vimeo90K dataset: (a) Blended input, (b) ground-truth, (c) ToFlow [32], (d) SepConv-L_f [22], (e) CyclicGen [16], (f) MEMC-Net* [3], (g) DAIN [2], and (h) BMBC (Ours).



(a) Blended input (b) GT (c) ToFlow (d) SepConv

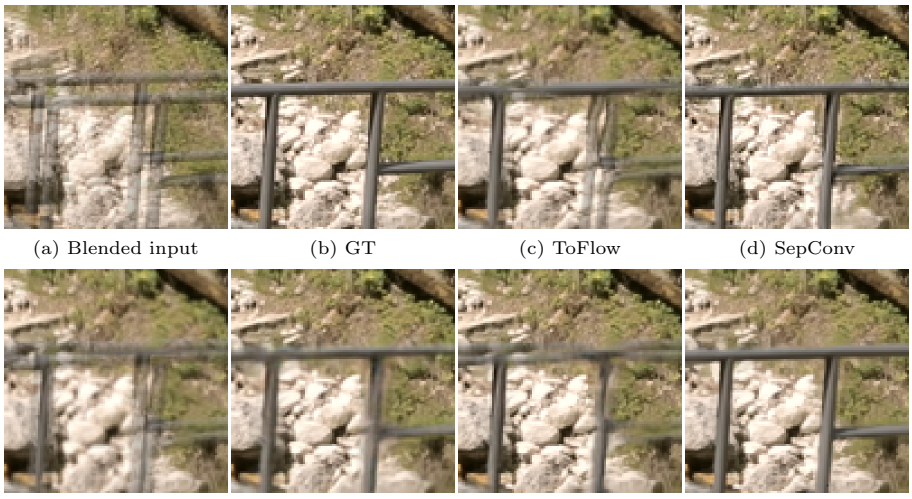
(e) CyclicGen (f) MEMC-Net* (g) DAIN (h) BMBC (Ours)

Fig. S-17. Zoomed results of the sky blue square region in Fig. S-16.

(a) Blended input (b) GT (c) ToFlow (d) SepConv

(e) CyclicGen (f) MEMC-Net* (g) DAIN (h) BMBC (Ours)

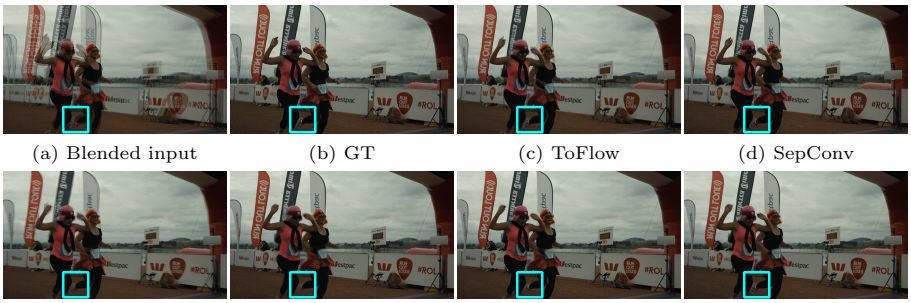
Fig. S-18. Visual comparison on the Vimeo90K dataset: (a) Blended input, (b) ground-truth, (c) ToFlow [32], (d) SepConv-L_f [22], (e) CyclicGen [16], (f) MEMC-Net* [3], (g) DAIN [2], and (h) BMBC (Ours).



(a) Blended input (b) GT (c) ToFlow (d) SepConv

(e) CyclicGen (f) MEMC-Net* (g) DAIN (h) BMBC (Ours)

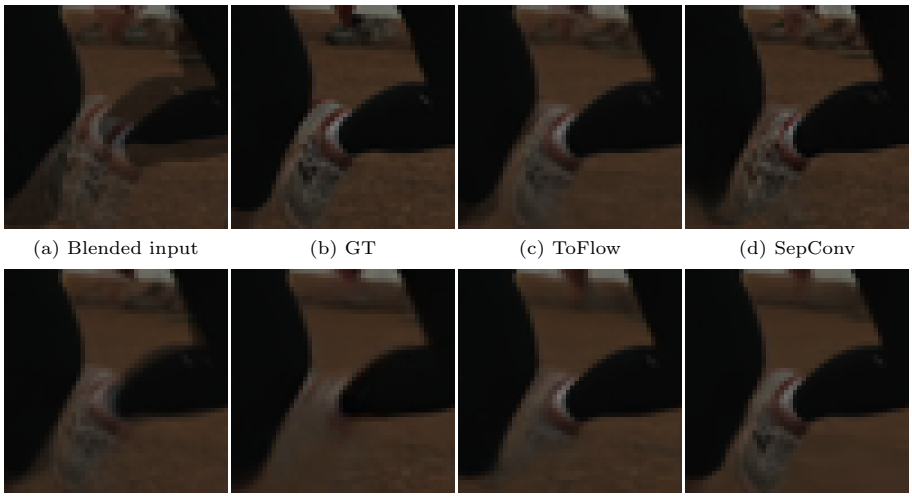
Fig. S-19. Zoomed results of the sky blue square region in Fig. S-18.



(a) Blended input (b) GT (c) ToFlow (d) SepConv

(e) CyclicGen (f) MEMC-Net* (g) DAIN (h) BMBC (Ours)

Fig. S-20. Visual comparison on the Vimeo90K dataset: (a) Blended input, (b) ground-truth, (c) ToFlow [32], (d) SepConv-L_f [22], (e) CyclicGen [16], (f) MEMC-Net* [3], (g) DAIN [2], and (h) BMBC (Ours).



(a) Blended input (b) GT (c) ToFlow (d) SepConv

(e) CyclicGen (f) MEMC-Net* (g) DAIN (h) BMBC (Ours)

Fig. S-21. Zoomed results of the sky blue square region in Fig. S-20.