Supplementary Material: Deep Bayesian Video Frame Interpolation

Zhiyang Yu[1,2], Yu Zhang[2], Xujie Xiang[2,3], Dongqing Zou[2,4], Xijun Chen[1], and Jimmy S. Ren[2,4]

[1] Harbin Institute of Technology, Harbin, China
[2] SenseTime Research and Tetras.AI, Beijing, China
[3] Beihang University, Beijing, China
[4] Qing Yuan Research Institute, Shanghai Jiao Tong University, Shanghai, China

1 Introduction

In this supplementary material, we elaborate the details on the following aspects:

1. **Additional remarks on implementing deep gradients (Sect. 2)**, where we provide more discussions on efficient implementation.
2. **Detailed benchmarking settings and reproducibility (Sect. 3)**, where we provide more details on experimental settings and the guaranteed reproducibility of our experiments.
3. **More visual results (Sect. 4)**, providing more visual comparisons between our approach and the state-of-the-art methods on test datasets.
4. **Video results**. In https://www.youtube.com/watch?v=8KvFwN1_3DY we provide more visual results in videos.

2 Additional Remarks on Implementing Deep Gradients

In Eq. (12) of submitted paper, we formulate the deep gradients $\frac{\partial \log P(\Delta\hat{\mathbf{F}}_{i\to t}^{(k)})|\mathbf{I}_t^{(k)},\cdot)}{\partial \Delta\hat{\mathbf{F}}_{i\to t}}$. Specifically, by the change of variable rule,

$$P(\Delta\hat{\mathbf{F}}_{i\to t}^{(k)})|\mathbf{I}_t^{(k)},\cdot) = P(\mathbf{X}_{i\to t}^{(k)}|\mathbf{I}_t^{(k)},\cdot)\left|\det\left(\frac{\partial \mathbf{X}_{i\to t}}{\partial \Delta\hat{\mathbf{F}}_{i\to t}}\right)\right|, \tag{1}$$

after simple derivations, we achieve at the formula Eq. (12) in the paper:

$$\frac{\partial \log P(\Delta\hat{\mathbf{F}}_{i\to t}^{(k)})|\mathbf{I}_t^{(k)},\cdot)}{\partial \Delta\hat{\mathbf{F}}_{i\to t}} = \frac{\partial \log\left|\det\left(\frac{\partial \mathbf{X}_{i\to t}}{\partial \Delta\hat{\mathbf{F}}_{i\to t}}\right)\right|}{\partial \Delta\hat{\mathbf{F}}_{i\to t}} + \frac{\partial \mathbf{X}_{i\to t}}{\partial \Delta\hat{\mathbf{F}}_{i\to t}}\frac{\partial \log P(\mathbf{X}_{i\to t}^{(k)}|\mathbf{I}_t^{(k)},\cdot)}{\partial \mathbf{X}_{i\to t}}. \tag{2}$$

Here, the quantity $\frac{\partial \log P(\mathbf{X}_{i\to t}^{(k)}|\mathbf{I}_t^{(k)},\cdot)}{\partial \mathbf{X}_{i\to t}}$ can be easily computed by Eq. (13) in the paper. The quantity $\frac{\partial \mathbf{X}_{i\to t}}{\partial \Delta\hat{\mathbf{F}}_{i\to t}}$ is the first order gradient achieveable by differentiating through the invertible layers. For the remaining quantity $\frac{\partial \log\left|\det\left(\frac{\partial \mathbf{X}_{i\to t}}{\partial \Delta\hat{\mathbf{F}}_{i\to t}}\right)\right|}{\partial \Delta\hat{\mathbf{F}}_{i\to t}}$,

$$\frac{\partial \log \left|\det\left(\frac{\partial \mathbf{X}_{i\to t}}{\partial \Delta\hat{\mathbf{F}}_{i\to t}}\right)\right|}{\partial \Delta\hat{\mathbf{F}}_{i\to t}} = \frac{\partial \mathbf{X}_{i\to t}}{\partial^2 \Delta\hat{\mathbf{F}}_{i\to t}} \left(\frac{\partial \mathbf{X}_{i\to t}}{\partial \hat{\mathbf{F}}_{i\to t}}\right)^{-1}, \tag{3}$$

Since that $\mathbf{X}_{i\to t} = f(\hat{\mathbf{F}}_{i\to t})$ and $f(\cdot)$ is a continuous and invertible, $\left(\frac{\partial \mathbf{X}_{i\to t}}{\partial \hat{\mathbf{F}}_{i\to t}}\right)^{-1} = \frac{\partial \hat{\mathbf{F}}_{i\to t}}{\partial \mathbf{X}_{i\to t}}$, which can also be computed by multiplying the inverse of per-layer Jacobian. Since each layer is invertible, the inverse Jacobian exists.

To calculate remaining second-order graident $\frac{\partial \mathbf{X}_{i\to t}}{\partial^2 \Delta\hat{\mathbf{F}}_{i\to t}}$, let we write the function f as a composite of multiple invertible functions: $f = f_1 \circ f_2 \circ f_3 \cdots f_n$, and denote the Jacobian of f_i as g_i. By the chain rule,

$$\frac{\partial \mathbf{X}_{i\to t}}{\partial \Delta\hat{\mathbf{F}}_{i\to t}} = g_n \times g_{n-1} \times g_{n-2} \cdots g_1. \tag{4}$$

Define $s_i = g_i \times g_{i-1} \cdots g_1$, then $\frac{\partial \mathbf{X}_{i\to t}}{\partial \Delta\hat{\mathbf{F}}_{i\to t}} = s_n$ and we aim to compute $\frac{\partial s_n}{\partial \Delta\hat{\mathbf{F}}_{i\to t}}$. It is trivial to see

$$\frac{\partial s_i}{\partial \Delta\hat{\mathbf{F}}_{i\to t}} = \frac{\partial g_i}{\partial \Delta\hat{\mathbf{F}}_{i\to t}} s_{i-1} + g_i \frac{\partial s_{i-1}}{\partial \hat{\mathbf{F}}_{i\to t}}, \tag{5}$$

Let the transformed features by f_i be $\mathbf{Z}_i$, where $\mathbf{Z}_{n+1} = \Delta\hat{\mathbf{F}}_{i\to t}$ and $\mathbf{Z}_0 = \mathbf{X}_{i\to t}$. Then Eq. (5) can be written as

$$\frac{\partial s_i}{\partial \Delta\hat{\mathbf{F}}_{i\to t}} = \frac{\partial \mathbf{Z}_i}{\partial \Delta\hat{\mathbf{F}}_{i\to t}} \frac{\partial g_i}{\partial \mathbf{Z}_i} s_{i-1} + g_i \frac{\partial s_{i-1}}{\partial \hat{\mathbf{F}}_{i\to t}}, \quad \frac{\partial \mathbf{Z}_i}{\partial \Delta\hat{\mathbf{F}}_{i\to t}} = g_i \times g_{i-1} \cdots g_1. \tag{6}$$

Above equation indicates that we can compute the second-order gradient $\frac{\partial s_n}{\partial \Delta\hat{\mathbf{F}}_{i\to t}}$ by dynamic programming. In details, we can keep update of intermediate variables $\frac{\partial \mathbf{Z}_i}{\partial \Delta\hat{\mathbf{F}}_{i\to t}}$ and $\frac{\partial s_i}{\partial \hat{\mathbf{F}}_{i\to t}}$, and evaluate the new terms g_i and $\frac{\partial g_i}{\partial \mathbf{Z}_i}$ (which is the Hessian of the function of ith invertible layer) at each new step. As a result, all the derivatives in Eq. (2) can be evaluated efficiently within single pass.

Though here we present method to efficiently evaluate the gradients for deep invertible transformations, we note that in our submitted paper, the transformation is simply implemented with a single linear layer. It gets rid of the second order gradient, as linear functions have all-zero Hessian. However, we have shown that such a single transformation is already beneficial in our ablation study experiments (Sect. 4.3). The further advantage of modeling deeper gradients remain to be explored and will be leaved in our future work.

3 Detailed Benchmarking Settings and Reproducibility

Inconsistency of evaluation protocols in prior works. Evaluation polices of previous works have several inconsistent aspects, while here we discuss on a few of them. For 8× VFI, SloMo [5], QVI [15] and [3] train their models on self-collected, mostly distinct datasets from YouTube, while DAIN [1] is trained

Table 1. Comparing our reimplementations with reference results on the GoPro dataset in terms of PSNR.

	SloMo	QVI	DAIN	FLAVR	XVFI
Reference	28.52[6]	30.55[6]	29.00[6]	31.31[6]	29.75[13]
Ours	30.12	31.65	29.70	31.30	30.16

on the videos from *triplet* subset of Vimeo-90K [16], XVFI is trained on the new-public dataset of X4K1000FPS [13], and FLAVR [6] is trained on the training set of GoPro [7]. When it comes to the evaluation policy, FLAVR report results on the 512×512 patches cropped from the original 1280×720 test images of GoPro, while the results reported by [15] do not do such cropping. XVFI and FLAVR test their method on self-selected subsets of Adobe240 [14], while QVI is tested on the downsampled versions of such images.

In Sect. 4 of our paper, we aim to provide standard and unified benchmarking results on the dataset of GoPro, X4K1000FPS, Adobe240, Vimeo-90K, UCF101, DAVIS and the *easy*, *medium*, *hard* and *extreme* subsets from SNU-FILM. In the following, we first describe the general setting of our evaluation policy, then explain the details for the reproducibility and reliability of experiments related to each approach.

3.1 Experiments for 8× VFI

Experimental settings. For fair comparisons, we trained all the methods whose training code is public available on the training set of GoPro [1,5,6,13,15]. The data sampling strategy applied in [6] is adopted in our experiment resulting in total 22128 frame sequences with a length of 25. The 1st, 9th, 17th and 25th frames of each sequence are used as our inputs. The 7 frames between the 9th and 17th frames are used as ground truth. The test set of GoPro is sampled in the similar way resulting in 1500 non-cropped sequences with length of 25.

For evaluations on X4K1000FPS, we use the same test set defined by [13]. For Adobe240 evaluations, we follow [13] to sample 630 non-overlapped sequences. Note that randomly sampling test data on Adobe240 is also performed by [13] and [15], while the scale of our sampled test data is about 3× larger than [13]. We do not do image downsampling as applied in [15].

Reproducibility of SloMo [5], QVI [16], DAIN [1], FLAVR [6] and XVFI [13]. To guarantee the reproducibility of results for these methods, we first show reproducibility of these methods on the GoPro test set using the same settings of [6], then adapt these verified re-implementations to perform comparisons on the GoPro, X4K1000FPS and Adobe240 dataset. For reproducibility of XVFI, we show that our retrained model get similar results with [13] by training on the X4K1000FPS dataset [13], compared with the released pretrained model by [13]. After verifying the reproducibility of XVFI, we then retrain it on the GoPro dataset. See Table 1 for comparisons between our replementations and

Table 2. Comparing our reimplementations with reference results on 2× VFI datasets in terms of PSNR. Results of our reimplementations and those from reference are interleaved by "/" in each table unit, respectively.

	Vimeo-90K (septulets)	UCF101	DAVIS	SNU-FILM			
				Easy	Medium	Hard	Extreme
SloMo	34.43/32.90 [6]	32.45/32.33 [6]	26.10/25.65 [6]	36.12/-	33.44/-	29.17/-	24.14/-
QVI	34.98/35.15 [12]	32.87/32.89 [12]	27.20/27.17 [12]	39.53/-	36.43/-	31.07/-	24.96/-
CAIN	34.69/34.83 [12]	32.40/32.52 [12]	27.12/27.21 [12]	39.33/-	35.34/-	30.15/-	24.88/-
ABME	35.67/-	32.81/-	27.00/-	39.59/39.59 [10]	35.77/35.77 [10]	30.58/30.58 [10]	25.42/25.42 [10]
EDSC	34.52/-	32.67/-	26.28/-	40.01/40.01 [2]	35.37/35.37 [2]	29.59/29.59 [2]	24.39/24.39 [2]
XVFI[5]	35.21/-	32.68/-	26.89/-	39.21/39.76[11]	34.96/35.12[11]	29.43/29.30[11]	24.02/23.98[11]
DAIN	33.57/33.35 [12]	31.65/31.64 [12]	26.61/26.12 [12]	38.53/-	34.34/-	29.50/-	24.54/-
BMBC	34.76/34.76 [12]	32.61/32.61 [12]	26.42/26.42 [12]	39.90/39.90 [10]	35.34/35.31 [10]	29.34/39.33 [10]	23.65/23.92 [10]
Softsplat	35.76/35.76 [12]	32.89/32.89 [12]	27.42/27.42 [12]	-	-	-	-
VFIT-B	36.96/36.96 [12]	33.44/33.44 [12]	28.09/28.09 [12]	-	-	-	-
FLAVR	36.30/36.30 [6]	33.33/33.33 [6]	27.44/27.44 [6]	40.44/-	36.37/-	30.87/-	25.18/-

references. It can be seen that the results of our retrained models are no worse than those reported in the reference.

Note that for EDSC [2], there lacks open-source training code and reference results for 8× VFI. Therefore, we resort to use the released pretrained model provided by authors. Specifically, the model *EDSC_m* is used for evaluation.

3.2 Experiments for 2× VFI

In addition to the methods used for 8× evaluation [1,5,6,13,15], we additionally retrained CAIN[4] on the septuplet subset of Vimeo-90K for 2× evaluation. For 2× VFI, our training set consists of 64612 septuplets. We use the 1st, 3rd, 5th and 7th frames from each set as the input, and the $4th$ as the ground truth. The evaluations on Vimeo-90K, UCF101 and DAVIS follow the identical setting as in [6]. We follow the same settings used in [2,4,10] to report results on the 4 datasets from SNU-FILM. For ABME [10], BMBC [9] and EDSC [2], we evaluate with their pre-trained weights due to lacking the official training code. Softsplat [8] and VFIT-B[12] also do not open-source their code or models, and we take the results from [12]. To show the reproducibility, see Table 2 for comparisons between our reimplementations and references. All of our reimplementations are comparable with corresponding reference.

4 More results

More visual results are illustrated in Fig. 1, 2, 3, 4, 5, 6.

[5] The XVFI in [11] is trained on the combination of triplets and septulets subsets of Vimeo-90K.

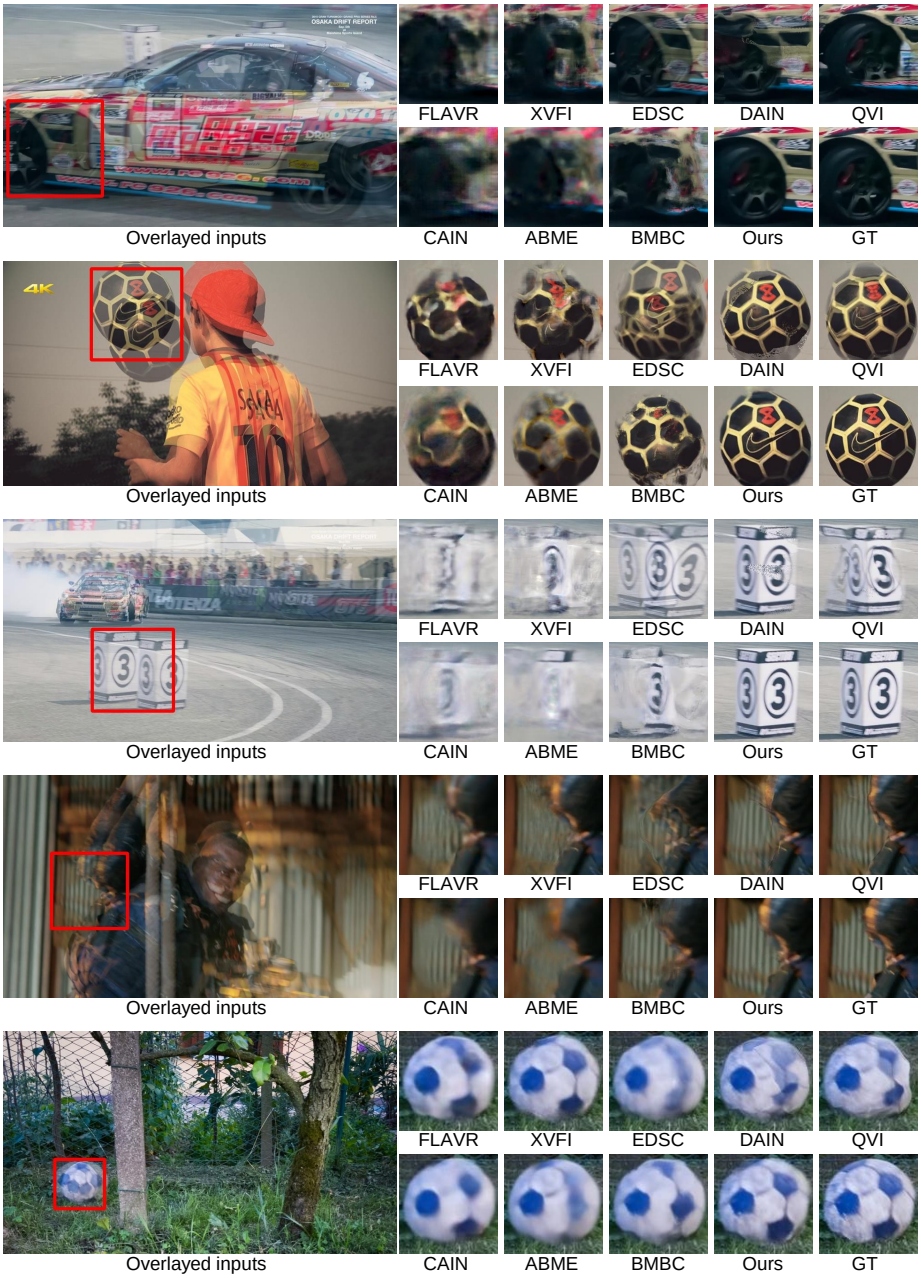

Fig. 1. Visual comparisons on DAVIS. We overlay the nearest 2 input frames to illustrate the input motion. Best compared in the electronic version of this paper with zoom.

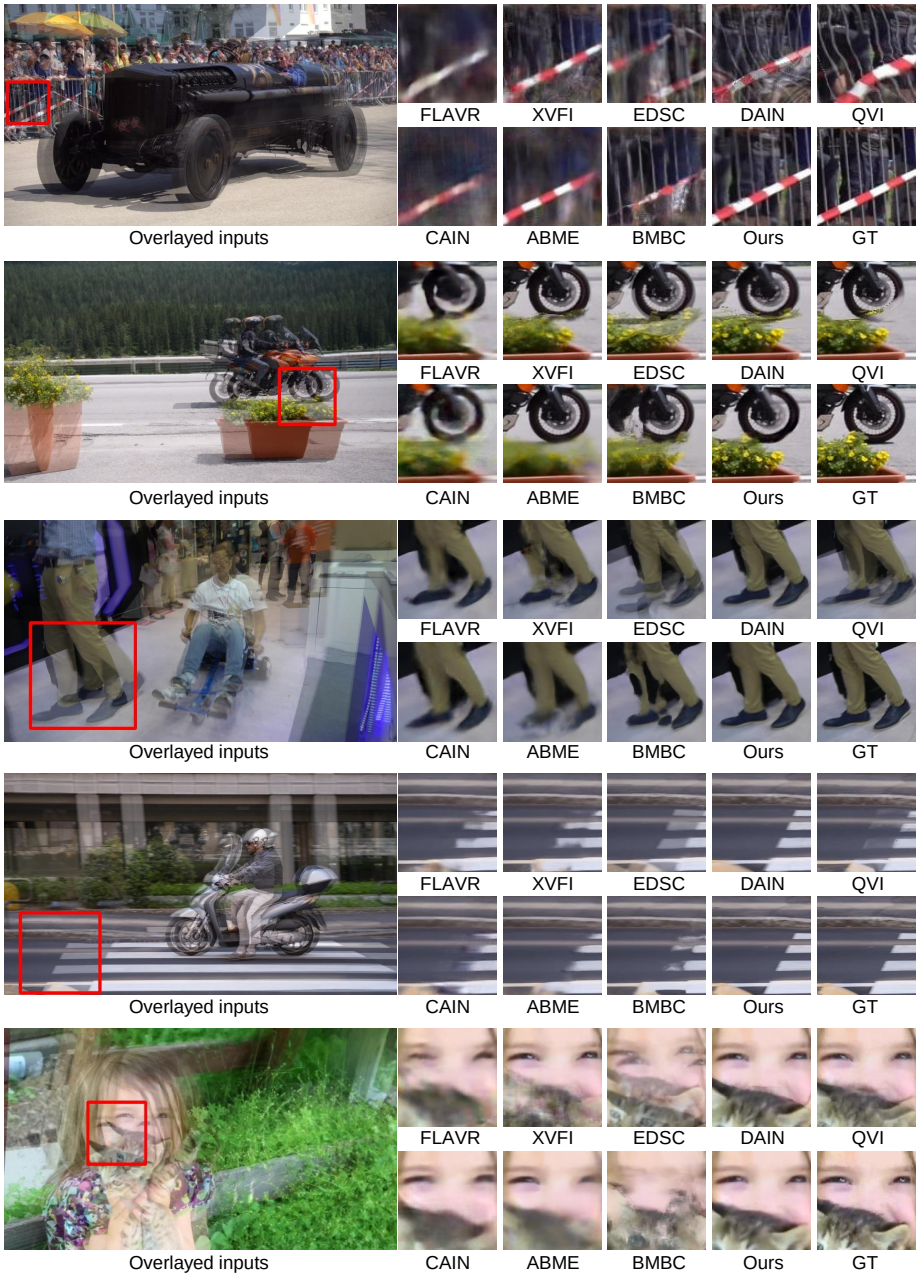

Fig. 2. Visual comparisons on DAVIS.

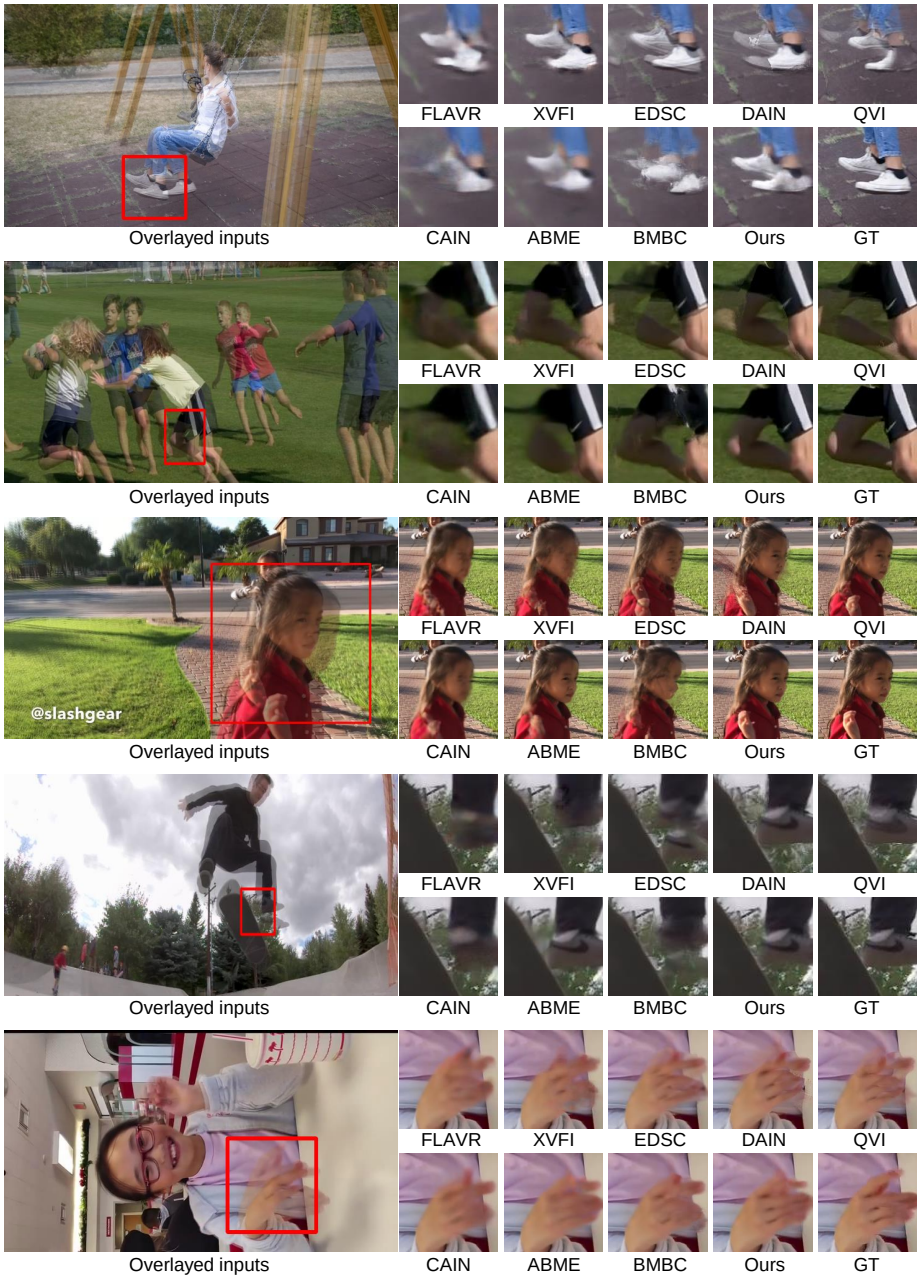

Fig. 3. Visual comparisons on DAVIS (top 1 raw) and SNU-FILM (bottom 4 rows).

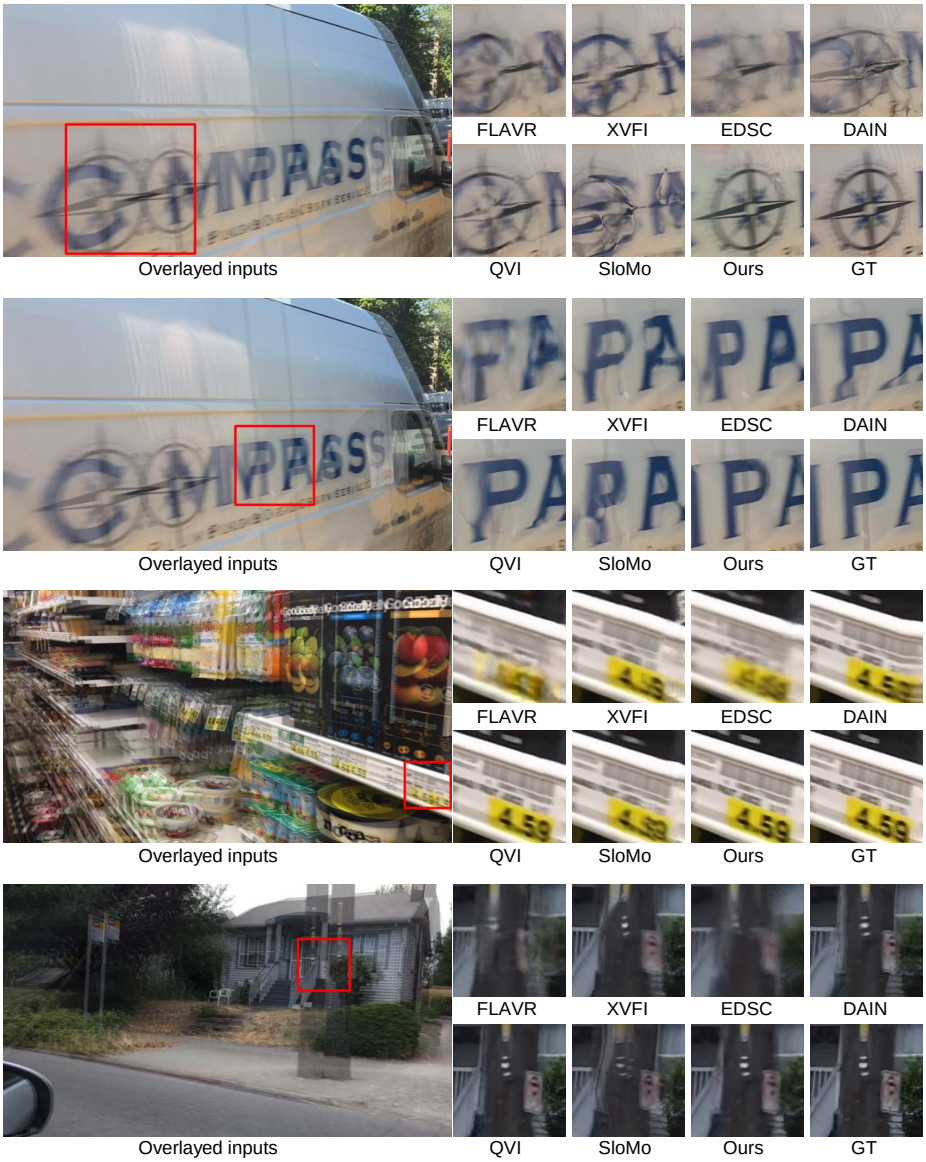

Fig. 4. Visual comparisons on Adobe240.

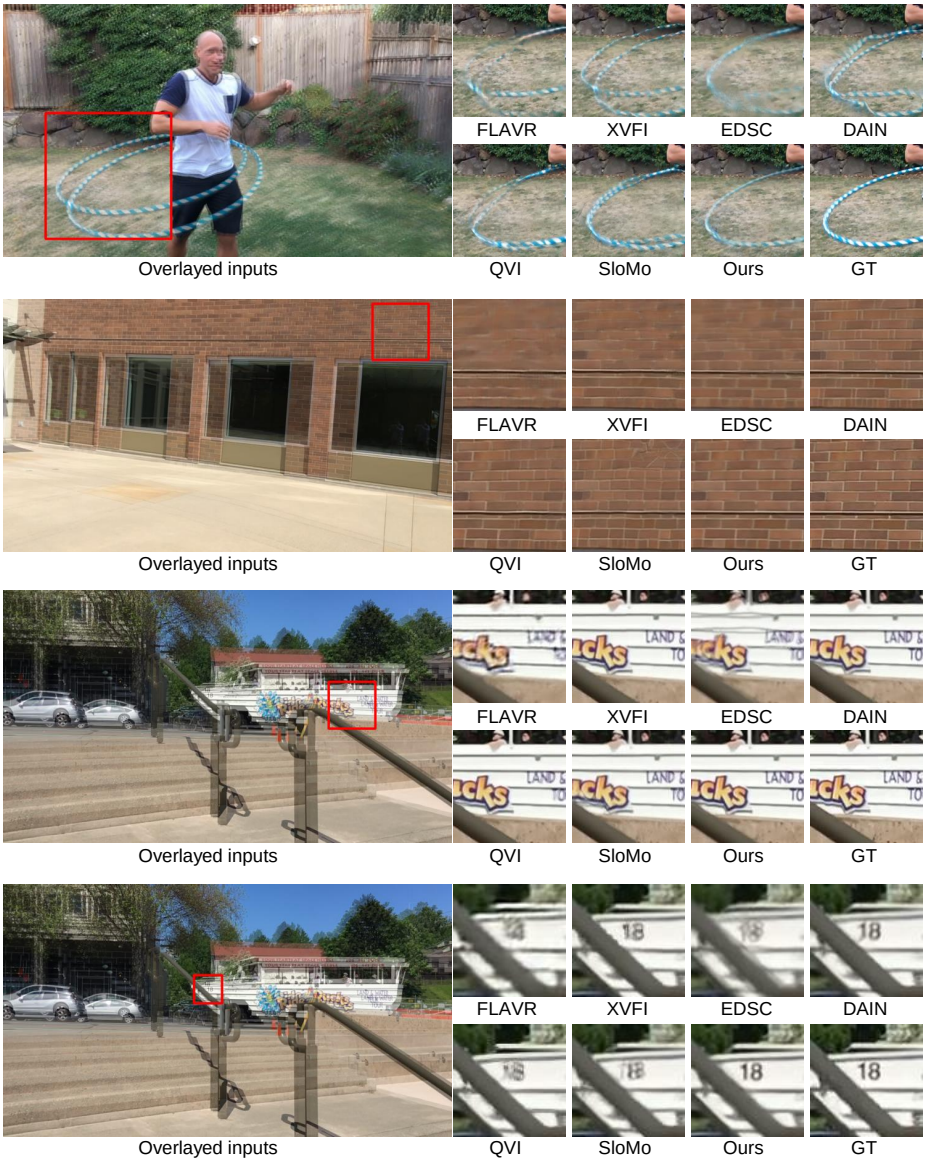

Fig. 5. Visual comparisons on Adobe240.

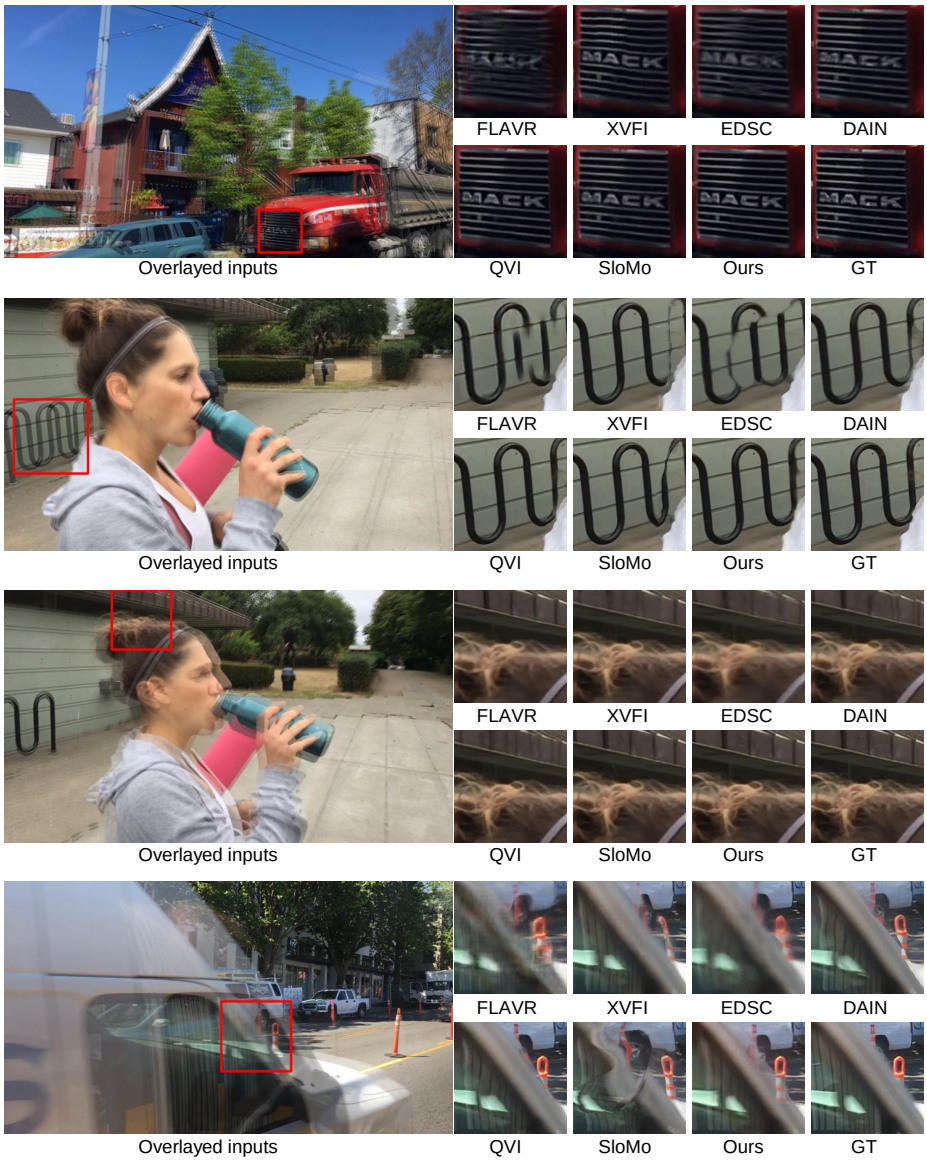

Fig. 6. Visual comparisons on Adobe240.

References

1. Bao, W., Lai, W.S., Ma, C., Zhang, X., Gao, Z., Yang, M.H.: Depth-aware video frame interpolation. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 3703–3712 (2019)
2. Cheng, X., Chen, Z.: Multiple video frame interpolation via enhanced deformable separable convolution. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) (2021)
3. Chi, Z., Nasiri, R.M., Liu, Z., Lu, J., Tang, J., Plataniotis, K.N.: All at once: Temporally adaptive multi-frame interpolation with advanced motion modeling. In: European Conference on Computer Vision (ECCV). vol. 12372, pp. 107–123 (2020)
4. Choi, M., Kim, H., Han, B., Xu, N., Lee, K.M.: Channel attention is all you need for video frame interpolation. In: AAAI Conference on Artificial Intelligence (AAAI). pp. 10663–10671 (2020)
5. Jiang, H., Sun, D., Jampani, V., Yang, M., Learned-Miller, E.G., Kautz, J.: Super slomo: High quality estimation of multiple intermediate frames for video interpolation. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 9000–9008 (2018)
6. Kalluri, T., Pathak, D., Chandraker, M., Tran, D.: Flavr: Flow-agnostic video representations for fast frame interpolation. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2021)
7. Nah, S., Hyun Kim, T., Mu Lee, K.: Deep multi-scale convolutional neural network for dynamic scene deblurring. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 3883–3891 (2017)
8. Niklaus, S., Liu, F.: Softmax splatting for video frame interpolation. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 5437–5446 (2020)
9. Park, J., Ko, K., Lee, C., Kim, C.: BMBC: bilateral motion estimation with bilateral cost volume for video interpolation. In: European Conference on Computer Vision (ECCV). vol. 12359, pp. 109–125 (2020)
10. Park, J., Lee, C., Kim, C.S.: Asymmetric bilateral motion estimation for video frame interpolation. In: IEEE International Conference on Computer Vision (ICCV). pp. 14539–14548 (2021)
11. Shangguan, W., Sun, Y., Gan, W., Kamilov, U.S.: Learning cross-video neural representations for high-quality frame interpolation. Arxiv preprint, 2203.00137 [cs.CV] (2022)
12. Shi, Z., Xu, X., Liu, X., Chen, J., Yang, M.H.: Video frame interpolation transformer. Arxiv preprint, 2111.13817 [cs.CV] (2021)
13. Sim, H., Oh, J., Kim, M.: Xvfi: Extreme video frame interpolation. In: IEEE International Conference on Computer Vision, (ICCV). pp. 14489–14498 (2021)
14. Su, S., Delbracio, M., Wang, J., Sapiro, G., Heidrich, W., Wang, O.: Deep video deblurring for hand-held cameras. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 1279–1288 (2017)
15. Xu, X., Siyao, L., Sun, W., Yin, Q., Yang, M.H.: Quadratic video interpolation. Advances in Neural Information Processing Systems (NeurIPS) **32**, 1645–1654 (2019)
16. Xue, T., Chen, B., Wu, J., Wei, D., Freeman, W.T.: Video enhancement with task-oriented flow. International Journal of Computer Vision (IJCV) **127**(8), 1106–1125 (2019)