FLAVR: Flow-Agnostic Video Representations for Fast Frame Interpolation

Supplementary Material

Tarun Kalluri * UCSD
Deepak Pathak CMU
Manmohan Chandraker UCSD
Du Tran Meta AI

https://tarun005.github.io/FLAVR/

We have provided additional qualitative results for frame interpolation as well as downstream applications in supplementary video available using this web link (link).

1. Ablations

In Tab. 1, we present a detailed ablation study of the proposed architecture design in terms of the skip connections, strides and loss functions. In addition to the brief insight provided in the main text, we explain each of them in detail next. We conduct all the ablation studies on the Vimeo-90K dataset.

Backbone Architecture In this work, we propose using 3D convolutions that model space-time relations for improved frame interpolation. To verify this hypothesis, we train a video interpolation network using 2D convolutions instead, and present the results in Tab. 1a. While training 2D Resnet, we concatenate RGB channels of the input before feeding into the network. We observe that the *R2D-18-2I* baseline, which uses a 2D ResNet-18 encoder decoder with 2 input frames ($C = 1$) performs worse than *2D-R18-4I* baseline, which uses 4 input frames ($C = 2$) justifying the need for a larger input context. Next, our proposed architecture *3D-R18-4I* which uses 3D convolutions along with 4 inputs, clearly outperforms both these baselines by 1.3 and 2.3dB, respectively. This indicates the importance of temporal modeling for the task.

In Fig. 1, we present a more detailed ablation about the effect of input context (C) on the performance of interpolation. From Fig. 1, we observe that for both $2\times$ and $8\times$ interpolations, using two input frames ($C = 1$), one each from past and present is sub-optimal, as it fails to accurately reason about complex motion profiles and occlusions. Furthermore, for $2\times$ interpolation, we found that a value of $C = 2$ gave the best result, and beyond that the performance saturates. This is because the outer frame generally contain less useful information for interpolation and in some cases might contain significant scene shifts which hurts the interpolation accuracy. In the case of $8\times$ interpolation, the time gap between the frames is tinier, so we find that a value of $C = 3$ performs the best, while any larger value of C hurts the accuracy.

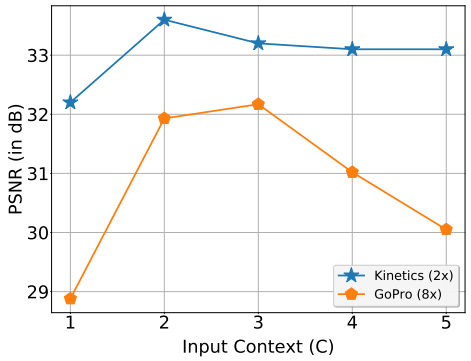

Figure 1: **Effect of Input Context** Comparison of the effect of input context, C, for video frame interpolation. For $2\times$ interpolation, we observed that a value of $C = 2$ which corresponds to using 4 input frames, 2 each from the past and future, gives best results. Beyond $C = 3$, we observe no further improvements. For $8\times$ interpolation, a value of $C = 3$ gave the best accuracy.

Choice of Fusion Tab. 1b compares and reports the different choices for the skip connection (in Figure 2 of the main submission) used for combining features across encoder and corresponding decoder. *No fusion* corresponds to having no skip connection between the layers of the encoder and decoder. While *fusion - add* corresponds to adding the features from the encoder to the decoder, *fusion - concat* refers to concatenating the corresponding feature maps along the channels. We find that using some kind of feature transfer across encoder and decoder is essential, than having *No fusion* (PSNR of 36.11 vs. 35.1), as the complementary information learnt in the low level and high level features needs to be aggregated for accurate interpolation. We settle on using *fusion - concat* in our final model as it gives better performance than *fusion - add*.

Temporal Striding Striding or pooling in CNNs are known to remove lot of fine level details in images, which are

*Work done during TK's internship at Meta AI.

Model	PSNR	SSIM
R2D-18-2I	33.98	0.966
R2D-18-4I	34.97	0.967
R3D-18-4I	**36.3**	**0.975**

(a) Effect of encoder arch.

Model	PSNR	SSIM
No fusion	35.1	0.9713
fusion - add	35.7	0.9737
fusion - concat	**36.3**	**0.975**

(b) Type of feature fusion

Model	PSNR	SSIM
w/o stride	**36.3**	**0.975**
w/ 2x stride	35.4	0.961
w/ 4x stride	35.21	0.96

(c) Effect of Temporal striding

Model	PSNR	SSIM
L1 Loss	**36.3**	**0.975**
L2 Loss	35.3	0.965
Huber Loss	35.3	0.964
L1+VGG Loss	35.91	0.962

(d) Effect of loss function

Table 1: **Ablation results** for FLAVR architecture on (a) different backbones, (b) fusion methods, (c) temporal striding, and (d) loss functions.

(a) Overlayed inputs (b) Activation w/ gating (c) Activation w/o gating

Figure 2: **Visualization of attention weighted feature maps** (a) The overlayed input frames. (b) The feature map of the channel with the highest attention weight in the network with feature gating. (c) The same feature map without using the gating module. We observe higher activation (red) in (b) along the motion boundaries. Best viewed in color.

essential for generative tasks like frame interpolation. We verify this with experiments using $2\times(1/2\times)$ and $4\times(1/4\times)$ temporal striding in the encoder(decoder), and observe from Tab. 1c that the performance decreases from 36.3 to 35.2 with larger temporal striding. We conclude that temporal striding hurts, and use a temporal stride of 1 in all the 3D convolution layers.

Channel Gating We visualize the role of channel gating module in the network in Fig. 2. We show the overlapped input frames in Fig. 2a to highlight the parts which have motion. In Fig. 2b and Fig. 2c, we plot the feature maps corresponding to the channel dimension with the largest activation while using and without using the feature gating respectively. We observe that the network trained with spatio-temporal gating (Fig. 2b) learns to focus on parts of input with visible motions (high activations in red), thus resulting in confident predictions of the interpolated motion estimates compared to Fig. 2c. In fact, training without spatio-temporal gating results in a drop in PSNR value from 36.3 to 36.1, further validating the utility of having the gating module.

Loss Function Many previous works [8] have studied the effect of using purely pixel loss vs. perception based losses [5]. Using only L1 or L2 loss would improve on the PSNR metric, but would cause blur in predictions. On the other hand, adding VGG based perception loss would result in sharper images visually. We observe from Tab. 1d that we did not improve upon the PSNR or SSIM metric by using any additional loss functions like VGG loss or Huber loss, apart from just L1 loss which also resulted in visually sharper images in our case.

	SSMO [4]	DAIN [1]	QVI [14]	FLAVR
PSNR	30.8	32.49	<u>36.29</u>	**37.82**
SSIM	0.924	0.957	<u>0.980</u>	**0.983**

Table 2: **Comparison with state-of-the-art methods for 4x interpolation** on Adobe dataset.

2. Multi-frame Interpolation

We show the results for $4\times$ interpolation in Tab. 2. FLAVR significantly outperforms other approaches on $4\times$ interpolation, in addition to results on $2\times$ and $8\times$ shown in the main paper.

3. Experiment Settings for downstream applications

3.1. Low-fps video object segmentation details

To examine the effectiveness of using the outputs of FLAVR, we choose the task of object segmentation in videos using mask propagation.

Motivation Achieving good label (or mask) propagation requires estimating perfect pixel level correspondences between frames of a video, using similarity between the respective feature maps. However, estimating such correspondences might be challenging if the frame sequences are extracted from low-fps videos. We want to validate if using FLAVR can improve low-fps video object segmentation.

Setup and baseline. DAVIS is the standard benchmark popularly used for video object segmentation which include videos at 30FPS. To adopt to low-fps setup, we purposely downsample DAVIS videos into lower frame rates, e.g., 15FPS or 8 FPS, and evaluate different object segmentation approaches on these low-fps videos. We choose CRW [3], the current state-of-the-art method for video object segmentation, as a baseline which is applied directly on downsampled low-fps videos. We then compare this baseline with using the same method, i.e. CRW, on interpolated videos generated by FLAVR by 2x or 4x interpolation from 15FPS or 8FPS videos. Results are shown in the main submission showing that FLAVR helps to improve low-fps video object tracking. The label propagation mechanism is the same as used in [3].

3.2. Motion magnification

Motion Magnification [9, 12, 13] deals with magnifying subtle yet important motions from videos, which are often imperceptible by human eyes. From [9], we define motion magnification as follows. For an Image $I(\mathbf{x}, t) = f(\mathbf{x} + \delta(\mathbf{x}, t))$, the goal of motion magnification is to generate an output image $\tilde{I}(\mathbf{x}, t)$ such that

$$\tilde{I}(\mathbf{x}, t) = f(\mathbf{x} + (1 + \alpha)\delta(\mathbf{x}, t)) \quad (1)$$

for a magnification factor α. For frame interpolation, $\alpha < 1$, since we are interested in what happens between two frames while for motion magnification, $\alpha > 1$, since we look to extrapolate existing motions beyond visible regime. While prior works [9,12,13] used custom architectures along with various post processing filters for this task, we offer a complementary perspective and look into how much a simple architecture like FLAVR pretrained on frame interpolation helps motion magnification. For this purpose, we use the synthetic dataset *CoCo-Synth* [9] to perform the training. We train the network for a fixed magnification factor of 10 ($\alpha = 10$). On this dataset, when compared to no pretraining at all, pretraining on FLAVR improved the SSIM values on a held-out validation set from 0.732 to 0.801. We provide sample videos after magnification and compare it with phase based approach [13] in our supplementary video. We emphasize that we do not apply any post processing such as temporal or spatial filters for removing noise on the outputs. The videos are generated directly as an output of the FLAVR architecture pretrained on frame interpolation, and finetuned for motion magnification.

3.3. Experiment setting for action recognition

For downstream experiments on action recognition, we use the train and validation split 1 of UCF101 [6] and HMDB51 [7]. We remove the decoder from the architecture and use the pretrained encoder along with a classifier (a global average pooling, a fully-connected layer, and a softmax) for training on downstream actions and add a temporal stride of 4. For UCF101, we use an input size of $32 \times 3 \times 224 \times 224$ and for HMDB51 we use an input size of $16 \times 3 \times 224 \times 224$ with a batch size of 16. The networks are fine-tuned using SGD with batch norm with a learning rate of 0.02 for 40 epochs. During inference, we sample 10 consecutive overlapping clips of length 32 from the test video and average predictions over all the clips.

3.4. Experiment setting for optical flow estimation

One crucial point to consider in downstream training on optical flow is that the flow networks generally take only two input frames which is considered too short for 3D CNN. Nevertheless, to examine the effectiveness of features learnt using frame interpolation for optical flow, we use the same encoder and decoder, and initialize the last prediction layer to output two channels instead (corresponding to x and y values of flow at each pixel). Since the interpolation network was trained to take 4-frame inputs, we apply copy padding to the inputs, e.g. repeating each input frame 2 times. We use an EPE (end point error) loss and train our network for 200 epochs. We report numbers using 5-fold cross validation over the MPI-Sintel clean and fina as well as Kitti subsets.

4. Qualitative Results

We show additional qualitative results by applying frame interpolation technique on insect motion videos in Fig. 3. We believe that this application is of immense use for closer inspection of biological properties from videos. We obtain videos from AntLab Youtube channel[1] that have insect take-off and flying captured at very high FPS. We down-sample the frame rate to 15FPS and apply our interpolation network to recover videos of higher frame rate. We apply our $8\times$ model once to obtain videos of 120FPS. The images are shown in Fig. 3. Complete videos are available in our supplementary video.

Middlebury Dataset.

We evaluate FLAVR on the publicly available test images from Middleburry dataset [11] on the task of single frame interpolation. However, Middleburry has test samples with only two input frames while FLAVR requires 4-frame inputs. In those examples, we simply duplicate them into 4 frames and evaluate with FLAVR. For two frame sequences like *teddy*, duplicating inputs is obviously sub-optimal. On sequences where multi-frame inputs are available, FLAVR outperforms most prior interpolation works like SuperSloMo [4], BMBC [10] and EDSC [2]. Qualitative results for some such sequences are presented in Fig. 4. The complete results are available on the public leaderboard.

5. User study

We carry the user study on the Amazon Mechanical Turk (AMT) platform. We select two representative works that belong to two broad families that perform linear (SuperSloMo [4]) and quadratic (QVI [14]) warping for multi-frame interpolation. Then, we compare each video generated by FLAVR with videos generated using each of SuperSloMo and QVI separately. For this purpose we use *all* 90 HD videos from the DAVIS dataset, generate $8\times$ interpolated videos and place the two interpolated videos one beside the other and randomly shuffle the order of videos. We then show each pair of videos to 6 AMT workers and ask them to choose which video, right or left, looked more realistic. The method preferred by more users is chosen as a winner for that particular video. In case of tie, that is if each method is chosen by 3 users, we place the video under "no

[1] https://www.youtube.com/user/adrianalansmith

Figure 3: Qualitative Results for $8\times$ video frame interpolation on Insect Motion Videos. Frame at $t = 0$ and $t = 1$ are given as inputs to the network to predict the remaining 7 intermediate frames. Original Videos acquired from `AntLab` Youtube Channel.

preference" category. Workers are paid in accordance with minimum wages rules. With this setting, we find that users overwhelmingly chose our videos in preference against SuperSloMo [4]. More details are provided in subsection 5.1 of the main paper.

6. Training details

We train the $2\times$ interpolation network on Vimeo-90K dataset and $4\times$ and $8\times$ interpolation networks on the GoPro dataset and use the official train and validation splits with the sampling strategy explained in subsection 3 of the paper. We use a crop size of 256×256 and 512×512 for Vimeo-90K and GoPro datasets, respectively. We employ random frame order reversal and random horizontal flipping as augmentation strategies on both the datasets. We use use an initial learning rate of 2×10^{-4} and divide the learning rate by 2 whenever the training plateaus. We train the $2\times$ interpolation network for 200 epochs, while $4\times$ and $8\times$ interpolation network were trained for 120 epochs. We use a mini-batch size of 64 on Vimeo-90K dataset and 32 on GoPro dataset, and train our network on 8 2080Ti GPUs. We reduce the learning rate by half whenever the training plateaus which is cross-validated by the validation set. We apply mean normalization once for every mini-batch of input frames separately rather than using global mean normalization or batch normalization inside the network to achieve training stability. We use 8 GPUs and a mini-batch of 32 to train each model, and training is completed in about 36 hours for $2\times$ and 22 hours for $8\times$ interpolation networks on Vimeo and GoPro datasets respectively.

7. Benchmarking inference time

The inference time benchmarking was performed using an NVIDIA-2080Ti GPU with 12GB memory. The calculated time only includes forward pass excluding the data pre-processing time and CPU/GPU transfer. The results were obtained by averaging over 100 samples from Adobe-240FPS dataset using 512×512 crop size. For multi-frame interpolation, the time required is calculated as the aggregate time required for interpolating all the frames. Non-blocking CUDA operations as well as GPU warm start time were accounted for during inference time computation.

8. Statement on potential negative impact

Frame interpolation aims to generate non-existent frames between existing frames of a video. While achieving state-of-the-art performance using simple architectures through FLAVR is a plus, any kind of generative models can be misused to forge or tamper a video which may have a negative impact on applications where outputs of FLAVR have a bearing on reliability. Moreover, one of the applications of FLAVR is to improve object tracking in videos, which might have a potential to be used in surveillance for nefarious purposes.

9. Overview of attached code

The training and testing code is provided along with the supplementary attachment. The trained models have also been provided for direct inference, but as download links due to supplementary size limits. Our trained model can also be used to create slomo videos starting from arbitrary videos (requires OpenCV 4.2.0). The README.md file contains instructions to run the code.

References

[1] Wenbo Bao, Wei-Sheng Lai, Chao Ma, Xiaoyun Zhang, Zhiyong Gao, and Ming-Hsuan Yang. Depth-aware video frame interpolation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3703–3712, 2019.

[2] Xianhang Cheng and Zhenzhong Chen. Multiple video frame interpolation via enhanced deformable separable convolution. *arXiv preprint arXiv:2006.08070*, 2020.

[3] Allan Jabri, Andrew Owens, and Alexei A Efros. Space-time correspondence as a contrastive random walk. *arXiv preprint arXiv:2006.14613*, 2020.

[4] Huaizu Jiang, Deqing Sun, Varun Jampani, Ming-Hsuan Yang, Erik Learned-Miller, and Jan Kautz. Super slomo: High quality estimation of multiple intermediate frames for video interpolation. In *Proceedings of*

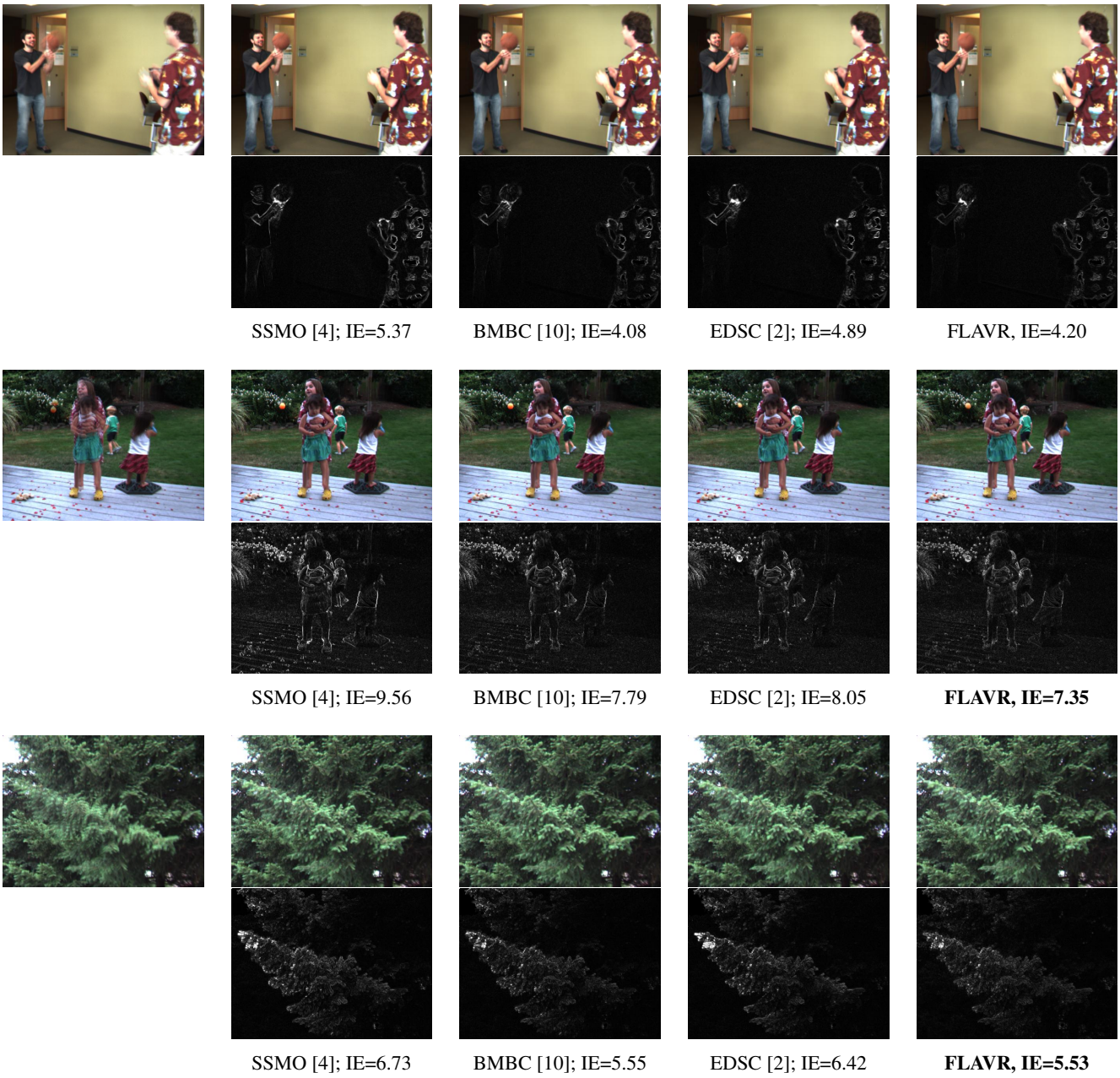

Figure 4: Interpolation results for 2× interpolation on Middleburry test set. The leftmost images in each row repesent the overlayed inputs. The first row in each set represents the interpolated frame, while the second row shows the error maps with respect to the ground truth. IE shows the interpolation error of the method. The interpolation errors for all the baselines are reported on the official leaderboard.

the IEEE Conference on Computer Vision and Pattern Recognition, pages 9000–9008, 2018.

[5] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *European conference on computer vision*, pages 694–711. Springer, 2016.

[6] Soomro Khurram, Zamir Amir, and Shah Mubarak. UCF101: A dataset of 101 human action classes from videos in the wild. *CRCV-TR-12-01*, 2012.

[7] Hildegard Kuehne, Hueihan Jhuang, Estíbaliz Garrote, Tomaso Poggio, and Thomas Serre. Hmdb: a large video database for human motion recognition. In *2011 International Conference on Computer Vision*, pages 2556–2563. IEEE, 2011.

[8] Simon Niklaus, Long Mai, and Feng Liu. Video frame interpolation via adaptive separable convolution. In

Proceedings of the IEEE International Conference on Computer Vision, pages 261–270, 2017.

[9] Tae-Hyun Oh, Ronnachai Jaroensri, Changil Kim, Mohamed Elgharib, Fr'edo Durand, William T Freeman, and Wojciech Matusik. Learning-based video motion magnification. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 633–648, 2018.

[10] Junheum Park, Keunsoo Ko, Chul Lee, and Chang-Su Kim. Bmbc: Bilateral motion estimation with bilateral cost volume for video interpolation. *arXiv preprint arXiv:2007.12622*, 2020.

[11] Daniel Scharstein, Heiko Hirschmüller, York Kitajima, Greg Krathwohl, Nera Nešić, Xi Wang, and Porter Westling. High-resolution stereo datasets with subpixel-accurate ground truth. In *German conference on pattern recognition*, pages 31–42. Springer, 2014.

[12] Neal Wadhwa, Michael Rubinstein, Frédo Durand, and William T. Freeman. Phase-based video motion processing. *ACM Trans. Graph. (Proceedings SIGGRAPH 2013)*, 32(4), 2013.

[13] Hao-Yu Wu, Michael Rubinstein, Eugene Shih, John Guttag, Frédo Durand, and William Freeman. Eulerian video magnification for revealing subtle changes in the world. *ACM transactions on graphics (TOG)*, 31(4):1–8, 2012.

[14] Xiangyu Xu, Li Siyao, Wenxiu Sun, Qian Yin, and Ming-Hsuan Yang. Quadratic video interpolation. In *Advances in Neural Information Processing Systems*, pages 1647–1656, 2019.