Exploring Discontinuity for Video Frame Interpolation

Sangjin Lee*,[1] Hyeongmin Lee*,[1] Chajin Shin[1] Hanbin Son[1] Sangyoun Lee[1,2]
[1] Yonsei University
[2] Korea Institute of Science and Technology (KIST)
{sglee97, minimonia, chajin, hbson, syleee}@yonsei.ac.kr

Abstract

Video frame interpolation (VFI) is the task that synthesizes the intermediate frame given two consecutive frames. Most of the previous studies have focused on appropriate frame warping operations and refinement modules for the warped frames. These studies have been conducted on natural videos containing only continuous motions. However, many practical videos contain various unnatural objects with discontinuous motions such as logos, user interfaces and subtitles. We propose three techniques that can make the existing deep learning-based VFI architectures robust to these elements. First is a novel data augmentation strategy called figure-text mixing (FTM) which can make the models learn discontinuous motions during training stage without any extra dataset. Second, we propose a simple but effective module that predicts a map called discontinuity map (D-map), which densely distinguishes between areas of continuous and discontinuous motions. Lastly, we propose loss functions to give supervisions of the discontinuous motion areas which can be applied along with FTM and D-map. We additionally collect a special test benchmark called Graphical Discontinuous Motion (GDM) dataset consisting of some mobile games and chatting videos. Applied to the various state-of-the-art VFI networks, our method significantly improves the interpolation qualities on the videos from not only GDM dataset, but also the existing benchmarks containing only continuous motions such as Vimeo90K, UCF101, and DAVIS.

1. Introduction

Video frame interpolation (VFI) task is to generate the intermediate frame given some consecutive frames from a video. When the time interval of each input frames is fixed, we can get smoother video, and when the frame rate is fixed, we can get slow-motion video. This can also be applied to other vision tasks such as video compression [2, 34], view synthesis [7, 13, 39], and other real-world applications [23, 29, 37].

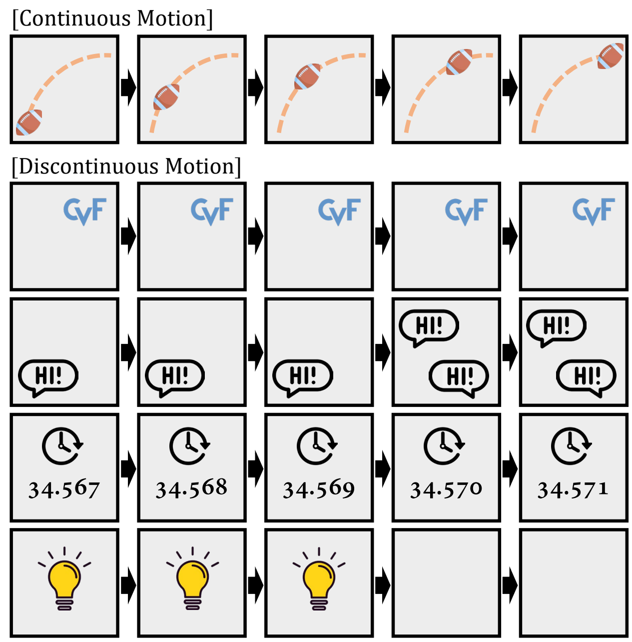


Figure 1. The examples of discontinuous motion.

Most of the previous works focus on the motion of the objects in videos. They utilize the estimated flow maps [12, 16], kernels [6, 15, 17, 21, 22], or externally estimated optical flow maps [19, 20, 24, 25, 29] to place each object in the middle of its position on the adjacent frames. However, as personal broadcast and cloud gaming contents increase, many of the practical videos contain special objects which do not move continuously such as user interfaces, watermarks, logos, chatting windows and subtitles (as the examples on Figure 1). Besides, these elements are received at the display devices as part of each frame, not as additional information. Therefore, many of the video enhancement frameworks, including VFI, should be improved to be robust to the discontinuous motions.

In this paper, our purpose is to expand the spectrum of motions to address efficiently both continuous and discon-

*Both authors contributed equally to this work.

tinuous ones, not focusing only on special videos with discontinuous motions. To achieve this goal, we propose three techniques to process the videos containing both types of motions. First, we propose novel data augmentation method called Figure-Text Mixing (FTM) which consists of Figure Mixing (FM) and Text Mixing (TM). FM is an augmentation of fixed random figures and TM is an augmentation of discontinuity moving random texts. The networks can learn to be robust on both continuous and discontinuous motions using FTM without any additional train datasets. Second, we propose a lightweight module which estimates a map called discontinuity map (D-map), which determines whether the motion of each output pixel is continuous or discontinuous. When estimating the pixels in the discontinuous area, a pixel on the same location of one of the input frames is copied instead of predicting the interpolated value. To prove the versatility and adaptivity of our D-map estimation module, we apply it to various state-of-the-art VFI models instead of proposing our original VFI architecture. Lastly, if we utilize both FTM and D-map, it is possible to supervise the model by giving the ground-truth of D-map. Therefore, we propose an additional loss function to help our model estimate D-map easily.

We construct a special test set called Graphic Discontinuous Motion (GDM) dataset to evaluate how our method and the competitive works deal with the discontinuous motions. Our approach shows significantly improved results compared to the other methods on GDM dataset. Moreover, our method outperforms those methods on the regular benchmarks that only contain continuous motions such as Vimeo90K test [36], DAVIS [26] and UCF101 [30] datasets. Our main contributions can be summarized as follows:

- *New Data Augmentation Strategy.* We propose a new data augmentation strategy called FTM that, when applied to existing video datasets, makes models learn both continuous and discontinuous motions without any additional train datasets.
- *New Module & Loss Function.* We propose a new module which can separate continuous and discontinuous motions. This module can be applied to many recent deep learning-based VFI architectures. We also propose a loss function to supervise the module.
- *Performance.* Applied to various state-of-the-art VFI models, our method achieves performance improvement on not only the dataset containing discontinuous motions, but also many of the other benchmarks with only continuous motions.

2. Related Work

Most existing video frame interpolation algorithms consist of two parts: motion estimation and motion compensation. Motion estimation modules estimate the pixel level correspondences between two consecutive frames to get motion information. Then motion compensation parts warp the frames according to the estimated motion. Recent video frame interpolation researches utilize deep neural networks (DNN) to obtain high quality results in two ways.

One method is the end-to-end learning approach. Several works train their neural networks, which perform both motion estimation and compensation at the same time. Niklaus *et al.* [21] propose a network that estimates big kernel weights for all pixels of input frames. Then they adaptively convolve the input frames with the estimated kernels to obtain the output frame. Since the large kernel size requires excessively many weights, Niklaus *et al.* [22] solve this problem by using separable kernels. On the other hand, Liu *et al.* [16] and Jiang *et al.* [12] propose the neural network that estimates dense flow map, which consists of the vectors directly pointing to the reference pixels. However, the above methods have limitations that the kernel-based ones cannot deal with the motion beyond the kernels, and the flow-based ones only refer to one pixel for each output pixel. To solve the problem, Lee *et al.* [21] combine the two methods using the deformable convolution [5]. Some approaches propose the neural networks that directly estimate the intermediate frames without motion compensation. Long *et al.* [17] train a simple U-Net [27] to estimate the intermediate frames, but the results tend to be blurry. Therefore, Choi *et al.* [4] propose a new architecture based on channel attention to obtain sharper results. Recently, transformer-based algorithms [18, 28] have been proposed. Lu *et al.* [18] identify a limitation of convolution operations which can only handle small motion and solve this with transformer architecture. Shi *et al.* [28] expand from Lee *et al.* [21] to produce high quality result by adding transformer-based block.

The second method is the optical flow-based approach. Recently, many approaches to estimate high-quality optical flow maps have been introduced [11, 24, 25, 31, 32]. Therefore, several works make use of the optical flow maps as motion information and train additional networks for motion compensation or output frame refinement. Niklaus *et al.* [19] utilize the context information extracted from ResNet-18 [9] along with the optical flows and refine the warped frames using their own neural network based on GridNet [8]. Using pre-trained optical flow has a problem that the flow maps consist of the vectors starting from the input frames. However, the vectors starting from the output frame are necessary to warp the frames clearly. To deal with this problem, Bao *et al.* [1] use the depth maps obtained from the hourglass architecture-based mono depth estimation network [3] to invert the optical flow maps clearly. Niklaus *et al.* [20] propose to combine all pixel values that are projected into the same locations using SoftMax function. Park *et al.* [24] use the symmetric bilateral motions estimated as a linear to improve the quality of the motion. However, considering the limitation of handling the region where the constraint does not exist, Park *et al.* [25] expands the symmetric to asymmetric motion for robust video frame interpolation. On the other hand, several algorithms are proposed in terms of efficiency. Kong *et al.* [14] proposes warp and refine immediately with pyramid architecture and demonstrates that this approach is efficient compared with previous flow-based algorithms. Huang *et al.* [10] achieves real-time flow estimation by employing not pre-trained flow network but end-to-end convolution network.

There are some researches to expand the domain of videos to be interpolated. Xiangyu *et al.* [35] propose the quadratic video interpolation approach which utilizes four frames to cover not only linear motions, but also quadratic motions. However, they still cannot incorporate discontinuous motions. Lastly, Li *et al.* [29] propose the network that can interpolate cartoon videos. However, they focus on the characteristics of cartoon images, not the motion of the videos. In this paper, we expand the video interpolation task to cover not only the natural motions, but also discontinuous transitions between the frames.

3. Proposed Approach

Given the two consecutive video frames $I_1, I_2 \in \mathbb{R}^{H\times W\times C}$ (sometimes along with additional frames), recent deep learning-based VFI methods try to build a network $\mathcal{F}$ to estimate the intermediate frame $\hat{I}$. Omitting the additional frames, we can express the VFI frameworks as follows:

$$\hat{I} = \mathcal{F}(I_1, I_2) \tag{1}$$

However, most of the networks are designed to deal with only continuous motions. To make them robust to the discontinuous motions like in Figure 1, there are two main problems that need to be solved. First, they are mainly trained on the datasets containing only continuous motions such as Vimeo90K [36]. To solve this problem without collecting additional train dataset, we propose a data augmentation method called FTM, which is introduced in Section 3.1. Second, many of the architectures make use of motion information such as optical flow, which limits the models to considering only the continuous motions. To solve this problem, we propose a new module that predicts discontinuity map (D-map), which is introduced in Section 3.2. Applied to many existing networks, it makes them able to deal with both continuous and discontinuous motions better.

Figure 2. Example of augmented training data. Figure-Text Mixing is applied randomly in units of sequence.

3.1. Figure-Text Mixing (FTM)

Previous studies have only applied flip augmentation with spatial and temporal axes. However, this general augmentation is not sufficient alone to address various types of videos. We proposed a new data augmentation method for frame interpolation, called *Figure-Text mixing* (FTM), to handle general video frame interpolation. FTM consisted of two types of data augmentation methods: *figure mixing* (FM) and *text mixing* (TM). The discontinuity map, i.e., the results of our network, is the difference map between the original ground-truth frame and augmented ground-truth frame in the training stage. We can train the proposed network with the supervision of the discontinuity map by applying the data augmentation technique.

Figure Mixing. We added figures to the input frames to address the static objects in videos. The added figures had the same position and property on all input frames. We randomly applied the augmentation for the video sequence (see Figure 2). This technique can be the guideline where the discontinuous motion area exists. This method also maintained the edge of an object from collapsing, even if the object had continuous motion.

Text Mixing. There were many letters or sentences in videos, such as chatting and watermark. We included text on the video in the following four methods owing to the difference between the property of discontinuous and continuous motion areas: 1) the position of the text is static in the entire video, 2) where the text does not exist in the previous frame and appears in the future frame, 3) in the opposite of case 2), 4) the position of the text changes. We set the method of FTM in which the ground-truth frame and the discontinuity map follow the previous frame augmentation

Figure 3. Visualization of Discontinuity map (D-map) estimated for a sample. AdaCoF and VFIT-B are the baseline networks where we apply our methods. D-map commonly shows the user interface elements (UI) that are fixed on their position and some digits that change dynamically (highlighted in red boxes).

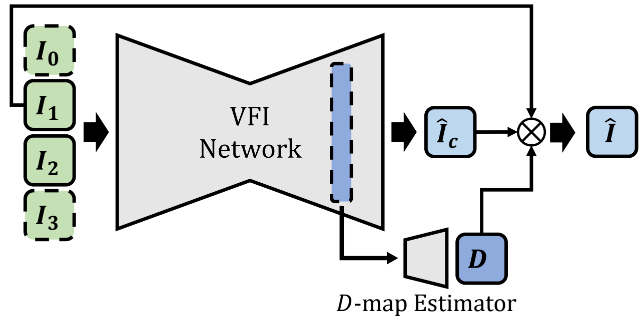


Figure 4. The architecture of our method. VFI network can be various state-of-the-art models. Our modifications from the baseline are the number of frames and D-map estimator.

in text mixing. Therefore, we applied data augmentation, as given in Figure 2. The first row of Figure 2 is the case of 2), and the second row is the case of 4).

3.2. Discontinuity Map (D-map)

Some examples of discontinuous motions in Figure 1 share an important property: interpolation of discontinuously moving object can be done by simply copying it onto one of the adjacent frames. However, this method may pose minor problems. First is the problem of determining which frame to copy between the front and back frames. The second is some samples where the *copying strategy* does not work like in the fourth row of Figure 1. However, the problem of interpolating these cases is ill-posed fundamentally. For example, it is difficult to identify which digit should be generated when it changes frame-by-frame. Considering the visual quality, even for the above two cases, it is more appropriate to simply copy the previous or next frame than let the motion compensation model to solve this problem.

Therefore, we additionally estimate a 1-channel map called Discontinuity map (D-map) to distinguish the continuous and discontinuous motion areas. The pixels in continuous areas are estimated by the baseline VFI network, and those in discontinuous areas are copied from the previous frame. When $D \in (0,1)^{H\times W\times 1}$ represents a D-map, the proposed VFI process can be expressed as follows:

$$\hat{I}_c = \mathcal{F}(I_1, I_2) \tag{2}$$

$$\hat{I}(\mathbf{x}) = \hat{I}_c(\mathbf{x}) \cdot (1 - D(\mathbf{x})) + I_1(\mathbf{x}) \cdot D(\mathbf{x}), \tag{3}$$

where $\hat{I}_c$ is the *continuously* interpolated frame and $\mathbf{x} \in [1, H] \times [1, W]$ represents the location. For example, we discovered that D-map commonly indicates UI elements or digits which have discontinuous motion regardless of baseline networks in Figure 3. More examples of D-map are shown in supplementary material.

Discontinuity map estimation. Given a conventional VFI network $\mathcal{F}$ as a baseline, D-map is estimated by a lightweight CNN-based module. The input of this module is a feature map obtained from a specific layer of $\mathcal{F}$ in Figure 4. We select the most appropriate layer for each network and the details are presented in supplementary material. In addition, we increase the number of input frames of the baseline networks to four because the *continuity* and *discontinuity* cannot be defined with only two frames. For fair comparison with the previous methods, we also compare the 2-frame versions in Section 4. Finally, the modified network (4 frame input and D-map estimation module-added) is trained with the loss functions that are introduced in Section 3.3 (with FTM applied as well).

3.3. Objective Functions

Loss Function. First, we reduced the difference between the model output I_{out} and ground truth I_{gt}. We used ℓ_1 norm for the loss as follows:

$$\mathcal{L}_1 = \|I_{out} - I_{gt}\|_1 \; . \tag{4}$$

We use the Charbonnier Function $\Phi(x) = (x^2 + \epsilon^2)^{1/2}$ for optimizing ℓ_1 distance, where $\epsilon = 0.001$ by following Liu *et al.* [16].

Discontinuity map Supervision. When both FTM and D-map are applied in training stage, the ground truth of D-map D_{gt} can be obtained by simply getting the areas of added figures and texts. Therefore, we can provide the D-map supervision to the network to learn the location of discontinuous motion areas. We use ℓ_1 loss between D and D_{gt} as follows:

$$\mathcal{L}_D = \|D - D_{gt}\|_1 \; . \tag{5}$$

We find that just adding the two losses with same rate works well. Therefore, the total loss for training is as follows:

$$\mathcal{L}_{total} = \mathcal{L}_1 + \mathcal{L}_D \; . \tag{6}$$

	Vimeo-90k			UCF101			DAVIS			GDM		
	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS
DVF [16]	32.792	0.9359	0.0395	32.122	0.9366	**0.0356**	25.596	**0.8084**	0.1452	28.709	0.9118	0.0753
DVF-*FTM*	**32.833**	**0.9361**	**0.0395**	**32.162**	**0.9369**	0.0360	**25.605**	0.8078	**0.1449**	**28.984**	**0.9134**	**0.0728**
SepConv [22]	32.889	0.9306	0.0681	32.993	0.9416	0.0429	**25.952**	**0.8108**	0.2209	29.667	0.9197	0.0870
SepConv-*FTM*	**33.064**	**0.9325**	**0.0665**	**33.157**	**0.9428**	**0.0418**	25.873	0.8103	**0.2188**	**29.914**	**0.9235**	**0.0797**
AdaCoF [15]	34.103	0.9459	0.0427	33.320	**0.9438**	0.0353	**26.791**	**0.8353**	0.1643	29.980	0.9227	0.0803
AdaCoF-*FTM*	**34.167**	**0.9497**	**0.0396**	**33.333**	0.9437	**0.0345**	26.719	0.8342	**0.1561**	**30.064**	**0.9278**	**0.0728**
CAIN [4]	34.699	0.9514	0.0421	33.306	0.9444	0.0373	**27.449**	0.8511	**0.1855**	30.238	0.9284	0.0807
CAIN-*FTM*	**34.987**	**0.9537**	**0.0414**	**33.448**	**0.9453**	**0.0363**	27.427	**0.8528**	0.1879	**30.429**	**0.9328**	**0.0776**
VFIT-B [28]	**36.743**	**0.9638**	**0.0318**	33.769	0.9472	**0.0363**	28.090	**0.8640**	**0.1442**	30.019	0.9280	0.0736
VFIT-B-*FTM*	36.695	0.9633	0.0322	**33.821**	**0.9474**	0.0377	**28.140**	0.8627	0.1506	**30.473**	**0.9321**	**0.0693**

Table 1. Effect of the Figure-Text Mixing (FTM) augmentation. The label '-*FTM*' means the model trained with FTM applied. The **bold highlights** mean that the performances are improved by FTM.

4. Experiments

4.1. Experimental Settings

Training Settings. We use Vimeo90K [36] septuplet train dataset for training, which consists of 91,701 sequences of seven 448×256 frames. For the data augmentations, we randomly crop the 256×256 patches and flip them horizontally, vertically, and temporally. We then add the proposed data augmentation (FTM). For the other options such as learning rates, scheduling, batch sizes, and the optimizers, we follow the respective papers where each baseline network is proposed. Note that the proposed methods are all that can be applied to various baseline models.

Evaluation Setting. For the evaluation, we select some recent VFI networks each representing the warping-based [15], direct prediction-based [4], and transformer-based [28] architecture. Then we observe the performance changes as the three proposed ideas (FTM, D-map, and $\mathcal{L}_D$) are applied. We select three test benchmarks that are commonly used: Vimeo90K [36] test dataset, UCF101 [30], DAVIS test-dev dataset [26]. In addition, we construct a new test set called Graphic Discontinuous Motion (GDM) dataset which consists of high resolution videos of game scenes with abundant discontinuous motions. There are some more details of GDM dataset in the supplementary material. We evaluate each algorithm by comparing PSNR (Peak Signal-to-Noise Ratio), SSIM (Structural Similarity) [33], and LPIPS (Learned Perceptual Image Patch Similarity) [38] for all test datasets.

4.2. Ablation Study

Figure-Text Mixing (FTM). To figure out the effectiveness of FTM, we first train some recent VFI networks (AdaCoF [15], DVF [16], SepConv [22], CAIN [4], and VFIT-B [28]) following their own training strategy. Then we train them again in the same ways with FTM applied. Table 1 presents that FTM significantly improves the interpolation performance especially in GDM dataset. FTM is also effective on the conventional test benchmarks with only continuous motions such as Viemo90k, UCF101, and DAVIS in many cases.

Discontinuity Map. As mentioned in Section 4.1, we apply D-map to the three recent VFI networks: AdaCoF [15], CAIN [4], and VFIT-B [28]. Then we train three or four versions of each network. First is the baseline network, second is the D-map applied version and the third is the D-map applied one with the supervision loss $\mathcal{L}_D$. Except for VFIT-B which requires four input frames, we additionally train the versions with two input frames for fair comparison (note that D-map applied versions basically require four frames). Table 2 presents how D-map and $\mathcal{L}_D$ affect the performance. In short, applying D-map estimation module significantly improves the interpolation quality on GDM dataset. Moreover, some results show improved performance for the other datasets as well, and relatively small amounts of degradation gaps in some cases. For the models with only two input frames, AdaCoF-D (2) illustrates better performance than the baseline. CAIN-D (2) sometimes reveals lower quality on Vimeo 90K and DAVIS dataset, while it shows improvement on GDM dataset. Considering that our objective is to make VFI models robust to discontinuous motions without significantly degrading performance over continuous motions, the ideas clearly achieve our intention. In addition, we can also confirm that $\mathcal{L}_D$ is more effective on AdaCoF, compared to the other baselines.

4.3. Quantitative Results

We evaluate our network with several previous algorithms including DVF [16], SuperSlomo [12], SepConv [22], AdaCoF [15], Softsplat [20], CAIN [4],

	Vimeo-90k			UCF101			DAVIS			GDM		
	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS
AdaCoF (2) [15]	34.103	0.9459	0.0427	33.320	0.9438	0.0353	26.791	0.8353	0.1643	29.980	0.9227	0.0803
AdaCoF-D (2)	34.385	0.9483	0.0412	33.355	0.9439	0.0358	26.754	0.8356	0.1605	30.097	0.9260	0.0787
AdaCoF-D (4)	35.240	0.9534	**0.0384**	33.595	0.9452	0.0353	27.171	**0.8425**	0.1635	30.212	0.9255	0.0793
AdaCoF-D-$\mathcal{L}_D$ (4)	**35.267**	**0.9535**	0.0385	**33.611**	**0.9453**	**0.0350**	**27.226**	0.8424	**0.1597**	**30.242**	**0.9296**	**0.0760**
CAIN (2) [4]	34.699	0.9514	0.0421	33.306	0.9444	**0.0373**	27.449	0.8511	0.1855	30.238	0.9284	0.0807
CAIN-D (2)	34.546	0.9500	0.0455	33.319	0.9445	0.0375	27.301	0.8499	0.1927	30.616	0.9314	0.0792
CAIN-D (4)	**35.540**	**0.9560**	**0.0405**	**33.618**	**0.9461**	0.0376	**28.021**	**0.8612**	**0.1852**	**31.011**	**0.9375**	**0.0664**
CAIN-D-$\mathcal{L}_D$ (4)	35.169	0.9526	0.0451	33.589	0.9456	0.0390	27.897	0.8571	0.2019	30.936	0.9293	0.0703
VFIT-B (4) [28]	**36.743**	**0.9638**	**0.0318**	33.769	0.9472	**0.0363**	**28.090**	**0.8640**	**0.1442**	30.019	0.9280	0.0736
VFIT-B-D (4)	36.650	0.9634	0.0320	33.819	0.9474	0.0367	28.026	0.8621	0.1507	**30.965**	**0.9381**	0.0652
VFIT-B-D-$\mathcal{L}_D$ (4)	36.671	0.9631	0.0324	**33.823**	**0.9475**	0.0370	28.056	0.8625	0.1507	30.921	0.9371	**0.0645**

Table 2. Effect of applying D-map and $\mathcal{L}_D$. '-D' and '-$\mathcal{L}_D$' indicate the versions applying D-map and $\mathcal{L}_D$ each. The labels '(2)' and '(4)' mean the number of input frames. The **bold highlights** imply that the performances are the best among the versions sharing the same baseline.

	Vimeo-90k			UCF101			DAVIS			GDM		
	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS
DVF (2) [16]	32.792	0.9359	0.0395	32.333	0.9397	0.0340	24.087	0.7852	0.1588	28.709	0.9118	0.0753
SuperSlomo (2) [12]	30.812	0.9291	0.0482	28.500	0.9228	0.0564	26.259	0.8303	0.1206	27.651	0.8911	0.1117
SepConv-$\mathcal{L}_1$ (2) [22]	33.729	0.9454	0.0335	33.075	0.9419	0.0333	26.550	0.8376	0.1478	29.696	0.9082	0.1037
AdaCoF (2) [15]	34.103	0.9459	0.0427	33.320	0.9438	0.0353	26.791	0.8353	0.1643	29.980	0.9227	0.0803
Softsplat-$\mathcal{L}_1$ (2) [20]	33.723	0.9452	0.0336	33.112	0.9419	0.0332	26.542	0.8376	0.1479	29.667	0.9086	0.1039
CAIN (2) [4]	34.699	0.9514	0.0421	33.306	0.9444	0.0373	27.449	0.8511	0.1855	30.238	0.9284	0.0807
ABME (2) [25]	35.846	0.9584	0.0309	33.542	0.9458	0.0383	27.661	0.8601	0.1320	29.472	0.9209	0.0958
RIFE (2) [10]	34.048	0.9449	**0.0233**	33.184	0.9412	**0.0284**	27.246	0.8471	**0.0925**	30.085	0.9088	0.0801
IFRNet (2) [14]	35.837	0.9597	**0.0274**	33.451	0.9450	**0.0330**	27.467	0.8596	**0.1261**	30.239	0.9277	**0.0706**
VFIT-B (4) [28]	**36.963**	**0.9649**	0.0304	**33.837**	**0.9474**	0.0367	**28.153**	**0.8652**	0.1440	30.217	0.9274	0.0760
CAIN-D (2)	34.546	0.9500	0.0455	33.319	0.9445	0.0375	27.301	0.8499	0.1927	**30.616**	**0.9314**	0.0792
VFIT-B-D-$\mathcal{L}_D$ (4)	**36.671**	**0.9631**	0.0324	**33.823**	**0.9475**	0.0370	**28.056**	**0.8625**	0.1507	**30.921**	**0.9371**	**0.0645**

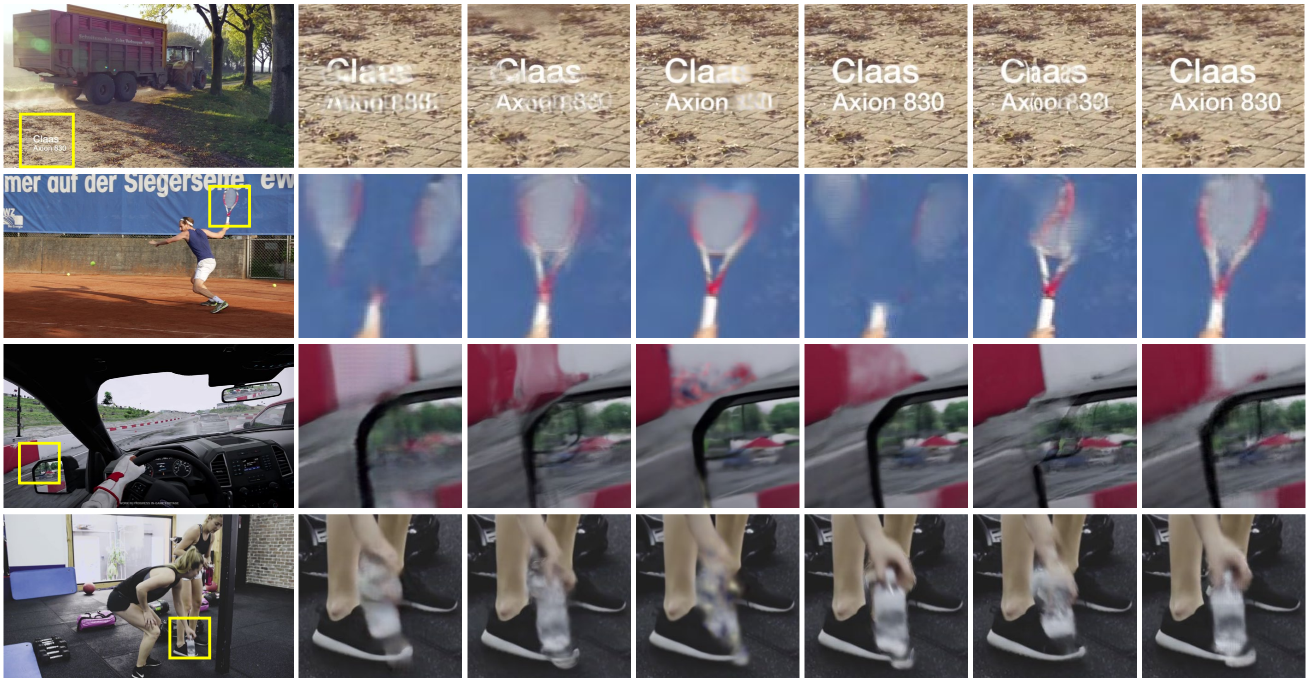
Table 3. Quantitative evaluation on three datasets. The labels '(2)' and '(4)' indicate the number of input frames. The **red highlights** define the best, **blue highlights** second best.

ABME [25], RIFE [10], IFRNet [14], and VFIT-B [28]. Since we have many possible versions, we select two models for comparison: CAIN-D (2) representing two-input-frames model and VFIT-B + D-map-4 + $\mathcal{L}_D$ representing four-input-frames model. As shown in Table 3, our methods outperform the previous algorithms on GDM dataset for three metrics by a high margin (note that the previous algorithms are pretrained models provided and our two networks are re-trained with the proposed method). Especially, despite the lower performance of CAIN baseline compared to the other networks, CAIN-D (2) achieves second best performance on GDM dataset. These results clearly show both the failures of the previous methods and the robustness of our approach for discontinuous motion. Moreover, our methods still show competitive performance among the state-of-the-art VFI networks even on continuous motions.

Table 1 shows that FTM can be used for training any algorithms and leads high performance for discontinuous motion. We can obtain better performance on GDM dataset for five algorithms without using extra training datasets. Table 2 shows that our ideas clearly improve the existing models. These results proves that our module can apply to any previous models and successfully improve the performance of baseline models on GDM dataset. Our methods achieve high performances on GDM dataset with slightly lower or higher quality on other three datasets.

4.4. Qualitative Results

We apply our methods on various types of videos to demonstrate the effectiveness of our ideas. We visually compare the results on two types of datasets each containing continuous and discontinuous motions. Since there are multiple versions of the proposed method, we select the results from VFIT-B-D-$\mathcal{L}_D$ in Figures 5, 6, 7 as *Ours*.



GT CAIN AdaCoF ABME IFRNet RIFE Ours

Figure 5. Qualitative results on DAVIS test-dev dataset.

Continuous motion. We select the DAVIS test-dev dataset [26] to compare the results on continuous motions, generally addressed in previous studies. We compare our method with CAIN [4], AdaCoF [15], ABME [25], IFRNet [14], and RIFE [10]. As shown in Figure 5, the other algorithms suffer from afterimage effects for large motion (see 2nd 4th row of Figure 5). On the other hand, our method shows relatively clear results. In the case of the sequence that has both continuous and discontinuous motions (1st row of Figure 5), the texts which need to maintain their rigid shape are distorted by the previous methods while our method show relatively clear results. In conclusion, compared to the other approaches, our method not only maintains the quality for continuous domain, but also produces sharp and clear results for large motion by maintaining the structure of the objects.

Discontinuous motion. We conduct two types of comparisons to demonstrate the effectiveness on discontinuous motion: comparison with previous algorithms on discontinuous video sequences from GDM dataset in Figure 6, and comparison between baseline networks and ones with our methods applied in Figure 7. In Figure 6, there are various types of discontinuous motions that were illustrated in Figure 1. We focus on two issues from the results of previous methods: 1) they sometimes fail to distinguish between continuous and discontinuous motion area, and 2) they have difficulty maintaining the structure of the discontinuously moving object, while they perform well in continuous motion area. Especially for the fixed objects, although all the VFI models need to do is just copying them from the previous frame, they sometimes suffer from some artifacts, while our method shows the clear shape compared with previous algorithms (See 2nd and 4th row of Figure 6). Similarly, the previous algorithms produce interpolated or inaccurate image for changing numbers and suddenly appearing objects (See 1st and 3rd row of Figure 6). However, our method shows better results by recognizing the momentarily changing scene. Figure 6 proves the limitations of previous algorithms on discontinuous motions and shows that our ideas can effectively cover both continuous and discontinuous domains.

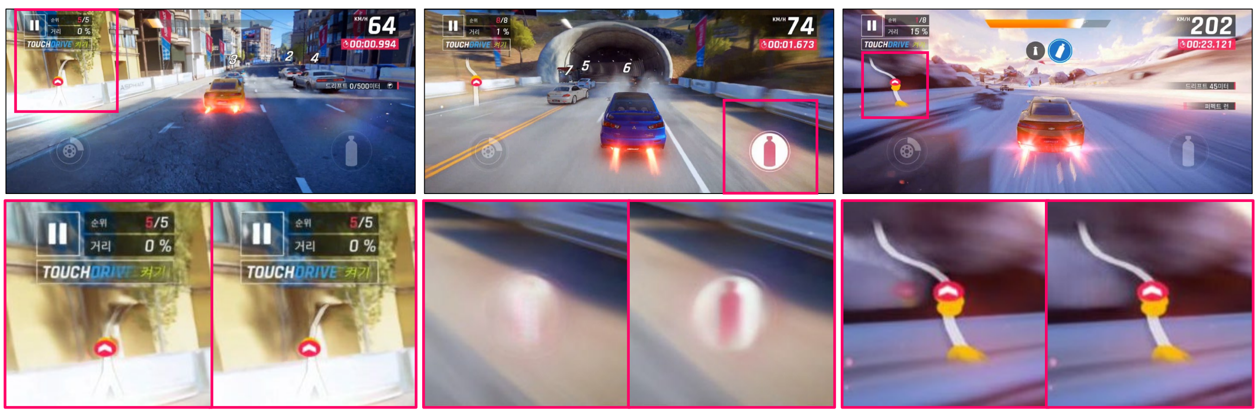
Figure 7 shows the flexibility and effectiveness of our methods by revealing each baseline networks with applied our ideas. +Ours indicates applying FTM, D-map and $\mathcal{L}_D$ to each baseline network. The results from baseline algorithms suffer from some artifacts, but applying our ideas leads to clear and sharper results on discontinuous motions. Figure 7 clearly shows that our methods can help the existing VFI models to be robust to the discontinuous motions and can be applied to various baselines.

5. Limitations

While FTM and D-map approaches can cover most cases of discontinuous motions, there are many other cases beyond that coverage. The main reason of the limitations is

GT CAIN AdaCoF ABME IFRNet RIFE Ours

Figure 6. Visual comparison of with discontinuous motion.



AdaCoF AdaCoF+Ours CAIN CAIN+Ours VFIT VFIT+Ours

Figure 7. Visual comparison between baseline algorithms and re-training with our methods.

that our definition of discontinuous motion and the proposed ideas are somewhat naive, which make the solution sub-optimal. However, it is true that our approach can make the existing VFI models significantly more robust. Therefore, we hope some further researchers focus on more fundamental natures of discontinuous motions and propose more optimal solutions.

6. Conclusion

In this paper, we present the novel network for general video frame interpolation to address both continuous and discontinuous motion areas. We also propose a new data augmentation, named figure-text mixing (FTM), to handle discontinuous motion areas and to overcome large motion. The evaluation shows state-of-the-art results for all kinds of datasets and indicators. These results imply that our network performs well for the general motion area without using extra datasets. We also demonstrated that the proposed architecture and data augmentation are effective solutions to handle general motion in various domains in ablation studies.

Acknowledgements. This research was supported by R&D program for Advanced Integrated-intelligence for Identification (AIID) through the National Research Foundation of KOREA(NRF) funded by Ministry of Science and ICT (NRF-2018M3E3A1057289) and the KIST Institutional Program(Project No.2E32283-23-064).

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

[2] Wenbo Bao, Xiaoyun Zhang, Li Chen, Lianghui Ding, and Zhiyong Gao. High-order model and dynamic filtering for frame rate up-conversion. *IEEE Transactions on Image Processing*, 27(8):3813–3826, 2018. 1

[3] Weifeng Chen, Zhao Fu, Dawei Yang, and Jia Deng. Single-image depth perception in the wild. In *Advances in neural information processing systems*, pages 730–738, 2016. 2

[4] Myungsub Choi, Heewon Kim, Bohyung Han, Ning Xu, and Kyoung Mu Lee. Channel attention is all you need for video frame interpolation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 10663–10671, 2020. 2, 5, 6, 7

[5] Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable convolutional networks. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017. 2

[6] Tianyu Ding, Luming Liang, Zhihui Zhu, and Ilya Zharkov. Cdfi: Compression-driven network design for frame interpolation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8001–8011, 2021. 1

[7] John Flynn, Ivan Neulander, James Philbin, and Noah Snavely. Deepstereo: Learning to predict new views from the world's imagery. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5515–5524, 2016. 1

[8] Damien Fourure, Rémi Emonet, Elisa Fromont, Damien Muselet, Alain Tremeau, and Christian Wolf. Residual conv-deconv grid network for semantic segmentation. *arXiv preprint arXiv:1707.07958*, 2017. 2

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016. 2

[10] Zhewei Huang, Tianyuan Zhang, Wen Heng, Boxin Shi, and Shuchang Zhou. Rife: Real-time intermediate flow estimation for video frame interpolation. *arXiv preprint arXiv:2011.06294*, 2020. 3, 6, 7

[11] Eddy Ilg, Nikolaus Mayer, Tonmoy Saikia, Margret Keuper, Alexey Dosovitskiy, and Thomas Brox. Flownet 2.0: Evolution of optical flow estimation with deep networks. In *IEEE conference on computer vision and pattern recognition (CVPR)*, volume 2, page 6, 2017. 2

[12] 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 *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 1, 2, 5, 6

[13] Nima Khademi Kalantari, Ting-Chun Wang, and Ravi Ramamoorthi. Learning-based view synthesis for light field cameras. *ACM Transactions on Graphics (TOG)*, 35(6):1–10, 2016. 1

[14] Lingtong Kong, Boyuan Jiang, Donghao Luo, Wenqing Chu, Xiaoming Huang, Ying Tai, Chengjie Wang, and Jie Yang. Ifrnet: Intermediate feature refine network for efficient frame interpolation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1969–1978, 2022. 3, 6, 7

[15] Hyeongmin Lee, Taeoh Kim, Tae-young Chung, Daehyun Pak, Yuseok Ban, and Sangyoun Lee. Adacof: Adaptive collaboration of flows for video frame interpolation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5316–5325, 2020. 1, 5, 6, 7

[16] Ziwei Liu, Raymond A. Yeh, Xiaoou Tang, Yiming Liu, and Aseem Agarwala. Video frame synthesis using deep voxel flow. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017. 1, 2, 4, 5, 6

[17] Gucan Long, Laurent Kneip, Jose M Alvarez, Hongdong Li, Xiaohu Zhang, and Qifeng Yu. Learning image matching by simply watching video. In *European Conference on Computer Vision*, pages 434–450. Springer, 2016. 1, 2

[18] Liying Lu, Ruizheng Wu, Huaijia Lin, Jiangbo Lu, and Jiaya Jia. Video frame interpolation with transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3532–3542, 2022. 2

[19] Simon Niklaus and Feng Liu. Context-aware synthesis for video frame interpolation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 1, 2

[20] Simon Niklaus and Feng Liu. Softmax splatting for video frame interpolation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5437–5446, 2020. 1, 2, 5, 6

[21] Simon Niklaus, Long Mai, and Feng Liu. Video frame interpolation via adaptive convolution. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017. 1, 2

[22] Simon Niklaus, Long Mai, and Feng Liu. Video frame interpolation via adaptive separable convolution. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017. 1, 2, 5, 6

[23] Genady Paikin, Yotam Ater, Roy Shaul, and Evgeny Soloveichik. Efi-net: Video frame interpolation from fusion of events and frames. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1291–1301, 2021. 1

[24] Junheum Park, Keunsoo Ko, Chul Lee, and Chang-Su Kim. Bmbc: Bilateral motion estimation with bilateral cost volume for video interpolation. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XIV 16*, pages 109–125. Springer, 2020. 1, 2, 3

[25] Junheum Park, Chul Lee, and Chang-Su Kim. Asymmetric bilateral motion estimation for video frame interpolation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14539–14548, 2021. 1, 2, 3, 6, 7

[26] Jordi Pont-Tuset, Federico Perazzi, Sergi Caelles, Pablo Arbeláez, Alexander Sorkine-Hornung, and Luc Van Gool.

The 2017 davis challenge on video object segmentation. *arXiv:1704.00675*, 2017. 2, 5, 7

[27] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015. 2

[28] Zhihao Shi, Xiangyu Xu, Xiaohong Liu, Jun Chen, and Ming-Hsuan Yang. Video frame interpolation transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17482–17491, 2022. 2, 5, 6

[29] Li Siyao, Shiyu Zhao, Weijiang Yu, Wenxiu Sun, Dimitris Metaxas, Chen Change Loy, and Ziwei Liu. Deep animation video interpolation in the wild. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6587–6595, 2021. 1, 3

[30] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*, 2012. 2, 5

[31] Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz. Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 2

[32] Zachary Teed and Jia Deng. Raft: Recurrent all-pairs field transforms for optical flow. In *European conference on computer vision*, pages 402–419. Springer, 2020. 2

[33] Zhou Wang, Alan C Bovik, Hamid R Sheikh, Eero P Simoncelli, et al. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004. 5

[34] Chao-Yuan Wu, Nayan Singhal, and Philipp Krahenbuhl. Video compression through image interpolation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 416–431, 2018. 1

[35] Xiangyu Xu, Li Siyao, Wenxiu Sun, Qian Yin, and Ming-Hsuan Yang. Quadratic video interpolation. *Advances in Neural Information Processing Systems*, 32:1647–1656, 2019. 3

[36] Tianfan Xue, Baian Chen, Jiajun Wu, Donglai Wei, and William T Freeman. Video enhancement with task-oriented flow. *International Journal of Computer Vision*, 127(8):1106–1125, 2019. 2, 3, 5

[37] Zhiyang Yu, Yu Zhang, Deyuan Liu, Dongqing Zou, Xijun Chen, Yebin Liu, and Jimmy S. Ren. Training weakly supervised video frame interpolation with events. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 14589–14598, October 2021. 1

[38] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 586–595, 2018. 5

[39] Tinghui Zhou, Shubham Tulsiani, Weilun Sun, Jitendra Malik, and Alexei A Efros. View synthesis by appearance flow. In *European conference on computer vision*, pages 286–301. Springer, 2016. 1