Motion-blurred Video Interpolation and Extrapolation

Dawit Mureja Argaw, Junsik Kim, Francois Rameau, In So Kweon
KAIST Robotics and Computer Vision Lab., Daejeon, Korea
dawitmureja@kaist.ac.kr, {mibastro, rameau.fr}@gmail.com, iskweon77@kaist.ac.kr

Abstract

Abrupt motion of camera or objects in a scene result in a blurry video, and therefore recovering high quality video requires two types of enhancements: visual enhancement and temporal upsampling. A broad range of research attempted to recover clean frames from blurred image sequences or temporally upsample frames by interpolation, yet there are very limited studies handling both problems jointly. In this work, we present a novel framework for deblurring, interpolating and extrapolating sharp frames from a motion-blurred video in an end-to-end manner. We design our framework by first learning the pixel-level motion that caused the blur from the given inputs via optical flow estimation and then predict multiple clean frames by warping the decoded features with the estimated flows. To ensure temporal coherence across predicted frames and address potential temporal ambiguity, we propose a simple, yet effective flow-based rule. The effectiveness and favorability of our approach are highlighted through extensive qualitative and quantitative evaluations on motion-blurred datasets from high speed videos.

Introduction

Video frame interpolation aims at predicting one or more intermediate frames from given input frames for high frame-rate conversion. Existing frame interpolation approaches can be broadly categorized into flow-based (Mahajan et al. 2009; Zitnick et al. 2004; Liu et al. 2017), kernel-based (Niklaus, Mai, and Liu 2017b,a; Lee et al. 2020) and a fusion of the two (Bao et al. 2019b,a). Intermediate frames are interpolated either by directly warping the input frames with estimated optical flows (motion kernels) or using a trainable frame synthesis network. Extending these approaches for motion-blurred videos, however, is not a trivial process. Blurry video is a result of abrupt motions and long exposure time. As a result, contents in the video are degraded by motion blur and the gap between frames is relatively large compared to normal videos. This makes the computation of optical flow or motion kernel very challenging resulting in a subpar network performance (see Table 1).

There have been limited studies on joint deblurring and interpolation of a motion-blurred video. A naïve approach to the task at hand would be to cascade deblurring and interpolation methods interchangeably. With the recent progress in motion deblurring, several deep network based single image (Nah, Kim, and Lee 2017; Zhang et al. 2018; Tao et al. 2018) and video (Hyun Kim and Mu Lee 2015; Su et al. 2017; Nah, Son, and Lee 2019) deblurring approaches have been proposed. Given a blurry video, deploying deblurring frameworks followed by interpolation methods to predict sharp intermediate frames is not optimal since deblurring artifacts would propagate across the interpolated frames. Similarly, interpolation followed by deblurring would result in the propagation of interpolation artifacts caused by imprecise optical flow (motion kernel) predictions.

Recent video restoration works (Jin, Meishvili, and Favaro 2018; Purohit, Shah, and Rajagopalan 2019) attempted to extract multiple clean frames from a single motion-blurred image. Applying these works for blurry video interpolation (and extrapolation) by successively feeding blurry inputs, however, is problematic due to temporal ambiguity. A closely related work by Jin *et al.* (Jin, Hu, and Favaro 2019) jointly optimized deblurring and interpolation networks to predict clean frames from four blurry inputs. A concurrent work by Shen *et al.* (Shen et al. 2020) proposed an interpolation module that outputs a single sharp frame from two blurry inputs. More frames are generated by applying the interpolation module on the predicted sharp frames in a recurrent manner.

In this work, we propose a novel framework to *interpolate* and *extrapolate* multiple sharp frames from two blurry inputs. Inspired by the fact that motion-blurred image is a temporal aggregation of several latent frames during the exposure time of a camera, we exploit the input blurs as motion cues that can be leveraged to better infer and account for inter-frame motion. This is achieved by decoding latent frame features via optical flow estimation. We also design a flow-based rule to address temporal ambiguity and predict frames by warping the decoded features with the estimated flows. Unlike previous works (Jin, Hu, and Favaro 2019; Shen et al. 2020) that implicitly follow a deblurring $\rightarrow$ interpolation pipeline to predict intermediate frames between the deblurred middle latent frames, we adopt a motion-based approach to interpolate and extrapolate the entire latent frame sequence directly from given inputs in a temporally coherent manner.

We evaluated the proposed approach qualitatively and quantitatively on real image blur datasets generated from high speed videos (Nah, Kim, and Lee 2017; Jin, Hu, and Favaro 2019). We also comprehensively analyzed our work in connection to various related approaches on motion-blurred video interpolation, extrapolation and deblurring tasks to highlight the effectiveness and favourability of our approach. Moreover, we provide generalization experiments on real motion-blurred videos from (Nah et al. 2019; Su et al. 2017). In short, our contributions are: (1). We present a novel and intuitive framework for motion-blurred video interpolation and extrapolation (2). We propose a simple, yet effective, flow-based rule to address potential ambiguity and to restore latent frames in a temporally coherent manner (3). We extensively analyze our approach in relation to previous works and obtain a favourable performance. (4). We showcase the applicability of our model for related tasks such as video deblurring and optical flow estimation from motion-blurred inputs (5). We provide detailed ablation study on different network components to shed a light on the network design choice.

Methodology

Background. Motion-blurred image is a temporal average of multiple latent frames captured due to a sudden camera shake or dynamic motion of objects in a scene during the exposure time of a camera.

$$B_t = \int_t^{t+e} L(\tau) d\tau, \tag{1}$$

where $L(\tau)$ is a latent frame at time τ, e is the exposure time and B is the resulting motion-blurred image. As manually capturing a large blur dataset is a daunting task, a common practice in computer vision research is to synthesize a motion-blurred image by averaging consecutive frames in a high speed video (Kupyn et al. 2018; Nah, Kim, and Lee 2017; Tao et al. 2018; Su et al. 2017; Jin, Meishvili, and Favaro 2018; Jin, Hu, and Favaro 2019; Shen et al. 2020).

$$B_t = \frac{1}{N} \sum_{i=t-N/2}^{t+N/2} I_i, \tag{2}$$

where N is the number of frames to average and I_i is a clean frame in high speed video at time index i.

Given a blurred input B_t, image and video deblurring approaches (Kupyn et al. 2018; Nah, Kim, and Lee 2017; Tao et al. 2018; Su et al. 2017) recover the middle latent frame I_t. Recent works (Jin, Meishvili, and Favaro 2018; Purohit, Shah, and Rajagopalan 2019) attempted to restore the entire latent frame sequence from a single motion-blurred input, *i.e.* $\{I_{t-N/2}, \ldots, I_{t+N/2}\}$. However, these works suffer from a highly ill-posed problem known as *temporal ambiguity*. Without the help of external sensors such as IMU or other clues on the camera motion, it is not possible to predict the correct temporal direction from a single motion-blurred input as averaging does not preserve temporal order, *i.e.* both backward and forward averaging of sequential latent frames result in the same blurred image. Hence, deploying such methods (Jin, Meishvili, and Favaro 2018; Purohit, Shah, and Rajagopalan 2019) for motion-blurred video interpolation by successively feeding blurry frames is problematic as temporal coherence in the interpolated frames cannot be guaranteed.

Given two (or more) blurry frames, motion-blurred video interpolation aims at predicting sharp intermediate frames.

$$B_{t_0} = \frac{1}{N} \sum_{i=t_0-N/2}^{t_0+N/2} I_i \quad \ldots \quad B_{t_n} = \frac{1}{N} \sum_{i=t_n-N/2}^{t_n+N/2} I_i \tag{3}$$

, where $t_n \leq t_{n-1} + N$. Recent work by Jin *et al.* (Jin, Hu, and Favaro 2019) attempted to extract clean frames from four blurry inputs $\{B_{t_0}, \ldots, B_{t_4}\}$ by first recovering the corresponding middle latent frames, *i.e.* $\{I_{t_0}, \ldots, I_{t_4}\}$ using a deblurring network and then generating more intermediate frames between the recovered latent frames using an interpolation network. Compared to a naive approach of cascading deblurring and interpolation frameworks, their method is optimized in an end-to-end manner. A concurrent work by Shen *et al.* (Shen et al. 2020) proposed a pyramidal recurrent framework without explicit deblurring followed by interpolation. Given two blurry frames B_{t_0} and B_{t_1}, their approach directly outputs a sharp intermediate frame $I_{t_0+\Delta}$, where $\Delta \approx (t_1 - t_0)/2$. They addressed joint blur reduction and frame rate up-conversion by consecutively inputting blurry frame pairs and recursively applying the same procedure on the predicted sharp frames.

Problem formulation. In this work, we tackle the problem of interpolating and extrapolating multiple clean frames from blurry inputs in a single stage. Given two motion-blurred inputs B_{t_0} and B_{t_1}, we aim to recover all latent frames, *i.e.* $\{I_{t_0-N/2}, \ldots, I_{t_o+N/2}, I_{t_1-N/2}, \ldots, I_{t_1+N/2}\}$. For brevity, we refer to the middle latent frames (I_{t_0} and I_{t_1}) as *reference* frames. We propose a novel method to interpolate the intermediate latent frames between the reference frames ($\{I_{t_0}, \ldots, I_{t_1}\}$) and to extrapolate the past ($\{I_{t_0-N/2}, \ldots, I_{t_0-1}\}$) and future ($\{I_{t_1+1}, \ldots, I_{t_1+N/2}\}$) latent frames in one forward pass. We design our algorithm as follows. First, we encode and decode latent features by learning the pixel-level motion that occurred between the latent frames via optical-flow estimation. Second, we establish a simple, yet effective, flow-based rule to address potential temporal ambiguity. Third, we interpolate multiple clean frames by warping the decoded reference features with the estimated optical flows (see Fig. 1).

Our work is different from previous works (Jin, Hu, and Favaro 2019; Shen et al. 2020) in the following aspects: 1. Our approach interpolates multiple clean frames directly from two blurry inputs in a single stage while previous works recursively apply an interpolation module on the predicted clean frames. 2. We adopt a motion-based approach to interpolate intermediate latent frames rather than predicting frames in a generic manner, thereby showing that our approach is relatively robust in handling large motions. 3. Previous works only focus on interpolation *i.e.* deblurring the reference latent frames and interpolating intermediate

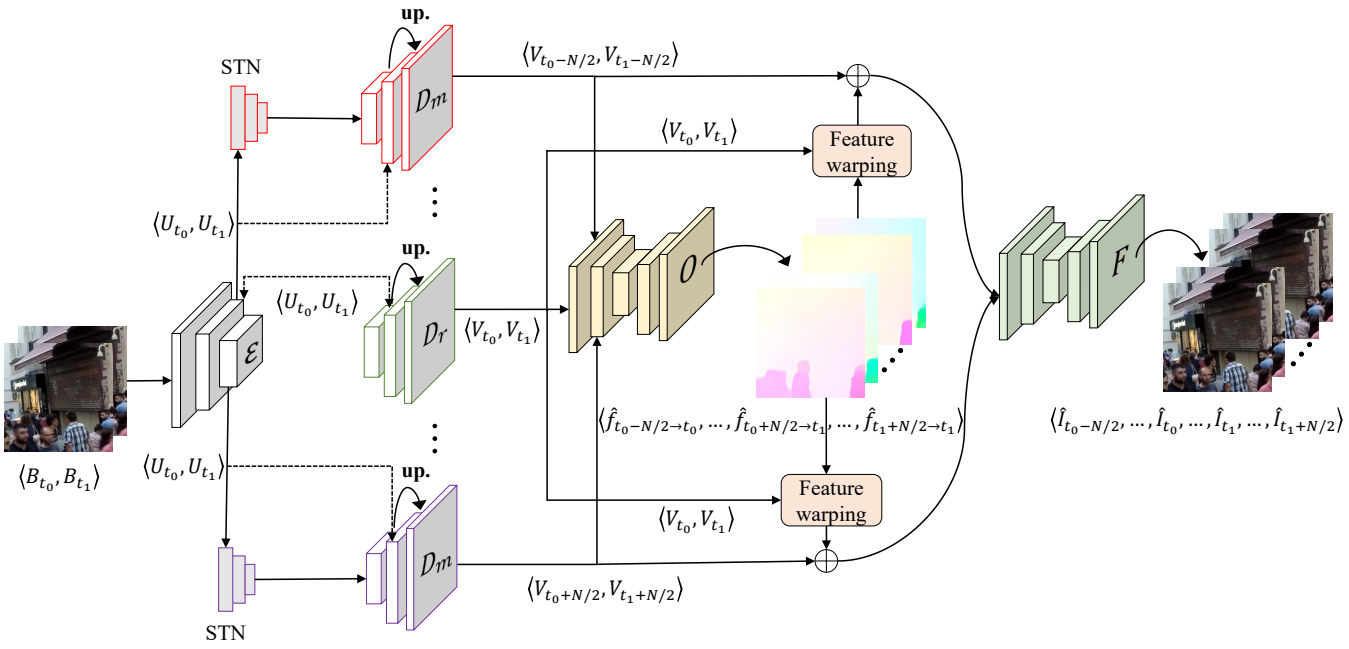

Figure 1: Overview of the proposed framework. First, we encode features from the given blurry inputs. Then, we decode latent frame features from the encoded features using global and local motion decoders that are supervised via optical flow estimation. Finally, we reconstruct multiple sharp frames in a bottom-up manner by warping the decoded features with the estimated flows.

frames between them. They ignore the other latent frames in order not to deal with temporal ambiguity, and hence, their work can not be extended for extrapolation task. By contrast, we interpolate and extrapolate latent frames in a temporally coherent manner by addressing temporal ambiguity with the proposed motion-based approach.

Proposed Approach

Feature encoding and decoding. Given two blurry inputs B_{t_0} and B_{t_1}, an encoder network $\mathcal{E}$ is used to extract feature representations of each input at different levels of abstractions (Eq. (4)). The encoder $\mathcal{E}$ is a feed-forward CNN with five convolutional blocks each with two layers of convolutions of kernel size 3×3 and stride size of 2 and 1, respectively.

$$\{U^l_{t_0}\}^K_{l=1} = \mathcal{E}(B_{t_0}) \quad \{U^l_{t_1}\}^K_{l=1} = \mathcal{E}(B_{t_1}) \tag{4}$$

, where $U^l_{t_0}$ is an encoded feature of B_{t_0} at level l and K (fixed to 6 in our experiments) is the number of levels (scales) in the feature pyramid.

The encoded features are then decoded into latent frame features as shown in Fig. 1. Reference (middle) features are directly decoded by successively upsampling the encoded features using layers of transposed convolution of kernel size 4×4 and a stride size of 2. A reference latent feature decoder $\mathcal{D}_r$ inputs the $\times 2$ upsampled decoded reference feature from level $l+1$ and the corresponding encoded feature concatenated channel-wise as shown in Eq. (5).

$$\{V^l_{t_0}, V^l_{t_1}\} = \mathcal{D}^l_r\big(\textbf{up.}\{V^{l+1}_{t_0}, V^{l+1}_{t_1}\} \oplus \{U^l_{t_0}, U^l_{t_1}\}\big) \tag{5}$$

where **up.** stands for upsampling, $\oplus$ is for channel-wise concatenation, V_{t_0} denotes the decoded reference latent feature.

The other (non-middle) features are decoded by inferring the blur motion from the encoded features. In order to learn the global motion of the other latent frames with respect to the reference latent frame, we used spatial transformer networks (STNs) (Jaderberg et al. 2015). Given an encoded feature, STN estimates an affine transformation parameter $\theta_{[R|T]}$ to spatially transform the input feature. As STN is limited to capturing only global motion, in order to compensate for the apparent local motions, we further refine the transformed feature using a motion decoder. A motion decoder $\mathcal{D}_m$ inputs the globally transformed feature along with the encoded feature (via skip connection shown in Fig. 1 in dotted lines) and the $\times 2$ upscaled non-middle latent feature from level $l+1$, and outputs a decoded a non-middle latent feature at level l as follows,

$$V^l_{s_0} = \mathcal{D}^l_m\big(\mathrm{STN}^l_{s_0}\{U^l_{t_0}\} \oplus \textbf{up.}\{V^{l+1}_{s_0}\} \oplus U^l_{t_0}\big) \tag{6}$$

$$V^l_{s_1} = \mathcal{D}^l_m\big(\mathrm{STN}^l_{s_1}\{U^l_{t_1}\} \oplus \textbf{up.}\{V^{l+1}_{s_1}\} \oplus U^l_{t_1}\big) \tag{7}$$

, where $s_0 = \{t_0 - N/2, \ldots, t_0 - 1, t_0 + 1, \ldots, t_0 + N/2\}$ and $s_1 = \{t_1 - N/2, \ldots, t_1 - 1, t_1 + 1, \ldots, t_1 + N/2\}$.

Optical flow estimation. Our network learns to decode latent frame features from the blurry inputs via optical flow estimation, *i.e.* the optical flow between the latent frames is computed using the respective decoded features. For instance, to estimate the optical flow between $I_{t_0 - N/2}$ and I_{t_0}, the corresponding decoded features $\{V_{t_0 - N/2}\}^K_{l=1}$ and $\{V_{t_0}\}^K_{l=1}$ are used. The two sets of decoded features here are equivalent to the encoded features of two clean input images in standard optical flow estimation algorithms. We estimate flow in a coarse-to-fine manner mimicking the vanilla

pipeline for optical flow estimation from two images (Sun et al. 2018; Fischer et al. 2015; Hui, Tang, and Change Loy 2018; Ranjan and Black 2017; Ilg et al. 2017). Given two decoded features at feature level l (*e.g.* $V^l_{t_0-N/2}$ and $V^l_{t_0}$), a warping layer $\mathcal{W}$ is used to back-warp the second feature $V^l_{t_0}$ (to the first feature $V^l_{t_0-N/2}$) with $\times 2$ upsampled flow from level $l+1$ as shown in Eq. (8). A correlation layer $\mathcal{C}$ (Fischer et al. 2015; Sun et al. 2018) is then used to compute the matching cost (cost volume) between the first feature $V^l_{t_0-N/2}$ and the back-warped second feature $\hat{V}^l_{t_0}$. The optical flow $\hat{f}^l$ is estimated using an optical flow estimator network $\mathcal{O}$ that inputs the cost volume, the first feature and the upsampled optical flow concatenated channel-wise and outputs a flow (Eq. (9)). Following (Sun et al. 2018), we use a context network to refine the estimated full-scale flow.

$$\hat{V}^l_{t_0} = \mathcal{W}\big(V^l_{t_0-N/2}, \mathbf{up}.\{\hat{f}^{l+1}\}\big) \tag{8}$$

$$\hat{f}^l = \mathcal{O}\big(\mathcal{C}\{V^l_{t_0-N/2}, \hat{V}^l_{t_0}\} \oplus V^l_{t_0-N/2} \oplus \mathbf{up}.\{\hat{f}^{l+1}\}\big) \tag{9}$$

In the same manner, we predict multiple optical flows between latent frames (see Fig. 2). Since the ground truth optical flow is not available to train the flow estimator, we used a pretrained FlowNet 2 (Ilg et al. 2017) network to obtain pseudo-ground truth flows (between sharp latent frames) to supervise the optical flow estimation between the decoded features. The flow supervision via imperfect ground truth flows is further enhanced by the frame supervision as our network is trained in an end-to-end manner.

Temporal ordering and ambiguity. Estimating optical flow between decoded features is crucial for maintaining temporal coherence across the predicted frames. We estimate optical flow (shown in red in Fig. 2) between the reference latent frame and non-middle latent frames within each blurry input, *i.e.* $\{\hat{f}_{t_0-N/2\to t_0}, \ldots, \hat{f}_{t_0+N/2\to t_0}\}$ and $\{\hat{f}_{t_1-N/2\to t_1}, \ldots, \hat{f}_{t_1+N/2\to t_1}\}$. Constraining these flows enforces our model to learn motions in a symmetric manner with STNs and motion decoders close to the reference features decoding smaller motions, and those further from the reference features decoding larger motions. This in turn preserves temporal ordering within the decoded features of each blurry input avoiding random shuffling. However, correct temporal direction can not be still guaranteed as features can be decoded in a reverse order. To address this potential temporal ambiguity, we propose a simple, yet effective flow-based rule. We predict optical flow (shown in green in Fig. 2) between the non-middle latent frames of the first input and the reference latent frame of the second input and vice versa, *i.e.* $\{\hat{f}_{t_0-N/2\to t_1}, \ldots, \hat{f}_{t_0+N/2\to t_1}\}$ and $\{\hat{f}_{t_1-N/2\to t_0}, \ldots, \hat{f}_{t_1+N/2\to t_0}\}$. By constraining these flows via endpoint error supervision, we establish the following rules:

Rule 1. if $\|\hat{f}_{t_0-N/2\to t_1}\| > \|\hat{f}_{t_0+N/2\to t_1}\|$, it means that the features of B_{t_0} are decoded in the correct order *i.e.* $\{V_{t_0-N/2}, \ldots, V_{t_0}, \ldots, V_{t_0+N/2}\}$.

Rule 2. if $\|\hat{f}_{t_0-N/2\to t_1}\| < \|\hat{f}_{t_0+N/2\to t_1}\|$, it means that the features of B_{t_0} are decoded in a reverse order *i.e.* $\{V_{t_0+N/2}, \ldots, V_{t_0}, \ldots, V_{t_0-N/2}\}$ and hence, should be reversed to the correct order.

, where $\|\cdot\|$ denotes the magnitude of the flow. In a similar manner, we can use the optical flows $\hat{f}_{t_1-N/2\to t_0}$ and $\hat{f}_{t_1+N/2\to t_0}$ to ensure that the features of B_{t_1} are decoded in the correct order. These rules need to be applied only on the four flows between the latent decoded features on the extrema $(V_{t_0-N/2}, V_{t_0+N/2}, V_{t_1-N/2}, V_{t_1+N/2})$ and the reference decoded features, since temporal ordering within each input is maintained. Hence, the proposed flow-based rule can be used to interpolate and extrapolate larger number of frames with no additional computational cost.

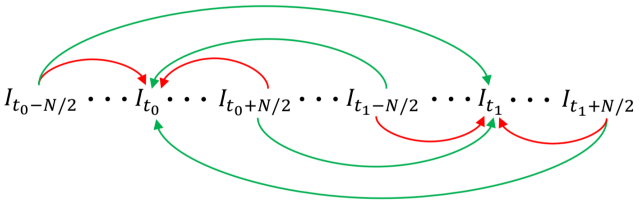

Figure 2: Optical flow estimation between latent frames

Frame synthesis. The decoded features and the estimated optical flows are then used to interpolate and extrapolate sharp frames from the blurry inputs. The reference latent frames are predicted at different spatial scales directly from the decoded reference features using a frame synthesis network $\mathcal{F}_r$ (Eq. (10)). This is equivalent to deblurring each input frame except for the fact that we output deblurred middle frames at different scales. The other (non-middle) latent frames are predicted by back-warping the decoded reference features with the corresponding optical flows. For better reconstruction of occluded regions, we also use the corresponding non-middle decoded feature along with the warped features during frame synthesis as shown in Eq. (11). Similarly to the optical flow estimation stage, frames are synthesized in a bottom-up manner from the smallest to the full-scale resolution.

$$\{\hat{I}^l_{t_0}, \hat{I}^l_{t_1}\} = \mathcal{F}^l_r\big(\{V^l_{t_0}, V^l_{t_1}\} \oplus \{\hat{I}^{l+1}_{t_0}, \hat{I}^{l+1}_{t_1}\}\big) \tag{10}$$

$$\hat{I}^l_s = \mathcal{F}^l_m\big(\mathcal{W}\{V^l_{t_0}, \hat{f}^l_{s\to t_0}\} \oplus \mathcal{W}\{V^l_{t_1}, \hat{f}^l_{s\to t_1}\} \oplus V^l_s \oplus \hat{I}^{l+1}_s\big) \tag{11}$$

, where $s = \{t_0 - N/2, \ldots, t_0 - 1, t_0 + 1, \ldots, t_1 - 1, t_1 + 1, \ldots, t_1 + N/2\}$, $\mathcal{W}$ denotes a warping layer and $\mathcal{F}_m$ is a frame synthesis network for non-middle latent frames.

The proposed approach incorporates decoded features from both blurry inputs when estimating optical flows and predicting frames. This allows the frame synthesis network to exploit temporal and contextual information across inputs when interpolating and extrapolating latent frames. For instance, if the two consecutive inputs are substantially different in terms of blur sizes, our model leverages the less blurred input when predicting latent frames from the heavily blurred input, and hence, outputs a temporally smooth video with consistent visual quality.

Network training. We train our network in an end-to-end manner by optimizing the estimated intermediate flows and predicted latent frames. For sharp frame reconstruction, we computed the $\ell 1$ photometric loss between the predicted and ground truth frames. As our network predicts images at different scales, we used bilinear interpolation to downsample the ground truth frames to respective sizes. The weighted multi-scale photometric loss for reconstructing N frames from two blurry frames is written as follows,

$$\mathcal{L}_{frame} = \sum_{n=1}^{N} \sum_{l=1}^{K} w^l \cdot \left| I_n^l - \hat{I}_n^l \right|_1 \tag{12}$$

, where w^l is the frame loss weight coefficient at feature level l, n is an index for the reconstructed frame sequence.

For optical flow training, we use endpoint error between the predicted flows and pseudo-ground truth flows. As mentioned earlier, we used pretrained FlowNet 2 (Ilg et al. 2017) to compute the flows between the corresponding ground truth frames (from which the motion-blurred inputs are averaged) to guide the optical flow estimator. We predict a total of $2N - 4$ optical flows when interpolating N frames, and the weighted multi-scale endpoint error for supervising the estimated flows is computed as follows,

$$\mathcal{L}_{flow} = \sum_{m=1}^{2N-4} \sum_{l=1}^{K} \hat{w}^l \cdot \left| f_m^l - \hat{f}_m^l \right|_2 \tag{13}$$

, where $\hat{w}^l$ is a flow loss weight coefficient and m is an index for estimated flows. The total training loss for interpolating and extrapolating N number of sharp frames from two blurry input is given as a weighted sum of the two losses as shown below.

$$\mathcal{L} = \alpha_1 \mathcal{L}_{frame} + \alpha_2 \mathcal{L}_{flow} \tag{14}$$

Experiment

Dataset. To train our network for the task at hand, we take advantage of two publicly available high speed video datasets to generate motion-blurred images. The GoPro high speed video dataset (Nah, Kim, and Lee 2017), a benchmark for dynamic scene deblurring, provides 33 720P videos taken at 240*fps*. We used 22 videos for training and generated motion-blurred images by averaging 7 consecutive frames. We also used the recently proposed Sony RX V high-frame rate video dataset (Jin, Hu, and Favaro 2019) which provides more than 60 1080P videos captured at 250*fps*. We used 40 videos during training and generated motion-blurred images by averaging 7 consecutive frames. To qualitatively and quantitatively analyze our approach on a diverse set of motion blurs, we choose 8 videos from each dataset (nonoverlapping with the training set) according to different blur sizes (small and large), blur types (static or dynamic) and complexity of the motion involved in the blurry video. We also provide generalization experiments on real motion-blurred videos from (Su et al. 2017; Nah et al. 2019).

Implementation details. We implemented and trained our model in PyTorch (Paszke et al. 2019). We used Adam (Kingma and Ba 2015) optimizer with parameters β_1, β_2 and *weight decay* fixed to 0.9, 0.999 and $4e-4$, respectively. We trained our network using a mini-batch size of 4 image pairs by randomly cropping image patch sizes of 256×256. The pseudo-ground truth optical flows for supervising the predicted flows are computed on-the-fly during training. The loss weight coefficients are fixed to $w_6 = 0.32$, $w_5 = 0.08$, $w_4 = 0.04$, $w_3 = 0.02$, $w_2 = 0.01$ and $w_1 = 0.005$ from the lowest to the highest resolution, respectively, for both frames and flows. We trained our model for 120 epochs with initial learning rate fixed to $\lambda = 1e - 4$ and gradually decayed by half at 60, 80 and 100 epochs. For the first 15 epochs, we only trained the optical flow estimator by setting $\alpha_1 = 0$ and $\alpha_2 = 1$ to facilitate feature decoding and flow estimation. For the rest of the epochs, we fixed $\alpha_1 = 1$ and $\alpha_2 = 1$. During inference, we interpolate and extrapolate frames by successively passing disjoint blurry frame pairs.

	GoPro		Sony RX V	
Method	PSNR	SSIM	PSNR	SSIM
SepConv (Niklaus *et al.*)	26.977	0.769	26.181	0.716
SloMo (Jiang *et al.*)	27.240	0.785	26.360	0.728
DAIN (Bao *et al.*)	27.220	0.783	26.410	0.731
Ours	**32.202**	**0.914**	**31.019**	**0.894**

Table 1: Comparison with standard interpolation methods

	GoPro		Sony RX V	
Method	PSNR	SSIM	PSNR	SSIM
DVD $\oplus$ DAIN	25.650	0.722	27.885	0.791
DAIN $\oplus$ DVD	28.885	0.843	28.157	0.797
DeepDeblur $\oplus$ DAIN	28.154	0.831	27.192	0.782
DAIN $\oplus$ DeepDeblur	28.176	0.829	27.195	0.778
SRN $\oplus$ DAIN	29.966	0.870	29.245	0.828
DAIN $\oplus$ SRN	30.045	0.867	29.074	0.822
Ours	**32.202**	**0.914**	**31.019**	**0.894**

Table 2: Comparison with cascaded approaches

Quantitative Analysis

In this section, we comprehensively analyze our work in connection to related works. Except for (Jin, Meishvili, and Favaro 2018) and our approach, other related methods fail to restore the first and the last few video frames. For fair evaluation purely based on the interpolated frames, we aligned the GT frames with the interpolated frames and discard the missing GT frames when evaluating such methods. We perform motion-blurred video interpolation ($\times 7$ slower video) and middle frame deblurring ($\times 1$ video) comparisons on peak signal-to-noise ratio (PSNR) and structural similarity index measure (SSIM) metrics.

Cascaded approaches. One possible way to interpolate clean frames from given blurry inputs is to cascade inter-

Figure 3: Qualitative analysis on interpolated frames. The 1^{st} column shows blurry inputs from GoPro test set. The 2^{nd} column depicts the outputs of cascaded approach (SRN (Tao et al. 2018) + DAIN (Bao et al. 2019a)). The 3^{rd} column shows the outputs of Jin-Seq (Jin, Meishvili, and Favaro 2018). The 4^{th} column shows frames interpolated by Jin-SloMo (Jin, Hu, and Favaro 2019) and the 5^{th} column depicts the outputs of our network.

polation and deblurring frameworks. To quantitatively analyze our method in comparison with such approaches, we experimented with state-of-the-art single image deblurring (DeepDeblur (Nah, Kim, and Lee 2017), SRN (Tao et al. 2018)) and video deblurring (DVD (Su et al. 2017)) works cascaded with state-of-the-art interpolation methods (DAIN (Bao et al. 2019a), SloMo (Jiang et al. 2018)). As can be inferred from Table 2, our method performed consistently better than cascaded approaches. For instance, our approach outperforms the strongest baseline (SRN $\oplus$ DAIN) by a margin of 2.00 dB on average. This performance gain is mainly because cascaded approaches are prone to error propagation while our method directly interpolates clean frames from blurry inputs by estimating the motion within and across inputs. The effect of propagation of deblurring and interpolation artifacts can also be noticed from Table 4. Our method shows an average performance decrease of 0.71 dB on the interpolated videos ($\times 7$) compared to deblurred videos ($\times 1$) while SRN $\oplus$ DAIN shows an average performance decrease of 2.50 dB.

Comparison with previous works. We compared our approach with works that restore sequence of latent frames from a single blurry input (Jin-Seq (Jin, Meishvili, and Favaro 2018)). Directly deploying such methods for motion-blurred video interpolation is not optimal since temporal ambiguity is a problem (see Table 3). To address this challenge, we applied our proposed flow-based rule during the inference stage by computing the necessary flows (between the restored frames) using pretrained FlowNet 2. This fix significantly improved performance by an average margin of 2.24 dB. While the sequence restoration can be achieved, contextual information between input frames is not exploited (as they are processed independently) leading to lower performances when compared to our approach. We also analyzed our model in comparison with the recently proposed approach by Jin *et al.* (Jin-SloMo (Jin, Hu, and Favaro 2019)). As can be inferred from Table 3, our method outperforms Jin-SloMo by a margin of 1.82 dB and 1.50 dB on average on interpolated and deblurred videos, respectively. This is mainly because our method is relatively robust to large blurs while Jin-SloMo is limited to small motions (small blurs) as frames are deblurred and interpolated without taking pixel-level motion into consideration (see Fig. 3).

Middle frame deblurring. Besides motion-blurred interpolation, we also analyzed the performance of our model for video deblurring *i.e.* we evaluated the predicted reference (middle) latent frames. As can be inferred from Table 4, our approach performs competitively against state-of-

Figure 4: Qualitative analysis on extrapolated frames. Previous works (Jin, Hu, and Favaro 2019; Shen et al. 2020) ignore the first few latent frames in order not to deal with temporal ambiguity. In comparison, our approach outputs the entire latent frame sequence.

	GoPro		Sony RX V	
Method	PSNR	SSIM	PSNR	SSIM
Jin-Seq (2018)	26.848	0.785	25.785	0.735
Jin-Seq + *flow fix*	29.761	0.877	27.348	0.779
Jin-SloMo (2019)	30.321	0.878	29.267	0.816
Ours	**32.202**	**0.914**	**31.019**	**0.894**

Table 3: Comparison with previous works

	GoPro		Sony RX V	
Method	PSNR	SSIM	PSNR	SSIM
DVD (Su *et al.*)	26.547	0.742	28.937	0.805
DeepDeblur (Nah *et al.*)	29.671	0.867	27.882	0.788
SRN (Tao *et al.*)	**33.382**	**0.931**	30.827	0.851
Jin-Seq (2018)	31.442	0.906	29.752	0.812
Jin-SloMo (2019)	31.318	0.900	30.325	0.829
Ours	32.994	0.927	**31.650**	**0.904**

Table 4: Middle frame deblurring

the-art deblurring approaches. The slight performance loss can be attributed to the fact that deblurring works in general are trained with larger blurs (by averaging large number of frames, *e.g.* DeepDeblur and SRN averaged 7-13 frames) while our work is trained on motion-blurred images generated by averaging 7 or 9 frames.

Qualitative Analysis

Interpolated frames. We qualitatively compared our approach with related works on the quality of the interpolated frames. As can be seen from Fig. 3, our approach is relatively robust to heavily blurred inputs and interpolates visually sharper images with clearer contents compared to other related methods (Tao et al. 2018; Bao et al. 2019a; Jin, Meishvili, and Favaro 2018; Jin, Hu, and Favaro 2019).

Extrapolated frames. Previous works (Jin, Hu, and Favaro 2019; Shen et al. 2020) implicitly follow a *deblurring* $\rightarrow$ *interpolation* pipeline, and hence can only interpolate frames between the reference latent frames. Our approach, on the other hand, not only interpolates intermediate frames but also extrapolates the latent frames underlying to the left and right side of the reference latent frames. As shown in Fig. 4, the 1^{st} frame interpolated by Jin-SloMo is aligned with the 11^{th} frame predicted by our approach. Their approach ignores the first 10 latent frames so as not to deal with potential temporal ambiguity. By contrast, our work reconstructs the entire latent frame sequence in a temporally coherent manner.

Estimated optical flows. We qualitatively analyzed the intermediate optical flows estimated by our approach in comparison with pseudo-ground truth (p-GT) flows predicted from the corresponding sharp latent frames using pretrained FlowNet 2. As can be inferred from Fig. 5, our network estimates accurate optical flows from decoded features of blurry inputs for different blur types involving dynamic motion of multiple objects in a close to static or moving scene. This further explains the quantitative performance of our approach for motion-blurred video interpolation in the previous section as estimating correct pixel-level motion is crucial for accurate frame interpolation.

Ablation Studies

Optical flow estimation. To examine the importance of optical flow estimation, we directly regressed latent frames from the decoded features (without estimating flow) using only the frame synthesis network *i.e.*

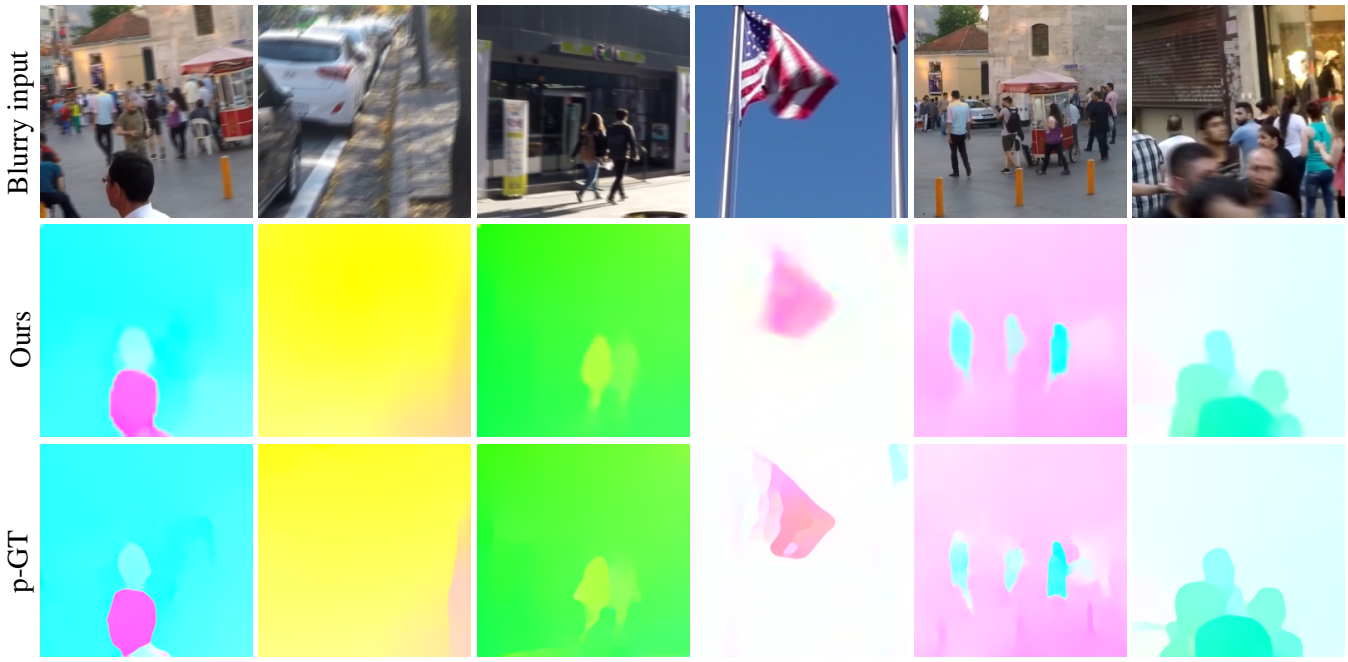

Figure 5: Qualitative analysis on estimated optical flows. The second row depicts optical flows estimated by our model from blurry inputs and the third row shows the corresponding p-GT flows from sharp latent frames.

$\{\hat{I}^l_{t_0-N/2}, \ldots, \hat{I}^l_{t_1+N/2}\} = \mathcal{F}^l\big(V^l_{t_0-N/2}, \ldots, V^l_{t_1+N/2}\big)$. The results on motion-blurred video interpolation ($\times 7$) are summarized in Table 5. We experimentally observed that $\mathcal{L}_{frame}$ is a strong enough constraint to guide the STNs and motion decoders to decode features of each blurry input in a temporally ordered manner (without random shuffling), yet, correct temporal direction can not be guaranteed. To ensure temporal coherence, we ordered the predicted frames using the proposed flow-based rule. This post-processing step improves performance by 0.78 dB. Even with the flow fix, however, directly predicting frames without motion estimation causes a performance decrease of 1.92 dB on average compared to our full model. This highlights that estimating optical flows is not only useful to address temporal ambiguity but also important to warp decoded features for sharp reconstruction of latent frames.

Feature decoding. Motion decoders ($\mathcal{D}_m$) decode non-middle latent features with respect to reference latent features by refining the STN transformed features (see Eq. (6) and Eq. (7)). Training our network without motion decoders, *i.e.* decoding features only via STN transformation, results in a subpar performance as shown Table 5. This is mainly because local motions that are apparent in the high speed videos can not be effectively captured only using STNs. In principle, both global and local motions can be implicitly learnt by guiding motion decoders (without the need to explicitly model global motions with STNs) via optical flow supervision as CNNs have been shown to be effective in motion estimation tasks. This is also empirically evident as a network trained with only motion decoders results in a good performance (see Table 5). However, incorporating STNs to learn global motions also proved to give a significant performance boost of 0.89 dB.

			GoPro		Sony RX V	
STN	$\mathcal{D}_m$	Flow	PSNR	SSIM	PSNR	SSIM
✓	✓	✗	29.509	0.836	28.316	0.805
✓	✓	*flow fix*	30.219	0.870	29.163	0.812
✓	✗	✓	28.789	0.855	27.467	0.798
✗	✓	✓	31.317	0.893	30.125	0.857
✓	✓	✓	**32.202**	**0.914**	**31.019**	**0.894**

Table 5: Ablation studies

Conclusion

In this work, we tackle the problem of multi-frame interpolation and extrapolation from a given motion-blurred video. We adopt a motion-based approach to predict frames in a temporally coherent manner without ambiguity. As a result, our method can interpolate, extrapolate and recover high quality frames in a single pass. Our method is extensively analyzed in comparison with existing approaches. We also experimented with the applicability of our approach on related tasks such as video deblurring and flow estimation.

Acknowledgements

This work was supported by NAVER LABS Corporation [SSIM: Semantic & scalable indoor mapping].

References

Bao, W.; Lai, W.-S.; Ma, C.; Zhang, X.; Gao, Z.; and Yang, M.-H. 2019a. Depth-Aware Video Frame Interpolation. In *IEEE Conference on Computer Vision and Pattern Recognition*.

Bao, W.; Lai, W.-S.; Zhang, X.; Gao, Z.; and Yang, M.-H. 2019b. MEMC-Net: Motion estimation and motion compensation driven neural network for video interpolation and enhancement. *IEEE transactions on pattern analysis and machine intelligence* .

Fischer, P.; Dosovitskiy, A.; Ilg, E.; Häusser, P.; Hazırbaş, C.; Golkov, V.; Van der Smagt, P.; Cremers, D.; and Brox, T. 2015. Flownet: Learning optical flow with convolutional networks. *arXiv preprint arXiv:1504.06852* .

Hui, T.-W.; Tang, X.; and Change Loy, C. 2018. Liteflownet: A lightweight convolutional neural network for optical flow estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 8981–8989.

Hyun Kim, T.; and Mu Lee, K. 2015. Generalized video deblurring for dynamic scenes. In *IEEE Conference on Computer Vision and Pattern Recognition*.

Ilg, E.; Mayer, N.; Saikia, T.; Keuper, M.; Dosovitskiy, A.; and Brox, T. 2017. FlowNet 2.0: Evolution of Optical Flow Estimation with Deep Networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. URL http://lmb.informatik.uni-freiburg.de//Publications/2017/IMKDB17.

Jaderberg, M.; Simonyan, K.; Zisserman, A.; et al. 2015. Spatial transformer networks. In *Conference on Neural Information Processing Systems*.

Jiang, H.; Sun, D.; Jampani, V.; Yang, M.; Learned-Miller, E. G.; and Kautz, J. 2018. Super SloMo: High Quality Estimation of Multiple Intermediate Frames for Video Interpolation. In *IEEE Conferene on Computer Vision and Pattern Recognition*.

Jin, M.; Hu, Z.; and Favaro, P. 2019. Learning to Extract Flawless Slow Motion from Blurry Videos. In *IEEE Conference on Computer Vision and Pattern Recognition*.

Jin, M.; Meishvili, G.; and Favaro, P. 2018. Learning to extract a video sequence from a single motion-blurred image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 6334–6342.

Kingma, D. P.; and Ba, J. 2015. Adam: A Method for Stochastic Optimization. In *International Conference on Learning Representations*.

Kupyn, O.; Budzan, V.; Mykhailych, M.; Mishkin, D.; and Matas, J. 2018. Deblurgan: Blind motion deblurring using conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 8183–8192.

Lee, H.; Kim, T.; Chung, T.-y.; Pak, D.; Ban, Y.; and Lee, S. 2020. AdaCoF: Adaptive Collaboration of Flows for Video Frame Interpolation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Liu, Z.; Yeh, R. A.; Tang, X.; Liu, Y.; and Agarwala, A. 2017. Video frame synthesis using deep voxel flow. In *Proceedings of the IEEE International Conference on Computer Vision*, 4463–4471.

Mahajan, D.; Huang, F.-C.; Matusik, W.; Ramamoorthi, R.; and Belhumeur, P. 2009. Moving gradients: a path-based method for plausible image interpolation. *ACM Transactions on Graphics (TOG)* 28(3): 1–11.

Nah, S.; Baik, S.; Hong, S.; Moon, G.; Son, S.; Timofte, R.; and Lee, K. M. 2019. NTIRE 2019 Challenge on Video Deblurring and Super-Resolution: Dataset and Study. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*.

Nah, S.; Kim, T. H.; and Lee, K. M. 2017. Deep Multi-Scale Convolutional Neural Network for Dynamic Scene Deblurring. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Nah, S.; Son, S.; and Lee, K. M. 2019. Recurrent neural networks with intra-frame iterations for video deblurring. In *IEEE Conference on Computer Vision and Pattern Recognition*.

Niklaus, S.; Mai, L.; and Liu, F. 2017a. Video frame interpolation via adaptive convolution. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 670–679.

Niklaus, S.; Mai, L.; and Liu, F. 2017b. Video frame interpolation via adaptive separable convolution. In *Proceedings of the IEEE International Conference on Computer Vision*, 261–270.

Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; et al. 2019. PyTorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, 8024–8035.

Purohit, K.; Shah, A.; and Rajagopalan, A. 2019. Bringing alive blurred moments. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 6830–6839.

Ranjan, A.; and Black, M. J. 2017. Optical flow estimation using a spatial pyramid network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 4161–4170.

Shen, W.; Bao, W.; Zhai, G.; Chen, L.; Min, X.; and Gao, Z. 2020. Blurry Video Frame Interpolation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 5114–5123.

Su, S.; Delbracio, M.; Wang, J.; Sapiro, G.; Heidrich, W.; and Wang, O. 2017. Deep Video Deblurring for Hand-held Cameras. In *IEEE Conference on Computer Vision and Pattern Recognition*.

Sun, D.; Yang, X.; Liu, M.-Y.; and Kautz, J. 2018. PWC-Net: CNNs for Optical Flow Using Pyramid, Warping, and Cost Volume. In *IEEE Conference on Computer Vision and Pattern Recognition*.

Tao, X.; Gao, H.; Shen, X.; Wang, J.; and Jia, J. 2018. Scale-recurrent Network for Deep Image Deblurring. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Zhang, J.; Pan, J.; Ren, J.; Song, Y.; Bao, L.; Lau, R. W.; and Yang, M.-H. 2018. Dynamic Scene Deblurring Using Spatially Variant Recurrent Neural Networks. In *IEEE Conference on Computer Vision and Pattern Recognition*.

Zitnick, C. L.; Kang, S. B.; Uyttendaele, M.; Winder, S.; and Szeliski, R. 2004. High-quality video view interpolation using a layered representation. *ACM transactions on graphics (TOG)* 23(3): 600–608.