Enhanced Bi-directional Motion Estimation for Video Frame Interpolation

Xin Jin[1] Longhai Wu[1] Guotao Shen[1] Youxin Chen[1] Jie Chen[1] Jayoon Koo[2] Cheul-hee Hahm[2]
[1]Samsung Electronics (China) R&D Center [2]Samsung Electronics, South Korea
{xin.jin,longhai.wu,guotao.shen,yx113.chen,ada.chen,j.goo,chhahm}@samsung.com

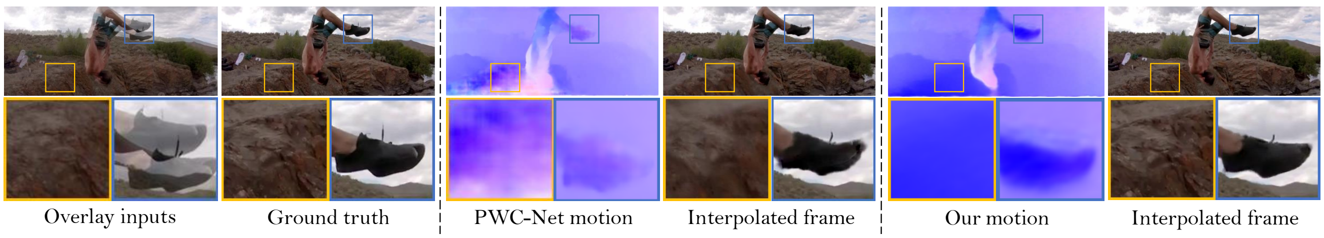


Figure 1. **First two columns:** Overlay inputs and ground truth frame. **Middle two columns:** Motion field (from first to second frame) by PWC-Net [32] and corresponding interpolation. PWC-Net is end-to-end trained with our frame synthesis network. **Last two columns:** Motion field and interpolated frame by our bi-directional motion estimator (15x smaller than PWC-Net) and synthesis network.

Abstract

*We propose a simple yet effective algorithm for motion-based video frame interpolation. Existing motion-based interpolation methods typically rely on an off-the-shelf optical flow model or a U-Net based pyramid network for motion estimation, which either suffer from large model size or limited capacity in handling various challenging motion cases. In this work, we present a novel compact model to simultaneously estimate the bi-directional motions between input frames. It is designed by carefully adapting the ingredients (*e.g.*, warping, correlation) in optical flow research for simultaneous bi-directional motion estimation within a flexible pyramid recurrent framework. Our motion estimator is extremely lightweight (15x smaller than PWC-Net), yet enables reliable handling of large and complex motion cases. Based on estimated bi-directional motions, we employ a synthesis network to fuse forward-warped representations and predict the intermediate frame. Our method achieves excellent performance on a broad range of frame interpolation benchmarks. Code and trained models are available at* `https://github.com/srcn-ivl/EBME`.

1. Introduction

Video frame interpolation aims to increase the frame rate of videos, by synthesizing non-existent intermediate frames between original successive frames. Increasing frame rate is beneficial for human perception [13], and has wide applications in novel view synthesis [7], video compression [19], adaptive streaming [34], *etc*.

The key challenge for frame interpolation is the possible complex, large motions between input frames and intermediate frame. Based on whether a motion model is employed to capture the per-pixel motion (*i.e*., optical flow) between frames, existing methods can be classified into two categories: motion-agnostic methods [25, 22, 5, 6], and motion-based methods [12, 17, 23, 3, 24, 26, 27, 20]. With recent advances in optical flow [11, 9, 32, 33], motion-based interpolation has developed into a promising framework.

Motion-based interpolation involves two steps: (i) motion estimation, and (ii) frame synthesis. Motion field is estimated to guide the synthesis of intermediate frame, by forward-warping [23, 24] or backward-warping [12, 27, 30] input frames towards intermediate frame. Forward-warping is guided by motion from input frames to intermediate frame, while backward-warping requires motion in reversed direction. In particular, when the bi-directional motions between input frames have been estimated, the motions from input frames to *arbitrary* intermediate frame required by forward-warping, can be easily approximated by linearly scaling the motion magnitude [23, 24].

Bi-directional motion estimation is a crucial step for most motion-based interpolation methods [23, 24, 12, 2, 30]. Many of existing methods [23, 2, 24] employ an off-the-shelf optical flow model (*e.g*., PWC-Net [32]) for bi-directional motions, which however suffer from large model size, need to run the model twice, and can hardly handle extreme large motion beyond the training data. Recently, a BiOF-I module [30] is proposed for simultane-

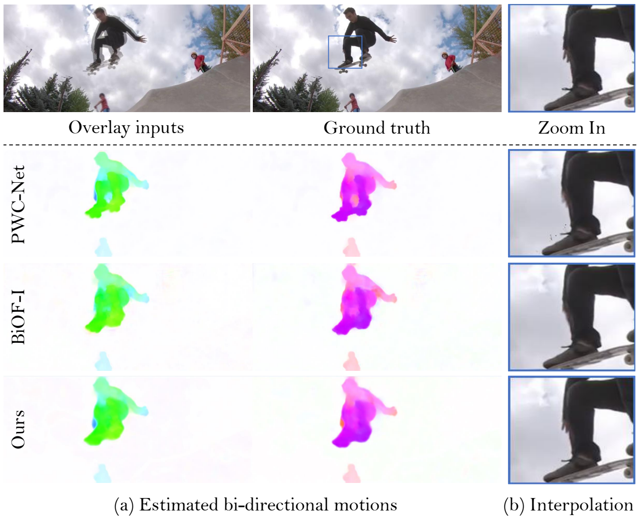


Figure 2. Visual comparisons between PWC-Net [32], BiOF-I [30], and our motion estimator, when combined with our synthesis network for frame interpolation. BiOF-I fails to capture the motion of fingers, due to the lack of correlation volume.

ous bi-directional motion estimation. It is based on a flexible pyramid recurrent structure, which enables customizable pyramid levels in testing to handle large motions. At each pyramid level, BiOF-I uses current motion estimate to backward-warp the features of both input frames towards each other, and employs a shared plain U-Net to refine current motion. However, U-Net is over-simplified for optical flow, due to the lack of correlation volume, which is a vital ingredient in modern optical flow models [32, 33].

In this work, we present a simple but effective algorithm for frame interpolation. Our main contribution is a novel bi-directional motion estimator. Cast in a flexible pyramid recurrent framework, we adapt the ingredients (*e.g.*, warping, correlation) in optical flow research to simultaneously estimate the bi-directional motions between input frames. In particular, at each pyramid level, we forward-warp both input frames towards a hidden middle frame. This middle-oriented forward-warping improves robustness against large motion, and allows us to construct a single correlation volume for simultaneous bi-directional motion estimation. Based estimated bi-directional motions, we forward-warp input frames and their context features to intermediate frame, and employ a synthesis network to predict the intermediate frame from warped representations.

Our bi-directional motion estimator enables better interpolation performance than its single-directional counterpart which needs to run twice. It is 15x smaller than PWC-Net [32], yet can better handle large motion cases and produce better interpolation result (see Figure 1). Compared to BiOF-I [30], our motion estimator can capture the motion of fast-moving small objects, giving better interpolation for local details (see Figure 2).

We conduct extensive experiments to verify the effectiveness of our interpolation method named EBME – **E**nhanced **B**i-directional **M**otion **E**stimation for frame interpolation. Despite its small model size, EBME performs favorably against state-of-the-art methods on a broad range of benchmarks, from low resolution UCF101 [31], Vimeo90K [35], to moderate-resolution SNU-FILM [6] and extremely high-resolution 4K1000FPS [30].

2. Related Work

Optical flow and correlation volume. Optical flow is a low-level vision task that aims to estimate the per-pixel motion between successive frames. Modern optical flow models [32, 10, 33] follow similar design philosophy: extract CNN features for both input frames, construct correlation volume with CNN features, and update the flow field upon a pyramid structure [32] or at fixed high resolution [33].

Correlation volume, which stores the matching scores between the pixels of two frames, is a discriminative representation for optical flow. Before constructing correlation volume, backward-warping is typically employed to align the second frame to the first frame to compensate for estimated motion. With the warping operation (and downsampled features), a partial correlation volume with limited matching range is sufficient for optical flow estimation [32].

Off-the-shelf flow models for frame interpolation. PWC-Net [32] and RAFT [33] are two representative modern optical flow models. In particular, PWC-Net has been widely adopted in frame interpolation to estimate the bi-directional motions by running twice [2, 23, 24]. PWC-Net builds a 6-level feature pyramids to handle large motion. At each level, it uses current motion estimate to backward-warp the feature of second frame to the first frame, constructs a correlation volume with warped feature and the feature of first frame, and then infers a refined motion from correlation-injected representation.

Off-the-shelf optical flow models have two disadvantages when applied for frame interpolation. First, they typically have a large number of parameters. Second, when end-to-end trained with a synthesis network for frame interpolation, they are prone to overfit the motion magnitude of training data. Our bi-directional motion estimator borrows some designs from modern optical flow models, but is much more lightweight, robust to large motion, and specially-optimized for simultaneous bi-directional motion estimation.

U-Net motion estimator for frame interpolation. U-Net [29] provides a powerful framework for dense prediction tasks. In recent years, U-Net and U-Net based pyramid networks have been adopted to estimate bi-directional mo-

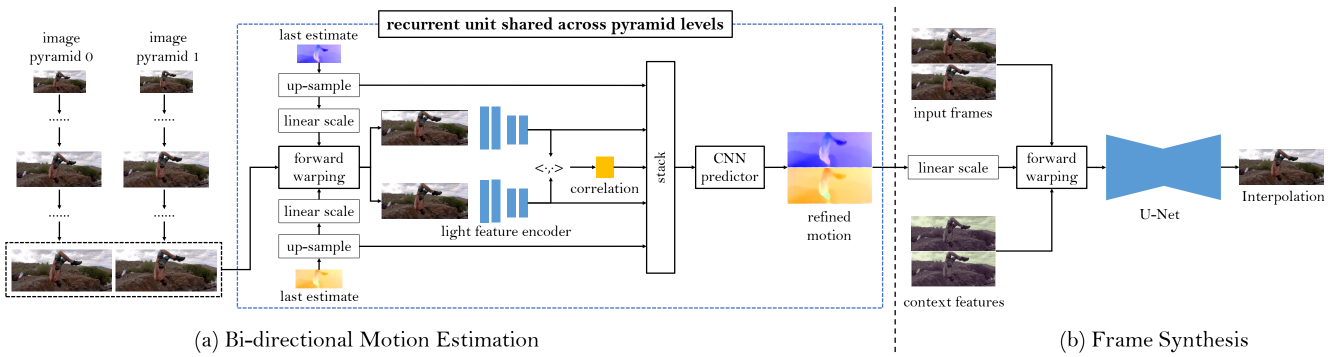


Figure 3. Overview of our frame interpolation pipeline. **(a)** We repeatedly apply a novel recurrent unit across image pyramids to refine estimated bi-directional motions between input frames. The recurrent unit is integrated with middle-oriented forward-warping, lightweight feature encoder, and a single correlation volume for simultaneous bi-directional motion estimation. **(b)** Based on estimated bi-directional motions, we forward-warp input frames and their context features, and employ synthesis network to predict the intermediate frame.

tions [12, 30] or bilateral intermediate motions [36, 8] for frame interpolation.

However, due to the lack of correlation-based representations, these models suffer from limited capacity in handling challenge motions (*e.g.*, local complex motion, small fast-moving objects). In addition, analogous to off-the-shelf optical flow models, plain U-Net has difficulty in estimating extreme large motion beyond the training data.

Flexible pyramid recurrent motion estimator. With recurrent design for both feature encoder and motion updater, recently proposed pyramid recurrent motion estimators can flexibly handle extreme large motion cases [36, 30, 15]. Since the recurrent unit (base estimator) can be applied on pyramid structure for multiple times, using a larger number of pyramid levels in testing can handle larger motions beyond the training phase.

The BiOF-I module [30] combines U-Net and pyramid recurrent structure for simultaneous bi-directional motion estimation. While BiOF-I enables excellent high-resolution frame interpolation[1], its U-Net based recurrent unit is over-simplified to handle challenging motion cases. Lee *et al.* [15] proposed Enhanced Correlation Matching (ECM) within a pyramid recurrent network. However, it is not designed for simultaneous bi-directional motion estimation. Furthermore, BiOF-I backward-warps input frames towards each other and ECM forward-warps one input frame towards another. Both warping strategies are not optimal in case of large motions, based on our experiments.

Forward-warping for frame interpolation. Compared to backward-warping, the motion field required by forward-warping is easier to acquire, and thus enables simpler pipeline for frame interpolation. However, forward-warping is less adopted for frame interpolation, partially because it may lead to holes in warped output. Niklaus and Liu [23] demonstrated that this issue may be remedied by warping both input frames. The holes in one warped frame can be filled by the context information from anther warped frame. Another limitation of forward-warping is that multiple pixels in source image may be mapped to the same target location. To solve this, softmax splatting [24] is developed to adaptively assigns weights to conflicted pixels.

With recent advances in forward-warping, we employ forward-warping for both motion estimation and frame synthesis. In particular, we use the average splatting operation in [24] as forward-warping, which directly averages the conflicted pixels to generate the pixel in target position. Average splatting is simpler than softmax splatting operation which relies on a confidence map.

3. Our Approach

3.1. Overview of the Pipeline

As shown in Figure 3, our frame interpolation pipeline involves two steps: (a) bi-directional motion estimation, and (b) frame synthesis. Our main innovation is the bi-directional motion estimator.

Formally, given two input frames I_0 and I_1, our goal is to predict the intermediate frame I_t at arbitrary time $t \in (0, 1)$. Firstly, we employ our novel bi-directional motion estimator to calculate the motion $F_{0\to1}$ and $F_{1\to0}$ between I_0 and I_1, and linearly scale them to obtain $F_{0\to t}$ and $F_{1\to t}$, *i.e.*, the motion from I_0 and I_1 to I_t:

$$\begin{aligned} F_{0\to t} &= t \cdot F_{0\to1} \\ F_{1\to t} &= (1-t) \cdot F_{1\to0} \end{aligned} \tag{1}$$

With $F_{0\to t}$ and $F_{1\to t}$, we forward-warp input frames and their context features, and feed warped representations into

[1]This is achieved by training on 4K dataset, and combining extra module to approximate the bilateral intermediate motions for backward-warping based frame synthesis.

a synthesis network to predict I_t. The synthesis network outputs a mask M for combining the warped frames, and a residual image ΔI_t for further refinement.

$$I_t = M \odot \overrightarrow{\mathcal{W}}(I_0, F_{0\to t}) + (1-M) \odot \overrightarrow{\mathcal{W}}(I_1, F_{1\to t}) + \Delta I_t \quad (2)$$

where $\odot$ denotes element-wise multiplication, $\overrightarrow{\mathcal{W}}$ denotes the forward-warping operation (average splatting [24]).

In testing, our bi-directional motion estimator can operate on flexible customizable image pyramids to handle large motion. Since motion magnitude scales with resolution, we suggest a simple method to calculate the number of pyramid levels in testing. Assume that the number of pyramid levels in training is L^{train}, and the averaged width (or height) of test images is n times of training images. Then, we can calculate the number of test pyramid levels as follows.

$$L^{test} = \texttt{ceil}(L^{train} + \texttt{log}_2 n) \quad (3)$$

where $\texttt{ceil}()$ rounds up a float number to get an integer.

3.2. Bi-directional Motion Estimation

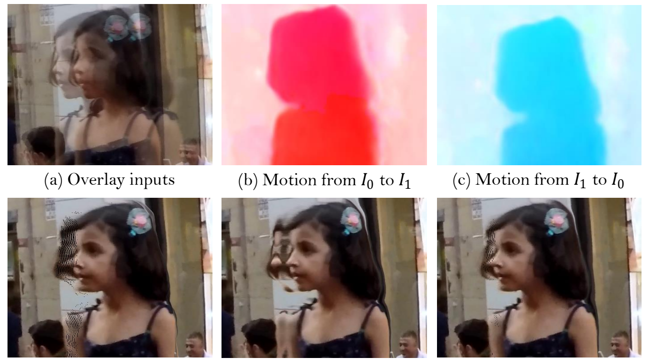
Pyramid recurrent framework and recurrent unit. As shown in Figure 3 (a), the macro structure of our bi-directional motion estimator is a pyramid recurrent network. Given two input frames, we firstly construct image pyramids for them, then repeatedly apply a novel recurrent unit across the pyramid levels to refine estimated bi-directional motions from coarse-to-fine.

At each pyramid level, we first up-sample the estimated bi-directional motions from previous level as initial motion (zero initialization for the top level). Based on scaled initial motion, we forward-warp both input frames to a hidden middle frame. Then, we employ an extremely lightweight feature encoder to extract CNN features for both warped frames. Lastly, we construct a correlation volume with CNN features of warped frames, and estimate the bi-directional motions from correlation injected features.

In the following, we detail the three key components involved in our recurrent unit: *middle-oriented forward-warping*, *extremely lightweight feature encoder*, and *correlation based bi-directional motion estimation*.

Middle-oriented forward-warping. Warping both input frames towards each other is a natural idea for simultaneous bi-directional motion estimation [30]. However, this comes with two disadvantages. First, it may lead to serious artifacts in warped output in case of large motions (see Figure 4 (d) and (e)). Second, two (rather than one) correlation volumes are required to record the matching scores between two original frames and the frames warped towards them.

Considering these, at i-th pyramid level, we firstly forward-warp both input frames I_0^i and I_1^i towards a hidden middle frame $I_{0.5}^i$, using linearly-scaled motions that have smaller magnitude than initial motions. Due to reduced motion magnitude, our middle-oriented forward-warping has the chance to reduce the impacts of possible artifacts caused by warping (see Figure 4 (f)). Furthermore, warping both input frames to a hidden frame allows us to construct a single correlation volume for simultaneous bi-directional motion estimation.



(a) Overlay inputs (b) Motion from I_0 to I_1 (c) Motion from I_1 to I_0

(d) forward-warped I_0 (e) backward-warped I_0 **(f) Our warped I_0**

Figure 4. Comparisons of different warping strategies in cases of large motion. Our *middle-oriented forward-warping* can reduce the possible artifacts caused by warping, as it uses linearly-scaled motion that has smaller magnitude.

Extremely lightweight feature encoder. Pyramidal optical flow models like PWC-Net [32] typically require a feature encoder with many down-sampling layers to construct feature pyramids. To handle large motion, PWC-Net employs a feature encoder of 6 down-sampling layers.

Our motion estimator handles large motion by customizing the number of pyramid levels of *outer* image pyramids. Thus, the feature encoder involved in *inner* recurrent unit does not need many down-sampling layers. We employ an extremely lightweight feature encoder with only two down-sampling layers to extract CNN features for both warped frames. It has only about 0.1 M parameters, while PWC-Net's feature encoder has 1.7 M parameters.

Correlation-based bi-directional motion estimation. Existing works construct a correlation volume between one original frame and another frame warped towards it to estimate single-directional motion between input frames [32, 15]. While for simultaneous bi-directional motion estimation, two correlation volumes are required, if input frames are warped towards each other.

Instead, we construct a *single* correlation volume for simultaneous bi-directional motion estimation, using CNN features of both warped frames that have compensated for estimated bi-directional motions. Following PWC-Net [32], we set the local search range on the feature map of the second warped frame as 4. We concatenate the correlation volume, CNN features, and up-sampled bi-directional motions to form input features, and use a 6-layer convolutional network to predict the bi-directional motions. Since our feature encoder has two down-sampling layers, the estimated motion is at 1/4 resolution of the input frame. We use bi-linear interpolation to up-scale the motion to original scale.

3.3. Frame Synthesis

Based on estimated bi-directional motions, we employ a synthesis network to predict the intermediate frame from forward-warped representations.

A simple baseline synthesis network. Our synthesis network follows the design of previous context-aware synthesis networks [24, 8], which take both warped frames and warped context features as input. We extract 4-level pyramid context features for both input frames.

We employ a simple U-Net as our synthesis network, which has four down-sampling layers, and four up-sampling layers. It takes warped frames, warped context features, original images, and bi-directional motions as input, and outputs a mask M for combining the warped frames, and a residual image ΔI_t for further refinement (see Equation 2). We refer to this synthesis network as our *base* synthesis network.

High-resolution synthesis with convex down-sampling. Higher resolution input often has advantages for dense prediction tasks [28, 16]. We verify this for frame synthesis. Specifically, we up-sample the input frames and estimated bi-directional motions to 2x resolution, feed them to our synthesis network, and obtain a 2x resolution interpolation. To recover the original scale, we add a lightweight head to our synthesis network to predict 5×5 dynamic filters for the pixels with stride 2 on the 2x resolution interpolation. These filters allow us to take a convex weighted combination over 5×5 neighborhoods on the 2x resolution interpolation to predict each pixel of the target frame of original scale.

This convex down-sampling strategy achieves better performance than bi-linear down-sampling, 0.1 dB improvement on the "extreme" subset of SNU-FILM [6]. We refer to this structure as *high-resolution* synthesis network.

3.4. Architecture Variants

We name our frame interpolation method as EBME – **E**nhanced **B**i-directional **M**otion **E**stimation for frame interpolation. We construct three versions of EBME, with almost the same model size but increased computational cost:

- EBME: It combines our bi-directional motion estimator with the base version of synthesis network.
- EBME-H: It combines our motion estimator with the high-resolution version of synthesis network.
- EBME-H*: It uses the test-time augmentation (refer to Section 3.5) with EBME-H, which doubles the computational cost but further improves performance.

3.5. Implementation Details

Loss function. For fair comparisons with recent works, all models are trained only with the synthesis loss, without auxiliary supervision for motion. Our loss is weighted sum of Charbonnier loss [4] and census loss [21] between ground truth I_t^{GT} and our interpolation I_t:

$$L = \rho(I_t^{GT} - I_t) + \lambda \cdot L_{cen}(I_t^{GT}, I_t), \tag{4}$$

where $\rho(x) = (x^2 + \epsilon^2)^\alpha$ is the Charbonnier function, L_{cen} is the census loss, and λ is a trade-off hyper-parameter. We empirically set $\alpha = 0.5$, $\epsilon = 10^{-6}$, $\lambda = 0.1$.

Training dataset. We train our model on the Vimeo90K dataset [35]. Vimeo90K contains 51,312 triplets with resolution of 448×256 for training. We augment the training images by randomly cropping 256×256 patches. We also apply random flipping, rotating, reversing the order of the triplets for data augmentation.

Optimization. Our optimizer is AdamW [18] with weight decay 10^{-4} for 0.8 M iterations, using a batch size of 32. We gradually reduce the learning rate during training from 2×10^{-4} to 2×10^{-5} using cosine annealing.

Test-time augmentation. We verify a practice strategy described in [8]. We flip the input frames horizontally and vertically to get augmented test data, and use our model to infer two results and reverse the flipping. A more robust prediction can be obtained by averaging these two results.

4. Experiments

4.1. Experiment Settings

Evaluation datasets. While our method is trained only on Vimeo90K [35], we evaluate it on a broad range of benchmarks with different resolutions.

- **UCF101 [31]**: The test set of UCF101 contains 379 triplets with a resolution of 256×256. UCF101 contains a large variety of human actions.
- **Vimeo90K [35]**: The test set of Vimeo90K contains 3,782 triplets with a resolution of 448×256.
- **SNU-FILM [6]**: This dataset contains 1,240 triplets, and most of them are of the resolution around 1280×720. It contains four subsets with increasing motion scales – easy, medium, hard, and extreme.
- **4K1000FPS [30]**: This is a 4K resolution benchmark that supports multi-frame (×8) interpolation.

methods	UCF101	Vimeo90K	SNU-FILM				parameters	runtime
			easy	medium	hard	extreme	(millions)	(seconds)
DAIN [2]	34.99/0.968	34.71/0.976	39.73/0.990	35.46/0.978	30.17/0.934	25.09/0.858	24.0	0.15
CAIN [6]	34.91/0.969	34.65/0.973	39.89/0.990	35.61/0.978	29.90/0.929	24.78/0.851	42.8	0.04
SoftSplat [24]	35.39/0.952	36.10/0.970	-	-	-	-	-	-
AdaCoF [14]	34.90/0.968	34.47/0.973	39.80/0.990	35.05/0.975	29.46/0.924	24.31/0.844	22.9	0.03
BMBC [26]	35.15/0.969	35.01/0.976	39.90/0.990	35.31/0.977	29.33/0.927	23.92/0.843	11.0	0.82
ABME [27]	35.38/**0.970**	36.18/**0.981**	39.59/0.990	35.77/0.979	30.58/0.936	**25.42**/**0.864**	18.1	0.28
XVFI$_v$ [30]	35.18/0.952	35.07/0.968	39.78/0.984	35.37/0.964	29.91/0.894	24.73/0.778	5.5	0.10
ECM$_v$ [15]	34.97/0.951	34.95/0.975	-	-	-	-	4.7	-
EBME (ours)	35.30/0.969	35.58/0.978	40.01/**0.991**	35.80/0.979	30.42/0.935	25.25/0.861	**3.9**	**0.02**
EBME-H (ours)	35.35/0.969	36.06/0.980	40.20/**0.991**	36.00/**0.980**	30.54/0.936	25.30/0.862	**3.9**	0.04
EBME-H* (ours)	**35.41**/**0.970**	**36.19**/**0.981**	**40.28**/**0.991**	**36.07**/**0.980**	**30.64**/**0.937**	25.40/0.863	**3.9**	0.08

Table 1. Qualitative (PSNR/SSIM) comparisons to state-of-the-art methods on UCF101 [31], Vimeo90K [35] and SNU-FILM [6] benchmarks. **RED**: best performance, BLUE: second best performance.

Figure 5. Visual comparisons on two examples from the "extreme" subset of SNU-FILM [6]. The first two rows show the synthesis results for detailed textures, while the last two rows demonstrate the results with complex and large motion.

Metrics. We calculate peak signal-to-noise ratio (PSNR) and structure similarity (SSIM) for quantitative evaluation of interpolation. For the running time, we follow the practice of [27], and test all models with a RTX 2080 Ti GPU for interpolating the "Urban" sequence in Middle-bury benchmark [1], which has a resolution of 640 $\times$ 480.

Customized number of pyramid levels. We use 3-level image pyramids when training on low-resolution Vimeo90K [35]. For benchmark datasets, UCF101 [31] has similar resolution with Vimeo90K, SNU-FILM has a resolution of about 720p, and 4K1000FPS has a resolution of 4K. Based on our suggested calculation method by Equation 3, we set the test pyramid levels for UCF-101, SNU-FILM and 4K1000FPS as 3, 5 and 7, respectively.

4.2. Comparisons with State-of-the-art Methods

We compare with state-of-the-art methods, including DAIN [2], CAIN [6], SoftSplat [24], AdaCoF [14], BMBC [26], ABME [27], XVFI [30], and ECM [15]. We report their results by executing the source code and trained models, except for SoftSplat and ECM which have not released the full code. For SoftSplat and ECM, we copy the results from original paper. To test XVFI$_v$ on SNU-FILM, we adjust the number of scale levels so that it has the same down-sampling factor with our motion estimator.

Parameter and inference efficiency. As shown the last two columns in Table 1, our frame interpolation algorithm has much less parameters than state-of-the-art methods and

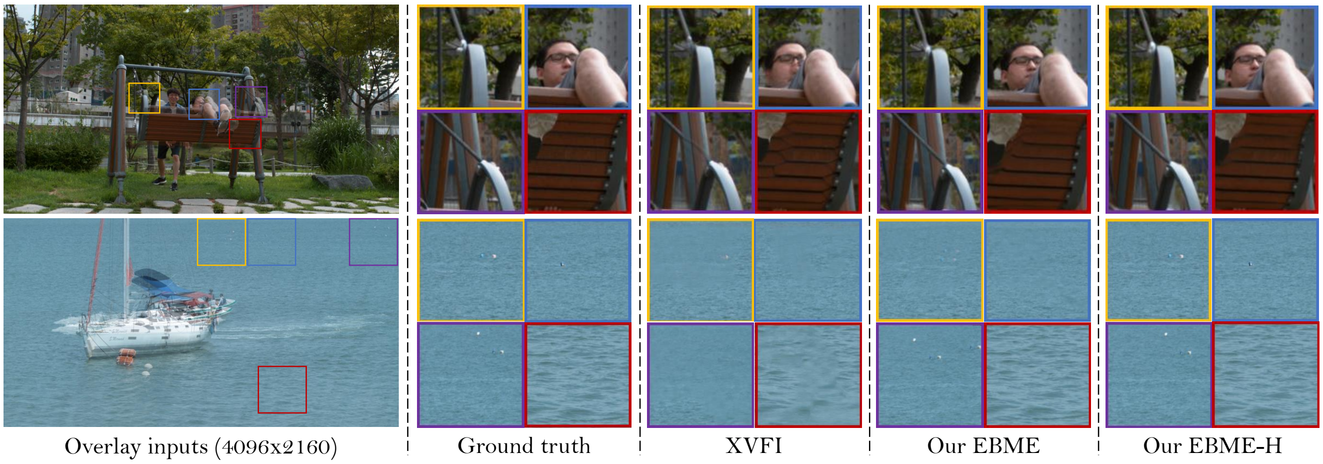


Figure 6. Visual comparisons on 4K1000FPS [30]. XVFI [30] trends to miss the moving small objects, while our EBME-H gives interpolation results close to the ground truth.

methods	arbitrary	reuse flow	4K1000FPS	
			PSNR	SSIM
DAIN [2]	✓	✓	26.78	0.807
AdaCoF [14]	×	×	23.90	0.727
ABME [27]	✓	×	**30.16**	0.879
XVFI [30]	✓	partial	30.12	0.870
EBME (ours)	✓	✓	27.86	0.881
EBME-H (ours)	✓	✓	28.72	0.889
EBME-H* (ours)	✓	✓	29.46	**0.902**

Table 2. Comparisons on 4K1000FPS [35] for 8x interpolation.

runs very fast. In particular, due to the macro recurrent design and the lightweight feature encoder, our bi-directional motion estimator only has about 0.6 M parameters.

Low and moderate resolution frame interpolation. Table 1 reports the comparison results on low-resolution UCF101 and Vimeo90K datasets. Our EBME-H* achieves best performance on both benchmarks. Our EBME also outperforms many state-of-the-art models including DAIN, CAIN, AdaCoF, BMBC, XVFI$_v$, and ECM.

Table 1 also reports the comparison results on SNU-FILM. Our EBME-H and EBME-H* perform similar with ABME [27] on the "hard" and "extreme" subsets, but have better performance on the "easy" and "medium" subsets. It is worth noting that our models are about 4.5x smaller than ABME, and run much faster.

Figure 5 gives two examples from the "extreme" subset from SNU-FILM. Our methods produce better interpolation results than ABME for some detailed textures (first two rows), and give promising results for large motion cases (last two rows), much better than CAIN and AdaCoF, and sightly better than ABME.

4K resolution multiple frame interpolation. Table 2 reports the 8x interpolation results on 4K1000FPS. Our method achieves the best performance by SSIM, but slight inferior results to ABME and XVFI by PSNR. Note that XVFI is trained on 4K high-resolution data, while other models are trained on low-resolution data. Our method supports arbitrary-time frame interpolation, and can fully re-use estimated bi-directional motions when interpolating multiple intermediate frames at different time positions. By contrast, while XVFI [30] can reuse the bi-directional motions, it must refine the approximated intermediate flow with an extra network at each time position.

Figure 6 shows two interpolation examples. Our methods give better performance for moving small objects. The U-Net based pyramid motion estimator in XVFI might have difficulty in capturing the motion of extreme small objects.

4.3. Analysis of Our Motion Estimator

We present analysis of our motion estimator on the "hard" and "extreme" subsets of SNU-FILM [6], which contain various challenging motion cases.

Design Choices of Motion Estimator. In Table 3, we report the ablation results for the design choices of our bi-directional motions estimator.

- **Simultaneous bi-directional estimation**: Our bi-directional motion estimator performs better than its single-directional counterpart that forward-warps the first frame to the second and constructs a correlation volume with warped frame and second frame. We run the single-directional counterpart twice to obtain bi-directional motions. We verify that simultaneous bi-directional motion estimation can improve per-

experiments	methods	SNU-FILM (PSNR ↑)	
		hard	extreme
bi-directional	simultaneous	30.42	25.25
	single-direction	30.19	25.12
warping type	forward	30.36	25.21
	middle-forward	30.42	25.25
	backward	30.28	25.11
feature encoder	1-stage	30.36	25.20
	2-stage	30.42	25.25
	3-stage	30.26	25.15
correlation	without	30.29	25.17
	with	30.42	25.25
test pyramid	3-level	30.15	24.80
	4-level	30.42	25.20
	5-level	30.42	25.25
	6-level	30.40	25.22

Table 3. Impacts of the design choices of our bi-directional motion estimator, integrated with base synthesis network for frame interpolation. Default settings are marked in gray .

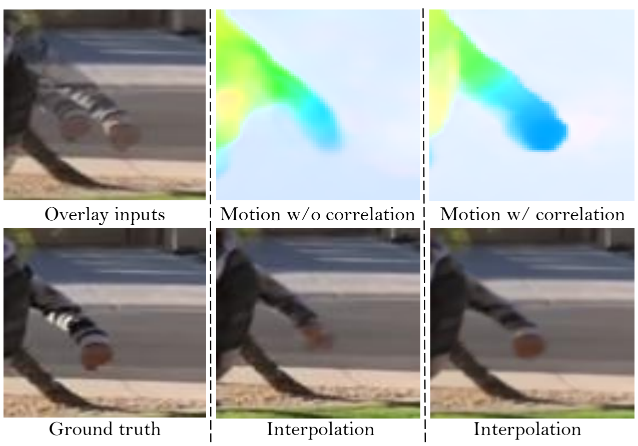


Figure 7. Without correlation volume, our estimator may fail to estimate complex motion, and lead to artifacts on interpolated frame.

formance, and our middle-oriented warping also improves robustness against large motion.

- **Warping type**: Our middle-oriented forward-warping (denoted as "middle-forward") achieves better performance than forward-warping and backward-warping that align input frames towards each other. Note that aligning input frames to each other needs to build two correlation volumes for the original two frames and warped frames, while our warping method enables the building of single correlation volume.

- **Feature encoder**: We investigate three settings for our feature encoder: one convolutional stage of 9 layers; two-stage with 3 layers for first stage, and 6 layers for second stage; three-stage with 3 layers for each stage. We double the number of filters with down-sampling layers. More down-sampling layers might be beneficial for large motion, but may lead to rough estimate. Two-stage feature encoder achieves the best trade-off.

experiments	methods	SNU-FILM (PSNR ↑)		param.
		hard	extreme	(M)
warp approx.	PWC-Net	28.37	23.59	9.4
	BiOF-I	28.13	23.68	2.6
	Ours	**28.62**	**24.00**	**0.6**
full pipeline	PWC-Net	30.04	24.53	12.7
	BiOF-I	30.03	24.80	5.9
	Ours	**30.42**	**25.25**	**3.9**

Table 4. Quantitative results of frame interpolation, enabled by PWC-Net [32], BiOF-I [30], and our motion estimator.

- **Correlation volume**: Removing correlation volume from our motion model leads to inferior quantitative results. Furthermore, as shown in Figure 7, without a correlation volume, our estimator may have difficulty in estimating complex nonlinear motions, and lead to blurry artifacts in local regions.

- **Test pyramid level**: A 5-level image pyramid achieves good performance on the "extreme" subset. Further increasing pyramid level does not lead to better results. This is consistent with our suggested calculation method described by Equation 3.

Motion Quality Comparison. We compare our bi-directional motion estimator with PWC-Net [32] and BiOF-I [30] for frame interpolation. We end-to-end train PWC-Net and BiOF-I from scratch with our basic synthesis network. We adjust the number of scale levels for BiOF-I so that it has the same down-sampling factor with our bi-directional motion estimator when testing on SNU-FILM.

We compare motion estimators for frame interpolation from two aspects: interpolation by averaging two forward-warped frames, and interpolation by our full pipeline. As shown in Table 4, our motion estimator enables much better interpolation results on the "extreme" subset. In addition, it is much smaller in size than PWC-Net and BiOF-I.

5. Conclusion

This work presented a lightweight yet effective frame interpolation algorithm, based on a novel bi-directional motion estimator. Our method achieved excellent performance on various frame interpolation benchmarks. This work aims at motion-based frame interpolation, and does not pursue the motion accuracy on optical flow benchmarks. In the future, we will verify the effectiveness of our lightweight motion estimator for general-purpose optical flow.

References

[1] Simon Baker, Daniel Scharstein, JP Lewis, Stefan Roth, Michael J Black, and Richard Szeliski. A database and evaluation methodology for optical flow. *IJCV*, 2011.

[2] Wenbo Bao, Wei-Sheng Lai, Chao Ma, Xiaoyun Zhang, Zhiyong Gao, and Ming-Hsuan Yang. Depth-aware video frame interpolation. In *CVPR*, 2019.

[3] Wenbo Bao, Wei-Sheng Lai, Xiaoyun Zhang, Zhiyong Gao, and Ming-Hsuan Yang. MEMC-Net: Motion estimation and motion compensation driven neural network for video interpolation and enhancement. *TPAMI*, 2019.

[4] Pierre Charbonnier, Laure Blanc-Feraud, Gilles Aubert, and Michel Barlaud. Two deterministic half-quadratic regularization algorithms for computed imaging. In *ICIP*, 1994.

[5] Xianhang Cheng and Zhenzhong Chen. Video frame interpolation via deformable separable convolution. In *AAAI*, 2020.

[6] Myungsub Choi, Heewon Kim, Bohyung Han, Ning Xu, and Kyoung Mu Lee. Channel attention is all you need for video frame interpolation. In *AAAI*, 2020.

[7] John Flynn, Ivan Neulander, James Philbin, and Noah Snavely. Deepstereo: Learning to predict new views from the world's imagery. In *CVPR*, 2016.

[8] 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.

[9] Tak-Wai Hui, Xiaoou Tang, and Chen Change Loy. LiteFlowNet: A lightweight convolutional neural network for optical flow estimation. In *CVPR*, 2018.

[10] Junhwa Hur and Stefan Roth. Iterative residual refinement for joint optical flow and occlusion estimation. In *CVPR*, 2019.

[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 *CVPR*, 2017.

[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 *ICCV*, 2018.

[13] Yoshihiko Kuroki, Haruo Takahashi, Masahiro Kusakabe, and Ken-ichi Yamakoshi. Effects of motion image stimuli with normal and high frame rates on EEG power spectra: comparison with continuous motion image stimuli. *Journal of the Society for Information Display*, 2014.

[14] Hyeongmin Lee, Taeoh Kim, Tae-young Chung, Daehyun Pak, Yuseok Ban, and Sangyoun Lee. AdaCoF: Adaptive collaboration of flows for video frame interpolation. In *CVPR*, 2020.

[15] Sungho Lee, Narae Choi, and Woong Il Choi. Enhanced correlation matching based video frame interpolation. In *WACV*, 2022.

[16] Ze Liu, Han Hu, Yutong Lin, Zhuliang Yao, Zhenda Xie, Yixuan Wei, Jia Ning, Yue Cao, Zheng Zhang, Li Dong, et al. Swin transformer v2: Scaling up capacity and resolution. *arXiv preprint arXiv:2111.09883*, 2021.

[17] Ziwei Liu, Raymond A Yeh, Xiaoou Tang, Yiming Liu, and Aseem Agarwala. Video frame synthesis using deep voxel flow. In *ICCV*, 2017.

[18] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

[19] Guo Lu, Xiaoyun Zhang, Li Chen, and Zhiyong Gao. Novel integration of frame rate up conversion and HEVC coding based on rate-distortion optimization. *TIP*, 2017.

[20] Liying Lu, Ruizheng Wu, Huaijia Lin, Jiangbo Lu, and Jiaya Jia. Video frame interpolation with transformer. In *CVPR*, 2022.

[21] Simon Meister, Junhwa Hur, and Stefan Roth. Unflow: Unsupervised learning of optical flow with a bidirectional census loss. In *AAAI*, 2018.

[22] Simone Meyer, Abdelaziz Djelouah, Brian McWilliams, Alexander Sorkine-Hornung, Markus Gross, and Christopher Schroers. Phasenet for video frame interpolation. In *CVPR*, 2018.

[23] Simon Niklaus and Feng Liu. Context-aware synthesis for video frame interpolation. In *CVPR*, 2018.

[24] Simon Niklaus and Feng Liu. Softmax splatting for video frame interpolation. In *CVPR*, 2020.

[25] Simon Niklaus, Long Mai, and Feng Liu. Video frame interpolation via adaptive separable convolution. In *ICCV*, 2017.

[26] Junheum Park, Keunsoo Ko, Chul Lee, and Chang-Su Kim. BMBC: Bilateral motion estimation with bilateral cost volume for video interpolation. In *ECCV*, 2020.

[27] Junheum Park, Chul Lee, and Chang-Su Kim. Asymmetric bilateral motion estimation for video frame interpolation. In *ICCV*, 2021.

[28] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018.

[29] 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*, 2015.

[30] Hyeonjun Sim, Jihyong Oh, and Munchurl Kim. XVFI: Extreme video frame interpolation. In *ICCV*, 2021.

[31] 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.

[32] Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz. PWC-Net: CNNs for optical flow using pyramid, warping, and cost volume. In *CVPR*, 2018.

[33] Zachary Teed and Jia Deng. RAFT: Recurrent all-pairs field transforms for optical flow. In *ECCV*, 2020.

[34] Jiyan Wu, Chau Yuen, Ngai-Man Cheung, Junliang Chen, and Chang Wen Chen. Modeling and optimization of high frame rate video transmission over wireless networks. *IEEE Transactions on Wireless Communications*, 2015.

[35] Tianfan Xue, Baian Chen, Jiajun Wu, Donglai Wei, and William T Freeman. Video enhancement with task-oriented flow. *IJCV*, 2019.

[36] Haoxian Zhang, Yang Zhao, and Ronggang Wang. A flexible recurrent residual pyramid network for video frame interpolation. In *ECCV*, 2020.