RIFE: Real-Time Intermediate Flow Estimation for Video Frame Interpolation

Zhewei Huang[1] Tianyuan Zhang[1] Wen Heng[1] Boxin Shi[2] Shuchang Zhou[1]
[1]Megvii Inc [2]Peking University
{huangzhewei, zhangtianyuan, hengwen, zsc}@megvii.com, shiboxin@pku.edu.cn

Abstract

We propose RIFE, a Real-time Intermediate Flow Estimation algorithm for Video Frame Interpolation (VFI). Many recent flow-based VFI methods first estimate the bi-directional optical flows, then scale and reverse them to approximate intermediate flows, leading to artifacts on motion boundaries and complex pipelines. RIFE uses a neural network named IFNet that can directly estimate the intermediate flows from coarse-to-fine with much better speed. We design a privileged distillation scheme for training IFNet, resulting in a large performance improvement. RIFE does not rely on pre-trained optical flow models and can support arbitrary-timestep frame interpolation with the temporal encoding input. Experiments demonstrate that RIFE achieves state-of-the-art performance on several public benchmarks. Compared with the popular SuperSlomo and DAIN methods, RIFE is 4–27 times faster and produces better results. The code is available at https://github.com/hzwer/arXiv2020-RIFE.

1. Introduction

Video Frame Interpolation (VFI) aims to synthesize intermediate frames between two consecutive video frames. VFI supports various applications like slow-motion generation, video compression [44], and novel view synthesis. Moreover, real-time VFI algorithms running on high-resolution videos have many potential applications, such as reducing bandwidth requirements for live video streaming, providing video editing services for users with limited computing resources, and video frame rate adaption on the display devices.

VFI is challenging due to the complex, large non-linear motions and illumination changes in real-world videos. Recently, flow-based VFI algorithms have offered a framework to address these challenges and achieved impressive results [25, 17, 32, 48, 3, 47, 23]. Common approaches for these methods involve two steps: 1) warping the input frames according to approximated optical flows and 2) fusing and refining the warped frames using Convolutional Neural Networks (CNNs).

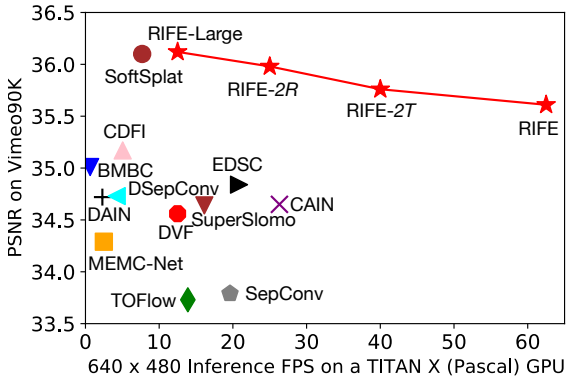


Figure 1: **Speed and accuracy trade-off for different model size settings.** Results are reported for Vimeo90K benchmark [48].

Optical flow models can not be directly used in VFI. Given the input frames I_0, I_1, flow-based methods [25, 17, 3] need to approximate the intermediate flows $F_{t\to0}, F_{t\to1}$ from the perspective of the frame I_t that we are expected to synthesize. There is a "chicken-and-egg" problem between intermediate flows and frames because I_t is not available beforehand, and its estimation is a difficult problem [17, 36]. Many practices [17, 3, 47, 23] first compute bi-directional flows from optical flow models, then reverse and refine them to generate intermediate flows. However, such flows may have flaws in motion boundaries, as the object position changes from frame to frame ("object shift" problem). Another pioneering work, DVF [25] proposes voxel flow to jointly model the intermediate flow and occlusion mask by using CNNs to estimate them end-to-end. AdaCoF [21] further extends intermediate flows to adaptive collaborative flows. BMBC [36] designs a bilateral cost volume operator for obtaining more accurate intermediate flows.

In this paper, we aim to build a lightweight pipeline that achieves state-of-the-art performance while maintaining the conciseness of direct intermediate flow estimation. Our pipeline has two main design concepts:

1) Not requiring additional components, like image depth model [3], flow refinement model [17] and flow reversal layer [47], which are introduced to compensate for the defects of intermediate flow estimation. We also want to eliminate reliance on pre-trained state-of-the-art optical flow models that are not tailored for VFI tasks.

2) Providing direct supervision for the approximated intermediate flows: To the best of our knowledge, most VFI models are trained with only the final reconstruction loss. Lacking supervision explicitly designed for the flow estimation, degrades the performance of interpolation.

We propose IFNet, which directly estimates intermediate flow from adjacent frames and a temporal encoding input. IFNet adopts a coarse-to-fine strategy [16] with progressively increasing resolution: it iteratively updates intermediate flows and soft fusion mask via successive IFBlocks. Conceptually, according to the iteratively updated flow fields, we could move corresponding pixels from two input frames to the same location in a latent intermediate frame and use a fusion mask to combine pixels from two input frames. To make our model lightweight, unlike most previous optical flow models [12, 16, 42, 15, 43], IFBlocks do not contain expensive operators like cost volume and only use 3×3 convolution and deconvolution as building blocks, which are found to be efficient on resource-constrained devices [11].

Employing intermediate supervision is very important. When training the IFNet end-to-end using the final reconstruction loss, our method produces worse results than state-of-the-art methods because of the inaccurate optical flow estimation. The situation dramatically changes after we design a privileged distillation scheme that employs a teacher model with access to the intermediate frames to guide the student to learn.

Combining these designs, we propose the Real-time Intermediate Flow Estimation (**RIFE**). RIFE can achieve satisfactory results when trained from scratch, without requiring pre-trained optical flow models or datasets with optical flow labels. We illustrate the speed and accuracy trade-off compared with other methods in Figure 1.

To sum up, our main contributions include:

- We design an effective coarse-to-fine IFNet using simple operators to directly approximate the intermediate flows.
- We introduce a privileged distillation scheme for IFNet, which leads to large performance improvement.
- Our experiments demonstrate that RIFE achieves state-of-the-art performance on several public benchmarks.
- In addition, we show RIFE can be extended to applications such as depth map interpolation and dynamic scene panorama stitching, thanks to its high-quality flow estimation and flexible temporal encoding.

2. Related Works

Optical flow estimation. Optical flow estimation is a long-standing vision task that aims to estimate the per-pixel motion, useful in many downstream tasks. Since the milestone work of FlowNet [12] based on U-net autoencoder [40], architectures for optical flow models have evolved for several years, yielding more accurate results while being more efficient, such as FlowNet2 [16], PWC-Net [42] and LiteFlowNet [15]. Recently Teed *et al*. [43] introduce RAFT, which iteratively updates a flow field through a recurrent unit and achieves a remarkable breakthrough in this field. Another important research direction is unsupervised optical flow estimation [29, 18, 28] which tackles the difficulty of optical flow labeling.

Video frame interpolation. Recently, the optical flow has been a prevalent component in video interpolation. In addition to the method of directly estimating the intermediate flow [25, 21, 36], Jiang *et al*. [17] propose SuperSlomo using the linear combination of the two bi-directional flows as an initial approximation of the intermediate flows and then refine them using U-Net. Reda *et al*. [39] and Liu *et al*. [24] propose to improve intermediate frames using cycle consistency. Bao *et al*. [3] propose DAIN to estimate the intermediate flow as a weighted combination of bidirectional flow. Niklaus *et al*. [33] propose SoftSplat to forward-warp frames and their feature map using softmax splatting. Xu *et al*. [47] propose QVI to exploit four consecutive frames and flow reversal filter to get the intermediate flows. Liu *et al*. [23] further extends QVI with rectified quadratic flow prediction to EQVI.

Along with flow-based methods, flow-free methods have also achieved remarkable progress in recent years. Meyer *et al*. [31, 30] utilize phase information to learn the motion relationship for multiple video frame interpolation. Niklaus *et al*. [34, 35] formulate VFI as a spatially adaptive convolution whose convolution kernel is generated using a CNN given the input frames. Cheng *et al*. propose DSepConv [8] to extend kernel-based method using deformable separable convolution and further propose EDSC [7] to perform multiple interpolation. Choi *et al*. [9] propose an efficient flow-free method named CAIN, which employs the PixelShuffle operator and channel attention to capture the motion information implicitly. Some work further focus on increasing the resolution and frame rate of the video together and has achieved good visual effect [45, 46].

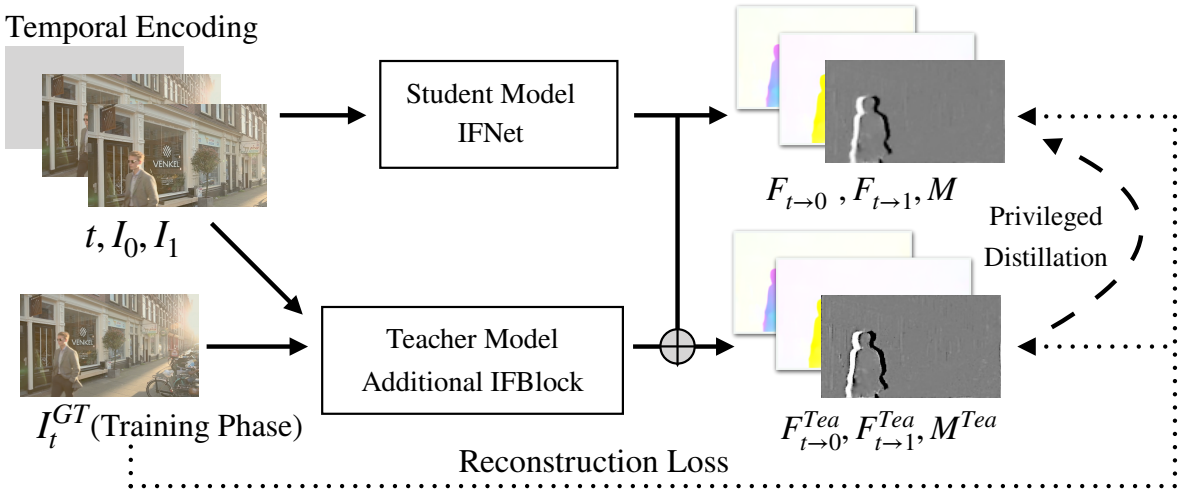


Figure 2: **Overview of RIFE pipeline.** Given two input frames I_0, I_1 and temporal encoding t (timestep encoded as an separate channel), we directly feed them into the IFNet to approximate intermediate flows $F_{t\to0}, F_{t\to1}$ and the fusion map M. During the training phase, a privileged teacher refines student's results based on ground truth I_t using a special IFBlock. The student model and the teacher model are jointly trained from scratch using the reconstruction loss. The teacher's approximations are more accurate so that they can guide the student to learn.

Knowledge distillation. Our privileged distillation [26] for intermediate flow conceptually belongs to the knowledge distillation [14] method, which originally aims to transfer knowledge from a large model to a smaller one. In privileged distillation, the teacher model gets more input than the student model, such as scene depth, images from other views, and even image annotation. Therefore, the teacher model can provide more accurate representations to guide the student model to learn. This idea is applied to some computer vision tasks, such as image super resolution [22], hand pose estimation [49], re-identification [37] and video style transfer [6]. Our work is also related to codistillation [1] where student and teacher have the same architecture and different inputs during training.

3. Method

We first provide an overview of RIFE. Then we describe the major components in RIFE, elaborate on our proposed distillation scheme, and explain the training details.

3.1. Pipeline Overview

We illustrate the overall pipeline of RIFE in Figure 2. Given a pair of consecutive RGB frames, I_0, I_1 and target timestep t $(0 \le t \le 1)$, our goal is to synthesize an intermediate frame $\widehat{I}_t$. We estimate the intermediate flows $F_{t\to0}$, $F_{t\to1}$ and fusion map M by feeding input frames and t as an additional channel into the IFNet. We can get reconstructed image $\widehat{I}_t$ using following formulation:

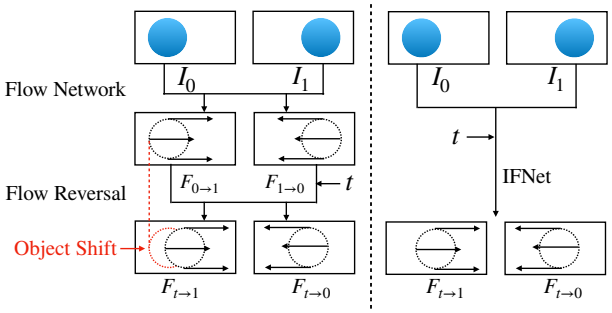


Figure 3: **Comparison between indirect intermediate flow estimation approaches [17, 47, 3, 23] (left) and the IFNet (right).** These previous methods contain two stages: 1) bi-directional flow estimation and 2) flow reversal modules. As the object shifts, they may have flaws in motion boundaries. RIFE directly estimates the intermediate flows using coarse-to-fine IFNet.

$$\widehat{I}_t = M \odot \widehat{I}_{t\leftarrow0} + (1 - M) \odot \widehat{I}_{t\leftarrow1}, \quad (1)$$

$$\widehat{I}_{t\leftarrow0} = \overleftarrow{\mathcal{W}}(I_0, F_{t\to0}), \quad \widehat{I}_{t\leftarrow1} = \overleftarrow{\mathcal{W}}(I_1, F_{t\to1}). \quad (2)$$

where $\overleftarrow{\mathcal{W}}$ is the image backward warping, $\odot$ is an element-wise multiplier, and M is the fusion map $(0 \le M \le 1)$. We use another encoder-decoder CNNs named RefineNet following previous methods [17, 33] to refine the high-frequency area of $\widehat{I}_t$ and reduce artifacts of the student model. Its computational cost is similar to the IFNet. The RefineNet finally produce a reconstruction residual

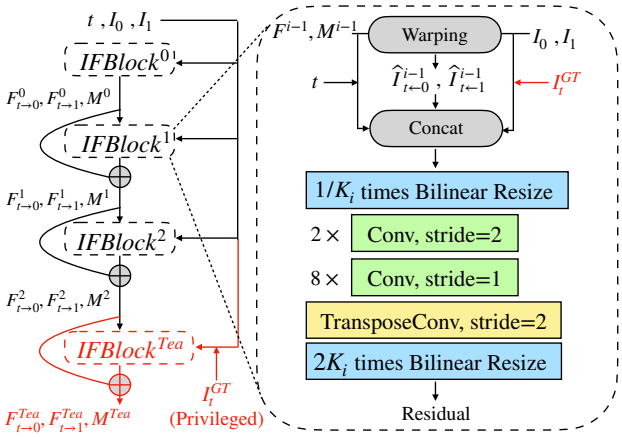


Figure 4: **Left**: The IFNet is composed of several stacked IFBlocks operating at different resolution. **Right**: In an IFBlock, we first backward warp the two input frames based on current approximated flow F^{i-1}. Then the input frames I_0, I_1, warped frames $\widehat{I}_{t\leftarrow 0}, \widehat{I}_{t\leftarrow 1}$, the previous results F^{i-1}, M^{i-1} and timestep t are fed into the next IFBlock to approximate the residual of flow and mask. The privileged information I_t^{GT} is only provided for teacher.

Δ $(-1 \leq \Delta \leq 1)$. And we will get a refined reconstructed image $\widehat{I}_t + \Delta$. The detailed architecture of RefineNet is in the Appendix.

3.2. Intermediate Flow Estimation

Some previous VFI methods reverse and refine bi-directional flows [17, 47, 3, 23] as depicted in Figure 3. The flow reversal process is usually cumbersome due to the difficulty of handling the changes of object positions. Intuitively, the previous flow reversal method hopes to perform spatial interpolation on the optical flow field, which is not trivial because of "object shift" problem. The role of our IFNet is to directly and efficiently predict $F_{t\to 0}, F_{t\to 1}$ and fusion mask M given two consecutive input frames I_0, I_1 and timestep t. When $t = 0$ or $t = 1$, IFNet is similar to the classical optical flow models.

To handle the large motion encountered in intermediate flow estimation, we employ a coarse-to-fine strategy with gradually increasing resolution, as illustrated in Figure 4. Specifically, we first compute a rough prediction of the flow on low resolution, which is believed to capture large motions easier, then iteratively refine the flow fields with gradually increasing resolution. Following this design, our IFNet has a stacked hourglass structure, where a flow field is iteratively refined via successive IFBlocks:

$$\begin{bmatrix} F^i \\ M^i \end{bmatrix} = \begin{bmatrix} F^{i-1} \\ M^{i-1} \end{bmatrix} + \text{IFB}^i\left(\begin{bmatrix} F^{i-1} \\ M^{i-1} \end{bmatrix}, t, \widehat{I}^{i-1}\right), \quad (3)$$

Method	PWC-Net [42]	RAFT [43]	IFNet
Runtime	2×21ms	2×52ms	**7ms**

Table 1: **Average inference time on the** 640×480 **frames**. Recent flow-based VFI methods [17, 3, 33] run the optical flow model twice to obtain bi-directional optical flows.

where F^{i-1} and M^{i-1} denote the current estimation of the intermediate flows and fusion map from the $(i-1)^{th}$ IFBlock, and IFB^i represents the i^{th} IFBlock. We use a total of 3 IFBlocks, and each has a resolution parameter, $(K_0, K_1, K_2) = (4, 2, 1)$. During inference time, the final estimation is F^n and $M^n (n = 2)$. To keep our design simple, each IFBlock has a feed-forward structure consisting of serveral convolutional layers and an up-sampling operator. Except for the layer that outputs the optical flow residuals and the fusion map, we use PReLU [13] as the activation function.

We compare the runtime of the state-of-the-art optical flow models [42, 43] and IFNet in Table 1. Current flow-based VFI methods [17, 3, 33] usually need to run their flow models twice then scale and reverse the bi-directional flows. Therefore the intermediate flow estimation in RIFE runs at a faster speed than previous methods, achieving an acceleration of about 6–15 times. Although these optical models can estimate inter-frame motion accurately, as previously noted, they are not suitable for direct migration to VFI tasks.

3.3. Priveleged Distillation for IFNet

Directly approximating the intermediate flows is challenging because of no access to the intermediate frame and the lack of supervision. To address this problem, we design a privileged distillation loss to IFNet. We stack an additional IFBlock (teacher model IFB^{Tea}, $K_{Tea} = 1$) that refines the results of IFNet referring to the target frame I_t^{GT}:

$$\begin{bmatrix} F^{Tea} \\ M^{Tea} \end{bmatrix} = \begin{bmatrix} F^n \\ M^n \end{bmatrix} + \text{IFB}^{Tea}\left(\begin{bmatrix} F^n \\ M^n \end{bmatrix}, t, \widehat{I}^n, I_t^{GT}\right). \quad (4)$$

With the access of I_t^{GT} as privileged information, the teacher model produces more accurate flows. We define the distillation loss $\mathcal{L}_{dis}$ as follows:

$$\mathcal{L}_{dis} = \sum_{i\in\{0,1\}} ||F_{t\to i} - F_{t\to i}^{Tea}||_1. \quad (5)$$

We apply the distillation loss over the full sequence of predictions generated from the iteratively updating process in the student model. The gradient of this loss will not be backpropagated to the teacher model. The teacher block will be discarded after the training phase, hence this would incur no extra cost for inference.

Inputs (Overlay) SepConv [34] DAIN [3] CAIN [9] RIFE (Ours) GT

Figure 5: **Qualitative comparison on Vimeo90K [48] testing set**. We cut out the objects and zoom in them [9]. While other methods cause various artifacts, our method produces good effects on the moving objects.

3.4. Implementation Details

Supervisions. Our training loss $\mathcal{L}$ is a linear combination of the reconstruction losses $\mathcal{L}_{rec}, \mathcal{L}_{rec}^{Tea}$ and privileged distillation loss $\mathcal{L}_{dis}$:

$$\mathcal{L} = \mathcal{L}_{rec} + \mathcal{L}_{rec}^{Tea} + \lambda_d \mathcal{L}_{dis}, \tag{6}$$

where we set $\lambda_d = 0.01$ to balance the scale of losses.

The reconstruction loss $\mathcal{L}_{rec}$ models the reconstruction quality of the intermediate frame. We denote the synthesized frame by $\widehat{I}_t$ and the ground-truth frame by I_t^{GT}.

The reconstruction loss has the formulation of :

$$\mathcal{L}_{rec} = d(\widehat{I}_t, I_t^{GT}), \mathcal{L}_{rec}^{Tea} = d(\widehat{I}_t^{Tea}, I_t^{GT}), \tag{7}$$

where d is often a pixel-wised loss. Following previous work [32, 33], we use $\mathcal{L}_1$ loss between two Laplacian pyramid representations of the reconstructed image and ground truth (denoted as $\mathcal{L}_{Lap}$).

Training dataset. We use the Vimeo90K dataset [48] to train RIFE. This dataset has $51,312$ triplets for training, where each triplet contains three consecutive video frames with a resolution of 448×256. We randomly augment the training data using horizontal and vertical flipping, temporal order reversing, and rotating by 90 degrees.

Training strategy. We train RIFE on the Vimeo90K training set and fix $t = 0.5$. RIFE is optimized by AdamW [27] with weight decay 10^{-4} for 300 epochs on 224×224 patches from the Vimeo90K training set. Our training uses a batch size of 64. We gradually reduce the learning rate from 3×10^{-4} to 3×10^{-5} using cosine annealing during the whole training process. We train RIFE on four TITAN X (Pascal) GPUs for about 15 hours.

Inspired by previous work [19, 7], we use the Vimeo90K-Septuplet [48] dataset to extend RIFE to support arbitrary-timestep frame interpolation. This dataset has $91,701$ sequence with a resolution of 448×256, each of which contains 7 consecutive frames. For each training sample, we randomly select 3 frames $(I_{n_0}, I_{n_1}, I_{n_2})$ and calculate the target timestep $t = (n_1 - n_0)/(n_2 - n_0)$, where $0 \leq n_0 < n_1 < n_2 < 7$. We keep other training setting unchanged and denote this model as RIFE$_m$.

4. Experiments

We first introduce the benchmarks for evaluation. Then we provide variants of our models with different computational costs. We compare these models with representative state-of-the-art methods, both quantitatively and visually. In addition, we show the capability of generating arbitrary-timestep frames and other applications using our models. An ablation study is carried out to analyze our design. Finally, we discuss some limitations of RIFE.

4.1. Benchmarks and Evaluation Metrics

We train our models on the Vimeo90K training dataset and directly test it on the following benchmarks.

Middlebury. The Middlebury-OTHER (M.B.) benchmark [2] is widely used to evaluate VFI methods. The image resolution in this dataset is around 640×480.

Vimeo90K. There are 3,782 triplets in the Vimeo90K testing set [48] with resolution of 448×256.

UCF101. The UCF101 dataset [41] contains videos with various human actions. There are 379 triplets with a resolution of 256×256.

HD. Bao *et al*. [4] collect 11 high-resolution videos for evaluation. The HD benchmark consists of four 1080p, three 720p and four 1280×544 videos. Following the author of this benchmark, we use the first 100 frames of each video for evaluation. We further use this benchmark for quantitative comparison of multiple frame interpolation.

We measure the peak signal-to-noise ratio (PSNR), structural similarity (SSIM), and interpolation error (IE) for quantitative evaluation. All the methods are tested on a TITAN X (Pascal) GPU. We calculate the average processing time for 100 runs after a warm-up process of 100 runs.

Table 2: **Quantitative comparisons on the UCF101, Vimeo90K, Middlebury-OTHER set, and HD benchmarks.** The images of each dataset are directly inputted to each model. Some models are unable to run on 1080p images due to exceeding the memory available on our graphics card (denoted as "OOM"). To report the runtime, we test all models for processing a pair of 640×480 images using the same device. **Bold** and underlined numbers represent the best and second-best performance. We use gray backgrounds to mark the methods that require pre-trained depth models or optical flow models.

Method	# Parameters (Million)	Runtime (ms)	UCF101 [41] PSNR	UCF101 [41] SSIM	Vimeo90K [48] PSNR	Vimeo90K [48] SSIM	M.B. [2] IE	HD [3] PSNR
DVF [48] (Vimeo90K)	1.6	80	34.92	0.968	34.56	0.973	2.47	31.47
Superslomo [17] (Vimeo90K)	19.8	62	35.15	0.968	34.64	0.974	2.21	31.55
SepConv [34]	21.6	51	34.78	0.967	33.79	0.970	2.27	30.87
TOFlow [2]	**1.1**	84	34.58	0.967	33.73	0.968	2.15	29.37
MEMC-Net [4]	70.3	401	35.01	0.968	34.29	0.970	2.12	31.39
DAIN [3]	24.0	436	35.00	0.968	34.71	0.976	2.04	31.64†
DSepConv [8]	21.8	236	35.08	0.969	34.73	0.974	2.03	OOM
CAIN [9]	42.8	38	34.98	0.969	34.65	0.973	2.28	31.77
SoftSplat [33]†	7.7	135	35.39	**0.970**	36.10	**0.980**	**1.81**	-
AdaCoF [21]	21.8	34	34.91	0.968	34.27	0.971	2.31	31.43
BMBC [36]	11.0	1580	35.15	0.969	35.01	0.976	2.04	OOM
CDFI [10]	5.0	198	35.21	0.969	35.17	0.977	1.98	OOM
EDSC [7]	8.9	46	35.13	0.968	34.84	0.975	2.02	31.59
RIFE	9.8	**16**	35.28	0.969	35.61	0.978	1.96	32.14
RIFE-Large (2T2R)	9.8	80	**35.41**	**0.970**	**36.13**	**0.980**	1.86	**32.32**

†: copy from the original papers.

Table 3: **Increase model complexity to achieve better quantitative results.**

Model Setting	RIFE	*2T*	*2R*	*2T2R*
UCF101 [41] PSNR	35.28	35.35	35.34	**35.41**
Vimeo90K [48] PSNR	35.61	35.76	35.98	**36.13**
M.B. [2] IE	1.96	1.93	1.89	**1.86**
HD [3] PSNR	32.14	32.26	32.16	**32.32**
# Parameters	9.8M	9.8M	9.8M	9.8M
Runtime*	**16ms**	25ms	42ms	80ms
Complexity*	**67G**	133G	263G	527G

*: on 640×480 frames

4.2. Test-Time Augmentation and Model Scaling

To provide models with different computation overheads that meet different needs, we introduce two modifications following: test-time augmentation and resolution multiplying. 1) We flip the input images horizontally and vertically to get augmented test data. Then we use the same model to infer some results and reverse the flipping on the results. We average these two results finally. Its effect is similar to training a model with twice the computational cost. This model is denoted as RIFE-*2T*. 2) We remove the first downsample layer of IFNet and add a downsample layer before its output to match the origin pipeline. We also perform this modification on RefineNet. It enlarges the process resolution of the feature maps and produces a model named RIFE-*2R*. We combine these two modifications to extend RIFE to RIFE-Large (*2T2R*). The performance and runtime for these models is reported in Table 3 and depicted in Figure 1.

4.3. Comparisons with Previous Methods

We compare RIFE with previous VFI models [48, 35, 4, 3, 9, 33, 36, 8, 21, 10, 7]. These models are officially released except SoftSplat. SoftSplat is not fully open-sourced and its results are reported by the origin authors. In addition, we train DVF [25] model and SuperSlomo [17] using our training pipeline on Vimeo90K dataset because the released models of these methods are trained on early datasets. We report the performance in Table 2.

RIFE runs considerably faster than other methods. Meanwhile, RIFE needs only 3.1 gigabytes of GPU memory to process 1080p videos, while some methods exceed 12 gigabytes. We get a larger version of our model (RIFE-Large) by model scaling and test-time augmentation, which runs about 1.7 times faster than the previous state-of-the-art method SoftSplat [33] with comparable performance. Compared with SoftSplat, RIFE does not require a specially designed forward warping operator or any pre-trained optical flow model. We provide a visual comparison of video clips with large motions from the Vimeo90K testing set in Figure 5, where SepConv [35] and DAIN [3] produce ghost-

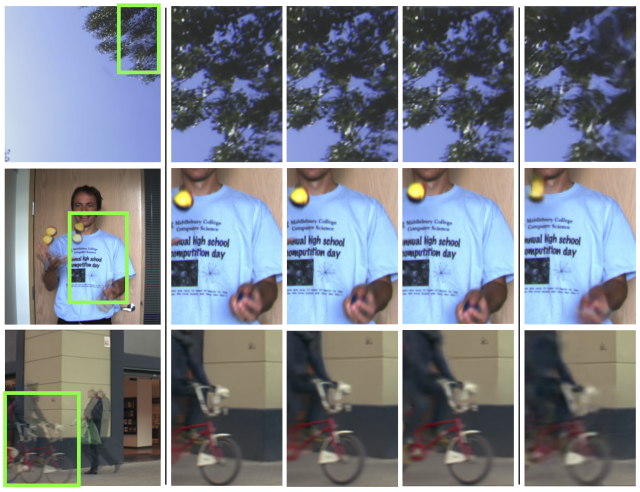


Figure 6: **Interpolating multiple frames using RIFE$_m$.** These images are from HD [3], M.B. [2], Vimeo90K [48] benchmarks, respectively. We attach the results of CAIN [9] for comparison. RIFE$_m$ provides smooth and continuous motions without artifacts.

ing artifacts, and CAIN [9] causes missing-parts artifacts. Overall, our method can produce more reliable results.

4.4. Generating Arbitrary-timestep Frame

We apply RIFE$_m$ to interpolate multiple intermediate frames at different timesteps $t \in (0, 1)$, as shown in Figure 6. It's worth noting that $t = 0.125$ is not included in the training data, and RIFE$_m$ can successfully handle it.

To provide a quantitative comparison of multiple frame interpolation, we further extract every fourth frame of videos from HD benchmark [4] and use them to interpolate other frames. We divide the HD benchmark into three subsets with different resolution to test these methods. We show the quantitative PSNR between generated frames and frames of the original videos in Table 4. Note that DAIN [3], BMBC [36] and EDSC$_m$ [8] can generate a frame at an arbitrary timestep. However, some other methods can only interpolate the intermediate frame at $t = 0.5$. So we use them recursively to produce $4\times$ results. Specifically, we firstly apply the single interpolation method once to get intermediate frame $\widehat{I}_{0.5}$. We then feed I_0 and $\widehat{I}_{0.5}$ to get $\widehat{I}_{0.25}$. RIFE$_m$ has more than twice the speed of CAIN [9] and improves the performance by 1.5 dB. Overall, RIFE$_m$ is very effective in the $4\times$ interpolation. Furthermore, since RIFE$_m$ can support arbitrary-timestep frame interpolation, it is suitable for non-multiplier frame rate conversion.

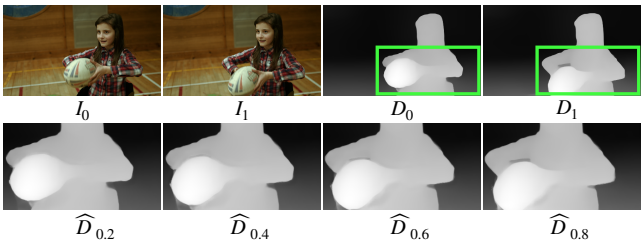


Figure 7: **Interpolating depth map using RIFE$_m$.** D_0 and D_1 are estimated by the MiDaS-large [38] model. RIFE$_m$ correctly maintains the occlusion relationship and object boundaries.

Table 4: **Quantitative evaluation for $4\times$ interpolation on the HD benchmark [4].** The 544p videos of HD benchmark are relatively more dynamic. Thus the PSNR index of these 544p videos is lower.

Method	Recursion	544p	720p	1080p
DAIN [3]	-	22.17	30.25	OOM
CAIN [9]	✓	21.81	31.59	31.08
BMBC [36]	-	19.51	23.47	OOM
DSepconv [8]	✓	19.28	23.48	OOM
CDFI [10]	✓	21.85	29.28	OOM
EDSC$_s$ [7]	✓	21.20	31.50	29.75
EDSC$_m$ [7]	-	21.89	30.35	30.39
RIFE$_m$	-	**22.95**	**31.87**	**34.25**

Image Representation Interpolation. RIFE$_m$ can interpolate other image representations using the intermediate flows and fusion map approximating from images. For instance, we interpolate the results of MiDaS [38] which is a popular monocular depth model, shown in Figure 7. The synthesis formula is simply as follows:

$$\widehat{D}_t = M \odot \overleftarrow{\mathcal{W}}(D_0, F_{t\to 0}) + (1-M) \odot \overleftarrow{\mathcal{W}}(D_1, F_{t\to 1}), \quad (8)$$

where D_0, D_1 are estimated by MiDas-large [38] and F, M are estimated by RIFE$_m$.

4.5. General Temporal Encode

To show the generalization capability of RIFE, we demonstrate that we can control the temporal encoding t to implement diverse applications. As shown in Figure 8, if we input a gradient encoding t_p, the RIFE$_m$ will synthesize the two images from dynamic scenes in a "panoramic" view. Similarly, this method may have the potential to eliminate the rolling shutter of the videos by having different timestamps for each horizontal row.

4.6. Ablation Studies

We design some ablation studies on the distillation scheme, intermediate flow estimation, and model design,

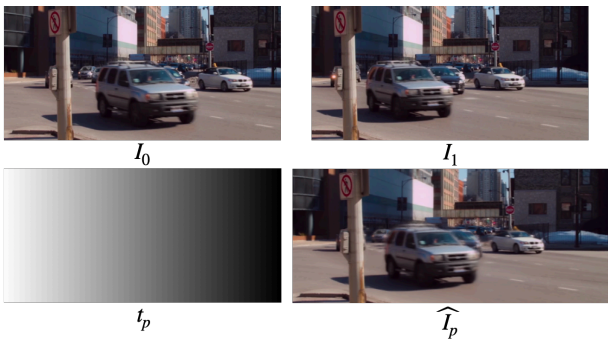


Figure 8: **Synthesize images from two views on one "panoramic" image $\widehat{I}_p$ using RIFE$_m$.** The temporal encoding t_p is gradually changed from 0 to 1 in columns. $\widehat{I}_p$ has been stretched for better visualization. Note that RIFE$_m$ produces smooth results in the dynamic scene.

shown in Table 5. These experiments use the same hyper-parameter setting and evaluation on Vimeo90K [48] and MiddleBury [2] benchmarks.

Ablation on the distillation scheme. We show the importance of our privileged distillation loss in three experiments. 1) Remove the whole scheme; 2) Remove the privileged teacher block and use the last IFBlock's results to guide the first two IFBlocks, denoted as "self-consistency"; 3) Use pre-trained RAFT [43] to estimate the intermediate flows based on the ground truth image, denoted as "RAFT-KD". This guidance is inspired by the pseudo-labels method [20]. However, this implementation relies on the pre-trained optical flow model and extremely increases the training duration. We found 1) leads to a large performance degradation while 2) and 3) suffer in quality. These experiments demonstrate the importance of optical flow supervision.

IFNet vs. flow reversal. We compare IFNet with previous intermediate flow estimation methods. Specifically, we use RAFT [43] and PWC-Net [42] with officially pre-trained parameters to estimate the bi-directional flows. Then we implement three flow reversal methods, including linear reversal [17], using a hidden convolutional layer with 128 channels, and the flow reversal method from EQVI [23] consisting of a reversal layer and a U-Net filter. The optical flow models and flow reversal modules are combined together to replace the IFNet. They are jointly trained with RefineNet. Because these models can not directly approximate the fusion map, the fusion map is subsequently approximated by RefineNet. As shown in Table 5, RIFE is more efficient and gets better interpolation performance. These flow models can estimate accurate bi-directional optical flow, but the flow reversal has difficulties in dealing with object shift problem illustrated in Figure 3.

Table 5: **Ablation study on distillation scheme, intermediate flow estimation, model design and loss function.**

Setting	Vimeo90K PSNR	M.B. IE	Runtime 640×480
Distillation Scheme			
RIFE w/o distillation	35.22	2.15	16ms
RIFE w/ self-consistency	35.38	2.02	16ms
RIFE w/ RAFT-KD	35.42	1.99	16ms
Intermediate Flow Estimation			
RAFT + linear reversal	34.68	2.31	60ms
RAFT + CNN reversal	34.82	2.24	65ms
RAFT + reversal layer	35.16	2.04	101ms
PWC-Net + reversal layer	35.24	2.06	83ms
Model Design and Loss Function			
RIFE w/ one IFBlock	35.22	2.13	**12ms**
RIFE w/ two IFBlocks	35.49	1.98	14ms
RIFE w/ $\mathcal{L}_1$	35.46	**1.94**	16ms
RIFE	**35.61**	1.96	16ms
(Priviledged Distillation, IFNet, three IFBlocks, $\mathcal{L}_{Lap}$)			

Ablation on RIFE's architecture and loss function. To verify the coarse-to-fine strategy of IFNet, we removed the first IFBlock and the first two IFBlocks in two experiments, respectively, shown in Table 5. We provide a pair of experiments to show $\mathcal{L}_{Lap}$ (default setting of RIFE) is quantitatively better than $\mathcal{L}_1$.

4.7. Limitations

Our work may not cover some practical application requirements. Firstly, RIFE focuses on only using two input frames and multi-frame input [47, 23, 19] is left to future work. One straightforward approach is to extend IFNet to use more frames as input. Secondly, most experiments are done with SSIM and PSNR as quantitative indexes. If human perception quality is preferred, RIFE can readily be changed to use the perceptually related loss functions, such as LPIPS [5, 34].

5. Conclusion

We develop an efficient and flexible algorithm for VFI, namely RIFE. A separate neural module IFNet directly estimates the intermediate optical flows, supervised by a privileged distillation scheme, where the teacher model can access ground truth intermediate frames. Experiments confirm RIFE can effectively process videos of different scenes. Furthermore, an extra input with temporal encoding enables RIFE for arbitrary-timestep frame interpolation and dynamic scene panorama stitching. The lightweight nature of RIFE makes it much more accessible for downstream tasks.

References

[1] Rohan Anil, Gabriel Pereyra, Alexandre Passos, Robert Ormandi, George E Dahl, and Geoffrey E Hinton. Large scale distributed neural network training through online distillation. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2018. 3

[2] Simon Baker, Daniel Scharstein, JP Lewis, Stefan Roth, Michael J Black, and Richard Szeliski. A database and evaluation methodology for optical flow. *International journal of computer vision (IJCV)*, 92(1):1–31, 2011. 5, 6, 7, 8, 12

[3] 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 (CVPR)*, 2019. 1, 2, 3, 4, 5, 6, 7

[4] 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. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2018. 5, 6, 7

[5] Yochai Blau and Tomer Michaeli. The perception-distortion tradeoff. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 8

[6] Xinghao Chen, Yiman Zhang, Yunhe Wang, Han Shu, Chunjing Xu, and Chang Xu. Optical flow distillation: Towards efficient and stable video style transfer. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020. 3

[7] Xianhang Cheng and Zhenzhong Chen. Multiple video frame interpolation via enhanced deformable separable convolution. *arXiv preprint arXiv:2006.08070*, 2020. 2, 5, 6, 7, 12

[8] Xianhang Cheng and Zhenzhong Chen. Video frame interpolation via deformable separable convolution. In *AAAI Conference on Artificial Intelligence*, 2020. 2, 6, 7

[9] Myungsub Choi, Heewon Kim, Bohyung Han, Ning Xu, and Kyoung Mu Lee. Channel attention is all you need for video frame interpolation. In *AAAI Conference on Artificial Intelligence*, 2020. 2, 5, 6, 7

[10] Tianyu Ding, Luming Liang, Zhihui Zhu, and Ilya Zharkov. Cdfi: Compression-driven network design for frame interpolation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 6, 7

[11] Xiaohan Ding, Xiangyu Zhang, Ningning Ma, Jungong Han, Guiguang Ding, and Jian Sun. Repvgg: Making vgg-style convnets great again. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 2

[12] Alexey Dosovitskiy, Philipp Fischer, Eddy Ilg, Philip Hausser, Caner Hazirbas, Vladimir Golkov, Patrick Van Der Smagt, Daniel Cremers, and Thomas Brox. Flownet: Learning optical flow with convolutional networks. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2015. 2

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2015. 4

[14] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015. 3

[15] Tak-Wai Hui, Xiaoou Tang, and Chen Change Loy. Liteflownet: A lightweight convolutional neural network for optical flow estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 2

[16] 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 *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 2

[17] Huaizu Jiang, Deqing Sun, Varun Jampani, Ming-Hsuan Yang, Erik Learned-Miller, and Jan Kautz. Super slomo: High quality estimation of multiple intermediate frames for video interpolation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 1, 2, 3, 4, 6, 8, 11

[18] Rico Jonschkowski, Austin Stone, Jonathan T Barron, Ariel Gordon, Kurt Konolige, and Anelia Angelova. What matters in unsupervised optical flow. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020. 2

[19] Tarun Kalluri, Deepak Pathak, Manmohan Chandraker, and Du Tran. Flavr: Flow-agnostic video representations for fast frame interpolation. *arXiv preprint arXiv:2012.08512*, 2020. 5, 8

[20] Dong-Hyun Lee et al. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *Proceedings of the IEEE International Conference on Machine Learning Workshop (ICMLW)*, 2013. 8

[21] 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 Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 1, 2, 6

[22] Wonkyung Lee, Junghyup Lee, Dohyung Kim, and Bumsub Ham. Learning with privileged information for efficient image super-resolution. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020. 3

[23] Yihao Liu, Liangbin Xie, Li Siyao, Wenxiu Sun, Yu Qiao, and Chao Dong. Enhanced quadratic video interpolation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020. 1, 2, 3, 4, 8

[24] Yu-Lun Liu, Yi-Tung Liao, Yen-Yu Lin, and Yung-Yu Chuang. Deep video frame interpolation using cyclic frame generation. In *AAAI Conference on Artificial Intelligence*, 2019. 2

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

[26] David Lopez-Paz, Léon Bottou, Bernhard Schölkopf, and Vladimir Vapnik. Unifying distillation and privileged information. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2016. 3

[27] Ilya Loshchilov and F. Hutter. Fixing weight decay regularization in adam. *arXiv preprint arXiv:1711.05101*, 2017. 5

[28] Kunming Luo, Chuan Wang, Shuaicheng Liu, Haoqiang Fan, Jue Wang, and Jian Sun. Upflow: Upsampling pyramid for unsupervised optical flow learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 2

[29] Simon Meister, Junhwa Hur, and Stefan Roth. UnFlow: Unsupervised learning of optical flow with a bidirectional census loss. In *AAAI Conference on Artificial Intelligence*, 2018. 2

[30] Simone Meyer, Abdelaziz Djelouah, Brian McWilliams, Alexander Sorkine-Hornung, Markus Gross, and Christopher Schroers. Phasenet for video frame interpolation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 2

[31] Simone Meyer, Oliver Wang, Henning Zimmer, Max Grosse, and Alexander Sorkine-Hornung. Phase-based frame interpolation for video. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 2

[32] Simon Niklaus and Feng Liu. Context-aware synthesis for video frame interpolation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 1, 5

[33] Simon Niklaus and Feng Liu. Softmax splatting for video frame interpolation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 2, 3, 4, 5, 6, 11

[34] Simon Niklaus, Long Mai, and Feng Liu. Video frame interpolation via adaptive convolution. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 2, 5, 6, 8

[35] Simon Niklaus, Long Mai, and Feng Liu. Video frame interpolation via adaptive separable convolution. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2017. 2, 6

[36] Junheum Park, Keunsoo Ko, Chul Lee, and Chang-Su Kim. Bmbc: Bilateral motion estimation with bilateral cost volume for video interpolation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020. 1, 2, 6, 7

[37] Angelo Porrello, Luca Bergamini, and Simone Calderara. Robust re-identification by multiple views knowledge distillation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020. 3

[38] René Ranftl, Katrin Lasinger, David Hafner, Konrad Schindler, and Vladlen Koltun. Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2020. 7

[39] Fitsum A Reda, Deqing Sun, Aysegul Dundar, Mohammad Shoeybi, Guilin Liu, Kevin J Shih, Andrew Tao, Jan Kautz, and Bryan Catanzaro. Unsupervised video interpolation using cycle consistency. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2019. 2

[40] 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 (MICCAI)*, 2015. 2

[41] 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. 5, 6

[42] Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz. Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 2, 4, 8

[43] Zachary Teed and Jia Deng. Raft: Recurrent all-pairs field transforms for optical flow. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020. 2, 4, 8

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

[45] Xiaoyu Xiang, Yapeng Tian, Yulun Zhang, Yun Fu, Jan P Allebach, and Chenliang Xu. Zooming slow-mo: Fast and accurate one-stage space-time video super-resolution. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 2

[46] Gang Xu, Jun Xu, Zhen Li, Liang Wang, Xing Sun, and Ming-Ming Cheng. Temporal modulation network for controllable space-time video super-resolution. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 2

[47] Xiangyu Xu, Li Siyao, Wenxiu Sun, Qian Yin, and Ming-Hsuan Yang. Quadratic video interpolation. In *Advances in Neural Information Processing Systems (NIPS)*, 2019. 1, 2, 3, 4, 8

[48] Tianfan Xue, Baian Chen, Jiajun Wu, Donglai Wei, and William T Freeman. Video enhancement with task-oriented flow. *International Journal of Computer Vision (IJCV)*, 127(8):1106–1125, 2019. 1, 5, 6, 7, 8

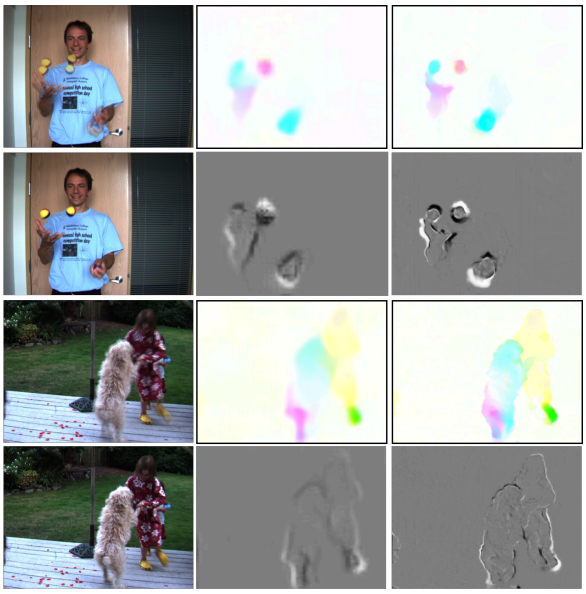
[49] Shanxin Yuan, Bjorn Stenger, and Tae-Kyun Kim. Rgb-based 3d hand pose estimation via privileged learning with depth images. In *Proceedings of the IEEE International Conference on Computer Vision Workshop (ICCVW)*, 2019. 3

6. Appendix

6.1. Architecture of RefineNet

Following the previous work [33], we design a RefineNet with an encoder-decoder architecture similar to U-Net and a context extractor. The context extractor and encoder part have similar architectures, consisting of four convolutional blocks, and each of them is composed of two 3×3 convolutional layers, respectively. The decoder part in the FusionNet has four transpose convolution layers.

Specifically, the context extractor first extracts the pyramid contextual features from input frames separately. We denote the pyramid contextual feature as C_0: $\{C_0^0, C_0^1, C_0^2, C_0^3\}$ and C_1: $\{C_1^0, C_1^1, C_1^2, C_1^3\}$. We then perform backward warping on these features using estimated intermediate flows to produce aligned pyramid features, $C_{t\leftarrow 0}$ and $C_{t\leftarrow 1}$. The origin frames I_0, I_1, warped frames $\widehat{I}_{t\leftarrow 0}, \widehat{I}_{t\leftarrow 1}$, intermediate flows $F_{t\to 0}, F_{t\to 1}$ and fusion mask M are fed into the encoder. The output of $i-th$ encoder block is concatenated with the $C_{t\leftarrow 0}^i$ and $C_{t\leftarrow 1}^i$ before being fed into the next block. The decoder parts finally produce a reconstruction residual Δ. And we will get a refined reconstructed image $\widehat{I}_t + \Delta$, where $\widehat{I}_t$ is the reconstruct image before the RefineNet. We show some visualization results in Figure 12. RefineNet seems to make some uncertain areas more blurred to improve quantitative results.



Inputs and Ground Truth 1st IFBlock IFNet

Figure 9: **Visualization of intermediate flow $F_{t\to 0}$ and blend mask M.** We show that stack 3 IFblocks can get finer intermediate flow and blend mask.

6.2. Intermediate Flow Visualization

In Figure 10, we provide visual results of our IFNet and compare them with the linearly combined bi-directional optical flows [17]. IFNet produces clear motion boundaries.

Inputs (Overlay) Combination IFNet

Figure 10: **Visual comparison between linearly combined bi-directional flows [17] and the result of IFNet.**

6.3. Training Dynamic

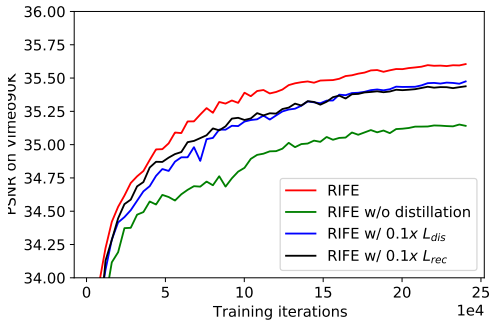


Figure 11: **PSNR on Vimeo90K benchmark during the whole training process.** The distillation scheme helps RIFE converge to better performance

We study the dynamic during the RIFE training. As shown in Figure 11, the privileged distillation scheme helps RIFE converge to better performance. Furthermore, we try to adjust the weights of losses. We found that larger scale $(10\times)$ of weights will cause the model to not converge and smaller weights $(0.1\times)$ will slightly reduce model performance.

6.4. Model Efficiency Comparison

The speed of models is very important in frame interpolation applications. However, to the best of our knowledge, currently published papers do not test the speed of each

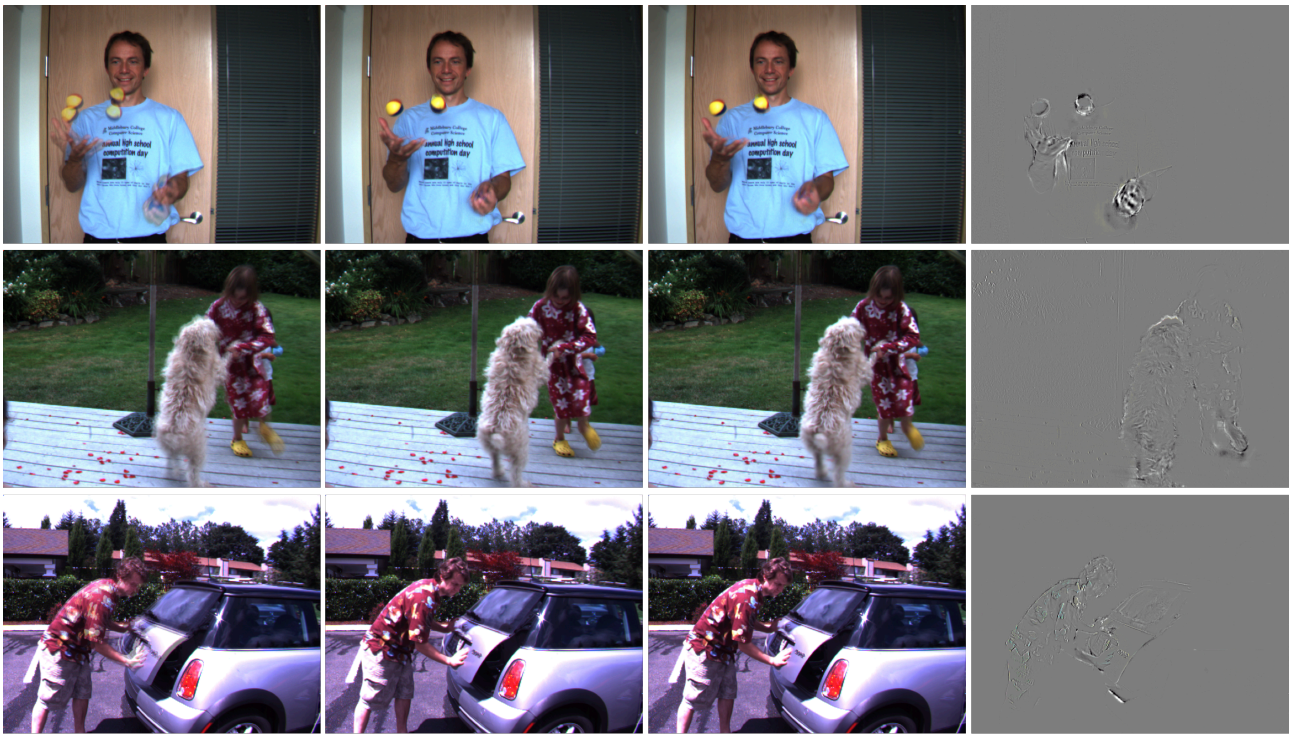


Figure 12: **Interpolating results on the MiddleBury [2] dataset.** We show that the function of RefineNet is mainly to refine the high frequency content of the results.

state-of-the-art VFI model on same hardware, and rarely report the complexity of the model. Some previous works report runtime data from the MiddleBury public leaderboard without indicating running devices. These data are reported by the submitters of various methods. A more reliable survey comes from EDSC (Table 10) [7]. We collect the models of each paper and test them on a NVIDIA TITAN X(Pascal) GPU with same environment.