Test-Time Adaptation for Video Frame Interpolation via Meta-Learning

Myungsub Choi, *Member, IEEE,* Janghoon Choi, *Member, IEEE,* Sungyong Baik, *Student Member, IEEE,* Tae Hyun Kim, *Member, IEEE,* and Kyoung Mu Lee, *Fellow, IEEE,*

Abstract—Video frame interpolation is a challenging problem that involves various scenarios depending on the variety of foreground and background motions, frame rate, and occlusion. Therefore, generalizing across different scenes is difficult for a single network with fixed parameters. Ideally, one could have a different network for each scenario, but this will be computationally infeasible for practical applications. In this work, we propose *MetaVFI*, an adaptive video frame interpolation algorithm that uses additional information readily available at test time but has not been exploited in previous works. We initially show the benefits of *test-time adaptation* through simple fine-tuning of a network and then greatly improve its efficiency by incorporating meta-learning. Thus, we obtain significant performance gains with only a single gradient update without introducing any additional parameters. Moreover, the proposed MetaVFI algorithm is model-agnostic which can be easily combined with any video frame interpolation network. We show that our adaptive framework greatly improves the performance of baseline video frame interpolation networks on multiple benchmark datasets.

Index Terms—Video frame interpolation, test-time adaptation, meta-learning, slow motion, self-supervision, image synthesis, MAML

1 Introduction

Video frame interpolation aims to upscale the temporal resolution of a video by synthesizing intermediate frames in-between the neighboring frames of the original input video. Owing to its wide range of applications, including slow-motion generation and frame-rate up-conversion to provide better visual experiences with more details and less motion blur, video frame interpolation has gained substantial interest in the computer vision community. Recent advances of deep convolutional neural networks (CNNs) for video frame interpolation [1], [2], [3], [4], [5], [6], [7] have led to a significant performance boost. However, generating high-quality frames is still a challenging problem due to large motion and heavy occlusion in a diverse set of scenes.

Previous video frame interpolation approaches [1], [2], [3], [4], [5], [6], [7], as well as other learning-based video processing models [8], [9], [10], [11], [12], typically require a large amount of data. However, videos in the wild comprise various scenes with many different types of visual patterns, including low-level patterns, such as noise or blurs, to different types of high-level motion. Hence, a single pre-trained model can hardly perform well on all possible test cases, even if trained with a large dataset.

This problem can be alleviated by making the model adaptive to the specific input data. Utilizing the additional information available at test time and customizing the model to each of the test (input) data samples has shown to be effective in numerous areas. Examples include single-image super-resolution approaches exploiting self-similarities inherent in the target image [13], [14], [15], [16], [17], [18], [19], or many visual tracking methods where online adaptation is crucial for performance [20], [21], [22]. However, most works either increase the number of trainable parameters or require a significant amount of extra inference time for test-time adaptation of the network parameters.

Meta-learning, also known as *learning to learn*, can take a step forward to remedy the current limitations in test-time adaptation. The goal of meta-learning is to design algorithms or models that can quickly adapt to new tasks from a small set of training examples given during the testing phase. Meta-learning has been gaining tremendous interest in solving few-shot classification and regression problems as well as some reinforcement learning applications [23]. However, employing meta-learning techniques for low-level computer vision problems has yet to be fully explored.

To this end, we propose *MetaVFI*, a scene-adaptive video frame interpolation algorithm that can rapidly adapt to new, unseen videos (or tasks from a meta-learning viewpoint) at test time and achieve substantial performance gain. Fig. 1 shows a brief overview of our approach. Using any off-the-shelf existing video frame interpolation framework, our algorithm updates its parameters using the frames only available at test time. Then, the adapted model is used to interpolate the intermediate frames in the same way as the conventional approaches. In this way, the model can adapt to the dominant types of motion and texture in the current input video sequence. The adapted model can then reason for the difficult motion and occlusion as well as the textural details significantly better than the original frame interpolation model.

Overall, our contributions are summarized as follows:

- We propose MetaVFI, a novel adaptation framework that can further improve conventional frame interpolation models without changing their architectures.
- To the best of our knowledge, the proposed approach is the first integration of meta-learning techniques for test-time adaptation in video frame interpolation.

- *M. Choi is with Google Research. This work is done while he was at Seoul National University, Seoul, South Korea. E-mail: cms6539@gmail.com*
- *J. Choi is with Kookmin University. This work is done while he was at Seoul National University, Seoul, South Korea. E-mail: jhchoi09@kookmin.ac.kr*
- *S. Baik, and K. M. Lee are with Automation and Systems Research Institute (ASRI), Department of Electrical and Computer Engineering, Seoul National University, Seoul, South Korea. E-mail: {dsybaik, kyoungmu}@snu.ac.kr*
- *T. H. Kim is with the Department of Computer Science, Hanyang University, Seoul, South Korea. E-mail: taehyunkim@hanyang.ac.kr*

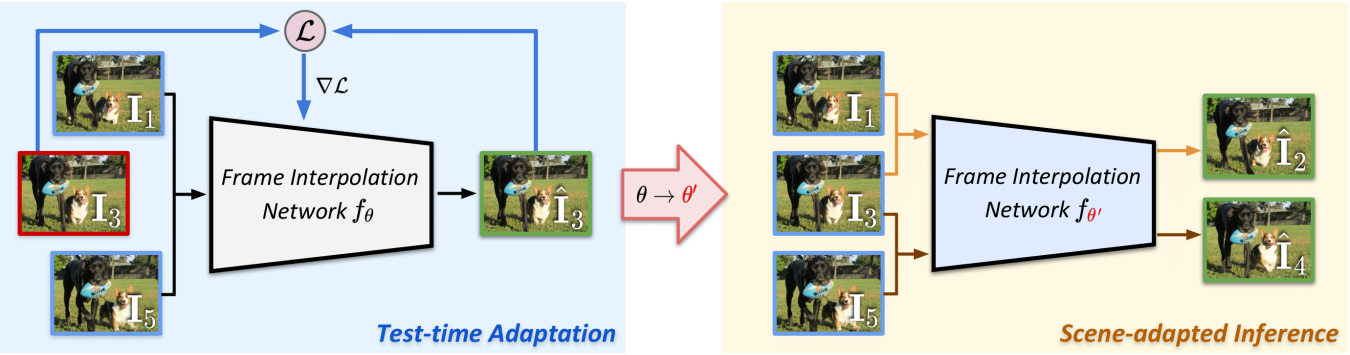

Fig. 1. **Motivation of the proposed adaptive frame interpolation method.** Our video frame interpolation framework incorporates a test-time adaptation process followed by scene-adapted inference. The adaptation process utilizes the additional information from the input frames at test-time, and is quickly performed with only a single gradient update to the network parameters.

- We confirm that our framework consistently improves upon many recent state-of-the-art methods.
- We extend our previous work [24] to greatly expand the experimental results and analyses. Notably, we include comparisons among different meta-learning algorithms (Sec. 4.2) and show quantitative/qualitative results for more recent state-of-the-art frame interpolation models (Sec. 4.3). Moreover, the reasons for the performance improvement are discussed rigorously (Sec. 4.4).

The rest of the sections are organized as follows. We first review related works in Section 2, and present our model in Section 3. Experimental results are shown in Section 4, with extensive ablation study and visualizations.

2 Related Works

In this section, we review the extensive literature of video frame interpolation. Existing test-time adaptation schemes for other low-level vision applications and the history of meta-learning algorithms are also described.

Video frame interpolation: Although video frame interpolation has a long-established history, we focus on more recent deep-learning-based algorithms, particularly CNN-based approaches.

The first attempt to incorporate CNNs to video frame interpolation was done by Long *et al.* [25], where interpolation is obtained as a byproduct of self-supervised learning of optical flow estimation. Since then, numerous approaches have focused on effectively modeling motion and handling occlusions. These methods include representing motion as per-pixel phase shift [26], [27], modeling the sequential process of motion estimation and frame synthesis into a single spatially-adaptive convolution step [5], [6], [28], or direct frame synthesis with simple feedforward networks and channel attention/gating [1], [29].

Another line of research uses optical flow estimation as an intermediate step [2], [3], [4], [7], [30], [31], [32], [33], [34], [35], [36], [37], [38]. Using the estimated motion map, the original input frames are warped to the intermediate time step for alignment, followed by further refinement and occlusion handling steps to obtain the final interpolations. These flow-based models are generally able to synthesize sharp and natural frames, but some models heavily depend on the pretrained optical flow estimation network and show ghost artifacts in cases with large motion when flow estimation fails. Many novel ideas have been proposed to compensate for the errors in flow estimation, such as additionally using a depth map estimation model [30], refining the estimated bi-directional flow maps [32], designing a new bilateral cost volume layer for better flow estimation [35], or using an additional supporting frame for better motion reasoning [38]. On the other hand, Niklaus *et al.* [34] focused on the splatting operation and proposed softmax splatting, showing impressive results.

Test-time adaptation: Contrary to previous works, we explore an orthogonal area of research, adaptation to the inputs at test time, to further improve the accuracy of given video frame interpolation models. Our work is inspired by the success of self-similarity-based approaches in image super-resolution [13], [14], [15], [16], [17], [18], [19]. Notably, recent zero-shot super-resolution (ZSSR) method [17] has shown impressive results by incorporating deep learning. Specifically, ZSSR at test time extracts the patches only from the input image and trains a small image-specific CNN, thereby naturally exploiting the information that is only available after observing the test inputs. However, ZSSR suffers from slow inference time due to its self-training step, and it is prone to overfitting because using a pretrained network trained with large external datasets is not viable for internal training. Recently, the efficiency issue for ZSSR has been greatly alleviated by incorporating meta-learning [18], [19], which are concurrent works that share a similar philosophy to the proposed method [24].

For video frame interpolation, Reda *et al.* [39] proposed the first approach to adapt to the test data in an unsupervised manner by using a cycle-consistency constraint. However, their method adapts to the general domain of the test data, and cannot adapt to each test sample. On the other hand, the proposed algorithm allows for updating the model parameters *w.r.t.* each local part of the test sequence, thus better adapting to local motions and scene textures in a video.

Meta-learning: Recently, meta-learning has drawn much attention for its capability to adapt to new tasks with only a few examples. Such an adaptation ability is made possible through learning the prior knowledge shared across a distribution of tasks [40], [41], [42], [43], [44]. Viewing tasks as individual videos, we find the adaptation capability to be a compelling motivation for applying meta-learning to test-time adaptation in video frame interpolation.

In general, there are three major categories of meta-learning algorithms, namely, metric-based, network-based, and optimization (or gradient)-based algorithms. In metric-based algorithms, the prior knowledge is often encoded by learning a feature embedding space [45], [46], [47], [48]. In contrast, network-based meta-learners achieve more flexibility by encoding the prior knowledge into the architecture of a neural network [49], [50], [51], [52]. However, the metric or network-based systems have limitations in either applications or scalability issues. On the other hand, the optimization-based meta-learners regulate the optimization

algorithm (or weight-update rule) for the adaptation process to encode the prior knowledge. Model-Agnostic Meta-Learning (MAML) [23] is one of the most representative optimization-based learners for its simplicity and model-agnostic property. MAML encodes the prior knowledge into the initialization of neural network parameters and thus does not require extra parameters. Such a simple and model-agnostic design of MAML motivates us to hinge the test-time adaptation framework on MAML algorithm.

3 Proposed Method

In this section, we describe the general problem settings for video frame interpolation. Then, we show the advantage of test-time adaptation empirically with a feasibility test, and justify the need for meta-learning in this scenario.

3.1 Video frame interpolation problem set-up

Video frame interpolation algorithms aim to generate a high-quality, high frame-rate video given a low frame-rate input video by synthesizing the intermediate frames between the two neighboring frames. A conventional setting for most frame interpolation models receives two input frames and outputs a single intermediate frame. Specifically, if we let $\mathbf{I}_1$ and $\mathbf{I}_3$ be the two consecutive input frames, our goal is to synthesize the middle frame $\hat{\mathbf{I}}_2$. Given the true middle frame as $\mathbf{I}_2$, we can define the base unit of the training data as a frame triplet, $(\mathbf{I}_1, \mathbf{I}_2, \mathbf{I}_3)$. Recent frame interpolation models also consider more complex problem settings including synthesizing an arbitrary intermediate time step or multi-frame interpolation, but we constrain our discussions to the single middle-frame interpolation models in this work. However, note that our proposed meta-learning framework described in Sec. 3.4 is model-agnostic and can be easily generalized to different settings as long as the model is differentiable.

3.2 Exploiting extra information at test time

We demonstrate the effectiveness of test-time adaptation with a feasibility test and describe the details on our design choices. This section aims to verify whether adapting to the input low frame-rate videos is beneficial to the interpolation performance. To this end, we start from a baseline pre-trained video frame interpolation model and fine-tune its parameters using only the test-time inputs (separately for each test video sequence). Specifically, we adapt to four input frames, denoted as $\{\mathbf{I}_1, \mathbf{I}_3, \mathbf{I}_5, \mathbf{I}_7\}$, and evaluate the performance of interpolating the middle frame, $\mathbf{I}_4$. Using the triplet notation described in Sec. 3.1, two triplets, $(\mathbf{I}_1, \mathbf{I}_3, \mathbf{I}_5)$ and $(\mathbf{I}_3, \mathbf{I}_5, \mathbf{I}_7)$, are constructed from the input frames for adaptation. Then, we aim to achieve good performance for the triplet $(\mathbf{I}_3, \mathbf{I}_4, \mathbf{I}_5)$. If fine-tuning with the test-time input triplets improves the performance of our intended estimation $\hat{\mathbf{I}}_4$, then we can conclude that test-time adaptation is advantageous, as shown in the following results of our feasibility test.

The main reason for checking the feasibility is because the frame triplets used for adaptation, $(\mathbf{I}_1, \mathbf{I}_3, \mathbf{I}_5)$ and $(\mathbf{I}_3, \mathbf{I}_5, \mathbf{I}_7)$, have some characteristics that are different from the triplet used for the final inference, $(\mathbf{I}_3, \mathbf{I}_4, \mathbf{I}_5)$. Notably, the time gap between frames is wider, since only the low frame-rate video is available at test time. The larger gap introduces larger motion and heavier occlusion, making the adaptation problem more difficult than our original goal. Nevertheless, the result of our feasibility test has shown that adapting to the harder data can still boost the overall

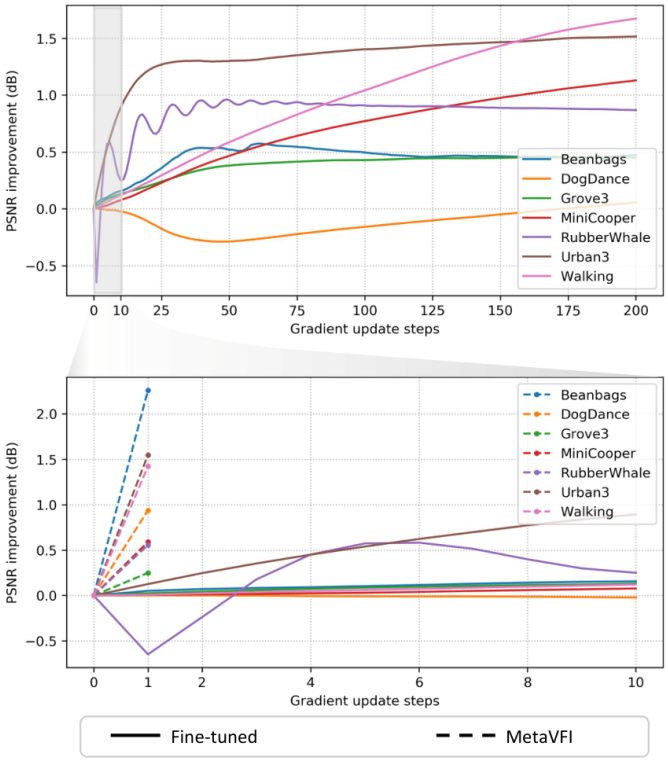

Fig. 2. **Feasibility test for test-time adaptation.** The upper graph shows that fine-tuning with the test input data can improve performance in general, but the number of required steps greatly differs for each sequence. The lower graph shows a zoomed-in version of the shaded region in the upper graph, additionally denoting the large performance gain obtained with SepConv+MetaVFI with a single gradient update. The solid lines denote the fine-tuned result of our feasibility test, and the dotted lines show the results with the proposed MetaVFI algorithm.

interpolation performance. This result implies the importance of the context and attributes of the given video scene, such as its unique motion, and signifies the benefit of test-time adaptation.

In practice, we fine-tune a pre-trained SepConv [6] model on each sequence from Middlebury-OTHERS [53] dataset. For each sequence, we build two triplets from the four input frames and fine-tune the baseline model repeatedly up to 200 iterations. We measure the peak signal-to-noise ratio (PSNR) for performance evaluation, and Fig. 2 depicts the resulting changes with respect to the number of gradient updates (solid lines).

The characteristics for performance improvements, shown in the upper graph of Fig. 2, greatly differs from sequence to sequence. The PSNR scores for *Minicooper* and *Walking* steadily improve for 200 gradient updates and do not overfit even after over 1dB gain. On the contrary, updating with *DogDance* sequence negatively affects the performance of the original model in its early stage. Notably, the graph for *RubberWhale* shows a strange characteristic, where the performance severely drops after the first gradient update but suddenly shifts back to the positive side after the subsequent steps. From these results, we can arguably conclude that test-time adaptation is beneficial for video frame interpolation. However, the problem of deciding how much to adapt (or not adapt at all) for different sequences still remains.

The proposed method can enhance the original SepConv model by incorporating meta-learning techniques to rapidly adapt to the test sequence, without changing any architectural structures or introducing additional parameters. With just a single gradient update at test time, our SepConv+MetaVFI can achieve large performance gain, as illustrated in the lower graph of Fig. 2 (we use the

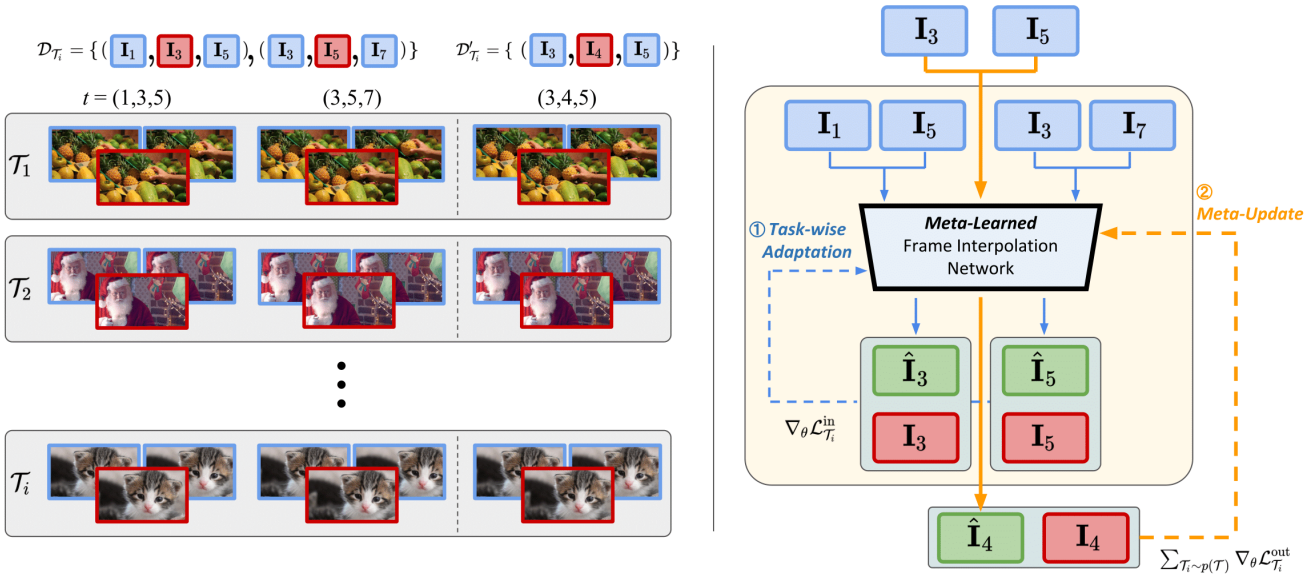

Fig. 3. **Overview of the training process for the proposed video frame interpolation network. *Left***: Each task $\mathcal{T}_i$ consists of three frame triplets chosen from a video sequence where two are used for task-wise adaptation (*i.e.*, inner-loop update) and one is used for meta-update (*i.e.*, outer-loop update). ***Right***: Network parameters θ are adapted by gradient descent on loss $\mathcal{L}^{\text{in}}_{\mathcal{T}_i}$ using triplets in $\mathcal{D}_{\mathcal{T}_i}$ and stored for each task. Meta-update is performed by minimizing the sum of each loss $\mathcal{L}^{\text{out}}_{\mathcal{T}_i}$ using the triplets in $\mathcal{D}'_{\mathcal{T}_i}$ for all tasks.

first-order variant of MAML [23] for meta-learning algorithm; see Sec. 4.2 for further discussions). Compared with hundreds of iterations required for fine-tuning the baseline model, SepConv+MetaVFI extremely reduces the computation time needed to obtain the same amount of performance boost.

3.3 Background on MAML

Meta-learning aims to quickly adapt to novel tasks with only a few examples. One of the most representative meta-learning algorithms, MAML [23], attempts to achieve this goal with only a few gradient update iterations by preparing the model to be readily adaptable to incoming test data. In other words, MAML finds a good initialization of the parameters that are responsive to task changes, so that small updates can lead to large improvements in performance for each new task. Before diving into the proposed method, we provide an overview of the formulation of MAML.

Under the assumption of the existence of task distribution, $p(\mathcal{T})$, MAML learns the initialization parameters of a network to encode the prior knowledge across the task distribution. In the setting of k-shot learning, a set of k number of examples $\mathcal{D}_{\mathcal{T}_i}$ are sampled from each task $\mathcal{T}_i \sim p(\mathcal{T})$. The sampled examples, along with its corresponding loss $\mathcal{L}_{\mathcal{T}_i}$, roughly represent the task $\mathcal{T}_i$ and are used for the model to adapt to the task. In MAML, this adaptation is achieved by fine-tuning:

$$\theta'_i = \theta - \alpha \nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_\theta). \tag{1}$$

Once the model is adapted to each task $\mathcal{T}_i$, new unseen examples $\mathcal{D}'_{\mathcal{T}_i}$ are sampled from the same task to evaluate the generalization of the adapted model. The evaluation of adapted models acts as a feedback for MAML to adjust its initialization parameters to achieve better generalization for tasks. This evaluation is conducted by minimizing the total loss of all tasks:

$$\theta \leftarrow \theta - \beta \nabla_\theta \sum_{\mathcal{T}_i} \mathcal{L}_{\mathcal{T}_i}(f_{\theta'_i}). \tag{2}$$

In the following section, we assume MAML as the representative meta-learning algorithm incorporated in the proposed framework. However, note that more advanced MAML-based meta-learning algorithms exist, such as MAML++ [54] and Meta-SGD [55], which are also easy to be plugged into our framework. We have evaluated them, and the results are analyzed in Sec. 4.2.

3.4 MetaVFI: Meta-learning for frame interpolation

For video frame interpolation, we define a *task* as performing frame interpolation on a sequence of frames (video). Fast adaptation to new video scenes via MAML introduces our MetaVFI algorithm, which is described in detail later in this section.

We consider a frame interpolation model f_θ, parameterized by θ. This model receives two input frames $(\mathbf{I}_t, \mathbf{I}_{t+2T})$ and outputs the estimated middle frame $\hat{\mathbf{I}}_{t+T}$ for any time step t and interval T. Thus, a training sample needed to update the model parameters can be formalized as a frame triplet $(\mathbf{I}_t, \mathbf{I}_{t+T}, \mathbf{I}_{t+2T})$. We define a task $\mathcal{T}$ as minimizing the sum of the losses $\mathcal{L} : \{(\mathbf{I}_t, \mathbf{I}_{t+T}, \mathbf{I}_{t+2T})\} \rightarrow \mathbb{R}$ for all time steps t in low frame-rate input video. In our scene-adaptive frame interpolation setting, each new task $\mathcal{T}_i$ drawn from $p(\mathcal{T})$ consists of frames in a single sequence, and the model is adapted to the task using a task-wise training set $\mathcal{D}_{\mathcal{T}_i}$, where the training triplets are constructed only with frames existent in the low frame-rate input. Updating the parameters at the meta-training stage is governed by the loss $\mathcal{L}^{\text{out}}_{\mathcal{T}_i}$ for a task-wise test set $\mathcal{D}'_{\mathcal{T}_i}$, where the test triplets consist of two input frames and the target ground-truth intermediate frame that is non-existent in the low frame-rate input. Let us assume $t = T = 1$ to reduce notation clutter. In practice, we use four input frames $\{\mathbf{I}_1, \mathbf{I}_3, \mathbf{I}_5, \mathbf{I}_7\}$ as described in Sec. 3.2 and a single target middle frame $\mathbf{I}_4$. The task-wise training and test set then become $\mathcal{D}_{\mathcal{T}_i} = \{(\mathbf{I}_1, \mathbf{I}_3, \mathbf{I}_5), (\mathbf{I}_3, \mathbf{I}_5, \mathbf{I}_7)\}$ and $\mathcal{D}'_{\mathcal{T}_i} = \{(\mathbf{I}_3, \mathbf{I}_4, \mathbf{I}_5)\}$. Fig. 3 depicts these configurations (left part).

Given the above notations, we now describe the flow of our MetaVFI algorithm in detail. Since our method is model-agnostic due to integration with MAML, we can use any existing video frame interpolation model as a baseline. However, unlike MAML where the model parameters begin from random initialization, we initialize the model parameters from a pre-trained model that is

already capable of generating sensible interpolations. Thus, our algorithm can be also viewed as a post-processing step, where the baseline model is updated to be readily adaptive to each test video for further performance boost.

Training. The detailed flow for training the proposed framework is illustrated in the right part of Fig. 3. The update iterations for each task are denoted as *inner-loop* and the meta-update iterations as *outer-loop*. For inner-loop training, given the two frame triplets from the task-wise training set $\mathcal{D}_{\mathcal{T}_i}$ for each task $\mathcal{T}_i$, we first calculate the model predictions as follows:

$$\hat{\mathbf{I}}_3 = f_\theta(\mathbf{I}_1, \mathbf{I}_5), \quad \hat{\mathbf{I}}_5 = f_\theta(\mathbf{I}_3, \mathbf{I}_7). \tag{3}$$

These outputs are then used to compute the loss for the inner-loop update $\mathcal{L}^{\text{in}}_{\mathcal{T}_i}(f_\theta)$, calculated as the sum of two losses as follows:

$$\mathcal{L}^{\text{in}}_{\mathcal{T}_i}(f_\theta) = \mathcal{L}_{\mathcal{T}_i}(\hat{\mathbf{I}}_3, \mathbf{I}_3) + \mathcal{L}_{\mathcal{T}_i}(\hat{\mathbf{I}}_5, \mathbf{I}_5). \tag{4}$$

Next, we calculate the gradients for $\mathcal{L}^{\text{in}}_{\mathcal{T}_i}(f_\theta)$ and update θ with gradient descent to obtain the customized parameters θ'_i for each task $\mathcal{T}_i$. Notably, we can use any gradient-based optimizer (*e.g.* Adam [56]) for the updating step, and we match the type of the optimizer used to train the baseline pre-trained model in practice. Moreover, the inner-loop update can optionally consist of multiple iterations such that θ'_i is a result of k gradient updates from θ, where k is the number of iterations. We analyze the effect of the hyper-parameter k in Sec. 4.4, and choose $k = 1$ throughout our experiments for performance and simplicity (Table 3). To further reduce computation, we employ a first-order approximation as suggested in [23] and avoid calculating the second-order derivatives required for the nested-loop updates in meta-training.

When training the outer-loop, the parameters are updated to minimize the losses for $f_{\theta'_i}$ with respect to θ, on each of the task-wise test triplet $\{(\mathbf{I}_3, \mathbf{I}_4, \mathbf{I}_5)\} \in \mathcal{D}'_{\mathcal{T}_i}$. The loss function for the outer-loop meta-update is defined as

$$\mathcal{L}^{\text{out}}_{\mathcal{T}_i}(f_{\theta'_i}) = \mathcal{L}_{\mathcal{T}_i}(f_{\theta'_i}(\mathbf{I}_3, \mathbf{I}_5), \mathbf{I}_4), \tag{5}$$

and the summation of all losses for the sampled batch of sequences $\mathcal{T}_i \sim p(\mathcal{T})$ is used to calculate the gradient and update the model parameters. The overall training process is summarized in Algorithm 1.

Inference. At test time, the base parameters θ for the outer-loop are fixed, and only the inner-loop update is performed to modify the parameter values to θ'_i for each test sequence $\mathcal{T}_i$. Then, the final interpolations can be obtained as the output of the adapted model $f_{\theta'_i}$. The full inference process for interpolating a new test video sequence is shown in Algorithm 2. For long video inputs with many frames, the inference is performed in a sliding-window manner with the window size of four. We add the first and the last frames at the beginning and the end of the video, respectively, to maintain the number of input frames to four (*e.g.* $\{\mathbf{I}_1, \mathbf{I}_1, \mathbf{I}_3, \mathbf{I}_5\}$ are used as the input frames to generate $\mathbf{I}_2$, which corresponds to $t = -1$ in Alg. 2). Similarly, if there are only two input frames, we replicate the frames to make the number of frames to four.

Notably, the biggest difference from our algorithm from the original MAML is that the distributions for the task-wise training and test set, $\mathcal{D}_{\mathcal{T}_i}$ and $\mathcal{D}'_{\mathcal{T}_i}$, are not the same. That is, $\mathcal{D}_{\mathcal{T}_i}$ have a broader spectrum of motion and includes $\mathcal{D}'_{\mathcal{T}_i}$, since the time gap between the frame triplets is twice as large. Although this case with a distribution gap is an unexplored area in the meta-learning literature, it shows an encouraging effect for the task of video frame interpolation; the model trained with our algorithm learns to update itself in considerably more difficult scenarios with larger motion, learning the overall context and motion present in the video as a result. Then, interpolations for the original input frames become an easy task for our well-adapted model, which results in a performance gain. Both quantitative and qualitative results in the experiments show that our algorithm improves the original model to better handle bigger motion. A thorough empirical analysis regarding this issue is shown Sec. 4.4, where we study the effects of modifying the batch configurations for composing the support set and the query set.

Algorithm 1: Training with MetaVFI

Require: $p(\mathcal{T})$: uniform distribution over sequences
Require: α, β: step size hyper-parameters
Input : Training video sequences $\mathcal{T}$
Output : Learned initialization θ

Initialize parameters θ
while *not converged* **do**
  Sample batch of sequences $\mathcal{T}_i \sim p(\mathcal{T})$
  foreach *i* **do**
    Generate triplets $\mathcal{D}_{\mathcal{T}_i} = \{(\mathbf{I}_1, \mathbf{I}_3, \mathbf{I}_5), (\mathbf{I}_3, \mathbf{I}_5, \mathbf{I}_7)\}$ from $\mathcal{T}_i$
    Compute $\hat{\mathbf{I}}_3, \hat{\mathbf{I}}_5$ in Eq. (3)
    Evaluate $\nabla_\theta \mathcal{L}^{\text{in}}_{\mathcal{T}_i}(f_\theta)$ using $\mathcal{L}_{\mathcal{T}_i}$ in Eq. (4)
    Compute adapted parameters with gradient descent: $\theta'_i = \theta - \alpha \nabla_\theta \mathcal{L}^{\text{in}}_{\mathcal{T}_i}(f_\theta)$
    Generate and save triplet $\mathcal{D}'_{\mathcal{T}_i} = \{(\mathbf{I}_3, \mathbf{I}_4, \mathbf{I}_5)\}$ from $\mathcal{T}_i$ for the meta-update
  end
  Update $\theta \leftarrow \theta - \beta \nabla_\theta \sum_{\mathcal{T}_i \sim p(\mathcal{T})} \mathcal{L}^{\text{out}}_{\mathcal{T}_i}(f_{\theta'_i})$ using each $\mathcal{D}'_{\mathcal{T}_i}$ and $\mathcal{L}_{\mathcal{T}_i}$ in Eq. (5)
end

Algorithm 2: Inference with MetaVFI

Require: θ: meta-trained parameter initialization
Require: α: inner-loop learning rate hyper-parameter
Input : Test video sequence $\mathcal{T} = \{\mathbf{I}_1, \mathbf{I}_3, \cdots\}$
Output : Interpolations $\{\mathbf{I}_2, \mathbf{I}_4, \cdots\}$

foreach *t* **do**
  Build input triplets from $\mathcal{T}$: $\mathcal{D}_{\mathcal{T}_t} = \{(\mathbf{I}_{2t+1}, \mathbf{I}_{2t+3}, \mathbf{I}_{2t+5}), (\mathbf{I}_{2t+3}, \mathbf{I}_{2t+5}, \mathbf{I}_{2t+7})\}$
  Compute $\hat{\mathbf{I}}_{2t+3}, \hat{\mathbf{I}}_{2t+5}$ as in Eq. (3)
  Evaluate $\nabla_\theta \mathcal{L}^{\text{in}}_{\mathcal{T}}(f_\theta)$ using $\mathcal{L}_{\mathcal{T}}$ as in Eq. (4)
  Compute adapted parameters with gradient descent: $\theta' = \theta - \alpha \nabla_\theta \mathcal{L}^{\text{in}}_{\mathcal{T}}(f_\theta)$
  Compute $\hat{\mathbf{I}}_{2t+4} = f_{\theta'}(\mathbf{I}_{2t+3}, \mathbf{I}_{2t+5})$
end

4 Experiments

4.1 Settings

Datasets Most of the existing works on video frame interpolation use the video data pre-processed into frame triplets. Although our baseline model is pre-trained with conventional triplet datasets, it is not applicable for training the outer-loop because multiple input frames are needed to construct the task-wise training samples for the inner-loop update. To this end, we use Vimeo90K-Septuplet

(VimeoSeptuplet) dataset [7], which consists of 91,701 seven-frame sequences with a fixed resolution of 448×256. This dataset is originally designed for video super-resolution or denoising / deblocking but is also well suited for training video frame interpolation models that require multiple frames at test time. We train all of our models with the training split of the VimeoSeptuplet dataset. For evaluation, we use the test split of VimeoSeptuplet dataset, as well as sequences from Middlebury-`OTHERS` [53], HD [31], and SNU-FILM [1] dataset.

The `OTHERS` set from Middlebury contains a total of 12 examples, with a maximum resolution of 640×480. We use 10 sequences with multiple input frames and remove the other two that only have two input frames and are thus not suitable for test-time adaptation. We denote the 10-sequence subset with an asterisk (*) to distinguish our evaluation setting with the original setting that uses all 12 examples.

The HD dataset proposed by Bao *et al.* [31] consists of relatively high-resolution frames, from 1280×544 to 1920×1080. The length of the sequences in the HD dataset is either 70 or 100, enabling test-time updates to our model.

Choi *et al.* [1] proposed the SNU-FILM dataset for a comprehensive evaluation of frame interpolation models with respect to the motion magnitude. The dataset consists of 31 different sequences, and performance is measured on 10 frames per sequence, totalling 310 frames. Similar to the HD dataset, the resolution ranges from 1280×544 to 1920×1080. In this work, we use the *Hard* setting for evaluation, since it contains challenging scenarios with sufficiently large motion.

Training settings for comparison. For our experiments, we use six conventional video frame interpolation models as baselines: DVF [3], SuperSloMo [2], SepConv [6], DAIN [30], CAIN [1], and RRIN [32]. We first initialize each model with the pre-trained parameters, provided by the authors if possible.[1] We denote these models as ***Baseline***. Then, since we use the additional training set from VimeoSeptuplet for meta-training, we also fine-tune each *Baseline* models with VimeoSeptuplet training set, denoted as ***Re-trained*** models. For our final models trained with ***MetaVFI***, we start from the *Baseline* model parameters and follow the iterative steps for inner and outer-loop training in Algorithm 1. Following the results from Sec. 4.2 (Table 1), we report two versions of *MetaVFI*: FO-MAML and Meta-SGD (Table 2). The reported performance for all meta-trained models uses a single inner-loop update iteration at test time. We examine the effects of increasing the number of gradient updates in the ablation study (Sec. 4.4).

Implementation details. We match the type of loss functions, normalization, and optimization schemes for the gradient updates with the original methods used to train the *Baseline* models, which differs for each method. However, since we are fine-tuning from the pre-trained networks, we modify the inner / outer-loop learning rates to be small and set $\alpha = \beta = 10^{-5}$. Throughout the training, α is kept fixed, whereas β is decayed by a factor of five whenever the validation loss does not decrease for more than 10,000 outer-loop iterations. We train until $\beta = 4 \times 10^{-7}$ (*i.e.* the learning rate is decayed twice) and the validation loss plateaus for 10,000 iterations. We also set the maximum number of training epochs to be 60 ($\sim$ 300,000 iterations) to restrict too long training time when this convergence criterion is not met. We crop 256×256 sized patches from VimeoSeptuplet sequences and train with a mini-batch size of 8. Although the number of training iterations differs for each interpolation model, the full meta-training step for any model requires approximately 3$\sim$4 days with a single NVIDIA RTX 2080Ti GPU. In our implementation for this work, we merge all considered frame interpolation models and meta-learning algorithms into a single repository and enable easy experimentation with various settings by just changing the options. For this integration, models that were using old library versions are modified to run with the recent version Thus, the exact numbers may slightly differ from our previous work [24], although the tendency of *Baseline–Re-trained–MetaVFI* settings remains the same. The source code for our framework is made public[2] along with the pre-trained models to facilitate reproduction.

1. For SuperSloMo [2], we use the implementations and pre-trained models from [57]. For all of the other interpolation models, we use the official codes provided by the authors.

TABLE 1
Quantitative results on VimeoSeptuplet [7] dataset for two representative frame interpolation models trained by the proposed algorithm with different meta-learning methods.

Method	SepConv [6]	SuperSloMo [2]
FO-MAML [23]	34.17 / 0.9482	34.41 / 0.9497
MAML [23]	34.17 / 0.9483	–
MAML++ [54]	34.35 / 0.9493	34.63 / 0.9509
FO-MAML+L2F [58]	34.17 / 0.9482	34.50 / 0.9504
Meta-SGD [55]	**34.45 / 0.9501**	**34.66 / 0.9514**

4.2 Meta-learning algorithm selection

In this section, we analyze the effects of using different MAML-based meta-learning algorithms on the proposed MetaVFI framework. Using two representative video frame interpolation models, SepConv [6] and SuperSloMo [2], we apply four different meta-learning algorithms: MAML [23], MAML++ [54], L2F [58], and Meta-SGD [55]. The results are summarized in Table 1. Notably, we can observe that the performance of meta-learning algorithms shows a very different tendency when applied to video frame interpolation, compared with the few-shot classification literature.

For all compared meta-learning algorithms except MAML (second row in Table 1), we do not calculate the full second-order gradients and use the first-order variant of MAML (FO-MAML) as the baseline. This is because of two reasons: computational inefficiency and negligible performance improvements. Since video frame interpolation models are typically much larger than commonly-used few-shot classification models, computing the full gradients for such models is much more burdensome. Specifically, training our framework with the full MAML requires nearly four times more memory and more than twice the training time in practice, compared with its first-order variant. Also, for SuperSloMo, second-order gradient calculation is not supported for the PyTorch `grid_sampler` module used for warping the input frames with the estimated optical flow, which made training more difficult. The top 2 rows of Table 1 show that no significant performance improvements are found when using the full MAML, instead of FO-MAML. We believe that the effects for second-order gradients are limited in our current application, because the proposed framework only uses a single iteration for the inner-loop update. Given the aforementioned reasons, we apply the other meta-learning algorithms (MAML++, L2F, and Meta-SGD) based on FO-MAML.

Among the compared meta-learning algorithms in this ablation study, Meta-SGD showed the best performance, closely followed

2. https://github.com/myungsub/meta-interpolation

TABLE 2
Quantitative results for meta-training for recent frame interpolation algorithms. We evaluate the benefits of our scene-adaptive framework on four datasets: VimeoSeptuplet [7], Middlebury-OTHERS* [53], HD [31], and SNU-FILM [1]. Performance is measured in PSNR (dB) / SSIM. Notably, our *+MetaVFI* performance consistently improves upon the *Baseline* or *Re-trained* correspondents, regardless of the meta-learning algorithm.

Model	Training Method	VimeoSeptuplet [7]	Middlebury-OTHERS* [53]	HD [31]	SNU-FILM [1]
DVF [3]	Baseline	26.58 / 0.8203	23.04 / 0.6784	20.29 / 0.5641	22.01 / 0.6744
	Re-trained	32.19 / 0.9194	29.50 / 0.8538	24.73 / 0.7488	24.80 / 0.7570
	+MetaVFI (FO-MAML)	**32.25 / 0.9210**	**29.68 / 0.8560**	**24.95 / 0.7522**	**25.18 / 0.7632**
	+MetaVFI (Meta-SGD)	32.24 / **0.9210**	29.59 / 0.8536	24.87 / 0.7485	25.14 / 0.7617
SepConv [6]	Baseline	33.70 / 0.9445	35.14 / 0.9581	29.60 / 0.8752	29.27 / 0.8835
	Re-trained	33.62 / 0.9446	34.90 / 0.9582	29.62 / 0.8789	29.28 / 0.8845
	+MetaVFI (FO-MAML)	34.17 / 0.9482	35.73 / 0.9634	29.99 / 0.8812	29.49 / 0.8866
	+MetaVFI (Meta-SGD)	**34.45 / 0.9501**	**36.06 / 0.9649**	**30.12 / 0.8833**	**29.77 / 0.8909**
SuperSloMo [2]	Baseline	30.80 / 0.9284	32.64 / 0.9408	28.77 / 0.8668	28.58 / 0.8794
	Re-trained	34.28 / 0.9487	34.96 / 0.9582	29.86 / 0.8802	29.60 / 0.8893
	+MetaVFI (FO-MAML)	34.41 / 0.9497	34.91 / 0.9571	29.84 / 0.8804	29.81 / 0.8932
	+MetaVFI (Meta-SGD)	**34.66 / 0.9514**	**35.25 / 0.9586**	**29.98 / 0.8815**	**30.02 / 0.8991**
DAIN [30]	Baseline	34.78 / 0.9522	36.27 / 0.9657	30.35 / 0.8900	30.09 / 0.8975
	Re-trained	34.88 / 0.9537	36.27 / 0.9651	30.33 / 0.8927	30.17 / 0.8992
	+MetaVFI (FO-MAML)	34.92 / 0.9536	36.40 / 0.9662	30.37 / 0.8932	30.19 / 0.8994
	+MetaVFI (Meta-SGD)	**35.17 / 0.9553**	**36.53 / 0.9672**	**30.56 / 0.8954**	**30.45 / 0.9037**
CAIN [1]	Baseline	34.43 / 0.9489	**34.90 / 0.9511**	30.18 / 0.8823	29.87 / 0.8889
	Re-trained	34.68 / 0.9509	34.68 / 0.9485	30.36 / 0.8853	30.07 / 0.8925
	+MetaVFI (FO-MAML)	34.86 / 0.9520	34.65 / 0.9479	30.43 / 0.8864	30.24 / 0.8949
	+MetaVFI (Meta-SGD)	**35.01 / 0.9527**	34.74 / 0.9489	**30.50 / 0.8870**	**30.46 / 0.9003**
RRIN [32]	Baseline	33.42 / 0.9487	35.21 / 0.9603	29.71 / 0.8853	29.31 / 0.8887
	Re-trained	35.27 / 0.9563	35.92 / 0.9642	30.36 / 0.8906	29.82 / 0.8938
	+MetaVFI (FO-MAML)	35.37 / 0.9568	36.07 / 0.9652	30.39 / 0.8912	29.93 / 0.8959
	+MetaVFI (Meta-SGD)	**35.56 / 0.9580**	**36.21 / 0.9662**	**30.42 / 0.8929**	**30.24 / 0.9036**

by MAML++. As previously mentioned, this finding is very different from the results for few-shot classification accuracy, where L2F greatly outperforms MAML++, which already improved upon Meta-SGD or MAML by a significant margin. We believe that this difference is mainly because of the different characteristics of the target application. While preventing overfitting is arguably the most crucial issue when training for few-shot classification, its effect for video frame interpolation is not as dramatic; the PSNR values for the training and validation set remain similar all the way until convergence. Therefore, any regularization effects such as attenuation in L2F did not show impressive performance improvements, and better fitting capability resulted in better performance. Regarding the fact that Meta-SGD uses twice the number of parameters than MAML or L2F due to parameter-wise learning rates, it has the biggest fitting capacity and hence the best performance. MAML++ also introduces a few additional number of parameters for layer-wise learning rates, which we believe is the reason for the second-best performance.

4.3 Video frame interpolation results

Quantitative results. Table 2 shows the summary of quantitative performance for all considered baseline frame interpolation models for all evaluated datasets. We report two results for (Baseline interpolation model)+*MetaVFI*. First, +*MetaVFI (FO-MAML)* follows the base settings from [24], and second, +*MetaVFI (Meta-SGD)* follows the best-performing settings shown in Sec. 4.2. For the evaluation metrics, we report peak signal-to-noise ratio (PSNR) and structural similarity measure (SSIM), which are widely-used full-reference measures for evaluating performance.

Notably, Table 2 shows the consistent performance boost achieved by adding *MetaVFI* compared with *Baseline* and *Re-trained* models, regardless of the method used for video frame interpolation. Moreover, although meta-training for our scene-adaptive algorithm is only done in the VimeoSeptuplet dataset, *MetaVFI* generalizes well to the other datasets with different characteristics, presenting the benefits of test-time adaptiveness of our approach. Between the two baselines, the *Re-trained* model generally performs better than the *Baseline* model. We believe that the reason is the quality (*i.e.*, degree of noise, artifacts, blurriness, *etc.*) of the training frames, because the frame sequences in VimeoSeptuplet are relatively clean. Since DVF is trained with videos from UCF-101 [59] dataset that has severe artifacts, its performance improvement for fine-tuning to VimeoSeptuplet was the largest. The original training set, Adobe-240fps [60], for SuperSloMo [2] implementation also contains some degree of noise, and thus, re-training helps to build a much stronger baseline. An exception to this is SepConv [6], where re-training rather hurts the model's generalization capability to some of the other datasets.

Both models trained with *MetaVFI* considerably outperform the *Baseline* or *Re-trained* models, even for the most recent state-of-the-art frameworks [1], [30], [32]. As demonstrated in our previous work [24], +*MetaVFI (FO-MAML)* already achieves notable performance gain, which is consistent regardless of the frame interpolation model or the evaluated dataset. Incorporating Meta-SGD further boosts the performance by better adapting to the test-time inputs, exploiting the remaining room for improvements attainable by meta-training. Thus, +*MetaVFI (Meta-SGD)* setting almost always shows the best result, but with one exception of DVF where using FO-MAML results in better performance.

Visual comparison. Fig. 4 shows the qualitative results for the VimeoSeptuplet data-set, where we compare the *MetaVFI*-trained models with *Baseline* and *Re-trained* models for each video frame interpolation algorithm. Note that our focus is on analyzing the benefits of *MetaVFI* training with its corresponding baselines, rather than comparing between different frame interpolation algorithms. For many cases where the baseline models fail due to

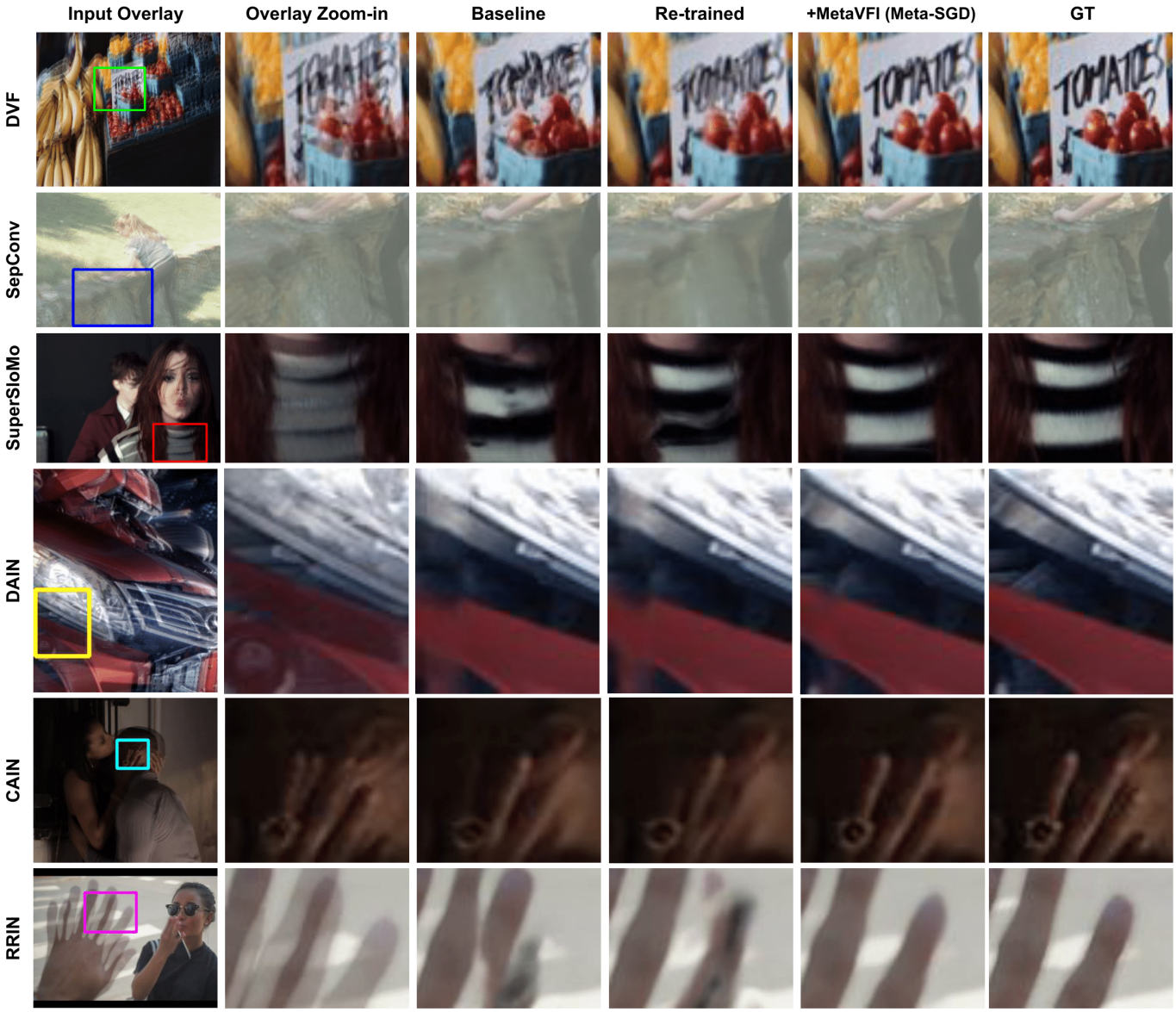

Fig. 4. Qualitative results on VimeoSeptuplet [7] dataset for recent frame interpolation algorithms. Note how our *+MetaVFI (Meta-SGD)* outputs infer motion substantially better than the *Baseline* or *Re-trained* models and generate realistic textures similar to the ground truth.

large motion, our *+MetaVFI* model adapts to the input sequence remarkably well to synthesize better texture and more precise position of the moving regions. In particular, SepConv+MetaVFI greatly improves the textural details of the rocks, because it has adapted to the nearby frames that contain the same rocks with different positions. Also, existing frame interpolation models sometimes do not synthesize the intermediate frame but just copy the region with large motion from one of the neighboring frames. For CAIN, the result for the *Baseline* setting looks relatively good at first sight, but the position of the fingers is not correct; in fact, it is simply copied from one of the input frame. While *Re-trained* result cannot fix the errors and show unpleasing artifacts, *+MetaVFI (Meta-SGD)* output correctly finds the correct position and synthesize clearer boundaries.

Additional qualitative results for Middlebury-OTHERS* and HD datasets are shown in Fig. 5 and 6, respectively. We mainly compare the results obtained with SepConv, but the other models also show similar characteristics. For Fig. 5, the original SepConv model and the re-trained version almost completely fail to reason for the motion of the bean bag, whereas the *+MetaVFI* result

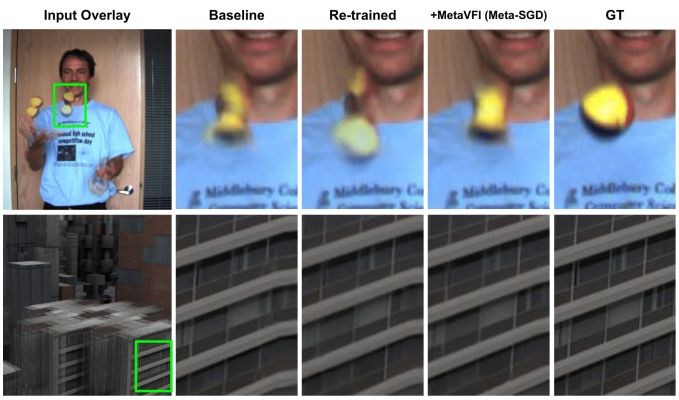

Fig. 5. Qualitative results on Middlebury-OTHERS* [53] dataset for SepConv [6]. We show the cropped regions for *Beanbags* and *Urban2* sequences. Note how meta-training helps in fixing most of the errors in the baseline models occurring due to large motion.

significantly improves in fixing this error. Adaptation with meta-training also fixes the curved part of the building which should be straight. Fig. 6 shows the result obtained with SepConv and

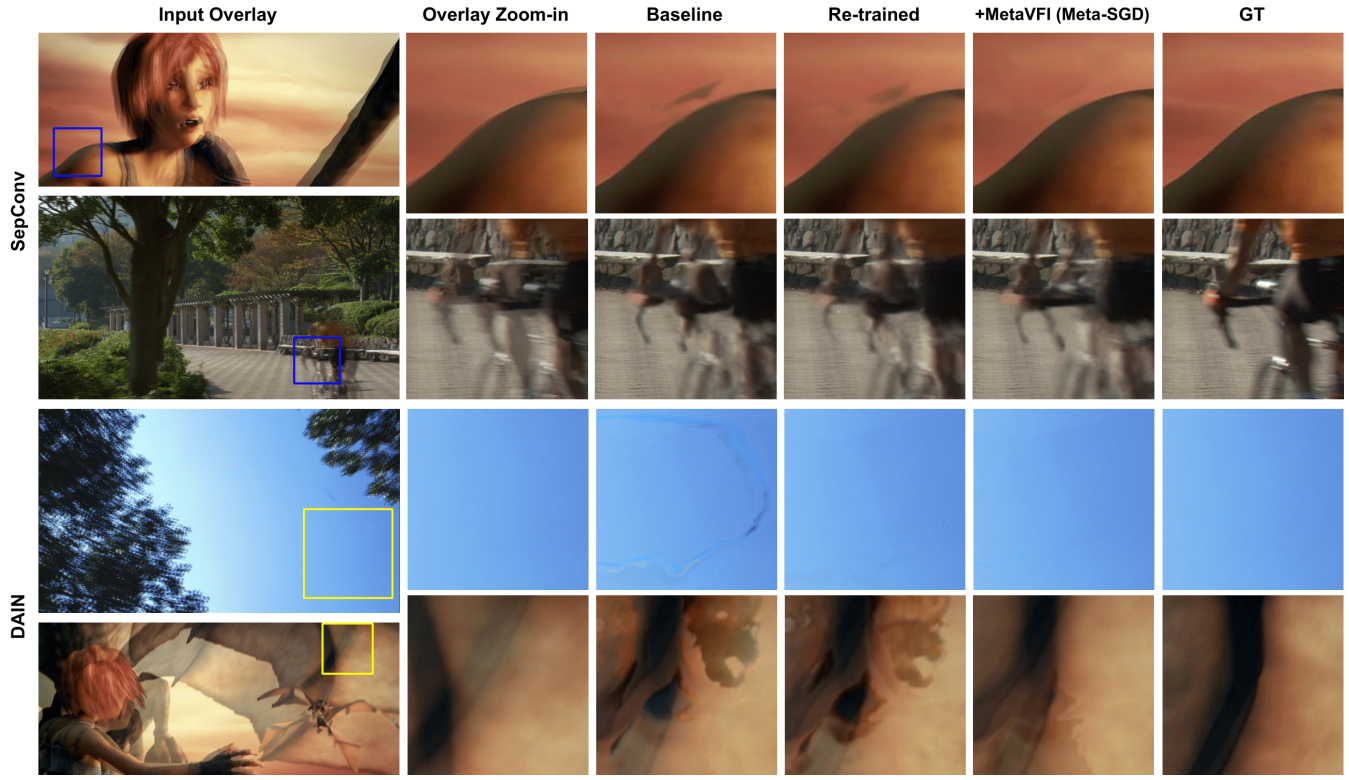

Fig. 6. Qualitative results on HD [31] dataset for SepConv [6] and DAIN [30]. We show the cropped regions for *Temple1*, *ParkScene*, and *BlueSky* sequences. Note how the artifacts are removed for *+MetaVFI (Meta-SGD)* results.

DAIN. Due to the high resolution of the HD dataset, the magnitude of motion becomes very large in terms of the number of pixels, and motion estimation can fail for baseline models. This is more notable in DAIN, where unpleasing artifacts appear in the middle of the sky due to failure in optical flow estimation sub-module.[3] However, these artifacts are removed when we adapt to the input frames, illustrated by clean output frames from *+MetaVFI* model.

Fig. 7 depicts the visual comparisons with all compared interpolation models for the SNU-FILM dataset. Similar characteristics can be observed as in Figs. 4, 5, and 6, and our models trained with *+MetaVFI* produces clearer interpolations with less artifacts. Notably, the missing shadows appear for CAIN, and the missing logo of the shorts becomes present for RRIN. Since the qualitative improvements are mostly from fixing the errors for incorrectly inferred motion, we believe that meta-training is beneficial because of its ability to adapt to the dominant types of local motion in the nearby frames within the small temporal window. For more qualitative results and the sample generated videos, please also refer to our project page.[4]

Computational complexity. Compared with the corresponding *Baseline* or *Re-trained* models, *MetaVFI* model needs approximately $\times 7$ slower inference time, which includes the 3 forward passes and 2 backward passes (backward pass for PyTorch on a GPU is approximately $\times 2$ slower than the forward pass). However, this additional computational complexity can be reduced by *e.g.* updating the parameter values of just a small portion of the interpolation network, or performing test-time adaptation only once in a few sequences for a continuous video input.

3. The reason for this artifact is discussed in http://vllab1.ucmerced.edu/~wenbobao/DAIN/cvpr19_DAIN_supp.pdf, page 16, section 5.3.

4. https://myungsub.github.io/meta-interpolation/

4.4 Ablation studies

Effects on the number of inner-loop updates. We analyze the effects of modifying the number of iterations for test-time adaptation. Table 3 demonstrates how the final performance changes while varying the number of inner-loop updates from 1 to 5. We also show the results for naive test-time fine-tuning (from the *Re-trained* model) along with our *+MetaVFI* results.

In summary, meta-training for just a single inner-loop update, which is used in most of our experiment settings, shows the most PSNR gain, while increasing the number of updates did not have any benefits on performance. More updates even showed diminishing results, which is somewhat counter-intuitive compared with the tendency reported in MAML [23]. We believe there are two possible reasons for this phenomenon. First is overfitting to the data used for inner-loop update ($\mathcal{D}_{\mathcal{T}_i}$). In Sec 3.2, we have shown that it is beneficial to use $\mathcal{D}_{\mathcal{T}_i}$ as a proxy for achieving good performance for $\mathcal{D}'_{\mathcal{T}_i}$ regardless of their distribution gap. However, the current ablation study suggests that *over*-fitting to $\mathcal{D}_{\mathcal{T}_i}$ can have negative effects on the final performance. This finding points out the need for finding the sweet spot in the trade-off between extracting from $\mathcal{D}_{\mathcal{T}_i}$ useful information that aids improving the interpolations in $\mathcal{D}'_{\mathcal{T}_i}$, and overfitting to $\mathcal{D}_{\mathcal{T}_i}$. For video frame interpolation, an example of common useful information can be the direction of existing motion or the details on background textures. If overfitting occurs, the inner-loop may excessively focus on handling the existing large motion and disregard the generic prior knowledge learned by the *Baseline* pre-trained model and its *Re-trained* version. The second reason is due to the growing complexity of training as the number of gradient updates increases, which makes the model susceptible to falling into local minima [23], [61]. Presumably, incorporating recent meta-learning techniques with better inner-loop training

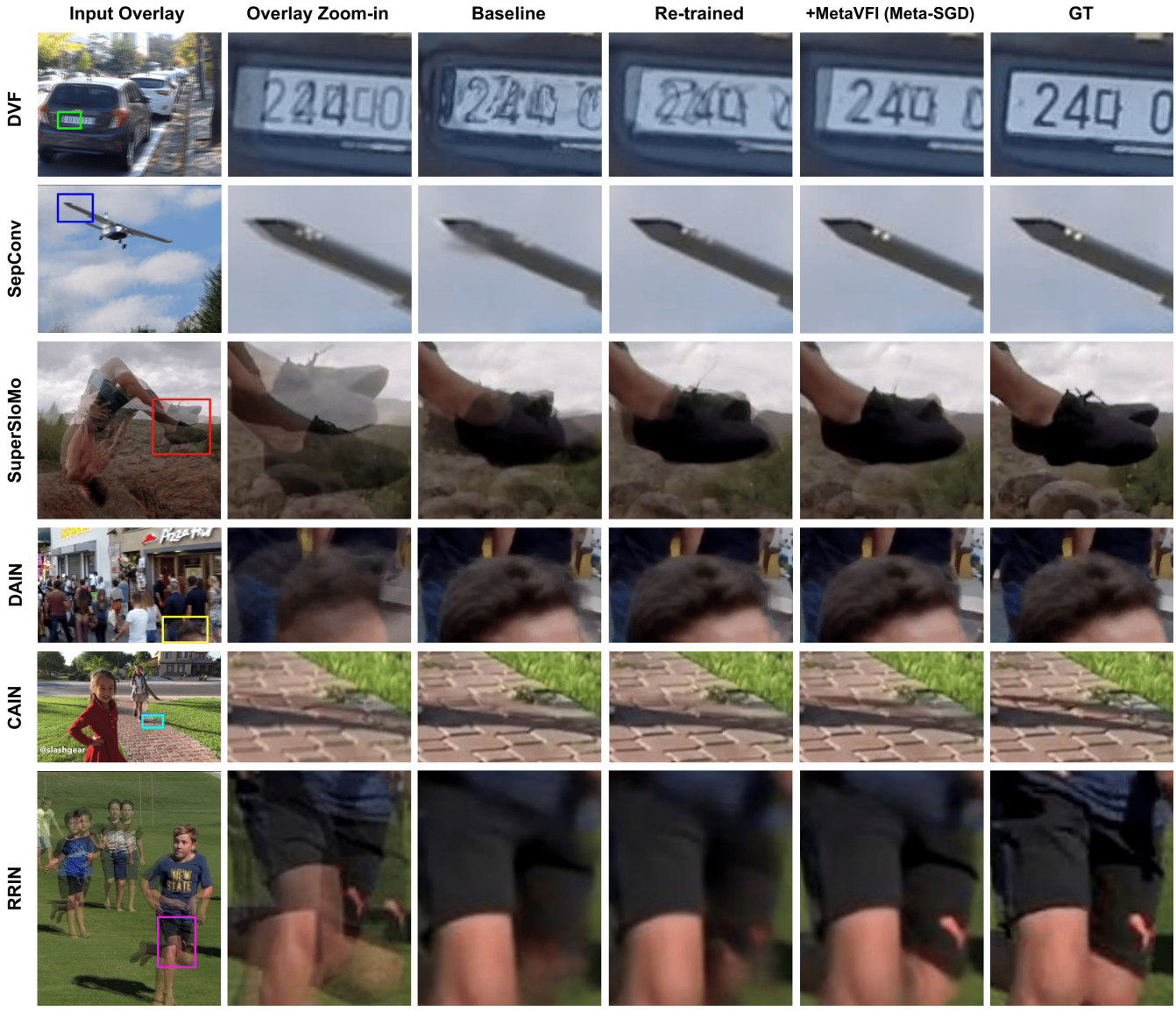

Fig. 7. Qualitative results on SNU-FILM [1] dataset (Hard setting) for recent frame interpolation algorithms. Our *+MetaVFI (Meta-SGD)* output handles the regions with large motion better than the *Baseline* or *Re-trained* models, thereby substantially reducing the ghost artifacts and blurs to generate sharper boundaries and restore missing objects.

TABLE 3
Effects on varying the the number of inner-loop updates. Zero updates correspond to the *Re-trained* setting. PSNR (dB) for SepConv [6] is shown for the VimeoSeptuplet [7] dataset.

# gradient updates	0	1	2	3	5
Naive Fine-tune	33.62	33.74	33.90	33.94	33.97
+MetaVFI (FO-MAML)	33.62	**34.17**	34.08	34.03	33.99
+MetaVFI (Meta-SGD)	33.62	**34.45**	34.29	34.26	34.24

TABLE 4
Effects on varying the learning rates for the inner-loop updates. We use SepConv [6] for performance comparison on the VimeoSeptuplet [7] dataset. Since Meta-SGD [55] learns per-parameter learning rates, we only report the results for meta-training with FO-MAML [23].

Learning rate α	0	10^{-6}	10^{-5}	10^{-4}
PSNR (dB)	33.62	34.10	**34.17**	34.15

and proper regularization [62] can help mitigate this issue, which remains as our future work.

Effects on inner-loop learning rate. MetaVFI algorithm starts training from a pre-trained video frame interpolation model. Hence, we believe that large learning rates for the inner-loop update (α in Algorithm 1) can break the model's original performance at the early stage of training, whereas too small learning rates restrict the adaptive capability of the model. To support this claim, we report the performances on setting different values of α in Table 4 using SepConv. Since Meta-SGD can show varying learning rates for each model parameter, we perform this ablation study with the model meta-trained by FO-MAML. The final performance is maximized for the learning rate of 10^{-5}, with small gaps in PSNR compared with 10^{-4} or 10^{-6}. However, regardless of the values of α, the final performance is always better than when $\alpha = 0$, which demonstrates the effectiveness of test-time adaptation with MetaVFI algorithm.

Effects on inner-loop batch configuration. We test whether the performance gain in the proposed algorithm is due to test-time adaptation via meta-learning or just because the model can observe bigger motions at training time. Table 5 shows the results of this analysis. The first row shows the performance of our *Re-trained* model, since it does not have any inner-loop updates (hence, no

TABLE 5
Effects on varying the batch configuration for inner-loop adaptation. We re-train the SepConv [6] baseline for the fine-tuned models (models without inner-loop), and meta-train with FO-MAML or Meta-SGD. We show the performance on VimeoSeptuplet [7] dataset.

MetaVFI Alg.	Inner-loop	Outer-loop	PSNR
—	—	(3, 4, 5)	33.62
—	—	(3, 4, 5), (1, 3, 5), (3, 5, 7)	34.09
FO-MAML	(1, 3, 5)	(3, 4, 5)	34.12
Meta-SGD	(1, 3, 5)	(3, 4, 5)	34.33
FO-MAML	(1, 3, 5), (3, 5, 7)	(3, 4, 5)	34.17
Meta-SGD	(1, 3, 5), (3, 5, 7)	(3, 4, 5)	**34.45**

meta-learning is involved). The *Re-trained* model is trained with the original, relatively small motion between the two input frames, $\mathbf{I}_3$ and $\mathbf{I}_5$, to predict the target frame, $\mathbf{I}_4$. In the second row, triplets with larger motion are added; the time step between the two input frames is twice bigger, thereby having twice larger motion on average. This results in 0.47 dB PSNR gain, which means that observing larger motion during the training stage is quite effective, even without any adaptation to the current inputs. From the few-shot learning perspective, we can name the first two rows of Table 5 as *zero-shot* models.

The third to the last rows demonstrate the performance improvements in applying meta-learning. Instead of just adding the two large-motion triplets, $(1, 3, 5)$ and $(3, 5, 7)$, to the training set for fine-tuning, using them to adapt the model parameters in the inner-loop achieves further improvements: 0.36 dB gain in PSNR for Meta-SGD, and 0.08 dB gain for FO-MAML. While using FO-MAML with MetaVFI demonstrated relatively marginal PSNR improvement compared to the zero-shot model (2nd row), Meta-SGD better exploits adaptation to the test-time inputs and find significant more rooms for performance gain. Considering the last two rows as models with two-shot updates, the third and fourth row introduce one-shot variants, which shows some performance drop but with better computational efficiency. However, using a single triplet for model adaptation still improves upon the zero-shot baseline by a notable margin, which demonstrates the effectiveness of the proposed MetaVFI framework.

5 Conclusion

In this study, we introduced a novel framework for video frame interpolation. We incorporated a meta-learning algorithm to train the network that can quickly adapt its parameters to the input frames at test-time. Experimental results show consistently improved performance of the proposed method on multiple benchmark datasets. Moreover, through extensive ablation studies, the roles and effects of hyper-parameters are analyzed and the reasons for the benefits of meta-learning are examined. The proposed MetaVFI algorithm can be easily employed on top of any existing video frame interpolation network to significantly improve its performance without changing its architecture.

Acknowledgments

This work was supported in part by IITP grant funded by the Korea government [No. 2021-0-01343, Artificial Intelligence Graduate School Program (Seoul National University)], and in part by Hyundai Motor Group through HMG-SNU AI Consortium fund (No. 5264-20190101).

References

[1] M. Choi, H. Kim, B. Han, N. Xu, and K. M. Lee, "Channel attention is all you need for video frame interpolation," in *AAAI*, 2020.
[2] H. Jiang, D. Sun, V. Jampani, M.-H. Yang, E. Learned-Miller, and J. Kautz, "Super slomo: High quality estimation of multiple intermediate frames for video interpolation," in *CVPR*, 2018.
[3] Z. Liu, R. A. Yeh, X. Tang, Y. Liu, and A. Agarwala, "Video frame synthesis using deep voxel flow." in *ICCV*, 2017.
[4] S. Niklaus and F. Liu, "Context-aware synthesis for video frame interpolation," in *CVPR*, 2018.
[5] S. Niklaus, L. Mai, and F. Liu, "Video frame interpolation via adaptive convolution," in *CVPR*, 2017.
[6] ——, "Video frame interpolation via adaptive separable convolution," in *ICCV*, 2017.
[7] T. Xue, B. Chen, J. Wu, D. Wei, and W. T. Freeman, "Video enhancement with task-oriented flow," in *CVPR*, 2018.
[8] J. Carreira and A. Zisserman, "Quo vadis, action recognition? a new model and the kinetics dataset," in *CVPR*, 2017.
[9] J. Cheng, Y.-H. Tsai, S. Wang, and M.-H. Yang, "Segflow: Joint learning for video object segmentation and optical flow," in *ICCV*, 2017.
[10] K. Simonyan and A. Zisserman, "Two-stream convolutional networks for action recognition in videos," in *NIPS*, 2014.
[11] X. Zhu, J. Dai, L. Yuan, and Y. Wei, "Towards high performance video object detection," in *CVPR*, 2018.
[12] X. Zhu, Y. Wang, J. Dai, L. Yuan, and Y. Wei, "Flow-guided feature aggregation for video object detection," in *ICCV*, 2017.
[13] D. Glasner, S. Bagon, and M. Irani, "Super-resolution from a single image," in *ICCV*, 2009.
[14] J.-B. Huang, A. Singh, and N. Ahuja, "Single image super-resolution from transformed self-exemplars," in *CVPR*, 2015.
[15] J.-J. Huang, T. Liu, P. Luigi Dragotti, and T. Stathaki, "Srhrf+: Self-example enhanced single image super-resolution using hierarchical random forests," in *CVPR Workshops*, 2017.
[16] T. Michaeli and M. Irani, "Nonparametric blind super-resolution," in *ICCV*, 2013.
[17] A. Shocher, N. Cohen, and M. Irani, ""zero-shot" super-resolution using deep internal learning," in *CVPR*, 2018.
[18] S. Park, J. Yoo, D. Cho, J. Kim, and T. H. Kim, "Fast adaptation to super-resolution networks via meta-learning," *arXiv:2001.02905*, 2020.
[19] J. W. Soh, S. Cho, and N. I. Cho, "Meta-transfer learning for zero-shot super-resolution," in *CVPR*, 2020, pp. 3516–3525.
[20] J. Choi, J. Kwon, and K. M. Lee, "Deep meta learning for real-time visual tracking based on target-specific feature space," *arXiv:1712.09153*, 2017.
[21] M. Danelljan, G. Bhat, F. Shahbaz Khan, and M. Felsberg, "Eco: Efficient convolution operators for tracking," in *CVPR*, 2017.
[22] H. Nam and B. Han, "Learning multi-domain convolutional neural networks for visual tracking," in *CVPR*, 2016.
[23] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *ICML*, 2017.
[24] M. Choi, J. Choi, S. Baik, T. H. Kim, and K. M. Lee, "Scene-adaptive video frame interpolation via meta-learning," in *CVPR*, 2020.
[25] G. Long, L. Kneip, J. M. Alvarez, H. Li, X. Zhang, and Q. Yu, "Learning image matching by simply watching video," in *ECCV*, 2016.
[26] S. Meyer, A. Djelouah, B. McWilliams, A. Sorkine-Hornung, M. Gross, and C. Schroers, "Phasenet for video frame interpolation," in *CVPR*, 2018.
[27] S. Meyer, O. Wang, H. Zimmer, M. Grosse, and A. Sorkine-Hornung, "Phase-based frame interpolation for video," in *CVPR*, 2015.
[28] S. Niklaus, L. Mai, and O. Wang, "Revisiting adaptive convolutions for video frame interpolation," in *WACV*, 2021.
[29] T. Kalluri, D. Pathak, M. Chandraker, and D. Tran, "Flavr: Flow-agnostic video representations for fast frame interpolation," *arXiv:2012.08512*, 2020.
[30] W. Bao, W.-S. Lai, C. Ma, X. Zhang, Z. Gao, and M.-H. Yang, "Depth-aware video frame interpolation," in *CVPR*, 2019.
[31] W. Bao, W.-S. Lai, X. Zhang, Z. Gao, and M.-H. Yang, "Memc-net: Motion estimation and motion compensation driven neural network for video interpolation and enhancement," *arXiv:1810.08768*, 2018.
[32] L. Haopeng, Y. Yuan, and W. Qi, "Video frame interpolation via residue refinement," in *ICASSP*, 2020.
[33] Y.-L. Liu, Y.-T. Liao, Y.-Y. Lin, and Y.-Y. Chuang, "Deep video frame interpolation using cyclic frame generation," in *AAAI*, 2019.
[34] S. Niklaus and F. Liu, "Softmax splatting for video frame interpolation," in *CVPR*, 2020.
[35] J. Park, K. Ko, C. Lee, and C.-S. Kim, "Bmbc: Bilateral motion estimation with bilateral cost volume for video interpolation," in *ECCV*, 2020.

[36] X. Xu, S. Li, W. Sun, Q. Yin, and M.-H. Yang, "Quadratic video interpolation," in *NeurIPS*, 2019.
[37] Z. Huang, T. Zhang, W. Heng, B. Shi, and S. Zhou, "Rife: Real-time intermediate flow estimation for video frame interpolation," *arXiv:2011.06294*, 2020.
[38] J. Choi, J. Park, and I. S. Kweon, "High-quality frame interpolation via tridirectional inference," in *WACV*, 2021.
[39] F. A. Reda, D. Sun, A. Dundar, M. Shoeybi, G. Liu, K. J. Shih, A. Tao, J. Kautz, and B. Catanzaro, "Unsupervised video interpolation using cycle consistency," in *ICCV*, 2019.
[40] S. Bengio, Y. Bengio, J. Cloutier, and J. Gecsei, "On the optimization of a synaptic learning rule," in *Preprints Conf. Optimality in Artificial and Biological Neural Networks*. Univ. of Texas, 1992, pp. 6–8.
[41] S. Hochreiter, A. Younger, and P. Conwell, "Learning to learn using gradient descent," *ICANN 2001*, 2001.
[42] J. Schmidhuber, "Evolutionary principles in self-referential learning," *On learning how to learn: The meta-meta-... hook.) Diploma thesis, Institut f. Informatik, Tech. Univ. Munich*, 1987.
[43] ——, "Learning to control fast-weight memories: An alternative to dynamic recurrent networks," *Neural Computation*, vol. 4, no. 1, pp. 131–139, 1992.
[44] S. Thrun and L. Pratt, *Learning to learn*. Springer Science & Business Media, 2012.
[45] G. Koch, R. Zemel, and R. Salakhutdinov, "Siamese neural networks for one-shot image recognition," in *ICML Workshop*, 2015.
[46] J. Snell, K. Swersky, and R. Zemel, "Prototypical networks for few-shot learning," in *NIPS*, 2017.
[47] F. Sung, Y. Yang, L. Zhang, T. Xiang, P. H. Torr, and T. M. Hospedales, "Learning to compare: Relation network for few-shot learning," in *CVPR*, 2018.
[48] O. Vinyals, C. Blundell, T. Lillicrap, k. kavukcuoglu, and D. Wierstra, "Matching networks for one shot learning," in *NIPS*, 2016.
[49] T. Munkhdalai and H. Yu, "Meta networks," in *ICML*, 2017.
[50] B. N. Oreshkin, P. Rodriguez, and A. Lacoste, "Tadam: Task dependent adaptive metric for improved few-shot learning," in *NIPS*, 2018.
[51] T. Munkhdalai, X. Yuan, S. Mehri, and A. Trischler, "Rapid adaptation with conditionally shifted neurons," in *ICML*, 2018.
[52] A. Santoro, S. Bartunov, M. Botvinick, D. Wierstra, and T. Lillicrap, "Meta-learning with memory-augmented neural networks," in *ICLR*, 2016.
[53] S. Baker, D. Scharstein, J. P. Lewis, S. Roth, M. J. Black, and R. Szeliski, "A database and evaluation methodology for optical flow," *IJCV*, vol. 92, no. 1, p. 1–31, 2010.
[54] A. Antoniou, H. Edwards, and A. Storkey, "How to train your MAML," in *ICLR*, 2019.
[55] Z. Li, F. Zhou, F. Chen, and H. Li, "Meta-sgd: Learning to learn quickly for few shot learning," *arXiv:1707.09835*, 2017.
[56] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *CoRR*, vol. abs/1412.6980, 2014.
[57] A. Paliwal, "Pytorch implementation of super slomo," https://github.com/avinashpaliwal/Super-SloMo, 2018.
[58] S. Baik, S. Hong, and K. M. Lee, "Learning to forget for meta-learning," in *CVPR*, 2020.
[59] K. Soomro, A. R. Zamir, and M. Shah, "Ucf101: A dataset of 101 human actions classes from videos in the wild," *CRCV-TR-12-01*, 2012.
[60] S. Su, M. Delbracio, J. Wang, G. Sapiro, W. Heidrich, and O. Wang, "Deep video deblurring for hand-held cameras," in *CVPR*, 2017.
[61] A. Nichol, J. Achiam, and J. Schulman, "On first-order meta-learning algorithms," *arXiv:1803.02999*, 2018.
[62] A. Rajeswaran, C. Finn, S. Kakade, and S. Levine, "Meta-learning with implicit gradients," in *NeurIPS*, 2019.

Myungsub Choi received the BS degree in electrical and computer engineering from Seoul National University (SNU), Seoul, Korea, in 2013, and the PhD degree in electrical engineering and computer science from Seoul National University in 2021. He is currently a visiting researcher at Google Research. He was a research intern at Snap Inc. in 2018. He has won the Runner-Up Award in AIM 2019 Challenge on Video Temporal Super-Resolution. He is interested in input-adaptive methodologies such as attention models and meta-learning that are applicable for various computer vision problems including video frame interpolation and super-resolution. He is a member of the IEEE.

Janghoon Choi received the BS degree in electrical and computer engineering from Seoul National University (SNU), Seoul, Korea, in 2013, and the PhD degree in electrical engineering and computer science from Seoul National University in 2021. He was a postdoctoral researcher at ASRI, Seoul National University in 2021. He is currently an assistant professor with the College of Computer Science at Kookmin University. He is interested in computer vision problems including visual tracking and meta-learning applications. He is a member of the IEEE.

Sungyong Baik received the BASc degree in engineering science with a major in electrical and computer engineering from University of Toronto, Toronto, Canada, in 2015. He is currently working towards the PhD degree in electrical and computer engineering from Seoul National University. He was a research intern at Facebook Reality Labs in 2019. He is interested in few-shot learning, meta-learning, and its applications. He is a student member of the IEEE.

Tae Hyun Kim received the BS and MS degrees from the Department of Electrical Engineering, KAIST, Daejeon, Korea, in 2008 and 2010, respectively, and the PhD degree in electrical and computer engineering from Seoul National University (SNU), Seoul, Korea, in 2016. He worked as a postdoctoral at the Max Planck Institute for Intelligent Systems. He is currently an assistant professor with the Department of Computer Science at Hanyang University, Seoul, Korea. His research interests include low level computer vision and computational imaging. He is a member of the IEEE.

Kyoung Mu Lee received the BS and MS degrees in control and instrumentation engineering from Seoul National University (SNU), Seoul, Korea, in 1984 and 1986, respectively, and the PhD degree in electrical engineering from the University of Southern California, in 1993. He is currently a professor of the Department of ECE and is the director of the Interdisciplinary Program in Artificial Intelligence in the Graduate School of SNU. He has received several awards, in particular, the medal of merit and the Scientist of Engineers of the month award from the Korean Government in 2020 and 2018, respectively, the Most Influential Paper over the Decade Award by the IAPR Machine Vision Application in 2009, the ACCV Honorable Mention Award in 2007, the Okawa Foundation Research Grant Award in 2006, the Distinguished Professor Award from the college of Engineering of SNU in 2009, and both the Outstanding Research Award and the Shinyang Engineering Academy Award from the College of Engineering of SNU in 2010. He has served as an Associate Editor in Chief (AEIC) and an Associate Editor of the IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI), an Associate Editor of the Machine Vision Application (MVA) Journal and the IPSJ Transactions on Computer Vision and Applications (CVA), the IEEE Signal Processing Letters (SPL), and an Area Editor of the Computer Vision and Image Understanding (CVIU). He also has served as a general chair of ICCV2019, ACMMM2018, and ACCV2018, a program chair of ACCV2012, a track chair of ICPR2020 and ICPR2012, and an area char of CVPR, ICCV and ECCV many times. He was a distinguished lecturer of the Asia-Pacific Signal and Information Processing Association (APSIPA) for 2012-2013. More information can be found on his homepage http://cv.snu.ac.kr/kmlee.