Generalizable Implicit Motion Modeling for Video Frame Interpolation

Zujin Guo, Wei Li, Chen Change Loy
S-Lab, Nanyang Technological University
{zujin.guo, wei.l, ccloy}@ntu.edu.sg
https://gseancdat.github.io/projects/GIMMVFI

Abstract

Motion modeling is critical in flow-based Video Frame Interpolation (VFI). Existing paradigms either consider linear combinations of bidirectional flows or directly predict bilateral flows for given timestamps without exploring favorable motion priors, thus lacking the capability of effectively modeling spatiotemporal dynamics in real-world videos. To address this limitation, in this study, we introduce Generalizable Implicit Motion Modeling (GIMM), a novel and effective approach to motion modeling for VFI. Specifically, to enable GIMM as an effective motion modeling paradigm, we design a motion encoding pipeline to model spatiotemporal motion latent from bidirectional flows extracted from pre-trained flow estimators, effectively representing input-specific motion priors. Then, we implicitly predict arbitrary-timestep optical flows within two adjacent input frames via an adaptive coordinate-based neural network, with spatiotemporal coordinates and motion latent as inputs. Our GIMM can be easily integrated with existing flow-based VFI works by supplying accurately modeled motion. We show that GIMM performs better than the current state of the art on standard VFI benchmarks.

1   Introduction

Video Frame Interpolation (VFI) is a fundamental task in computer vision, which involves generating intermediate frames between two adjacent video frames. This technique is crucial for various practical applications, including novel view synthesis [57, 14, 30], video generation [47], and video compression [52]. This task is highly challenging due to the complex motions typically found in real-world videos. To address this, recent research [29, 55, 42, 40, 20] has focused on flow-based frameworks, which have shown substantial success. Generally, these frameworks for VFI involve two main phases: 1) transforming the input frames based on estimated optical flows, and 2) merging and enhancing the warped frames to produce intermediate frames. Consequently, the accuracy of flow estimation is crucial for the fidelity of the synthesized frames.

Accurately modeling flow from two distant frames is challenging due to the complexities of capturing subtle and dynamic movements caused by varying motion speeds, object occlusions, and lighting conditions. A commonly used approach in the literature combines bidirectional flows derived from input frames [22, 40, 45, 53, 39] (Figure 1(a)). This method assumes *overlapped* and linear motion for flow estimation, which does not accurately reflect real-world dynamics. Several studies [20, 29, 55, 42] directly predict bilateral flows based on intermediate discrete timestamps (Figure 1(b)). These approaches model the correlation between frame motions and timestamps without leveraging motion priors within input frames; they thus fall short of capturing complex spatial-temporal changes and handling occluded regions and unexpected deformations. Moreover, this discrete-time-based modeling paradigm is ineffective for arbitrary-time interpolation.

38th Conference on Neural Information Processing Systems (NeurIPS 2024).
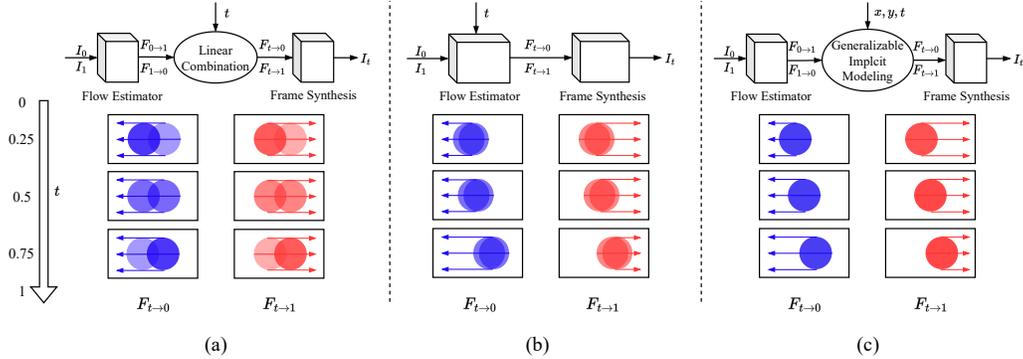
Figure 1: **Schematic of motion modeling paradigms** in video frame interpolation. (**a**) A naïve *linear combination* of bidirectional flows $F_{0\to1}$, $F_{1\to0}$ (i.e., flows between input frames) may lead to ambiguous and coarse motion estimation due to strong overlapped and linear assumptions. (**b**) A *time-condition-based modeling* approach may predict suboptimal bilateral flows $F_{t\to0}$, $F_{t\to1}$ (i.e., flows between estimated and input frames), capturing spatiotemporal changes for moving objects ineffectively. (**c**) Our *generalizable implicit motion modeling* properly represents spatiotemporal dynamics across videos and predict better bilateral flows via an adaptive coordinate-based neural network.

In this study, we explore a more effective and generalizable implicit approach to motion modeling for VFI. Inspired by the success of implicit neural representations [48] in encoding complex high-dimensional data such as 2D images [5], 3D scenes [34], and videos [3], we propose to implicitly model optical flows between two adjacent video frames using coordinate-based neural networks. This implicit paradigm can directly take arbitrary spatiotemporal coordinates as inputs and effectively decode the desired space-time outputs, making it a promising framework for learning highly dynamic optical flows in real-world videos. However, leveraging implicit neural networks for effective motion modeling poses challenges. First, standard implicit neural networks typically perform per-instance modeling, optimizing model parameters for a single specific input. This limitation restricts their applicability across different input video frames. Therefore, we need to develop a more adaptive implicit model capable of capturing motions in any given video. Second, efficiently integrating spatiotemporal information within implicit neural networks is complex, especially when dealing with the intricate motions occurring between two video frames. This necessitates designing appropriate implicit neural architectures that can accurately predict and represent both the spatial and temporal dynamics in videos.

To this end, we propose a novel generalizable implicit flow encoding for motion modeling in VFI, called Generalizable Implicit Motion Modeling (GIMM) (Figure 1(c)). Our method only assumes the availability of bidirectional flows ($F_{0\to1}$, $F_{1\to0}$) of two input frames obtained from a pre-trained optical flow estimator (e.g., RAFT [50], FlowFormer [19]). The input flows can be noisy as this prior will be refined to estimate the bilateral flows ($F_{t\to0}$, $F_{t\to1}$) between arbitrary intermediate timestamps. Our method is unique in that it can represent input-specific motion priors effectively. For instance, it can accurately capture the complex dynamics of a somersault. This is achieved by introducing a motion encoding pipeline to extract spatiotemporal motion latent from the bidirectional flows. Our method can further enable frame interpolation at *arbitrary timestamps*, thanks to the adaptive coordinate-based neural network that takes spatiotemporal coordinates and motion latent as inputs. This capability allows our method to generate frames at various temporal granularities, providing flexibility and precision in video frame interpolation.

Our contributions are summarized as follows: We present an effective motion modeling paradigm for video frame interpolation characterized by a novel generalizable implicit motion modeling framework. Our GIMM is capable of accurately predicting optical flow for arbitrary timesteps between two adjacent video frames at any resolution, allowing seamless integration with existing flow-based VFI methods. We demonstrate the advantages of our GIMM in motion modeling for arbitrary-timestep VFI tasks, achieving state-of-the-art performances on various benchmarks.

2

2 Related Work

Video frame interpolation. Conventional VFI studies primarily rely on either direct frame synthesis via convolutional networks [11, 1, 24] or interpolation using dynamic kernels with learnable weights and offsets [37, 38, 41, 9, 8, 12, 28]. Recent approaches have shifted towards flow-based methods to synthesize frames at desired timesteps, where motion modeling plays a crucial role [20, 29, 54, 26, 49, 17]. Some flow-based methods combine estimated bidirectional flows between input frames [22, 40, 45, 53, 39, 1], often leading to inaccurate motion predictions, especially in occluded areas. This simplistic combination results in ambiguous and coarse motion estimation, causing object shifts in interpolated frames. Several recent approaches [20, 29, 55, 20, 32] address these issues by directly predicting the desired motion within an end-to-end framework conditioned on timesteps, showing impressive results in synthesized frames. However, these methods mainly rely on discrete-time-based fitting of variable relationships between motion and timesteps, making it challenging to achieve consistent, continuous interpolation outcomes.

Implicit neural representations. Implicit Neural Representations (INRs) have been shown effective in modeling complex, high-dimensional data for various applications, including video compression [3], novel view synthesis [34, 30], and image super-resolution [5]. Typically, INRs learn a continuous mapping from a set of coordinates to a specific signal using a coordinate-based neural network to encode data implicitly. The flexible and expressive modeling capabilities of INRs motivate us to explore their use to encode intricate motions and capture subtle, dynamic movements of objects in real-world videos. A recent work related to ours is IFE [13]. While both approaches consider implicit flow encoding, our method significantly differs from IFE. Unlike IFE, which focuses on per-scene encoding—where each coordinate-based network is parameterized by a specific video instance—we aim to develop a generalizable motion modeling approach that can be applied across different videos.

Generalizable INRs. Recent works [5, 44, 10, 21, 15, 7, 4] further extend INRs for generalizable encoding by conditioning coordinate-based neural networks with additional instance-specific inputs. For example, some approaches [6, 27, 25] employ Transformers [51] as meta-learners to predict instance-specific weights or modulation features for coordinate-based neural networks at high computational costs. Several notable studies [4, 7, 45] leverage generalizable INRs for video encoding to facilitate video interpolation and super-resolution, mainly focusing on directly learning implicit space-time continuous neural representations from video. In contrast, we aim to explore effective motion modeling paradigms to improve intermediate frame synthesis for flow-based VFI. To our knowledge, we make the first attempt to utilize generalizable INRs for motion modeling in the context of VFI.

3 Method

Given a pair of adjacent video frames $I_0, I_1 \in \mathbb{R}^{H \times W \times 3}$ with timesteps $\{0, 1\}$, a general flow-based video frame interpolation process is defined as follows:

$$F_t = \mathcal{G}(I_0, I_1, t), \tag{1}$$

$$I_t = \mathcal{H}(F_t, I_0, I_1), \forall t \in [0, 1], \tag{2}$$

where $\mathcal{G}$ denotes a motion modeling (or flow estimation) module, and $\mathcal{H}$ indicates a frame interpolation process under the guidance of estimated motion F_t at a timestep of $t \in [0, 1]$ for input frames I_0 and I_1. In this work, we mainly focus on studying an effective motion modeling framework $\mathcal{G}$ for flow-based VFI.

3.1 Generalizable Implicit Motion Modeling

The goal of GIMM is to estimate bilateral flows $F_{t \to 0}, F_{t \to 1}$ for any timestep $t \in [0, 1]$, deriving from initial bidirectional flows $F_{0 \to 1}, F_{1 \to 0}$:

$$F_{t \to 0}, F_{t \to 1} = \mathcal{G}(F_{0 \to 1}, F_{1 \to 0}, t), \forall t \in [0, 1], \tag{3}$$

where $F_{0 \to 1}, F_{1 \to 0}$ are predicted from a pre-trained optical flow estimator (e.g., RAFT [50], Flow-Former [19]) with input frames I_0 and I_1. Figure 2 depicts the overall generalizable motion modeling framework of GIMM. Motivated by the great success of INRs in modeling complex video data [7, 4, 13, 34], GIMM uses an adaptive coordinate-based neural network for continuous motion

3
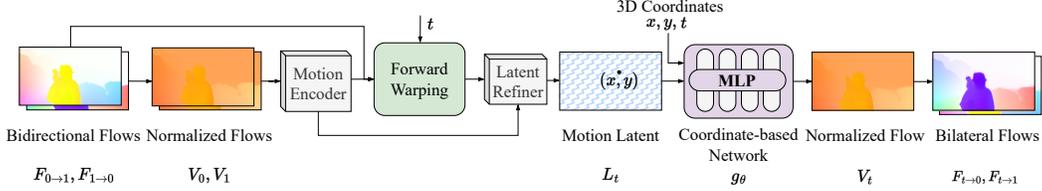
Figure 2: Our GIMM first transforms initial bidirectional flows $F_{0\to1}, F_{1\to0}$ as normalized flows V_0, V_1. The motion encoder extracts motion features K_0, K_1 from V_0, V_1, respectively. K_0, K_1 are then forward warped at a given timestep t using bidirectional flows to obtain the warped features $K_{t\to0}, K_{t\to1}$. We pass the warped and initial motion features into a Latent Refiner that outputs motion latent L_t, representing motion information at t. Conditioned on $L_t(x, y)$, the coordinate-based network g_θ predicts the corresponding normalized flow V_t with 3D coordinates $\mathbf{x} = (x, y, t)$. For interpolation usage, V_t is then transferred into bilateral flows $F_{t\to0}, F_{t\to1}$ through denormalization.

modeling. Unlike the existing IFE [13] that performs per-scene optimization and requires test-time learning for different videos, our GIMM takes additional instance-specific motion latent inputs L_t to enhance model generalizability across various input videos.

Flow normalization. Following IFE [13], we perform normalization for the initial bidirectional flows as follows:

$$V_0 = \phi(F_{0\to1}), V_1 = \phi(-F_{1\to0}), \tag{4}$$
$$V_t = \phi(F_{t\to1} - F_{t\to0}), \tag{5}$$

where ϕ is a scale operator. This reversible normalization process aligns the scale and temporal direction of input bidirectional flows $F_{0\to1}, F_{1\to0}$ with output bilateral flows $F_{t\to0}, F_{t\to1}$, allowing the normalized flows $V_i \in [0, 1]^{H\times W\times 2}, i \in 0, 1$ to be effectively encoded in subsequent implicit motion modeling in GIMM.

Motion latent. To achieve generalizable implicit modeling of motion dynamics, we learn implicit neural representations for motions with conditions on instance-specific motion latent inputs $L_t(x, y)$, which provides instance-specific motion priors at spatiotemporal coordinates $\mathbf{x} = (x, y, t)$ for target motion $V_t(x, y)$. Specifically, we introduce a Motion Encoder to extract motion features K_i from normalized flows V_i. To maintain spatiotemporal consistencies of modeled motion, we derive time-dependent motion features K_t for given timestep t from both motion features K_0 and K_1. To realize this, we apply forward warping $\overrightarrow{\omega}$ process to map every location of motion features K_i to the target timestep t:

$$K_{i\to t} = \overrightarrow{\omega}(K_i, F_{i\to t}, Z_i). \tag{6}$$

Here, $K_{i\to t}$ are warped features with timesteps $i \in \{0, 1\}$ that integrate the correspondences from source (input) timesteps to the target timestep, Z_i represents splatting weights, and $F_{i\to t}$ denotes scaled bidirectional flows at timestep t. We compute scaled bidirectional flows $F_{i\to t}$ as follows:

$$F_{i\to t} = \begin{cases} (t - 0) \cdot F_{0\to1}, & \text{if } i = 0 \\ (1 - t) \cdot F_{1\to0}, & \text{if } i = 1 \end{cases} \tag{7}$$

To mitigate warping errors in multi-to-one cases or occluded regions, we calculate splatting weights Z_i using flow consistency U_{flow}^i and variance U_{var}^i metrics [35, 36] as follows:

$$Z_i = \frac{1}{1 + \alpha_{flow} \cdot U_{flow}^i} + \frac{1}{1 + \alpha_{var} \cdot U_{var}^i}, \tag{8}$$

where α_{flow} and α_{var} are learnable parameters. To obtain the final motion latent L_t, we concatenate the warped features $\texttt{Concat}(K_{0\to t}, K_{1\to t})$ as the coarse motion latent at timestep t and further refine via a Latent Refiner module to deal with potential information loss and ambiguous motion in the forward warping process. Both the Latent Refiner and the Motion Encoder mentioned above are structured as shallow convolutional networks. We provide their network configurations and detailed calculations for two flow metrics U_{flow}^i and U_{var}^i in the supplementary.

4
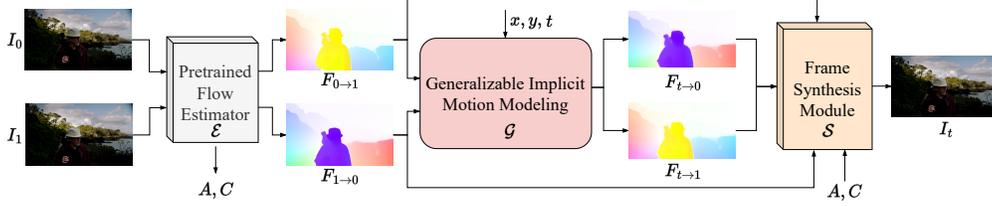
Figure 3: An overview of GIMM-VFI architecture. GIMM-VFI employs a pre-trained flow estimator, $\mathcal{E}$, to predict bidirectional flows $(F_{0\to1}, F_{1\to0})$ and extracts context features A as well as correlation features C from the input frames (I_0, I_1). Given the timestep t, a generalizable implicit motion modeling (GIMM) module $\mathcal{G}$ (detailed in Figure 2) takes the bidirectional flows as inputs and predicts bilateral flows $(F_{t\to0}, F_{t\to1})$, which are then passed into a frame synthesis module $\mathcal{S}$, together with extracted features (A, C), to synthesize the target frame I_t.

Implicit motion prediction. To realize INRs for generalizable motion modeling, we devise an adaptive coordinate-based network g_θ for implicit and continuous motion encoding, which maps spatiotemporal coordinates $\mathbf{x} \in [-1, 1]^{H \times W \times 3}$ along with the corresponding D-dimension motion latent code $L_t \in \mathbb{R}^{H \times W \times D}$ to predicted normalized flows $\hat{V}_t$:

$$\hat{V}_t = g_\theta(\mathbf{x}, L_t). \tag{9}$$

The predicted normalized flow V_t can be converted into predicted bilateral flows $F_{t\to0}, F_{t\to1}$ via the reverse flow normalization process for VFI. See the supplementary for the details of the coordinate-based network.

Optimization. In practice, we optimize the GIMM module by minimizing a Mean-Square-Error (MSE) loss between predicted $(\hat{V}_t)$ and ground-truth (V_t) normalized optical flows:

$$\mathcal{L}_{gimm}(V_t, \hat{V}_t) = \frac{1}{H \times W} \sum_{k=0}^{H-1} \sum_{l=0}^{W-1} ||V_t(k, l) - \hat{V}_t(k, l)||_2. \tag{10}$$

3.2 Integrating GIMM with Frame Interpolation

GIMM-VFI. Given two adjacent video frames I_i with $i \in \{0, 1\}$, we start with a pre-trained optical flow estimator (e.g., RAFT [50], FlowFormer [19]) to extract context features A_i, correlation features C_i, and initial bidirectional flows $F_{0\to1}$ and $F_{1\to0}$. We first predict bilateral flows $F_{t\to i}$ for a given timestep t from initial bidirectional flows using a pre-trained GIMM module. The predicted bilateral flows $F_{t\to i}$, context features A_i, and correlation features C_i are then passed into a frame synthesis module $\mathcal{S}$ (adapted from the AMT [29]), which refines the input flows and generates warping masks $M_t \in [0, 1]^{H \times W}$. Following previous flow-based VFI studies [29, 55, 42, 20], we obtain warped images $I_{t\to i}$ via backward warping $\overleftarrow{\omega}$ under the guidance of predicted bilateral flows $F_{t\to i}$ from input frames I_0 and I_1:

$$I_{t\to i} = \overleftarrow{\omega}(I_i, F_{t\to i}), \tag{11}$$

$$I_{t\to i}(x, y) = I_i(x + F_{t\to i}^h(x, y), y + F_{t\to i}^v(x, y)), \tag{12}$$

where $F_{t\to i}^h, F_{t\to i}^v$ represents horizontal and vertical motion of $F_{t\to i}$, respectively. To generate the final interpolated image $\hat{I}_t$, the warped images $I_{t\to0}, I_{t\to1}$ are fused with the warping mask M_t as follows:

$$\hat{I}_t = M_t \cdot I_{t\to0} + (1 - M_t) \cdot I_{t\to1}. \tag{13}$$

See the supplementary for the details of the frame synthesis module.

Overall objectives. We use the following objective function for VFI:

$$\mathcal{L}_{interp}(I_t, \hat{I}_t) = \mathcal{L}_{lap}(I_t, \hat{I}_t) + \mathcal{L}_{char}(I_t, \hat{I}_t) + \mathcal{L}_{census}(I_t, \hat{I}_t), \tag{14}$$

where $\mathcal{L}_{lap}$, $\mathcal{L}_{char}$, and $\mathcal{L}_{census}$ denote the Laplacian loss [55, 36], Charbonnier loss [29, 2] , and census loss [29, 33], respectively. In addition, to preserve the motion modeling in the pre-trained

5

Table 1: **Comparisons of different motion modeling methods.** We assess the modeled motion on Vimeo-Triplet-Flow (VTF) and Vimeo-Septuplet-Flow (VSF) by employing PSNR and EPE metrics. Additionally, we demonstrate their impact on the interpolation task by presenting the PSNR values of their interpolation results on the 'hard' split of the SNU-FILM-arb dataset. We denote the calculated PSNR values as $PSNR_f$ for optical flows in the motion assessment and $PSNR_i$ for interpolated images in the interpolation task.

Motion method	Vimeo-Triplet-Flow (VTF)		Vimeo-Septuplet-Flow (VSF)		SNU-FILM-arb-Hard
	$PSNR_f\uparrow$	$EPE\downarrow$	$PSNR_f\uparrow$	$EPE\downarrow$	$PSNR_i\uparrow$
Linear	35.03	0.44	30.09	2.87	32.42
Forward Warp	32.80	0.47	28.22	3.38	32.31
End-to-End	29.23	1.02	25.99	5.12	32.28
BMBC [39]	28.89	0.95	23.19	8.23	28.51
GIMM (-VFI-R)	**37.56**	**0.34**	**30.45**	**2.68**	**32.62**

GIMM module, we reconstruct and supervise the flows $\hat{V}_0, \hat{V}_1$ with pseudo ground truth V_0, V_1. This objective $\mathcal{L}_{rec}$ is defined as $||V_0 - \hat{V}_0||_2 + ||V_1 - \hat{V}_1||_2$. The overall objective $\mathcal{L}$ is as follows:

$$\mathcal{L} = \mathcal{L}_{interp}(I_t, \hat{I}_t) + \lambda_{rec}\mathcal{L}_{rec}, \qquad (15)$$

where λ_{rec} is the hyperparameter. We optimize the entire GIMM-VFI model that contains the pre-trained flow estimator, pre-trained GIMM module, and frame synthesis module.

4 Experiments

We present quantitative and qualitative evaluations of our motion modeling method GIMM in Section 4.1, and the corresponding interpolation method (GIMM-VFI) in Section 4.4. Specifically, we evaluate both motion quality and performance on the downstream interpolation task. We compare GIMM-VFI with current state-of-the-art VFI methods on arbitrary-timestep interpolation.

Implementation details. We train the GIMM model on the training split of Vimeo90K [54] triplets dataset using optical flows extracted by off-the-shelf flow estimators. Our GIMM-VFI is trained on the complete Vimeo90K septuplet dataset. Specifically, we implement two variants of GIMM-VFI, using two different flow estimators: the RAFT [50] and FlowFormer [19], designated as GIMM-VFI-R and GIMM-VFI-F, respectively. More details are provided in the supplementary materials.

4.1 Motion Modeling

VTF and VSF benchmarks. To assess the modeled motion quality with GIMM, we use Flow-former [19] to produce pseudo ground truth motion. Specifically, we establish two motion evaluation benchmarks: Vimeo-Triplet-Flow (VTF) and Vimeo-Septuplet-Flow (VSF). These benchmarks are derived from the triplet and septuplet splits of the Vimeo90K [54] test set, tailored for evaluating 2X and 6X motion modeling, respectively.

Baselines. We compare GIMM with other motion modeling approaches, including *Linear*, *Forward Warp*, *End-to-End*, and BMBC [39]. Specifically, *Linear* replaces our GIMM module with the linear approximation strategy [22]. Similarly, *Forward Warp* substitutes GIMM with a forward warping strategy [35]. Meanwhile, the *End-to-End* denotes the strategy that directly predicts motion at arbitrary timesteps through an end-to-end fitting. In this end-to-end setting, we select the current state-of-the-art EMA-VFI [55] as the representative method. Additionally, we make further comparisons with BMBC [39], which is supposed to model complex motion effectively through several stages of motion predictions and motion refinements. For fair comparisons, we employ RAFT [50] as the flow estimator for different motion modeling methods.

Evaluation settings. We measure the modeled motion quality on both VTF and VSF benchmarks by calculating PSNR on normalized flows and End-Point-Error (EPE) on the unscaled flows. For the downstream interpolation task, we calculate PSNR on the 'hard' split of the SNU-FILM-arb benchmark (SNU-FILM-arb-Hard). Details about SNU-FILM-arb are presented in Section 4.4.

6

| Overlaid Input | Linear | Forward Warp | End-to-End | BMBC | GIMM-VFI-R | Ground Truth |

Figure 4: Qualitative comparisons of different motion modeling methods on SNU-FILM-arb-Hard. All the results are predicted at $t = 0.75$, and ground truth flows are obtained by FlowFormer [19].

Results. We provide quantitative comparisons of our GIMM and baselines in Table 1. We first report the results of motion quality on VTF and VSF. We find that GIMM can continuously model motions in videos. GIMM predicts the best flows on both the VTF benchmark (37.56dB PSNR/ 0.34 EPE) and the VSF benchmark (30.45dB PSNR/ 2.68 EPE) that involves unseen timesteps during training. Moreover, our study highlights the importance of motion priors in both motion modeling and the downstream interpolation task. The *Linear*, *Forward Warp*, and our GIMM methods that leverage motion priors from RAFT [50] can provide better motion and interpolation performances than the *End-to-End* and BMBC [39] across all benchmarks. Notably, GIMM benefits the interpolation task the most and achieves the highest PSNR of 32.62 dB on SNU-FILM-arb-Hard. While motion modeling plays a critical role in interpolation, other modules also impact interpolation outcomes. The *End-to-End* method has a narrower performance gap with other advanced motion modeling approaches on the interpolation benchmark than on motion modeling benchmarks, likely due to its more complex and extensive feature extraction and frame synthesis modules.

Visualizations. Besides the quantitative evaluation of motion modeling, we also qualitatively evaluate them on SNU-FILM-arb-Hard, as shown in Figure 4. Since SNU-FILM-arb-Hard requires 8X interpolation for input frames of large motion, classical methods like BMBC [39] may struggle to model the intermediate motion. In contrast, we observe the capacity of GIMM for accurate motion modeling at arbitrary timesteps. Take the Somersault in Figure 4 for example, the ankle flow and the corresponding interpolation generated by GIMM have the best consistency with the ground truth, pointing to the same direction. Moreover, GIMM is capable of reducing ambiguities when interpolating moving objects. We can observe clearer silhouettes with fewer blurs in both the 'dog-leg' and 'dog-head' cases in Figure 4.

4.2 Ablation Study on Motion Model Choices

As our core contribution, we present an effective motion modeling paradigm for video frame interpolation, characterized by a novel generalizable implicit motion modeling framework (GIMM). In this section, we further investigate the necessity of implicit neural representations (INRs) and approaches

7

Table 2: **Quantitative comparisons of different motion model choices.**

Model Variants	Vimeo-Triplet-Flow (VTF)		Vimeo-Septuplet-Flow (VSF)		Param.(M)
	PSNR↑	EPE↓	PSNR↑	EPE↓	
Meta-learning (INR)	30.19	0.88	24.50	6.80	43.92
GIMM (U-Net)	36.96	0.39	29.96	2.90	4.27
GIMM (INR)	**37.56**	**0.34**	**30.45**	**2.68**	**0.25**

to generalizable INRs. The experiments are conducted on the motion modeling benchmarks, *i.e.,* Vimeo-Triplet-Flow (VTF) and Vimeo-Septuplet-Flow (VSF).

Necessity of INRs. A straightforward replacement of INRs in our GIMM is a timestep-conditioned U-Net as in diffusion literature [43]. We substitute the INRs with a U-Net of a comparable number of channels to ensure fair comparisons. In Table 2, replacing INR with U-Net results in worse performance, especially on the 6X motion modeling benchmark VSF. This demonstrates the strong continuous modeling ability of INR. Besides, GIMM with INR has a much lighter architecture, improving efficiency. Therefore, it is necessary and proper to use INR for our motion modeling.

Approaches to generalizable INRs. To realize generalizable implicit modeling, an alternative way is to modulate the weights of INRs with meta-learners [6, 25]. Following [25], we employ Transformers [51] as the meta-learner. Compared to GIMM, this meta-learning approach performs significantly worse across all benchmarks while introducing a model with over 170X more parameters.

4.3 Ablation Study on GIMM

In this section, we investigate the effectiveness of model designs in GIMM in Table 3. The experiments are conducted on the VTF and VSF benchmarks.

Forward warping. We substitute forward warping operation in GIMM with a straightforward linear combination of the input motion features (*Non-Fwarp*). In Table 3, we observe a significant performance drop with this modification. For example, the EPE on VSF increases by 0.20. This result highlights the importance of forward warping for GIMM's continuous motion modeling.

Implicit modeling. In GIMM, the motion can be modeled even without the presence of implicit modeling. We conduct experiments without using any coordinates (*Non-Imp*). In Table 3, this variant yields performance gains, with reductions of 0.05 and 0.13 in the EPEs on VTF and VSF respectively, highlighting the importance of implicit modeling in GIMM.

Motion encoder. We leverage the motion encoder in our GIMM to extract features from flows, alleviating the negative impacts brought by the possible bias and noises in the estimated flows. To justify this design, we remove the motion encoder for a direct comparison (*Non-ME*). As a result, we observe increases in error on both benchmarks, *e.g.*, the EPE on VSF rises 0.17. Therefore, the motion encoder benefits the motion modeling in GIMM.

Latent refinement. To validate the efficacy of latent refinement in GIMM, we conducted experiments excluding the latent refiner (*Non-Refiner*). This leads to notable declines of 0.53 dB and 0.43 dB in PSNRs across both benchmarks. Thus, refining the motion latent proves essential for accurate motion modeling.

Spatial coordinates. Since the modeled motion consists of spatiotemporal changes, we use 3D spatiotemporal coordinates in the implicit modeling of GIMM. To assess the necessity of spatial coordinates, we replace 3D coordinates with temporal coordinates (*T-coord only*). This removal causes a 0.16 dB drop of PSNR on Vimeo-Triplet-Flow and a 0.06 increase of EPE on Vimeo-Septuplet-Flow. This emphasizes the crucial role of spatial coordinates in implicit modeling.

4.4 Arbitrary-timestep Video Frame Interpolation

SNU-FILM-arb benchmark. We introduce the SNU-FILM-arb benchmark for a more generalized evaluation, to facilitate research on the arbitrary-timestep frame interpolation task. Specifically, we incorporate the SNU-FILM dataset [11] that encompasses 310 clips from 31 videos captured at 240 fps, with heights ranging from 384 to 1280 pixels and widths from 368 to 720 pixels. Different from

8

Table 3: **Quantitative comparisons of different model variants.** For each model variant, we evaluate its motion modeling performance on Vimeo-Triplet-Flow and Vimeo-Septuplet-Flow, respectively. We adopt EPE and PSNR as the metrics for motion quality.

Model Variants	Vimeo-Triplet-Flow (VTF)		Vimeo-Septuplet-Flow (VSF)	
	PSNR↑	EPE↓	PSNR↑	EPE↓
Non-Fwarp	37.07	0.37	30.09	2.88
Non-Imp	37.04	0.39	30.11	2.81
Non-ME	37.05	0.42	30.26	2.85
Non-Refiner	37.03	0.37	30.02	2.77
T-coord only	37.40	0.36	30.39	2.74
Full	**37.56**	**0.34**	**30.45**	**2.68**

Table 4: **Quantitative results for arbitrary-timestep interpolation.** We report the quantitative metrics reported as PSNR↑/LPIPS↓/FID↓, with the best results highlighted in **boldface** and the second best results in underline. The analysis categorizes methods into two groups: non-INR methods (first half) and INR-based methods (second half). We employ a '-' symbol to denote 'out-of-memory' scenarios, and a dagger ('†') to indicate that the results are drawn from prior studies [20, 18, 55].

Method	XTest		SNU-FILM-arb		
	2K	4K	Medium (4X)	Hard (8X)	Extreme (16X)
RIFE [20]	31.43† / 0.126 / 11.99	30.58† / 0.152 / 13.52	36.33 / 0.038 / 6.65	31.87 / 0.072 / 11.99	27.21 / 0.134 / 19.82
M2M [18]	32.13† / 0.098 / 9.25	30.88† / 0.158 / 8.67	36.56 / 0.036 / 5.98	31.92 / 0.061 / 10.13	27.14 / 0.112 / 17.37
IFRNet [26]	31.53† / 0.108 / 23.93	30.46† / 0.164 / 23.75	34.88 / 0.046 / 9.92	31.15 / 0.066 / 11.65	26.32 / 0.115 / 16.91
AMT [29]	28.88 / 0.153 / 13.92	28.17 / 0.187 / 13.97	34.49 / 0.072 / 9.25	31.03 / 0.089 / 10.34	26.44 / 0.136 / 14.72
UPR-Net [23]	31.16 / 0.104 / 10.75	30.50 / 0.154 / 9.45	36.78 / 0.033 / 6.09	31.96 / 0.064 / 9.93	27.14 / 0.111 / 16.76
EMA-VFI [55]	32.53 / **0.097** / 7.21	31.21 / 0.156 / 8.61	36.65 / 0.041 / 7.07	32.28 / 0.074 / 12.17	27.72 / 0.130 / 19.58
CURE [45]	30.24 / 0.111 / 26.42	-	36.09 / 0.035 / 6.98	31.32 / 0.063 / 12.72	26.61 / 0.114 / 22.62
GIMM-VFI-R	32.71 / 0.113 / **6.52**	31.88 / 0.149 / **6.49**	37.02 / 0.033 / 5.89	**32.62** / 0.060 / **9.59**	27.99 / 0.110 / **16.45**
GIMM-VFI-F	**32.91** / 0.103 / 6.74	**31.97** / **0.142** / 6.58	**37.03** / **0.031** / 5.86	32.56 / **0.059** / 9.95	**28.01** / **0.109** / 16.79

its original setting of 2X interpolation, we conduct 4X, 8X, and 16X interpolation evaluations on the original 'Medium', 'Hard', and 'Extreme' subsets of SNU-FILM, respectively.

Evaluation settings. We calculate PSNR and perceptual indicators, e.g., LPIPS [56], FID [16], on both SNU-FILM-arb and X4K-1000FPS (XTest) [46]. XTest comprises 15 clips from 15 videos for 8X interpolation. Following the established protocol from prior studies [18, 55], we assess interpolation quality at both 2K and 4K resolutions. For fair comparisons, we disable test-time augmentations during evaluations.

Current advanced VFI methods. We compare GIMM-VFI with various advanced interpolation methods. For non-INR methods, we include RIFE [20], IFRNet [26], M2M [18], AMT [29], UPR-Net [23], and EMA-VFI [55]. We also make comparisons with the INR-based method CURE [45].

Results. We present the quantitative results of GIMM-VFI on arbitrary-timestep interpolation benchmarks in Table 4. Our proposed method achieves high-quality continuous interpolation across various timesteps (i.e., 4X, 8X, 16X). In terms of PSNR, we observe that our RAFT-based [50] method GIMM-VFI-R achieves significant improvements of **0.18** dB on XTest-2K, **0.67** dB on XTest-4K, and approximately **0.30** dB on each subset of the SNU-FILM-arb, in comparison with the previous state-of-the-art method EMA-VFI [55]. For perceptual metrics, *i.e.,* LPIPS and FID, our proposed GIMM-VFI still outperforms previous methods on most benchmarks, delivering competitive performance. These observations demonstrate that our GIMM can offer effective motion modeling for the arbitrary-timestep VFI task.

Visualizations. Besides the quantitative evaluations, we further qualitatively compare both GIMM-VFI-R and GIMM-VFI-F with existing VFI techniques, as illustrated in Figure 5. Our methods achieve better interpolation across various timesteps. For example, our methods maintain the integrity of moving object silhouettes (3rd case). Moreover, GIMM-VFI-R and GIMM-VFI-F preserve detailed textures within both occluded regions and moving objects from significant deformations, *e.g.,* the rectangle pole tag(1st case), truck (2nd case), swing (3rd case).

9

| Overlaid Input | CURE | EMA-VFI | GIMM-VFI-R | GIMM-VFI-F | Ground Truth |
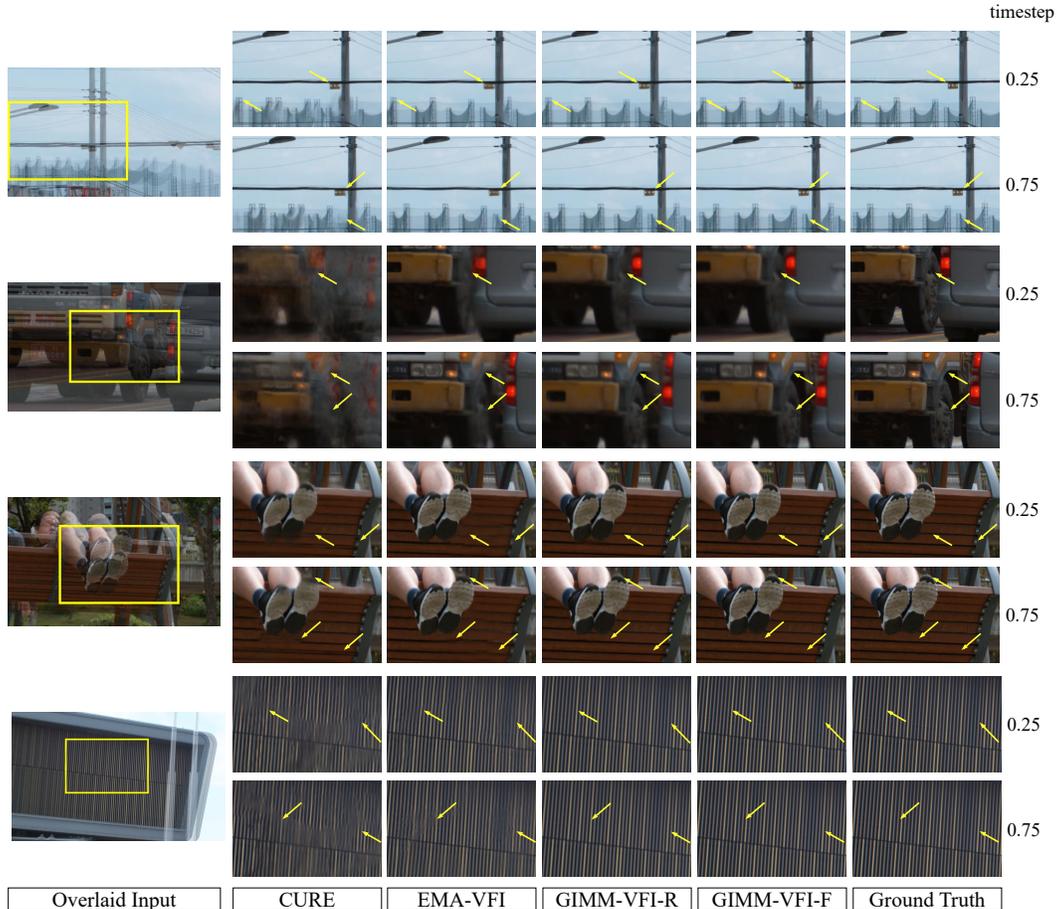
Figure 5: Qualitative comparisons of arbitrary-timestep interpolation on XTest-2K [46]. Positions pointed by the yellow arrow indicate the distinct performance of our method.

5 Limitation

There are several known limitations to our method. First of all, the GIMM-VFI is closely related to the pre-trained flow estimator, which estimates bidirectional flows and extracts image features. Therefore, GIMM-VFI inherits the limitations of the chosen pre-trained flow estimators, which may include high computational costs. Additionally, GIMM-VFI only accepts two consecutive frames as inputs, which is not favorable for frame interpolations where larger and nonlinear motion exists.

6 Conclusion

We propose a novel motion modeling method, GIMM, which performs continuous motion modeling for video frame interpolation. Our proposed GIMM is the first attempt to learn generalizable implicit neural representations for continuous motion modeling. The method can properly predict arbitrary-timestep optical flows within two adjacent video frames at any resolution, and be easily integrated with existing flow-based VFI approaches (*e.g.,* AMT [29]) without further modifications by supplying accurately modeled motion. Extensive experiments on the VFI benchmarks show that our GIMM can effectively perform generalizable motion modeling across videos.

Acknowledgement. This study is supported under the RIE2020 Industry Alignment Fund – Industry Collaboration Projects (IAF-ICP) Funding Initiative, as well as cash and in-kind contributions from the industry partner(s).

10

References

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

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

[3] Hao Chen, Bo He, Hanyu Wang, Yixuan Ren, Ser Nam Lim, and Abhinav Shrivastava. NeRV: Neural representations for videos. In *NeurIPS*, 2021.

[4] Yi-Hsin Chen, Si-Cun Chen, Yen-Yu Lin, and Wen-Hsiao Peng. MoTIF: Learning motion trajectories with local implicit neural functions for continuous space-time video super-resolution. In *ICCV*, 2023.

[5] Yinbo Chen, Sifei Liu, and Xiaolong Wang. Learning continuous image representation with local implicit image function. In *CVPR*, 2021.

[6] Yinbo Chen and Xiaolong Wang. Transformers as meta-learners for implicit neural representations. In *ECCV*, 2022.

[7] Zeyuan Chen, Yinbo Chen, Jingwen Liu, Xingqian Xu, Vidit Goel, Zhangyang Wang, Humphrey Shi, and Xiaolong Wang. VideoINR: Learning video implicit neural representation for continuous space-time super-resolution. In *CVPR*, 2022.

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

[9] Xianhang Cheng and Zhenzhong Chen. Multiple video frame interpolation via enhanced deformable separable convolution. *IEEE TPAMI*, 2021.

[10] Julian Chibane, Thiemo Alldieck, and Gerard Pons-Moll. Implicit functions in feature space for 3d shape reconstruction and completion. In *CVPR*, 2020.

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

[12] Tianyu Ding, Luming Liang, Zhihui Zhu, and Ilya Zharkov. CDFI: Compression-driven network design for frame interpolation. In *CVPR*, 2021.

[13] Pedro Figueirêdo, Avinash Paliwal, and Nima Khademi Kalantari. Frame interpolation for dynamic scenes with implicit flow encoding. In *WACV*, 2023.

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

[15] Kyle Genova, Forrester Cole, Avneesh Sud, Aaron Sarna, and Thomas Funkhouser. Local deep implicit functions for 3d shape. In *CVPR*, 2020.

[16] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *NeurIPS*, 2017.

[17] Xinyu Hou, Liming Jiang, Rui Shao, and Chen Change Loy. Video infilling with rich motion prior. In *BMVC*, 2023.

[18] Ping Hu, Simon Niklaus, Stan Sclaroff, and Kate Saenko. Many-to-many splatting for efficient video frame interpolation. In *CVPR*, 2022.

[19] Zhaoyang Huang, Xiaoyu Shi, Chao Zhang, Qiang Wang, Ka Chun Cheung, Hongwei Qin, Jifeng Dai, and Hongsheng Li. FlowFormer: A transformer architecture for optical flow. In *ECCV*, 2022.

[20] Zhewei Huang, Tianyuan Zhang, Wen Heng, Boxin Shi, and Shuchang Zhou. Real-time intermediate flow estimation for video frame interpolation. In *ECCV*, 2022.

11

[21] Chiyu Jiang, Avneesh Sud, Ameesh Makadia, Jingwei Huang, Matthias Nießner, and Thomas Funkhouser. Local implicit grid representations for 3d scenes. In *CVPR*, 2020.

[22] 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 *CVPR*, 2018.

[23] Xin Jin, Longhai Wu, Jie Chen, Youxin Chen, Jayoon Koo, and Cheul-Hee Hahm. A unified pyramid recurrent network for video frame interpolation. In *CVPR*, 2023.

[24] Tarun Kalluri, Deepak Pathak, Manmohan Chandraker, and Du Tran. FLAVR: Flow-agnostic video representations for fast frame interpolation. In *WACV*, 2023.

[25] Chiheon Kim, Doyup Lee, Saehoon Kim, Minsu Cho, and Wook-Shin Han. Generalizable implicit neural representations via instance pattern composers. In *CVPR*, 2023.

[26] Lingtong Kong, Boyuan Jiang, Donghao Luo, Wenqing Chu, Xiaoming Huang, Ying Tai, Chengjie Wang, and Jie Yang. IFRNet: Intermediate feature refine network for efficient frame interpolation. In *CVPR*, 2022.

[27] Doyup Lee, Chiheon Kim, Minsu Cho, and Wook-Shin Han. Locality-aware generalizable implicit neural representation. In *NeurIPS*, 2023.

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

[29] Zhen Li, Zuo-Liang Zhu, Ling-Hao Han, Qibin Hou, Chun-Le Guo, and Ming-Ming Cheng. AMT: All-pairs multi-field transforms for efficient frame interpolation. In *CVPR*, 2023.

[30] Zhengqi Li, Simon Niklaus, Noah Snavely, and Oliver Wang. Neural scene flow fields for space-time view synthesis of dynamic scenes. In *CVPR*, 2021.

[31] Chengxu Liu, Huan Yang, Jianlong Fu, and Xueming Qian. TTVFI: Learning trajectory-aware transformer for video frame interpolation. *IEEE TIP*, 2023.

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

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

[34] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. NeRF: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020.

[35] Simon Niklaus, Ping Hu, and Jiawen Chen. Splatting-based synthesis for video frame interpolation. In *WACV*, 2023.

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

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

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

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

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

[41] Tomer Peleg, Pablo Szekely, Doron Sabo, and Omry Sendik. Im-net for high resolution video frame interpolation. In *CVPR*, 2019.

12

[42] Fitsum Reda, Janne Kontkanen, Eric Tabellion, Deqing Sun, Caroline Pantofaru, and Brian Curless. FILM: Frame interpolation for large motion. In *ECCV*, 2022.

[43] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *CVPR*, 2022.

[44] Shunsuke Saito, Zeng Huang, Ryota Natsume, Shigeo Morishima, Angjoo Kanazawa, and Hao Li. PIFu: Pixel-aligned implicit function for high-resolution clothed human digitization. In *ICCV*, 2019.

[45] Wentao Shangguan, Yu Sun, Weijie Gan, and Ulugbek S Kamilov. Learning cross-video neural representations for high-quality frame interpolation. In *ECCV*, 2022.

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

[47] Uriel Singer, Adam Polyak, Thomas Hayes, Xi Yin, Jie An, Songyang Zhang, Qiyuan Hu, Harry Yang, Oron Ashual, Oran Gafni, Devi Parikh, Sonal Gupta, and Yaniv Taigman. Make-A-Video: Text-to-video generation without text-video data. *arXiv preprint arXiv:2209.14792*, 2022.

[48] Vincent Sitzmann, Julien Martel, Alexander Bergman, David Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. In *NeurIPS*, 2020.

[49] Li Siyao, Shiyu Zhao, Weijiang Yu, Wenxiu Sun, Dimitris Metaxas, Chen Change Loy, and Ziwei Liu. Deep animation video interpolation in the wild. In *CVPR*, 2021.

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

[51] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017.

[52] Chao-Yuan Wu, Nayan Singhal, and Philipp Krahenbuhl. Video compression through image interpolation. In *ECCV*, 2018.

[53] Xiangyu Xu, Li Siyao, Wenxiu Sun, Qian Yin, and Ming-Hsuan Yang. Quadratic video interpolation. In *NeurIPS*, 2019.

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

[55] Guozhen Zhang, Yuhan Zhu, Haonan Wang, Youxin Chen, Gangshan Wu, and Limin Wang. Extracting motion and appearance via inter-frame attention for efficient video frame interpolation. In *CVPR*, 2023.

[56] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018.

[57] Tinghui Zhou, Shubham Tulsiani, Weilun Sun, Jitendra Malik, and Alexei A Efros. View synthesis by appearance flow. In *ECCV*, 2016.

13

7   Appendix

In this supplementary material, we first provide details of our method in Section 7.1, including descriptions of flow metrics and reverse flow normalization. Besides, we present the network architectures of both the GIMM module and frame synthesis module in Section 7.2. Moreover, we provide details of our implementation and the corresponding training hyperparameters in Section 7.3 and Section 7.4 respectively. We show qualitative results of motion modeled by GIMM and its model variants in Section 7.5. Furthermore, we present integrations with other flow-based VFI methods in Section 7.6. Finally, we discuss the potential broader impacts in Section 7.7.

7.1   Method Details

Flow metrics. Following Splatting-based Synthesis [35], the implementation of the forward warping operation within our GIMM module leverages flow metrics to enhance its accuracy. More precisely, we employ flow consistency U^i_{flow}, and flow variance U^i_{var}. For illustrative purposes, taking $F_{0 \to 1}$ as a representative example, we compute its associated metrics as follows:

$$U^0_{flow} = ||F_{0 \to 1} + \overleftarrow{\omega}(F_{1 \to 0}, F_{0 \to 1})||_1, \tag{16}$$

$$U^0_{var} = ||\sqrt{G((F_{0 \to 1})^2) - G(F_{0 \to 1})^2}||, \tag{17}$$

where $\overleftarrow{\omega}$ is the backward warping operator, and $G(\cdot)$ denotes a 3×3 Gaussian filter.

Reverse flow normalization. In GIMM-VFI, we employ reverse flow normalization to convert the predicted normalized flow, V_t, into bilateral flows for interpolation purposes. This transformation is expressed through the following equations:

$$F_{t \to 0} = -t \cdot \phi^{-1}(V_t), \tag{18}$$

$$F_{t \to 1} = (1 - t) \cdot \phi^{-1}(V_t). \tag{19}$$

In this context, ϕ^{-1} represents the scaling operation that extends the value range from $[0, 1]$ to $\mathbb{R}$ using an instance-specific scaling factor. This reverse flow normalization operation is identical to the one used in IFE [13].

7.2   Network Architecture
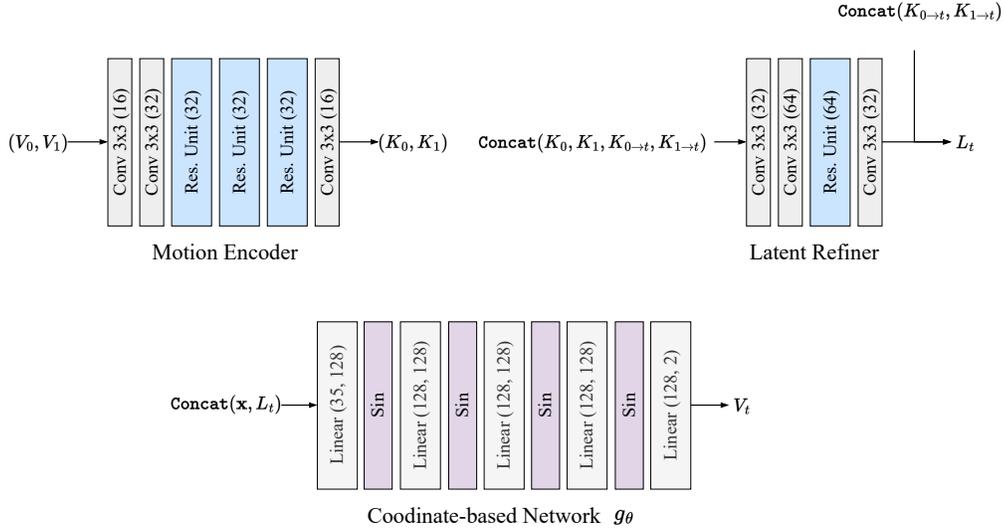
Figure 6: Details of the network architectures within the GIMM Module. The numbers in parentheses denote the output channel count for convolutional layers or the dimensionality shift from input to output for linear layers.

14
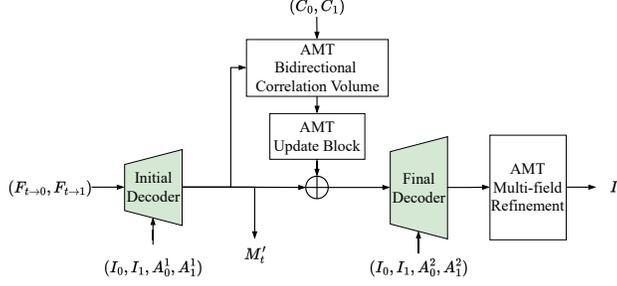
Figure 7: Overview of the frame synthesis module. The frame synthesis module conducts multi-scale flow refinements and predicts warping masks for interpolation, including the intermediate warping mask M'_t predicted by the initial decoder. It integrates the images and their pyramid features $A_0^l, A_1^l | l \in 1, 2$ during decoding, updating predictions with correlation information from the Bidirectional Correlation Volume, which is constructed with correlation features (C_0, C_1). Ultimately, we use the outputs from the Final Decoder to perform the final interpolation of I_t through the Multi-field Refinement block. This block is adapted from AMT-G [29]. Modified components are highlighted in green for clarity.
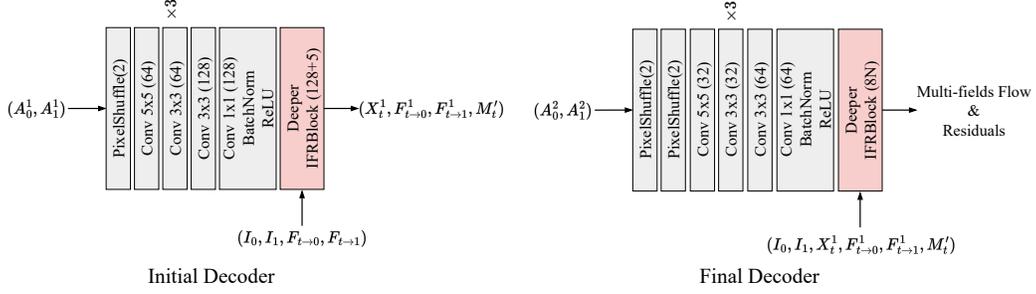
Figure 8: The network architectures of the modified decoders within the frame synthesis module. Unless specifically noted, each convolutional layer is activated by a PReLU function. The Deeper IFRBlock (highlighted in red) is based on the decoder architecture (IFRBlock) introduced in IFR-Net [26]. In particular, we enhance the original IFRBlock by adding two additional residual blocks to create the Deeper IFRBlock.

GIMM module. The architectural details of the networks used in GIMM are shown in Fig. 6. The Motion Encoder and Latent Refiner use a similar convolutional network architecture with residual units. Each Residual Unit consists of two convolutional layers with a skip connection. The major differences between them are the number of channels and residual units. On the other hand, the coordinate-based network g_θ is designed as a five-layer SIREN [48], incorporating the motion latent L_t as an auxiliary input.

Frame synthesis module. Adapted from AMT-G [29], the framework of the Frame synthesis module is illustrated in Fig. 7. This module integrates two unique decoding stages: an adapted initial decoder and a revised final decoder, while the rest of the module's architecture remains unchanged from AMT-G. The architectural specifics of these modified decoders are shown in Fig. 8.

7.3 Implementation Details

In addition to experiments in the main text, we present more detailed implementations of GIMM and GIMM-VFI in this section.

GIMM training. We begin by extracting optical flows from the training split of the Vimeo90K triplets dataset [54] using Flowformer [19]. With these extracted flows, we train GIMM, randomly cropping the flows to a resolution of 256×256. For each batch during training, we randomly select a timestep t from the set $\{0, 0.5, 1\}$ to supervise. We set the batch size to 64, and train the model for 240 epochs with a learning rate of 1×10^{-4}, using 2 NVIDIA V100 GPUs.

15

GIMM-VFI training. We integrate the pre-trained GIMM module and flow estimator into the GIMM-VFI model for end-to-end training on the arbitrary-timestep interpolation task. We implement two variants of GIMM-VFI, using two different flow estimators: the RAFT [50] and FlowFormer [19], designated as GIMM-VFI-R and GIMM-VFI-F, respectively. However, both versions of GIMM-VFI share the same training process. Similar to previous works [55, 20], we train our model on the complete Vimeo90K septuplet split [54] for 60 epochs with a batch size of 32 and a learning rate of 8×10^{-5}. We randomly select triple subsets for training from each septuplet, following the same sampling strategy as previous research [55, 20]. We resize and randomly crop each frame into a resolution of 224×224 and perform a series of augmentations including rotation, flipping, temporal order reversing and channel order reversing. We train our model on 8 NVIDIA V100 GPUs.

7.4 Training hyperparameters

In addition to the implementation details in Sec. 7.3, we summarize the training settings for both GIMM and GIMM-VFI in Tab. 5.

Table 5: **Training settings** for GIMM and GIMM-VFI.

Configuration	GIMM	GIMM-VFI
Optimizer	AdamW	
Peak learning rate	1e-4	8e-5
Minimum learning rate	1e-4	8e-6
Epochs	240	60
Batch size per GPU	16	4
Weight decay	0	4e-5
Optimizer momentum	$\beta_1, \beta_2 = 0.9, 0.999$	
Learning rate schedule	55	Cosine annealing
Warmip epochs	0	1
Training Resolution	256×256	224×224
Num. V100 GPUs	2	8

7.5 Qualitative Results of GIMM Motion Modeling

We present additional qualitative results of GIMM motion modeling in Figure 9. The visualization showcases the modeled motion of GIMM and variants in the ablation study. Our observation confirms the effectiveness of GIMM's design and is consistent with the results from the ablation study in the main text. For example, the head motion in Figure 9 significantly deteriorates with noises across all timesteps when forward warping is replaced or latent refinement is skipped.

7.6 Integration with other flow-based VFI methods

Our method GIMM focuses on continuous motion modeling, which further enables frame interpolation at arbitrary timesteps. We additionally integrate GIMM with other existing flow-based VFI approaches (*e.g.*, TTVFI [31], IFRNet [26]), by supplying accurately modeled motion. For evaluations, we calculate PSNRs on arbitrary-timestep interpolation benchmark SNU-FILM-arb in Table 6. We observed that integrating the GIMM module resulted in significant improvements. This further demonstrates the effectiveness of GIMM for continuous motion modeling when integrated with existing flow-based VFI works.

7.7 Broader Impacts

Our proposed method, GIMM-VFI, demonstrates strong performance in continuous frame interpolation, with applications spanning real-world scenarios such as creating cinematic special effects and producing slow-motion videos. This technique not only enhances these use cases but also opens doors for further research in video technology. At its core, GIMM-VFI employs the GIMM module for continuous motion modeling in video frame interpolation, a feature that could also be adapted for related tasks like video compression and predictive video analysis. Despite its strengths, our approach shares limitations with other frame interpolation methods and, in some cases, may introduce artifacts that could pollute visual data on the Internet.

16
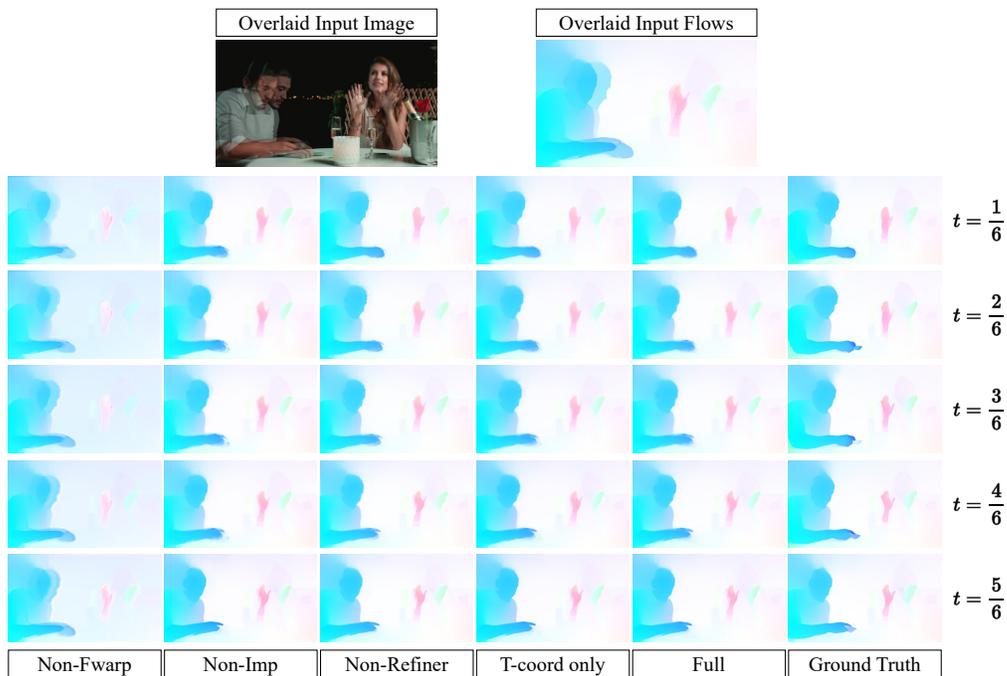
Figure 9: Visual comparisons for 6X motion modeling on the Vimeo-septuplet-flow. We compare GIMM with its model variants mentioned in the ablation study of the main text.

Table 6: **Integrating GIMM with TTVFI and IFRNet.**

Motion method	SNU-FILM-arb-Medium	SNU-FILM-arb-Hard	SNU-FILM-arb-Extreme
	PSNR↑	PSNR↑	PSNR↑
TTVFI	34.48	30.39	26.24
TTVFI+GIMM	**35.55 (+1.07dB)**	**31.60 (+1.21dB)**	**27.40 (+1.16dB)**
IFRNet	34.88	31.15	26.32
IFRNet+GIMM	**36.46 (+1.58dB)**	**32.20 (+1.05dB)**	**27.73 (+1.41dB)**

17

NeurIPS Paper Checklist

The checklist is designed to encourage best practices for responsible machine learning research, addressing issues of reproducibility, transparency, research ethics, and societal impact. Do not remove the checklist: **The papers not including the checklist will be desk rejected.** The checklist should follow the references and follow the (optional) supplemental material. The checklist does NOT count towards the page limit.

Please read the checklist guidelines carefully for information on how to answer these questions. For each question in the checklist:

- You should answer [Yes] , [No] , or [NA] .
- [NA] means either that the question is Not Applicable for that particular paper or the relevant information is Not Available.
- Please provide a short (1–2 sentence) justification right after your answer (even for NA).

The checklist answers are an integral part of your paper submission. They are visible to the reviewers, area chairs, senior area chairs, and ethics reviewers. You will be asked to also include it (after eventual revisions) with the final version of your paper, and its final version will be published with the paper.

The reviewers of your paper will be asked to use the checklist as one of the factors in their evaluation. While "[Yes] " is generally preferable to "[No] ", it is perfectly acceptable to answer "[No] " provided a proper justification is given (e.g., "error bars are not reported because it would be too computationally expensive" or "we were unable to find the license for the dataset we used"). In general, answering "[No] " or "[NA] " is not grounds for rejection. While the questions are phrased in a binary way, we acknowledge that the true answer is often more nuanced, so please just use your best judgment and write a justification to elaborate. All supporting evidence can appear either in the main paper or the supplemental material, provided in appendix. If you answer [Yes] to a question, in the justification please point to the section(s) where related material for the question can be found.

IMPORTANT, please:

- **Delete this instruction block, but keep the section heading "NeurIPS paper checklist",**
- **Keep the checklist subsection headings, questions/answers and guidelines below.**
- **Do not modify the questions and only use the provided macros for your answers**.

1. **Claims**

    Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

    Answer: [Yes]

    Justification: We have made clear claims in the abstract and introduction that accurately reflect our paper's contributions to the studied problem. Please refer to the abstract and introduction for more detailed information.

    Guidelines:

    - The answer NA means that the abstract and introduction do not include the claims made in the paper.
    - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
    - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
    - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

    Question: Does the paper discuss the limitations of the work performed by the authors?

    Answer: [Yes]

18

Justification: We have discussions on the limitations of our proposed GIMM-VFI method in Section 5 of the main text.

Guidelines:

- The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [NA]

Justification: Our work does not include theoretical results.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: We provide the implementation details and training hyperparameters in the supplementary materials.

Guidelines:

19

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [No]

Justification: We are unable to provide our code upon submission, but releasing the code to the public in the future is our plan.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.

20

- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyper-parameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [Yes]

Justification: We provide the general training information and evaluation settings in the main text. The full implementation details and training hyperparameters are presented in the supplementary materials.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [No]

Justification: This paper does not report error bars.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [Yes]

Justification: We provide the type and the number of GPUs used in our experiments.

Guidelines:

21

- The answer NA means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

    Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

    Answer: [Yes]

    Justification: We carefully follow the NeurIPS Code of Ethics.

    Guidelines:
    - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
    - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
    - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    We discuss the potential impacts in the supplementary materials.

    Guidelines:
    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

22

Justification: Our paper focuses on the problem of motion modeling in the video frame interpolation task, and poses no such risks.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: In our paper, we have cited the related papers.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [No]

Justification: In our paper, we propose a new method GIMM-VFI. Upon submission, we do not provide the source code. But we will make the code public in the future after acceptance.

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

23

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [NA]

Justification: Our paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [NA]

Justification: Our paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

24