A Temporally-Aware Interpolation Network for Video Frame Inpainting

Ryan Szeto, Ximeng Sun, Kunyi Lu, and Jason J. Corso, *Senior Member, IEEE*

Abstract—In this work, we explore video frame inpainting, a task that lies at the intersection of general video inpainting, frame interpolation, and video prediction. Although our problem can be addressed by applying methods from other video interpolation or extrapolation tasks, doing so fails to leverage the additional context information that our problem provides. To this end, we devise a method specifically designed for video frame inpainting that is composed of two modules: a bidirectional video prediction module and a temporally-aware frame interpolation module. The prediction module makes two intermediate predictions of the missing frames, each conditioned on the preceding and following frames respectively, using a shared convolutional LSTM-based encoder-decoder. The interpolation module blends the intermediate predictions by using time information and hidden activations from the video prediction module to resolve disagreements between the predictions. Our experiments demonstrate that our approach produces smoother and more accurate results than state-of-the-art methods for general video inpainting, frame interpolation, and video prediction.

Index Terms—Video Inpainting, Video Prediction, Frame Interpolation, Temporal Upsampling

1 Introduction

There exist multiple video interpolation/extrapolation tasks where the goal is to synthesize pixels across space and time conditioned on multiple input frames. In particular, three such tasks have received a substantial amount of attention in recent years. In the first task, *general video inpainting*, we are given a video that is missing arbitrary voxels (spatio-temporal pixels), and the goal is to fill each voxel with the correct value. In the second task, *frame interpolation*, the goal is to predict the appearance of one or more frames that lie in between two (typically subsequent) input frames. In the final task, *video prediction*, the goal is to take a sequence of input frames and extrapolate the appearance of multiple future frames.

Unfortunately, these three tasks are limited in terms of their assumptions or well-posed they are. For example, general video inpainting methods are designed to fill in relatively small spatio-temporal regions, and may therefore perform poorly when whole, contiguous frames are missing. Frame interpolation methods fill in whole frames, but they cannot leverage enough contextual information to rule out several plausible predictions (for instance, to predict the appearance of a swinging pendulum, we would need more than two frames to determine its speed). Similarly, video prediction methods cannot leverage information to determine which of several possible futures to predict.

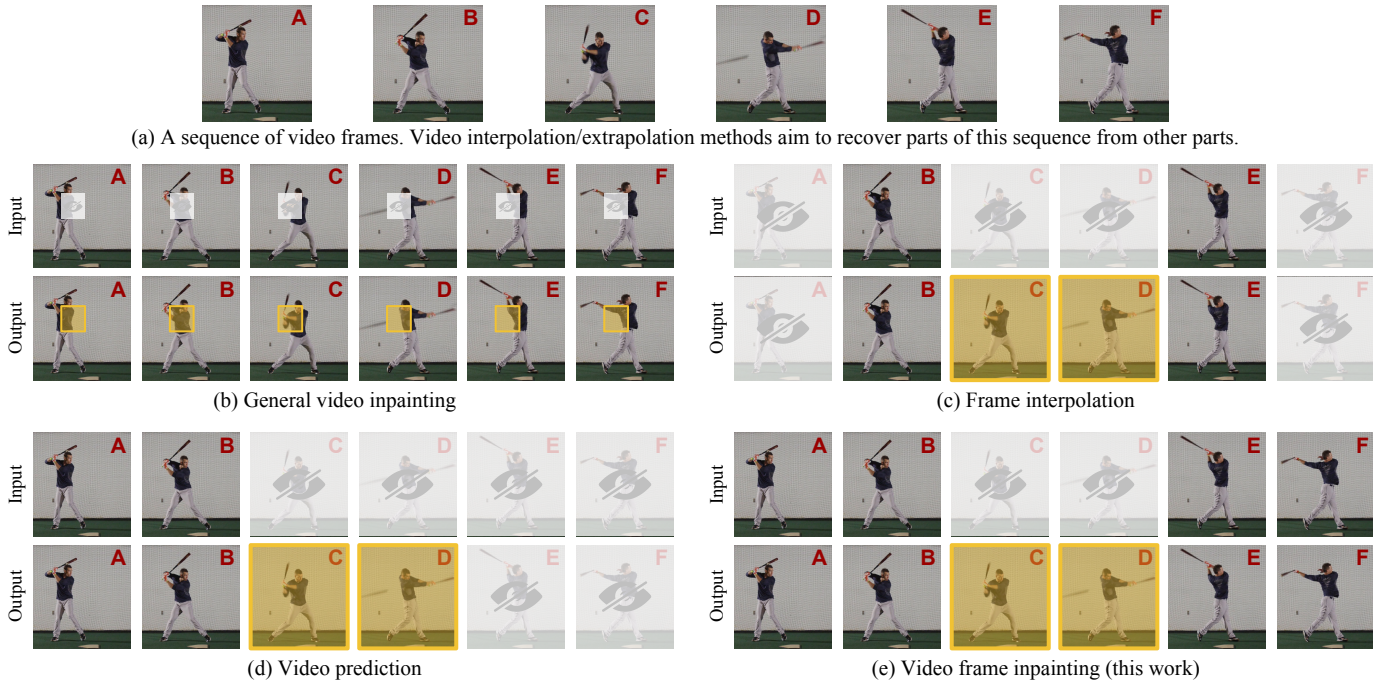
In light of the limitations of these video interpolation/extrapolation tasks, we focus on an underexplored task that lies at their intersection, *video frame inpainting* (shown in Fig. 1e). In this task, the goal is to reconstruct a missing sequence of middle video frames given contiguous sequences of frames that come immediately before and after it (the *preceding* and the *following* frames). For example, given a clip of someone winding up a baseball pitch and a clip of that person after he/she has released the ball, we would predict the clip of that person throwing the ball. Video frame inpainting methods can be used in multiple applications, e.g. to upsample videos temporally or to smoothly splice video clips taken at different times.

Although our task can be seen as a generalization or modification of other standard video interpolation/extrapolation tasks, ours provides a new combination of challenges that enable methodological insight. For example, it resembles general video inpainting in the case where the missing voxels are arranged in a certain pattern, but our formulation emphasizes the completion of multiple contiguous frames without ground-truth spatial information at the inpainted time steps, whereas general video inpainting typically emphasizes the completion of regions over a long temporal extent but a limited spatial extent. It also resembles frame interpolation and video prediction with additional input frames, but to the best of our knowledge, prior works in these areas have not leveraged extended temporal context both before and after the desired sequence. Furthermore, compared to frame interpolation and video prediction, the appearance of middle frames is greatly constrained by the extended context on either side, making our formulation more well-defined and reconstruction error more interpretable.

In this work, we present the first deep neural network (DNN) specifically designed for video frame inpainting. Inspired by the recent successes of DNNs for video prediction and frame interpolation, we address the problem in two steps as shown in Fig. 2. First, we use a video prediction subnetwork to generate two intermediate predictions of the middle frames: the "forward prediction" conditioned on the preceding frames, and the "backward prediction" conditioned on the following frames. Then, we blend each pair

- *Ryan Szeto, Kunyi Lu, and Jason J. Corso are with the Department of Electrical Engineering and Computer Science, University of Michigan, Ann Arbor, MI, 48109. Email: {szetor, lukunyi, jjcorso}@umich.edu.*
- *Ximeng Sun is with the Department of Computer Science, Boston University, Boston, MA, 02215. Email: sunxm@bu.edu*

Manuscript received ???; revised ???.



(a) A sequence of video frames. Video interpolation/extrapolation methods aim to recover parts of this sequence from other parts.

(b) General video inpainting

(c) Frame interpolation

(d) Video prediction

(e) Video frame inpainting (this work)

Fig. 1: A visual comparison of various video interpolation/extrapolation tasks. In this paper, we explore (e) video frame inpainting. Unlike general video inpainting methods, we recover whole, contiguous frames; and unlike frame interpolation and video prediction methods, we predict the desired sequence using *multiple* frames that appear both *before* and *after* it.

of frames corresponding to the same time step to obtain the final prediction. The blending process uses a convolutional neural network (CNN) that interpolates the final frame from the two input frames as well as the corresponding time step in a data-driven manner. In short, we perform *bidirectional prediction* followed by *temporally-aware interpolation (TAI)* to predict the middle frames; therefore, we name our full method and model **bi-TAI**. As we show in our experiments, bi-TAI yields the most accurate predictions among several state-of-the-art baselines and across multiple human action video datasets.

Blending the intermediate frames generated by the bidirectional prediction process is a non-trivial task; therefore, we develop a TAI strategy that exploits three characteristics of bidirectional prediction. First, a pair of intermediate frames for the same time step might be inconsistent, e.g. an actor might appear in two different locations. To address this, we introduce a CNN for blending/interpolation that modulates the pair of frames with adaptive convolutional kernels and then adds them together; this process can cleanly merge the pair of frames by reconciling the differences between them. Second, for any given time step, the forward and backward predictions are not equally reliable: the former is more accurate for earlier time steps, and the latter is more accurate for later time steps. Hence, we feed time step information directly into the blending network, making it *temporally-aware* by allowing it to blend differently depending on which time step it is operating at. Finally, the intermediate predictions come from a DNN whose hidden features may be useful for blending. To leverage these features, we use them to condition the adaptive convolutional kernels that are used to modulate the intermediate predictions. We implement this strategy with a novel blending CNN called the **Temporally-Aware Interpolation Network** (or TAI network for short), which we describe in Sec. 3.4.

In summary, we make the following contributions. First, we propose bi-TAI, a DNN for video frame inpainting that generates two intermediate predictions and blends them together with our novel TAI network. Second, we compare our approach to several state-of-the-art methods from the video inpainting, video prediction, and frame interpolation literature, and demonstrate that our method outperforms them quantitatively and qualitatively on several human action video datasets. Finally, we perform ablation studies to demonstrate that using a *temporally-aware* interpolation network (as opposed to simple blending strategies or an interpolation network that does not learn to use temporal information dynamically) is key to blending bidirectional predictions well. We provide an implementation of our model on GitHub[1].

2 Related Work

In the *general video inpainting task* (of which our video frame inpainting task is a challenging instance), the goal is to replace missing arbitrary voxels with the correct value. Existing methods generally fall into one of three categories: *patch-based* methods that search for complete spatio-temporal patches to copy into the missing area [1], [2], [3], [4]; *object-based* methods that separate spatial content into layers (e.g. foreground and background), repair them individually, and stitch them back together [5], [6]; and *probabilistic model-based* methods that assign values that maximize the likelihood under some probabilistic model [7], [8], [9]. Many of these

1. https://github.com/MichiganCOG/video-frame-inpainting

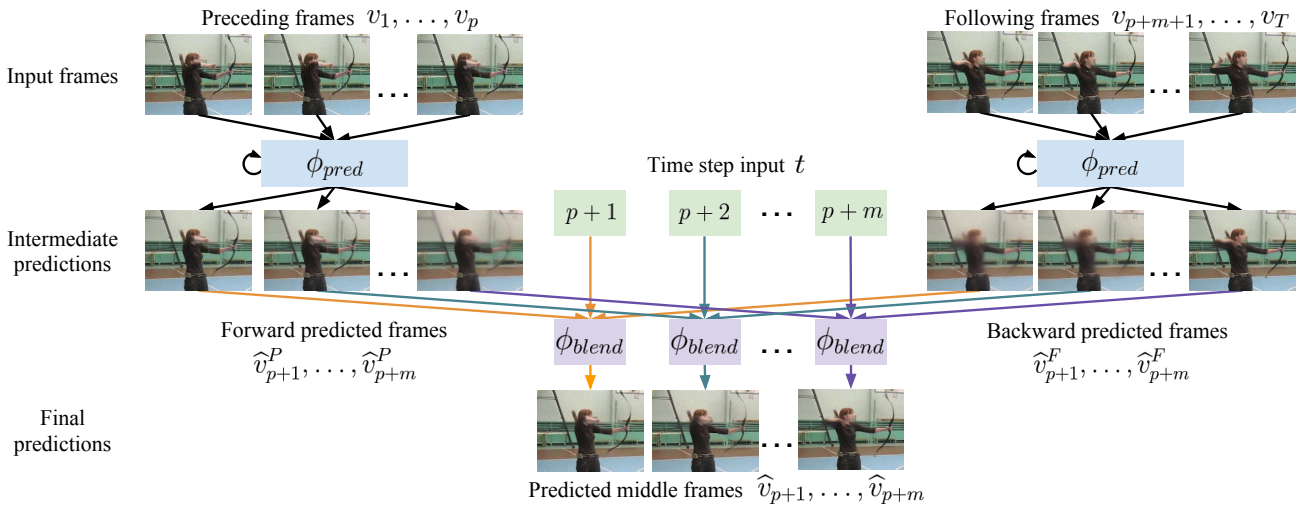


Fig. 2: An overview of our bi-TAI method for video frame inpainting. We predict middle frames by blending forward and backward intermediate video predictions (generated by ϕ_{pred}) with a Temporally-Aware Interpolation network (ϕ_{blend}).

approaches make strong assumptions about the video content, such as constrained camera pose/motion [3], [5], [6] or static backgrounds [5], [6], [8]. In addition, they are designed for the case in which the missing voxels are localized to small spatio-temporal regions, and may therefore perform poorly when whole, contiguous frames are missing. Furthermore, to the best of our knowledge, no existing solution has leveraged deep neural networks, which can potentially outperform prior work thanks to the vast amounts of video data available online.

In the *frame interpolation task*, the goal is to predict one or more frames in between two (typically subsequent) input frames. While most classical approaches linearly interpolate a dense optical flow field to an arbitrary number of intermediate time steps [10], [11], [12], recent approaches often train DNNs to predict one intermediate frame [13], [14], [15]. The problem of predicting multiple intermediate frames with DNNs is still in its early stages [16]. However, these approaches are generally evaluated on input frames that occur within a miniscule window of time (i.e. no more than 0.05 seconds apart). Furthermore, the task is ambiguous because a pair of individual frames without temporal context cannot sufficiently constrain the appearance of the intermediate frames. As a result, it is hard to evaluate predictions that are plausible, but deviate from the ground truth.

In the *video prediction task*, the goal is to generate the future frames that follow a given sequence of video frames. Note that this task is related to visual feature forecasting [17], which aims to predict high-level features for a specific task rather than raw pixels. The earliest approaches to video prediction draw heavily from language modeling literature by extending simple recurrent sequence-to-sequence models to predict patches of video [18], [19]; more recent methods utilize structured models that decompose the input data and/or the learned representations in order to facilitate training [20], [21], [22]. Zeng et al. [23] frame video prediction as an imitation learning problem instead of a regression problem, and yield a policy for predicting the future. As with frame interpolation, video prediction is inherently underconstrained since the past can diverge into multiple plausible futures, although there exists work that addresses this ambiguity [24].

3 Approach

3.1 Problem Statement

We define the video frame inpainting problem as follows. Let $V = \{v_1, v_2, \ldots, v_T\}$ be a sequence of frames from a real video, p, m, and f be the number of "preceding", "middle", and "following" frames such that $p+m+f=T$, and $P_V = \{v_1, \ldots, v_p\}, M_V = \{v_{p+1}, \ldots, v_{p+m}\}, F_V = \{v_{p+m+1}, \ldots, v_T\}$ be the sequences of preceding, middle, and following frames from V respectively. We seek an approximation to the oracle video frame inpainting function ϕ that satisfies $M_V = \phi(P_V, F_V)$ for all V.

3.2 Model Overview

We propose a DNN to approximate the oracle video frame inpainting function ϕ (see Fig. 2). Our model decomposes the problem into two sub-problems and tackles them sequentially with two modules: the Bidirectional Video Prediction Network (Sec. 3.3) and the Temporally-Aware Interpolation Network (Sec. 3.4).

- The **Bidirectional Video Prediction Network** generates two intermediate predictions of the middle sequence M_V, where each prediction is conditioned solely on the preceding sequence P_V and the following sequence F_V respectively.
- The **Temporally-Aware Interpolation (TAI) Network** blends corresponding frames from the predictions made by the Bidirectional Video Prediction Network, thereby producing the final prediction $\widehat{M}_V$. It accomplishes this by leveraging intermediate activations from the Bidirectional Video Prediction Network, as well as scaled time steps that explicitly indicate the relative temporal location of each frame in the final prediction.

Even though our model factorizes the video frame inpainting process into two steps, it is optimized end-to-end.

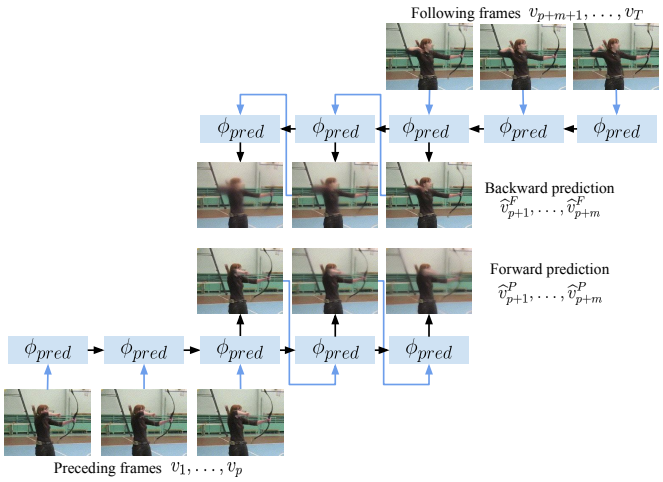


Fig. 3: The Bidirectional Video Prediction Network.

3.3 Bidirectional Video Prediction Network

We use the Bidirectional Video Prediction Network ϕ_{pred}, shown in Fig. 3, to produce two intermediate predictions—a "forward prediction" $\widehat{M}^P_V = \{\widehat{v}^P_{p+1}, \ldots, \widehat{v}^P_{p+m}\}$ and a "backward prediction" $\widehat{M}^F_V = \{\widehat{v}^F_{p+1}, \ldots, \widehat{v}^F_{p+m}\}$—by conditioning on the preceding sequence P_V and the following sequence F_V respectively:

$$\widehat{M}^P_V = \phi_{pred}\left(P_V\right), \tag{1}$$

$$\widehat{M}^F_V = \left[\phi_{pred}\big((F_V)^R\big)\right]^R, \tag{2}$$

where $(\cdot)^R$ is an operation that reverses the input sequence. We use the same parameters to generate the forward and backward predictions for two reasons: (i) it substantially reduces the size of the Bidirectional Video Prediction Network, and (ii) forward and backward motion behave similarly in terms of low-level pixel dynamics, so it is beneficial to share the parameters that predict such motion.

The Bidirectional Video Prediction Network recurrently generates each frame by conditioning on all previous frames. For example, for the forward prediction:

$$\widehat{v}^P_{k+1} = \phi_{pred}\big(\{\tilde{v}^P_1, \tilde{v}^P_2, \ldots, \tilde{v}^P_k\}\big), \tag{3}$$

where for a given t, $\tilde{v}^P_t$ is either v_t (an input frame) if $t \in \{1, \ldots, p\}$ or $\widehat{v}^P_t$ (an intermediate predicted frame) if $t \in \{p+1, \ldots, p+m\}$. During this phase, we also store a subset of the intermediate activations from the Bidirectional Video Prediction Network, denoted as $\pi_t = \{\pi^1_t, \pi^2_t, \ldots\}$, that serve as inputs to the TAI network. We apply an analogous procedure to obtain each frame in the backward prediction $\widehat{v}^F_t$ and its corresponding intermediate activations $\rho_t = \{\rho^1_t, \rho^2_t, \ldots\}$.

3.4 Temporally-Aware Interpolation Network

Following the Bidirectional Video Prediction Network, the TAI network ϕ_{blend} takes corresponding pairs of frames from $\widehat{M}^P_V$ and $\widehat{M}^F_V$ with the same time step, i.e. $\left(\widehat{v}^P_t, \widehat{v}^F_t\right)$ for each time step $t \in \{p+1, \ldots, p+m\}$, and blends them into the frames that make up the final prediction $\widehat{M}_V$:

$$\widehat{v}_t = \phi_{blend}\left(\widehat{v}^P_t, \widehat{v}^F_t\right), \tag{4}$$

$$\widehat{M}_V = \{\widehat{v}_t \mid t = p+1, \ldots, p+m\}. \tag{5}$$

Blending $\widehat{v}^P_t$ and $\widehat{v}^F_t$ is difficult because (i) they often contain mismatched content (e.g. between the pair of frames, objects might be in different locations), and (ii) they are not equally reliable (e.g. $\widehat{v}^P_t$ is more reliable for earlier time steps). As we show in Sec. 4.3, equally averaging $\widehat{v}^P_t$ and $\widehat{v}^F_t$ predictably results in ghosting artifacts (e.g. multiple faded limbs in human action videos), but remarkably, replacing a simple average with a state-of-the-art interpolation network (e.g. Niklaus et al. [14]) also exhibits this problem.

In order to blend corresponding frames more accurately, our TAI network utilizes two additional sources of information. Aside from the pair of frames to blend, it receives the scaled time step to predict, defined as $w_t = (t-p)/(m+1)$, and the intermediate activations from the Bidirectional Video Prediction Network π_t and ρ_t. We feed w_t to the TAI network so it can learn how to incorporate the unequal reliability of $\widehat{v}^P_t$ and $\widehat{v}^F_t$ into its final prediction; we feed π_t and ρ_t to leverage the high-level semantics that the Bidirectional Video Prediction Network has learned, as well as to backpropagate errors through the Bidirectional Video Prediction Network more easily. We contrast standard interpolation with TAI algebraically:

$$\widehat{v}_t = \phi_{interp}\left(\widehat{v}^P_t, \widehat{v}^F_t\right), \tag{6}$$

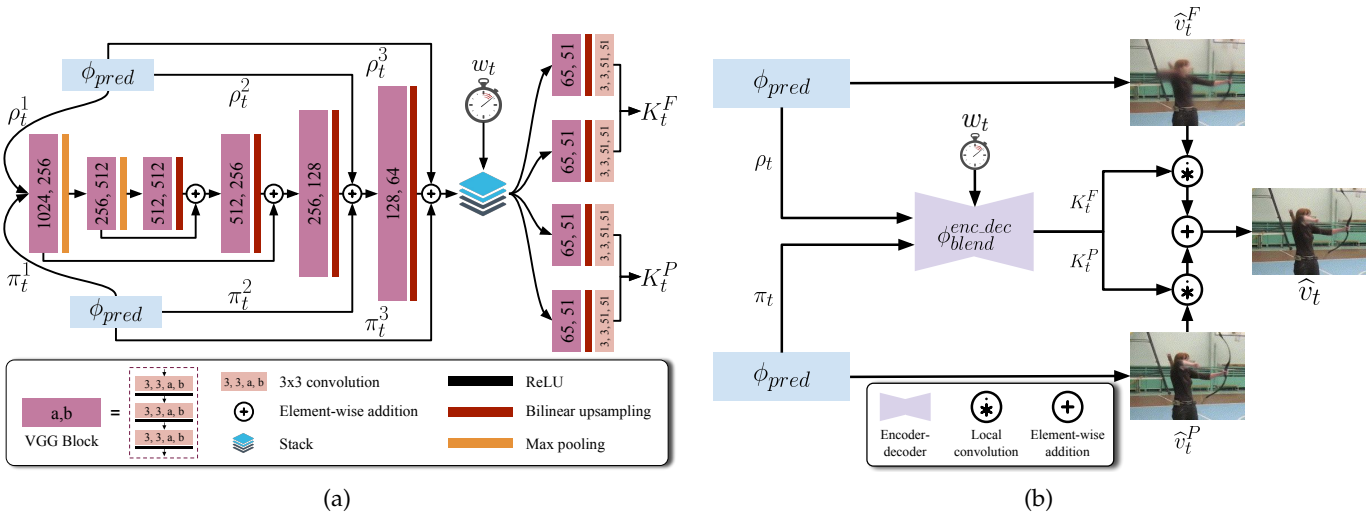
$$\widehat{v}_t = \phi_{TAI}\left(\widehat{v}^P_t, \pi_t, \widehat{v}^F_t, \rho_t, w_t\right). \tag{7}$$

3.5 Network Architecture Details

Our high-level approach to video frame inpainting places few constraints on the network architectures that can be used to implement each module (Sec. 3.2). To demonstrate the full potential of our approach, we base the network architectures for each module on the top-performing architectures for video prediction and frame interpolation (to the best of our knowledge as of this writing). We instantiate the Bidirectional Video Prediction Network ϕ_{pred} with MCnet [22] (we review the architecture of MCnet and its use in bi-TAI in Sec. 3.5.1). As for the TAI network, we modify the Separable Adaptive Kernel Network [14] to take as input the scaled time step w_t and the intermediate activations π_t and ρ_t (we elaborate on this extension in Sec. 3.5.2). An additional benefit of these architectures is that they are both fully-convolutional, which allows us to modify the video resolution at test time. We believe that the individual subnetworks can be improved to leverage the structure of our task more effectively, but we leave this for future work.

3.5.1 *Bidirectional Video Prediction Network Details*

MCnet. Instead of learning the spatiotemporal representation via a single encoding network, MCnet [22] disentangles the spatial and temporal encoding via two sets of three VGG [25] encoder blocks each (the weights are not shared). The spatial encoder operates on the previous RGB video frame, and the temporal encoder operates on the difference between the last two video frames $v_{t-1} - v_{t-2}$. The temporal encoder also employs a Convolutional LSTM [26] to encode



(a) (b)

Fig. 4: (a) The architecture of TAI's encoder-decoder network. (b) The TAI network applied to the intermediate predictions from the Bidirectional Video Prediction Network.

the temporal history of the sequence. After computing the spatial and temporal encodings, MCnet concatenates them along the feature dimension and decodes them into the next frame. The decoder also takes in the intermediate activations of corresponding VGG blocks in the content and temporal encoders, which are concatenated feature-wise and then fused via multiple convolution layers (residual layers).

Computing intermediate activations for TAI. To obtain the intermediate features π_t and ρ_t to be fed to the TAI network, we concatenate the intermediate activations from corresponding VGG blocks in the spatial and temporal encoders. For example, π_t^1 is the concatenation of the activations from the first VGG blocks in the forward prediction, ρ_t^2 is the concatenation of the activations from the second VGG blocks in the backward prediction, and so on. We thus have three intermediate features for each of the forward and backward predictions for each time step, i.e. $\pi_t = \{\pi_t^1, \pi_t^2, \pi_t^3\}$ and $\rho_t = \{\rho_t^1, \rho_t^2, \rho_t^3\}$.

3.5.2 TAI Network Details

Similarly to the Separable Adaptive Kernel Network [14], our TAI network blends a pair of intermediate frames $(\hat{v}_t^P, \hat{v}_t^F)$ by first applying a unique, adaptive 2D kernel to each patch in the two input frames, and then summing the resulting images pixel-wise. The primary way in which our TAI network differs from the Separable Adaptive Kernel Network is in how the adaptive kernels are computed. Both use an encoder-decoder network structure [27] that outputs the adaptive kernels; however, the Separable Adaptive Kernel Network uses the two frames to interpolate as inputs to the encoder-decoder network, whereas we use intermediate activations from the Bidirectional Video Prediction Network, π_t and ρ_t, as well as the scaled time step $w_t = (t-p)/(m+1)$. Note that we still apply the adaptive kernels to the intermediate frames $(\hat{v}_t^P, \hat{v}_t^F)$, not to the intermediate predictions π_t and ρ_t.

To be more precise, we take the scaled time step w_t and three sets of intermediate activations from the forward and backward predictions ($\pi_t = \{\pi_t^1, \pi_t^2, \pi_t^3\}$, $\rho_t = \{\rho_t^1, \rho_t^2, \rho_t^3\}$), and feed them to an encoder-decoder network to compute the parameters of the adaptive kernels K_t^P and K_t^F:

$$K_t^P, K_t^F = \phi_{blend}^{enc\_dec}(\pi_t, \rho_t, w_t) \ , \tag{8}$$

where K_t^P and K_t^F are 3D tensors whose height and width match the frame resolution and whose depth equals the number of parameters in each adaptive kernel. We inject the scaled time step by replicating it spatially and concatenating it to one of the decoder's hidden activations as an additional channel. As with the Separable Adaptive Kernel Network, we predict the parameters for a set of separable 2D kernels instead of standard 2D kernels to scale the size of the adaptive kernels more efficiently. The encoder-decoder network architecture is summarized in Fig. 4a.

Afterwards, we apply the adaptive kernels to each input frame and sum the resulting images pixel-wise:

$$\hat{v}_t(x,y) = K_t^P(x,y) * \mathcal{P}_t^P(x,y) + K_t^F(x,y) * \mathcal{P}_t^F(x,y) \ , \tag{9}$$

where $\hat{v}_t(x,y)$ is the pixel value of the final prediction $\hat{v}_t$ at (x,y), $K_t^{(\cdot)}(x,y)$ is the 2D kernel parameterized by the depth column of $K_t^{(\cdot)}$ at (x,y), $*$ is the convolution operator, and $\mathcal{P}_t^{(\cdot)}(x,y)$ is the patch centered at (x,y) in $\hat{v}_t^{(\cdot)}$. We summarize our use of the TAI network in Fig. 4b.

3.6 Training Strategy

To train bi-TAI, we use both reconstruction-based and adversarial objective functions, the latter of which has been shown by Mathieu et al. [32] to improve the sharpness of predictions. Elaborating on the adversarial objective, we train a discriminator D, which is a binary classification CNN, to distinguish between clips from the dataset and clips generated by bi-TAI. Meanwhile, we train bi-TAI—the "generator"—to not only fool the discriminator, but also generate predictions that resemble the ground truth.

TABLE 1: Summary of the training and testing sets used in our experiments. (*) The HMDB-51 source clips have varying aspect ratios, and thus varying widths.

Dataset	# source clips	Resolution (source)	Resolution (train)	Resolution (val/test)	Grayscale / color	Max p/f (train)	Max m (train)	p/f (val/test)	Small m (val/test)	Large m (val/test)
KTH [28]	2,391	120×160	128×128	128×128	Grayscale	5	5	5	5	10
UCF-101 [29]	13,320	240×320	160×208	240×320	Color	4	3	4	3	5
HMDB-51 [30]	6,849	240×var. (*)	160×208	240×320	Color	4	3	4	3	5
ImageNet-VID [31]	5,354	Varies	-	240×320	Color	-	-	4	3	5

We update the generator and the discriminator in an alternating fashion. In the generator update step, we update bi-TAI by minimizing the following structured loss:

$$\mathcal{L}_g = \alpha\mathcal{L}_{img}\left(\widehat{M}_V^P, M_V\right) + \alpha\mathcal{L}_{img}\left(\widehat{M}_V^F, M_V\right) + \alpha\mathcal{L}_{img}\left(\widehat{M}_V, M_V\right) + \beta\mathcal{L}_{GAN}\left(\widehat{M}_V\right), \quad (10)$$

$$\mathcal{L}_{GAN}\left(\widehat{M}_V\right) = -\log D\left(\left[P_V, \widehat{M}_V, F_V\right]\right), \quad (11)$$

where α and β are hyperparameters to balance the contribution of the reconstruction-based loss $\mathcal{L}_{img}$ and the adversarial loss $\mathcal{L}_{GAN}$. Note that we supervise the final prediction $\widehat{M}_V$ as well as the intermediate predictions $\widehat{M}_V^P$ and $\widehat{M}_V^F$ simultaneously. The loss $\mathcal{L}_{img}$ consists of the squared-error loss $\mathcal{L}_2$ and the image gradient difference loss $\mathcal{L}_{gdl}$ [32], which encourages sharper predictions by penalizing the difference between the image gradients of the ground truth frames and the intermediate/final predictions at every pixel:

$$\mathcal{L}_{img}\left(\widehat{M}_V^{(\cdot)}, M_V\right) = \mathcal{L}_2\left(\widehat{M}_V^{(\cdot)}, M_V\right) + \mathcal{L}_{gdl}\left(\widehat{M}_V^{(\cdot)}, M_V\right), \quad (12)$$

$$\mathcal{L}_2\left(\widehat{M}_V^{(\cdot)}, M_V\right) = \sum_t \left\| v_t - \widehat{v}_t^{(\cdot)} \right\|_2^2, \quad (13)$$

$$\mathcal{L}_{gdl}\left(\widehat{M}_V^{(\cdot)}, M_V\right) = \sum_{t,i,j,k} \left[\left| \nabla v_t - \nabla \widehat{v}_t^{(\cdot)} \right| \right]_{i,j,k}. \quad (14)$$

Here, $\widehat{M}_V^{(\cdot)}$ can be one of the intermediate predictions $\left\{\widehat{M}_V^P, \widehat{M}_V^F\right\}$ or the final prediction $\widehat{M}_V$. In the discriminator update step, we minimize the cross-entropy error:

$$\mathcal{L}_d = -\log D(V) - \log\left(1 - D\left(\left[P_V, \widehat{M}_V, F_V\right]\right)\right). \quad (15)$$

This loss encourages D to assign a high score to real video clips and a low score to clips that include generated middle frames. We use the same discriminator as Villegas et al. [22], but replace each layer that is followed by batch normalization [33] with a spectral normalization layer [34], which we have found results in more accurate predictions.

4 Experiments

4.1 Experimental Setup

4.1.1 Datasets

We perform our experiments on videos from several human action and object video datasets: KTH Actions [28], UCF-101 [29], HMDB-51 [30], and ImageNet-VID [31]. KTH Actions and UCF-101 are commonly used in video prediction and frame interpolation work [16], [22], [32], and HMDB-51 is a dataset we have added to further explore performance on challenging human action videos. ImageNet-VID is used to investigate performance on videos that do not primarily feature humans. We summarize the training and testing sets in Tab. 1, and now proceed to describe them in detail.

KTH Actions contains a total of 2,391 grayscale video clips with resolution 120×160 (height × width), which are divided into a standard training and testing set. Each video clip contains one of 25 subjects performing one of six actions (e.g. handwaving, jogging, boxing, etc.). We divide the standard training set into a smaller training set and a validation set based on the identity of the person in each video (the former includes subjects 1-14, and the latter includes subjects 15-16); these sets are used for training and hyperparameter search respectively. Following Villegas et al. [22], we reduce the resolution to 128×128. We train each model to predict up to five middle frames from up to five preceding and following frames (i.e. up to ten input frames in total); at inference time, we evaluate each model on its ability to predict either five or ten middle frames, given exactly five preceding and five following frames in both cases. The ten-frame case allows us to evaluate generalization performance (similarly to Villegas et al. [22], who double the number of frames to predict between training and testing time).

UCF-101 contains 13,320 RGB video clips from YouTube with resolution 240×320 across 101 action classes (e.g. horse riding, playing guitar, rowing, etc.). It provides three cross-validation folds for action recognition (each fold specifies a training and a test set); we take the test videos from the first fold as our test set and separate the remaining videos into our training and validation sets (clips are separated into an approximate 80-20 split such that the clips from any given source video do not appear in both sets). During training, we reduce the resolution of each video to 160×208 (due to the hardware limitations encountered with our bi-TAI model), and train each model to predict up to three middle frames from up to four preceding and following frames (i.e. up to eight input frames in total). At test time, we scale all videos to 240×320 resolution, and evaluate each model's ability to predict either three or five middle frames given exactly three preceding and three following frames. Due to the large size of this dataset, we only evaluate on the first clip of each test video.

HMDB-51 contains 6,849 RGB video clips across 51 action classes (e.g. golf, hugging, somersaulting, etc.) from movie clips, YouTube, and other publicly available datasets; each video has a fixed height of 240 pixels. The dataset provides three cross-validation folds; we construct the training, validation, and test sets using the same strategy used for UCF-101. The remainder of our experimental setup for HMDB-51 matches our setup for UCF-101.

ImageNet-VID is a video object detection dataset pro-

vided as part of the ImageNet Large Scale Visual Recognition Challenge [31]. We use the 2015 version, which contains 5,354 RGB video clips across 30 animal and vehicle object classes. For this dataset, we use all models pre-trained on UCF-101 and evaluate on the provided test set. To preprocess the test set, we resize all videos to 240×320 and filter out the videos with fewer than 13 frames.

Constructing video clips for training and testing. During training, we construct minibatches by randomly sampling the number of preceding, middle, and following frames (p, m, and f respectively), selecting a video subclip with the appropriate number of frames, and then splitting that clip into the ground truth preceding, middle, and following sequence. Each video clip is randomly flipped horizontally or time-reversed before splitting with probability 0.5 for data augmentation.

We construct the validation and test sets differently for each of the three datasets. For KTH, we extract all subclips across all validation/test video clips from a sliding window of size $T = p + m + f$ and stride s. In our experiments, $p = f = 5$, m is 5 or 10, and s depends on the action class ($s = 3$ for the running and jogging classes, and $s = m$ for the walking, boxing, handclapping, and handwaving classes, following the stride selection process used by Villegas et al. [22]). For UCF-101, HMDB-51, and ImageNet-VID, we only evaluate each model on the first T frames of each video in the test set (following Villegas et al. [22]), where $T = p + m + f$, $p = f = 4$, and m is 3 or 5.

4.1.2 Baselines

As a sanity check, we compare our bi-TAI method to a linear time-weighted average of the last preceding frame and the first following frame, where the weights for the following and preceding frames for time t are $w_t = (t - p)/(m + 1)$ and $1 - w_t$ respectively. We refer to this baseline as `TW_P_F` for ***t***ime-***w***eighted ***p***receding and ***f***ollowing frame. We also compare bi-TAI to state-of-the-art methods for general frame inpainting, video prediction, and frame interpolation [4], [16], [22] to demonstrate that casting our video frame inpainting problem as a different video interpolation/extrapolation task does not yield optimal performance.

The first baseline, Newson et al. [4], is a general video inpainting method that iteratively fills in missing voxel patches with nearest neighbors in the unoccluded portion of the video, using a multi-resolution pyramid to improve the distance metric. For this method, we completely mask the middle frames and run the authors' publicly-available code to recover them. We omit their pre- and post-alignment of video frames in order to compare fairly against the other methods, which lack this step. Note that this method does not have a training phase (it has no parameters to tune based on training data).

The second baseline, MCnet [22], is a video prediction method that sequentially predicts frames by first decomposing the preceding clip into motion difference frames and an RGB content frame, and then regressing this representation to the next frame using an encoder-decoder network. We use this method to predict the middle frames given only the preceding frames as input (this method cannot take following frames because it is a video prediction method). We re-implement their code in PyTorch [35] based on their available implementation in TensorFlow [36]. We use the same loss functions as the original authors to train this model, but for a fairer comparison with bi-TAI, we improve their discriminator by using spectral normalization [34] instead of batch normalization [33].

The final baseline, Super SloMo [16], is a frame prediction method that uses a CNN to predict the optical flow between the two input frames and the frame at an arbitrary intermediate time step. We use their method to predict each middle frame given only the last preceding frame and the first following frame as input (this method cannot take multiple preceding or following frames because it is a frame interpolation method). Since their code is unavailable, we re-implement their method from scratch in PyTorch [35]. To train the model, we use the same loss functions (and their relative weighting) as the original authors.

4.1.3 Training Hyperparameters

We train bi-TAI for 200,000 iterations with a batch size of 4. We use the Adam optimizer [37] with initial learning rate α = 1e-4, first decay rate $\beta_1 = 0.5$, and second decay rate $\beta_2 = 0.999$. In the generator loss, we set the weight of the reconstruction losses α to 1 and the weight of the adversarial loss β to 0.02. The discriminator's spectral normalization layers require a hyperparameter that specifies the number of power iterations used to approximate the spectral norm; we set this value to 3. We use Xavier initialization [38] for each convolutional layer and uniform initialization for each linear layer (with mean 0 and variance 1e-4 for the weights). The biases of each layer are initialized to 0.

For MCnet, we use the same optimization parameters and loss weights as the original authors, and the same number of spectral normalization power iterations as bi-TAI. For Super SloMo, the original authors use the Adam optimizer with an initial learning rate of 1e-4 and 500 total epochs, and divide the learning rate by 10 every 200 epochs. Since this model converges more rapidly on our datasets, we reduce the total epochs and the frequency of learning rate updates. Since the authors do not specify other Adam hyperparameters, we use the same ones used for bi-TAI.

The method by Newson et al. [4] requires hyperparameters for the size of the spatiotemporal patches and the number of levels in the multi-resolution pyramid. We set these values to (3, 3, 3) and 2 for KTH and (5, 3, 3) and 2 for UCF-101 and HMDB-51 (determined by performing grid search on the KTH and UCF-101 validation sets).

4.2 KTH

To evaluate the performance of our bi-TAI model and the proposed baselines, we report the Peak Signal-Noise Ratio (PSNR) and the Structural Similarity (SSIM) [39] between each predicted frame and the ground truth. We report these metrics to be consistent with existing video prediction literature [22], [32], but acknowledge that these metrics have a limited correlation with human perception as noted by Zhang et al. [40].

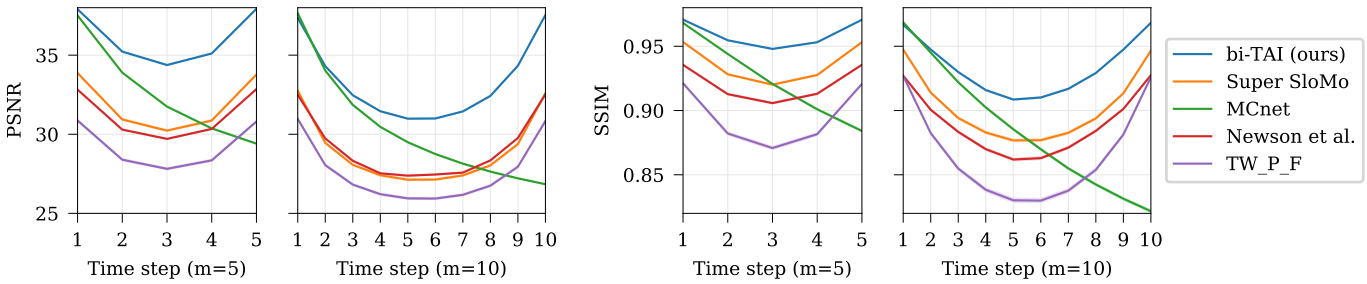


Fig. 5: Performance on the KTH test set for each time step (higher is better).

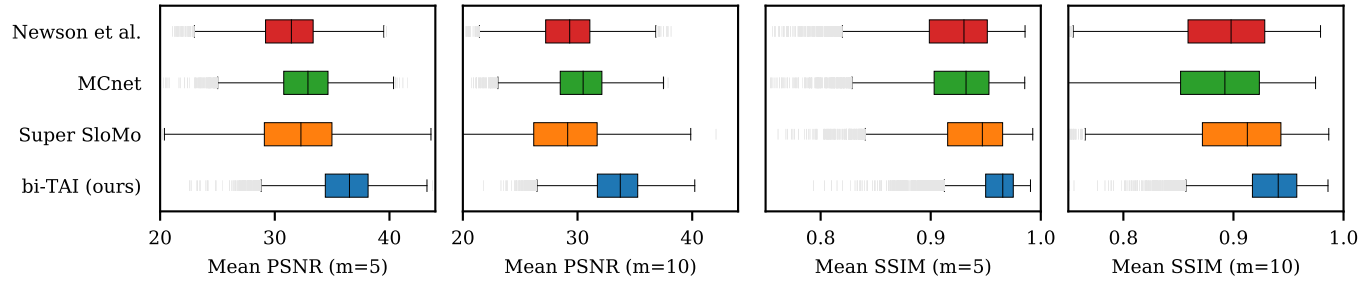


Fig. 6: The distributions of performance on the video clips in the KTH test set. Performance per video is computed as the mean score across all predicted middle frames (higher is better). Outliers are shown as light gray lines.

TABLE 2: Performance on the KTH test set where each value is computed as the mean score across all predicted frames (higher is better). We stylize values that are higher than the others by a statistically significant margin.

	$m = 5$		$m = 10$	
Model	**PSNR**	**SSIM**	**PSNR**	**SSIM**
TW_P_F	29.25±0.053	0.8953±7.27e-4	27.56±0.051	0.8661±8.61e-4
Newson et al.	31.20±0.034	0.9205±4.57e-4	29.11±0.033	0.8890±5.94e-4
MCnet	32.58±0.032	0.9236±4.34e-4	30.21±0.032	0.8844±5.96e-4
Super SloMo	31.93±0.046	0.9365±4.13e-4	28.94±0.045	0.9028±5.78e-4
bi-TAI (ours)	***36.11±0.031***	***0.9594±2.52e-4***	***33.33±0.031***	***0.9340±3.73e-4***

PSNR is defined as a logarithmic function of the inverse of pixel-wise mean-square error between two images:

$$\mathrm{PSNR}(\widehat{x}, x) = 10 \log_{10} \left(\frac{255^2}{\mathrm{MSE}(\widehat{x}, x)} \right) \tag{16}$$

SSIM measures structural similarity as a function of means, variances, and covariances between many corresponding patches between two images, and its value can range between -1 and 1 (we refer the reader to the original paper [39] for more information). We use the implementations of PSNR and SSIM provided by scikit-image [41].

Fig. 5 and Tab. 2 compare the quantitative performance of each method on the KTH test set when predicting five and ten middle frames. As expected, `TW_P_F` performs worse than our method and all state-of-the-art baselines because it does not use any temporal context or motion model to predict the middle frames. Newson et al. [4] does better, but its performance is restricted by its need to borrow spatio-temporal patches from the preceding and following sequence. MCnet yields good performance during the first few frames, but gradually does worse over time because it cannot reconcile the predicted middle frames with the following frames. Super SloMo yields the strongest performance across most metrics, but is restricted by only having access to one preceding and one following frame. Finally, our bi-TAI method significantly outperforms Super SloMo thanks to its ability to aggregate information across all preceding and following frames.

To understand the distribution of each model's performance across the dataset, we compute the average PSNR and SSIM score across all predicted frames for each video, and plot the distribution of these averages in Fig. 6. Our method obtains the highest median and the smallest interquartile range, indicating that its predictions are more stable and of higher quality than the baselines.

Next, we demonstrate in Fig. 7 that compared to the baselines, bi-TAI is better at predicting periodic motion, retaining body structure, and maintaining consistency with both the preceding and the following frames. Observe in the ground truth row that the man lowers his arms down to his sides and then raises them back up. MCnet and Newson et al. [4] fail to predict the arms moving up, despite this motion being observable in the following frames (but recall that MCnet is a video prediction model, so it does not have access to the following frames). Meanwhile, Super SloMo fails to predict the arms moving all the way in, since it lacks the context from multiple preceding and following frames that indicates that the man is moving his arms rather than keeping them still. In contrast, bi-TAI predicts both the inward and the outward motion of the arms. We present additional results generated by our method in Fig. 8, and encourage the reader to view the videos in the supplementary materials.

In Fig. 9, we present a negative result in which the strongest baseline, Super SloMo, outperforms bi-TAI quantitatively. In this example, Super SloMo predicts accurate,

Fig. 7: Qualitative results from the KTH dataset for predicting five middle frames from five preceding and five following frames (we depict every other frame for easier viewing). We indicate preceding and following frames with a green border, predicted middle frames with a yellow border, and ground-truth middle frames with a green border.

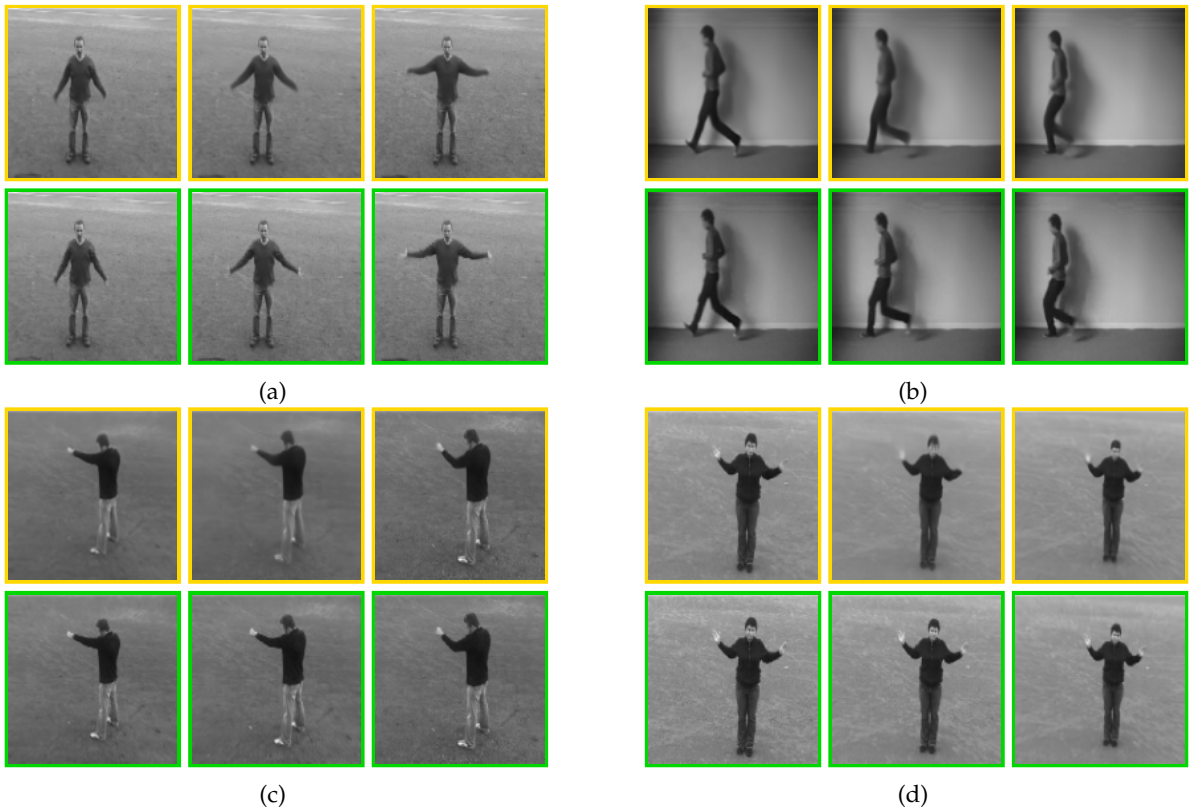


Fig. 8: Additional predictions from bi-TAI on the KTH test set ($m = 5$). The yellow frames indicate predictions from bi-TAI, and green frames indicate the ground truth. We show the first, third, and fifth middle frame for each video.

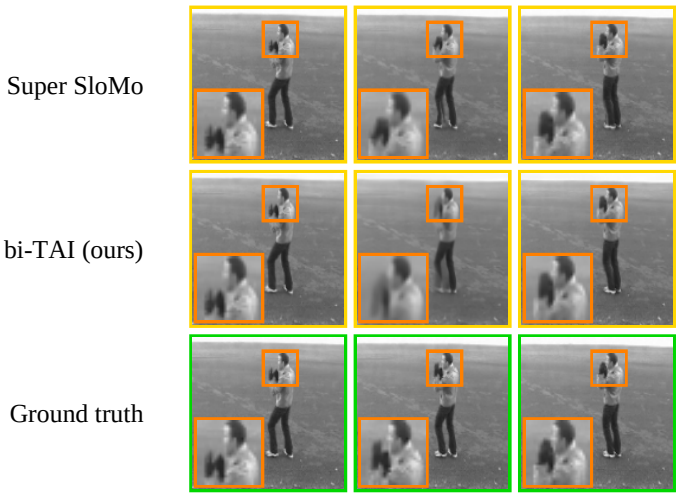


Fig. 9: Negative result on the KTH test set ($m = 5$). We show the first, third, and fifth middle frames, and zoom in on the area indicated in orange.

clear frames, whereas bi-TAI predicts slightly blurry frames (most evident in the most central middle frame). We have found that Super SloMo performs better than bi-TAI if it can accurately predict motion from two input frames (i.e. infer rate and direction of motion without the additional context provided by multiple preceding and following frames). However, the quantitatively lower performance shown in Fig. 5 and Tab. 2 suggests that such cases are rare.

4.3 Ablation Studies

In this section, we perform ablative studies to demonstrate that blending pairs of frames from the forward and backward predictions with a neural network—in particular, *one that is explicitly aware of the time steps corresponding to its inputs*—is key to producing high-quality predictions. For this experiment, we propose three approaches that perform bidirectional prediction as with our method, but blend corresponding pairs of intermediate frames in different ways:

- The *bidirectional simple average model (bi-SA)* blends a pair of frames by simply taking their average.
- The *bidirectional time-weighted average model (bi-TWA)* blends a pair of frames by taking a time-weighted average between them. The weights are $1-w_t$ for the forward prediction frame and w_t for the backward prediction frame, where $w_t = (t-p)/(m+1)$, p and m are the number of preceding and middle frames, and t is the index of a middle frame ($p < t \leq p+m$).
- The *bidirectional temporally-weighted interpolation model (bi-TWI)* is a variant of the bi-TAI model where the time weight w_t is used as a term for summing the modulated bidirectional predictions rather than as a feature channel in the encoder-decoder portion of the interpolation network ϕ_{blend}. To accomplish this, we first remove the injection of w_t as a feature channel within ϕ_{blend}. Then, we replace the simple sum in Eq. 9 with a time-weighted average. In short, we replace Eqs. 8 and 9 with Eqs. 17 and 18 respectively:

$$K_t^P, K_t^F = \phi_{blend}^{enc\_dec}(\pi_t, \rho_t) \ , \tag{17}$$

$$\widehat{v}_t(x,y) = (1-w_t)\left[K_t^P(x,y) * \mathcal{P}_t^P(x,y)\right] + w_t\left[K_t^F(x,y) * \mathcal{P}_t^F(x,y)\right] . \tag{18}$$

TABLE 3: Performance of our bi-TAI model and the ablative variants described in Sec. 4.3 on the KTH test set. Each value is computed as the mean score across all predicted frames (higher is better). We stylize values that are higher than the others by a statistically significant margin.

	$m = 5$		$m = 10$	
Model	**PSNR**	**SSIM**	**PSNR**	**SSIM**
bi-SA	33.69±0.031	0.9456±3.21e-4	30.95±0.030	0.9124±4.58e-4
bi-TWA	35.36±0.031	0.9553±2.73e-4	32.92±0.030	0.9296±3.94e-4
bi-TWI	***36.12±0.032***	0.9585±2.53e-4	***33.33±0.032***	0.9331±3.78e-4
bi-TAI (full)	***36.11±0.031***	***0.9594±2.52e-4***	***33.33±0.031***	***0.9340±3.73e-4***

As with bi-TAI, all of these models are trained from scratch.

In Fig 10b, we show a qualitative comparison between our approach and these methods when predicting the second-to-last middle frame (out of five). Unsurprisingly, bi-SA and bi-TWA produce ghosting artifacts (i.e. multiple faded copies of the actor appear in the predictions) because they do not possess a mechanism to make the bidirectional predictions consistent with each other. bi-TWI reduces the ghosting problem dramatically because the interpolation network can transfer information between the bidirectional predictions as it modulates them; however, ghosting artifacts do occasionally appear (e.g. we see an extraneous faded blob above the head). In contrast, bi-TAI manages to overcome the ghosting issue.

To understand what causes the difference in behavior between bi-TAI and its ablative variants, we visualize the intermediate pixel-space predictions from each model's Bidirectional Video Prediction Network and interpolation network. First, in Fig. 10b, we compare the final predictions of each model to the corresponding outputs of the Bidirectional Video Prediction Network. We observe that the forward and backward predictions can differ substantially from each other (in terms of body pose), but tend to be similar across methods. The discrepancy between the forward and backward predictions explains why simple and time-weighted averages lead to ghosting. bi-TWI and bi-TAI can reduce/eliminate ghosting by modulating the bidirectional predictions with convolutions before summing them.

Next, we emphasize the importance of making the interpolation network temporally-aware via injection of the time weight w_t. Fig. 10c compares the predictions of bi-TWI and bi-TAI with respect to the three pixel-space representations that the interpolation network handles: (i) the two intermediate predictions from the Bidirectional Video Prediction Network; (ii) the two frames output by the interpolation network after adaptive convolution, but before time-weighted averaging (in bi-TWI) or simple summing (in bi-TAI); and (iii) the final predicted frame. Again, we show an instance of predicting the second-to-last frame, where the backward prediction has more weight than the forward prediction. Although the outputs of the Bidirectional Video Prediction Network are similar for both methods, bi-TWI distorts the man's arms when it modulates the backward prediction.

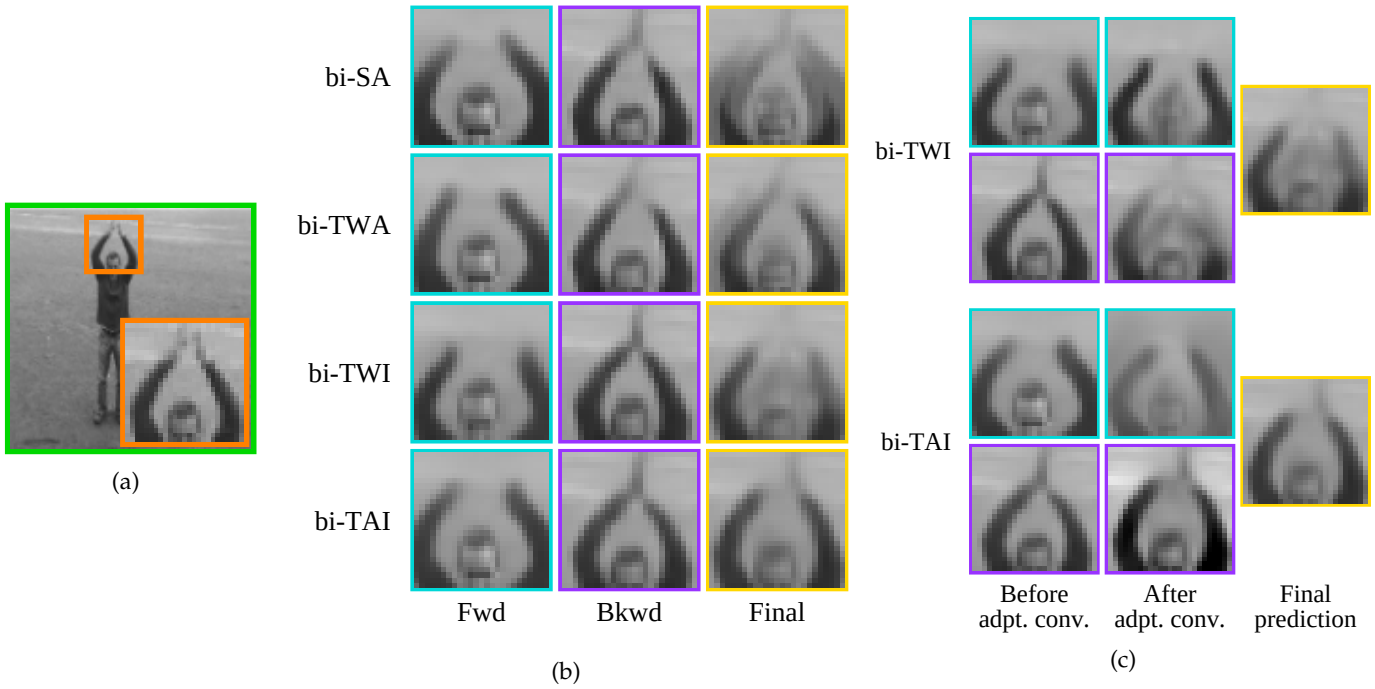


Fig. 10: Qualitative comparison of the various intermediate predictions made by bi-TAI and its ablative variants (Sec. 4.3). We visualize the fourth predicted middle frame (out of five) from a "handwaving" video clip, and zoom in on a specific region (indicated in orange). (a) The ground-truth middle frame. (b) Comparison of the forward and backward predictions from the Bidirectional Video Prediction Network and the final predictions. (c) Inputs and outputs of the interpolation networks of bi-TWI and bi-TAI. The cyan images correspond to the forward prediction frame before and after adaptive convolution, and the purple images correspond to the backward prediction frame before and after adaptive convolution.

TABLE 4: Performance on the UCF-101 and HMDB-51 test sets where each value is computed as the mean score across all predicted frames (higher is better).

	UCF-101				HMDB-51			
	$m = 3$		$m = 5$		$m = 3$		$m = 5$	
Model	**PSNR**	**SSIM**	**PSNR**	**SSIM**	**PSNR**	**SSIM**	**PSNR**	**SSIM**
TW_P_F	29.09±0.110	0.8696±2.203e-3	27.69±0.103	0.8429±2.371e-3	29.65±0.199	0.8474±4.047e-3	27.87±0.181	0.8148±4.155e-3
Newson et al.	28.20±0.091	0.8734±1.868e-3	26.80±0.087	0.8483±2.066e-3	28.94±0.168	0.8521±3.638e-3	27.21±0.156	0.8189±3.811e-3
MCnet	27.15±0.089	0.8447±2.117e-3	25.35±0.083	0.8067±2.308e-3	27.61±0.168	0.8160±4.055e-3	25.65±0.157	0.7725±4.311e-3
Super SloMo	28.86±0.088	0.8876±1.858e-3	27.42±0.087	0.8611±2.084e-3	29.50±0.162	0.8659±3.589e-3	27.85±0.153	***0.8333±3.810e-3***
bi-TAI (ours)	***30.65±0.095***	***0.9033±1.624e-3***	***28.62±0.091***	***0.8697±1.926e-3***	***30.72±0.175***	***0.8782±3.324e-3***	***28.28±0.165***	***0.8372±3.581e-3***

We believe this is because bi-TWI has no information about which of the two input frames is more reliable, which causes the inaccurate forward prediction to corrupt the backward prediction during the modulation process. When bi-TWI applies the time-weighted average to the two modulated outputs, the corruption heavily impacts the final prediction because the backward prediction has more weight. On the other hand, bi-TAI modulates the two predictions by enhancing the backward prediction and reducing the contribution of the forward prediction, which better matches our original intent of adding the interpolation network.

Moving on to quantitative results, Table 3 shows the average PSNR and SSIM score across all middle frames in the KTH test set when predicting five or ten middle frames with bi-TAI and the ablative methods. The quantitative results follow our qualitative analysis: bi-TWI and bi-TAI perform better than bi-SA and bi-TWA because they use an interpolation network to modulate the bidirectional predictions before summing. According to the quantitative metrics, it is not obvious whether bi-TAI or bi-TWI is better—bi-TAI performs better than bi-TWI according to SSIM, but comparably to bi-TWI according to PSNR. However, it is important to note that PSNR is less sensitive to structural changes in reconstructed images than SSIM (PSNR is a logarithmic function of *per-pixel* error, whereas SSIM is based on correlations between corresponding image *patches*). This suggests that SSIM is more suitable for evaluating predicted middle frames than PSNR, especially when comparing structural distortions that tend to occur with the ablated models, and further supports bi-TAI's superior performance.

4.4 UCF-101 and HMDB-51

We continue our analysis by comparing our model to the state-of-the-art baselines on video clips from the UCF-101 [29] and HMDB-51 [30] action classification datasets. Unlike the KTH dataset, these two datasets contain videos with unconstrained camera motion and dynamic lighting, making them substantially more challenging. Fig. 11 shows the average PSNR/SSIM score for each time step when the

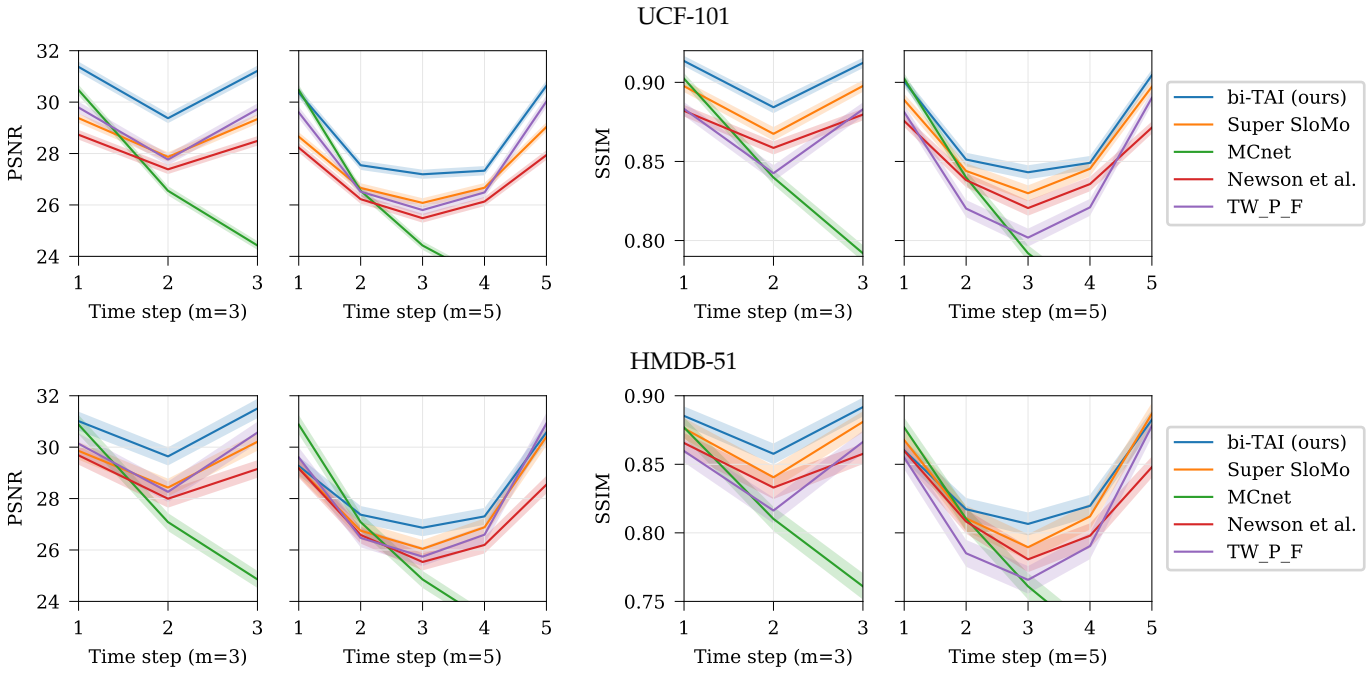


Fig. 11: Performance on the UCF-101 and HMDB-51 test sets for each time step (higher is better). Light colors are used to highlight within two standard errors of each curve.

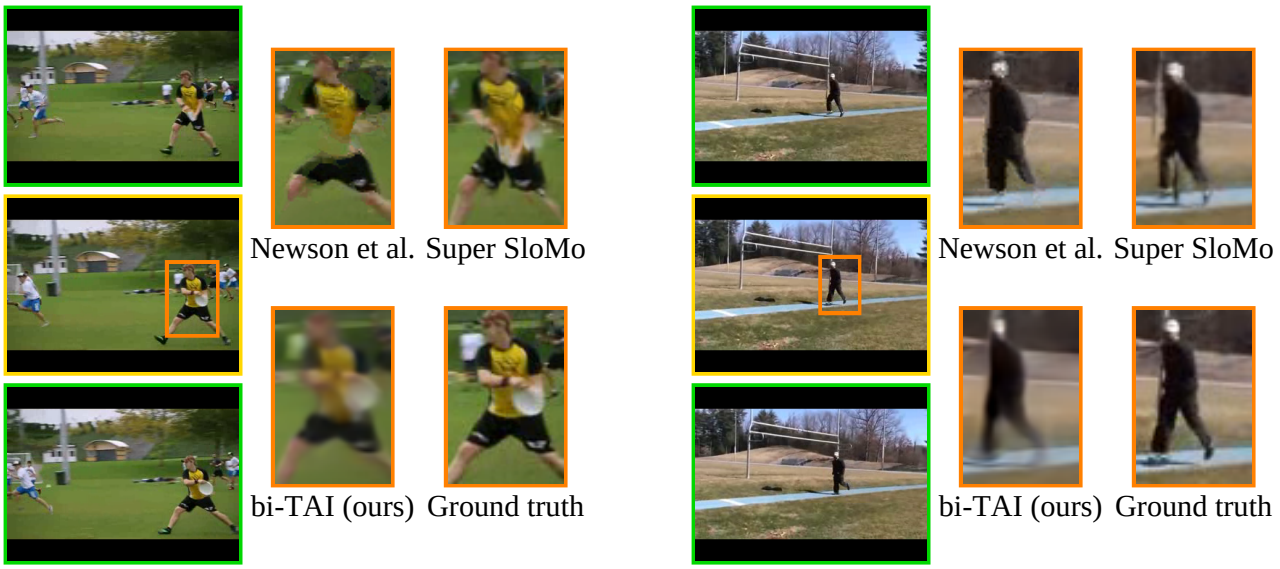
number of middle frames is either 3 or 5 (recall that for these datasets, we train each method to predict at most three middle frames). Our model clearly outperforms the baseline methods when predicting the most central middle frames, but yields a less decisive improvement when predicting the first and last middle frame out of five. In Table 4, we report the performance of each model averaged across all predicted frames, which shows that our method significantly outperforms the baseline methods except on HMDB-51 ($m = 5$) under SSIM, where it performs similarly to Super SloMo.

In Fig. 12, we present two video clips from UCF-101 in which our method outperformed the most competitive baselines, Newson et al. [4] and Super SloMo [16]. We have found that bi-TAI is better at preserving object structure than the baselines in many cases, but it may also generate blurrier predictions. For example, in Fig. 12a, Newson et al. [4] replaces parts of the torso with background pixels, and Super SloMo distorts the area around the arms and the back leg. Furthermore, the pose predicted by Super SloMo differs substantially from the ground truth. On the other hand, bi-TAI maintains a coherent structure for the athlete's body, and more accurately predicts his pose. We observe similar phenomena in Fig. 12b: Newson et al. [4] replaces the person's feet with sidewalk pixels, Super SloMo produces distorted, inconsistent outlines of the front leg, and bi-TAI generates a more consistent, accurate body pose.

In Figure 13, we show a negative result in which our bi-TAI method was outperformed by Newson et al. [4] and Super SloMo quantitatively. Our method performs worse when there is a large amount of camera motion: in these cases, the predictions become excessively blurry. Newson et al. [4] also performs poorly due to its inability to maintain the structure of objects; for example, in Figure 13, the face blends in with the trees. Super SloMo performs the best thanks to its ability to preserve structure and texture under heavy camera motion, but the camera poses in its predictions tend to differ from the ground truth (e.g. the biker's head is further to the left than in the ground truth).

Moving on to the HMDB-51 dataset, we present qualitative results on sampled video clips in Fig 14. Again, we observe that our method is often able to preserve object structure more coherently and more accurately than the baselines. In Fig. 14a, bi-TAI is the only method that successfully retains the entirety of the woman's right arm in its prediction. In Fig 14b, Super SloMo produces strange artifacts near the man's arms (likely remnants of the legs from the preceding and the following frames). Newson et al. [4] generates a more coherent body structure, but the pose is less accurate than in bi-TAI's prediction (e.g. the man's right leg does not bend downward). bi-TAI produces the most accurate and structurally coherent prediction among the evaluated methods.

In Fig. 15, we present a video clip from HMDB-51 where Newson et al. [4] and Super SloMo outperformed our model. Compared to the baselines, our model is relatively worse at handling shot transitions because it tends to predict smooth motion. On the other hand, Newson et al. [4] and Super SloMo can generate abrupt transitions by borrowing pixels/patches solely from the appropriate input sequence. For example, in Fig. 15, an outline of the boy appears in the first frame predicted by bi-TAI, whereas it does not appear in the first frames predicted by Newson et al. [4] and Super SloMo. Despite the existence of several shot transitions in HMDB-51 (where our model is at a disadvantage), bi-TAI still manages to achieve slightly higher performance overall; we suspect that this difference is even clearer when only considering videos without shot transitions.



(a) (b)

Fig. 12: Qualitative results from the UCF-101 dataset ($m = 3$). On the left of each figure, we visualize the last preceding frame in green, the second middle frame in yellow, and the first following frame in green. On the right, we show the prediction from each method at the region indicated in orange.

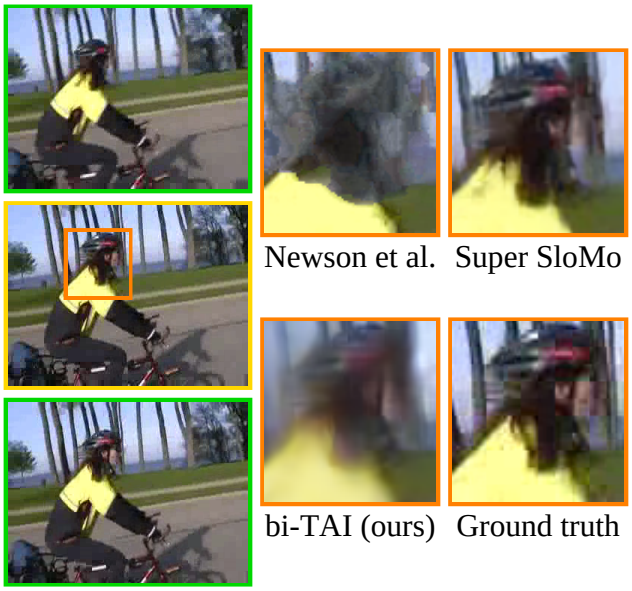


Fig. 13: Failure case from UCF-101 (heavy camera motion).

4.5 ImageNet-VID

We conclude our analysis with ImageNet-VID, where we pre-train each method on UCF-101 and evaluate on the ImageNet-VID test set. We investigate this setting to compare how well each method extrapolates motion from human action datasets to generic videos. In Fig. 16, we show that our model achieves comparable or better quantitative performance than the baselines across all time steps. Table 5 reveals a similar trend—bi-TAI outperforms other models by a significant margin except under the SSIM metric when $m = 5$, where it performs on par with Newson et al. [4].

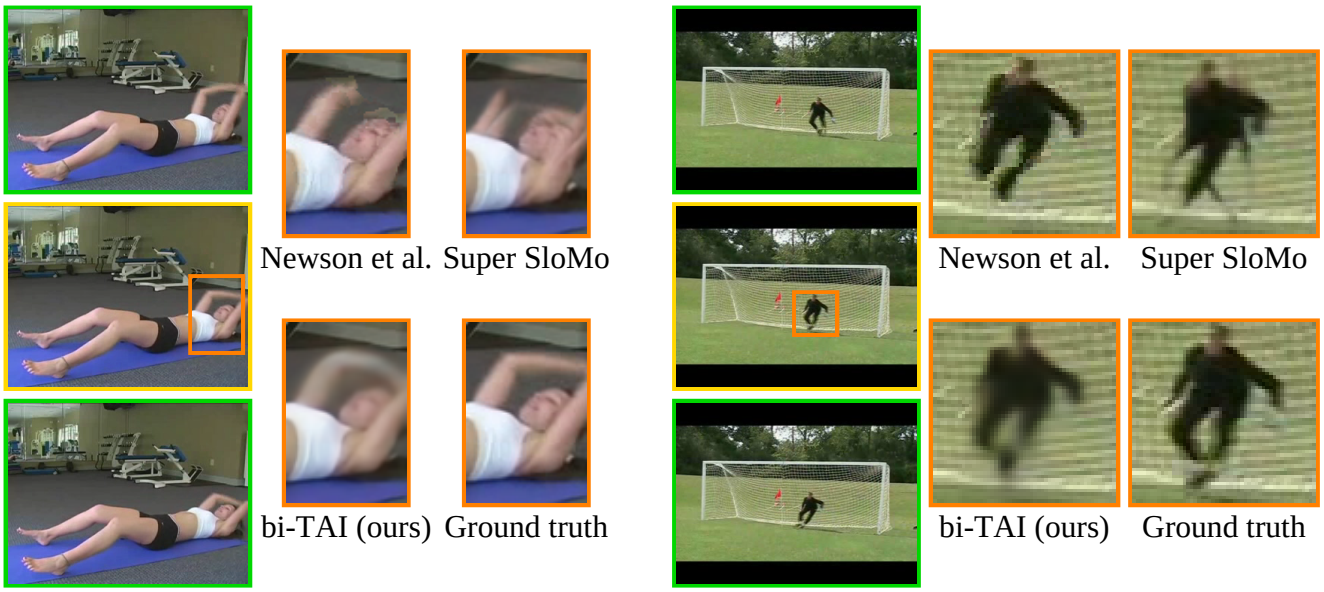
In Fig. 17, we present cases in which our model captured motion better than the baselines. We observe in Fig. 17a that the dog is most visible in bi-TAI's prediction. In Fig. 17b, Newson et al. [4] erase the horse's legs, and Super SloMo produces ghosting leg artifacts. Our model accurately predicts the positions of the horse's legs, but produces a blurry result. The quantitative and qualitative results indicate that our method can extrapolate from human action videos to predict motion in general videos, although blur can still be observed as with UCF-101 and HMDB-51.

TABLE 5: Performance on the ImageNet-VID test set where each value is computed as the mean score across all predicted frames (higher is better). We stylize values that are higher than the others by a statistically significant margin.

	$m = 3$		$m = 5$	
Model	**PSNR**	**SSIM**	**PSNR**	**SSIM**
TW_P_F	26.97±0.249	0.7523±7.11e-3	25.66±0.241	0.7147±7.35e-3
Newson et al.	27.54±0.236	0.8124±5.58e-3	26.12±0.226	***0.7781±6.03e-3***
MCnet	25.46±0.201	0.7471±6.22e-3	23.94±0.187	0.7033±6.44e-3
Super SloMo	26.96±0.205	0.7903±6.15e-3	25.61±0.203	0.7486±6.66e-3
bi-TAI (ours)	***28.39±0.213***	***0.8204±5.36e-3***	***26.74±0.200***	***0.7767±5.82e-3***

5 Conclusion

In this paper, we have tackled the video frame inpainting problem with bi-TAI, which generates two sets of intermediate predictions conditioned on the preceding and following frames respectively, and then blends them together with a novel TAI network. Our experiments on videos from multiple datasets show that our method generates smoother and more accurate predictions than state-of-the-art baselines, particularly on videos that contain articulated body motion and little camera movement. Furthermore, our in-depth analysis has revealed that our bi-TAI network successfully leverages time step information to reconcile inconsistencies in the intermediate predictions.



(a) (b)

Fig. 14: Qualitative results from the HMDB-51 dataset ($m = 3$).

Fig. 15: Failure case from HMDB-51 (shot transitions).

There are several challenges that we plan to address in future work. First, compared to state-of-the-art baselines, our model sometimes generates relatively blurry results, especially under heavy camera motion. This is a common artifact of "pixel synthesis" methods that generate pixels from scratch. Taking inspiration from optical flow-based approaches, which borrow pixels from the input, may help alleviate the blurriness problem. Additionally, our network is very large because it encodes and decodes several pixel-level representations of the preceding, middle, and following frames; this places a limit on the sizes of video clips that can be generated or used to train our model. To address this problem, we aim to explore more tightly-integrated network architectures that generate only one pixel-level prediction. Finally, we plan to explore methods that exploit semantic knowledge about the video content, e.g. by modeling human poses or the periodicity of certain actions.

Acknowledgments

This work is partly supported by ARO W911NF-15-1-0354, DARPA FA8750-17-2-0112 and DARPA FA8750-16-C-0168. It reflects the opinions and conclusions of its authors, but not necessarily the funding agents.

References

[1] Y. Wexler, E. Shechtman, and M. Irani, "Space-Time Video Completion," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2004.

[2] Y.-T. Jia, S.-M. Hu, and R. R. Martin, "Video Completion Using Tracking And Fragment Merging," *The Visual Computer*, vol. 21, no. 8-10, pp. 601–610, 2005.

[3] Y. Shen, F. Lu, X. Cao, and H. Foroosh, "Video Completion For Perspective Camera Under Constrained Motion," in *International Conference on Pattern Recognition*, vol. 3, 2006, pp. 63–66.

[4] A. Newson, A. Almansa, M. Fradet, Y. Gousseau, and P. Pérez, "Video Inpainting Of Complex Scenes," *SIAM Journal on Imaging Sciences*, vol. 7, no. 4, pp. 1993–2019, 2014.

[5] K. A. Patwardhan, G. Sapiro, and M. Bertalmío, "Video Inpainting Under Constrained Camera Motion," *IEEE Transactions on Image Processing*, vol. 16, no. 2, pp. 545–553, 2007.

[6] J. Jia, W. Tai-Pang, Y.-W. Tai, and C.-K. Tang, "Video Repairing: Inference Of Foreground And Background Under Severe Occlusion," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2004.

[7] V. Cheung, B. J. Frey, and N. Jojic, "Video Epitomes," *International Journal of Computer Vision*, vol. 76, no. 2, pp. 141–152, 2008.

[8] M. Granados, K. I. Kim, J. Tompkin, J. Kautz, and C. Theobalt, "Background Inpainting For Videos With Dynamic Objects And A Free-Moving Camera," in *European Conference on Computer Vision*, 2012, pp. 682–695.

[9] M. Ebdelli, O. Le Meur, and C. Guillemot, "Video Inpainting With Short-term Windows: Application To Object Removal And Error Concealment," *IEEE Transactions on Image Processing*, vol. 24, no. 10, pp. 3034–3047, 2015.

[10] A. Borzi, K. Ito, and K. Kunisch, "Optimal Control Formulation For Determining Optical Flow," *SIAM Journal On Scientific Computing*, vol. 24, no. 3, pp. 818–847, 2003.

[11] K. Chen and D. A. Lorenz, "Image Sequence Interpolation Using Optimal Control," *Journal of Mathematical Imaging and Vision*, vol. 41, no. 3, pp. 222–238, 2011.

[12] M. Werlberger, T. Pock, M. Unger, and H. Bischof, "Optical flow guided TV-L1 video interpolation and restoration," in *International Workshop on Energy Minimization Methods in Computer Vision and Pattern Recognition*, 2011, pp. 273–286.

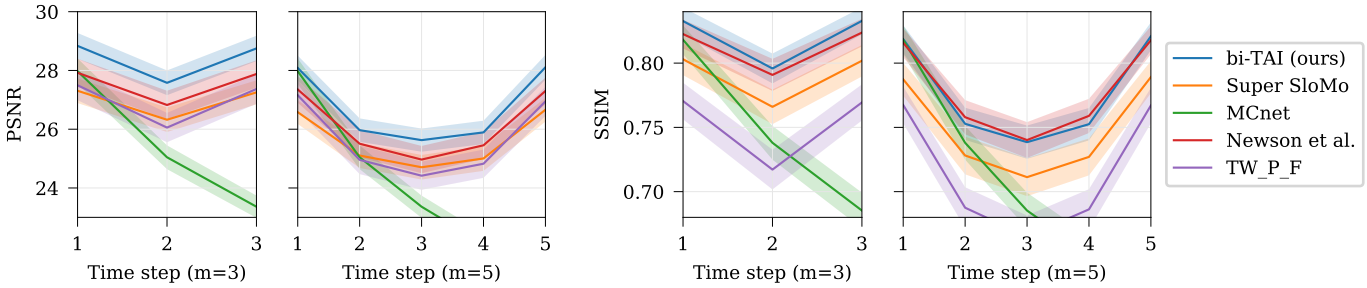


Fig. 16: Performance on the ImageNet-VID test set for each time step (higher is better). Light colors are used to highlight within two standard deviations of each curve.

Fig. 17: Qualitative results from the ImageNet-VID dataset ($m = 3$).

[13] G. Long, L. Kneip, J. M. Alvarez, H. Li, X. Zhang, and Q. Yu, "Learning Image Matching By Simply Watching Video," in *European Conference on Computer Vision*, 2016, pp. 434–450.

[14] S. Niklaus, L. Mai, and F. Liu, "Video Frame Interpolation via Adaptive Separable Convolution," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 261–270.

[15] Z. Liu, R. Yeh, X. Tang, Y. Liu, and A. Agarwala, "Video Frame Synthesis Using Deep Voxel Flow," in *International Conference on Computer Vision (ICCV)*, vol. 2, 2017.

[16] 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 *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[17] C. Vondrick, H. Pirsiavash, and A. Torralba, "Anticipating the future by watching unlabeled video," June 2016.

[18] M. Ranzato, A. Szlam, J. Bruna, M. Mathieu, R. Collobert, and S. Chopra, "Video (Language) Modeling: A Baseline For Generative Models Of Natural Videos," *arXiv preprint arXiv:1412.6604*, 2014.

[19] N. Srivastava, E. Mansimov, and R. Salakhudinov, "Unsupervised Learning Of Video Representations Using LSTMs," in *International Conference On Machine Learning*, 2015, pp. 843–852.

[20] W. Lotter, G. Kreiman, and D. Cox, "Deep Predictive Coding Networks for Video Prediction and Unsupervised Learning," *International Conference on Learning Representations*, 2017.

[21] N. Kalchbrenner, A. v. d. Oord, K. Simonyan, I. Danihelka, O. Vinyals, A. Graves, and K. Kavukcuoglu, "Video Pixel Networks," in *International Conference On Machine Learning*, 2017.

[22] R. Villegas, J. Yang, S. Hong, X. Lin, and H. Lee, "Decomposing Motion And Content For Natural Video Sequence Prediction," *International Conference on Learning Representations*, 2017.

[23] K.-H. Zeng, W. B. Shen, D.-A. Huang, M. Sun, and J. Carlos Niebles, "Visual forecasting by imitating dynamics in natural sequences," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 2999–3008.

[24] M. Babaeizadeh, C. Finn, D. Erhan, R. H. Campbell, and S. Levine, "Stochastic variational video prediction," in *International Conference on Learning Representations*, 2018. [Online]. Available: https://openreview.net/forum?id=rk49Mg-CW

[25] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," *International Conference on Learning Representations*, 2015.

[26] X. Shi, Z. Chen, H. Wang, D.-Y. Yeung, W.-K. Wong, and W.-C. Woo, "Convolutional Lstm Network: A Machine Learning Approach For Precipitation Nowcasting," in *Advances In Neural Information Processing Systems*, 2015, pp. 802–810.

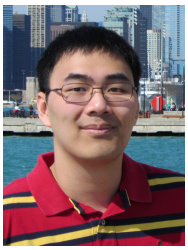
[27] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *science*, vol. 313, no. 5786, pp. 504–507, 2006.

[28] C. Schuldt, I. Laptev, and B. Caputo, "Recognizing Human Actions: A Local Svm Approach," in *International Conference on Pattern Recognition*, vol. 3, 2004, pp. 32–36.

[29] 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.

[30] H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, and T. Serre, "HMDB: A Large Video Database For Human Motion Recognition," in *IEEE International Conference on Computer Vision*, 2011, pp. 2556–2563.

[31] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein *et al.*, "Imagenet large scale visual recognition challenge," *International journal of computer vision*, vol. 115, no. 3, pp. 211–252, 2015.
[32] M. Mathieu, C. Couprie, and Y. LeCun, "Deep Multi-Scale Video Prediction Beyond Mean Square Error," *International Conference on Learning Representations*, 2016.
[33] S. Ioffe and C. Szegedy, "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift," in *International Conference On Machine Learning*, 2015, pp. 448–456.
[34] T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida, "Spectral Normalization for Generative Adversarial Networks," in *International Conference on Learning Representations*, 2018.
[35] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in pytorch," in *NIPS-W*, 2017.
[36] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard *et al.*, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015, software available from tensorflow.org. [Online]. Available: https://www.tensorflow.org/
[37] D. P. Kingma and J. L. Ba, "ADAM: A Method For Stochastic Optimization," in *International Conference on Learning Representations*, 2015.
[38] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, 2010, pp. 249–256.
[39] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image Quality Assessment: From Error Visibility To Structural Similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, 2004.
[40] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, "The unreasonable effectiveness of deep features as a perceptual metric," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 586–595.
[41] S. Van der Walt, J. L. Schönberger, J. Nunez-Iglesias, F. Boulogne, J. D. Warner, N. Yager, E. Gouillart, and T. Yu, "scikit-image: image processing in python," *PeerJ*, vol. 2, p. e453, 2014.



Ryan Szeto is a Ph.D. student of the Computer Science and Engineering division of Electrical Engineering and Computer Science at the University of Michigan. He received his M.S. in Computer Science and Engineering at the University of Michigan in 2017 and dual B.S. degrees in Computer Science and Mathematics at the University of Massachusetts in 2015. His research interests include computer vision and deep learning, with an emphasis on conditional generative video models.

Ximeng Sun is a Ph.D. student of Computer Science at Boston University. She received her M.S. in Electrical and Computer Engineering at the University of Michigan in 2018 and a B.S. in Communications Engineering at Beijing University of Posts and Telecommunications in 2016. Her interests include deep learning and computer vision. Her recent research focuses on applying deep generative models to video-level tasks and learning disentangled representations in an unsupervised manner.

Kunyi Lu is a M.S. student of the Computer Science and Engineering division of Electrical Engineering and Computer Science at the University of Michigan. He received his B.S. in Electrical Information Engineering at Wuhan University in 2017. His research focuses on computer vision and machine learning, especially on video understanding and generative video models.

Jason J. Corso is currently an Associate Professor of Electrical Engineering and Computer Science at the University of Michigan. He received his Ph.D. in Computer Science at The Johns Hopkins University in 2005. He is a recipient of the NSF CAREER award (2009), ARO Young Investigator award (2010), Google Faculty Research Award (2015) and on the DARPA CSSG. His main research thrust is high-level computer vision and its relationship to human language, robotics and data science. He primarily focuses on problems in video understanding such as video segmentation, activity recognition, and video-to-text. From biomedicine to recreational video, imaging data is ubiquitous. Yet, imaging scientists and intelligence analysts are without an adequate language and set of tools to fully tap the information-rich image and video. He works to provide such a language; specifically, he primarily studies the coupled problems of segmentation and recognition from a Bayesian perspective emphasizing the role of statistical models in efficient visual inference. His long-term goal is a comprehensive and robust methodology of automatically mining, quantifying, and generalizing information in large sets of projective and volumetric images and video.