A Temporally-Aware Interpolation Network for Video Frame Inpainting

Ximeng Sun⋆, Ryan Szeto⋆, and Jason J. Corso

University of Michigan, Ann Arbor, USA
{sunxm, szetor, jjcorso}@umich.edu

Abstract. We propose the first deep learning solution to video frame inpainting, a more challenging but less ambiguous task than related problems such as general video inpainting, frame interpolation, and video prediction. We devise a pipeline 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, using time information and hidden activations from the video prediction module to resolve disagreements between the predictions. Our experiments demonstrate that our approach produces more accurate and qualitatively satisfying results than a state-of-the-art video prediction method and many strong frame inpainting baselines. Our code is available at https://github.com/sunxm2357/TAI_video_frame_inpainting.

Keywords: Video Inpainting · Video Prediction · Frame Interpolation

1 Introduction

In this work, we explore the video frame inpainting problem, i.e. the task of reconstructing a missing sequence of frames given both a sequence of *preceding* frames and a sequence of *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. This task is more challenging than general video inpainting because we aim to fill in whole, temporally-contiguous frames rather than small spatio-temporal regions. It is also less ambiguous—and therefore more well-defined and easier to evaluate—than frame interpolation and video prediction, where methods cannot access the contextual information required to rule out many plausible predictions.

We present the first deep neural network for video frame inpainting, which approaches the problem in two steps as shown in Fig. 1. 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

---

⋆ indicates equal contribution

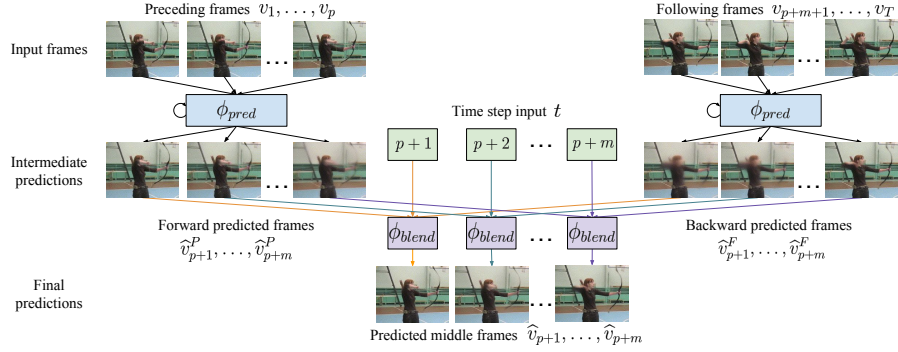Fig. 1: We predict middle frames by blending forward and backward intermediate video predictions with a Temporally-Aware Interpolation (TAI) network

"backward prediction" conditioned on the following frames. Then, we blend each pair of corresponding frames together to obtain the final prediction.

Our blending strategy exploits three characteristics of our intermediate prediction process. First, a pair of intermediate prediction 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 blending neural network that can cleanly merge a given pair of predictions 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 neural network whose hidden features may be useful for blending. To leverage these features, we feed them to the blending network as additional inputs. We call our blending module the **Temporally-Aware Interpolation Network** (or TAI network for short). As we show in our experiments, our approach yields the most visually satisfying predictions among several strong baselines and across multiple human action video datasets.

In summary, we make the following contributions. First, we propose a deep neural network for video frame inpainting that generates two intermediate predictions and blends them with our novel TAI network. Second, we propose and compare against several baselines that leverage the information provided by the preceding and following frames, but do not utilize our TAI blending approach. Finally, we demonstrate that our approach is quantitatively and qualitatively superior to the proposed baselines across three human action video datasets.

2   Related Work

In the *general video inpainting problem* (of which our video frame inpainting task is a challenging instance), 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. 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 [30,9,23,18]; *object-based* methods that separate spatial content into layers (e.g. foreground and background), repair them individually, and stitch them back together [20,8]; and *probabilistic model-based* methods that assign values that maximize the likelihood under some probabilistic model [3,6,4]. Many of these approaches make strong assumptions about the video content, such as constrained camera pose/motion [23,20,8] or static backgrounds [20,8,6]. In addition, they are designed for the case in which "holes" in the video are localized to small spatio-temporal regions, and may therefore perform poorly when whole, contiguous frames are missing. Finally, 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 [1,2,29], recent approaches train deep neural networks to predict one intermediate frame [14,19,13]. However, all of these approaches require input frames that occur within a miniscule window of time (i.e. no more than 0.05 seconds apart), whereas we are interested in predicting on larger time scales. Furthermore, the task is ambiguous because a pair of individual frames without temporal context cannot sufficiently constrain the appearance of the intermediate frames (for instance, if we observed two frames of a swinging pendulum, we would need its period of oscillation to rule out several plausible appearances). As a result, it is hard to evaluate plausible predictions that deviate from the actual data.

Video prediction, where the goal is to generate the future frames that follow a given sequence of video frames, is yet another actively-studied area with an important limitation. The earliest approaches draw heavily from language modeling literature by extending simple recurrent sequence-to-sequence models to predict patches of video [21,26]; more recent methods utilize structured models that decompose the input data and/or the learned representations in order to facilitate training [15,10,27]. As with frame interpolation, video prediction is inherently underconstrained since the past can diverge into multiple plausible futures.

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 se-
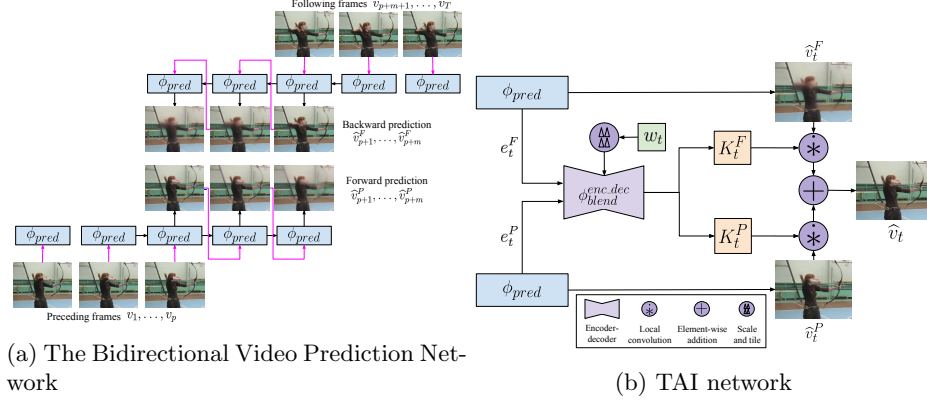
(a) The Bidirectional Video Prediction Network

(b) TAI network

Fig. 2: Architectures of two modules in our model

quences of preceding, middle, and following frames from V respectively. We seek a function ϕ that satisfies $M_V = \phi(P_V, F_V)$ for all V.

3.2 Model Overview

We propose a novel deep neural network to approximate the video inpainting function ϕ (see Fig. 1). Instead of learning a direct mapping from the preceding and following sequences to the middle sequence, our model decomposes the problem into two sub-problems and tackles each one sequentially with two tractable 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 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 in an end-to-end manner.

3.3 Bidirectional Video Prediction Network ϕ_{pred}

We first use the Bidirectional Video Prediction Network ϕ_{pred}, shown in Fig. 2a, to produce two intermediate predictions—a "forward prediction" $\widehat{M}_V^P = \{\widehat{v}_{p+1}^P, \ldots,$

$\widehat{v}_{p+m}^P\}$ and a "backward prediction" $\widehat{M}_V^F = \{\widehat{v}_{p+1}^F, \ldots, \widehat{v}_{p+m}^F\}$—by conditioning on the preceding sequence P_V and the following sequence F_V respectively:

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

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

where $(\cdot)^R$ is an operation that reverses the input sequence. Note that the same parameters are used to generate the forward and backward predictions.

In particular, the Bidirectional Video Prediction Network recurrently generates one frame at a time by conditioning on all previously generated frames. For example, in the case of the forward prediction:

$$\widehat{v}_{k+1}^P = \phi_{pred}\big(\{\tilde{v}_1^P, \tilde{v}_2^P, \ldots, \tilde{v}_k^P\}\big) , \tag{3}$$

where for a given t, $\tilde{v}_t^P$ is either v_t (an input frame) if $t \in \{1, \ldots, p\}$ or $\widehat{v}_t^P$ (an intermediate predicted frame) if $t \in \{p+1, \ldots, p+m\}$. We also store intermediate activations from the Bidirectional Video Prediction Network (denoted as e_t^P), which serve as inputs to the Temporally-Aware Interpolation Network. We apply an analogous procedure to obtain the backward prediction and its corresponding intermediate activations.

3.4   Temporally-Aware Interpolation Network ϕ_{blend}

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

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

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

Blending $\widehat{v}_t^P$ and $\widehat{v}_t^F$ 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. $\widehat{v}_t^P$ is more reliable for earlier time steps). As we show in Sec. 4, equally averaging $\widehat{v}_t^P$ and $\widehat{v}_t^F$ 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 also exhibits the same problem.

In order to blend corresponding frames more accurately, our Temporally-Aware Interpolation (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 e_t^P and e_t^F. We feed w_t to our interpolation network so it can learn how to incorporate the unequal reliability

of $\widehat{v}_t^P$ and $\widehat{v}_t^F$ into its final prediction; we feed e_t^P and e_t^F to leverage the high-level semantics that the Bidirectional Video Prediction Network has learned. We contrast standard interpolation with TAI algebraically:

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

$$\widehat{v}_t = \phi_{TAI}\left(\widehat{v}_t^P, e_t^P, \widehat{v}_t^F, e_t^F, w_t\right). \tag{7}$$

TAI blends pairs of intermediate frames $(\widehat{v}_t^P, \widehat{v}_t^F)$ by first applying a unique, adaptive 2D kernel to each patch in the two input frames, and then summing the resulting images pixel-wise *à la* Niklaus et al. [19]. To generate the set of adaptive kernels, we use an encoder-decoder model, shown in Fig. 2b, that takes in the intermediate activations from the Bidirectional Video Prediction Network, e_t^P and e_t^F, and the scaled time step w_t:

$$K_t^P, K_t^F = \phi_{blend}^{enc\_dec}\left(e_t^P, e_t^F, w_t\right), \tag{8}$$

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. Note that we inject the scaled time step by replicating it spatially and concatenating it to the output of one of the decoder's hidden layers as an additional channel. Afterwards, we apply the adaptive kernels to each input frame and sum the resulting images pixel-wise:

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

where $\widehat{v}_t(x, y)$ is the pixel value of the final prediction 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 $\mathcal{P}_{(\cdot)}(x, y)$ is the patch centered at (x, y) in $\widehat{v}_t^{(\cdot)}$.

3.5   Training Strategy

To train our complete video frame inpainting model, we use both reconstruction-based and adversarial objective functions, the latter of which has been shown by Mathieu et al. [16] to improve the sharpness of predictions. In our case, we train a discriminator D, which is a binary classification CNN, to distinguish between clips from the dataset and clips generated by our model. Meanwhile, we train our model—the "generator"—to not only fool the discriminator, but also generate predictions that resemble the ground truth.

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

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

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

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 $\mathcal{L}_{gdl}$ [16], which encourages sharper predictions by penalizing differences along the edges in the image:

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

$$\mathcal{L}_2\left(\widehat{M}_V^{(\cdot)}, M_V\right) = \sum_{t=p+1}^{p+m} \left\| v_t - \widehat{v}_t^{(\cdot)} \right\|_2^2 , \tag{13}$$

$$\mathcal{L}_{gdl}\left(\widehat{M}_V^{(\cdot)}, M_V\right) = \sum_{t=p+1}^{p+m} \sum_{i,j}^{h,w} \left(\left| |v_t(i,j) - v_t(i-1,j)| - |\widehat{v}_t^{(\cdot)}(i,j) - \widehat{v}_t^{(\cdot)}(i-1,j)| \right| \right.$$
$$\left. + \left| |v_t(i,j-1) - v_t(i,j)| - |\widehat{v}_t^{(\cdot)}(i,j-1) - \widehat{v}_t^{(\cdot)}(i,j)| \right| \right) . \tag{14}$$

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 update the discriminator by minimizing the cross-entropy error:

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

We use the same discriminator as Villegas et al. [27], but replace each layer that is followed by batch normalization [7] with a spectral normalization layer [17], which we have found results in more accurate predictions.

4   Experiments

4.1   Experimental Setup

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). We instantiate the Bidirectional Video Prediction Network with MC-Net [27]. As for the Temporally-Aware Interpolation Network, we modify the Separable Adaptive Kernel Network [19] to take as input intermediate activations and scaled time steps (refer to the supplementary materials for architectural details). These choices afford us two benefits: (i) the chosen networks are, to the best of our knowledge, the best-performing models in their original tasks, enabling us to demonstrate the full potential of our approach; and (ii) both networks are fully-convolutional, allowing us to modify the video resolution at test time.

We compare our video frame inpainting model to several baselines (Sec. 4.2) on videos from three human action datasets: KTH Actions [22], HMDB-51 [12],

and UCF-101 [25]. KTH Actions contains a total of 2,391 grayscale videos with resolution 120×160 (height $\times$ width) across six action classes; it also provides a standard training and testing set. We divide the standard training set into a smaller training set and a validation set, which are used for training and hyperparameter search respectively. Following Villegas et al. [27], we reduce the resolution to 128×128. We train each model to predict five middle frames from five preceding and five following frames; at inference time, we evaluate each model on its ability to predict ten middle frames from five preceding and five following frames. We double the number of frames to predict at test time in order to evaluate generalization performance (following Villegas et al. [27]).

HMDB-51 contains 6,849 RGB videos across 51 action classes; each video has a fixed height of 240 pixels. The dataset provides three cross-validation folds (each including 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. During training, we reduce the resolution of each video to 160×208, and train each model to predict three middle frames from four preceding and four following frames. At test time, we scale all videos to 240×320 resolution (following Villegas et al. [27]) and take in the same number of preceding/following frames, but predict five frames in the middle.

UCF-101 contains 13,320 RGB videos with resolution 240×320 across 101 action classes. It provides three cross-validation folds for action recognition; we take the test videos from the first fold as our test set and divide the remaining videos into our training and validation sets. The remainder of our experimental setup for UCF-101 matches our setup for HMDB-51.

4.2   Baselines

The first baseline we compare our method to is MC-Net [27]—we re-implement and train their model to predict the middle frames conditioned only on the preceding frames. We also introduce two classes of baselines specifically designed for the video frame inpainting problem. In the first class, instead of learning a function ϕ, we hand-craft several ϕ's that can perform well on certain video prediction tasks, particularly on videos with little movement or periodic motion. The baselines described by Eqs. 16-18 copy or take a simple average of the last preceding frame v_p and the first following frame v_{p+m+1}:

$$\phi_{\text{repeat\_P}}\left(P_V, F_V\right) = \{v_p, v_p, \ldots, v_p\}\,, \tag{16}$$

$$\phi_{\text{repeat\_F}}\left(P_V, F_V\right) = \{v_{p+m+1}, v_{p+m+1}, \ldots, v_{p+m+1}\}\,, \tag{17}$$

$$\phi_{\text{SA\_P\_F}}\left(P_V, F_V\right) = \{\widehat{v}, \widehat{v}, \ldots, \widehat{v}\}\,, \text{where } \widehat{v} = \frac{v_p + v_{p+m+1}}{2}\,. \tag{18}$$

Also, we try incorporating the scaled time step of the predicted frame $w_t = (t-p)/(m+1)$ by computing a time-weighted average of v_p and v_{p+m+1}:

$$\phi_{\text{TW\_P\_F}}\left(P_V, F_V\right) = \{\widehat{v}_{p+1}, \widehat{v}_{p+2}, \ldots, \widehat{v}_{p+m}\}\,, \tag{19}$$

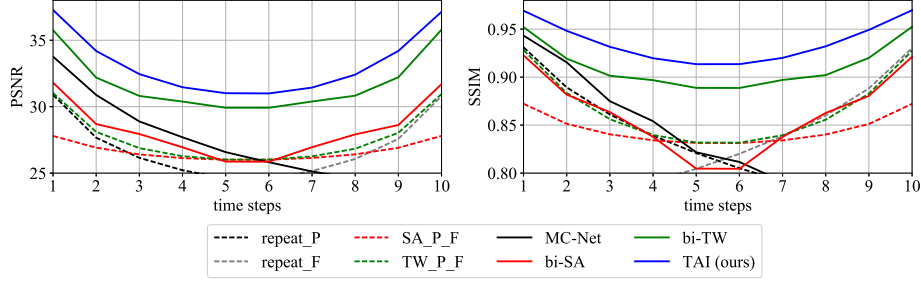$$\widehat{v}_t = \left(1 - w_t\right)v_p + w_t v_{p+m+1}\,. \tag{20}$$

Fig. 3: Quantitative results on the KTH Actions dataset (for both PSNR and SSIM, higher is better). We compare our full model (TAI) to the baselines described in Sec. 4.2

In the second class of baselines, we highlight the value of our TAI module by proposing two bidirectional prediction models that use the same Bidirectional Video Prediction Network architecture as our full model, but blend the forward and backward predictions without an interpolation network. Instead, they blend by computing either a simple average (bi-SA, Eq. 21) or a weighted average based on the scaled time step w_t (bi-TW, Eq. 22):

$$\widehat{v}_t = \left(\widehat{v}_t^P + \widehat{v}_t^F\right)/2\,, \tag{21}$$

$$\widehat{v}_t = (1 - w_t)\,\widehat{v}_t^P + w_t \widehat{v}_t^F\,. \tag{22}$$

All baselines are trained independently from scratch.

4.3   KTH Actions

To evaluate the performance of the proposed baselines and our full model with the TAI network (we refer to this full model as TAI for brevity), we follow existing video prediction literature [27,16] by reporting the Structural Similarity (SSIM) [28] and Peak Signal-Noise Ratio (PSNR) between each predicted frame and the ground truth. We draw a series of conclusions from the quantitative comparison shown in Fig. 3. First, the low performance of the hand-crafted baselines (the dashed curves in Fig. 3) indicate that our task is challenging, and requires a model that generates a *non-trivial* prediction from *multiple* preceding and following frames. Second, the performance of MC-Net drops quickly over time due to its lack of guidance from the following frames. Third, between the bidirectional prediction baselines, bi-TW does a better job than bi-SA since it incorporates the scaled time step w_t via a hand-crafted, time-weighted average. Finally, TAI outperforms bi-TW because it learns a complex blending function that leverages both time step information and intermediate activations from the Bidirectional Video Prediction Network.

In Fig. 4, we visualize the predictions made by MC-Net, bi-SA, bi-TW, and TAI (we encourage the reader to view additional results in the supplementary

Fig. 4: Comparison of predictions from our TAI model to baseline methods on KTH Actions. We visualize every other frame of the input and predicted sequences. Refer to the supplementary materials for more results

materials). MC-Net generates blob-like poses that fail to preserve the proper shape of the body and are inconsistent with the following frames. Meanwhile, bi-SA and bi-TW generate frames with a noticeable "ghosting" effect (e.g. both predictions contain two torsos overlapping with each other), leading to a drop in PSNR and SSIM scores. On the other hand, TAI overcomes these challenges: its predictions are consistent with both the preceding and following frames, and they contain one unified, well-shaped torso. We have found that SSIM drops more drastically than PSNR when ghosting occurs, suggesting that it correlates better with human-perceived quality than PSNR.

4.4   Qualitative Analysis of Blending Methods

Next, we visualize the forward, backward, and final predictions of bi-SA, bi-TW, and TAI in order to highlight the benefit of a learned blending function over a hand-crafted one. Across all three models, the forward prediction is inconsistent with the backward one for most videos. For instance, in Fig. 5, the scale of the actor always differs between the forward and backward predictions. However, the quality of the final prediction improves with the complexity of the blending strategy. For example, since bi-SA blends the two predictions evenly, we observe in the final prediction for Fig. 5a a blurry background and two outlines of the actor's body; in Fig. 5b, we see the outlines of two heads. bi-TW produces similar artifacts to bi-SA for both videos, but its final predictions are clearer. Finally, TAI reconciles the differences between the forward and backward predictions without introducing ghosting artifacts: in Fig. 5a, the final prediction compromises between the actor's sizes from the intermediate predictions, and in Fig. 5b, the difference in the actor's head position is resolved, resulting in a clean outline of the head. We conclude that even though all three methods generate
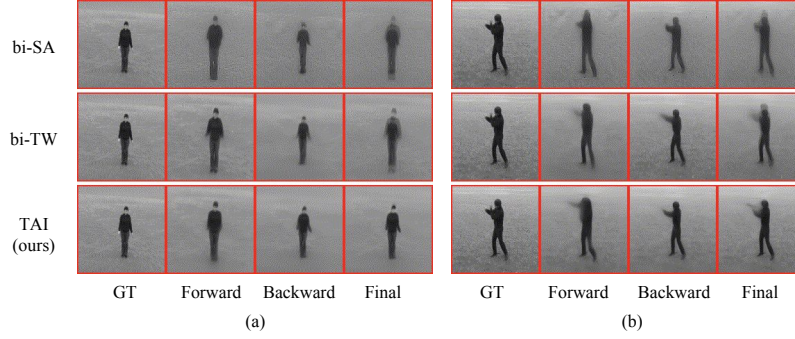
Fig. 5: Comparison of the forward, backward, and final predictions for the third middle frame (of ten) of two videos

(a) SSIM   (b) Example videos
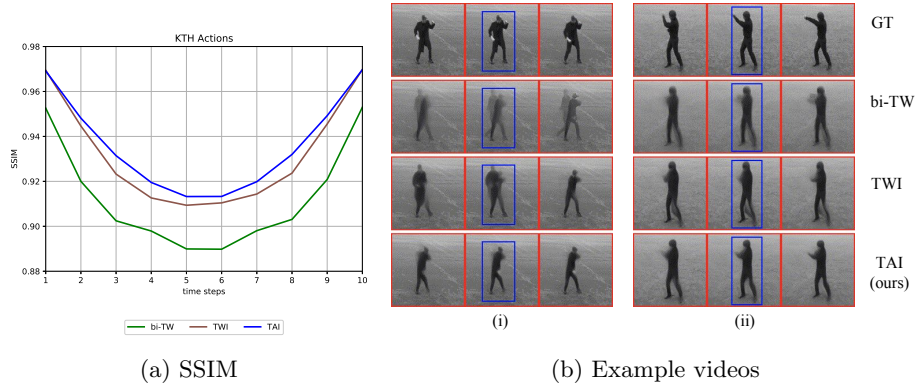
Fig. 6: Ablative comparison between bi-TW, TWI, and our full TAI model. Higher SSIM is better

inconsistent forward and backward predictions, TAI can successfully reconcile the differences to generate a crisp final prediction.

4.5   Ablation Studies

Feeding time information into the blending network such that it can *learn* to use that information most effectively is key to generating high-quality predictions. To verify this, we replace the blending module with a time-agnostic interpolation network and apply a time-weighted average to its outputs; we call this version the time-weighted interpolation (TWI) network. In Fig. 6, we compare bi-TW, TAI, and a bidirectional video prediction model with TWI. We see that TWI performs better than bi-TW both quantitatively and qualitatively because the ghosting artifacts in its predictions are less apparent. However, it still incorporates time information with a hand-crafted function, which prevents TWI from completely
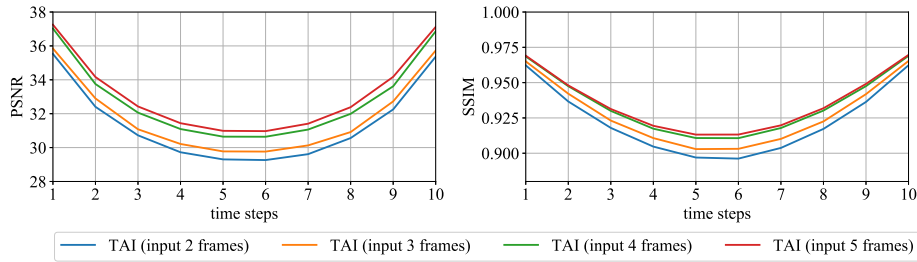
Fig. 7: Performance of our trained model with 2-5 preceding and following frames at test time on the KTH Actions dataset. Higher PSNR and SSIM is better

avoiding ghosting artifacts. For example, TWI generates two torsos in Fig. 6b (i) and a fake leg between two legs in Fig. 6b (ii). On the other hand, TAI avoids ghosting artifacts more successfully than TWI: for both videos in Fig. 6b, we see that TAI generates a clear, sharp outline around the actor without introducing artificial torsos or limbs.

4.6   Importance of Context Frames

In this section, we show our model's ability to leverage the context information from the preceding and the following sequences which, as argued in Sec. 1, is vital to performing well on the video frame inpainting task. In Fig. 7, we plot the quantitative performance of our trained model as we increase the number of available frames at test time from two to five (recall that we train our model on five preceding and following frames). Our model obtains progressively better PSNR and SSIM values as we increase the number of preceding and following frames; this shows that our model successfully leverages the increasing amount of context information to improve its predictions.

4.7   HMDB-51 and UCF-101

We conclude our experiments by demonstrating our model's ability to perform well on complex videos depicting challenging scenes and a wide variety of actions. We do this by comparing our full TAI model to the baselines proposed in Sec. 4.2 on videos from HMDB-51 and UCF-101. We see from the quantitative results in Fig. 8 that none of the baselines outperform the others by a definitive margin. This contrasts with our findings in Sec. 4.3 where we found that for KTH Actions, bi-TW produces substantially better predictions than all the other baselines. We note that the biggest difference between KTH Actions and HMDB-51/UCF-101 is that the scenes in HMDB-51 and UCF-101 are far more complex than in KTH Actions; this suggests that bi-TW performs poorly when observing complex scenes. However, our model still outperforms all baselines on HMDB-51 and UCF-101, suggesting that it is best equipped for handling complex videos.
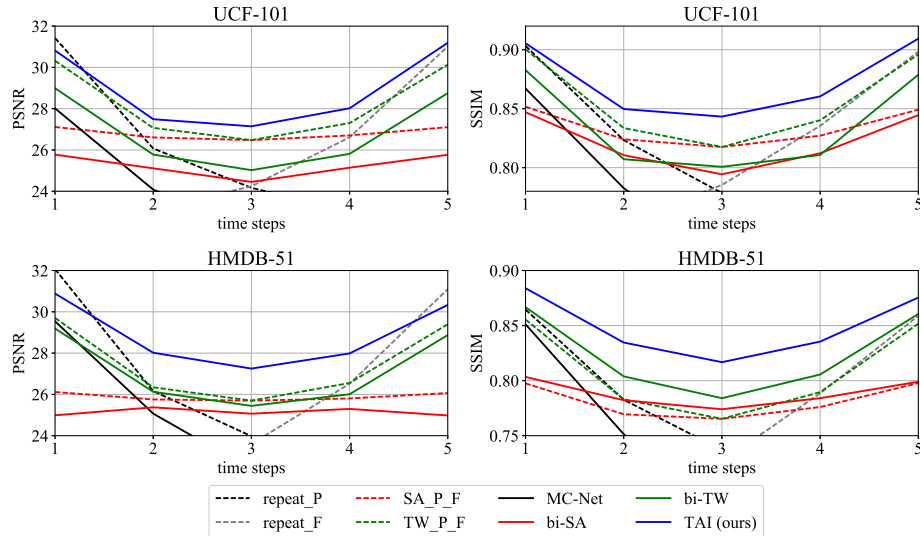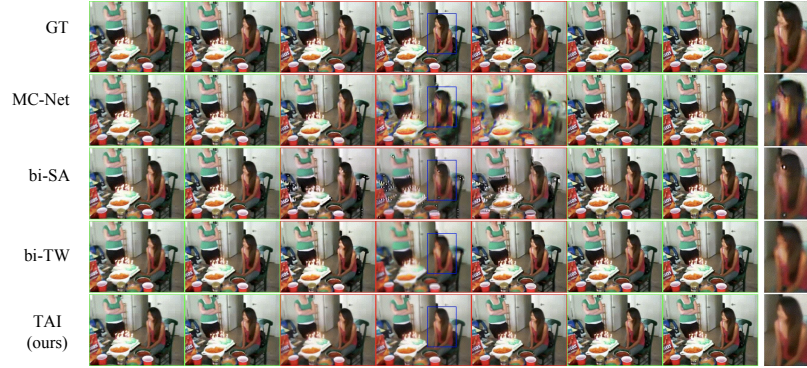
Fig. 8: Quantitative results on the UCF-101 and HMDB-51 datasets (higher PSNR and SSIM is better). We compare our method to the baselines described in Sec. 4.2
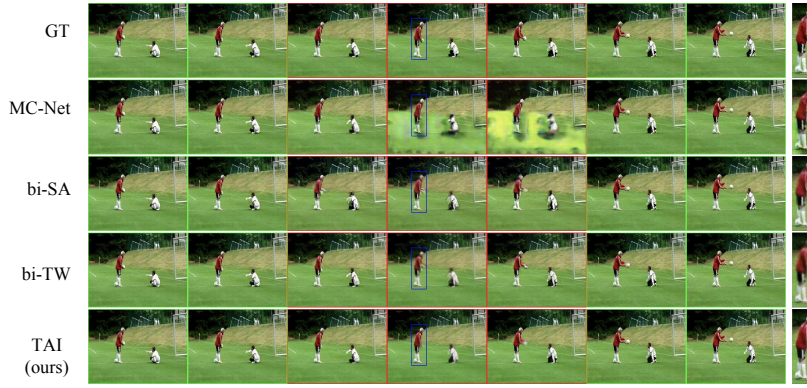
We present qualitative comparisons in Fig. 9. In Fig. 9a, we observe two contours of the girl's hair in the bi-SA prediction, and a blurry facial expression in the bi-TW prediction. On the other hand, our TAI model generates a unified contour of the hair and a clear facial expression. Moving on to Fig. 9b, we note that MC-Net distorts the background in the later middle frames, and that both bi-SA and bi-TW generate blurry patterns on the man's jacket and pants. However, TAI produces a clear white stripe on the man's pants, as well as a sharp outline around his jacket. Our results demonstrate that on video datasets containing complex scenes and a large number of action classes, TAI generates predictions that are more visually satisfying than several strong baselines.

5   Conclusion

In this paper, we have tackled the video frame inpainting problem by generating two sets of intermediate predictions conditioned on the preceding and following frames respectively, and then blending them together with our novel TAI network. Our experiments on KTH Actions, HMDB-51, and UCF-101 show that our method generates more accurate and visually pleasing predictions than multiple strong baselines. Furthermore, our in-depth analysis has revealed that our TAI network successfully leverages time step information to reconcile inconsistencies in the intermediate predictions, and that it leverages the full context provided by the preceding and following frames. In future work, we aim to improve performance by exploiting semantic knowledge about the video content,

(a) UCF-101

(b) HMDB-51

Fig. 9: Comparison of predictions from our approach to baseline methods on the UCF-101 and HMDB-51 datasets. We visualize every second frame of the input and predicted sequences. Refer to the supplementary materials for more results

e.g. by modeling human poses or the periodicity of certain actions. We also aim to explore models that can predict an even greater number of frames, i.e. several seconds of video instead of fractions of a second. To encourage innovations in deep learning for video frame inpainting, we have made our code publicly available at `https://github.com/sunxm2357/TAI_video_frame_inpainting`.

6   Acknowledgements

A Temporally-Aware Interpolation Network for Video Frame Inpainting: Supplementary Materials

A   Qualitative Results on KTH Actions

In this section, we give six more qualitative results on the KTH Actions dataset, one for each action class. To save space, we visualize every other frame. Our full model TAI gives more visually pleasing predictions on all six classes than MC-Net, bi-SA and bi-TW.

Fig. 10: Comparison on a "boxing" video from KTH Actions

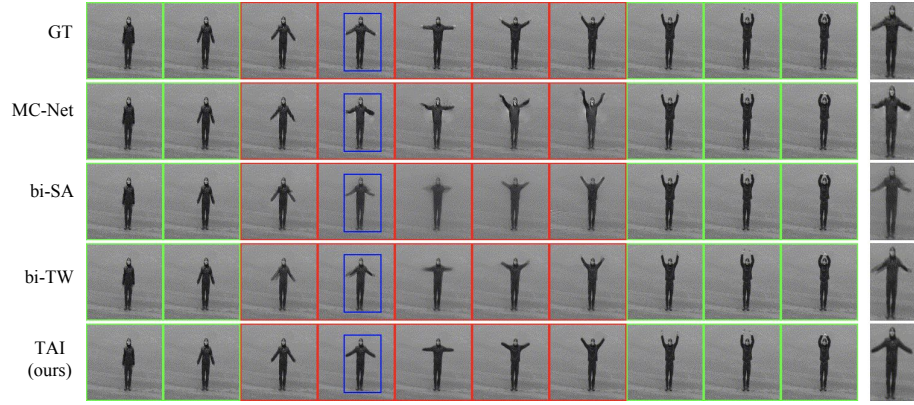Fig. 11: Comparison on a "handclapping" video from KTH Actions

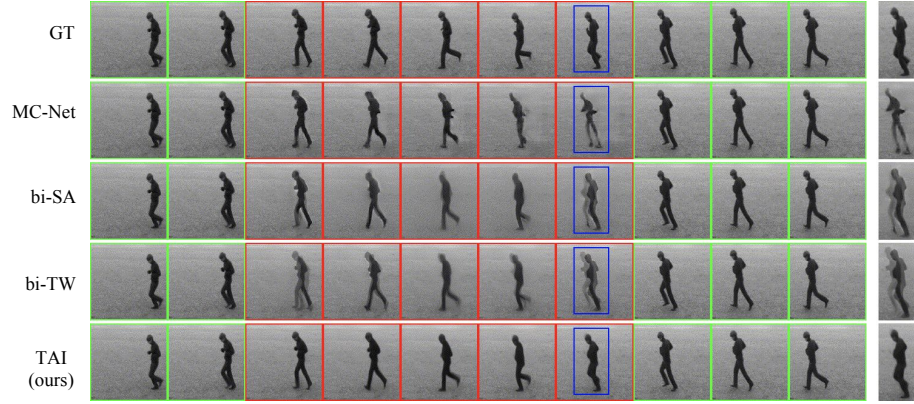Fig. 12: Comparison on a "handwaving" video from KTH Actions

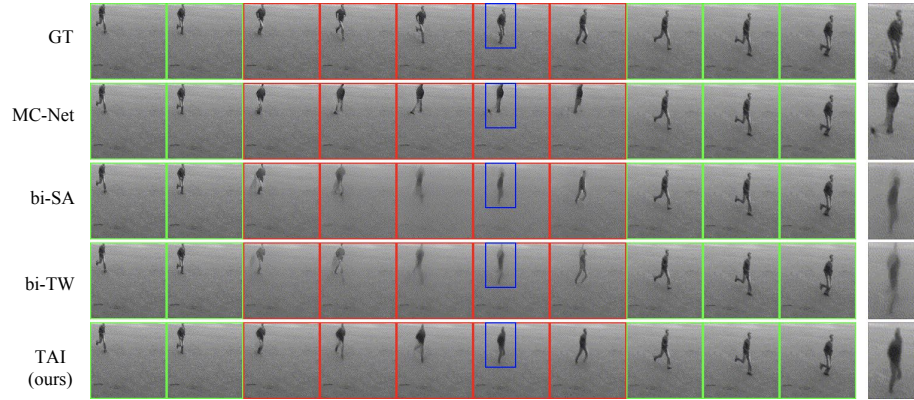Fig. 13: Comparison on a "jogging" video from KTH Actions

Fig. 14: Comparison on a "running" video from KTH Actions

Fig. 15: Comparison on a "walking" video from KTH Actions

B   Qualitative Results on UCF-101 and HMDB-51

In this section, we give six more qualitative results on different action classes, three from UCF-101 and three from HMDB-51. To save space, we visualize every other frame. Our full model TAI reconciles the misalignment of the forward and backward predictions and gives a crisp prediction.
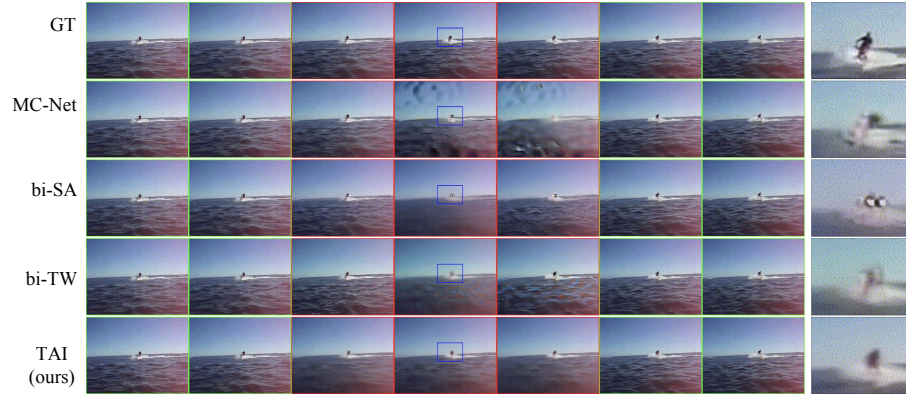

Fig. 16: Comparison on a "Skijet" video from UCF-101

Fig. 17: Comparison on a "RopeClimbing" video from UCF-101

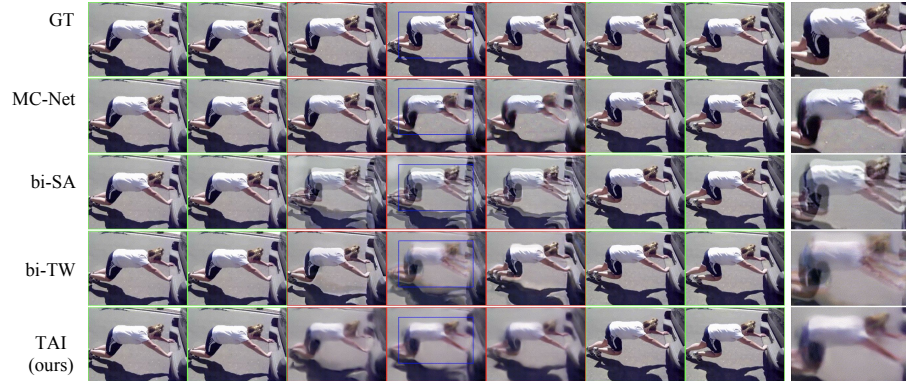Fig. 18: Comparison on an "Archery" video from UCF-101

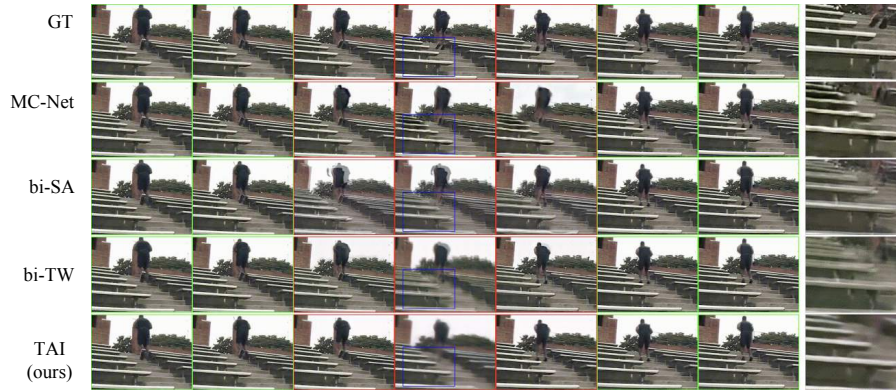Fig. 19: Comparison on a "push" video from HMDB-51

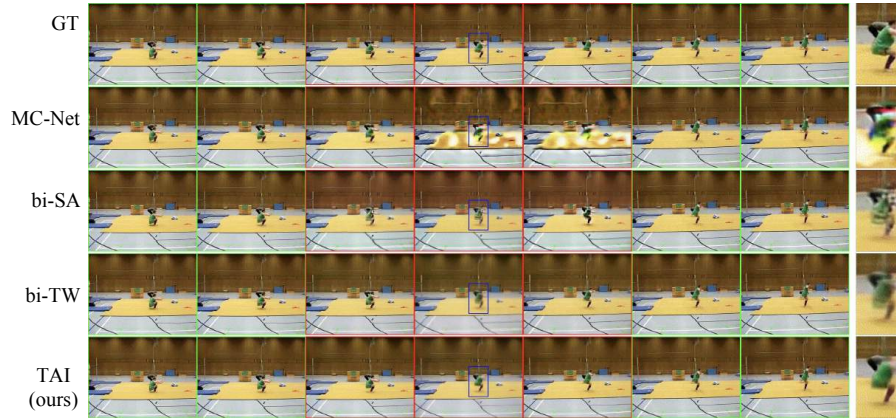Fig. 20: Comparison on a "jump" video from HMDB-51

Fig. 21: Comparison on a "somersault" video from HMDB-51

C   Importance of Context Frames: Qualitative Analysis

In addition to the quantitative results in the main paper, we provide the qualitative results on the KTH Actions dataset to demonstrate that our TAI model can leverage the context information from the preceding and following frames to better predict the middle frames.

Comparisons in Fig. 22 show how the final prediction changes when our model takes in a varying number of preceding and following frames at test time. When our model takes in only two frames from the preceding and following sequences, the prediction of the middle sequence fails to give one unified location for the actor due to the limited amount of available context information; however, when it takes in five frames, the final prediction contains much fewer ghosting artifacts. These results demonstrate that our TAI model can leverage the context

information from multiple preceding and following frames to enhance the quality
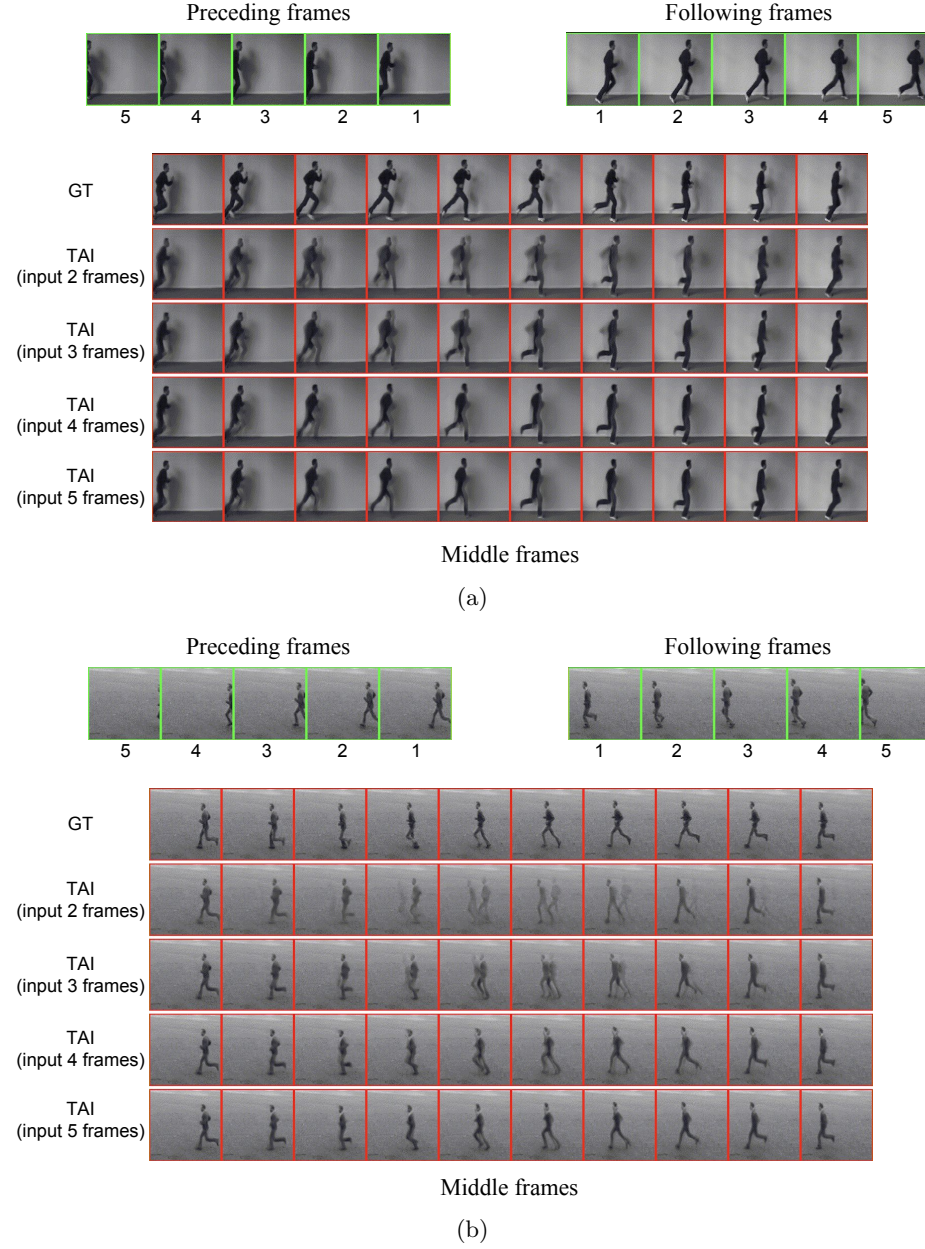of the final prediction.

(a)

(b)

Fig. 22: Qualitative results of our trained TAI model taking in two, three, four
and five preceding and following frames at test time on the KTH Actions Dataset

D   Comparison of Trained Models for Variable Number of Middle Frames

In this section, we demonstrate that our TAI model generates higher-quality predictions of the middle frames than MC-Net, bi-SA, and bi-TW across a variable number of middle frames to inpaint. To do this, we take all four models, which were all trained to predict five middle frames, and compare their performance when predicting six, seven, eight, or nine middle frames at test time on the KTH Actions dataset. From Fig. 23, we observe that our model yields higher PSNR and SSIM values than the other models when we increase the number of outputs from six to nine at the test time. This suggests that TAI incorporates the scaled time location in a way that generalizes to a variable number of middle frames, even though it has only seen scaled time locations corresponding to five middle frames.
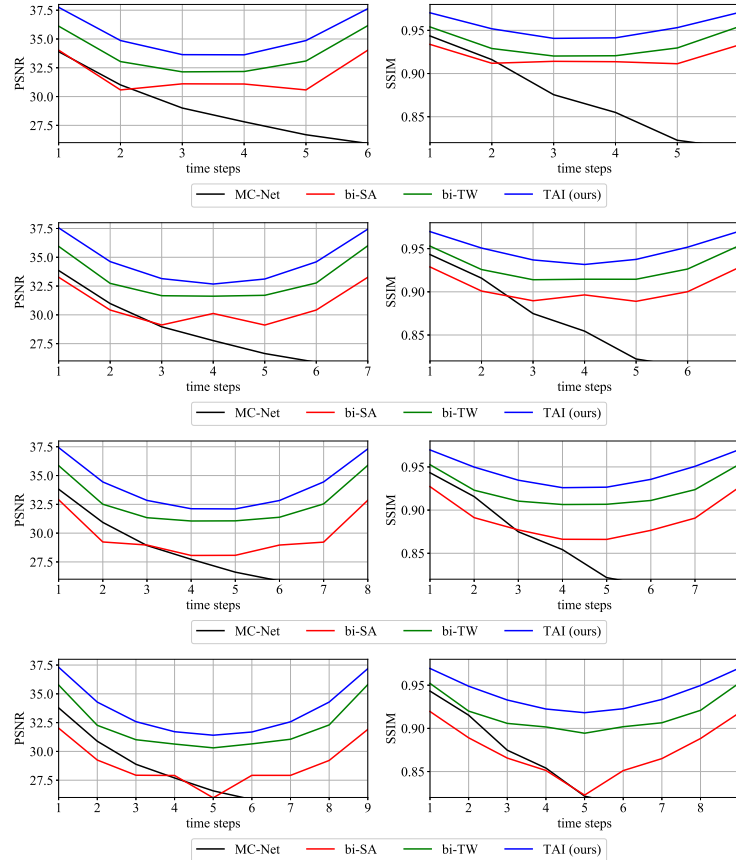

Fig. 23: Comparison of our trained model TAI with baselines when predicting 6-9 middle frames at test time on the KTH Actions dataset. Higher PSNR and SSIM is better

E   Architecture, Training, and Evaluation Details

E.1   Model Architecture

For the video prediction module, we use the same architecture as Villegas et
al. [27]. As for our TAI module, the encoder is a chain of two VGG blocks [24]
and the decoder is a chain of four VGG blocks. The kernel generation part,
ϕ_{blend}^{kg}, contains four independent sub-networks that each predict one 1D kernel.
The details of each VGG block are described in Table 1. The inputs of decoder
2 and decoder 1 are the element-wise sum of: (i) the output of the previous
decoder layer, and (ii) the residual activation from the video prediction module
with matching resolution.

Table 1: Architecture of TAI network

block name	type	kernel size	input channels	output channels	activation
encoder 1	conv	3x3	1024	256	RELU
	conv	3x3	256	256	RELU
	conv	3x3	256	256	RELU
			max pooling		
encoder 2	conv	3x3	256	512	RELU
	conv	3x3	512	512	RELU
	conv	3x3	512	512	RELU
			max pooling		
decoder 4	conv	3x3	512	512	RELU
	conv	3x3	512	512	RELU
	conv	3x3	512	512	RELU
			bilinear unsampling		
decoder 3	conv	3x3	512	256	RELU
	conv	3x3	256	256	RELU
	conv	3x3	256	256	RELU
			bilinear unsampling		
decoder 2	conv	3x3	256	128	RELU
	conv	3x3	128	128	RELU
	conv	3x3	128	128	RELU
			bilinear unsampling		
decoder 1	conv	3x3	128	64	RELU
	conv	3x3	64	64	RELU
	conv	3x3	64	64	RELU
			bilinear unsampling		
kernel generation x4	conv	3x3	65	64	RELU
	conv	3x3	64	64	RELU
	conv	3x3	64	51	RELU
			bilinear unsampling		

E.2   Training, Validation, and Test Set Construction

During training, we construct minibatches by first selecting clips with T frames
randomly from the videos in the training set, where $T = p + m + f$, and then

splitting each clip into p preceding, m middle, and f following frames. For KTH Actions, $p = m = f = 5$; for HMDB-51 and UCF-101, $p = f = 4$ and $m = 3$. To augment the training data, each video clip is randomly flipped horizontally or time-reversed. To validate a model, we take the first T frames from each video in the validation set and split the clips into p preceding, m middle, and f following frames.

We construct the test set differently for each dataset. For KTH Actions, we extract all clips from a sliding window of size $T' = p + m' + f$ and stride s across all test videos, and then split each clip into p preceding, m' middle, and f following frames. In our experiments, $p = f = 5$, $m' = 10$, 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. [27]). For HMDB-51 and UCF-101, we only evaluate each model on the first T' frames of each video in the test set, where $T' = p + m' + f$, $p = f = 4$, and $m' = 5$. We do not evaluate on all possible clips due to the large number of test videos in these datasets.

E.3   Training Hyperparameters

We train the bi-SA, bi-TW, TWI and TAI models for 100,000 iterations with batch size 4. We train MC-Net for 200,000 iterations with the same batch size because it effectively receives less data per minibatch than the other models (it does not explicitly receive the following frames, unlike the other models). We use the Adam optimizer [11] with initial learning rate $\alpha = 10^{-4}$, 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 0.002. We assume our discriminator can be represented as a 3-Lipschitz continuous function; thus, we apply spectral normalization [17] with a Lipschitz constant of 3. We use Xavier initialization [5] for each convolutional layer and uniform initialization for each linear layer (with mean 0 and variance 0.0001 for the weights). The bias of each layer is initialized with constant 0s.

References

1. Borzi, A., Ito, K., Kunisch, K.: Optimal Control Formulation For Determining Optical Flow. SIAM Journal On Scientific Computing **24**(3), 818–847 (2003)
2. Chen, K., Lorenz, D.A.: Image Sequence Interpolation Using Optimal Control. Journal of Mathematical Imaging and Vision **41**(3), 222–238 (2011)
3. Cheung, V., Frey, B.J., Jojic, N.: Video Epitomes. International Journal of Computer Vision **76**(2), 141–152 (2008)
4. Ebdelli, M., Le Meur, O., Guillemot, C.: Video Inpainting With Short-term Windows: Application To Object Removal And Error Concealment. IEEE Transactions on Image Processing **24**(10), 3034–3047 (2015)
5. Glorot, X., Bengio, Y.: Understanding the difficulty of training deep feedforward neural networks. In: Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics. pp. 249–256 (2010)
6. Granados, M., Kim, K.I., Tompkin, J., Kautz, J., Theobalt, C.: Background Inpainting For Videos With Dynamic Objects And A Free-Moving Camera. In: European Conference on Computer Vision. pp. 682–695 (2012)
7. Ioffe, S., Szegedy, C.: Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. In: International Conference On Machine Learning. pp. 448–456 (2015)
8. Jia, J., Tai-Pang, W., Tai, Y.W., Tang, C.K.: Video Repairing: Inference Of Foreground And Background Under Severe Occlusion. In: IEEE Conference on Computer Vision and Pattern Recognition (2004)
9. Jia, Y.T., Hu, S.M., Martin, R.R.: Video Completion Using Tracking And Fragment Merging. The Visual Computer **21**(8-10), 601–610 (2005)
10. Kalchbrenner, N., Oord, A.v.d., Simonyan, K., Danihelka, I., Vinyals, O., Graves, A., Kavukcuoglu, K.: Video Pixel Networks. In: International Conference On Machine Learning (2017)
11. Kingma, D.P., Ba, J.L.: ADAM: A Method For Stochastic Optimization. In: International Conference on Learning Representations (2015)
12. Kuehne, H., Jhuang, H., Garrote, E., Poggio, T., Serre, T.: HMDB: A Large Video Database For Human Motion Recognition. In: IEEE International Conference on Computer Vision. pp. 2556–2563 (2011)
13. Liu, Z., Yeh, R., Tang, X., Liu, Y., Agarwala, A.: Video Frame Synthesis Using Deep Voxel Flow. In: International Conference on Computer Vision (ICCV). vol. 2 (2017)
14. Long, G., Kneip, L., Alvarez, J.M., Li, H., Zhang, X., Yu, Q.: Learning Image Matching By Simply Watching Video. In: European Conference on Computer Vision. pp. 434–450 (2016)
15. Lotter, W., Kreiman, G., Cox, D.: Deep Predictive Coding Networks for Video Prediction and Unsupervised Learning. International Conference on Learning Representations (2017)
16. Mathieu, M., Couprie, C., LeCun, Y.: Deep Multi-Scale Video Prediction Beyond Mean Square Error. International Conference on Learning Representations (2016)
17. Miyato, T., Kataoka, T., Koyama, M., Yoshida, Y.: Spectral Normalization for Generative Adversarial Networks. In: International Conference on Learning Representations (2018)
18. Newson, A., Almansa, A., Fradet, M., Gousseau, Y., Pérez, P.: Video Inpainting Of Complex Scenes. SIAM Journal on Imaging Sciences **7**(4), 1993–2019 (2014)

19. Niklaus, S., Mai, L., Liu, F.: Video Frame Interpolation via Adaptive Separable Convolution. In: IEEE Conference on Computer Vision and Pattern Recognition. pp. 261–270 (2017)
20. Patwardhan, K.A., Sapiro, G., Bertalmío, M.: Video Inpainting Under Constrained Camera Motion. IEEE Transactions on Image Processing **16**(2), 545–553 (2007)
21. Ranzato, M., Szlam, A., Bruna, J., Mathieu, M., Collobert, R., Chopra, S.: Video (Language) Modeling: A Baseline For Generative Models Of Natural Videos. arXiv preprint arXiv:1412.6604 (2014)
22. Schuldt, C., Laptev, I., Caputo, B.: Recognizing Human Actions: A Local Svm Approach. In: International Conference on Pattern Recognition. vol. 3, pp. 32–36 (2004)
23. Shen, Y., Lu, F., Cao, X., Foroosh, H.: Video Completion For Perspective Camera Under Constrained Motion. In: International Conference on Pattern Recognition. vol. 3, pp. 63–66 (2006)
24. Simonyan, K., Zisserman, A.: Very Deep Convolutional Networks for Large-Scale Image Recognition. International Conference on Learning Representations (2015)
25. Soomro, K., Zamir, A.R., Shah, M.: UCF101: A Dataset Of 101 Human Actions Classes From Videos In The Wild. CRCV-TR-12-01 (2012)
26. Srivastava, N., Mansimov, E., Salakhudinov, R.: Unsupervised Learning Of Video Representations Using LSTMs. In: International Conference On Machine Learning. pp. 843–852 (2015)
27. Villegas, R., Yang, J., Hong, S., Lin, X., Lee, H.: Decomposing Motion And Content For Natural Video Sequence Prediction. International Conference on Learning Representations (2017)
28. Wang, Z., Bovik, A.C., Sheikh, H.R., Simoncelli, E.P.: Image Quality Assessment: From Error Visibility To Structural Similarity. IEEE Transactions on Image Processing **13**(4), 600–612 (2004)
29. Werlberger, M., Pock, T., Unger, M., Bischof, H.: Optical flow guided TV-L1 video interpolation and restoration. In: International Workshop on Energy Minimization Methods in Computer Vision and Pattern Recognition. pp. 273–286 (2011)
30. Wexler, Y., Shechtman, E., Irani, M.: Space-Time Video Completion. In: IEEE Conference on Computer Vision and Pattern Recognition (2004)