HyperCon: Image-To-Video Model Transfer for Video-To-Video Translation Tasks

Ryan Szeto[†] Mostafa El-Khamy[§] Jungwon Lee[§] Jason J. Corso[†]
[†]University of Michigan [§]Samsung Semiconductor, Inc.
{szetor,jjcorso}@umich.edu, {mostafa.e,jungwon2.lee}@samsung.com

Abstract

Video-to-video translation is more difficult than image-to-image translation due to the temporal consistency problem that, if unaddressed, leads to distracting flickering effects. Although video models designed from scratch produce temporally consistent results, training them to match the vast visual knowledge captured by image models requires an intractable number of videos. To combine the benefits of image and video models, we propose an image-to-video model transfer method called Hyperconsistency (HyperCon) that transforms any well-trained image model into a temporally consistent video model without fine-tuning. HyperCon works by translating a temporally interpolated video frame-wise and then aggregating over temporally localized windows on the interpolated video. It handles both masked and unmasked inputs, enabling support for even more video-to-video translation tasks than prior image-to-video model transfer techniques. We demonstrate HyperCon on video style transfer and inpainting, where it performs favorably compared to prior state-of-the-art methods without training on a single stylized or incomplete video. Our project website is available at `ryanszeto.com/projects/hypercon`.

1. Introduction

Developments in both large-scale datasets and deep neural networks (DNNs) have led to incredible advancements in image-to-image [15, 35] and video-to-video [31, 1, 24, 6] translation tasks such as color restoration [39, 41], super-resolution [7, 8], inpainting [20, 37, 14], and style transfer [9, 17]. But compared to images, videos pose an additional challenge: not only do the video frames need to satisfy the intended translation, they must also be temporally consistent. Otherwise, they will exhibit flickering artifacts.

Existing video-to-video translation techniques address temporal consistency through one of two types of strategies. The first incorporates temporal consistency directly into the method through optical flow-based losses [18, 27, 28, 12, 38] or network layers that operate on the time dimension, *e.g.*, 3D convolutional layers [5] or recurrent layers [18]. These techniques leverage self-supervised video data and tailor models to specific tasks using relevant losses, *e.g.*, reconstruction error [20, 37, 18] or style loss [17, 12]. However, they require models and losses that are defined exclusively on videos and tuned for a specific application.

The other type of strategy uses a *blind video consistency* model to reduce flicker in a frame-wise translated video as a post-processing step [2, 8, 19, 36]. For example, Lai *et al*. [19] train a recurrent encoder-decoder network to improve the temporal consistency of independently processed frames via warping and structural preservation losses. Blind video consistency methods relax the need for task-specific video models and losses, and also enables image-to-image models to be applied immediately to videos without sacrificing consistency. However, they require dense correspondences between unprocessed frames during optimization, making them unsuitable for tasks in which certain input regions have no meaningful structure, *e.g.*, video inpainting.

Similarly to blind video consistency methods, we opt to impart image-to-image models with temporal consistency, motivated primarily by generalization issues inherent to video-tailored approaches. To elaborate, consider in Figure 1 a failure case from the otherwise impressive state-of-the-art video inpainting network VINet [18]; here, it fails to hallucinate a sensible texture for the missing region. Now consider a state-of-the-art *image* inpainting model, Contextual Attention [37]: this method produces realistic textures on the same example (but exhibits temporal inconsistency). The reason for this difference lies in the diversity of data used to train each model. Whereas the video model was trained on about 5,000 examples from one of the largest video segmentation datasets to date [33], the image model was trained on over 1,000,000 images [42]. Regardless of application, the vast scale of image datasets enables image models to encapsulate broader visual knowledge than video models trained from scratch and, as a result, better generalize to new data. Because storage and cost limitations make it intractable to collect high-quality video datasets as

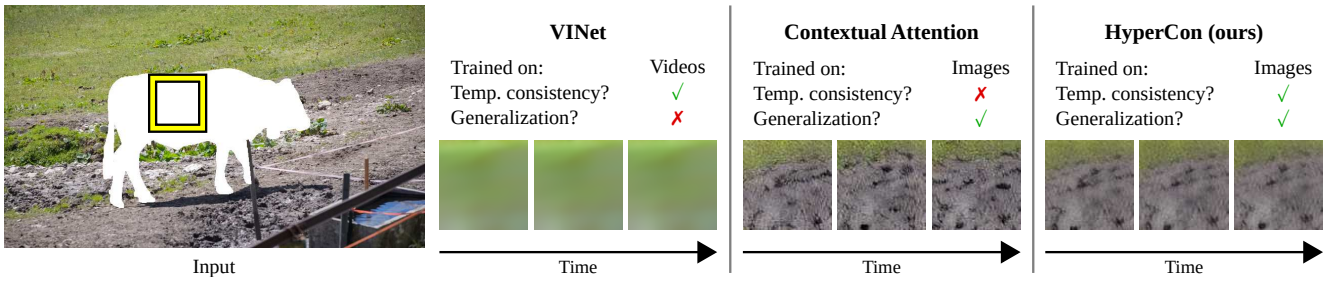


Figure 1: Video-to-video translation models designed and trained from scratch, *e.g.*, VINet [18] are temporally consistent, but exhibit poor generalization performance due to the limited size of high-fidelity video datasets (note the lack of defined texture). Image-to-image models, *e.g.*, Contextual Attention [37], generalize well thanks to large image datasets, but lack temporal consistency (note the changing texture). HyperCon leverages the generalization performance conferred by image datasets while enforcing the temporal consistency properties of video-to-video models.

diverse as modern image datasets, video-tailored models are doomed to generalize poorly compared to image models.

To overcome this challenging generalization issue, we propose a method of *image-to-video model transfer for video-to-video translation tasks*. Rather than optimize an image or video model from scratch, we aim to transform a black-box image-to-image translation model into a strong video-to-video translation model without fine-tuning—specifically, to automatically induce temporal consistency while achieving the same visual effect as the image-to-image model. Compared to prior techniques in blind video consistency [2, 19], which target these goals and therefore fall under the same problem space, ours broadens the scope of applicable tasks to include those in which the input and output videos have differing visual structure, such as video inpainting and edge-deforming style transfer (e.g., the *mosaic* style from PyTorch's examples repository).

Key to our approach is the view of image-to-image models as noisy models in which small changes in the input lead to substantial changes in the output (*e.g.*, Figure 1, middle). To strengthen the desired signal and filter out noise, we synthesize several perturbed, but related predictions and aggregate over them. We obtain perturbations in a way that conditions on multiple neighboring frames to enhance temporal consistency. Our *hyperconsistency* approach (*HyperCon* for short) implements these principles by inserting frames into the video with a frame interpolation network, translating the interpolated video's frames independently, and aggregating within overlapping windows of appropriate stride to obtain a final video whose length matches the original (Figure 2).

As the first method among image-to-video model transfer techniques to reason in interpolated video space, HyperCon forgoes the traditional post-processing paradigm by integrating frame-wise translation *within* itself as an intermediate step, not as a step that precedes it. Since it does not require dense correspondences in the unprocessed input video, it performs well on video-to-video translation tasks with or without masked inputs—*e.g.*, inpainting and style transfer respectively—as verified by our extensive experiments across these two widely differing applications. HyperCon outperforms a prior state-of-the-art video consistency model [19] in terms of reducing flicker and adhering to the intended translation; when combined with a strong image inpainting method, it also produces better predictions than a state-of-the-art video inpainting model [18]. It achieves competitive performance in both tasks *despite not being trained with any masked or stylized videos*.

Our contributions are as follows. First, we motivate image-to-video model transfer as a way to leverage the superior generalization performance of image models for video-to-video translation without sacrificing temporal consistency. Second, we propose HyperCon, which supports a wider span of tasks than prior video consistency work thanks to its support for both masked *and* unmasked inputs. Finally, we show that HyperCon performs favorably compared to state-of-the-art video consistency and inpainting methods without the need to be fine-tuned on these tasks. Our project website is available at `ryanszeto.com/projects/hypercon`.

2. HyperCon

Given an input video $V = \{v_1, \ldots, v_N\}$, our goal is to generate an output video $O = \{o_1, \ldots, o_N\}$ representing the N frames of V translated by some image-to-image model g. The frames of O should closely resemble the frames of V translated frame-wise by g; at the same time, O should be temporally consistent, *i.e.*, exhibit as few flickering effects as possible. We quantify these notions of resemblance and consistency concretely in Sections 3.2 and 4.2.

With HyperCon (Figure 2), we generate O as follows. First, we artificially insert i frames between each pair of frames in V with a frame interpolation network (Section 2.1). Denoting this interpolated version of V as $V^s = \{v_1^s, \ldots, v_{N'}^s\}$, where N' is the number of frames in the in-

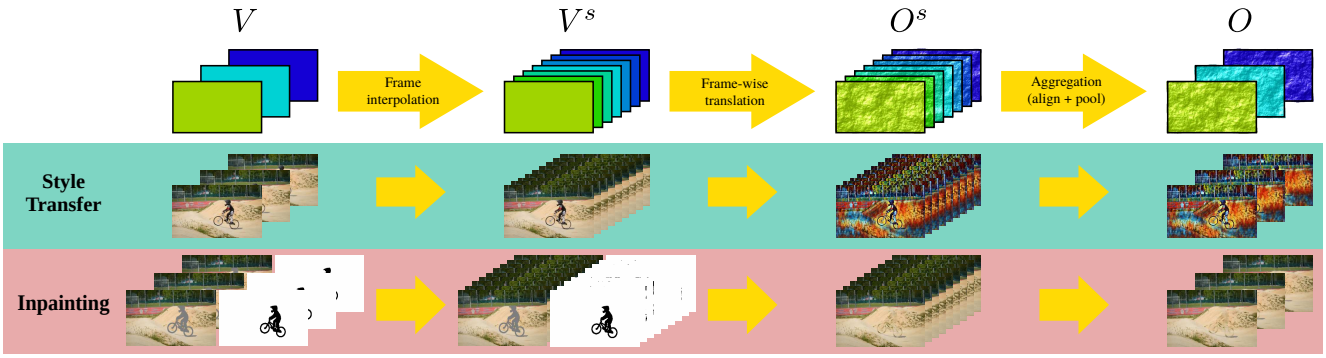


Figure 2: Visual overview of our Hyperconsistency (HyperCon) method. We begin by artificially inserting frames into the input video V with a frame interpolation network to produce an interpolated video V^s. Then, we independently translate each frame in the interpolated video with an image-to-image translation model. Finally, we aggregate frames (*i.e.*, align with optical flow and pool pixel-wise) within a local sliding window to produce the final temporally consistent output video O. This can be applied to tasks with or without masked inputs (*e.g.*, inpainting and style transfer, respectively).

terpolated video, we then independently translate frames in V^s with g, yielding $O^s = \{o^s_1, \ldots, o^s_{N'}\}$ (Section 2.2). Finally, we aggregate frames in O^s over a temporal sliding window with appropriate stride to produce the frames of the final output video O (Section 2.3). Additional considerations for masked inputs are discussed in Section 2.4.

2.1. Generating the Interpolated Video

To generate the interpolated video V^s, we insert i interpolated frames between each pair of frames in V, which essentially allows us to obtain several perturbed versions of each input frame for translation. We opt for a vector-based sampling method for frame interpolation instead of a kernel-based one[1] which, as we justify in Section 2.4, allows us to handle the case of masked inputs appropriately. Specifically, for each pair of consecutive frames v_a and v_{a+1} ($a \in 1, \ldots, N-1$) and intermediate frame index $b \in \{1, \ldots, i\}$, we predict two warping grids $(F^s_{a+b'\to a}, F^s_{a+b'\to a+1})$ and a weight map $w_{a+b'}$ (where $b' \equiv \frac{b+1}{i+1}$) with some function wrpgrd (*e.g.*, a pre-trained DNN), and use them to generate the corresponding interpolated frame v^s_j ($j \in \{1, \ldots, N'\}$):

$$(F^s_{a+b'\to a}, F^s_{a+b'\to a+1}, w_{a+b'}) = \text{wrpgrd}(v_a, v_{a+1}, b')\,, \tag{1}$$

$$\begin{aligned} v^s_j = (1 - w_{a+b'}) \odot \text{warp}(v_a, F^s_{a+b'\to a}) \\ + w_{a+b'} \odot \text{warp}(v_{a+1}, F^s_{a+b'\to a+1})\,. \end{aligned} \tag{2}$$

$\odot$ is an element-wise product; $\text{warp}(v, F)$ bilinearly samples from v via displacements specified by vector field F.

2.2. Translating the Interpolated Video

At this point, we have computed the interpolated video V^s. We generate the translated interpolated video O^s by simply translating each frame in V^s independently:

$$o^s_j = g(v^s_j), \quad j \in \{1, \ldots, N'\}\,. \tag{3}$$

Clearly, O^s is not temporally consistent. However, we expect that most spatial regions in this video will exhibit consensus within small temporal windows. For example, a patch might have a distinct color profile in one frame, but a common color profile in the other frames in the local temporal window. Since we have more frames in O^s than frames needed in the output, we can remove the spurious artifacts of frame-wise translation by mapping several neighboring frames in O^s to one frame in our desired output video O. We call this mapping *temporal aggregation* (Section 2.3).

2.3. Temporal Aggregation

We perform temporal aggregation over a sliding window on the translated interpolated video O^s (Figure 3). The stride is such that the frame in each window's center, *i.e.*, the reference frame, corresponds to a frame from the (non-interpolated) input video V, resulting in N windows. Within each window, we align the off-center frames, *i.e.*, the context frames, to the reference frame via optical flow warping, and then pool the reference and aligned context frames pixel-wise (*e.g.*, with a mean or median filter) to produce a final frame in the output video O. Note that we compute the flow between *interpolated, translated* frames O^s, *not* the unprocessed input frames V like prior work [2].

More precisely, for an interpolated frame index $j \in \{1, 1+(i+1), \ldots, N'-(i+1), N'\}$, we first estimate the optical flow between reference frame o_j and each context frame in $\{o^s_{j-d\gamma}, o^s_{j-d(\gamma-1)}, \ldots, o^s_{j-d}, o^s_{j+d}, \ldots, o^s_{j+d(\gamma-1)}, o^s_{j+d\gamma}\}$ (denoted $F^a_{j\to(*)}$), where γ and d respectively parameterize the number of frames in the sliding window and a temporal dilation factor. We then warp the context frames to align them to o^s_j, and afterwards perform pixel-wise pooling over

[1] This distinction is discussed in more detail in Reda et al. [26].

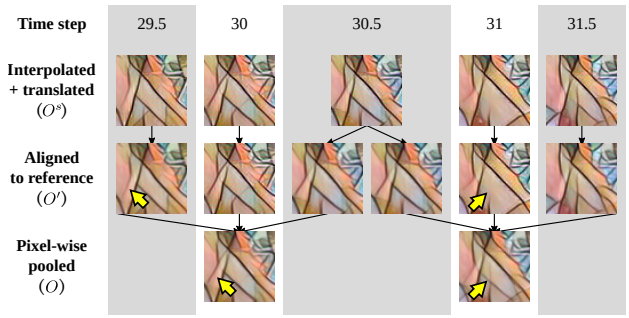


Figure 3: Temporal aggregation. Context frames from the translated interpolated video O^s are aligned via optical flow to reference frames associated with integer time steps (white columns) and then pooled at each pixel location to generate the final video O. Despite inconsistencies between aligned frames (*e.g.*, near the arrows), temporal aggregation selects stable components by majority vote.

o^s_j and the warped context frames:

$$o'_{j,k} = \begin{cases} o^s_j & k = 0 \\ \text{warp}(o^s_{j+dk}, F^a_{j \to j+dk}) & k \neq 0 \end{cases}, k \in \{-\gamma, \ldots, \gamma\}\,, \tag{4}$$

$$o_j = \text{pool}\big(o'_{j,k} \mid k \in \{-\gamma, \ldots, \gamma\}\big)\,. \tag{5}$$

Pooling is applied over values per spatial location, color channel, and time step; *i.e.*, if $P = \text{pool}(I_1, I_2, \ldots)$, then

$$P(l_h, l_w, l_c) = f\big(I_1(l_h, l_w, l_c), I_2(l_h, l_w, l_c), \ldots\big)\,, \tag{6}$$

where $P(l_h, l_w, l_c)$ and $I(l_h, l_w, l_c)$ denote the value of 3D image tensors P and I at location (l_h, l_w, l_c), and f is a mean or median operation. In the cases where the sliding window samples outside the valid frame range, we only align and pool over valid frames.

To illustrate why HyperCon induces temporal consistency, we visualize intermediate outputs for style transfer in Figure 3. Even among interpolated frames with similar appearances, flickering artifacts can occur. By selecting pixel values by a majority vote over several interpolated frames, our method automatically incorporates stable components into the final prediction, thereby reducing flicker.

2.4. HyperCon for Masked Videos

Having described HyperCon in the unmasked input case in Sections 2.1-2.3, we now extend it to handle tasks in which the input frames have masked pixels (*e.g.*, inpainting). This case differs from the unmasked input case in three ways. First, in addition to the normal RGB video V, we now have as input a mask video $M = \{m_1, \ldots, m_N\}$ in which 1 marks an unmasked pixel and 0 marks a masked pixel. Second, when generating the interpolated data, we must create an interpolated mask video $M^s = \{m^s_1, \ldots, m^s_{N'}\}$ to accompany the interpolated RGB video V^s. Finally, the image-to-image translation model g now takes a mask as input in addition to an RGB video frame.

We modify the interpolated video generation step (Section 2.1) to produce both V^s and M^s; this is done by generating i interpolated frames between each pair of frames in V and M. For this to be valid, the motion of the interpolated mask video must match that of the interpolated RGB video—for example, if we interpolate the motion of a removed person, the mask must cover that person throughout the interpolated sequence. If this is not handled properly, we risk polluting the final result with mask placeholder values. Thus, we opt for a vector-based sampling method for frame interpolation instead of a kernel-based one, since the same warping grid can be applied to both RGB and mask frames to achieve the desired result.

To generate V^s, recall that we predict warping grids and a weight map $(F^s_{a+b' \to a}, F^s_{a+b' \to a+1}, w_{a+b'})$ from frames in V using Equation 1. To obtain the interpolated masks M^s, we apply these parameters to the masks in M and follow up with a thresholding operation:

$$\begin{aligned} \dot{m}^s_j = (1 - w_{a+b'}) \odot \text{warp}(m_a, F^s_{a+b' \to a}) \\ + w_{a+b'} \odot \text{warp}(m_{a+1}, F^s_{a+b' \to a+1})\,, \end{aligned} \tag{7}$$

$$m^s_j = \text{thresh}(\dot{m}^s_j, 1)\,. \tag{8}$$

Warping the masks in this way allows us to detect the "partially-masked" pixels in v^s_j, *i.e.*, the ones that received a contribution from a masked pixel in either v_a or v_{a+1}. Specifically, if a pixel in $\dot{m}^s_j$ is not 1, then the warping operation used a source value of 0 from m_a or m_{a+1}, which corresponds to borrowing from a masked pixel. Thus, thresholding turns partially-masked pixels into fully-masked pixels in the interpolated masks so that the subsequent translation step is not incorrectly influenced by these pixels.

At this point, we have generated the interpolated RGB and mask videos V^s and M^s. We apply the image-to-image model g to them:

$$o^s_j = g(v^s_j, m^s_j), \quad j \in 1, \ldots, N'\,, \tag{9}$$

and then apply temporal aggregation (Section 2.3) as usual.

2.5. HyperCon Implementation Details

For the frame interpolation step, we use Super SloMo [16] as our wrpgrd function to predict warping grids and weight masks. This method is well-suited for our approach since it predicts warping parameters for multiple intermediate time steps, contrasting with kernel-based sampling methods that interpolate one frame [22]. To estimate optical flow in the temporal aggregation step, we use a third-party implementation of PWC-Net [29]. Since HyperCon is agnostic to the specific instantiations of frame interpolation and flow estimation, we have chosen state-of-the-art

models to demonstrate the full potential of our overall approach. Although we do not fine-tune network weights, we observe good performance regardless and postulate that performance can be further improved by fine-tuning.

3. Experiments: Video Style Transfer

To demonstrate HyperCon on unmasked inputs, we apply it to video style transfer. For the frame-wise style transfer subroutine, we use the Fast Style Transfer (FST) models from Johnson *et al*. [17] with the pre-trained *mosaic* and *rain-princess* weights from PyTorch's examples repository.

3.1. Datasets

For evaluation, we use the ActivityNet [3] and DAVIS [25] video datasets, which primarily consist of dynamic indoor and outdoor scenes of animals and people. We split them into validation and test sets to validate HyperCon hyperparameters and evaluate performance, respectively. For ActivityNet, we manually curate clips from the official training/validation and test splits (∼100 videos per split with ∼100 frames per video), filtering by properties of high-quality videos such as non-negligible motion, no splash screens, one continuous camera shot, *etc*. For DAVIS, we use the official training/validation set for validation, and combine the development and challenge sets from the DAVIS 2017 and 2019 challenges to produce two test sets, one for each year. We pre-process all videos by resizing and center-cropping them to 832×480 resolution, and scaling RGB values to (-1, 1).

3.2. Evaluation Metrics

For video style transfer, our goal is to transfer the desired style to all video frames while minimizing flickering effects (*i.e*., maximizing temporal consistency) in the output video as much as possible. To evaluate temporal consistency, we measure warping error E_{warp} [19] and patch-based consistency measures C_{PSNR} and C_{SSIM} [12]. E_{warp} is defined as the mean of e_{warp} over all consecutive pairs of output frames (o_a, o_{a+1}), where

$$e_{\text{warp}}(o_a, o_{a+1}) = \frac{1}{\sum_p M_a^f(p)} \sum_p M_a^f(p) \|D_a(p)\|_2^2 . \tag{10}$$

Here, $D_a = o_a - \text{warp}(o_{a+1}, F_{a\to a+1})$ (where $F_{a\to a+1}$ is the estimated flow between input frames v_a and v_{a+1}); p indexes the pixels in the frame; and M_a^f is a mask that indicates pixels with reliable flow (1 for reliable, 0 for unreliable). M_a^f is computed based on flow consistency and motion boundaries as defined by Ruder *et al*. [27]. C_{PSNR} is defined as the mean of c_{PSNR} over all consecutive pairs of output frames, where c_{PSNR} is computed by taking a random 50×50 patch in frame o_a and computing the maximum PSNR between it and all patches in its spatial neighborhood in the next frame o_{a+1} (within a Chebyshev distance of 20 pixels). C_{SSIM} is defined similarly to C_{PSNR}, except it computes Structural Similarity [32] in place of PSNR. To quantify how well a method adheres to the intended style, we measure Frechét Inception Distance (FID) [13] between the set of all frames generated by frame-wise translation and the set of all frames generated by the method under evaluation.

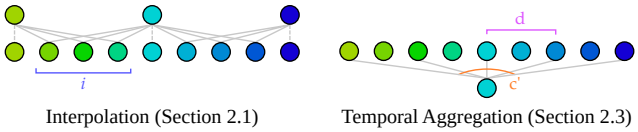


Figure 4: Visualization of the hyperparameters included in our grid search (Section 3.3).

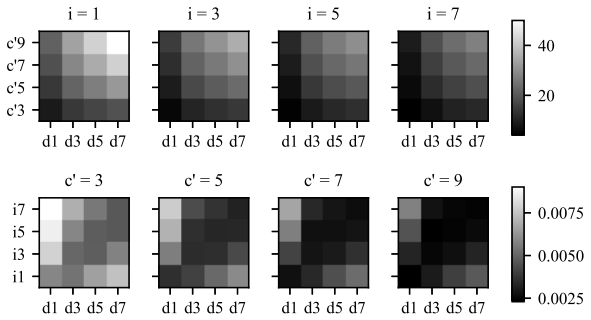


Figure 5: Hyperparameter grid search for *rain-princess* on the DAVIS train/val set. Row 1: FID (style adherence). Row 2: E_{warp} (temporal consistency). Lower is better.

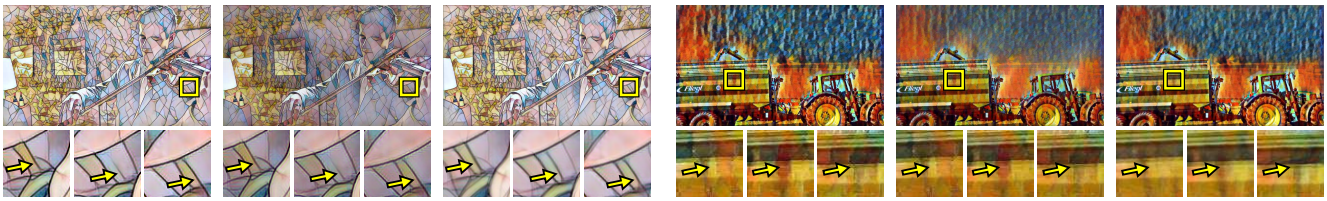
3.3. Hyperparameter Analysis

We perform a grid search over our method's hyperparameters to link them concretely to style adherence and temporal consistency. These hyperparameters consist of the number of frames to insert between each pair i, the total number of interpolated frames to aggregate for each output frame $c' = 2\gamma + 1$, and the spacing between aggregated interpolated frames d (Figure 4). We quantify each setting's performance and present their trends in Figure 5.

Lai et al. [19] observe that style adherence and temporal consistency are competing objectives; our hyperparameter grid search supports this claim. Specifically, FID decreases with more interpolated frames, fewer aggregated frames, and a smaller dilation rate, indicating that style adherence is maximized when HyperCon aggregates over few frames that are very similar to the reference frame. On the other hand, E_{warp} decreases gradually with more aggregated frames, and follows the trend of a parabolic cylinder given fixed c', *i.e*., there is a sweet spot for i given fixed d and vice-versa. The trends of E_{warp} suggest that temporal consistency is maximized when many frames are aggregated

		mosaic				rain-princess			
Dataset	Method	FID$^{\downarrow}$	$E_{\text{warp}}{}^{\downarrow}$	$C_{\text{PSNR}}{}^{\uparrow}$	$C_{\text{SSIM}}{}^{\uparrow}$	FID$^{\downarrow}$	$E_{\text{warp}}{}^{\downarrow}$	$C_{\text{PSNR}}{}^{\uparrow}$	$C_{\text{SSIM}}{}^{\uparrow}$
DAVIS 2017	FST [17]	-	0.024829 ± 0.001174	16.72 ± 1.43	0.5166 ± 0.0152	-	0.013934 ± 0.000903	19.75 ± 1.39	0.6030 ± 0.0156
	FST-vcons [19]	24.50	0.010194 ± 0.000500	20.51 ± 1.37	0.5830 ± 0.0140	10.87	0.007071 ± 0.000477	22.73 ± 1.35	0.6717 ± 0.0136
	HyperCon (ours)	**18.04**	$\mathbf{0.008810 \pm 0.000493}$	$\mathbf{21.04 \pm 1.37}$	$\mathbf{0.6662 \pm 0.0157}$	**10.53**	$\mathbf{0.006110 \pm 0.000487}$	$\mathbf{23.38 \pm 1.35}$	$\mathbf{0.7480 \pm 0.0143}$
DAVIS 2019	FST [17]	-	0.029379 ± 0.001684	14.88 ± 0.29	0.4737 ± 0.0184	-	0.017614 ± 0.001309	17.61 ± 0.38	0.5550 ± 0.0189
	FST-vcons [19]	24.89	0.011999 ± 0.000672	18.79 ± 0.29	0.5451 ± 0.0169	14.77	0.008938 ± 0.000644	20.86 ± 0.38	0.6365 ± 0.0159
	HyperCon (ours)	**23.11**	$\mathbf{0.010677 \pm 0.000710}$	$\mathbf{19.20 \pm 0.32}$	$\mathbf{0.6105 \pm 0.0192}$	**13.59**	$\mathbf{0.008117 \pm 0.000730}$	$\mathbf{21.13 \pm 0.43}$	$\mathbf{0.6960 \pm 0.0175}$
ActivityNet	FST [17]	-	0.017895 ± 0.000847	19.29 ± 0.81	0.6478 ± 0.0134	-	0.008428 ± 0.000516	23.45 ± 0.81	0.7469 ± 0.0116
	FST-vcons [19]	26.37	0.006727 ± 0.000311	22.58 ± 0.38	0.7075 ± 0.0115	9.07	0.003923 ± 0.000240	25.97 ± 0.42	0.8008 ± 0.0093
	HyperCon (ours)	**10.09**	$\mathbf{0.005946 \pm 0.000298}$	$\mathbf{23.61 \pm 0.77}$	$\mathbf{0.7848 \pm 0.0102}$	**5.50**	$\mathbf{0.003379 \pm 0.000233}$	$\mathbf{26.95 \pm 0.76}$	$\mathbf{0.8544 \pm 0.0083}$

Table 1: Quantitative comparison between frame-wise style transfer (FST) [17], baseline blind video consistency (FST-vcons) [19], and HyperCon (ours) for style transfer. $^{\uparrow}$ and $^{\downarrow}$ indicate where higher and lower is better; bold indicates the best score; and the values after $\pm$ are standard error over all videos. FID for FST is blank since this equates to comparing FST to itself. HyperCon obtains lower FID scores than FST-vcons, indicating greater coherence to the intended style of FST, as well as better warping errors and patch-based consistency scores, indicating better temporal consistency.



(a) FST [17] (b) FST-vcons [19] (c) HyperCon (ours) (d) FST [17] (e) FST-vcons [19] (f) HyperCon (ours)

Figure 6: Style transfer comparison for *mosaic* (left) and *rain-princess* (right) styles. We show one full frame and crops from three consecutive frames centered at the presented frame. Unlike the baselines, HyperCon draws three consistent lines across the top of the violin (left), and removes the flickering spot on the truck (right), thus producing temporally consistent results.

over some effective frame rate. Due to the inherent trade-off between style adherence and temporal consistency, our hyperparameter selection process considers the strongest models in the former criterion and chooses the one that best satisfies the latter; specifically, among the models ranked in the top 15% by FID, we select the one that minimizes E_{warp}.

3.4. Comparison To Prior State-of-the-Art

Now we compare HyperCon to frame-wise style transfer (FST) [17] and a state-of-the-art video consistency method from Lai *et al*. (FST-vcons) [19]. For FST-vcons, we first apply FST to the video, and then apply the consistency model of Lai *et al*. as a post-translation step. Note that FST-vcons must process inputs sequentially, whereas HyperCon can operate on several time steps in parallel due to its short-range dependencies. Given efficient implementations of both methods, HyperCon's parallelizability gives it an advantage in terms of inference time on long videos.

We provide a quantitative comparison of these methods in Table 1. Since FST stylizes each frame independently, it naturally yields the worst temporal consistency as indicated by its relatively poor warping errors and patch-based consistency scores. Between FST-vcons and our HyperCon approach, the latter obtains better FID scores, warping errors, and patch-based consistency scores, indicating that it is more temporally consistent and better captures the intended style and content compared to FST-vcons.

To ensure that these conclusions match those derived from human judgements, we devise a survey on Amazon Mechanical Turk to compare HyperCon and the FST-vcons baseline on our test videos (Section 3.1). To judge overall quality, we present subjects with corresponding videos from each method and ask which is more appealing—the percentage of responses that favor our method is **57.3%** for the rain-princess style and **55.5%** for the mosaic style. As for style adherence, we present the two aforementioned videos alongside the frame-wise translated one from FST, and ask method is more similar to FST—here, the percentage of responses that favor our method is **79.1%** for rain-princess and **76.8%** for mosaic. Survey design details are included in the supplementary materials.

In Figure 6, we visually compare FST, FST-vcons, and HyperCon on two DAVIS 2017 test videos. Naturally, FST generates temporally inconsistent predictions by operating on each frame independently: observe the changing arrangement of lines across the top of the violin in Figure 6a, as well as the flashing red spot in Figure 6d. Meanwhile, FST-vcons has two systematic failures. First, it greatly desaturates predictions, observable across the full frame in Figure 6b and in the dulled orange and blue hues of the sky in Figure 6e. Second, it leaves inconsistencies intact as a result of darkening regions instead of shifting their hue; for in-

Method	DAVIS 2017					DAVIS 2019					ActivityNet				
	$D_{\text{LPIPS}}^{\downarrow}$	$D_{\text{VLPIPS}}^{\downarrow}$	FID$^{\downarrow}$	VFID$^{\downarrow}$	$E_{\text{warp}}^{\downarrow}$	$D_{\text{LPIPS}}^{\downarrow}$	$D_{\text{VLPIPS}}^{\downarrow}$	FID$^{\downarrow}$	VFID$^{\downarrow}$	$E_{\text{warp}}^{\downarrow}$	$D_{\text{LPIPS}}^{\downarrow}$	$D_{\text{VLPIPS}}^{\downarrow}$	FID$^{\downarrow}$	VFID$^{\downarrow}$	$E_{\text{warp}}^{\downarrow}$
Cxtattn [37]	0.0457	0.5838	20.94	1.435	0.002186	0.0442	0.5575	**15.55**	1.361	0.002539	**0.0432**	0.5981	**21.5173**	1.4417	0.000894
Cxtattn-vcons [19]	0.0480	0.6076	23.23	1.502	0.001780	0.0478	0.5964	18.52	1.490	0.002166	0.0448	0.6067	23.1642	1.4383	0.000689
VINet [18]	0.0616	0.6062	29.24	1.465	0.001882	0.0539	0.5455	18.22	**1.195**	0.002292	0.0608	0.6139	29.3806	1.3783	**0.000678**
HyperCon-mean (ours)	**0.0450**	**0.5272**	**18.49**	**1.073**	**0.001540**	**0.0437**	**0.5179**	16.07	1.274	**0.001847**	0.0454	**0.5728**	22.5111	**1.2251**	**0.000640**
HyperCon-median (ours)	**0.0424**	**0.5217**	**17.75**	**1.074**	**0.001614**	**0.0419**	**0.5089**	**15.24**	**1.254**	**0.001950**	**0.0441**	**0.5812**	**22.2705**	**1.2601**	0.000683

Table 2: Comparison between baseline methods and HyperCon (ours) on the simulated video inpainting task. $^{\downarrow}$ indicates that lower is better. Bold+underline and bold respectively indicate the 1st- and 2nd-place methods. Among the evaluated methods, the HyperCon models consistently place in the top two across all performance measures.

stance, in Figure 6b, the pattern of lines on the violin match the inconsistent appearance of FST. HyperCon, on the other hand, properly addresses both of these challenges—in Figure 6c, HyperCon maintains a consistent pattern of violin lines by passing relevant information between frames, and in Figure 6f, it removes the flashing red spot without desaturating the rest of the frame like FST-vcons.

4. Experiments: Video Inpainting

To evaluate HyperCon on masked inputs, we apply it to video inpainting. For the frame inpainting subroutine, we re-train the Contextual Attention model from Yu *et al*. [37] using a modified training scheme that yields higher-quality predictions. Specifically, whereas Yu *et al*. use the WGAN-GP formulation [11] to update the discriminators of their adversarial loss, we use the original cross-entropy GAN formulation [10] with spectral normalization layers [21] in the discriminator, which stabilize GAN training. Our model achieves a PSNR of 20.41 dB on the Places dataset [42], surpassing the original reported performance of 18.91 dB [37] under the same evaluation setting.

4.1. Datasets

To evaluate video inpainting methods quantitatively, we automatically synthesize occlusion masks and use the method under evaluation to inpaint the masked area. As in prior work [30, 34], our masks take the form of a rectangle at a fixed location throughout time. For each video, we randomly select the corner locations of the rectangle, restricting its height and width to lie between 15-50% of the full frame's dimensions (the masks are the same for all evaluated methods). We use the same videos for hyperparameter tuning and evaluation as those described in Section 3.1.

4.2. Evaluation Metrics

The performance measures that we use for inpainting broadly assess temporal consistency, reconstruction quality, and realism of the composited frames, *i.e*., the predicted frames with unoccluded pixels replaced by those from the input. We measure temporal consistency using E_{warp} as described in Section 3.2. For reconstruction quality, we report the mean LPIPS distance (D_{LPIPS}) [40] between corresponding composited and ground-truth (*i.e*., unoccluded) frames. Although traditional quality measures such as PSNR and SSIM [32] have been used in prior image inpainting work [37, 20], Zhang *et al*. [40] have shown that LPIPS better correlates with human judgment for paired image comparisons. Additionally, to capture spatio-temporal similarity, we define a new video-based version of LPIPS, VLPIPS, which is a distance between several layers of activations from the I3D action recognition network [4]. We exhaustively compute VLPIPS between corresponding 10-frame segments from the composited and ground-truth videos and report the overall mean. Finally, to evaluate realism, we report FID [13] between the sets of composited and ground-truth frames, as well as a video variant VFID [31] between the sets of 10-frame segments from the composited and ground-truth videos. We compute VFID using the output of I3D's final average pooling layer.

4.3. Comparison to Prior State-of-the-Art

For inpainting, we consider two variants of our model: one using a mean filter during temporal aggregation (Section 2.3), and the other using a median filter. We compare our models to three state-of-the-art baselines. The first method, Contextual Attention (Cxtattn), inpaints frames independently using the model from Yu *et al*. [37] with our improved weights (Section 4). The second method, Cxtattn-vcons, inpaints frame-wise using Cxtattn, then reduces flickering with the blind video consistency method of Lai *et al*. [19]. The second phase of Cxtattn-vcons takes the original video and the frame-wise translated video as inputs, which in this case correspond to the masked video (with a placeholder value in the masked region) and the frame-wise inpainted video, respectively. The final method, VINet [18], is a state-of-the-art DNN specifically designed and trained for video inpainting; we use their publicly-available inference code in our experiments. Among these methods, VINet is the only one tasked with inpainting videos at training time, and thus has an advantage over the other methods from having observed natural motion in masked videos.

In Table 2, we report quantitative results on our test videos for the inpainting task. Across the three test sets, HyperCon-mean obtains the lowest warping error, indicating that its predictions are most consistent with the motion of the ground-truth video (since warping error effectively checks against the estimated flow of the ground-truth

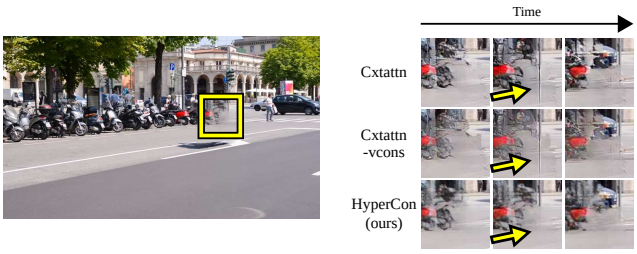
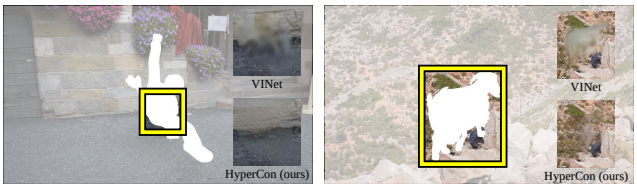


Figure 7: HyperCon substantially reduces flickering compared to the other image-to-video model transfer baselines Cxtattn and Cxtattn-vcons. Here, the pole appears for just one frame with the baselines but not with HyperCon.



(a) Boundary distortion (b) Texture comparison

Figure 8: Qualitative comparisons between VINet [18] and HyperCon (ours) for video inpainting. (a) VINet distorts the boundary of the masked wall, whereas HyperCon successfully recovers it. (b) VINet predicts textureless regions instead of realistic textures; in contrast, HyperCon produces sensible results due to its better generalization performance.

video). Between HyperCon-mean and HyperCon-median, the latter generally scores slightly worse in E_{warp} but better in the reconstruction and realism metrics; we suspect that the median filter produces sharper predictions that are more in line with real images, but are harder to match via optical flow (when computing E_{warp}) due to a wider variety of color intensities. Both HyperCon models generally outperform Cxtattn and Cxtattn-vcons across the evaluated datasets and metrics, suggesting that HyperCon is the most reliable method among image-to-video model transfer techniques. Impressively, HyperCon outperforms VINet by a substantial margin in all cases except on DAVIS 2019 under VFID *despite not having seen a single masked video at training time*; this highlights the potential of transferring frame-wise models to video.

Next, we provide a qualitative analysis in the context of real-world object removal for DAVIS 2017 training/validation videos. Comparing our HyperCon method to Cxtattn and Cxtattn-vcons, we found several cases where the baseline methods produce flickering artifacts while ours does not (*e.g.*, the pole in Figure 7). Additionally, due to the darkening behavior mentioned in Section 3.4, Cxtattn-vcons generates darkened predictions that do not blend in with the surrounding region as convincingly as those from HyperCon. Furthermore, HyperCon produces the fewest "checkerboard artifacts" [23] since they are temporally unstable and thus get filtered out by HyperCon during aggregation. We compare checkerboard artifacts and the darkening behavior in the supplementary materials.

We conclude our analysis by contrasting HyperCon with VINet [18], the only evaluated method to have seen masked videos during training. We highlight two systematic failures of VINet that HyperCon overcomes: (i) boundary distortion and (ii) textureless prediction. Regarding the first failure, VINet often corrupts its inpainting result over time by incorrectly warping the inpainted structure. For example, in Figure 8a, VINet distorts the outline of the wall due to the continuous occlusion in that region; meanwhile, HyperCon successfully hallucinates the rigid wall boundary. As for the second issue, VINet sometimes initializes the inpainted region with a textureless prediction and fails to populate it with realistic texture throughout the video. In Figure 8b, we compare this behavior with HyperCon, which generates sensible background textures. We posit that HyperCon is better able to hallucinate missing details because it has seen substantially more scenes than VINet during training (*i.e.*, millions of images versus thousands of videos).

5. Discussion and Conclusion

In terms of weaknesses, HyperCon does not enforce long-range temporal dependencies beyond the aggregation dilation rate, so it cannot commit to stylization or inpainting choices for the entirety of extremely long videos. In addition, as with other image-to-video model transfer methods, HyperCon is subject to egregious errors of the frame-wise model, which can propagate downstream in certain cases. Integrating the three steps of HyperCon into one trainable, parameter-efficient model could help alleviate these issues by sharing more information between steps and enabling more input frames per unit of memory. Despite these pitfalls, HyperCon is still a promising image-to-video model transfer method for video-to-video translation tasks as seen by its strength in two widely different tasks. We hope that it helps pave the way for further progress in this direction.

Acknowledgements

This work was completed in part while Ryan Szeto was an intern at Samsung Semiconductor, Inc. and was sponsored by DARPA and AFRL under agreement FA8750-17-2-0112, and the U.S. Department of Commerce, NIST under financial assistance award 60NANB17D191. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. This work reflects the opinions and conclusions of its authors, but not necessarily its funding agents.

References

[1] Aayush Bansal, Shugao Ma, Deva Ramanan, and Yaser Sheikh. Recycle-GAN: Unsupervised Video Retargeting. In *European Conference on Computer Vision*, pages 119–135, 2018.

[2] Nicolas Bonneel, James Tompkin, Kalyan Sunkavalli, Deqing Sun, Sylvain Paris, and Hanspeter Pfister. Blind Video Temporal Consistency. *ACM Transactions on Graphics*, 34(6):1–9, October 2015.

[3] Fabian Caba Heilbron, Victor Escorcia, Bernard Ghanem, and Juan Carlos Niebles. ActivityNet: A Large-Scale Video Benchmark for Human Activity Understanding. In *IEEE Conference on Computer Vision and Pattern Recognition*, June 2015.

[4] Joao Carreira and Andrew Zisserman. Quo Vadis, Action Recognition? A New Model and the Kinetics Dataset. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 6299–6308, 2017.

[5] Ya-LIang Chang, Zhe Yu Liu, Kuan-Ying Lee, and Winston Hsu. Learnable Gated Temporal Shift Module for Deep Video Inpainting. In *British Machine Vision Conference*, Sept. 2019.

[6] Yang Chen, Yingwei Pan, Ting Yao, Xinmei Tian, and Tao Mei. Mocycle-GAN: Unpaired Video-to-Video Translation. In *ACM International Conference on Multimedia*, pages 647–655, 2019.

[7] Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang. Image Super-Resolution Using Deep Convolutional Networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(2):295–307, 2015.

[8] Xuan Dong, Boyan Bonev, Yu Zhu, and Alan L. Yuille. Region-Based Temporally Consistent Video Post-Processing. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 714–722. IEEE, June 2015.

[9] Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge. Image Style Transfer Using Convolutional Neural Networks. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2414–2423, 2016.

[10] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative Adversarial Nets. In *Advances in Neural Information Processing Systems*, 2014.

[11] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron Courville. Improved Training of Wasserstein GANs. In *Advances in Neural Information Processing Systems*, 2017.

[12] Agrim Gupta, Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Characterizing and Improving Stability in Neural Style Transfer. In *IEEE International Conference on Computer Vision*, pages 4067–4076, 2017.

[13] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium. In *Advances in Neural Information Processing Systems*, pages 6626–6637, 2017.

[14] Xin Hong, Pengfei Xiong, Renhe Ji, and Haoqiang Fan. Deep Fusion Network for Image Completion. In *ACM International Conference on Multimedia*, pages 2033–2042, 2019.

[15] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-Image Translation with Conditional Adversarial Networks. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1125–1134, 2017.

[16] Huaizu Jiang, Deqing Sun, Varan Jampani, Ming Hsuan Yang, Erik Learned-Miller, and Jan Kautz. Super SloMo: High Quality Estimation of Multiple Intermediate Frames for Video Interpolation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 9000–9008, nov 2018.

[17] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual Losses for Real-Time Style Transfer and Super-Resolution. In *European Conference on Computer Vision*, 2016.

[18] Dahun Kim, Sanghyun Woo, Joon-Young Lee, and In So Kweon. Deep Video Inpainting. In *IEEE Conference on Computer Vision and Pattern Recognition*, May 2019.

[19] Wei-Sheng Lai, Jia-Bin Huang, Oliver Wang, Eli Shechtman, Ersin Yumer, and Ming-Hsuan Yang. Learning Blind Video Temporal Consistency. In *European Conference on Computer Vision*, August 2018.

[20] Guilin Liu, Fitsum A. Reda, Kevin J. Shih, Ting-Chun Wang, Andrew Tao, and Bryan Catanzaro. Image Inpainting for Irregular Holes Using Partial Convolutions. In *European Conference on Computer Vision*, 2018.

[21] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral Normalization for Generative Adversarial Networks. In *International Conference on Learning Representations*, 2018.

[22] Simon Niklaus, Long Mai, and Feng Liu. Video Frame Interpolation via Adaptive Separable Convolution. In *IEEE International Conference on Computer Vision*, 2017.

[23] Augustus Odena, Vincent Dumoulin, and Chris Olah. Deconvolution and Checkerboard Artifacts. *Distill*, 2016.

[24] Kwanyong Park, Sanghyun Woo, Dahun Kim, Donghyeon Cho, and In So Kweon. Preserving Semantic and Temporal Consistency for Unpaired Video-to-Video Translation. In *ACM International Conference on Multimedia*, pages 1248–1257, 2019.

[25] F. Perazzi, J. Pont-Tuset, B. McWilliams, L. Van Gool, M. Gross, and A. Sorkine-Hornung. A Benchmark Dataset and Evaluation Methodology for Video Object Segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 724–732. IEEE, jun 2016.

[26] Fitsum A. Reda, Guilin Liu, Kevin J. Shih, Robert Kirby, Jon Barker, David Tarjan, Andrew Tao, and Bryan Catanzaro. SDC-Net: Video Prediction Using Spatially-Displaced Convolution. In *European Conference on Computer Vision*, pages 718–733, 2018.

[27] Manuel Ruder, Alexey Dosovitskiy, and Thomas Brox. Artistic Style Transfer for Videos. In *German Conference on Pattern Recognition*, pages 26–36. Springer, 2016.

[28] Mehdi SM Sajjadi, Raviteja Vemulapalli, and Matthew Brown. Frame-Recurrent Video Super-Resolution. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 6626–6634, 2018.

[29] Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz. PWC-Net: CNNs for Optical Flow Using Pyramid, Warping, and Cost Volume. In *IEEE Conference on Computer Vision and Pattern Recognition*, sep 2017.

[30] Chuan Wang, Haibin Huang, Xiaoguang Han, and Jue Wang. Video Inpainting by Jointly Learning Temporal Structure and Spatial Details. In *AAAI Conference on Artificial Intelligence*, 2019.

[31] Ting-Chun Wang, Ming-Yu Liu, Jun-Yan Zhu, Guilin Liu, Andrew Tao, Jan Kautz, and Bryan Catanzaro. Video-to-Video Synthesis. In *Advances in Neural Information Processing Systems*, 2018.

[32] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image Quality Assessment: From Error Visibility to Structural Similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004.

[33] Ning Xu, Linjie Yang, Yuchen Fan, Jianchao Yang, Dingcheng Yue, Yuchen Liang, Brian Price, Scott Cohen, and Thomas Huang. Youtube-VOS: Sequence-to-Sequence Video Object Segmentation. In *European Conference on Computer Vision*, pages 585–601, 2018.

[34] Rui Xu, Xiaoxiao Li, Bolei Zhou, and Chen Change Loy. Deep Flow-Guided Video Inpainting. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2019.

[35] Xuewen Yang, Dongliang Xie, and Xin Wang. Crossing-Domain Generative Adversarial Networks for Unsupervised Multi-Domain Image-to-Image Translation. In *ACM International Conference on Multimedia*, pages 374–382, 2018.

[36] Chun-Han Yao, Chia-Yang Chang, and Shao-Yi Chien. Occlusion-Aware Video Temporal Consistency. In *ACM International Conference on Multimedia*, pages 777–785, New York, New York, USA, 2017. ACM Press.

[37] Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas S. Huang. Generative Image Inpainting with Contextual Attention. In *IEEE Conference on Computer Vision and Pattern Recognition*, January 2018.

[38] Haotian Zhang, Long Mai, Ning Xu, Zhaowen Wang, John Collomosse, and Hailin Jin. An Internal Learning Approach to Video Inpainting. In *IEEE International Conference on Computer Vision*, October 2019.

[39] Richard Zhang, Phillip Isola, and Alexei A. Efros. Colorful Image Colorization. In *European Conference on Computer Vision*, pages 649–666. Springer, 2016.

[40] Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The Unreasonable Effectiveness of Deep Features as a Perceptual Metric. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 586–595. IEEE, June 2018.

[41] Richard Zhang, Jun-Yan Zhu, Phillip Isola, Xinyang Geng, Angela S Lin, Tianhe Yu, and Alexei A Efros. Real-Time User-Guided Image Colorization with Learned Deep Priors. In *ACM Transactions on Graphics*, 2017.

[42] Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. Places: A 10 Million Image Database for Scene Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(6):1452–1464, June 2018.