

Unsupervised Video Interpolation Using Cycle Consistency

Fitsum A. Reda Deqing Sun* Aysegul Dundar Mohammad Shoeybi Guilin Liu
 Kevin J. Shih Andrew Tao Jan Kautz Bryan Catanzaro
 NVIDIA

Abstract

Learning to synthesize high frame rate videos via interpolation requires large quantities of high frame rate training videos, which, however, are scarce, especially at high resolutions. Here, we propose unsupervised techniques to synthesize high frame rate videos directly from low frame rate videos using cycle consistency. For a triplet of consecutive frames, we optimize models to minimize the discrepancy between the center frame and its cycle reconstruction, obtained by interpolating back from interpolated intermediate frames. This simple unsupervised constraint alone achieves results comparable with supervision using the ground truth intermediate frames. We further introduce a pseudo supervised loss term that enforces the interpolated frames to be consistent with predictions of a pre-trained interpolation model. The pseudo supervised loss term, used together with cycle consistency, can effectively adapt a pre-trained model to a new target domain. With no additional data and in a completely unsupervised fashion, our techniques significantly improve pre-trained models on new target domains, increasing PSNR values from 32.84dB to 33.05dB on the Slowflow and from 31.82dB to 32.53dB on the Sintel evaluation datasets. Code is available at <https://github.com/NVIDIA/unsupervised-video-interpolation>.

1. Introduction

With the advancement of modern technology, consumer-grade smartphones and digital cameras can now record videos at high frame rates (e.g. 240 frames-per-second). However, achieving this comes at the cost of high power consumption, larger storage requirements, and reduced video resolution. Given these limitations, regular events are not typically recorded at high frame rates. Yet, important life events happen unexpectedly and hence tend to be recorded at standard frame rates. It is thus greatly desirable to have the ability to produce arbitrarily high FPS videos

*Currently affiliated with Google.

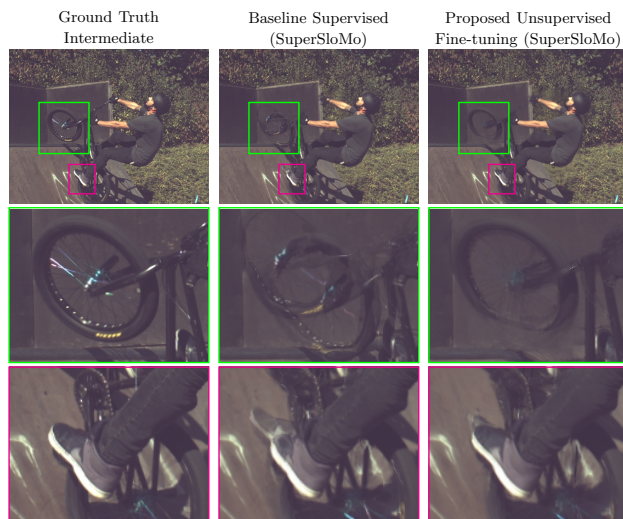


Figure 1. Visual results of a sample from Slowflow dataset. Baseline supervised model is trained with Adobe and YouTube datasets and proposed unsupervised model is fine-tuned with Slowflow. See project [website](#) for video comparisons.

from low FPS videos.

Video frame interpolation addresses this need by generating one or more intermediate frames from two consecutive frames. Increasing the number of frames in videos essentially allows one to visualize events in slow motion and appreciate content better. Often, video interpolation techniques are employed to increase the frame rate of already recorded videos, or in streaming applications to provide a high refresh rate or a smooth viewing experience.

Video interpolation is a classical vision and graphics problem [3, 21, 27] and has recently received renewed research attention [10, 16, 11, 14]. Particularly, supervised learning with convolutional neural networks (CNNs) has been widely employed to learn video interpolation from paired input and ground truth frames, often collected from raw video data. For instance, recent CNN-based approaches such as [6] and [16], trained with large quantities of public high FPS videos, obtain high quality interpolation results when the test videos are similar to the training ones.

However, these methods may fail if the training data dif-

fer from the target domain. For instance, the target domain might be to slow down videos of fish taken underwater, but available training data only contains regular outdoor scenes, thus leading to a content gap. Additionally, there might be more subtle domain gaps due to differences such as camera parameters, encoding codecs, and lighting, leading to the well-known co-variate shift problem [18]. It is impractical to address the issue by collecting high FPS videos covering all possible scenarios, because it is expensive to capture and store very high FPS videos, e.g., videos with more than 1000-fps at high spatial resolutions.

In this work, we propose a set of unsupervised learning techniques to alleviate the need for high FPS training videos and to shrink domain gaps in video interpolation. Specifically, we propose to learn video frame interpolation, without paired training data, by enforcing models to satisfy a cycle consistency constraint [2] in the time. That is, for a given triplet of consecutive frames, if we generate the two intermediate frames between the two consecutive frames, and generate back their intermediate frame, the resulting frame must match the original input middle frame (shown schematically in Fig. 3). We show such simple constraint alone is effective to learn video interpolation, and achieve results that compete with supervised approaches.

In domain adaptation applications, where we have access to models pre-trained on out-of-domain datasets, but lack ground truth frames in target domains, we also propose unsupervised fine-tuning techniques that leverage such pre-trained models (See Fig. 2). We fine-tune models on target videos, with no additional data, by optimizing to jointly satisfy cycle consistency and minimize the discrepancy between generated intermediate frames and corresponding predictions from the known pre-trained model. We demonstrate our joint optimization strategy leads to significantly superior accuracy in upscaling frame rate of target videos than fine-tuning with cycle consistency alone or directly applying the pre-trained models on target videos (see Fig. 1).

Cycle consistency has been utilized for image matching [25], establishing dense 3D correspondence over object instances [23], or in learning unpaired image-to-image translation in conjunction with Generative Adversarial Networks (GANs) [26]. To the best of our knowledge, this is the first attempt to use a cycle consistency constraint to learn video interpolation in a completely unsupervised way.

To summarize, the contributions of our work include:

- We propose unsupervised approaches to learn video interpolation in the absence of paired training data by optimizing neural networks to satisfy cycle consistency constraints in time domain.
- We learn to synthesize arbitrarily high frame rate videos by learning from only low frame rate raw videos.
- We demonstrate the effectiveness of our unsupervised

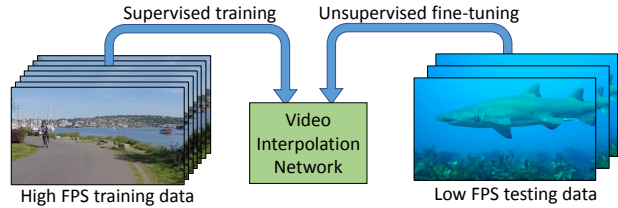


Figure 2. Video interpolation methods may fail if the training data differ from the test data. In this work, we propose an unsupervised fine-tuning technique to reduce domain gaps.

techniques in reducing domain gaps in video interpolation.

2. Related Works

Video Interpolation: The task is to interpolate intermediate frames between a pair of input frames. Classical approaches such as [4] and [12] estimate optical flow and interpolate intermediate frames at intermediate positions along the estimated trajectory of pixels, and further make use of forward and backward optical flow consistency to reason about occlusions.

Given the recent rise in popularity of deep learning methods, several end-to-end trainable methods have been proposed for video interpolation. Specifically, these methods can be trained to interpolate frames using just input and target frames and no additional supervision. Liu et al. [9] and Jiang et al. [6] both indirectly learn to predict optical flow using frame interpolation. Works such as [15, 16] are similarly end-to-end trainable, but instead of learning optical flow vectors to warp pixels, they predict adaptive convolutional kernels to apply at each location of the two input frames. Our work presents unsupervised techniques to train or fine-tune any video interpolation model, for instance the Super SloMo [6], which predicts multiple intermediate frames, or the Deep Voxel Flow [9], which predicts one intermediate frame.

Cycle Consistency: One of the key elements of our proposed method is the use of a cycle consistency constraint. This constraint encourages the transformations predicted by a model to be invertible, and is often used to regularize the model behavior when direct supervision is unavailable. Cycle consistency has been used in a variety of applications, including determining the quality of language translations [2], semi-supervised training for image-description generation [13], dense image correspondences [24], identifying false visual relations in structure from motion [22], and image-to-image translation [26], to name a few.

A cycle consistency constraint, in the context of video interpolation, means that we should be able to reconstruct the original input frames by interpolating between predicted intermediate frames at the appropriate time stamps. Most related to our work is [8], which uses such a constraint to

regularize a fully supervised video interpolation model. Our work differs in several critical aspects. First, our method is based on the Super SloMo [6] architecture, and is thus capable of predicting intermediate frames at arbitrary timestamps, whereas [8] is specifically trained to predict the middle timestamp. Next, and most critically, our proposed method is fully unsupervised. This means that the target intermediate frame is never used for supervision, and that it can learn to produce high frame rate interpolated sequences from any lower frame rate sequence.

3. Method

In this work, we propose to learn to interpolate arbitrarily many intermediate frames from a pair of input frames, in an unsupervised fashion, with no paired intermediate ground truth frames. Specifically, given a pair of input frames \mathbf{I}_0 and \mathbf{I}_1 , we generate intermediate frame $\hat{\mathbf{I}}_t$, as

$$\hat{\mathbf{I}}_t = \mathcal{M}(\mathbf{I}_0, \mathbf{I}_1, t), \quad (1)$$

where $t \in (0, 1)$ is time, and \mathcal{M} is a video frame interpolation model we want to learn without supervision. We realize \mathcal{M} using deep convolutional neural networks (CNN). We chose CNNs as they are able to model highly non-linear mappings, are easy to implement, and have been proven to be robust for various vision tasks, including image classification, segmentation, and video interpolation.

Inspired by the recent success in learning unpaired image-to-image translation using Generative Adversarial Networks (GAN) [26], we propose to optimize \mathcal{M} to maintain cycle consistency in time. Let $\mathbf{I}_0, \mathbf{I}_1$ and \mathbf{I}_2 are a triplet of consecutive input frames. We define the time-domain cycle consistency constraint such that for generated intermediate frames at time t between $(\mathbf{I}_0, \mathbf{I}_1)$ and between $(\mathbf{I}_1, \mathbf{I}_2)$, a subsequently generated intermediate frame at time $(1 - t)$ between the interpolated results $(\hat{\mathbf{I}}_t, \hat{\mathbf{I}}_{t+1})$ must match the original middle input frame \mathbf{I}_1 . Mathematically, a cycle reconstructed frame using \mathcal{M} is given by,

$$\hat{\mathbf{I}}_1 = \mathcal{M}(\mathcal{M}(\mathbf{I}_0, \mathbf{I}_1, t), \mathcal{M}(\mathbf{I}_1, \mathbf{I}_2, t), 1 - t). \quad (2)$$

We then optimize \mathcal{M} to minimize the reconstruction error between $\hat{\mathbf{I}}_1$ and \mathbf{I}_1 , as

$$\arg \min_{\theta(\mathcal{M})} (\|\hat{\mathbf{I}}_1 - \mathbf{I}_1\|_1). \quad (3)$$

Figure 3 schematically presents our cycle consistency based approach.

A degenerate solution to optimizing equation 3 might be to copy the input frames as the intermediate predictions (i.e. outputs). However, in practice this does not occur. In order for \mathcal{M} to learn to do copy frames in this way, it would have to learn to identify the input’s time information within a single forward operation (eq. 2), as \mathbf{I}_1 is a $t = 1$ input in the

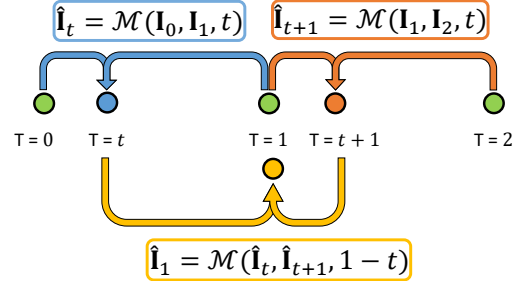


Figure 3. An overview of time-domain cycle consistency constrain. $\mathbf{I}_0, \mathbf{I}_1$ and \mathbf{I}_2 , shown as green circles, are a triplet of consecutive input frames. If we generate intermediate frames at time t between $(\mathbf{I}_0, \mathbf{I}_1)$ and between $(\mathbf{I}_1, \mathbf{I}_2)$, and subsequently generate back an intermediate frame at time $(1 - t)$ between $(\hat{\mathbf{I}}_t, \hat{\mathbf{I}}_{t+1})$, the resulting frame must match the original middle input frame, \mathbf{I}_1 .

first pass, and \mathbf{I}_1 is a $t = 0$ input in the second pass. This is difficult, since the same weights of \mathcal{M} are used in both passes. We support this claim in all of our experiments, where we compared our learned approach using equation 3 with the trivial case of using inputs as intermediate prediction.

It is true that triplets of input frames could be exploited directly. For example, the reconstruction error between $\mathcal{M}(\mathbf{I}_0, \mathbf{I}_2, t = 0.5)$ and \mathbf{I}_1 could be used without cycle consistency. However, our experiments in Section 4.4.2 suggest that larger time-step lead to significantly worse accuracy if used without cycle consistency.

Optimizing \mathcal{M} to satisfy cycle consistency (CC) constraint in time, as will show in our experiments in Sections 4.2 and 4.3, is effective and is able to generate arbitrarily many intermediate frames that are realistic and temporally smooth. It also produces results that are competitive with supervised approaches, including the same model \mathcal{M} , but trained with supervision.

In this work, we also propose techniques that can make unsupervised fine-tuning processes robust. It is quite common to have access to out-of-domain training videos in abundance or access to already pre-trained interpolation models. On the other hand, target domain videos are often limited in quantity, and most critically, lack ground truth intermediate frames. We aim to optimize \mathcal{M} in target videos to jointly satisfy cycle consistency as defined in equation 3 and also learn to approximate a known pre-trained interpolation model, denoted as \mathcal{F} . Mathematically, our modified objective is given as,

$$\arg \min_{\theta(\mathcal{M})} (\|\hat{\mathbf{I}}_1 - \mathbf{I}_1\|_1 + \|\hat{\mathbf{I}}_t - \mathcal{F}(\mathbf{I}_0, \mathbf{I}_1, t)\|_1 + \|\hat{\mathbf{I}}_{t+1} - \mathcal{F}(\mathbf{I}_1, \mathbf{I}_2, t)\|_1), \quad (4)$$

where $\hat{\mathbf{I}}_1$ is the cycle reconstructed frame given by equation

2, $\hat{\mathbf{I}}_t$ and $\hat{\mathbf{I}}_{t+1}$ are given by equation 1, and $\theta^{(\mathcal{M})}$ are the parameters of \mathcal{M} that our optimization processes update.

The added objective function to approximate \mathcal{F} , help regularize \mathcal{M} to generate realistic *hidden* intermediate frames $\hat{\mathbf{I}}_t$ an $\hat{\mathbf{I}}_{t+1}$ by constraining them to resemble predictions of a known frame interpolation model, \mathcal{F} . As optimization progresses and \mathcal{M} learns to pick-up interpolation concepts, one can limit the contribution of the regularizing ‘‘pseudo’’ supervised (PS) loss and let optimizations be guided more by the cycle consistency. Such a surrogate loss term, derived from estimated intermediate frames, can make our training processes converge faster or make our optimization processes robust by exposing it to many variations of \mathcal{F} .

In this work, for the sake of simplicity, we chose \mathcal{F} to be the same as our \mathcal{M} , but pre-trained with supervision on a disjoint dataset that has ground-truth high frame rate video, and denote it as \mathcal{M}_{pre} . Our final objective can be given by,

$$\arg \min_{\theta^{(\mathcal{M})}} \left(\lambda_{rc} \|\hat{\mathbf{I}}_1 - \mathbf{I}_1\|_1 + \lambda_{rp} \|\hat{\mathbf{I}}_t - \mathcal{M}_{pre}(\mathbf{I}_0, \mathbf{I}_1, t)\|_1 + \lambda_{rp} \|\hat{\mathbf{I}}_{t+1} - \mathcal{M}_{pre}(\mathbf{I}_1, \mathbf{I}_2, t)\|_1 \right), \quad (5)$$

where λ_{rc} and λ_{rp} are weights of CC and PS losses.

As we will show in the experiments, optimizing equation 5 by relying only on the PS loss, without cycle consistency, will teach \mathcal{M} to perform at best as good as \mathcal{M}_{pre} , i.e., the model used in the same PS loss. However, as we show in Section 4.4.1, by weighting cycle consistency and PS losses appropriately, we achieve frame interpolation results that are superior to those obtained by learning using either CC or PS losses alone.

Finally, we implement our \mathcal{M} using the Super SloMo video interpolation model [6]. Super SloMo is a state of the art flow-based CNN for video interpolation, capable of synthesizing an arbitrary number of high quality and temporally stable intermediate frames. Our technique is not limited to this particular interpolation model, but could be adopted with others as well.

In the subsequent subsections we provide a short summary of the Super SloMo model, our loss functions, and techniques we employed to make our unsupervised training processes stable.

3.1. Video Interpolation Model

To generate one or more intermediate frames $\hat{\mathbf{I}}_t$ from a pair of input frames $(\mathbf{I}_0, \mathbf{I}_1)$, first the Super SloMo model estimates an approximate bi-directional optical flow from any arbitrary time t to 0, $\mathbf{F}_{t \rightarrow 0}$, and from t to 1, $\mathbf{F}_{t \rightarrow 1}$. Then, it generates a frame by linearly blending the input frames after they are warped by the respective estimated op-

tical flows, as

$$\hat{\mathbf{I}}_t = \alpha \mathcal{T}(\mathbf{I}_0, \mathbf{F}_{t \rightarrow 0}) + (1 - \alpha) \mathcal{T}(\mathbf{I}_1, \mathbf{F}_{t \rightarrow 1}), \quad (6)$$

where \mathcal{T} is an operation that bilinearly samples input frames using the optical flows, and α weighs the contribution of each term. The blending weight α models both global property of temporal consistency as well as local or pixel-wise occlusion or dis-occlusion reasoning. For instance, to maintain temporal consistency, \mathbf{I}_0 must contribute more to $\hat{\mathbf{I}}_t$ when t is close to 0. Similarly, \mathbf{I}_1 contributes more to $\hat{\mathbf{I}}_t$, when t is close to 1.

To cleanly blend the two images, an important property of video frame interpolation is exploited, i.e. not all pixels at time t are visible in both input frames. Equation 6 can thus be defined by decomposing α to model both temporal consistency and occlusion or de-occlusions, as

$$\hat{\mathbf{I}}_t = \frac{1}{Z} \left((1-t) \mathbf{V}_{t \leftarrow 0} \mathcal{T}(\mathbf{I}_0, \mathbf{F}_{t \rightarrow 0}) + t \mathbf{V}_{t \leftarrow 1} \mathcal{T}(\mathbf{I}_1, \mathbf{F}_{t \rightarrow 1}) \right), \quad (7)$$

where $\mathbf{V}_{t \leftarrow 0}$ and $\mathbf{V}_{t \leftarrow 1}$ are visibility maps, and $Z = (1-t) \mathbf{V}_{t \leftarrow 0} + t \mathbf{V}_{t \leftarrow 1}$ is a normalisation factor. For a pixel p , $\mathbf{V}_{t \leftarrow 0}(p) \in [0, 1]$ denotes visibility of p at time t (0 means fully occluded or is invisible at t).

The remaining challenge is estimating the intermediate bi-direction optical flows $(\mathbf{F}_{t \rightarrow 0}, \mathbf{F}_{t \rightarrow 1})$ and the corresponding visibility maps $(\mathbf{V}_{t \leftarrow 0}, \mathbf{V}_{t \leftarrow 1})$. For more information, we refer the reader to [6].

3.2. Training and Loss Functions

We train \mathcal{M} to generate arbitrarily many intermediate frames $\{\hat{\mathbf{I}}_{t_i}\}_{i=1}^N$ without using the corresponding ground-truth intermediate frames $\{\mathbf{I}_{t_i}\}_{i=1}^N$, with N and $t_i \in (0, 1)$ being frame count and time, respectively. Specifically, as described in Section 3, we optimize \mathcal{M} to (a) minimize the errors between the cycle reconstructed frame $\hat{\mathbf{I}}_1$ and \mathbf{I}_1 and (b) to minimize the errors between the intermediately predicted frames $\hat{\mathbf{I}}_t$ and $\hat{\mathbf{I}}_{t+1}$ and the corresponding estimated or pseudo ground-truth frames $\mathcal{M}_{pre}(\mathbf{I}_0, \mathbf{I}_1, t)$ and $\mathcal{M}_{pre}(\mathbf{I}_1, \mathbf{I}_2, t)$.

Note that, during optimization a cycle reconstructed frame $\hat{\mathbf{I}}_1$ can be obtained via arbitrarily many intermediately generated frames $\{\hat{\mathbf{I}}_{t_i}, \hat{\mathbf{I}}_{t_i+1}\}_{i=1}^N$. Thus, many reconstruction errors can be computed from a single triplets of training frames $\{\mathbf{I}_0, \mathbf{I}_1, \mathbf{I}_2\}$. However, we found doing so makes optimizations unstable and often unable to converge to acceptable solutions. Instead, we found establishing very few reconstruction errors per triplet to make our training stable and generate realistic intermediate frames. In our experiments, we calculate one reconstruction error per triplet, at random time $t_i \in (0, 1)$.

Our training loss functions are given by,

$$\mathcal{L} = \lambda_{rc} \mathcal{L}_{rc} + \lambda_{rp} \mathcal{L}_{rp} + \lambda_p \mathcal{L}_p + \lambda_w \mathcal{L}_w + \lambda_s \mathcal{L}_s, \quad (8)$$

where \mathcal{L}_{rc} , defined as,

$$\mathcal{L}_{rc} = \|\hat{\mathbf{I}}_1 - \mathbf{I}_1\|_1, \quad (9)$$

models how good the cycle reconstructed frame is, and \mathcal{L}_{rp} , defined as,

$$\mathcal{L}_{rp} = \|\hat{\mathbf{I}}_{t_i} - \mathcal{M}_{pre}(\mathbf{I}_0, \mathbf{I}_1, t_i)\|_1 + \|\hat{\mathbf{I}}_{t_i+1} - \mathcal{M}_{pre}(\mathbf{I}_1, \mathbf{I}_2, t_i)\|_1, \quad (10)$$

models how close the *hidden* intermediate frames are to our pseudo intermediate frames. \mathcal{L}_p models a perceptual loss defined as the L_2 norm on the high-level features of VGG-16 model, pre-trained on ImageNet, and is given as,

$$\mathcal{L}_p = \|\Psi(\hat{\mathbf{I}}_1) - \Psi(\mathbf{I}_1)\|_2, \quad (11)$$

with Ψ representing the `conv4_3` feature of the VGG-16 model.

Our third loss, \mathcal{L}_w is a warping loss that make optical flow predictions realistic, and is given by,

$$\mathcal{L}_w = \|\mathcal{T}(\mathbf{I}_0, \mathbf{F}_{1 \rightarrow 0}) - \mathbf{I}_1\|_1 + \|\mathcal{T}(\mathbf{I}_1, \mathbf{F}_{0 \rightarrow 1}) - \mathbf{I}_0\|_1 + \|\mathcal{T}(\mathbf{I}_1, \mathbf{F}_{2 \rightarrow 1}) - \mathbf{I}_2\|_1 + \|\mathcal{T}(\mathbf{I}_2, \mathbf{F}_{1 \rightarrow 2}) - \mathbf{I}_1\|_1 + \|\mathcal{T}(\hat{\mathbf{I}}_t, \mathbf{F}_{t+1 \rightarrow t}) - \hat{\mathbf{I}}_{t+1}\|_1 + \|\mathcal{T}(\hat{\mathbf{I}}_{t+1}, \mathbf{F}_{t \rightarrow t+1}) - \hat{\mathbf{I}}_t\|_1. \quad (12)$$

In a similar way as the Super SloMo framework, we also enforce a smoothness constraint to encourage neighbouring optical flows to have similar optical flow values, and it is given as,

$$\mathcal{L}_s = \|\Delta \mathbf{F}_{t \rightarrow t+1}\|_1 + \|\Delta \mathbf{F}_{t+1 \rightarrow t}\|_1 + \|\Delta \mathbf{F}_{0 \rightarrow 1}\|_1 + \|\Delta \mathbf{F}_{1 \rightarrow 0}\|_1 + \|\Delta \mathbf{F}_{1 \rightarrow 2}\|_1 + \|\Delta \mathbf{F}_{2 \rightarrow 1}\|_1, \quad (13)$$

where $\mathbf{F}_{t \rightarrow t+1}$ and $\mathbf{F}_{t+1 \rightarrow t}$ are the forward and backward optical flows between the the intermediately predicted $\hat{\mathbf{I}}_t$ and $\hat{\mathbf{I}}_{t+1}$ frames.

Finally, we linearly combine our losses using experimentally selected weights: $\lambda_{rc} = 0.8$, $\lambda_{rp} = 0.8$, $\lambda_p = 0.05$, $\lambda_w = 0.4$, and $\lambda_s = 1$, see Section 4.4.1 for details on weight selection.

3.3. Implementation Details

We use Adam solver [7] for optimization with $\beta_1 = 0.9$, $\beta_2 = 0.999$, and no weight decay. We train our models for a total of 500 epochs, with a total batch size of 32 on 16 V100 GPUs, using distributed training over two nodes. Initial learning rate is set to $1e^{-4}$, and then scaled-down by 10 after 250, and again after 450 epochs.

4. Experiments

4.1. Datasets and Metrics

Table 1 summarizes datasets we used for training and evaluation. We used Adobe-240fps [20] (76.7K frames) and YouTube-240fps [6] (296.4K frames) for supervised training to establish baselines. For unsupervised training, we considered low FPS Battlefield1-30fps videos [17] (320K frames), and Adobe-30fps (9.5K frames), obtained by temporally sub-sampling Adobe-240fps videos, by keeping only every other 8th frame. We chose game frames because they contain a large range of motion that could make learning processes robust. We used UCF101 [19] datasets for evaluation.

To study our unsupervised fine-tuning techniques in bridging domain gaps, we considered two particularly distinct, high FPS and high resolution, target video datasets: Slowflow-240fps and Sintel-1008fps [5]. Slowflow is captured from real life using professional high speed cameras, whereas Sintel is a game content. We split Slowflow dataset into disjoint low FPS train (3.4K frames) and a high FPS test (414 frames) subsets, see Table 1. We create the test set by selecting nine frames in each of the 46 clips. We then create our low FPS train subset by temporally sub-sampling the remaining frames from 240-fps to 30-fps. During evaluation, our models take as input the first and ninth frame in each test clip and interpolate seven intermediate frames. We follow a similar procedure for Sintel-1008fps [5], but interpolate 41 intermediate frames, i.e., conversion of frame rate from 24- to 1008-fps.

To quantitatively evaluate interpolations we considered Peak-Signal-To-Noise (PSNR), the Structural-Similarity-Image-Metric (SSIM), and the Interpolation-Error (IE) [1], which is calculated as the root mean-squared-error between generated and ground truth frames. High PSNR and SSIM scores indicate better quality, whereas for IE score it is the opposite.

4.2. Large-Scale Unsupervised Training

In this section, we consider the scenario where we do not have any high frame rate videos to train a base model, but we have abundant low frame rate videos. We test our models on UCF101 dataset; for every triplet of frames, the first and third ones are used as input to predict the second frame.

Results are presented in Table 2. Our unsupervised technique trained on Adobe-30fps performs competitively with results obtained with supervision on Adobe-240fps, achieving PSNR of 34.47, and 34.63 respectively. Compared to the supervised training, our unsupervised training uses 1/8th of the frame count, and performs comparably to techniques trained with supervision. This shows the effectiveness of cycle consistency constraint alone in training models, from

	FPS	Frame count	Clip count	Resolution	Train	Test
UCF101 [19]	25	1,137	379	256 × 256		x
YouTube [6]	240	296,352	1,014	720 × 1280	x	
Battlefield-1 [17]	30	329,222	363	1080 × 1920	x	
Adobe [20]	30	9,551	112	720 × 1280	x	
	240	76,768			x	
Slowflow [5]	30	3,470	46	2048 × 2560	x	
	240	414				x
Sintel [5]	24	551	13	872 × 2048	x	
	1008	559				x

Table 1. Statistics of video datasets used in training or evaluation.

UCF101				
	Training data	PSNR(↑)	SSIM(↑)	IE(↓)
Trivial Copy	N/A	31.27	0.895	8.35
Baseline	Adobe-240fps	34.63	0.946	5.48
Proposed	Adobe-30fps	34.47	0.946	5.50
	BattleField-30fps	34.55	0.947	5.38

Table 2. Interpolation results for single intermediate frame interpolation on UCF101.

random initialization, for video frame interpolation. We further study the impact of frame count in unsupervised training. For this study, we used the low FPS Battlefield-1 sequences. The higher the frame count of low FPS frames, the better our unsupervised model performs, when evaluated on UCF101. Using Battlefield-30fps, at frame count four times larger than Adobe-240fps, we achieve results on par with supervised techniques, achieving IE of 5.38 and 5.48, respectively.

Table 2 also presents results of trivial copy, which is the simple case of using inputs as predictions. Compared to cycle consistency, trivial prediction leads to significantly worse interpolation, further indicating our approach does in fact allow us to synthesize intermediate frames from unpaired raw video frames.

4.3. Unsupervised Fine-tuning for Domain Transfer

One particularly common situation in video frame interpolation is that we have access to pre-trained models or access to high FPS out-of-domain videos in abundance, but lack ground truth frames in target videos, which are also commonly limited in quantity. Our unsupervised techniques allow us to fine-tune pre-trained models directly on target videos without additional data, and demonstrate significant gain in accuracy in upscaling frame rates of target videos.

First, we consider the scenario where train and test videos are collected with different camera set-ups. We assume we have access to high fps videos collected by hand-held cameras, which is the Adobe-240fps, YouTube-240fps, UCF101 datasets, and consider the Slowflow dataset as our target, a particularly high resolution and high FPS video captured by high speed professional cameras in real life.

Our baseline is a frame interpolation model trained with supervision. Specifically, we consider SuperSloMo and Deep Voxel Flow (DVF) [9]. DVF is another widely-used single-frame interpolation method. We apply our unsupervised fine-tuning directly on the low FPS train split of Slowflow, and evaluate on its test split.

Adobe→Slowflow				
	Loss	PSNR(↑)	SSIM(↑)	IE(↓)
Trivial Copy	N/A	25.00	0.718	14.86
Baseline	PairedGT	32.84	0.887	6.67
	CC	32.35	0.886	6.78
Proposed	CC + PS	33.05	0.890	6.62
	Adobe+YouTube→Slowflow			
Baseline	PairedGT	33.13	0.889	6.63
Proposed	CC + PS	33.20	0.891	6.56

Table 3. Multi-frame interpolation results on Slowflow for frame rate conversion from 30- to 240-FPS, and domain transfer experiments using baselines obtained by pre-training with supervision on Adobe- or Adobe+YouTube-240FPS.

Adobe→Sintel				
	Loss	PSNR(↑)	SSIM(↑)	IE(↓)
Trivial Copy	N/A	22.48	0.714	20.23
Baseline	PairedGT	31.82	0.912	5.61
	CC	30.08	0.872	7.67
Proposed	CC+PS	32.53	0.918	5.21
	Adobe+YouTube→Sintel			
Baseline	PairedGT	33.23	0.928	4.74
Proposed	CC+PS	33.34	0.928	4.71

Table 4. Multi-frame interpolation results on Sintel for frame rate conversion from 24 to 1008 FPS, and domain transfer experiments using baselines obtained by pre-training with supervision on Adobe- or Adobe+YouTube-240fps.

Adobe→Slowflow: Our unsupervised training with cycle consistency alone performs quite closely to the baseline (Super SloMo pre-trained with supervision), achieving PSNR of 32.35 and 32.84, respectively. While a total of 76.7K Adobe-240fps frames are used in supervised pre-training, our unsupervised training is performed with only 3K frames of Slowflow, which indicates the efficiency and robustness of our proposed unsupervised training technique.

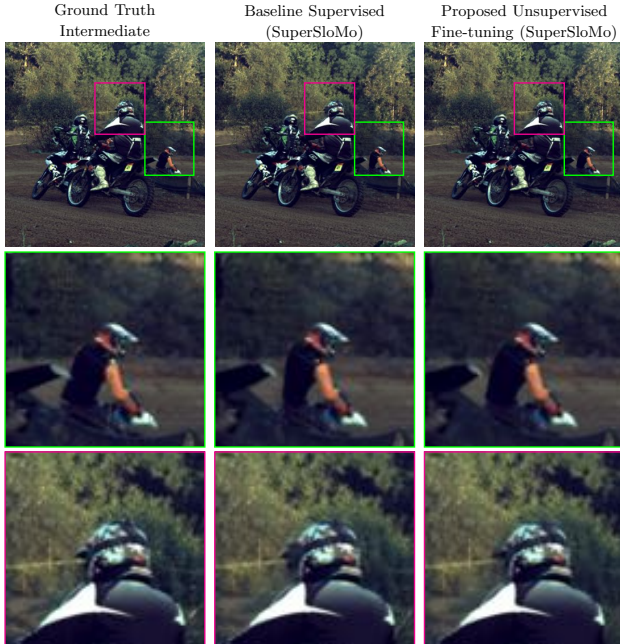


Figure 4. Visual results of a sample from Slowflow dataset. Baseline supervised model is trained with Adobe dataset and proposed unsupervised model is fine-tuned with Slowflow. Improvements seen as the person’s back squeezed in supervised (middle) but preserved in unsupervised (right). On bottom row, although both techniques blur the regions surrounding the helmet, the shape of helmet is preserved in our proposed technique.

Furthermore, fine-tuning the pre-trained model by jointly optimizing to satisfy cycle consistency and to minimize our proposed pseudo supervised loss (CC + PS), we outperform the pre-trained baseline by a large margin, with PSNR of 33.05 vs. 32.84. The PS loss relies on the same pre-trained baseline model, as discussed in Section 3, and regularizes our training process. If used alone, i.e without cycle consistency, it performs at best as good as the baseline pre-trained model, see Section 4.4.1 for more details.

Adobe+YouTube→Slowflow: Here, our baseline model is pre-trained on a larger dataset Adobe+YouTube, total of 372.7K frames, and achieves better accuracy than pre-training on Adobe alone, achieving PSNR 33.13 vs. 32.84, when directly applied on Slowflow test videos. Even with improved pre-trained baseline, we observe consistent benefit with our proposed unsupervised fine-tuning, improving PSNR from 33.13 to 33.20.

Another interesting observation from this study is that it takes an extra 296.K frames from YouTube-240fps to improve PSNR from 32.84 to 33.13 on Slowflow, via pre-training with supervision. We achieve a comparable improvement of PSNR from 32.84 to 33.05 by simply fine-tuning on the target low FPS frames in a completely unsupervised way. Sample interpolation results from this study can be found at Figure 1, where improvements on the bicy-

cle tire and the shoe are highlighted, and at Figure 4, where improvements on the persons’ back and helmet regions are highlighted.

Table 4 present results of unsupervised fine-tuning for domain transfer from Adobe→Sintel and Adobe+YouTube→Sintel for the task of upscaling frame rate from 24- to 1008-fps. Similarly to the Slowflow experiments, in Table 3, the results indicate the utility of our unsupervised techniques in shrinking domain gaps or achieving results that compete with supervised techniques.

	Slowflow			Sintel		
	PSNR(↑)	SSIM(↑)	IE(↓)	PSNR(↑)	SSIM(↑)	IE(↓)
Baseline	30.79	0.84	8.57	29.14	0.84	8.96
Proposed	31.42	0.86	7.99	29.71	0.86	8.23

Table 5. Comparison of supervised training at quarter resolution (baseline) and unsupervised fine-tuning at full resolution (proposed) for frame rate upscaling from 30 to 240 FPS (Slowflow) and 24 to 1008 FPS (Sintel).

UCF101→Slowflow: Table 6 and Figure 5 present results of fine-tuning DVF on Slowflow. We use an off-the-shelf implementation of DVF, pre-trained on UCF101¹. Our unsupervised techniques improve the PSNR from 24.64dB to 28.38dB, demonstrating that our method generalizes well to different interpolation techniques, and is not limited to SuperSloMo.

UCF101→Slowflow using DVF [9]				
	Loss	PSNR(↑)	SSIM(↑)	IE(↓)
Trivial Copy	N/A	24.26	0.698	15.60
Baseline	PairedGT	25.64	0.778	12.77
Proposed	CC + PS	28.38	0.820	9.79

Table 6. Slowflow 30- to 60-FPS conversion. The baseline DVF is pre-trained on UCF-101 with supervision.

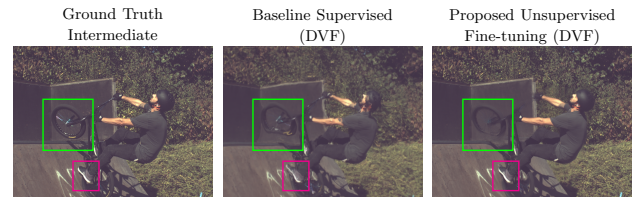


Figure 5. Visual comparison of unsupervised fine-tuning of DVF with supervised DVF pre-trained on UCF-101.

In our second domain transfer setting, we consider the scenario where target and test datasets share similarities in content and style but they are in different resolution. This is a very practical scenario given the scarcity of the high-frame high-resolution videos. Therefore, it is highly desirable to learn from low resolution videos, and be able to interpolate higher resolutions. We establish a low resolution

¹<https://github.com/lxx1991/pytorch-voxel-flow>

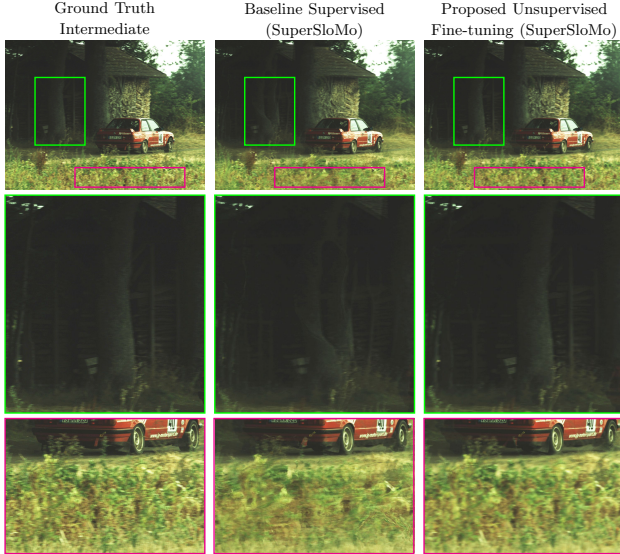


Figure 6. Visual results of a sample from Slowflow dataset. Baseline supervised model is trained with Slowflow dataset in quarter resolution and proposed unsupervised model is fine-tuned with full resolution Slowflow. The tree in the background is deformed, and also deformed in the supervised (middle), while it is predicted well in proposed (right). Bottom row, supervised shows blurriness in the grass, while it is crisper in the proposed.

baseline by training with supervision on 240 fps Slowflow-train dataset, after down-sampling its frames by 4 in each dimension. Our test video is Slowflow-test split at its original resolution. We repeat similar setting for Sintel. Table 5 shows results where our fine-tuning technique on the test domain improves PSNR from 30.79 to 31.42 for Slowflow, and from 29.14 to 29.71 for Sintel. Visual samples from this experiment can be found in Figure 6.

4.4. Ablation Studies

We conduct ablation studies to analyze various design choices of the proposed method.

4.4.1 Optimal CC and PS Weights

Figure 7 presents interpolation results in PSNR for our models trained with a range of PS weight, λ_{rp} , values. We fix CC’s weight, λ_{rc} to 0.8, and vary λ_{rp} in small increments from 0 to 64. When $\lambda_{rp} = 0$, optimization is guided entirely by CC, it achieves PSNR of 32.35 for unsupervised Adobe+YouTube→Slowflow domain transfer. Interpolation accuracy gradually increases, and plateaus approximately after 0.8. Based on this, we select $\lambda_{rp} = 0.8$, and fix its value for all our experiments. At large values of λ_{rp} , the optimization is mostly guided by PS loss, and as such, trained models perform very similarly to the pre-trained model that the PS loss depends on. Figure 7 shows this trend. Optimizing with optimally combined CC and PS

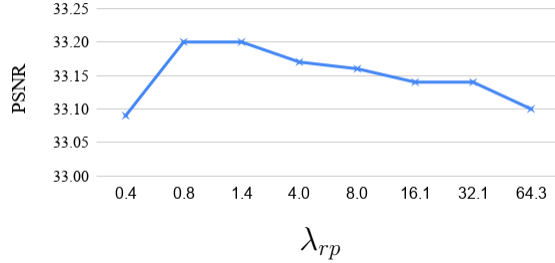


Figure 7. Interpolation accuracy in PSNR versus λ_{rp} (PS weight) used in our proposed joint CC and PS optimization techniques, when applied for Adobe+YouTube→Slowflow unsupervised domain transfer.

losses on the other hand leads to results that are superior to those obtained using either loss alone.

4.4.2 Large Time-step Supervision

We study the effect of using loss terms, such as $\|\mathcal{M}(\mathbf{I}_0, \mathbf{I}_2, t = 0.5) - \mathbf{I}_1\|$ or its variations, defined over a longer time. Table 7 presents Adobe+YouTube→Slowflow fine-tuning with cycle consistency, the loss derived from two step interpolation alone or together with cycle consistency. Optimizing using losses derived from long step interpolation result in worse accuracy than optimizing with cycle consistency. When used with cycle consistency, we also did not find it to lead to notable improvement. We attribute this because the model’s capacity might be spent to solve the harder problem of interpolating large steps, and provide little benefit to the task of synthesizing intermediate frames between consecutive frames.

Adobe+YouTube→Slowflow			
Loss	PSNR(↑)	SSIM(↑)	IE(↓)
CC	32.84	0.887	6.67
Long Step	29.03	0.824	8.98
CC + Long Step	32.24	0.884	6.81

Table 7. Comparison of cycle consistency with objectives derived from longer time step interpolation.

5. Conclusions

We have presented unsupervised learning techniques to synthesize high frame rate videos directly from low frame rate videos by teaching models to satisfy cycle consistency in time. Models trained with our unsupervised techniques are able to synthesize arbitrarily many, high quality and temporally smooth intermediate frames that compete with supervised approaches. We further apply our techniques to reduce domain gaps in video interpolation by fine-tuning pre-trained models on target videos using a pseudo supervised loss term and demonstrate significant gain in accuracy. Our work shows the potential of learning to interpolate

high frame rate videos using only low frame rate videos and opens new avenues to leverage large amounts of low frame rate videos in unsupervised training.

References

- [1] Simon Baker, Daniel Scharstein, JP Lewis, Stefan Roth, Michael J Black, and Richard Szeliski. A database and evaluation methodology for optical flow. *International Journal of Computer Vision*, 92(1):1–31, 2011. [5](#)
- [2] Richard W Brislin. Back-translation for cross-cultural research. *Journal of cross-cultural psychology*, 1(3):185–216, 1970. [2](#)
- [3] Evan Herbst, Steve Seitz, and Simon Baker. Occlusion reasoning for temporal interpolation using optical flow. Technical report, August 2009. [1](#)
- [4] Evan Herbst, Steve Seitz, and Simon Baker. Occlusion reasoning for temporal interpolation using optical flow. *Department of Computer Science and Engineering, University of Washington, Tech. Rep. UW-CSE-09-08-01*, 2009. [2](#)
- [5] Joel Janai, Fatma Guney, Jonas Wulff, Michael J Black, and Andreas Geiger. Slow flow: Exploiting high-speed cameras for accurate and diverse optical flow reference data. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3597–3607, 2017. [5, 6](#)
- [6] Huaizu Jiang, Deqing Sun, Varun Jampani, Ming-Hsuan Yang, Erik Learned-Miller, and Jan Kautz. Super slomo: High quality estimation of multiple intermediate frames for video interpolation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. [1, 2, 3, 4, 5, 6, 10](#)
- [7] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. [5](#)
- [8] Yu-Lun Liu, Yi-Tung Liao, Yen-Yu Lin, and Yung-Yu Chuang. Deep video frame interpolation using cyclic frame generation. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2019. [2, 3](#)
- [9] Ziwei Liu, Raymond A Yeh, Xiaoou Tang, Yiming Liu, and Aseem Agarwala. Video frame synthesis using deep voxel flow. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4463–4471, 2017. [2, 6, 7](#)
- [10] Gucan Long, Laurent Kneip, Jose M. Alvarez, Hongdong Li, Xiaohu Zhang, and Qifeng Yu. Learning image matching by simply watching video. In *ECCV*, 2016. [1](#)
- [11] Gucan Long, Laurent Kneip, Jose M Alvarez, Hongdong Li, Xiaohu Zhang, and Qifeng Yu. Learning image matching by simply watching video. In *European Conference on Computer Vision (ECCV)*, 2016. [1](#)
- [12] Dhruv Mahajan, Fu-Chung Huang, Wojciech Matusik, Ravi Ramamoorthi, and Peter Belhumeur. Moving gradients: a path-based method for plausible image interpolation. In *ACM Transactions on Graphics (TOG)*, volume 28, page 42, 2009. [2](#)
- [13] Junhua Mao, Jonathan Huang, Alexander Toshev, Oana Camburu, Alan L Yuille, and Kevin Murphy. Generation and comprehension of unambiguous object descriptions. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. [2](#)
- [14] Simon Niklaus and Feng Liu. Context-aware synthesis for video frame interpolation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. [1](#)
- [15] Simon Niklaus, Long Mai, and Feng Liu. Video frame interpolation via adaptive convolution. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. [2](#)
- [16] Simon Niklaus, Long Mai, and Feng Liu. Video frame interpolation via adaptive separable convolution. In *IEEE International Conference on Computer Vision (ICCV)*, 2017. [1, 2](#)
- [17] 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 *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 718–733, 2018. [5, 6](#)
- [18] Hidetoshi Shimodaira. Improving predictive inference under covariate shift by weighting the log-likelihood function. *Journal of statistical planning and inference*, 90(2):227–244, 2000. [2](#)
- [19] Khurram Soomro, Amir Roshan Zamir, and M Shah. A dataset of 101 human action classes from videos in the wild. *Center for Research in Computer Vision*, 2012. [5, 6](#)
- [20] Shuo Chen Su, Mauricio Delbracio, Jue Wang, Guillermo Sapiro, Wolfgang Heidrich, and Oliver Wang. Deep video deblurring for hand-held cameras. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1279–1288, 2017. [5, 6](#)
- [21] Richard Szeliski. Prediction error as a quality metric for motion and stereo. In *ICCV*, 1999. [1](#)
- [22] Christopher Zach, Manfred Klopschitz, and Marc Pollefeys. Disambiguating visual relations using loop constraints. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1426–1433. IEEE, 2010. [2](#)
- [23] Tinghui Zhou, Yong Jae Lee, Stella X Yu, and Alyosha A Efros. Flowweb: Joint image set alignment by weaving consistent, pixel-wise correspondences. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1191–1200, 2015. [2](#)
- [24] Tinghui Zhou, Philipp Krahenbuhl, Mathieu Aubry, Qixing Huang, and Alexei A Efros. Learning dense correspondence via 3d-guided cycle consistency. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. [2](#)
- [25] Xiaowei Zhou, Menglong Zhu, and Kostas Daniilidis. Multi-image matching via fast alternating minimization. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4032–4040, 2015. [2](#)
- [26] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *IEEE International Conference on Computer Vision (ICCV)*, 2017. [2, 3](#)
- [27] C Lawrence Zitnick, Sing Bing Kang, Matthew Uyttendaele, Simon Winder, and Richard Szeliski. High-quality video view interpolation using a layered representation. In *ACM Transactions on Graphics (TOG)*, volume 23, pages 600–608. ACM, 2004. [1](#)

Appendix

A. Insensitivity to Randomness

We train our models three times to account for random weight initialisation or random data augmentation. Table 8 presents the mean and standard deviation of domain adaptation results for Adobe→Slowflow and Adobe+YouTube→Slowflow. Results indicate our models’ insensitivity to randomness, as shown by the margins of improvements in PSNR from 32.84 to 33.05 for Adobe, or from 33.13 to 33.20 for Adobe+YouTube. Table 8 also shows fine-tuning with PS loss alone leads to results that are similar to the Baseline.

B. Interpolation Result vs Intermediate Time

Figure 8 presents mean PSNR score at each of the 41 time-points for Adobe→Sintel domain adaptation using (a) Super SloMo [6] pre-trained with supervision (Baseline), (b) unsupervised fine-tuning with cycle consistency loss alone (CC), and (c) unsupervised fine-tuning with cycle consistency and pseudo supervised losses (CC+PS).

For all models, interpolation accuracy decreases as time-points move away from $t = 0$ or $t = 1$. Compared to the baseline, our CC-based fine-tuning performs better at the end points (close to $t = 0$ or $t = 1$), and worse at midway points. On the other hand, our CC+PS-based unsupervised fine-tuning achieves the best of both CC and Baseline, performing better than both CC and Baseline at all time points.

Adobe→Slowflow				
	Loss	PSNR	SSIM	IE
Baseline	PairedGT	32.84	0.887	6.67
	CC	32.33±0.028	0.886±0.000	6.78±0.021
Proposed	PS	32.88±0.006	0.887±0.000	6.74±0.006
	CC + PS	33.05±0.006	0.890±0.000	6.62±0.000
Adobe+YouTube→Slowflow				
Baseline	PairedGT	33.13	0.889	6.63
	PS	33.14±0.006	0.889±0.000	6.63±0.006
Proposed	CC	32.33±0.028	0.886±0.000	6.78±0.021
Proposed	CC + PS	33.20±0.006	0.891±0.001	6.57±0.010

Table 8. Mean and Standard deviation of PSNR, SSIM, and IE for domain adaptation of upscaling frame rate from 30- to 240-fps for Adobe→or Adobe+YouTube→Slowflow. CC refers to cycle consistency, and PS pseudo supervised loss.

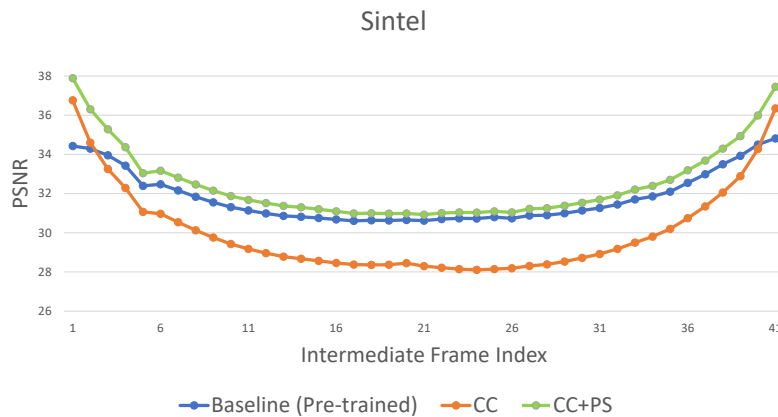


Figure 8. Mean PSNR score at each of the 41 time points for Adobe→Sintel domain adaptation using (a) Super SloMo [6] pre-trained with supervision (Baseline), (b) unsupervised fine-tuning with cycle consistency loss alone (CC), and (c) unsupervised fine-tuning with cycle consistency and pseudo supervised losses (CC+PS).