Softmax Splatting for Video Frame Interpolation

Simon Niklaus
Portland State University
sniklaus@pdx.edu

Feng Liu
Portland State University
fliu@cs.pdx.edu

SepConv - $\mathcal{L}_F$ [39] | ToFlow [51] | CyclicGen [28] | CtxSyn - $\mathcal{L}_F$ [37] | DAIN [3] | Ours - $\mathcal{L}_F$

Figure 1: A difficult example for video frame interpolation. Our approach produces a high-quality result in spite of the delicate flamingo leg that is subject to large motion. *Please see the arXiv version to be able to view this figure as a video.*

Abstract

Differentiable image sampling in the form of backward warping has seen broad adoption in tasks like depth estimation and optical flow prediction. In contrast, how to perform forward warping has seen less attention, partly due to additional challenges such as resolving the conflict of mapping multiple pixels to the same target location in a differentiable way. We propose softmax splatting to address this paradigm shift and show its effectiveness on the application of frame interpolation. Specifically, given two input frames, we forward-warp the frames and their feature pyramid representations based on an optical flow estimate using softmax splatting. In doing so, the softmax splatting seamlessly handles cases where multiple source pixels map to the same target location. We then use a synthesis network to predict the interpolation result from the warped representations. Our softmax splatting allows us to not only interpolate frames at an arbitrary time but also to fine tune the feature pyramid and the optical flow. We show that our synthesis approach, empowered by softmax splatting, achieves new state-of-the-art results for video frame interpolation.

1. Introduction

Video frame interpolation is a classic problem in computer vision with many practical applications. It can, for example, be used to convert the frame rate of a video to the refresh rate of the monitor that is used for playback, which is beneficial for human perception [24, 25]. Frame interpolation can also help in video editing tasks, such as temporally consistent color modifications, by propagating the changes that were made in a few keyframes to the remaining frames [33]. Frame interpolation can also support interframe compression for videos [49], serve as an auxiliary task for optical flow estimation [30, 50], or generate training data to learn how to synthesize motion blur [6]. While these applications employ frame interpolation in the temporal domain, it can also be used to synthesize novel views in space by interpolating between given viewpoints [23].

Approaches for video frame interpolation can be categorized as flow-based, kernel-based, and phase-based. We adopt the flow-based paradigm since it has proven to work well in quantitative benchmarks [2]. One common approach for these methods is to estimate the optical flow $F_{t\to0}$ and $F_{t\to1}$ between two input frames I_0 and I_1 from the perspective of the frame I_t that is ought to be synthesized. The interpolation result can then be obtained by backward warping I_0 according to $F_{t\to0}$ and I_1 according to $F_{t\to1}$ [20]. While it is intuitive, this approach makes it difficult to use an off-the-shelf optical flow estimator and prevents synthesizing frames at an arbitrary t in a natural manner. To address these concerns, Jiang *et al*. [22] and Bao *et al*. [3] approximate $F_{t\to0}$ and $F_{t\to1}$ from $F_{0\to1}$ and $F_{1\to0}$.

Different from backward warping, Niklaus *et al*. [37] directly forward-warp I_0 according to $t \cdot F_{0\to1}$ and I_1 according to $(1-t) \cdot F_{1\to0}$, which avoids having to approximate $F_{t\to0}$ and $F_{t\to1}$. Another aspect of their approach is to warp not only the images but also the corresponding context information, which a synthesis network can use to make better predictions. However, their forward warping uses the equivalent of z-buffering in order to handle cases where multiple source pixels map to the same target location. It is thus unclear how to fully differentiate this operation due to the z-buffering [36]. We propose softmax splatting to address this limitation, which allows us to jointly supervise all inputs to the forward warping. As a consequence, we are able to extend the idea of warping a generic context map to learning and warping a task-specific feature pyramid. Furthermore, we are able to supervise not only the optical flow estimator but also the metric that weights the importance of different pixels when they are warped to the same location. This approach, which is enabled by our proposed softmax splatting, achieves new state-of-the-art results and ranks first in the Middlebury benchmark for frame interpolation.

In short, we propose softmax splatting to perform differentiable forward warping and show its effectiveness on the application of frame interpolation. An interesting research question that softmax splatting addresses is how to handle different source pixels that map to the same target location in a differentiable way. Softmax splatting enables us to train and use task-specific feature pyramids for image synthesis. Furthermore, softmax splatting not only allows us to fine-tune an off-the-shelf optical flow estimator for video frame interpolation, it also enables us to supervise the metric that is used to disambiguate cases where multiple source pixels map to the same forward-warped target location.

2. Related Work

With the introduction of spatial transformer networks, Jaderberg *et al*. [20] proposed differentiable image sampling. Since then, this technique has found broad adoption in the form of backward warping to synthesize an image I_A from an image I_B given a correspondence $F_{A\to B}$ for each pixel in I_A to its location in I_B. Prominent examples where this approach has been used include unsupervised depth estimation [13, 31, 54], unsupervised optical flow prediction [32, 47, 52], optical flow prediction [18, 42, 45], novel view synthesis [8, 27, 55], video frame interpolation [3, 22, 28, 29], and video enhancement [7, 46, 51].

In contrast, performing forward warping to synthesize I_B from I_A based on $F_{A\to B}$ has seen less adoption with deep learning, partly due to additional challenges such as multiple source pixels in I_A possibly being mapped to the same target location in I_B. For optical flow estimation, Wang *et al*. [47] forward-warp an image filled with ones to obtain an occlusion mask. However, they sum up contributions of all the pixels that are mapped to the same output pixel without a mechanism to remove possible outliers, which limits the applicability of this technique for image synthesis. For frame interpolation, Niklaus *et al*. [37] use the equivalent of z-buffering which is well motivated but not differentiable [36]. Bao *et al*. [3] linearly weight the optical flow according to a depth estimate as an approach for dealing with multiple source pixels mapping to the same target location. However, adding a bias to the depth estimation affects the result of this linearly weighted warping and leads to negative side effects. In contrast, our proposed softmax splatting is not subject to any of these concerns.

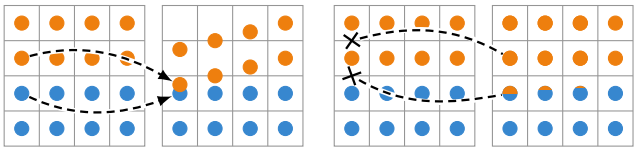
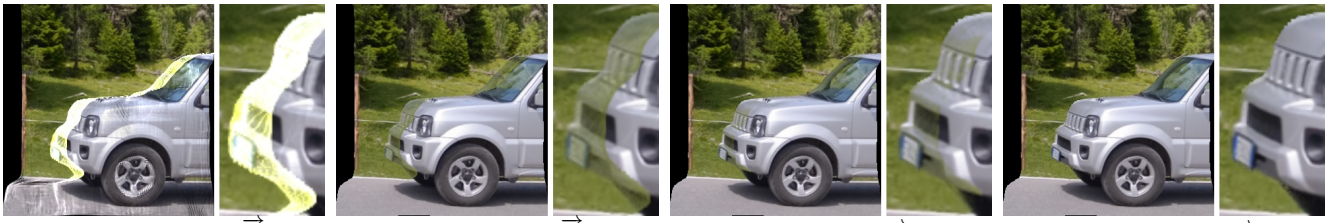


Figure 2: Splatting versus sampling, the blue pixels remain static while the red ones move down in a shearing manner. With splatting, the output is subject to holes and multiple source pixels can map to the same target pixel. On the upside, splatting makes it possible to scale the transform.

We demonstrate the effectiveness of our proposed softmax splatting on the example of frame interpolation. Research on frame interpolation has seen a recent resurgence, with multiple papers proposing kernel-based [3, 4, 38, 39], flow-based [3, 4, 22, 28, 29, 37, 41, 43, 51], and phase-based [34, 35] approaches. We base our approach on the one from Niklaus *et al*. [37] who estimate optical flow between two input images in both directions, extract generic contextual information from the input images using pre-trained filters, forward-warp the images together with their context maps according to optical flow, and finally employ a synthesis network to obtain the interpolation result. Enabled by softmax splatting, we extend their framework to warping task-specific feature pyramids for image synthesis in an end-to-end manner. This includes fine-tuning the off-the-shelf optical flow estimator for video frame interpolation and supervising the metric that is used to disambiguate cases where multiple pixels map to the same location.

For image synthesis, Niklaus *et al*. [37] warp context information from a pre-trained feature extractor that a synthesis network can use to make better predictions. Bao *et al*. [3] subsequently refined this approach through end-to-end supervision of the feature extractor. In contrast, we extract and warp feature pyramids which allows the synthesis network to make use of a multi-scale representation for better interpolation results. Our use of feature pyramids for image synthesis is inspired by recent work on video analysis. For video semantic segmentation, Gadde *et al*. [12] warp features that were obtained when processing the preceding



summation splatting $\overrightarrow{\Sigma}$ | average splatting $\overrightarrow{\Phi}$ | linear splatting $\overrightarrow{*}$ | softmax splatting $\overrightarrow{\sigma}$

Figure 3: Given two images I_0 and I_1 as well as an optical flow estimate $F_{0\to1}$, this figure shows an example of warping I_0 to I_t according to $F_{0\to t} = t \cdot F_{0\to1}$ with four different forward warping approaches. The summation warping $\overrightarrow{\Sigma}$ handles cases where multiple pixels in I_0 map to the same target location in I_t by taking their sum, which leads to brightness inconsistencies. The average warping $\overrightarrow{\Phi}$ takes their mean instead and is able to maintain the overall appearance of I_0 but blends overlapping regions. The linear splatting $\overrightarrow{*}$ weights the pixels in I_0 before warping them but still fails to clearly separate the front of the car from the grass in the background. In contrast, our proposed softmax splatting $\overrightarrow{\sigma}$ shows the expected behavior with the car correctly occluding the background. *Please see the arXiv version to be able to view this figure as a video.*

frame in order to support the segmentation of the current frame. For optical flow estimation, Hui *et al*. [18] and Sun *et al*. [45] extend this idea of warping features and employ it across multiple scales in the form of feature pyramids. These approaches do not target image synthesis though.

Temporal consistency is a common concern when synthesizing images in time [1, 16, 17, 26]. For frame interpolation, Jiang *et al*. [22] collect a specialized training dataset with frame-nonuples and supervise their network on seven intermediate frames at a time in order to ensure temporally consistent results. In the same vein, Liu *et al*. [28] and Reda *et al*. [43] utilize cycle consistency to better supervise their model. In comparison, our proposed softmax splatting leads to temporally consistent results without requiring a specialized training dataset or cycle-consistent training.

3. Softmax Splatting for Frame Interpolation

Given two frames I_0 and I_1, frame interpolation aims to synthesize an intermediate frame I_t where $t \in (0, 1)$ defines the desired temporal position. To address this problem, we first use an off-the-shelf optical flow method to estimate the optical flow $F_{0\to1}$ and $F_{1\to0}$ between the input frames in both directions. We then use forward warping in the form of softmax splatting $\overrightarrow{\sigma}$ to warp I_0 according to $F_{0\to t} = t \cdot F_{0\to1}$ and I_1 according to $F_{1\to t} = (1-t) \cdot F_{1\to0}$ as follows.

$$I_t \approx \overrightarrow{\sigma}\left(I_0, F_{0\to t}\right) = \overrightarrow{\sigma}\left(I_0, t \cdot F_{0\to1}\right) \tag{1}$$

$$I_t \approx \overrightarrow{\sigma}\left(I_1, F_{1\to t}\right) = \overrightarrow{\sigma}\left(I_1, (1-t) \cdot F_{1\to0}\right) \tag{2}$$

This is in contrast to backward warping $\overleftarrow{\omega}$, which would require $F_{t\to0}$ and $F_{t\to1}$ but computing this t-centric optical flow from $F_{0\to1}$ and $F_{1\to0}$ is complicated and subject to approximations [3]. We then combine these intermediate results to obtain I_t using a synthesis network. More specifically, we not only warp the input frame in color- but also feature-space across multiple resolutions which enables the synthesis network to make better predictions.

We subsequently first introduce forward warping via softmax splatting and then show how it enables us to establish new state-of-the-art results for frame interpolation.

3.1. Forward Warping via Softmax Splatting

Backward warping is a common technique that has found broad adoption in tasks like unsupervised depth estimation or optical flow prediction [20]. It is well supported by many deep learning frameworks. In contrast, forward warping an image I_0 to I_t according to $F_{0\to t}$ is not supported by these frameworks. We attribute this lack of support to the fact that there is no definitive way of performing forward warping. Forward warping is subject to multiple pixels in I_0 being able to possibly map to the same target pixel in I_t and there are various possibilities to address this ambiguity. We thus subsequently introduce common approaches to handle this mapping-ambiguity and discuss their limitations. We then propose softmax splatting which addresses these inherent limitations. Please note that we use the terms "forward warping" and "splatting" interchangeably.

Summation splatting. A straightforward approach of handling the aforementioned mapping-ambiguity is to sum all contributions. We define this summation splatting $\overrightarrow{\Sigma}$ as follows, where I_t^{Σ} is the sum of all contributions from I_0 to I_t according to $F_{0\to t}$ subject to the bilinear kernel b.

$$\text{let } \boldsymbol{u} = \boldsymbol{p} - \left(\boldsymbol{q} + F_{0\to t}[\boldsymbol{q}]\right) \tag{3}$$

$$b(\boldsymbol{u}) = \max(0, 1 - |\boldsymbol{u}_x|) \cdot \max(0, 1 - |\boldsymbol{u}_y|) \tag{4}$$

$$I_t^{\Sigma}[\boldsymbol{p}] = \sum_{\forall \boldsymbol{q} \in I_0} b(\boldsymbol{u}) \cdot I_0[\boldsymbol{q}] \tag{5}$$

$$\overrightarrow{\Sigma}\left(I_0, F_{0\to t}\right) = I_t^{\Sigma} \tag{6}$$

As shown in Figure 3, this summation splatting leads to brightness inconsistencies in overlapping regions like the front of the car. Furthermore, the bilinear kernel b leads to

pixels in I_t that only receive partial contributions from the pixels in I_0 which yet again leads to brightness inconsistencies like on the street. However, we use this summation splatting as the basis of all subsequent forward warping approaches. The relevant derivatives are as follows.

$$\text{let } \boldsymbol{u} = \boldsymbol{p} - \big(\boldsymbol{q} + F_{0\to t}[\boldsymbol{q}]\big) \tag{7}$$

$$\frac{\partial I_t^{\Sigma}[\boldsymbol{p}]}{\partial I_0[\boldsymbol{q}]} = b(\boldsymbol{u}) \tag{8}$$

$$\frac{\partial I_t^{\Sigma}[\boldsymbol{p}]}{\partial F_{0\to t}^{x}[\boldsymbol{q}]} = \frac{\partial b(\boldsymbol{u})}{\partial F_{0\to t}^{x}} \cdot I_0[\boldsymbol{q}] \tag{9}$$

$$\frac{\partial b(\boldsymbol{u})}{\partial F_{0\to t}^{x}} = \max(0, 1 - |\boldsymbol{u}_y|) \cdot \begin{cases} 0, & \text{if } |\boldsymbol{u}_x| \geq 1 \\ -\text{sgn}(\boldsymbol{u}_x), & \text{else} \end{cases} \tag{10}$$

and analogous for the y component of $F_{0\to t}$. It is not easy to obtain these through automatic differentiation since few frameworks support the underlying `scatter_nd` function that is necessary to implement this operator. We hence provide a PyTorch reference implementation[1] of this summation splatting $\overrightarrow{\Sigma}$ which is written in CUDA for efficiency.

Average splatting. To address the brightness inconsistencies that occur with summation splatting, we need to normalize I_t^{Σ}. To do so, we can reuse the definition of $\overrightarrow{\Sigma}$ and determine average splatting $\overrightarrow{\Phi}$ as follows.

$$\overrightarrow{\Phi}\left(I_0, F_{0\to t}\right) = \frac{\overrightarrow{\Sigma}\left(I_0, F_{0\to t}\right)}{\overrightarrow{\Sigma}\left(\mathbf{1}, F_{0\to t}\right)} \tag{11}$$

As shown in Figure 3, this approach handles the brightness inconsistencies and maintains the appearance of I_0. However, this technique averages overlapping regions like at the front of the car with the grass in the background.

Linear splatting. In an effort to better separate overlapping regions, one could try to linearly weight I_0 by an importance mask Z and define linear splatting $\overrightarrow{*}$ as follows.

$$\overrightarrow{*}\left(I_0, F_{0\to t}\right) = \frac{\overrightarrow{\Sigma}\left(Z \cdot I_0, F_{0\to t}\right)}{\overrightarrow{\Sigma}\left(Z, F_{0\to t}\right)} \tag{12}$$

where Z could, for example, relate to the depth of each pixel [3]. As shown in Figure 3, this approach can better separate the front of the car from the grass in the background. It is not invariant to translations with respect to Z though. If Z represents the inverse depth then there will be a clear separation if the car is at $Z = 1/1$ and the background is at $Z = 1/10$. But, if the car is at $Z = 1/101$ and the background is at $Z = 1/110$ then they will be averaged again despite being equally far apart in terms of depth.

[1] http://sniklaus.com/softsplat

Softmax splatting. To clearly separate overlapping regions according to an importance mask Z with translational invariance, we propose softmax splatting $\overrightarrow{\sigma}$ as follows.

$$\overrightarrow{\sigma}\left(I_0, F_{0\to t}\right) = \frac{\overrightarrow{\Sigma}\left(\exp(Z) \cdot I_0, F_{0\to t}\right)}{\overrightarrow{\Sigma}\left(\exp(Z), F_{0\to t}\right)} \tag{13}$$

where Z could, for example, relate to the depth of each pixel [3]. As shown in Figure 3, this approach is able to clearly separate the front of the car from the background without any remaining traces of grass. Furthermore, it shares resemblance to the softmax function. It is hence invariant to translations β with respect to Z, which is a particularly important property when mapping multiple pixels to the same location. If Z represents depth, then the car and the background in Figure 3 are treated equally whether the car is at $Z = 1$ and the background is at $Z = 10$ or the car is at $Z = 101$ and the background is at $Z = 110$. It is not invariant to scale though and multiplying Z by α will affect how well overlapping regions will be separated. A small α yields averaging whereas a large α yields z-buffering. This parameter can be learned via end-to-end training.

Importance metric. We use Z to weight pixels in I_0 in order to resolve cases where multiple pixels from I_0 map to the same target pixel in I_t. This Z could, for example, represent depth [3]. However, obtaining such a depth estimate is computationally expensive and inherently challenging which makes it prone to inaccuracies. We thus use brightness constancy as a measure of occlusion [2], which can be obtained via backward warping $\overleftarrow{\omega}$ as follows.

$$Z = \alpha \cdot \left\| I_0 - \overleftarrow{\omega}\left(I_1, F_{0\to 1}\right) \right\|_1 \tag{14}$$

Since our proposed softmax splatting is fully differentiable, we can not only learn α (initially set to -1) but also use a small neural network υ to further refine this metric.

$$Z = \upsilon\big(I_0, -\left\| I_0 - \overleftarrow{\omega}\left(I_1, F_{0\to 1}\right) \right\|_1\big) \tag{15}$$

One could also obtain Z directly from $\upsilon(I_0)$ but we were unable to make this υ converge. Lastly, when applying softmax splatting to tasks different from frame interpolation, the importance metric may be adjusted accordingly.

Efficiency. PyTorch's backward warping requires 1.1 ms to warp a full-HD image on a Titan X with a synthetic flow drawn from $\mathcal{N}(0, 10^2)$. In contrast, our implementation of softmax splatting requires 3.7 ms since we need to compute Z and handle race conditions during warping.

3.2. Feature Pyramids for Image Synthesis

We adopt the video frame interpolation pipeline from Niklaus *et al.* [37] who, given two input frames I_0 and I_1, first estimate the inter-frame motion $F_{0\to 1}$ and $F_{1\to 0}$ using an off-the-shelf optical flow method. They then extract

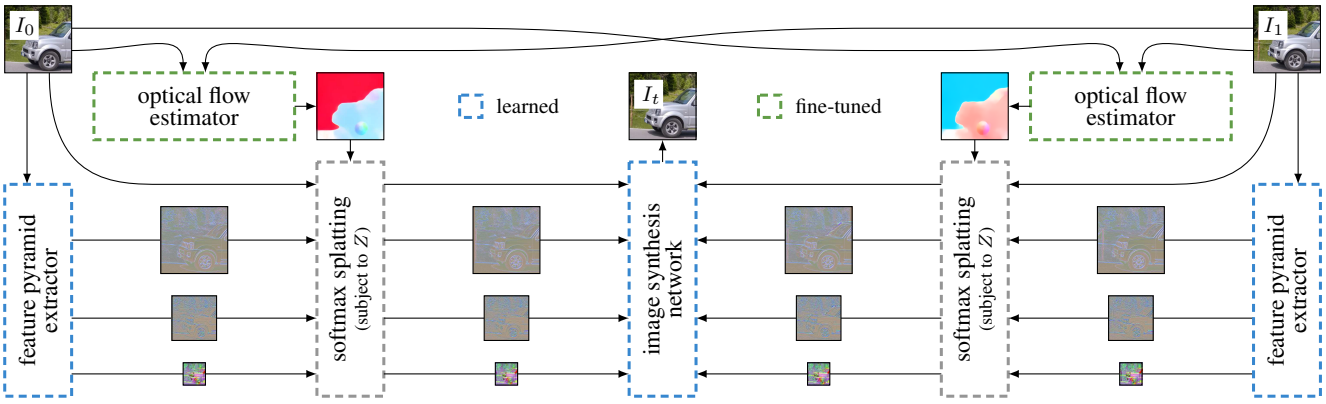


Figure 4: An overview of our frame interpolation framework. Given two input frames I_0 and I_1, we first estimate the bidirectional optical flow between them. We then extract their feature pyramids and forward-warp them together with the input frames to the target temporal position $t \in (0, 1)$ according to the optical flow. Using softmax splatting enables end-to-end training and thus allows the feature pyramid extractor to learn to gather features that are important for image synthesis. The warped input frames and feature pyramids are then fed to a synthesis network to generate the interpolation result I_t.

generic contextual information from the input images using a pre-defined filter ψ and forward-warp $\overrightarrow{\omega}$ the images together with their context maps according to $t \cdot F_{0\to1} = F_{0\to t}$ and $(1-t) \cdot F_{1\to0} = F_{1\to t}$, before employing a synthesis network ϕ to obtain the interpolation result I_t.

$$I_t = \phi\Big(\overrightarrow{\omega}\big(\{I_0, \psi(I_0)\}, F_{0\to t}\big), \overrightarrow{\omega}\big(\{I_1, \psi(I_1)\}, F_{1\to t}\big)\Big)$$

This approach is conceptually simple and has been proven to work well. However, Niklaus *et al*. were not able to supervise the context extractor ψ and instead used `conv1` of ResNet-18 [15] due to the limitations of their forward warping $\overrightarrow{\omega}$ approach. This limitation makes it an ideal candidate to show the benefits of our proposed softmax splatting.

Our proposed softmax splatting allows us to supervise ψ, enabling it to learn to extract features that are important for image synthesis. Furthermore, we extend this idea by extracting and warping features at multiple scales in the form of feature pyramids. This allows the synthesis network ϕ to further improve its predictions. Please see Figure 4 for an overview of our video frame interpolation framework. We will subsequently discuss its individual components.

Optical flow estimator. We use an off-the-shelf optical flow method to make use of the ongoing achievements in research on correspondence estimation. Specifically, we use PWC-Net [45] and show that FlowNet2 [19] and LiteFlowNet [18] perform equally well within our evaluation. In accordance with the findings of Xue *et al*. [51], we additionally fine-tune PWC-Net for frame interpolation.

Feature pyramid extractor. The architecture of our feature pyramid extractor is shown in Figure 5. Our proposed softmax splatting enables us to supervise this feature pyramid extractor in an end-to-end manner, allowing it to learn to extract features that are useful for the subsequent image

type	features	kernel	stride	padding
Input	–	–	–	–
Conv2d	$3 \to 32$	3×3	1×1	1×1
PReLU	–	–	–	–
Conv2d	$32 \to 32$	3×3	1×1	1×1
PReLU	–	–	–	–
Conv2d	$32 \to 64$	3×3	2×2	1×1
PReLU	–	–	–	–
Conv2d	$64 \to 64$	3×3	1×1	1×1
PReLU	–	–	–	–
Conv2d	$64 \to 96$	3×3	2×2	1×1
PReLU	–	–	–	–
Conv2d	$96 \to 96$	3×3	1×1	1×1
PReLU	–	–	–	–

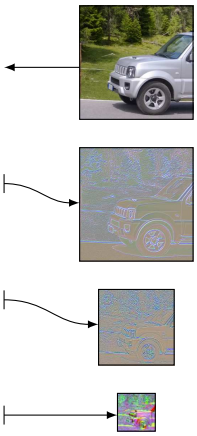


Figure 5: The architecture of our feature pyramid extractor. The feature visualization was obtained using PCA and is only serving an aesthetic purpose. See our evaluation for an analysis of the feature pyramid space for image synthesis.

synthesis. As shown in our evaluation, this approach leads to significant improvements in the quality of the interpolation result. We also show that the interpolation quality degrades if we use fewer levels of features.

Image synthesis network. The synthesis network generates the interpolation result guided by the warped input images and their corresponding feature pyramids. We employ a GridNet [11] architecture with three rows and six columns for this task. To avoid checkerboard artifacts [40], we adopt the modifications proposed by Niklaus *et al*. [37]. The GridNet architecture is a generalization of U-Nets and is thus well suited for the task of image synthesis.

Importance metric. Our proposed softmax splatting uses an importance metric Z which is used to resolve cases where multiple pixels forward-warp to the same target lo-

cation. We use brightness constancy to compute this metric as outlined in Section 3.1. Furthermore, we refine this occlusion estimate using a small U-Net consisting of three levels, which is trained end-to-end with the feature pyramid extractor and the image synthesis network.

Training. We adopt the training from Niklaus *et al*. [37]. We thus train two versions of our model to account for the perception-distortion tradeoff [5], one trained on color loss $\mathcal{L}_{Lap}$ which performs well in standard benchmarks and one trained on perceptual loss $\mathcal{L}_F$ which retains more details in difficult cases. However, instead of using a proprietary training dataset, we use frame-triples from the training portion of the publicly available Vimeo-90k dataset [51].

Efficiency. With an Nvidia Titan X, we are able to synthesize a 720p frame in 0.357 seconds as well as a 1080p frame in 0.807 seconds. The parameters of our entire pipeline amount to 31 megabytes when stored.

4. Experiments

We evaluate our method, which utilizes softmax splatting to improve an existing frame interpolation approach, and compare it to state-of-the-art methods quantitatively and qualitatively on publicly available datasets. To support examining the visual quality of the frame interpolation results, we additionally provide a supplementary video.

Methods. We compare our approach to several state-of-the-art frame interpolation methods for which open source implementations from the respective authors are publicly available. This includes SepConv [39], ToFlow [51], CyclicGen [28], and DAIN [3]. We also include the closed source CtxSyn [37] approach wherever possible.

Datasets. We perform the quantitative evaluation on common datasets for frame interpolation. This includes the Vimeo-90k [51] test dataset as well as the samples from the Middlebury benchmark with publicly-available ground truth interpolation results [2]. When comparing our approach to other state-of-the-art methods, we additionally incorporate samples from UCF101 [29, 44] and Xiph[2].

Metrics. We follow recent work on frame interpolation and use PSNR and SSIM [48] for all quantitative comparisons. We additionally incorporate the LPIPS [53] metric which strives to measure perceptual similarity. While higher values indicate better results in terms of PSNR and SSIM, lower values indicate better results with the LPIPS metric.

4.1. Ablation Experiments

We show the effectiveness of our proposed softmax splatting by improving the context-aware frame interpolation from Niklaus *et al*. [37]. We thus not only need to compare softmax splatting to alternative ways of performing differentiable forward warping, we also need to analyze the improvements that softmax splatting enabled.

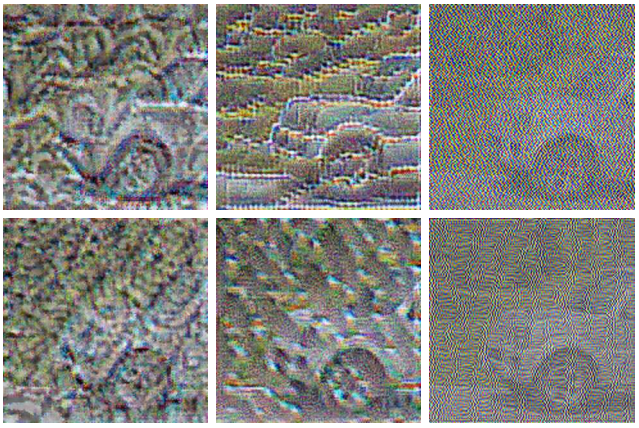
[2] https://media.xiph.org/video/derf

	Vimeo-90k [51]			Middlebury [2]		
	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
CtxSyn	34.39	0.961	0.024	36.93	0.964	0.016
Ours - CtxSyn-like	34.85	0.963	0.025	37.02	0.966	0.018
Ours - summation splatting	35.09	0.965	0.024	37.47	0.968	0.018
Ours - average splatting	35.29	0.966	0.023	37.53	0.969	0.017
Ours - linear splatting	35.26	0.966	0.024	37.73	0.968	0.017
Ours - softmax splatting	35.54	0.967	0.024	37.81	0.969	0.017
Ours - pre-defined Z	35.54	0.967	0.024	37.81	0.969	0.017
Ours - fine-tuned Z	35.59	0.967	0.024	37.97	0.970	0.017
Ours - 1 feature level	35.08	0.965	0.024	37.32	0.968	0.018
Ours - 2 feature levels	35.37	0.966	0.024	37.79	0.970	0.016
Ours - 3 feature levels	35.59	0.967	0.024	37.97	0.970	0.017
Ours - 4 feature levels	35.69	0.968	0.023	37.99	0.971	0.016
Ours - FlowNet2	35.83	0.969	0.022	37.67	0.970	0.016
Ours - LiteFlowNet	35.59	0.968	0.024	37.83	0.970	0.017
Ours - PWC-Net	35.59	0.967	0.024	37.97	0.970	0.017
Ours - PWC-Net-ft	36.10	0.970	0.021	38.42	0.971	0.016
Ours - $\mathcal{L}_{Lap}$	36.10	0.970	0.021	38.42	0.971	0.016
Ours - $\mathcal{L}_F$	35.48	0.964	0.013	37.55	0.965	0.008

Table 1: Ablation experiments to quantitatively analyze the effect of the different components of our approach.

Context-aware synthesis. Since we adopt the framework of Niklaus *et al*. [37], we first need to verify that we can match their performance. We thus replace our feature pyramid extractor with the `conv1` layer of ResNet-18 [15] and we do not fine-tune the utilized PWC-Net for frame interpolation. This leaves the training dataset as well as the softmax splatting as the only significant differences. As shown in Table 1 (first section), our implementation performs slightly better in terms of PSNR on the Middlebury examples. It is significantly better in terms of PSNR on the Vimeo-90k test data though, but this is to be expected since we supervise on the Vimeo-90k training data. We can thus confirm that the basis for our approach truthfully replicates CtxSyn.

Softmax splatting for frame interpolation. We discussed various ways of performing differentiable forward warping in Section 3.1 and outlined their limitations. We then proposed softmax splatting to address these limitations. To analyze the effectiveness of softmax splatting, we train four versions of our approach, each one using a different forward warping technique. As shown in Table 1 (second section), summation splatting performs worst and softmax splatting performs best in terms of PSNR. Notice that the PSNR of average splatting is better than linear splatting on the Mid-



PWC-Net LiteFlowNet Ours

Figure 6: Feature response visualization for different task-specific feature pyramids on the image from Figure 3 using the visualization technique from Erhan *et al*. [10].

dlebury examples but worse on the Vimeo-90k test data. We attribute this erratic behavior of linear splatting to its lack of translational invariance. These findings support the motivations behind our proposed softmax splatting.

Importance metric. Our proposed softmax splatting uses an importance metric Z to resolve cases where multiple pixels forward-warp to the same target location. We use brightness constancy [2] to obtain this metric. Since softmax splatting is fully differentiable, we can use a small U-Net to fine-tune this metric which, as shown in Table 1 (third section), leads to slight improvements in terms of PSNR. This demonstrates that softmax splatting can effectively supervise Z and that brightness constancy works well as the importance metric for video frame interpolation.

Feature pyramids for image synthesis. Softmax splatting enables us to synthesize images from warped feature pyramids, effectively extending the interpolation framework from Niklaus *et al*. [37]. In doing so, the softmax splatting enables end-to-end training of the feature pyramid extractor, allowing it to learn to gather features that are important for image synthesis. As shown in Table 1 (fourth section), the quality of the interpolation results improves when using more feature levels. Notice the diminishing returns when using more feature levels, with four levels of features overfitting on the Vimeo-90k dataset. We thus use three levels of features for our approach. We examine the difference between feature pyramids for frame interpolation and those for motion estimation by visualizing their feature responses [10]. Specifically, we maximize the activations of the last layer of our feature pyramid extractor as well as equivalent layers of PWC-Net [45] and LiteFlowNet [18] by altering the input image. Figure 6 shows representative feature activations, indicating that our feature pyramid focuses on fine details which are important to synthesize high-

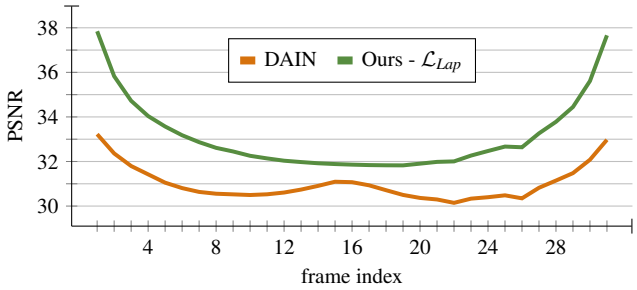


Figure 7: Assessment of the temporal consistency of our approach on the high frame-rate Sintel dataset [21].

quality results while the feature pyramids for optical flow exhibit large patterns to account for large displacements.

Optical flow estimation. To analyze how well our approach performs with different correspondence estimates, we consider three diverse state-of-the-art optical flow methods [18, 19, 45], each trained on FlyingChairs [9]. As shown in Table 1 (fifth section), they all perform similarly well. Due to softmax splatting being fully differentiable, we are further able to fine-tune the optical flow estimation for the task of frame interpolation [51]. Specifically, we fine-tune PWC-Net and see additional improvements with this PWC-Net-ft that has been optimized for the task of frame interpolation. We thus use PWC-Net-ft for our approach.

Perception-distortion tradeoff. We train two versions of our model, one trained on color loss and one trained on perceptual loss, in order to account for the perception-distortion tradeoff [5]. As shown in Table 1 (sixth section), the model trained using color loss $\mathcal{L}_{Lap}$ performs best in terms of PSNR and SSIM whereas the one trained using perceptual loss $\mathcal{L}_F$ performs best in terms of LPIPS. We further note that the $\mathcal{L}_F$-trained model better recovers fine details in challenging cases, making it preferable in practice.

Temporal consistency. Since we use forward warping to compensate for motion, we can interpolate frames at an arbitrary temporal position despite only supervising our model at $t = 0.5$. To analyze the temporal consistency of this approach, we perform a benchmark on a high frame-rate version of the Sintel dataset [21]. Specifically, we interpolate frames 1 through 31 from frame 0 and frame 32 on all of its 13 scenes. We include DAIN for reference since it is also able to interpolate frames at an arbitrary t. As shown in Figure 7, DAIN degrades around frame 8 and frame 24 whereas our approach via softmax splatting does not.

4.2. Quantitative Evaluation

We compare our approach to state-of-the-art frame interpolation methods on common datasets. Since these datasets are all low resolution, we also incorporate 4K video clips from Xiph which are commonly used to assess video compression. Specifically, we selected the eight 4K clips with

	training dataset	Vimeo-90k [51]			Middlebury [2]			UCF101 - DVF [29]			Xiph - 2K			Xiph - "4K"		
		PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
SepConv - $\mathcal{L}_1$	proprietary	33.80	0.956	0.027	35.73	0.959	0.017	34.79	0.947	0.029	34.77	0.929	0.067	32.06	0.880	0.169
SepConv - $\mathcal{L}_F$	proprietary	33.45	0.951	0.019	35.03	0.954	0.013	34.69	0.945	0.024	34.47	0.921	0.041	31.68	0.863	0.097
ToFlow	Vimeo-90k	33.73	0.952	0.027	35.29	0.956	0.024	34.58	0.947	0.027	33.93	0.922	0.061	30.74	0.856	0.132
CyclicGen	UCF101	32.10	0.923	0.058	33.46	0.931	0.046	35.11	0.950	0.030	33.00	0.901	0.083	30.26	0.836	0.142
CtxSyn - $\mathcal{L}_{Lap}$	proprietary	34.39	0.961	0.024	36.93	0.964	0.016	34.62	0.949	0.031	35.71	0.936	0.073	32.98	0.890	0.175
CtxSyn - $\mathcal{L}_F$	proprietary	33.76	0.955	0.017	35.95	0.959	0.013	34.01	0.941	0.024	35.16	0.921	0.035	32.36	0.857	0.081
DAIN	Vimeo-90k	34.70	0.964	0.022	36.70	0.965	0.017	35.00	0.950	0.028	35.95	0.940	0.084	33.49	0.895	0.170
Ours - $\mathcal{L}_{Lap}$	Vimeo-90k	36.10	0.970	0.021	38.42	0.971	0.016	35.39	0.952	0.033	36.62	0.944	0.107	33.60	0.901	0.234
Ours - $\mathcal{L}_F$	Vimeo-90k	35.48	0.964	0.013	37.55	0.965	0.008	35.10	0.948	0.022	35.74	0.921	0.029	32.50	0.856	0.071

Table 2: Quantitative comparison of various state-of-the-art methods for video frame interpolation.

the most amount of inter-frame motion and extracted the first 100 frames from each clip. We then either resized the 4K frames to 2K or took a 2K center crop from them before interpolating the even frames from the odd ones. Since cropping preserves the inter-frame per-pixel motion, this "4K" approach allows us to approximate interpolating at 4K while actually interpolating at 2K instead. Directly processing 4K frames would have been unreasonable since DAIN, for example, already requires 16.7 gigabytes of memory to process 2K frames. In comparison, our approach only requires 5.9 gigabytes to process 2K frames which can be halved by using half-precision floating point operations.

As shown in Table 2, our $\mathcal{L}_{Lap}$-trained model outperforms all other methods in terms of PSNR and SSIM whereas our $\mathcal{L}_F$-trained model performs best in terms of LPIPS. Please note that on the Xiph dataset, all methods are subject to a significant degradation across all metrics when interpolating the "4K" frames instead of the ones that were resized to 2K. This shows that frame interpolation at high resolution remains a challenging problem. For completeness, we also show the per-clip metrics for the samples from Xiph in the supplementary material. We also submitted the results of our $\mathcal{L}_{Lap}$-trained model to the Middlebury benchmark [2]. Our approach currently ranks first in this benchmark as shown in our supplementary material.

4.3. Qualitative Evaluation

Since videos are at the heart of this work, we provide a qualitative comparison in the supplementary video. These support our quantitative evaluation and show difficult examples where our approach yields high-quality results whereas competing techniques are subject to artifacts.

4.4. Discussion

Our proposed softmax splatting enables us to extend and significantly improve the approach from Niklaus *et al*. [37]. Specifically, softmax splatting enables end-to-end training which allows us to not only employ and optimize feature pyramids for image synthesis but also to fine-tune the optical flow estimator [51]. Our evaluation shows that these changes significantly improve the interpolation quality.

Another relevant approach is from Bao *et al*. [3]. They forward-warp the optical flow and then backward warp the input images to the target location according to the warped optical flow. However, they use linear splatting and nearest neighbor interpolation. In comparison, our approach employs softmax splatting which is translational invariant and yields better results than linear splatting. Our approach is also conceptually simpler due to not warping the flow and not incorporating depth- or kernel-estimates. In spite of its simplicity, our approach compared favorably in the benchmark and, unlike DAIN, is temporally consistent.

The success of adversarial training as well as cycle consistency in image generation shows that more advanced supervision schemes can lead to improved synthesis results [14, 28, 43, 56]. Such orthogonal developments could be used to further improve our approach in the future.

5. Conclusion

In this paper, we presented softmax splatting for differentiable forward warping and demonstrated its effectiveness on the application of frame interpolation. The key research question that softmax splatting addresses is how to handle cases where different source pixels forward-warp to the same target location in a differentiable way. Further, we show that feature pyramids can successfully be employed for high-quality image synthesis, which is an aspect of feature pyramids that has not been explored yet. Our proposed frame interpolation pipeline, which is enabled by softmax splatting and conceptually simple, compares favorably in benchmarks and achieves new state-of-the-art results.

Acknowledgments. We are grateful for the feedback from Long Mai and Jon Barron, this paper would not exist without their support. All source image footage shown throughout this paper originates from the DAVIS challenge.

References

[1] Tunç Ozan Aydin, Nikolce Stefanoski, Simone Croci, Markus H. Gross, and Aljoscha Smolic. Temporally Coherent Local Tone Mapping of HDR Video. *ACM Transactions on Graphics*, 33(6):196:1–196:13, 2014. 3

[2] Simon Baker, Daniel Scharstein, J. P. 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. 1, 4, 6, 7, 8

[3] Wenbo Bao, Wei-Sheng Lai, Chao Ma, Xiaoyun Zhang, Zhiyong Gao, and Ming-Hsuan Yang. Depth-Aware Video Frame Interpolation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2019. 1, 2, 3, 4, 6, 8

[4] Wenbo Bao, Wei-Sheng Lai, Xiaoyun Zhang, Zhiyong Gao, and Ming-Hsuan Yang. MEMC-Net: Motion Estimation and Motion Compensation Driven Neural Network for Video Interpolation and Enhancement. *arXiv/1810.08768*, 2018. 2

[5] Yochai Blau and Tomer Michaeli. The Perception-Distortion Tradeoff. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2018. 6, 7

[6] Tim Brooks and Jonathan T. Barron. Learning to Synthesize Motion Blur. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2019. 1

[7] Jose Caballero, Christian Ledig, Andrew P. Aitken, Alejandro Acosta, Johannes Totz, Zehan Wang, and Wenzhe Shi. Real-Time Video Super-Resolution With Spatio-Temporal Networks and Motion Compensation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 2

[8] Xiaodong Cun, Feng Xu, Chi-Man Pun, and Hao Gao. Depth-Assisted Full Resolution Network for Single Image-Based View Synthesis. In *IEEE Computer Graphics and Applications*, 2019. 2

[9] Alexey Dosovitskiy, Philipp Fischer, Eddy Ilg, Philip Häusser, Caner Hazirbas, Vladimir Golkov, Patrick van der Smagt, Daniel Cremers, and Thomas Brox. FlowNet: Learning Optical Flow With Convolutional Networks. In *IEEE International Conference on Computer Vision*, 2015. 7

[10] Dumitru Erhan, Yoshua Bengio, Aaron Courville, and Pascal Vincent. Visualizing Higher-Layer Features of a Deep Network. Technical report, 2009. 7

[11] Damien Fourure, Rémi Emonet, Élisa Fromont, Damien Muselet, Alain Trémeau, and Christian Wolf. Residual Conv-Deconv Grid Network for Semantic Segmentation. In *British Machine Vision Conference*, 2017. 5

[12] Raghudeep Gadde, Varun Jampani, and Peter V. Gehler. Semantic Video CNNs Through Representation Warping. In *IEEE International Conference on Computer Vision*, 2017. 2

[13] Clément Godard, Oisin Mac Aodha, and Gabriel J. Brostow. Unsupervised Monocular Depth Estimation With Left-Right Consistency. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 2

[14] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C. Courville, and Yoshua Bengio. Generative Adversarial Nets. In *Advances in Neural Information Processing Systems*, 2014. 8

[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2016. 5, 6

[16] Haozhi Huang, Hao Wang, Wenhan Luo, Lin Ma, Wenhao Jiang, Xiaolong Zhu, Zhifeng Li, and Wei Liu. Real-Time Neural Style Transfer for Videos. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 3

[17] Jia-Bin Huang, Sing Bing Kang, Narendra Ahuja, and Johannes Kopf. Temporally Coherent Completion of Dynamic Video. *ACM Transactions on Graphics*, 35(6):196:1–196:11, 2016. 3

[18] Tak-Wai Hui, Xiaoou Tang, and Chen Change Loy. LiteFlowNet: A Lightweight Convolutional Neural Network for Optical Flow Estimation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2018. 2, 3, 5, 7

[19] Eddy Ilg, Nikolaus Mayer, Tonmoy Saikia, Margret Keuper, Alexey Dosovitskiy, and Thomas Brox. FlowNet 2.0: Evolution of Optical Flow Estimation With Deep Networks. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 5, 7

[20] Max Jaderberg, Karen Simonyan, Andrew Zisserman, and Koray Kavukcuoglu. Spatial Transformer Networks. In *Advances in Neural Information Processing Systems*, 2015. 1, 2, 3

[21] Joel Janai, Fatma Güney, Jonas Wulff, Michael J. Black, and Andreas Geiger. Slow Flow: Exploiting High-Speed Cameras for Accurate and Diverse Optical Flow Reference Data. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 7

[22] Huaizu Jiang, Deqing Sun, Varun Jampani, Ming-Hsuan Yang, Erik G. 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*, 2018. 1, 2, 3

[23] Nima Khademi Kalantari, Ting-Chun Wang, and Ravi Ramamoorthi. Learning-Based View Synthesis for Light Field Cameras. *ACM Transactions on Graphics*, 35(6):193:1–193:10, 2016. 1

[24] Yoshihiko Kuroki, Tomohiro Nishi, Seiji Kobayashi, Hideki Oyaizu, and Shinichi Yoshimura. A Psychophysical Study of Improvements in Motion-Image Quality by Using High Frame Rates. *Journal of the Society for Information Display*, 15(1):61–68, 2007. 1

[25] Yoshihiko Kuroki, Haruo Takahashi, Masahiro Kusakabe, and Ken-ichi Yamakoshi. Effects of Motion Image Stimuli With Normal and High Frame Rates on EEG Power Spectra: Comparison With Continuous Motion Image Stimuli. *Journal of the Society for Information Display*, 22(4):191–198, 2014. 1

[26] 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*, 2018. 3

[27] Miaomiao Liu, Xuming He, and Mathieu Salzmann. Geometry-Aware Deep Network for Single-Image Novel View Synthesis. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2018. 2

[28] 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*, 2019. 1, 2, 3, 6, 8

[29] Ziwei Liu, Raymond A. Yeh, Xiaoou Tang, Yiming Liu, and Aseem Agarwala. Video Frame Synthesis Using Deep Voxel Flow. In *IEEE International Conference on Computer Vision*, 2017. 2, 6, 8

[30] 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*, 2016. 1

[31] Reza Mahjourian, Martin Wicke, and Anelia Angelova. Unsupervised Learning of Depth and Ego-Motion From Monocular Video Using 3D Geometric Constraints. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2018. 2

[32] Simon Meister, Junhwa Hur, and Stefan Roth. UnFlow: Unsupervised Learning of Optical Flow With a Bidirectional Census Loss. In *AAAI Conference on Artificial Intelligence*, 2018. 2

[33] Simone Meyer, Victor Cornillère, Abdelaziz Djelouah, Christopher Schroers, and Markus H. Gross. Deep Video Color Propagation. In *British Machine Vision Conference*, 2018. 1

[34] Simone Meyer, Abdelaziz Djelouah, Brian McWilliams, Alexander Sorkine-Hornung, Markus H. Gross, and Christopher Schroers. PhaseNet for Video Frame Interpolation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2018. 2

[35] Simone Meyer, Oliver Wang, Henning Zimmer, Max Grosse, and Alexander Sorkine-Hornung. Phase-Based Frame Interpolation for Video. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2015. 2

[36] Thu Nguyen-Phuoc, Chuan Li, Stephen Balaban, and Yong-Liang Yang. RenderNet: A Deep Convolutional Network for Differentiable Rendering From 3D Shapes. In *Advances in Neural Information Processing Systems*, 2018. 2

[37] Simon Niklaus and Feng Liu. Context-Aware Synthesis for Video Frame Interpolation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2018. 1, 2, 4, 5, 6, 7, 8

[38] Simon Niklaus, Long Mai, and Feng Liu. Video Frame Interpolation via Adaptive Convolution. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 2

[39] Simon Niklaus, Long Mai, and Feng Liu. Video Frame Interpolation via Adaptive Separable Convolution. In *IEEE International Conference on Computer Vision*, 2017. 1, 2, 6

[40] Augustus Odena, Vincent Dumoulin, and Chris Olah. Deconvolution and Checkerboard Artifacts. Technical report, 2016. 5

[41] Lars Lau Rakêt, Lars Roholm, Andrés Bruhn, and Joachim Weickert. Motion Compensated Frame Interpolation With a Symmetric Optical Flow Constraint. In *Advances in Visual Computing*, 2012. 2

[42] Anurag Ranjan and Michael J. Black. Optical Flow Estimation Using a Spatial Pyramid Network. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 2

[43] Fitsum A. Reda, Deqing Sun, Aysegul Dundar, Mohammad Shoeybi, Guilin Liu, Kevin J. Shih, Andrew Tao, Jan Kautz, and Bryan Catanzaro. Unsupervised Video Interpolation Using Cycle Consistency. In *IEEE International Conference on Computer Vision*, 2019. 2, 3, 8

[44] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. UCF101: A Dataset of 101 Human Actions Classes From Videos in the Wild. *arXiv/1212.0402*, 2012. 6

[45] 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*, 2018. 2, 3, 5, 7

[46] Xin Tao, Hongyun Gao, Renjie Liao, Jue Wang, and Jiaya Jia. Detail-Revealing Deep Video Super-Resolution. In *IEEE International Conference on Computer Vision*, 2017. 2

[47] Yang Wang, Yi Yang, Zhenheng Yang, Liang Zhao, Peng Wang, and Wei Xu. Occlusion Aware Unsupervised Learning of Optical Flow. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2018. 2

[48] 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. 6

[49] Chao-Yuan Wu, Nayan Singhal, and Philipp Krähenbühl. Video Compression Through Image Interpolation. In *European Conference on Computer Vision*, 2018. 1

[50] Jonas Wulff and Michael J. Black. Temporal Interpolation as an Unsupervised Pretraining Task for Optical Flow Estimation. In *German Conference on Pattern Recognition*, 2018. 1

[51] Tianfan Xue, Baian Chen, Jiajun Wu, Donglai Wei, and William T. Freeman. Video Enhancement With Task-Oriented Flow. *International Journal of Computer Vision*, 127(8):1106–1125, 2019. 1, 2, 5, 6, 7, 8

[52] Jason J. Yu, Adam W. Harley, and Konstantinos G. Derpanis. Back to Basics: Unsupervised Learning of Optical Flow via Brightness Constancy and Motion Smoothness. In *ECCV Workshops*, 2016. 2

[53] 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*, 2018. 6

[54] Tinghui Zhou, Matthew Brown, Noah Snavely, and David G. Lowe. Unsupervised Learning of Depth and Ego-Motion From Video. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 2

[55] Tinghui Zhou, Shubham Tulsiani, Weilun Sun, Jitendra Malik, and Alexei A. Efros. View Synthesis by Appearance Flow. In *European Conference on Computer Vision*, 2016. 2

[56] 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*, 2017. 8