Switchable Temporal Propagation Network

Sifei Liu[1(✉)], Guangyu Zhong[1,3], Shalini De Mello[1], Jinwei Gu[1], Varun Jampani[1], Ming-Hsuan Yang[1,2], and Jan Kautz[1]

[1] NVIDIA, Santa Clara, USA
sifeil@nvidia.com
[2] UC Merced, Merced, USA
[3] Dalian University of Technology, Dalian, China

Abstract. Videos contain highly redundant information between frames. Such redundancy has been studied extensively in video compression and encoding, but is less explored for more advanced video processing. In this paper, we propose a learnable unified framework for propagating a variety of visual properties of video images, including but not limited to color, high dynamic range (HDR), and segmentation mask, where the properties are available for only a few key-frames. Our approach is based on a temporal propagation network (TPN), which models the transition-related affinity between a pair of frames in a purely data-driven manner. We theoretically prove two essential properties of TPN: (a) by regularizing the global transformation matrix as orthogonal, the "style energy" of the property can be well preserved during propagation; and (b) such regularization can be achieved by the proposed switchable TPN with bi-directional training on pairs of frames. We apply the switchable TPN to three tasks: colorizing a gray-scale video based on a few colored key-frames, generating an HDR video from a low dynamic range (LDR) video and a few HDR frames, and propagating a segmentation mask from the first frame in videos. Experimental results show that our approach is significantly more accurate and efficient than the state-of-the-art methods.

1 Introduction

Videos contain highly redundant information between frames. Consider a pair of consecutive frames randomly sampled from a video, it is likely that they are similar in terms of appearance, structure and content in most regions. Such redundancy has been extensively studied in video compression to reduce storage and speedup the transmission of videos, but is less explored for more advanced video processing. A number of recent algorithms, such as optical-flow based warping [1], similarity-guided filtering [2,3] and the bilateral CNN model [4], explore

Electronic supplementary material The online version of this chapter (https://doi.org/10.1007/978-3-030-01234-2_6) contains supplementary material, which is available to authorized users.

V. Ferrari et al. (Eds.): ECCV 2018, LNCS 11211, pp. 89–104, 2018.
https://doi.org/10.1007/978-3-030-01234-2_6

the local relationships between frames to propagate information. These methods model the similarity among pixels, regions or frames from either hand-crafted pixel-level features (e.g., pixel intensities and locations) or apparent motion (e.g., optical flow). They have several potential issues: (a) the designed similarity may not faithfully reflect the image structure, and (b) such similarity may not express the high-level pairwise relationships between frames, e.g., for propagating a segmentation mask in the semantic domain.

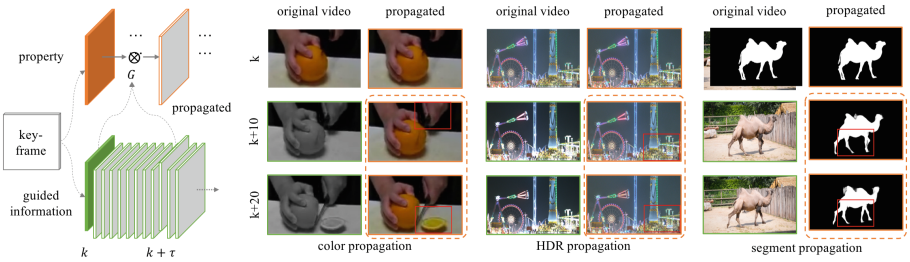

Fig. 1. We propose the TPN model that takes a known property (e.g., color, HDR, segmentation mask) from a key-frame (k), and transform it to a nearby frame ($k+\tau$), denoted by "propagated". The transformation is guided by a learnable matrix G, which is learned from some known information (e.g., lightness, LDR, RGB image). We show three tasks on the right size, where k denotes the key-frame for which the ground-truth property is provided. The orange bounding boxes shows the propagated results of our algorithm, when guided by the information in the left columns. We highlight (red bounding boxes) the regions where the proposed method successfully deals with large transitions or preserves fine details. Zoom-in to see details. (Color figure online)

In this paper, we develop a temporal propagation network (TPN) to explicitly learn pixel-level similarity between a pair of frames (see Fig. 1). It contains a propagation module that transfers a property (e.g., color) of one frame to a nearby frame through a global, linear transformation matrix which is learned with a CNN from any available guidance information (e.g., lightness).

We enforce two principles when learning propagation in the temporal domain: (a) **bi-directionality**, i.e., the propagation between a pair of frames should be invertible, and (b) **consistency**, i.e., the "style energy" (e.g., the global saturation of color) of the target property should be preserved during propagation. We theoretically prove that: enforcing both principles in the TPN is *equivalent* to ensuring that the transformation matrix is orthogonal with respect to each propagation direction. This theoretical result allows us to implement TPN as a novel, special network architecture—the switchable TPN (see Fig. 2)—without explicitly solving for the transformation matrix. It uses bi-directional training for a pair of frames in the propagation module, which is guided by switched output maps from the guidance CNN network. Experiments demonstrate that the proposed architecture is effective in preserving the style energy even between two widely separated frames.

We validate the proposed model for three propagation tasks: (a) video colorization from a few color key-frames and a grayscale video (Sect. 5.2).

With such temporal propagation, the workload of black-and-white video colorization can be largely reduced to only annotating a small number of key-frames. (b) HDR video reconstruction from an LDR video with a few HDR key-frames (Sect. 5.3). This is a new way for HDR video capture, where the whole video can be reconstructed with a few provided HDR frames. (c) video segmentation when only the segmentation mask of the target in the first frame is provided. We show that even without any image-based segmentation model, the proposed method can achieve comparable performance to the state-of-the-art algorithm. All of these tasks reveal that video properties between temporally close frames are highly redundant, and that the relationships between them can be learned from corresponding guidance information. Compared to the existing methods, and aside from the novel architecture, our proposed method also has the following advantages: **(a) High accuracy.** Compared to prior work [4,5], the TPN significantly improves video quality. More importantly, the switchable TPN preserves the style energy significantly better than the network without the switchable structure. **(b) High efficiency.** Our method runs in real-time on a single Titan XP GPU for all the three tasks, which is about 30x to 50x faster than the prior work [4,5] (see Table 1). Moreover, our model does not require sequential processing of video frames, i.e., all video frames can be processed in parallel, which can further improve its efficiency.

2 Related Work and Problem Context

Modeling Affinity for Pixel Propagation. Affinity is a generic measure of closeness between two pixels/entities and is widely used in vision tasks at all levels. Well-modeled affinity reveals how to propagate information from the known pixels to the unknown ones. Most prior methods design affinity measures based on simple, intuitive functions [2,3,6]. Recently, a deep CNN model is proposed to learn task-dependent affinity metric [7] by modeling the propagation of pixels as an image diffusion process.

While [7] is limited to the spatial propagation of pixels for image segmentation, its high-level idea inspires us to learn pixel affinity in other domains via CNNs, e.g., in video sequences as proposed in this work.

Considerably less attention has been paid to develop methods for propagating temporal information across video frames. Jampani et al. [4] propose to propagate video segmentation and color information by embedding pixels into a bilateral space [8] defined based on spatial, temporal and color information. While pixels of the same region from different frames can be closer in this space, it requires several previous frames stacked together to generate a new frame, which results in a high computational cost. Our proposed algorithm is different in that it explicitly *learns* the pixel affinity that describes the task-specific temporal frame transitions, instead of manually defining a similarity measure.

Colorizing Grayscale Videos. Colorization in images and videos is achieved via an interactive procedure in [3], which propagates manually annotated strokes spatially within or across frames, based on a matting Laplacian matrix and with

manually defined similarities. Recently, several methods based on CNNs have been developed to colorize pixels in images with fully-automatic or sparsely annotated colors [9,10]. Due to the multinomial nature of color pixels [10], the interactive procedure usually gives better results. While interactive methods can be employed for single images, it is not practical to annotate them for all frames of a monochrome video. In this work, we suggest a more plausible approach by using a few color key-frames to propagate visual information to all frames in between. To this end, colorizing a full video can be easily achieved by annotating at sparse locations in only a few key-frames, as described in Sect. 5.2.

Video Propagation for HDR Imaging. Most consumer-grade digital cameras have limited dynamic range and often capture images with under/over-exposed regions, which not only degrades the quality of the captured photographs and videos, but also impairs the performance of computer vision tasks in numerous applications. A common way to achieve HDR imaging is to capture a stack of LDR images with different exposures and to fuse them together [11,12]. Such an approach assumes static scenes and thus requires deghosting techniques [13–15] to remove artifacts. Capturing HDR videos for dynamic scenes poses a more challenging problem. Prior methods to create HDR videos are mainly based on hardware that either alternate exposures between frames [16,17], or use multiple cameras [18], or specialized image sensors with pixel-wise exposure controls [19,20]. A few recent methods based on deep models have been developed for HDR imaging. Kalantari et al. [21] use a deep neural network to align multiple LDR images into a single HDR image for dynamic scenes. Zhang et al. [22] develop an auto-encoder network to predict a single HDR panorama from a single exposed LDR image for image-based rendering. In addition, Eilertsen et al. [5] propose a similar network for HDR reconstruction from a single LDR input image, which primarily focuses on recovering details in the high intensity saturated regions.

In this paper, we apply the TPN for HDR video reconstruction from a LDR video. Given a few HDR key-frames and an LDR video, the TPN propagates the scene radiance information from the key-frames to the remaining frames. Note that unlike all the existing single LDR-based methods [5,22], which hallucinate the missing HDR details in images, we focus on propagating the HDR information from the input few HDR images to neighboring LDR frames, which provides an alternative solution for efficient, low cost HDR video reconstruction.

3 Proposed Algorithm

We exploit the redundancy in videos, and propose the TPN for learning affinity and propagating target properties between frames. Take the video colorization as an example. Given an old black-and-white movie with a few key-frames colorized by artists, can we automatically colorize the entire movie? This problem can be equivalently reformulated as propagating a target property (i.e., color) based on the affinity of some features (e.g., lightness) between two frames. Intuitively, this is feasible because (1) videos have redundancy over time—nearby frames tend

to have similar appearance, and (2) the pixel correlation between two frames in the lightness domain is often consistent with that in the color domain.

In this work, we model the propagation of a target property (e.g., color) between two frames as a *linear transformation*,

$$U_t = GU_k, \tag{1}$$

where $U_k \in \mathcal{R}^{n^2 \times 1}$ and $U_t \in \mathcal{R}^{n^2 \times 1}$ are the vectorized version of the $n \times n$ property maps of a key-frame and a nearby frame, and $G \in \mathcal{R}^{n^2 \times n^2}$ is the transformation matrix to be estimated[1]. Suppose we observe some features of the two frames (e.g., lightness) V_k and V_t, the transformation matrix G is thus a function of V_k and V_t,

$$G = g(\theta, V_k, V_t). \tag{2}$$

The matrix G should be dense in order to model any type of pixel transition in a global scope, but G should also be concise for efficient estimation and propagation. In Sect. 3.1, we propose a solution, called basic TPN, by formulating the linear transformation G as an image diffusion process similar to [7]. Following that, in Sect. 3.2, we introduce the key part of our work, the switchable TPN, which enforces the bi-directionality and the style consistency for temporal propagation. We prove that enforcing these two principles is equivalent to ensuring the transformation matrix G is orthogonal, which in turn can be easily implemented by equipping an ordinary temporal propagation network with a switchable structure.

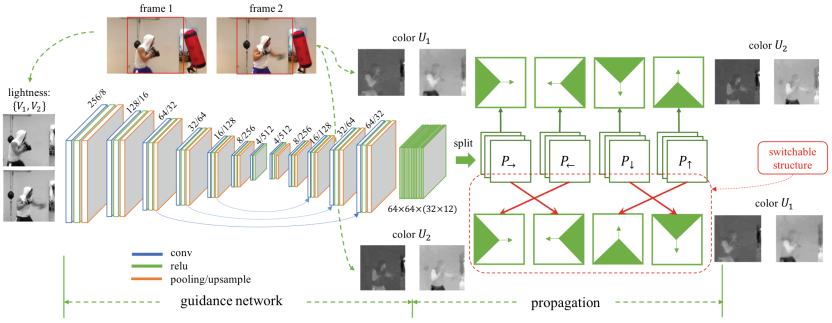

Fig. 2. The architectures of a switchable TPN, which contains two propagation modules for bi-directional training. We specifically use a red-dashed box to denote the switchable structure In the reversed pair, the output channels $\{P\}$ are switched (red) for horizontal and vertical propagation. (Color figure online)

[1] For property maps with multiple channels $n \times n \times c$, we treat each channel separately.

3.1 Learning Pixel Transitions via the Basic TPN

Directly learning the transformation matrix G via a CNN is prohibitive, since G has a huge dimension (e.g., $n^2 \times n^2$). Instead, inspired by the recent work [7], we formulate the transformation as a diffusion process, and implement it efficiently by propagating information along each row and each column in an image linearly. Suppose we keep only $k = 3$ nearest neighbors from a previous column (row) during the propagation, and we perform the propagation in $d = 4$ directions, the total number of parameters to be estimated is significantly reduced from $n^2 \times n^2$ to $n^2 \times k \times d$ (see example of Fig. 2).

Linear Transformation as a Diffusion Process. The diffusion process from frame k to frame t can be expressed with a partial differential equation (PDE) in the discrete form as:

$$\nabla U = U_t - U_k = -LU_k = (A - D)U_k, \tag{3}$$

where $L = D - A$ is the Laplacian matrix, D is the diagonal degree matrix and A is the affinity matrix. In our case, this represents the propagation of the property map U over time. (3) can be re-written as $U_t = (I - D + A)\, U_k = GU_k$, where G is the transformation matrix between the two states, as defined in (1), and I is an identity matrix.

Linear Propagation Network. With a propagation structure, the diffusion between frames can be implemented as a linear propagation along the rows or columns of an image. Here we briefly show their equivalence. Following [7], we take the left-to-right spatial propagation operation as an example:

$$y_i = (I - d_i)\, x_i + w_i y_{i-1}, \quad i \in [2, n], \tag{4}$$

where $x \in U_k$ and $y \in U_t$, and the $n \times 1$ vectors $\{x_i, y_i\}$ represent the i^{th} columns before and after propagation with an initial condition of $y_1 = x_1$, and w_i is the spatially varying $n \times n$ sub-matrix. Here, I is the identity matrix and d_i is a diagonal matrix, whose t^{th} element is the sum of all the elements of the t^{th} row of w_i as $d_i(t,t) = \sum_{j=1, j\neq t}^{n} w_i(j,t)$. Similar to [7], through (a) expanding its recursive term, and (b) concatenating all the rows/columns as a vectorized map, it is easy to prove that (4) is equivalent to the global transformation G between U_k and U_t, where each entry is the multiplication of several spatially variant w_i matrices [7]. Essentially, instead of predicting all the entries in G as independent variables, the propagation structure transfers the problem into learning each sub-matrix w_i in (4), which significantly reduces the output dimensions.

Learning the Sub-matrix $\{w_i\}$. We adopt an independent deep CNN, namely the guidance network, to output all the sub-matrices w_i. Note that the propagation in (1) is carried out for $d = 4$ directions independently, as shown in Fig. 2. For each direction, it takes a pair of images $\{V_k, V_t\}$ as its input, and outputs a feature map P that has the same spatial size as U (see Fig. 2). Each pixel in the feature map $p_{i,j}$ contains all the values of the jth row in w_i, which describes a local relation between the adjacent columns, but results in a global connection

in G though the propagation structure. Similar to [7], we keep only $k = 3$ nearest neighbors from the previous column, which results in w_i being a tridiagonal matrix. Thus, a total of $n \times n \times (k \times d)$ parameters are used to implement the transformation matrix G. Such a structure significantly compresses the guidance network while still ensuring that the corresponding G is a dense matrix that can describe global and dense pairwise relationships between a pair of frames.

3.2 Preserving Consistency via Switchable TPN

In this part, we show that there are two unique characteristics of propagation in the temporal domain, which do not exist for propagation in the spatial domain [7]. First, temporal propagation is bi-directional for two frames, i.e., a network capable of transforming a frame U_1 into a frame U_2, should also be able to transform from U_2 to U_1, with a corresponding reversed ordering of inputs to the guidance network. Second, during propagation, the overall "style" of the propagated property across the image should stay constant between frames, e.g., during color propagation, the color saturation of all frames within a short video clip should be similar. We call this feature "consistency property". As shown below, we prove that enforcing the bi-directionality and the consistency is equivalent to ensure the transformation matrix G to be orthogonal, which in turn can be easily implemented by equipping an ordinary temporal propagation network with a switchable structure.

Bi-directionality of TPN. We assume that properties in nearby video frames do not have a causal relationship. This assumption holds for most properties that naturally exist in the real-world, e.g., color and HDR. Hence, temporal propagation of these properties can often be switched in direction without breaking the process. Given a diffusion model G and a pair of frames $\{U_1, U_2\}$, we have a pair of equations:

$$U_2 = G_{1\to 2}U_1, \quad U_1 = G_{2\to 1}U_2, \tag{5}$$

where the arrow denotes the propagation direction. The bi-directionality property implies that reversing the roles of the two frames as inputs by $\{V_1, V_2\} \to \{V_2, V_1\}$, and the corresponding supervision signals to the network corresponds to applying an inverse transformation matrix $G_{2\to 1} = G_{1\to 2}^{-1}$.

Style Preservation in Sequences. Style consistency refers to whether the generated frames can maintain similar chromatic properties or brightness when propagating color or HDR information, which is important for producing high-quality videos without the property vanishing over time. In this work, we ensure such global temporal consistency by minimizing the difference in style loss of the propagated property for the two frames. Style loss has been intensively used in style transfer [23], but has not yet been used for regularizing temporal propagation. In our work, we represent the style by the Gram matrix, which is proportional to the un-centered covariance of the property map. The style loss is the squared Frobenius norm of the difference between the Gram matrices of the key-frame and the succeeding frame:

Theorem 1. *By regularizing the style loss we have the following optimization w.r.t. the guidance network:*

$$\min \frac{1}{N} \parallel U_1^\top U_1 - U_2^\top U_2 \parallel_F^2 \tag{6}$$

$$s.t. \qquad U_2 = GU_1. \tag{7}$$

The optimal solution is reached when G is orthogonal.

Proof. Since the function (6) is non-negative, the minimum is reached when $U_1^\top U_1 = U_2^\top U_2$. Combining it with (7) we have $G^\top G = I$.

Given that G is orthogonal, the $G_{2\to1}$ in (5) can be replaced by $G_{1\to2}^\top$, which equals to $G_{1\to2}^{-1}$. Therefore, the bi-directionality propagation can be represented via a pair of transformation matrices that are *transposed* w.r.t each other. In the following part, we show how to enforce this property for the transformation matrix G in the linear propagation network via a special network architecture. Note that in our implementation, even though we use the channel-wise propagation described in Sect. 3.1, where the $U^\top U$ actually reduces to an uncentered variance, the conclusions of Theorem 1 still hold.

A Switchable Propagation Network. The linear transformation matrix G has an important property: since the propagation is directed, the transformation matrix G is a triangular matrix. Consider the two directions along the horizontal axis (i.e., $\rightarrow, \leftarrow$) in Fig. 2. G is an upper triangular matrix for a particular direction (e.g., $\rightarrow$), while it is lower triangular for the opposite one (e.g., $\leftarrow$). Suppose $P_\rightarrow$ and $P_\leftarrow$ are the output maps of the guidance network w.r.t. these two opposite directions. This means that the transformation matrix, which is lower-triangular for propagation in the left-to-right direction, becomes upper-triangular for the opposite direction of propagation. Since the upper-triangular matrix: (a) corresponds to propagating in the right-to-left direction, and (b) contains the same set of weight sub-matrices, *switching the CNN output channels w.r.t. the opposite directions $P_\rightarrow$ and $P_\leftarrow$ is equivalent to transposing the transformation matrix G in the TPN.* This fact is exploited as a regularization structure (see the red box in Fig. 2) during training.

To summarize, the switchable structure of the TPN is derived from the two principles (i.e., the bi-directionality and the style consistency) for temporal propagation and the fact that the matrix G is triangular due to the specific form of propagation. Note that [7] did not address the triangulation of the matrix and thus were limited to propagation in the spatial domain only. We show the switchable TPN (STPN) largely improve performance over the basic TPN, with no computational overhead at inference time.

4 Network Implementation

We provide the network implementation details shared by color, HDR and segmentation mask propagation, which are proposed in this work. These settings can be potentially generalized to other properties of videos as well.

The Basic TPN. The basic TPN contains two separate branches: (a) a deep CNN for the guidance network, which takes as input the provided information $\{V_1, V_2\}$ from a pair of frames, and outputs all elements (P) that constitute the state transformation matrix G, and (b) a linear propagation module that takes the property map of one frame U_1 and outputs U_2. It also takes as input $\{P\}$ the propagation coefficients following the formulation of (4), where $\{P\}$ contains kd channels ($k = 3$ connections for each pixel per direction, and $d = 4$ directions in total). $\{V, U, P\}$ have the same spatial size according to (4). We use node-wise max-pooling [7,24] to integrate the hidden layers and to obtain the final propagation result. All submodules are differentiable and jointly trained using stochastic gradient descent (SGD), with the base learning rate of 10^{-5} (Fig. 3).

The Switchable TPN. Figure 2 shows how the switchable structure of the TPN is exploited as an additional regularization loss term during training. For each pair (U_1, U_2) of the training data, the first term in (8) shows the regular supervised loss between the network estimation $\hat{U}_2$ and the groundtruth U_2. In addition, as shown in Fig. 2(b), since we want to enforce the bi-directionality and the style consistency in the switchable TPN, the same network should be able to propagate from U_2 back to U_1 by simply switching the channels of the output of the guidance networks, i.e., switching the channels of $\{P \rightarrow, P \leftarrow\}$ and $\{P \downarrow, P \uparrow\}$ for propagating information in the opposite direction. This will form the second loss term in (8), which serves as a regularization (weighted by λ) during the training. We set $\lambda = 0.1$ for all the experiments in this paper.

$$L(U_1, \hat{U}_1, U_2, \hat{U}_2) = \left\| U_2(i) - \hat{U}_2(i) \right\|^2 + \lambda \left\| U_1(i) - \hat{U}_1(i) \right\|^2. \tag{8}$$

At inference time, the switchable TPN reduces to the basic TPN introduced in Sect. 3.1 and therefore does not have any extra computational expense.

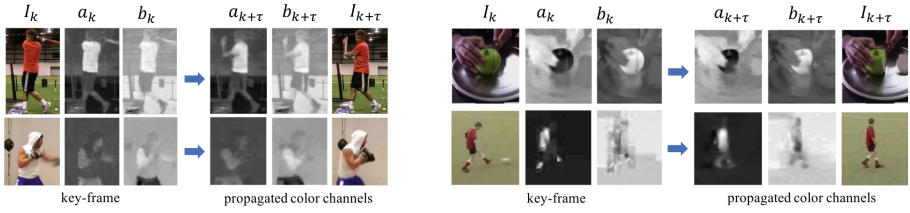

Fig. 3. We show two groups of color transitions output through a basic TPN. For each group, the left side is key-frames with the ground truth color images provided, and the right side is new frames propagated from the left. $\{a_k, b_k\}$ and $\{a_{k+\tau}, b_{k+\tau}\}$ are the inputs and outputs of the TPN. All four examples show obvious appearance transitions caused by movement of objects. Zoom-in to see details.

Table 1. Run-times of different methods. We set $K = 30$ for VPN color propagation [4] to calculate its run-time. The last four columns are our methods.

Method	VPN [4] (color)	VPN [4] (seg)	HDRCNN [5]	Color	HDR	SEG(t)	SEG(t+s)
(ms)	730	750	365	15	25	17	84

5 Experimental Results

In this section, we present our experimental results for propagating color channels, HDR images, and segmentation mask across videos. We note that propagating information across relatively longer temporal intervals may not satisfy the assumptions of a diffusion model, especially when new objects or scenes appear. Hence, for color and HDR propagation, instead of considering such complex scenarios, we set "key-frames" at regular fixed intervals for both tasks. That is, the ground truth color or HDR information is provided for every K frames and propagated to all frames in between them. This is a practical strategy for real-world applications. Note that for video segmentation mask propagation, we still follow the protocol of the DAVIS dataset [25] and only use the mask from the first frame.

5.1 General Network Settings and Run-Times

We use a guidance network and a propagation module similar to [7], with two cascaded propagation units. For computational and memory efficiency, the propagation is implemented with a smaller resolution, where U is downsampled from the original input space to a hidden layer before being fed into the propagation module. The hidden layer is then bi-linearly upsampled to the original size of the image. We adopt a symmetric U-net shaped, light-weight deep CNN with skip links for all tasks, but with slightly different numbers of layers to accommodate the different input resolutions (see Fig. 2 as an example for color propagation). We first pre-train the model on synthesized frame pairs generated from an image dataset. (e.g., the MS-COCO dataset [26] for color and segmentation propagation, and a self-collected one for HDR propagation, see the supplementary material). Given an image, we augment it in two different ways via a similarity transform with uniformly sampled parameters from $s \in [0.9, 1.1], \theta \in [-15°, 15°]$ and $dx \in [-0.1, 0.1] \times b$, where $b = \min(H, W)$. We also apply this data augmentation while training with patches from video sequences. We present the run-times for different methods on an 512×512 image using a single TITAN X (Pascal) NVIDIA GPU (without cuDNN) in Table 1.

5.2 Color Propagation in Videos

We use the ACT dataset [27], which contains 7260 training sequences with about 600K frames in total of various categories of actions. All the sequences are short

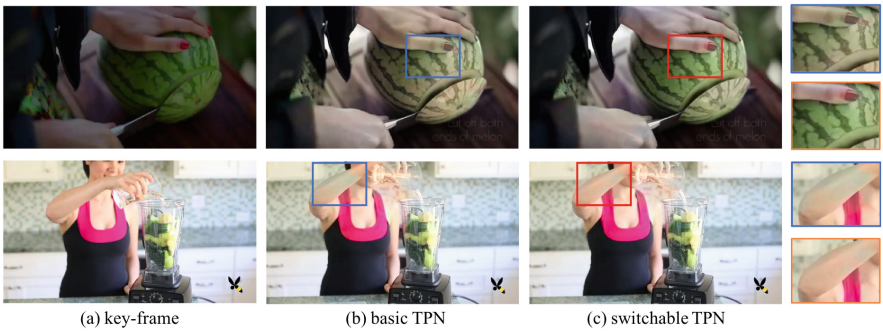

Fig. 4. An example of color propagation from a key-frame (a) to a new frame with considerably large appearance transitions, using either (b) the basic TPN or (c) the switchable TPN. The closeups show the detailed comparison. Zoom-in to see details. (Color figure online)

with small camera or scene transitions, and thus are more suitable for the proposed task. We re-train and evaluate the VPN network on the ACT dataset for a fair comparison. The original testing set contains 3974 sequences with more than 300K frames. For faster processing, we randomly select five videos from every action category in order to maintain the prior distribution of the original ACT dataset. We use one for testing and the remaining four for training.

We perform all computations in the *CIE-Lab* color space. After pretraining on the MS-COCO dataset, we fine-tune the models on the ACT dataset by randomly selecting two frames from a sequence and cropping both frames at the same spatial location as a single training sample. Specifically, our TPN takes as input the concatenated ab channels that are randomly cropped to 256×256 from a key-frame. The patches are then transformed to $64 \times 64 \times 32$ via 2 convoluitional layers with $stride = 2$ before being input to the propagation module. After propagation, the output maps are upsampled as a transformed ab image map for the frames following the key-frame. The guidance CNN takes as input a pair of lightness images (L) for the two frames. We optimize the Euclidean loss (in the ab color space) between the ground truth and the propagated color channels generated by our network. Note that for the switchable TPN, we have two losses with different weights according to (8). During testing, we combine the estimated ab channels with the given L channel to generate a color RGB image. All our evaluation metrics are computed in the *RGB* color space.

We compare the models with three combinations. We refer to the basic and switchable TPN networks as BTPN and STPN, the suffix "im" and "vd" denote pretraining on MS-COCO and finetuning on ACT, respectively. The methods that we compare include: (a) BTPNim+BTPNvd, (b) BTPNim+STPNvd, and (c) STPNim+STPNvd; and evaluate different key-frames intervals, including $K = \{10, 20, 30, 40\}$. The quantitative results for root mean square error (RMSE) and peak signal-to-noise ratio (PSNR) are presented in Table 2. Two trends can be inferred from the results. First, the switchable TPN consistently outperforms the basic TPN and the VPN [4], and using the switchable TPN structure for

Table 2. RMSE and PSNR (in parentheses) for video color propagation on the ACT dataset for different key-frame interval K. We compared VPN with $K = 30$.

Eval	RMSE				PSNR			
Interval	$K = 10$	$K = 20$	$K = 30$	$K = 40$	$K = 10$	$K = 20$	$K = 30$	$K = 40$
BTPNim+BTPNvd	4.43	5.46	6.04	6.44	36.65	35.22	34.46	33.96
BTPNim+STPNvd	4.00	5.00	5.58	6.01	37.63	36.09	35.26	34.70
STPNim+STPNvd	**3.98**	**4.97**	**5.55**	**5.99**	**37.64**	**36.12**	**35.29**	**34.73**
VPN (stage-1) [4]	-	-	6.86	-	-	-	32.86	-

both the pre-training and fine-tuning stages generates the best results. Second, while the errors decrease drastically on reducing time intervals between adjacent key-frames, the colorized video maintains overall high-quality even when K is set close to a common frame rate (e.g., 25 to 30 fps). We also show in Fig. 4(b) and (c) that the switchable structure significantly improves the qualitative results by preserving the saturation of color, especially when there are large transitions between the generated images and their corresponding key-frames. The TPN also maintains good colorization for fairly long video sequences, which is evident from a comparison of the colorized video frames with the ground truth in Fig. 5. Over longer time intervals, the quality of the switchable TPN degrades much more gracefully than that of the basic TPN and the VPN [4].

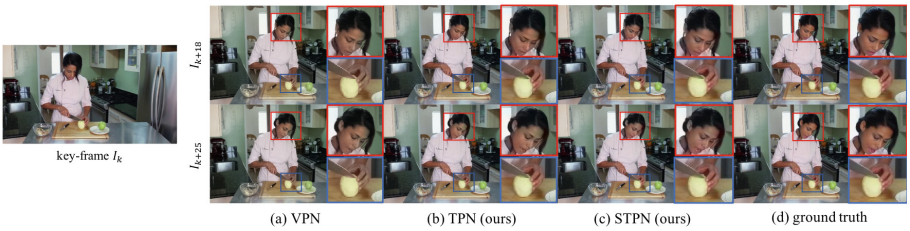

Fig. 5. Results using propagation of color from a key-frame to two proceeding frames (the 18^{th} and the 25^{th}) at different time intervals with the basic/switchable TPN, and the VPN [4] models. Zoom-in to see details. (Color figure online)

5.3 HDR Propagation in Videos

We compare our method against the work of [5], which directly reconstructs the HDR frames given the corresponding LDR frames as inputs. While this is not an apples-to-apples comparison because we also use an HDR key-frame as input, the work [5] is the closest related state-of-the-art method to our approach for HDR reconstruction. To our knowledge, no prior work exists on propagating HDR information in videos using deep learning and ours is the first work to address this problem. We use a similar network architecture as color propagation except that U is transformed to $128 \times 128 \times 16$ via one convolution layer to preserve

more image details, and a two-stage training procedure by first pre-training the network with randomly augmented pairs of patches created from a dataset of HDR images, and then fine-tuning on an HDR video dataset. We collect the majority of the publicly available HDR image and video datasets listed in the supplementary material, and utilize all the HDR images and every 10-th frame of the HDR videos for training in the first stage [5]. Except for the four videos (the same as [5]) that we use for testing, we train our TPN with all the collected videos. We evaluate our method on the four videos that [5] used for testing and compare against their method.

To deal with the long-tail, skewed distribution of pixel values in HDR images, similarly to [5], we use the logarithmic space for HDR training with $U = \log(H + \varepsilon)$, where H denotes an HDR image and ε is set to 0.01. Since the image irradiance values recorded in HDR images vary significantly across cameras, naively merging different datasets together often generates domain differences in the training samples. To resolve this issue, before merging the datasets acquired by different cameras, we subtract from each input image the mean value of its corresponding dataset. We use the same data augmentation as in [5] of varying exposure values and camera curves [28] during training. During testing, we follow [5] to blend the inverse HDR image created from the input LDR image with the HDR image predicted by our TPN network to obtain the final output HDR image. More details are presented in the supplementary material.

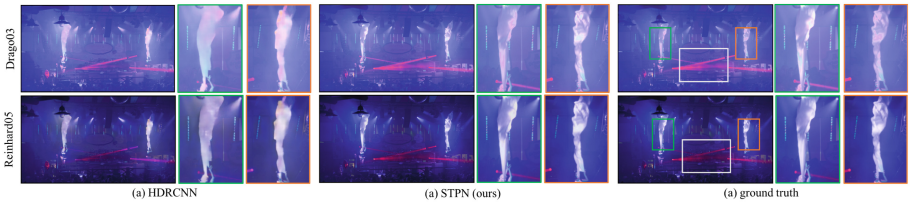

Fig. 6. Results of HDR Video Propagation. We show one HDR frame ($\tau = 19$ frames away from the key frame) reconstructed with our switchable TPN (middle column). The top row shows the ground truth HDR, and the bottom row shows the output of HDRCNN [5]. The HDR images are displayed with two popular tone mapping algorithms, Drago03 [29] and Reinhard05 [30]. The insets show that the switchable TPN can effectively propagate the HDR information to new frames and preserve the dynamic range of scene details. Zoom-in to see details.

We compare the RMSE of the generated HDR frames for different intervals between the key-frames, with or without the blending of LDR information with the HDR image generated by the TPN in Table 3. Our results indicate that the switchable TPN can also significantly improve the results for HDR propagation compared to the basic TPN. We also compare with the frame-wise reconstruction method [5], with and without the blending-based post-processing in Fig. 6. As shown, our TPN recovers HDR images with up to $K = 30$ frames away from each key frame. The reconstructed HDR images preserve the same scene details as the ground truth, under different tone mapping algorithms. More results are

Table 3. RMSE for video HDR propagation for the TPN output, the output with LDR blending, for different intervals for the key-frames K. Reconstruction from single LDR [5] is compared under the same experimental settings.

Settings	HDR with blending				HDR without blending			
Interval	$K = 10$	$K = 20$	$K = 30$	$K = 40$	$K = 10$	$K = 20$	$K = 30$	$K = 40$
BTPNim+BTPNvd	0.031	0.034	0.038	0.042	0.119	0.160	0.216	0.244
BTPNim+BTPNvd	0.028	0.031	0.034	0.038	**0.096**	**0.115**	0.146	**0.156**
BTPNim+BTPNvd	**0.027**	**0.030**	0.034	**0.037**	0.098	0.121	**0.142**	0.159
HDRCNN [5]	0.038				0.480			

presented in the supplementary material. As noted earlier, since we have additional HDR key-frames as input, it is not an apples-to-apples comparison with single-image based HDR methods like [5]. Nevertheless, the results in Fig. 6 show the feasibility of using sparsely-sampled HDR key-frames to reconstruct HDR videos from LDR videos with the proposed TPN approach.

5.4 Segmentation Mask Propagation in Videos

In addition, we conduct video segmentation on the DAVIS dataset [25] with the same settings as VPN [4], to show that the proposed method can also generalized to semantic-level propagation in videos. We note that maintaining style consistency does not apply to semantic segmentation. For each frame to be predicted, we use the segmentation mask of the first frame as the only key-frame, while using the corresponding RGB images as the input to the guidance network. We train two versions of the basic TPN network for this task: (a) A basic TPN with the input/output resolution reduced to 256×256, the U transformed to $64 \times 64 \times 16$, in the same manner as the color propagation model. We used the same guidance network architecture as [7], while removing the last convolution unit to fit the dimensions of the propagation module. This model, denoted as SEG(t) in Table 1, is much more efficient than the majority of the recent video segmentation methods [4,25,31]. (b) A more accurate model with an SPN [7] refinement applied to the output of the basic TPN, denoted as SEG(t+s). This model utilizes the same architecture as [7], except that it replaces the loss with Sigmoid cross entropy for the per-pixel classification task. Similar to color and HDR propagation, We pretrain (a) on the MS-COCO dataset and then finetune it on the DAVIS training set. For the SPN model in (b), we first train it on the VOC image segmentation task as described in [7]. We treat each class in an image as binary mask in order to transfer the original model to a two-class classification model, while replacing the corresponding loss module. We then finetune the SPN on the coarse masks from the DAVIS training set, which are produced by an intermediate model – the pre-trained version of (a) from the MS-COCO dataset. More details are introduced in the supplementary materiel.

We compare our method to VPN [4] and one recent state-of-the-art method [31]. Both VPN and our method rely purely on the propagation module

Table 4. Comparisons for video segmentation on the DAVIS dataset.

J-mean				F-mean			
VPN [4]	OSVOS [31]	SEG(t)	SEG(t+s)	VPN [4]	OSVOS [31]	SEG(t)	SEG(t+s)
70.2	79.8	71.1	76.19	65.5	80.6	75.65	73.53

from the first frame and does not utilize any image segmentation pre-trained modules (in contrast with [31]). Similar to the other two tasks, both models significantly outperform VPN [4] for video segmentation propagation (see Table 4), while all running one order of magnitude faster (see Table 1). The SEG(t+s) model performs comparatively to the OSVOS [31] method, which utilizes the pretrained image segmentation model and requires a much long inference time (7800 ms).

References

1. Gadde, R., Jampani, V., Gehler, P.V.: Semantic video CNNs through representation warping. In: Proceedings of IEEE International Conference on Computer Vision (ICCV) (2017)
2. He, K., Sun, J., Tang, X.: Guided image filtering. IEEE Trans. Pattern Anal. Mach. Intell. (TPAMI) **35**(6), 1397–1409 (2013)
3. Levin, A., Lischinski, D., Weiss, Y.: Colorization using optimization. In: ACM Transactions on Graphics (TOG) (2004)
4. Jampani, V., Gadde, R., Gehler, P.: Video propagation networks. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017)
5. Eilertsen, G., Kronander, J., Denes, G., Mantiuk, R., Unger, J.: HDR image reconstruction from a single exposure using deep CNNs. In: ACM Transactions on Graphics (SIGGRAPH Asia) (2017)
6. Levin, A., Lischinski, D., Weiss, Y.: A closed-form solution to natural image matting. IEEE Trans. Pattern Anal. Mach. Intell. (TPAMI) **30**(2), 228–242 (2008)
7. Liu, S., Mello, S.D., Gu, J., Zhong, G., Yang, M., Kautz, J.: Learning affinity via spatial propagation networks. In: Neural Information Processing Systems (NIPS) (2017)
8. Jampani, V., Kiefel, M., Gehler, P.V.: Learning sparse high dimensional filters: image filtering, dense CRFs and bilateral neural networks. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2016)
9. Zhang, R., Isola, P., Efros, A.A.: Colorful image colorization. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9907, pp. 649–666. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46487-9_40
10. Zhang, R., Zhu, J.Y., Isola, P., Geng, X., Lin, A.S., Yu, T., Efros, A.A.: Real-time user-guided image colorization with learned deep priors. In: ACM Transactions on Graphics (SIGGRAPH) (2017)
11. Debevec, P., Malik, J.: Recovering high dynamic range radiance maps from photographs. In: ACM Transactions on Graphics (SIGGRAPH) (1997)
12. Reinhard, E., Heidrich, W., Debevec, P., Pattanaik, S., Ward, G., Myszkowski, K.: High Dynamic Range Imaging: Acquisition, Display, and Image-based Lighting. Morgan Kaufmann, San Francisco (2010)

13. Hu, J., Gallo, O., Pulli, K., Sun, X.: HDR deghosting: how to deal with saturation? In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2013)
14. Oh, T., Lee, J., Tai, Y., Kweon, I.: Robust high dynamic range imaging by rank minimization. IEEE Trans. Pattern Anal. Mach. Intell. (TPAMI) **37**(6), 1219–1232 (2015)
15. Gallo, O., Troccoli, A., Hu, J., Pulli, K., Kautz, J.: Locally non-right registration for mobile HDR photography. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2015)
16. Kang, S., Uyttendaele, M., Winder, S., Szeliski, R.: High dynamic range video. In: ACM Transactions on Graphics (SIGGRAPH) (2003)
17. Kalantari, N., Shechtman, E., Barnes, C., Darabi, S., Goldman, D., Sen, P.: Patch-based high dynamic range video. In: ACM Transactions on Graphics (SIGGRAPH) (2013)
18. Tocci, M., Kiser, C., Tocci, N., Sen, P.: A versatile HDR video production system. In: ACM Transactions on Graphics (SIGGRAPH) (2011)
19. Nayar, S., Mitsunaga, T.: High dynamic range imaging: Spatially varying pixel exposure. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2000)
20. Gu, J., Hitomi, Y., Mitsunaga, T., Nayar, S.: Coded rolling shutter photography: flexible space-time sampling. In: Proceedings of IEEE International Conference on Computational Photography (ICCP) (2010)
21. Kalantari, N., Ramamoorthi, R.: Deep high dynamic range imaging of dynamic scenes. In: ACM Transactions on Graphics (SIGGRAPH) (2017)
22. Zhang, J., Lalonde, J.: Learning high dynamic range from outdoor panoramas. In: Proceedings of IEEE International Conference on Computer Vision (ICCV) (2017)
23. Gatys, L.A., Ecker, A.S., Bethge, M.: A neural algorithm of artistic style. CoRR abs/1508.06576 (2015)
24. Liu, S., Pan, J., Yang, M.-H.: Learning recursive filters for low-level vision via a hybrid neural network. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9908, pp. 560–576. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46493-0_34
25. Perazzi, F., Pont-Tuset, J., McWilliams, B., Van Gool, L., Gross, M., Sorkine-Hornung, A.: A benchmark dataset and evaluation methodology for video object segmentation. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2016)
26. Lin, T.-Y., et al.: Microsoft COCO: common objects in context. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8693, pp. 740–755. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10602-1_48
27. Wang, X., Farhadi, A., Gupta, A.: Actions ∼ transformations. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2016)
28. Grossberg, M.D., Nayar, S.K.: Modeling the space of camera response functions. IEEE Trans. Pattern Anal. Mach. Intell. (TPAMI) **26**(10), 1272–1282 (2004)
29. Drago, F., Myszkowski, K., Annen, T., Chiba, N.: Adaptive logarithmic mapping for displaying high contrast scenes. Comput. Graph. Forum **22**(3), 419–426 (2003)
30. Reinhard, E., Devlin, K.: Dynamic range reduction inspired by photoreceptor physiology. IEEE Trans. Vis. Comput. Graph. **11**(1), 13–24 (2005)
31. Caelles, S., Maninis, K.K., Pont-Tuset, J., Leal-Taixé, L., Cremers, D., Van Gool, L.: One-shot video object segmentation. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017)