Recurrent Temporal Aggregation Framework for Deep Video Inpainting

Dahun Kim*, *Student Member, IEEE,* Sanghyun Woo*, *Student Member, IEEE,*
Joon-Young Lee, *Member, IEEE,* and In So Kweon, *Member, IEEE*

Abstract—Video inpainting aims to fill in spatio-temporal holes in videos with plausible content. Despite tremendous progress on deep learning-based inpainting of a single image, it is still challenging to extend these methods to video domain due to the additional time dimension. In this paper, we propose a recurrent temporal aggregation framework for fast deep video inpainting. In particular, we construct an encoder-decoder model, where the encoder takes multiple reference frames which can provide visible pixels revealed from the scene dynamics. These hints are aggregated and fed into the decoder. We apply a recurrent feedback in an auto-regressive manner to enforce temporal consistency in the video results. We propose two architectural designs based on this framework. Our first model is a blind video decaptioning network (BVDNet) that is designed to automatically remove and inpaint text overlays in videos without any mask information. Our BVDNet wins the first place in the ECCV Chalearn 2018 LAP Inpainting Competition Track 2: Video Decaptioning. Second, we propose a network for more general video inpainting (VINet) to deal with more arbitrary and larger holes. Video results demonstrate the advantage of our framework compared to state-of-the-art methods both qualitatively and quantitatively. The codes are available at https://github.com/mcahny/Deep-Video-Inpainting, and https://github.com/shwoo93/video_decaptioning.

Index Terms—Video inpainting, Video completion, Video object removal, Video caption removal, Video decaptioning, Video editing

1 Introduction

Removing unwanted items in a video is a practical and crucial problem as it can help numerous video restoration and editing tasks such as scratch restoration, automatic caption removal, and undesired object removal. This also opens more opportunities to video content generation and manipulation tasks such as inserting new elements in a scene [1], [2], [3], [4]. Furthermore, there are several semi-online streaming scenarios such as automatic content filtering and visual privacy filtering.

Despite tremendous progress on deep learning-based inpainting of a single image, it is still challenging to extend these methods to video domain due to the additional time dimension. A straightforward way of video inpainting is to apply image inpainting on each frames individually, but this comes with a clear limitation that the video results are unstable and inconsistent over time. The second row in Figure 10 shows such an example when using the state-of-the-art feed-forward image inpainting [5] in a frame-by-frame manner.

The challenge of video inpainting is to fill in the holes with contents that are spatially plausible and temporally coherent at the same time. Early works address this problem by using a patch based greedy selection [6], a per-frame diffusion based technique [7], or a global flow field based optimization [8]. While the last shows the state-of-the art [8] (3rd row in Figure 10) quality video results, the trade-off against the effectiveness is its limited practicality due to its intensive computational cost and vulnerability to noisy optical flows. Recently, Wang *et al.* [9] proposed a deep learning based method for video inpainting, CombCN, by combining 3D and 2D CNNs. However, their setting works on low-resolution videos (*e.g.*, 128×128 pixels) with fixed square holes, limiting its application to general scenarios.

In this paper, we propose a deep feed-forward framework for video inpainting which performs two core functions: 1) temporal aggregation and 2) recurrent propagation of visible information over time. We use a set of sampled video frames as the reference to take visible contents to fill the hole of a target frame. With the recurrent propagation, we reuse the useful information from the previous time step and enforce temporal consistency. As real-world applications, we consider two types of distractor in a video: overlaid caption and unwanted foreground object. First, in the context of media and video from various languages, there are frequently text captions or encrusted commercials. These text overlays occlude parts of frames and hinder visual perception for both human and machines. In terms of a video caption removal problem, the holes are mostly narrow and have regularized locations and patterns. This makes it difficult to create mask annotations, but instead enables using the captions themselves as pseudo indicators for the holes. On the other hand, the task of foreground object removal deals with large and arbitrary holes with more motions involved, and the mask annotations can be obtained by human labeling or off-the-shelf video object segmentation algorithms.

To deal with two different video inpainting scenarios, we propose two network designs based on the same proposed framework. Our first model is a blind video decaptioning network (**BVDNet**) which is designed to automatically detect and inpaint overlaid captions in a video without any mask information (*i.e. blind* to input mask). Second, we propose a network for video object removal (**VINet**) which explicitly warps visible contents between frames to inpaint arbitrary and object-level holes indicated by input masks.

We conduct extensive experiments to validate the contributions

- *D. Kim, S. Woo and I.S. Kweon are KAIST, Daejeon, Korea. E-mail: {mcahny, shwoo93, iskweon77}@kaist.ac.kr*
- *J. Lee is with Adobe Research, San Jose, CA, USA. E-mail: jolee@adobe.com*

** Both authors have equally contributed.*
Manuscript received XXXX XX, XXXX; revised XXXX XX, XXXX.

of our design choices. We show that our formulation of temporal feature aggregation and recurrent propagation leads to the video results that are much more accurate and visually pleasing than existing frame-by-frame learning based methods (*e.g.* [5]), and similar to computation-heavy optimization methods (*e.g.* [8]). The example results by our proposed VINet are shown in the last row of Figure 10. Our model sequentially processes video frames of arbitrary length and requires no optical flow computation at test time.

Our contributions can be summarized as follows:

- We propose a novel video inpainting meta-architecture equipped with two core functions: temporal feature aggregation and recurrent propagation. Based on our 3D-2D encoder-decoder framework, we design a blind video decaptioning network (BVDNet) for caption removal, and a more general video inpainting network (VINet) for foreground object removal.
- Our BVDNet automatically detects and inpaints overlaid captions in a video without any mask information. Trained by the residual learning and our robust loss function, BVDNet outperforms other competing methods and runs in real time (50+ fps). We took the first place in the ECCV Chalearn 2018 LAP Video Decaptioning Challenge.
- Our VINet deals with arbitrary and large (object-level) holes. It learns to explicitly compensate motions and pick up visible contents from neighbor frames. We also propose to recurrently propagate information from the previous time step to enforce temporal consistency.

2 Related Work

2.1 Image and Video Inpainting

Significant progress has been made on image inpainting [5], [10], [11], [12], [13], [14], [15], [16], [17], [18], to the point where commercial solutions are now available [19]. However, video inpainting algorithms have been under-investigated. This is due to the additional time dimension which introduces major challenges such as severe viewpoint changes, temporal consistency preserving, and high computational complexity. Most recent methods found in the literature address these issues using either object-based or patch-based approaches.

In object-based methods, a pre-processing is required to split a video into foreground objects and background, and it is followed by an independent reconstruction and merging step at the end of algorithms. Previous efforts which fall under this category are homography-based algorithms that are based on the graph-cut [20], [21]. However, the major limitation of these object-based methods is that the synthesized content has to be copied from the visible regions. Therefore, these methods are mostly vulnerable to abrupt appearance changes such as scale variations, *e.g.* when an object moves away from the camera.

In patch-based methods, the patches from known regions are used to fill in a mask region. For example, Patwardhan *et al.* [22], [23] extend the well-known texture synthesis technique [12] to video inpainting. However, these methods either assume static cameras [22] or constrained camera motion [23] and are based on a greedy patch-filling process where the early errors are inevitably propagated, yielding globally inconsistent outputs.

To ensure the global consistency, patch-based algorithms have been cast as a global optimization problem. Wexler *et al.* [24] present a method that optimizes a global energy minimization problem for 3D spatio-temporal patches by alternating between patch search and reconstruction steps. Newson *et al.* [25] extend this by developing a spatio-temporal version of PatchMatch [19] to strengthen the temporal coherence and speed up the patch matching. Recently, Huang *et al.* [8] modify the energy term of [24] by adding an optical flow term to enforce temporal consistency. Although these methods are effective, their biggest limitations are high computational complexity and the absolute dependence upon the pre-computed optical flow which cannot be guaranteed to be accurate in complex sequences.

To tackle these issues, we propose a deep learning based video inpainting meta-architecture. To efficiently exploit temporal information coming from multiple frames, we construct a 3D encoder - 2D decoder model, that can provide traceable features revealed from the video dynamics. Based on the proposed framework, we design two video inpainting networks that can be applied to blind and non-blind video inpainting task respectively.

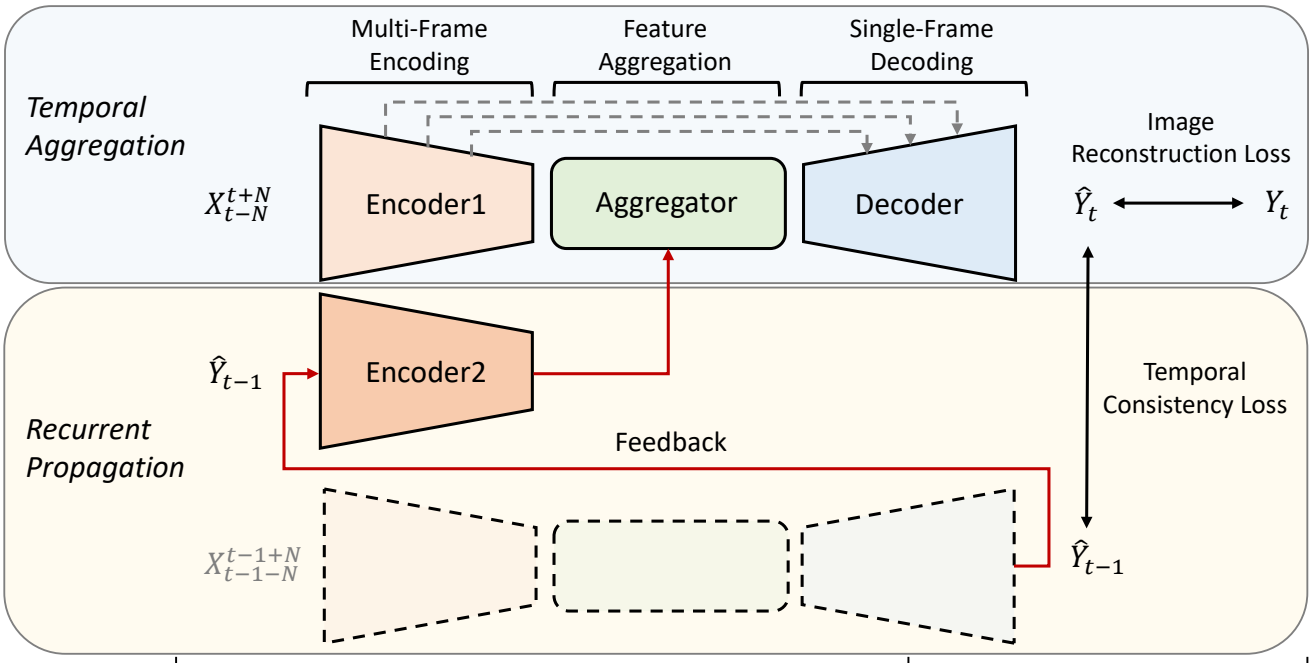
For the blind video inpainting task, we choose an example of video decaptioning task which is to inpaint text overlays in a "blind" manner. This is of the practical use because media/video data from various languages frequently include text captions or subtitles which reduce visual attention for both machine and human; Also, annotating pixel masks for every frames is impractical since many video subtitles include semi-transparent shadows (as in Figure 5) where it is ambiguous to label the pixels in binary. We show that our method successfully inpaints text overlays in videos without any mask information.

We then extend our framework to perform a general video inpainting task with the inpainting masks are given (*i.e.*, non-blind). The missing regions are more arbitrary and larger than the previous task. We argue that our method provides a better prospect than the previous optimization-based techniques in that deep CNNs are excellent at learning spatial semantics and temporal dynamics from an ever-growing vast amount of video data.

There is a relevant and concurrent work on deep video inpainting presented by Wang et al. [9]. However, we tackle more challenging and general scenarios. Wang et al. tested their method on the datasets (FaceForensics, 300Vw, Caltech) where each set has narrow semantic fdiversity and trivial camera movements. Such strong semantic and motion prior is not readily available in real-world videos. Also, they use a fixed and stationary mask over time, which is also not plausible in the real-world setting. On the other hand, we validate our method using videos *in the wild* from the Youtube-VOS dataset along with real dynamic object masks. Our experiment shows that joint function of temporal aggregation and recurrence is crucial to handle real dynamic contents.

2.2 Other Multi-frame Video Methods

Most multi-frame methods for video restoration [26], [27] or enhancement [7], [28] (e.g. video super-resolution, video interpolation) usually focus on pixel-level fusion with low-level and local motion compensation. On the other hand, video inpainting has to deal with semantic-level and object-sized holes, which requires a high-level understanding of context with a larger receptive field in both space and time. When compared to video super-resolution task (VSR), video inpainting should address the challenging flow synthesis problem because the pixel-level flow on the missing pixels cannot be linearly scaled as in (VSR) due to the large size and arbitrariness of the hole.



	Network Design			Objective Functions	
	Multi-Frame Encoding	Feature Aggregation	Single-Frame Decoding	Reconstruction Loss	Temporal Loss
BVDNet	Encoder1: 3D Conv Encoder2: 2D Conv	Element-Wise Sum	In-Out Residual & Skip-Connections	L1 + SSIM + Grad. L1	Short-term
VINet	Encoder1, Encoder2: 2D Siamese Convs	Feature Flow Warping & Composition	Skip-Connections	L1 + SSIM + Grad. L1	Short-term Long-term Flow-warping

Fig. 1: **Overview of the proposed meta-architecture.** The skip connections, recurrent feedback, and the used objective functions are denoted by gray, red, and black arrows, respectively. During inference, we apply the model in an auto-regressive manner to obtain output sequences.

To this end, we propose a non-trivial well designed system for video inpainting. We identify the key functions: content generation, hierarchical feature transfer, motion detection, and coherency enforcement. We then incorporate them into a single pipeline, achieving an end-to-end architecture. Our method is comparably effective to the optimization based method, and superior to the per-frame deep inpainting method [5], while having fast computation speed (i.e. 50fps for decaptioning, and 10fps for inpainting).

3 Problem Formulation

Video inpainting aims to fill in arbitrary missing regions in video frames $X_1^T := \{X_1, X_2, ..., X_T\}$. The reconstructed regions should be either accurate as in the ground truth frames $Y_1^T := \{Y_1, Y_2, ..., Y_T\}$ or seamlessly merged into the surrounding space and time. We formulate video inpainting as a problem of learning a mapping function from X_1^T to the prediction $\hat{Y}_1^T := \{\hat{Y}_1, \hat{Y}_2, ..., \hat{Y}_T\}$, where the conditional distribution $p(\hat{Y}_1^T|X_1^T)$ should be identical to $p(Y_1^T|X_1^T)$. Learning this mapping leads to the realistic and temporally consistent video generation. To simplify the problem, we factorize the conditional distribution to a product form based on a Markov assumption. Accordingly, the naive *frame-by-frame* method can be represented as

$$p(\hat{Y_1^T}|X_1^T) = \prod_{t=1}^{T} p(\hat{Y}_t|X_t). \quad (1)$$

However, for the video results to be consistent in both space and time, we propose that the generation of t-th frame $\hat{Y}_t$ should be coherent with 1) the neighboring pixels X_{t-N}^{t+N} within a temporal radius N, and 2) the previously generated frame $\hat{Y}_{t-1}$. Thus, we propose to learn the conditional distribution

$$p(\hat{Y_1^T}|X_1^T) = \prod_{t=1}^{T} p(\hat{Y}_t|X_{t-N}^{t+N}, \hat{Y}_{t-1}). \quad (2)$$

4 Method

Figure 1 provides an overview of our proposed meta-architecture f which consists of two parallel pathways for temporal aggregation and recurrent propagation, respectively. For *temporal aggregation*, we use a set of sampled frames as the reference to pick up visible contents to fill the hole of a target frame. We set the radius N to 2, such that we use two lagging and two leading frames as the reference at each time step. To maximize useful temporal information, we attempt to find the optimal sampling interval. With the minimum interval of 1, the reference frames will contain non-significant changes and be redundant. If we jump

with a large stride, on the other hand, irrelevant new scenes will be included. We empirically find that the stride of 3 performs the best in our preliminary experiments. That is, we sample $X_{t-N}^{t+N} := \{X_{t-6}, X_{t-3}, X_t, X_{t+3}, X_{t+6}\}$ at each time step, which gives the temporal window spanning over about 15 frames. For *recurrent propagation*, we connect the output $\hat{Y}_{t-1}$ from the previous time step $t-1$ to the current time step t via a feedback. This is to reuse the previously collected cues, and to enforce temporal consistency in the video results. Therefore, a tuple of $(X_{t-N}^{t+N}, \hat{Y}_{t-1})$ constructs the total inputs at each time step.

At inference, the output video $\hat{Y}_1^T$ is obtained by sequentially running the function f in an auto-regressive manner. Our formulation of multi-to-single frame aggregation and recurrent propagation works as a backbone structure (meta-architecture) for our downstream network designs: a blind video decaptioning network (BVDNet) in Section 4.1, and a video inpainting network (VINet) for foreground object removal in Section 4.2.

4.1 BVDNet for Video Caption Removal

In the context of an inpainting problem, the overlaid captions in a video create holes that are mostly narrow and systematic in their locations and patterns. We exploit such regularity as an indicator for the corrupted regions in video frames, and construct the input with a stack of corrupted RGB frames without any mask information (3-channel). The overall decaptioning algorithm is illustrated in Figure 2.

Aggregation via 3D convolutions. A stack of reference frames and a target frame is fed into the aggregation pathway. Since most captions in a video are not continuously moving, it is efficient to use 3D convolutions to directly aggregate spatio-temporal features in a search window, without serious motion compensation between frames. Another pathway consists in 2D CNN, and takes an output frame from the previous time step to recurrently propagate useful information throughout time.

Residual Learning. Directly estimating all pixels in a frame may needlessly touch uncorrupted pixels. To compensate the absence of the mask information, we train our model by a residual learning algorithm. Specifically, the final output is yielded by summing the input target frame $\{X_t\}$ and the predicted residual image $\{R_t\}$ in a pixel-wise manner. This trains our network to automatically detect corrupted pixels, and also prevent the global tone distortion.

Formally, with the proposed BVDNet f_{BVD}, the blind video decaptioning problem can be modeled as

$$\hat{Y}_t = f_{BVD}(X_{t-N}^{t+N}, \hat{Y}_{t-1}) + X_t. \quad (3)$$

4.1.1 Network Design

Our core design is a hybrid encoder-decoder model, where the encoder consists of two sub-networks: 3D CNN and 2D CNN. The decoder follows a normal 2D CNN design as in other image generation networks. The network is designed to be fully convolutional to handle arbitrary size input. The final output video is obtained by applying f_{BVD} in an auto-regressive manner.

3D-2D Hybrid Encoder. Our strategy is to collect potential hints from multiple reference frames that can provide visible pixels revealed from the scene dynamics. Also, we enforce the generation of the target frame to be consistent with the previous generation. We construct a hybrid encoder consisting of two streams: *temporal aggregation* stream and *recurrence* stream. The first stream consists in 3D convolutions which can directly capture spatio-temporal features from the neighbor frames. This helps in understanding the short-term video-level context which is required to recover the target frame. The input tensor shape is $H \times W \times T \times C$, where H, W and C are the height, width and channels of the input frame $\{X\}$, and $T = 2N + 1$. Here, the goal is to remove text overlays in the middle frame (3th out of 5). The temporal dimension of feature gradually reduces to 1 as the features are temporally pooled through the 3D convolution layers.

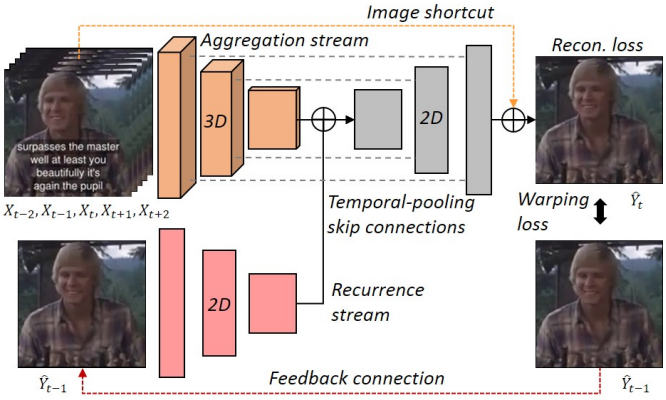


Fig. 2: **Overview of BVDNet: blind video decaptioning network.** We propose a hybrid encoder-decoder model, where the aggregation encoder stream takes multiple input frames and the decoder reconstructs the middle frame. The temporal-pooling skip connections carry low-level information. By a residual learning algorithm, our model directly learns to recover the corrupted pixels in the input. The output is then fed into a feedback connection for a recurrent learning to the next time step.

The second is a 2D CNN recurrence stream which takes the previously generated frame, of size $H \times W \times 1 \times C$, as input. This stream works as a reference that the current generation should be coherent with. Then, the output feature of this recurrence stream is combined with the *temporally-pooled one-frame* feature from the aggregation stream by element-wise summation. Since the feature maps from the two streams are comparatively different on the corrupted regions, the combined hybrid feature map implicitly encodes the knowledge on where to attend.

Bottleneck and Time-Pooling Skip Connections. The encoder is followed by bottleneck layers that consist of several dilated convolutions, as suggested in [16]. The large receptive field size helps to capture wide spatial context which supports the recovery of the corrupted pixels. The following is a 2D CNN decoder which is symmetric to the recurrence encoder stream.

We apply skip connections only between the 3D encoder stream and the decoder. Each skip connections pass through a 3D convolution layer that pools the temporal dimension into *one frame*, so that the feature map can be directly concatenated with the decoder feature maps of equal dimension. Despite the concern raised by Yu *et al.* [18] that the skip connections carry almost zero features on the corrupted regions, our temporal-pooling skip connections are immune to this problem since they can adaptively aggregate low-level features that are complementary to the occluded points in the middle frame.

Our full BVDNet is trained to generate the residual frame $\{R_t\}$, which is added with the input middle frame $\{X_t\}$ to produce the final output $\{\hat{Y}_t\}$.

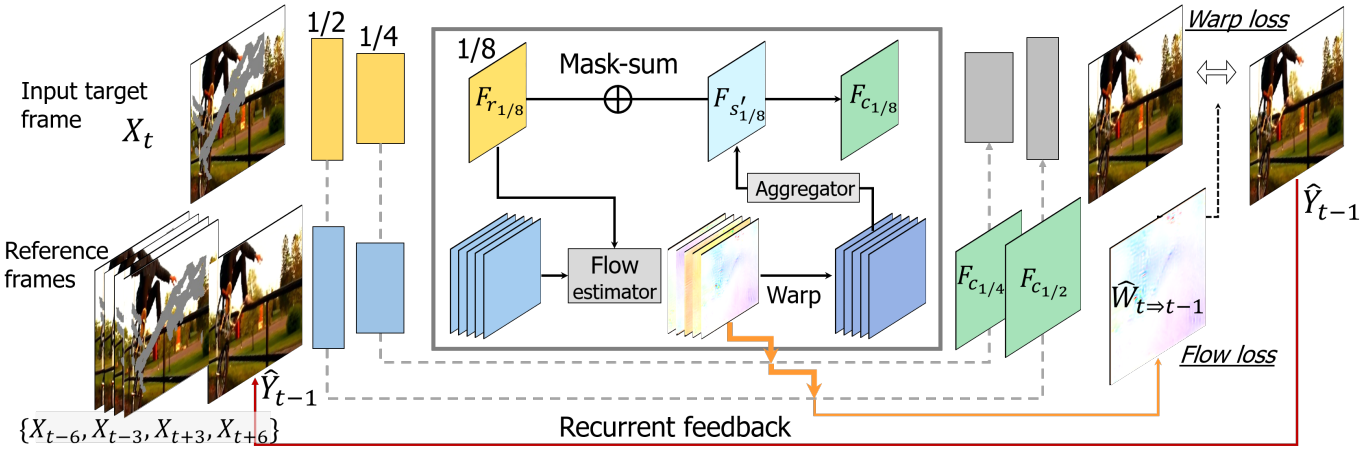


Fig. 3: **Overview of VINet.** Our proposed VINet takes in multiple input frames (X_{t-6}, X_{t-3}, X_t, X_{t+3}, X_{t+6}) and the previously generated frame ($\hat{Y}_{t-1}$), and generates the inpainted frame ($\hat{Y}_t$) as well as the flow map ($\hat{W}_{t\Rightarrow t-1}$). We employ both flow sub-networks and mask sub-networks at 4 scales (1/8, 1/4, 1/2, and 1) to aggregate and synthesize feature points progressively. For temporal consistency, we use a recurrent feedback along with two losses: flow loss and warp loss. The orange arrows denote the $\times 2$ upsampling for residual flow learning as in [29] for 5 reference streams, while the thinner orange arrow denotes only the stream from $\hat{Y}_{t-1}$. The mask sub-networks are omitted in the figure for the simplicity.

4.1.2 Loss Functions

We train the BVDNet by minimizing the following loss function,

$$\mathcal{L} = \lambda_R \mathcal{L}_R + \lambda_{st} \mathcal{L}_{st}, \tag{4}$$

where $\mathcal{L}_R$ is an **image reconstruction loss**, and $\mathcal{L}_{st}$ is a **temporal warping loss**. λ_R and λ_{st} are the weighting coefficients which are set to 1 and 2 respectively throughout the experiments.

A simple way for image reconstruction is to use the L1 loss following the previous studies [5], [17]. For the structural details, We apply the SSIM loss [30] with a small patch window according to the setting of the competition evaluation metric. Inspired by [31], we also use a first-order matching term, which compares image gradients of the prediction with the ground truth, and encourages the prediction to have not only close-by values but also similar local structure. To this end, the **image reconstruction loss** $\mathcal{L}_R$ includes three terms as

$$\mathcal{L}_1 = \left\| \hat{Y}_t - Y_t \right\|_1, \tag{5}$$

$$\mathcal{L}_{SSIM} = \left(\frac{(2\mu_{\hat{Y}_t}\mu_{Y_t} + c_1)(2\sigma_{\hat{Y}_t Y_t} + c_2)}{(\mu^2_{\hat{Y}_t} + \mu^2_{Y_t} + c_1)(\sigma^2_{\hat{Y}_t} + \sigma^2_{Y_t} + c_2)}\right), \tag{6}$$

$$\mathcal{L}_{grad.} = \left\| \nabla_W (\hat{Y}_t - Y_t) \right\|_1 + \left\| \nabla_H (\hat{Y}_t - Y_t) \right\|_1, \tag{7}$$

$$\mathcal{L}_R = \mathcal{L}_1 + \mathcal{L}_{SSIM} + \mathcal{L}_{grad.}, \tag{8}$$

where $\hat{Y}_t, Y_t$ denote the predicted and target ground truth frames respectively. μ, σ denote the average, variance. c_1, c_2 denote two stabilization constants which are respectively set to $0.01^2, 0.03^2$. ∇_W, ∇_H are the image gradients along the horizontal and vertical axis.

With the recurrence stream in the encoder, we optimize our model with additional temporal warping loss which is widely used in video generation works [32], [33], [34]. The **temporal consistency loss** $\mathcal{L}_{st}$ is defined as

$$\mathcal{L}_{st} = \sum_{t=2}^{K} \left\| O_{t\Rightarrow t-1} \odot (\hat{Y}_t - \phi_{t\Rightarrow t-1}(Y_{t-1})) \right\|_1, \tag{9}$$

where $\odot$ is the element-wise product operator. O denotes the binary occlusion mask and ϕ denotes the flow warping operation using the optical flow between consecutive target frames Y_t and Y_{t-1}. The occlusion is detected by the method of [35] and the optical flow are obtained by the pretrained FlowNet2 [36]. For the training, we set the number of recurrences to 5 ($K = 5$).

4.1.3 Training and Testing

Dataset. We used the ECCV Chalearn 2018 LAP Video Decaptioning Challenge dataset for training, validation, and testing. It is a large dataset of 5 seconds (125 frames) MP4 video clips in 128×128 pixel RGB frames, containing both encrusted subtitles ($\{X\}$) and without subtitles ($\{Y\}$). The dataset contains a wide variety of captions with different colors, size, positions, and shadows. The training and validation set consist in 70K and 5K sample pairs of input and ground truth video clips, respectively. The testing set consists of 5K input video clips without ground truth. We convert every video clip into PNG images in our experiments.
Training. We adopt horizontal flipping and color jittering for data augmentation. We train our model for 200 epochs with a batch size of 128. Adam optimizer is used with β = (0.9, 0.999) and a learning rate of 0.001. The training takes 3 days on two NVIDIA GTX 1080 Ti GPUs. For the competition, we train our model without the recurrence stream in the encoder and the warping loss.
Testing. For the pixels where the absolute difference between the input middle frame $\{X_t\}$ and the prediction $\{\hat{Y}_t\}$ is less than 0.01 in $[0, 1]$ scale, we copy the values from the input frame. Finally, we convert PNG files back to MP4 videos.
Evaluation Metric. To evaluate the quality of the reconstruction, the mean square error (MSE), the peak signal-to-noise ratio (PSNR), and the structural dissimilarity, DSSIM *i.e.* (1-SSIM)/2, are used.

4.2 VINet for Video Object Removal

Removing foreground objects in a video involves arbitrary and large holes with various motions. In contrast to the video caption

removal, we assume that the inpainting masks are given for all video frames, and construct the input by concatenating each RGB frame with a corresponding mask channel. To deal with these differences, we propose several modifications to the network design and the training scheme. We train VINet to explicitly fetch, arrange, and combine visible feature points from neighbor reference frames. The overview of our VINet is illustrated in Figure 3.

Modification 1: Aggregation via Explicit Feature Alignment. Instead of relying on the 3D convolutions with the input frame stack, VINet learns to align each of the reference frames onto the target frame in a feature level. The visible patches from the reference feature maps are then picked up and aggregated onto the missing regions of the target feature map. To this end, we use flow and mask sub-networks to learn the flow and composition mask between the frames, respectively.

Modification 2: Recurrence. Another modification is made on the recurrence. Instead of having a separate pathway until the bottleneck, the previous prediction frame is fed into the aggregation pathway and is considered as another reference frame for the next time step.

In the following, we provide more detailed description on the network design, training, and testing of our VINet and the differences to BVDNet.

4.2.1 Network Design

Encoder. The major difference resides in the encoder part which is a multiple-tower network. All the towers consist in the 2D CNN and represent the target and reference streams. We consider the center frame as the target, and the other frames as the references which can provide visible contents to the target. All input frames are concatenated with their corresponding masks along the channel axis, and fed into each of the streams. In practice, we use a 6-tower encoder; There are 1 target stream that takes the center target frame X_t, and 5 weight-sharing reference streams that each take two lagging (X_{t-6}, X_{t-3}), two leading frames (X_{t+3}, X_{t+6}), and the previous prediction $(\hat{Y}_{t-1})$. The reference feature points are explicitly copy-pasted and refined through the following *feature flow learning* and *learnable feature composition*.

Feature Flow Learning. Before directly combining the target and reference features, we propose to explicitly align the feature points from each streams. This helps our model easily borrow traceable features from the neighbor reference frames. To this end, we insert flow sub-networks to estimate the flows from each of the reference feature maps onto the target feature map in four different scales (1/8, 1/4, 1/2, and 1). We adopt the coarse-to-fine structure of the PWCNet [29] to model this hierarchical flow learning. For the aggregation part, the feature flow warping for the first three scales is supervised only by pixel reconstruction loss, *i.e.*, good feature warping will successfully pick up visible contents and inpaint the target hole. The recurrence part involves the explicit flow supervision (warp loss and flow loss in Section 4.2.2) which is only given at the finest scale (*i.e.* 1) and *only between* the consecutive two predictions, where we use the pseudo-ground truth flow $W_{t\Rightarrow t-1}$ between Y_t and Y_{t-1} obtained from FlowNet2 [36].

Learnable Feature Composition. Given the aligned feature maps from the five reference streams, they are concatenated along the time dimension and fed into a $5 \times 3 \times 3$ (THW) convolution layer that produces a spatio-temporally aggregated feature map $F_{s'}$ with the time dimension of 1. This is designed to dynamically select reference feature points across the time axis, by highlighting the features complementary to the target features and ignoring otherwise. For each 4 scales, we employ a mask sub-network to combine the aggregated feature map $F_{s'}$ with the reference feature map F_r. The mask sub-network consists of three convolution layers and takes the absolute difference of the two feature maps $|F_{s'} - F_r|$ as input and produces single channel composition mask m, as suggested in [37]. By using the mask, we can gradually combine the warped features and the reference features. At the scale of 1/8, the composition is done by

$$F_{c_{1/8}} = (1 - m_{1/8}) \odot F_{r_{1/8}} + m_{1/8} \odot F_{s'_{1/8}}. \quad (10)$$

Decoder. To pass image details to the decoder, we employ skip connections as in U-net [38]. To prevent the concern raised by [18] that skip connections contain zero values at the masked region, our skip-connections pass the composite features similarly to Equation (10), as

$$F_{c_{1/4}} = (1 - m_{1/4}) \odot F_{r_{1/4}} + m_{1/4} \odot F_{s'_{1/4}}, \quad (11)$$

$$F_{c_{1/2}} = (1 - m_{1/2}) \odot F_{r_{1/2}} + m_{1/2} \odot F_{s'_{1/2}}. \quad (12)$$

At the finest scale, the estimated optical flow $\hat{W}_{t\Rightarrow t-1}$ is used to warp the previous output $\hat{Y}_{t-1}$ to the current raw output $\hat{Y'}_t$. We then blend this warped image and the raw output with the composition mask m_1, to obtain our final output $\hat{Y}_t$ as

$$\hat{Y}_t = (1 - m_1) \odot \hat{Y'}_t + m_1 \odot \hat{W}_{t\Rightarrow t-1}(\hat{Y}_{t-1}). \quad (13)$$

4.2.2 Loss Functions

We train our network to minimize the following loss function,

$$\mathcal{L} = \lambda_R \mathcal{L}_R + \lambda_{st} \mathcal{L}_{st} + \lambda_{lt} \mathcal{L}_{lt} + \lambda_F \mathcal{L}_F, \quad (14)$$

where $\mathcal{L}_R$ and $\mathcal{L}_W$ are the reconstruction loss, and warping loss respectively, as in Section 4.1.2.

The difference in the warping loss is that $\mathcal{L}_W$ includes not only the short-term warping loss $\mathcal{L}_{st}$, but also the long-term warping loss $\mathcal{L}_{lt}$ as

$$\mathcal{L}_{st} = \sum_{t=2}^{K} \left\| O_{t\Rightarrow t-1} \odot (\hat{Y}_t - \phi_{t\Rightarrow t-1}(Y_{t-1})) \right\|_1, \quad (15)$$

$$\mathcal{L}_{lt} = \sum_{t=2}^{K} \left\| O_{t\Rightarrow 1} \odot (\hat{Y}_t - \phi_{t\Rightarrow 1}(Y_1)) \right\|_1. \quad (16)$$

Similar to Equation (9), we use [35] to obtain the occlusion mask (O) and FlowNet2 [36] to extract the optical flow between the target frames (W). ϕ denotes the warping operation. We use both short-term and long-term temporal losses. Note that we use ground truth target frames in the warping operation since the synthesizing ability is imperfect during training. We set the number of recurrence to 5 ($K = 5$) as in Section 4.1.

We learn the explicit flow learning on the consecutive output frames. The flow loss $\mathcal{L}_F$ is defined as

$$\mathcal{L}_F = \sum_{t=2}^{T} \left(\left\| W_{t\Rightarrow t-1} - \hat{W}_{t\Rightarrow t-1} \right\|_1 + \left\| Y_t - \hat{\phi}_{t\Rightarrow t-1}(Y_{t-1}) \right\|_1 \right), \quad (17)$$

where $W_{t\Rightarrow t-1}$ is the pseudo-ground truth backward flow between the target frames, Y_t and Y_{t-1}, extracted by FlowNet2 [36]. In Equation (17), the first term is the endpoint error between the ground truth and the estimated flow, and the second term is the warping error when the estimated flow, $\hat{W}_{t\Rightarrow t-1}$, is used to warp the previous target frame to the next target frame.

4.2.3 Two-Stage Training

We employ a two-stage training scheme to gradually train the core functions for video inpainting; 1) We first train the model without the recurrent feedback to focus on learning the temporal feature aggregation. At this stage, we only use the reconstruction loss $\mathcal{L}_R$; 2) We then add the recurrent feedback, and fine-tune the model using the full loss function (Equation (14)) for temporally coherent generation. We use videos in the Youtube-VOS dataset [39] as ground truth for the training. It is a large-scale dataset for video object segmentation containing 4000+ YouTube videos with 70+ common objects. All video frames are resized to 256×256 pixels for training and testing. We further finetune VINet on 512×512 pixels frames to process higher resolution videos.

Video Mask Dataset. In general video inpainting, the spatio-temporal holes consist in diverse motion and shape changes. To simulate this complexity during training, we create the following four types of video masks.

1) Random square: We randomly mask a square box in each frame. The visible regions each of input frames are mostly complementary so that the network can clearly learn how to align, copy, and paste neighboring feature points.
2) Flying square: The motion of the inpainting holes is rather regularized than random in real scenarios. To simulate such regularity, we shift a square by a uniform step size in one direction across the input frames.
3) Arbitrary mask: To simulate diverse hole shapes and sizes, we use the irregular mask dataset [17] which consists of random streaks and holes of arbitrary shapes. During training, we apply random transformations (translation, rotation, scaling, sheering).
4) Video object mask: In the context of the video object removal task, masks with the most realistic appearance and motion can be obtained from video object segmentation datasets. We use the foreground segmentation masks of the YouTube-VOS dataset [39].

Synthetic training Data. We create the training video data to simulate the object removal scenarios. In particular, we overlay the aforementioned different types of masks onto the background video frames. The overlaid region is then filled with zero values to simulate the foreground object that has been taken away. The original background pixels are considered as the ground truth values during training. We construct the input by concatenating the corrupted frames and the corresponding binary masks.

4.2.4 Testing

We assume that the inpainting masks for all video frames are given. To avoid any data overlap between training and testing, we obtain object masks from the DAVIS dataset [40], [41], the public benchmark dataset for video object segmentation. It contains dynamic scenes, complex camera movements, motion blur effects, and large occlusions. The inpainting mask is constructed by dilating the ground truth segmentation mask. Our method processes frames recursively in a sliding window manner similarly to the BVDNet.

5 Experimental Results

We evaluate our video inpainting framework and its two downstream network designs both quantitatively and qualitatively. We conduct ablation studies to validate the effectiveness of the different design components. We measure visual quality and temporal smoothness of their video results, and conduct user studies to compare human subjective preferences on different methods.

5.1 BVDNet Results on Video Caption Removal

We conduct ablation studies using the publicly released validation dataset to investigate the effectiveness of different design components. Our evaluation is mainly based on the metrics used in the competition. We report the competition results and visualize the learned convolutional filters.

5.1.1 The Impact of 3D Aggregation Encoder Stream

One of our core design choices is to use a 3D CNN aggregation encoder stream in conjunction with a following 2D decoder. To validate the effectiveness of this design, we construct two naive baselines to compare with: a 3D encoder-3D decoder and a 2D encoder-2D decoder models. We note that all models in this experiment contain a single-stream encoder without the recurrence encoder stream for a clearer comparison. We construct all the models with a comparable number of parameters. As shown in Exp 1, 2, and 3 in Table 1, our 3D-2D model shows the best performances in all three metrics. This implies that the spatio-temporal feature aggregation from the neighbor frames indeed helps our target task, providing our model with a distinct advantage over the *frame-by-frame* competitor. On the other hand, it is emprically shown that adopting heavy 3D-3D operation does more harm than good. This implies that making use of the neighbor frames does not always work, but a careful architectural design is required.

5.1.2 The Impact of Loss Functions

We test our loss functions both quantitatively and qualitatively. First, we remove each loss terms gradually from our full loss function. Again, we use a model with the single-stream encoder, and thus the temporal warping loss is not considered in this experiment. As shown in Exp 3, 4, 5 and 6 in Table 1. We observe complementary effects of each loss terms. The grad.L1 loss improves the performance when used together with L1 loss. Also, adding SSIM loss improves the structural similarity score (DSSIM) more than adding the gradient loss does. This leads us to use our full combination (*i.e.* $L_1 + L_{grad.} + L_{SSIM}$), which achieves best scores.

We also provide qualitative analysis as shown in Figure 4. The model trained with the L1 loss alone produces relatively blurry outputs. We alleviate this problem by adding the gradient L1 loss and the SSIM loss. We observe that adding these losses helps recovering fine structures such as texture and edges.

5.1.3 The Impact of Recurrence Encoder Stream

We investigate the effectiveness of our recurrence stream in the encoder, together with the temporal warping loss. We evaluate both frame-level image quality and temporal consistency. As shown in Exp 5 and 6 in Table 1, the recurrence stream improves the visual quality of the video results. In addition, we quantitatively compare the temporal consistency in video results with and without the recurrence stream. We measure the temporal error over a video sequence, which is the average pixel-wise Euclidean color difference between consecutive frames. We use FlowNet2 [36] to obtain pseudo-ground truth optical flow as in the training. Table 4 shows that having the recurrence significantly reduces

Exp	Architecture			Losses			Recurrence	Evaluation Metric		
	3D-3D	2D-2D	3D-2D	L1	grad. L1	SSIM	Enc. Stream	MSE	PSNR	DSSIM
1	✓			✓				0.0031	28.4590	0.0652
2		✓		✓				0.0012	33.6803	0.0279
3			✓	✓				0.0011	34.1029	0.0261
4			✓	✓	✓			0.0010	34.2251	0.0276
5			✓	✓		✓		0.0011	34.2089	0.0240
6			✓	✓	✓	✓		0.0010	34.6544	0.0225
7 (**Full BVDNet**)			✓	✓	✓	✓	✓	**0.0010**	**34.7055**	**0.0222**

TABLE 1: The ablation studies on architectural design, loss functions, and recurrence stream. We evaluate on ChaLearn 2018 LAP Inpainting Track2 *validation* set.

	Ours	+ GAN loss	- Skip
MSE	**0.0010**	0.0015	0.0010
PSNR	**34.7055**	31.2257	34.3892
DSSIM	**0.0222**	0.0384	0.0233

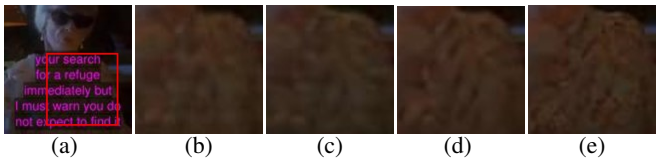
TABLE 2: The ablation studies on additional GAN loss and without residual learning. We evaluate on ChaLearn 2018 LAP Inpainting Track2 *validation* set.

	Value	MSE	PSNR	DSSIM
Number of frames	3	0.0011	33.7895	0.0247
	5	**0.0010**	**34.7055**	**0.0222**
	7	0.0010	34.5063	0.0229
	9	0.0010	34.6260	0.0226

TABLE 3: The ablation studies on the hyperparamter: *number of input frames*. We evaluate on ChaLearn 2018 LAP Inpainting Track2 *validation* set.

Encoder version	Temporal Errors
without recurrence (**BVDNet-Exp 6**)	0.00117
with recurrence (**BVDNet-Exp 7**)	**0.00090**

TABLE 4: Temporal errors (warping errors) of BVDNet with and without the temporal consistency constraint. We evaluate on 500 clips of ChaLearn 2018 LAP Inpainting Track2 *validation* set.



(a) (b) (c) (d) (e)

Fig. 4: **The impact of each loss terms.** (a) An input center frame. (b-d) The reconstructed frames with: (b) L_1 loss, (c) $L_1 + L_{grad.}$ loss (d) $L_1 + + L_{grad.} + L_{SSIM}$ loss (e) Ground truth frame. *Best viewed when zoomed-in*.

the temporal error. Our approach does not sacrifice either visual quality and temporal stability, and the qualitative examples are shown in Figure 5.

5.1.4 Adding GAN Loss

The adversarial training encourages the decaptioning results to move towards the natural image manifold. We test the effect of the adversarial training by adding the GAN loss on top of our full loss function. We use 8×8 PatchGAN [42] as our discriminator network which aims to classify whether 8×8 overlapping image patches are real or fake. However, we observe no visible qualitative improvement and the quantitative performance slightly dropped (Table 2), which is consistent with the results in [43].

5.1.5 Removing Residual Image Shortcut

We investigate the importance of the residual learning. If we remove the skip connection from the input middle frame to the decoder output, the network should recover all the pixels from scratch without referencing the input pixels. As shown in Table 2, the residual learning leads to better performances, demonstrating its effectiveness in video decaptioning task.

5.1.6 Number of Input Frames

In Table 3, we perform an experiment to determine the hyperparameter for our BVDNet, which is the number of input frames. The *number of input frames* directly relates to the size of input batches, which enables controlling the amount of temporal information to be dealt with for each time step. Table 3 shows the comparison results with four different input frame values. We observe an overall tendency of better performances with larger number of input frames, while the value of 5 gives the base results. This indicates that having a proper temporal view range is crucial for our target task.

5.1.7 Model Inference Time

Our BVDNet has a total of 23 layers and 10.5M parameters. Our model is implemented on Pytorch v0.3, CUDNN v6.0, CUDA v8.0, and runs on the hardware with Intel(R) Xeon(R) (2.10GHz) CPU and NVIDIA GTX 1080 Ti GPU. The model runs at 62.5 fps on a GPU for frames of resolution 128×128 pixel.

5.1.8 Final Challenge Results

Quantitative results. Table 5 summarizes the top entries from the leaderboard of ECCV ChaLearn 2018 Inpainting Challenge Track2. We participated with our BVDNet *without* the recurrence stream and achieved the first place on the final test phase. Our full BVDNet is shown to be even stronger on the validation set in Table 4, but we cannot provide the evaluation on the testing set because the test server is closed.

Qualitative results. We visualize the learned feature maps of our BVDNet in Figure 6. We observe a hierarchical attention where the encoder features are highly activated on the surrounding pixels, while the decoder features are more *attending* to the corrupted regions. We visualize the learned feature maps of our BVDNet in Figure 6. We observe a hierarchical attention where

Fig. 5: **The impact of recurrence on temporal consistency.** For each sample, we visualize four consecutive input frames in the top row. In the bottom rows are the zoomed-in views of our results *without recurrence* (2nd row), *with recurrence* (3rd row), and the ground truth frames (4th row). Without the recurrence, the change in the subtitles leads to temporally flickering artifacts (a)

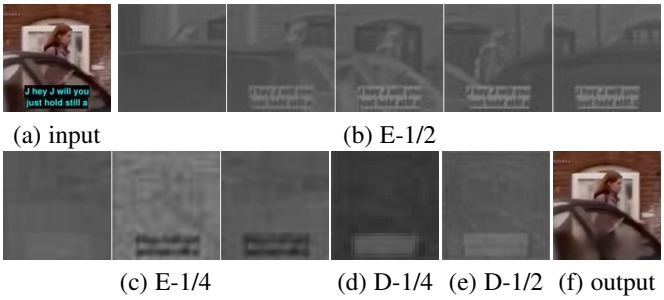


Fig. 6: **Visualization of learned feature activation.** E and D denote the encoder and decoder layers, respectively. For the visualization, we average each feature maps along the channel axis, and up-sample to 128×128 pixel. The fractional numbers denote the spatial resolutions. We observe hierarchical attention operations across layers. In the early encoder layers (b, c), low-level features such as background textures (*e.g.around the subtitles*) are aggregated along the time dimension. The latter decoder layers (d, e) then gradually focus on the exact target regions.

the encoder features are highly activated on the surrounding pixels, while the decoder features are more *attending* to the corrupted regions.

Figure 7 shows several examples of our decaptioning results. Our full model successfully recovers the fine details and textures with smooth temporal transition, even when there are active object movements, *e.g.* Figure 7-(a), or heavy illumination changes, *e.g.* Figure 7-(b). When there are texts in a video as the content

	MSE	PSNR	DSSIM
stephane	0.0022	30.1856	0.0613
hcilab	0.0012	33.0228	0.0424
anubhap93	0.0012	32.0021	0.0499
arnavkj95	0.0012	32.1713	0.0482
BVDNet-Exp 6	**0.0011**	**33.3527**	**0.0404**

TABLE 5: Final performances of the top entries in the ECCV ChaLearn 2018 LAP Inpainting Challenge Track2 **test phase**. We note that stephane's is the baseline from the organizers [44].

themselves, *e.g.* Figure 7-(c), our model is trained to separate between the overlaid captions and the ones coming from the video; This is an intended action since the video content itself should be preserved. In the future, we can construct synthetic training data to nearly unlimited amount to include more various real-world captions. More qualitative video results can be found in the supplementary materials.

5.1.9 *User study on video caption removal results*

As both quantitative and qualitative studies have their limitations in evaluating image / video quality, we conduct a user study to see the human preferences between our video results and the ground truth using 25 randomly selected videos. We exclude the videos containing solid shadow regions. In each comparison, we show the input video, our result, and the ground truth. The display order of our result and the ground truth is randomly shuffled. Each participant is asked to choose a preferred video or equally good. A total of 30 participants aged from 25 to 35 participated in this study. The user study results are summarized in Figure 8. In order to provide the statistical significance, we conduct F-test and T-test on our survey result. F-test gives a two-tailed p-value of 0.21, making us assume the equal variances. Then, on the assumption that the mean preference between ours and the ground truth is the same (null hypothesis), we obtain a two-tailed p-value of 0.83, which fails to reject the null hypothesis. This implies that the human preference between our video results and the ground truth are statistically similar.

5.1.10 *Limitation*

We observe that the results are relatively blurry when the input frames have solid shadow regions, as in Figure 7-(d). This is because these shadows completely occlude the background pixels and make larger holes. Not only the given training set lacks the amount of samples with such large holes (solid shadows), but also the feature aggregation of our BVDNet relies largely on implicit 3D conv operations, so it is not enough to estimate the motion behind the large holes.

5.2 VINet Results on Video Object Removal

In this section, we visualize the learned temporal aggregation mechanism and show the effectiveness of aggregation and recurrence pathways.

Baselines. We compare our approach to state-of-the-art baselines in three representative streams of study: deep image inpainting [5], deep video inpainting [9], and optimization based video inpainting [8].

- Yu *et al.* [5]: A feed-forward CNN based method, which is designed for single image inpainting. We processes videos

Fig. 7: **Qualitative decaptioning results.** For each example, the top rows are the input sequences and the bottom rows are the decaptioning results using our full model. For visualization, we determine the time interval between the frames to be 0.1 seconds. Our model performs well on various types of subtitles with complex background variations and also is able to separate the non-caption texts in a video.

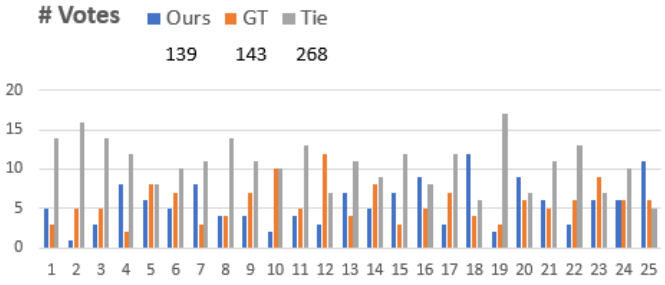


Fig. 8: **User study on our BVDNet results vs. ground truth.**

frame-by-frame without using any temporal information. We run their official test codes.

- Wang *et al.* [9]: A feed-forward CNN based method for video inpainting, CombCN, which consists of a temporal structure inference network, and a spatial detail recovering network. We re-implemented CombCN [9], since their code is not publicly available.
- Huang *et al.* [8]: An optimization-based video completion method, which jointly estimates global flow and color. It requires on-the-fly optical flow computation and is extremely time-consuming. We run their official test codes.

5.2.1 Visualization of Learned Feature Composition

Figure 9 shows that the proposed VINet explicitly borrows visible neighbor features to synthesize the missing content. For the visualization, we use the VINet of the first training stage and plot the learned feature flow from the four reference streams (without the recurrent feedback) to the target stream, at 128×128 pixel resolution. We observe that even with a large and complex hole in the target frame, VINet is able to align the reference feature maps with respect to the target, and integrate them to fill in the hole. Even without explicit flow supervision, our flow sub-networks are able to warp the feature points in visible regions while shrinking the unhelpful zero features in masked regions. Moreover, these potential hints are adjusted according to the spatio-temporal context, rather than copied-and-pasted in a fixed manner. One example is shown in Figure 9-(b) where the eyes of the hamster are synthesized *half-closed*.

5.2.2 Improvement on Temporal Consistency

We first provide self-comparison which shows the temporal consistency of the video results before and after using the recurrence stream. Also, to validate the competitiveness of our method, we compare with the three representative baseline methods [5], [8], [9]. To provide the quantitative evaluation, we measure *flow warping errors* [32] using the Sintel dataset [45] which contains ground truth optical flows between video frames. We use the foreground object masks in the DAVIS video dataset [40], [41] as our inpainting mask sequences. We take 32 frames each from 21 videos in Sintel to constitute our inputs and experiment for five trials. For each trial, we randomly select 21 videos of length 32+ from DAVIS to create corresponding mask sequences and keep them unchanged for all the methods.

In Table 6, we report the flow warping errors averaged over the videos and trials. It shows that our full model outperforms other baselines by large margins. Understandably, Yu *et al.*'s method

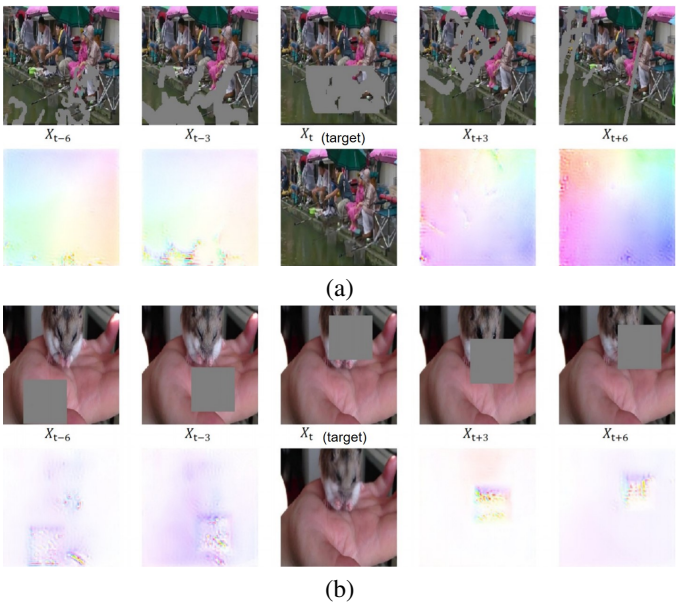


Fig. 9: **Visualization of the learned feature composition.** Input frames are on the odd rows, and corresponding feature flows referential to the middle stream, and the inpainting results are on the even rows. VINet successfully aligns and integrates the reference features onto the target frame to fill in the large and complex hole region.

	DAVIS masks on Sintel frames
Frame-by-frame [5]	0.00369
CombCN [9]	0.00216
Optimization [8]	0.00161
VINet (agg. only)	0.00156
VINet (agg. + rec.)	**0.00148**

TABLE 6: **Flow warping errors.** We evaluate the flow warping errors on the Sintel dataset using 21 videos and ground truth flows.

	DAVIS masks on DAVIS frames
Frame-by-frame [5]	0.00798
CombCN [9]	0.01403
Optimization [8]	0.00533
VINet (agg. only)	0.00727
VINet (agg. + rec.)	**0.00467**

TABLE 7: **FID scores.** We evaluate the FID scores on the DAVIS dataset using 20 videos.

turns out to be the least temporally consistent. CombCN [9] improves over the single image inpainting [5] in this metric, but falls behind all the other video based methods, and their video results still show considerable flicker artifacts. Surprisingly, even the global (heavy) optimization method [8] performs marginally better than our 1st-stage method and has a much larger error than our full model.

Note that the error of our full model is reduced by a factor of 10 after adding the recurrent feedback, implying that it significantly improves the temporal stability.

5.2.3 *Spatio-Temporal Video Quality*

Wang *et al.* [33] proposed a video version of the inception score (FID) to quantitatively evaluate the quality of video generation. We take this metric to evaluate the quality of video inpainting as it measures the spatio-temporal quality in a perceptual level. As in [33], we follow the protocol that uses the I3D network [46] pretrained on a video recognition task to measure the distance between the spatio-temporal features extracted from the output videos and the ground truth videos.

For this experiment, we take 20 videos in the DAVIS dataset. For each video, we ensure to choose a different video out of the other 19 videos to make a mask sequence, so that we have the setting where our algorithm is supposed to recover the original videos rather than remove any parts. We use the first 64 frames for both input and mask videos. We run five trials as in Section 5.2.2 and average the FID scores over the videos and trials. Table 7 summarizes the results. Our method has the smallest FID among the optimization based and all the learning based methods. This implies that our method achieves both better visual quality and temporal consistency.

5.2.4 *User Study on Video Object Removal*

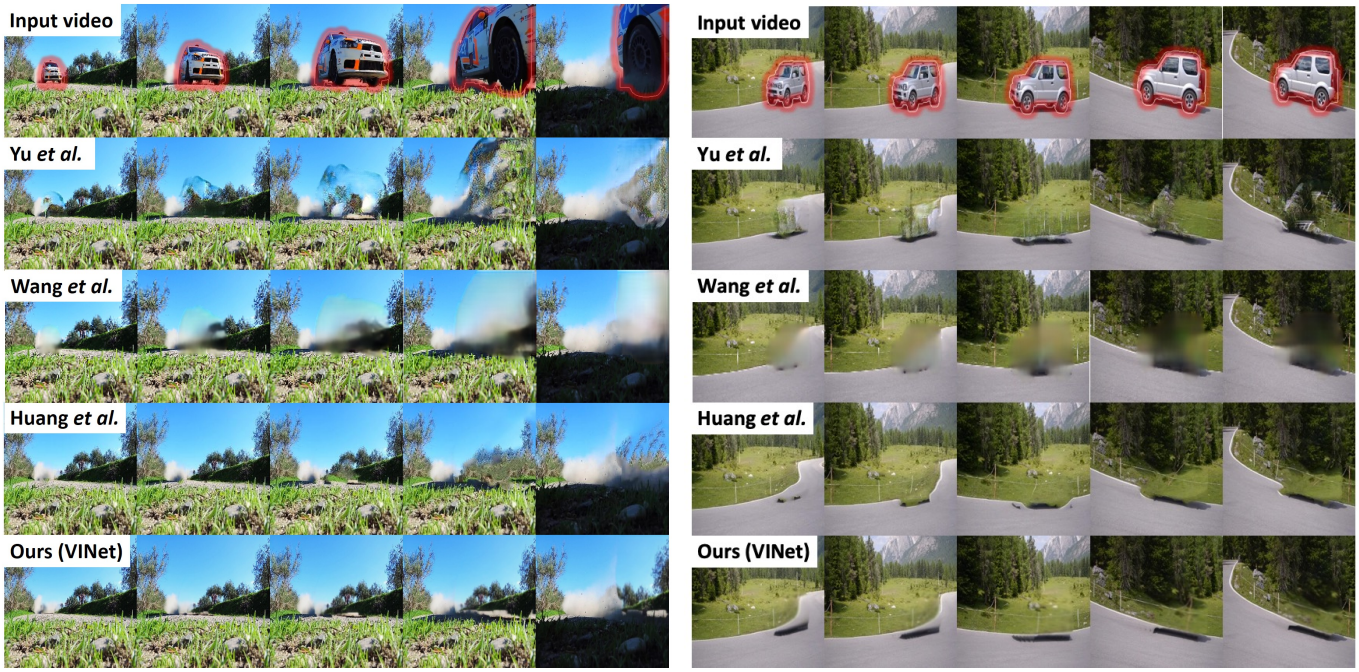
We apply our approach to remove dynamically moving objects in videos. We use 24 videos from the DAVIS dataset [40], [41] of which the names are listed in Figure 11. Examples of our results are in Figure 12. We perform a human subjective test for evaluating the visual quality of inpainted videos. We compare our method with the strong optimization baseline [8] which is specifically aimed for the video completion task.

In each testing case, we show the original input video, our removal result and the result of Huang *et al.*on the same screen. The order of the two removal video results is shuffled. To ensure that a user has enough time to distinguish the difference and make a careful judge, we play all the video results once at the original speed and then once at $0.5\times$ speed. Also, a user is allowed to watch videos multiple times. Each participant is asked to choose a preferred result or tie. A total of 30 users participated in this study. We specifically ask each participant to check for both image quality and temporal consistency. The user study results are summarized in Figure 11. To provide the statistical significance, we conduct F-test and T-test on our survey results. F-test gives a two-tailed p-value of 0.08, making us assume the equal variances. Then, on the assumption that the mean preference between ours (μ=22) and the optimization method [8] (μ=18.7) is statistically same (null hypothesis), we obtain a two-tailed p-value of 0.11 in T-test, which fails to reject the null hypothesis, *i.e.*, the null hypothesis holds. This supports our argument that both methods show comparable performances.

In terms of visual quality, there are certain cases where the proposed method is advantageous over its competitor [8]. As shown in Figure 10, the optimization method is vulnerable to fast object motion (*rallye*) and camera movement (*car-turn*) which lead to inaccurate optical flow between frames, while our learned semantics and recurrence are able to refine such errors. The complete video results can be found in the supplementary materials.

5.2.5 *Extension to Higher Resolution Videos*

Since our VINet is designed to be fully convolutional, we can simply feed higher resolution frames at testing. However, to better perform feature alignment and pixel reconstruction, we finetune



(a) *rallye* (b) *car-turn*

Fig. 10: **Comparison with state-of-the-art methods.** Input video with mask boundaries in red (row-1). Video inpainting by frame-by-frame image inpainting [5] (row-2), CombCN [9] (row-3), optimization-based method [8] (row-4), and the results by our VINet (row-5). *Best viewed when zoomed-in.*

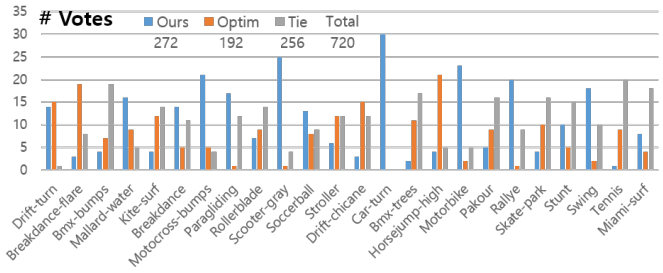


Fig. 11: **User study on results of VINet vs. optimization method [8].**

the VINet on high resolution (512×512) pixels frames with the doubled stride value in the first encoder layer and the last decoder layer. The complete video results can be found in the supplementary materials.

5.2.6 *Limitation and Future Work*

We observe color saturation artifacts when there is a large and long occlusion in a video. The discrepancy error of the synthesized color propagates over time, causing inaccurate warping. The regions that have not been revealed in the temporal radius is synthesized blurry. One possible way to resolve these issues is integrating recently proposed techniques such as multi-stage flow inpainting [47] or Temporal PatchGAN loss [48].

For the future work, we would like to further improve the feature aggregation part to ensure even larger temporal-window size [49]. Another interesting direction is to allow user intervention by training the model with additional user inputs such as referral expressions [50] or scribbles [51], which would be useful for video editing.

6 Conclusion

In this paper, we propose a novel framework for video inpainting based on two guiding principles: aggregating temporal information and recurrently connecting past predictions for temporally coherent generation. Building upon these principles, we present two network designs for two video inpainting tasks. First, we focus on a blind video decaptioning task, and our proposed BVDNet automatically removes text overlays in videos without any mask information. Second, we extend our framework to handle a more general video inpainting scenario: video foreground object removal. Our proposed VINet deals with arbitrary and large holes indicated by object-level input masks. Our extensive experiments demonstrate that both BVDNet and VINet achieve significantly better visual quality than the per-frame deep CNN based competitors. Our BVDNet is ranked in the first place in the ECCV Chalearn 2018 LAP Inpainting Challenge - Video decaptioning. Furthermore, we show that our VINet performs similarly to an optimization method. Despite some limitations, we argue that a well-posed feed-forward network has a great potential to avoid computation-heavy methods and to boost its applicability in many related vision tasks.

References

[1] D. Lee, T. Pfister, and M.-H. Yang, "Inserting videos into videos," in *Proc. of Computer Vision and Pattern Recognition (CVPR)*.
[2] S. Hong, X. Yan, T. S. Huang, and H. Lee, "Learning hierarchical semantic image manipulation through structured representations," in *Proc. of Neural Information Processing Systems (NIPS)*, 2018, pp. 2708–2718.

Fig. 12: **More object removal results on DAVIS video sequences.** For each input sequence, we show representative frames with mask boundaries in red. We show the inpainted results using our method in even rows.

[3] D. Lee, S. Liu, J. Gu, M.-Y. Liu, M.-H. Yang, and J. Kautz, "Context-aware synthesis and placement of object instances," in *Proc. of Neural Information Processing Systems (NIPS)*, 2018, pp. 10 393–10 403.

[4] X. Ouyang, Y. Cheng, Y. Jiang, C.-L. Li, and P. Zhou, "Pedestrian-synthesis-gan: Generating pedestrian data in real scene and beyond," *arXiv preprint arXiv:1804.02047*, 2018.

[5] J. Yu, Z. Lin, J. Yang, X. Shen, X. Lu, and T. S. Huang, "Generative image inpainting with contextual attention," in *Proc. of Computer Vision and Pattern Recognition (CVPR)*, 2018.

[6] T. Shiratori, Y. Matsushita, X. Tang, and S. B. Kang, "Video completion by motion field transfer," in *Proc. of Computer Vision and Pattern Recognition (CVPR)*, vol. 1. IEEE, 2006, pp. 411–418.

[7] Y. Matsushita, E. Ofek, W. Ge, X. Tang, and H.-Y. Shum, "Full-frame video stabilization with motion inpainting," *IEEE Trans. Pattern Anal. Mach. Intell. (TPAMI)*, vol. 28, no. 7, pp. 1150–1163, 2006.

[8] J.-B. Huang, S. B. Kang, N. Ahuja, and J. Kopf, "Temporally coherent completion of dynamic video," in *ACM Trans. on Graph. (ToG)*, vol. 35, no. 6. ACM, 2016, p. 196.

[9] C. Wang, H. Huang, X. Han, and J. Wang, "Video inpainting by jointly learning temporal structure and spatial details," *arXiv preprint arXiv:1806.08482*, 2018.

[10] C. Ballester, M. Bertalmio, V. Caselles, G. Sapiro, and J. Verdera, "Filling-in by joint interpolation of vector fields and gray levels," in *IEEE Trans. Image Processing (TIP)*, vol. 10, no. 8. IEEE, 2001, pp. 1200–1211.

[11] M. Bertalmio, G. Sapiro, V. Caselles, and C. Ballester, "Image inpainting," in *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, 2000, pp. 417–424.

[12] A. A. Efros and T. K. Leung, "Texture synthesis by non-parametric sampling," in *Proc. of Int'l Conf. on Computer Vision (ICCV)*. IEEE, 1999, p. 1033.

[13] D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, and A. A. Efros, "Context encoders: Feature learning by inpainting," in *Proc. of Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 2536–2544.

[14] C. Yang, X. Lu, Z. Lin, E. Shechtman, O. Wang, and H. Li, "High-resolution image inpainting using multi-scale neural patch synthesis," in *Proc. of Computer Vision and Pattern Recognition (CVPR)*, vol. 1, no. 2, 2017, p. 3.

[15] R. A. Yeh, C. Chen, T.-Y. Lim, A. G. Schwing, M. Hasegawa-Johnson, and M. N. Do, "Semantic image inpainting with deep generative models." in *Proc. of Computer Vision and Pattern Recognition (CVPR)*, vol. 2, no. 3, 2017, p. 4.

[16] S. Iizuka, E. Simo-Serra, and H. Ishikawa, "Globally and locally consistent image completion," in *ACM Trans. on Graph. (ToG)*, vol. 36, no. 4. ACM, 2017, p. 107.

[17] G. Liu, F. A. Reda, K. J. Shih, T.-C. Wang, A. Tao, and B. Catanzaro, "Image inpainting for irregular holes using partial convolutions," in *Proc. of European Conf. on Computer Vision (ECCV)*, 2018.

[18] J. Yu, Z. Lin, J. Yang, X. Shen, X. Lu, and T. S. Huang, "Free-form image inpainting with gated convolution," *arXiv preprint arXiv:1806.03589*, 2018.

[19] C. Barnes, E. Shechtman, A. Finkelstein, and D. B. Goldman, "Patchmatch: A randomized correspondence algorithm for structural image editing," in *ACM Trans. on Graph. (ToG)*, vol. 28, no. 3. ACM, 2009, p. 24.

[20] M. Granados, J. Tompkin, K. Kim, O. Grau, J. Kautz, and C. Theobalt, "How not to be seenobject removal from videos of crowded scenes," in *Computer Graphics Forum*, vol. 31, no. 2pt1. Wiley Online Library, 2012, pp. 219–228.

[21] M. Granados, K. I. Kim, J. Tompkin, J. Kautz, and C. Theobalt, "Background inpainting for videos with dynamic objects and a free-moving camera," in *Proc. of European Conf. on Computer Vision (ECCV)*. Springer, 2012, pp. 682–695.

[22] K. A. Patwardhan, G. Sapiro, and M. Bertalmio, "Video inpainting of occluding and occluded objects," in *IEEE International Conference on Image Processing (ICIP) 2005.*, vol. 2. IEEE, 2005, pp. II–69.

[23] K. A. Patwardhan, G. Sapiro, and M. Bertalmío, "Video inpainting under constrained camera motion," in *IEEE Trans. Image Processing (TIP)*, vol. 16, no. 2. IEEE, 2007, pp. 545–553.

[24] Y. Wexler, E. Shechtman, and M. Irani, "Space-time video completion," in *IEEE Trans. Pattern Anal. Mach. Intell. (TPAMI)*. IEEE, 2004, pp. 120–127.

[25] A. Newson, A. Almansa, M. Fradet, Y. Gousseau, and P. Pérez, "Video inpainting of complex scenes," *SIAM Journal on Imaging Sciences*, vol. 7, no. 4, pp. 1993–2019, 2014.

[26] X. Tao, H. Gao, R. Liao, J. Wang, and J. Jia, "Detail-revealing deep video super-resolution," in *Proc. of Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 4472–4480.

[27] M. S. Sajjadi, R. Vemulapalli, and M. Brown, "Frame-recurrent video super-resolution," in *Proc. of Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 6626–6634.

[28] S. Niklaus and F. Liu, "Context-aware synthesis for video frame interpolation," in *Proc. of Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 1701–1710.

[29] D. Sun, X. Yang, M.-Y. Liu, and J. Kautz, "Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume," in *Proc. of Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 8934–8943.

[30] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," in *IEEE Trans. Image Processing (TIP)*, vol. 13, no. 4. IEEE, 2004, pp. 600–612.

[31] D. Eigen and R. Fergus, "Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture," in *Proc. of Int'l Conf. on Computer Vision (ICCV)*, 2015, pp. 2650–2658.

[32] W.-S. Lai, J.-B. Huang, O. Wang, E. Shechtman, E. Yumer, and M.-H. Yang, "Learning blind video temporal consistency," in *Proc. of European Conf. on Computer Vision (ECCV)*, 2018.

[33] T.-C. Wang, M.-Y. Liu, J.-Y. Zhu, G. Liu, A. Tao, J. Kautz, and B. Catanzaro, "Video-to-video synthesis," in *Proc. of Neural Information Processing Systems (NIPS)*, 2018.

[34] H. Huang, H. Wang, W. Luo, L. Ma, W. Jiang, X. Zhu, Z. Li, and W. Liu, "Real-time neural style transfer for videos," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2017, pp. 7044–7052.

[35] M. Ruder, A. Dosovitskiy, and T. Brox, "Artistic style transfer for videos," in *German Conference on Pattern Recognition*. Springer, 2016, pp. 26–36.

[36] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, and T. Brox, "Flownet 2.0: Evolution of optical flow estimation with deep networks," in *Proc. of Computer Vision and Pattern Recognition (CVPR)*, 2017.

[37] D. Chen, J. Liao, L. Yuan, N. Yu, and G. Hua, "Coherent online video style transfer," in *Proc. of Int'l Conf. on Computer Vision (ICCV)*, 2017.

[38] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015, pp. 234–241.

[39] N. Xu, L. Yang, Y. Fan, J. Yang, D. Yue, Y. Liang, B. Price, S. Cohen, and T. Huang, "Youtube-vos: Sequence-to-sequence video object segmentation," in *Proc. of European Conf. on Computer Vision (ECCV)*, 2018.

[40] F. Perazzi, J. Pont-Tuset, B. McWilliams, L. Van Gool, M. Gross, and A. Sorkine-Hornung, "A benchmark dataset and evaluation methodology for video object segmentation," in *Proc. of Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 724–732.

[41] J. Pont-Tuset, F. Perazzi, S. Caelles, P. Arbeláez, A. Sorkine-Hornung, and L. Van Gool, "The 2017 davis challenge on video object segmentation," *arXiv preprint arXiv:1704.00675*, 2017.

[42] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *Proc. of Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 1125–1134.

[43] C. Wang, C. Xu, C. Wang, and D. Tao, "Perceptual adversarial networks for image-to-image transformation," in *IEEE Trans. Image Processing (TIP)*, vol. 27, no. 8. IEEE, 2018, pp. 4066–4079.

[44] Chalearn-Baseline, "Eccv 2018 chalearn challenge track2 official website," http://chalearnlap.cvc.uab.es/challenge/26/track/31/baseline/, 2018.

[45] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black, "A naturalistic open source movie for optical flow evaluation," in *Proc. of European Conf. on Computer Vision (ECCV)*. Springer, 2012, pp. 611–625.

[46] J. Carreira and A. Zisserman, "Quo vadis, action recognition? a new model and the kinetics dataset," in *Proc. of Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2017, pp. 4724–4733.

[47] R. Xu, X. Li, B. Zhou, and C. C. Loy, "Deep flow-guided video inpainting," in *Proc. of Computer Vision and Pattern Recognition (CVPR)*, 2019.

[48] Y.-L. Chang, Z. Y. Liu, and W. Hsu, "Free-form video inpainting with 3d gated convolution and temporal patchgan," in *Proc. of Int'l Conf. on Computer Vision (ICCV)*, 2019.

[49] S. Woo, D. Kim, K. Park, J.-Y. Lee, and I. S. Kweon, "Align-and-attend network for globally and locally coherent video inpainting," *arXiv preprint arXiv:1905.13066*, 2019.

[50] F. Xiao, L. Sigal, and Y. Jae Lee, "Weakly-supervised visual grounding of phrases with linguistic structures," in *Proc. of Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 5945–5954.

[51] D. Lin, J. Dai, J. Jia, K. He, and J. Sun, "Scribblesup: Scribble-supervised convolutional networks for semantic segmentation," in *Proc. of Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 3159–3167.

Dahun Kim is is a Ph.D. student in EE department of Korea Advanced Institute of Science and Technology (KAIST), Korea. He received the BS and MS degrees in EE department of KAIST, Korea, in 2016 and 2018, respectively. He receives a global doctoral fellowship, funded by National Research Foundation (NRF). His research interests include object recognition, image/video processing, and computer vision. He is a student member of IEEE.

Sanghyun Woo is a Ph.D student in Electrical Engineering department of Korea Advanced Institute of Science and Technology (KAIST), South Korea. He received his B.S and M.S degrees in Electrical Engineering from Seoul National University (SNU) and KAIST in 2017 and 2019 respectively. His research interests include object recognition, image/video processing, and deep learning. He is a student member of the IEEE.

Joon-Young Lee is a Senior Research Scientist at Adobe Research. He received his Ph.D. and M.S. degrees in Electrical Engineering from KAIST, Korea in 2015 and 2009, respectively. He received the B.S. degree in Electrical and Electronic Engineering from Yonsei University, Korea in 2008. His research interests include deep learning and computer vision. He served as an Area Chair of ICCV 2019 and CVPR 2020.

In So Kweon is an professor in EE department of KAIST, South Korea. He received the BS and MS degrees in mechanical design and production engineering from Seoul National University, Seoul, Korea, in 1981 and 1983, respectively, and the PhD degree in robotics from the Robotics Institute, Carnegie Mellon University, Pittsburgh, Pennsylvania, in 1990. He worked for the Toshiba R&D Center, Japan, and joined the Department of Automation and Design Engineering, KAIST, Seoul, Korea, in 1992, where he is now a professor with the Department of Electrical Engineering. He is a recipient of the best student paper runner-up award at the IEEE Conference on Computer Vision and Pattern Recognition (CVPR 09). His research interests are in camera and 3D sensor fusion, color modeling and analysis, visual tracking, and visual SLAM. He was the program co-chair for the Asian Conference on Computer Vision (ACCV 07) and was the general chair for the ACCV 12. He is also on the editorial board of the International Journal of Computer Vision. He is a member of the IEEE and the KROS.