

Towards An End-to-End Framework for Flow-Guided Video Inpainting

Zhen Li^{1*} Cheng-Ze Lu^{1*} Jianhua Qin² Chun-Le Guo^{1†} Ming-Ming Cheng¹

¹TMCC, CS, Nankai University ²Hisilicon Technologies Co. Ltd.

zhenli1031@gmail.com, czlu919@outlook.com, qinjianhua@hisilicon.com
 {guochunle, cmm}@nankai.edu.cn

Abstract

Optical flow, which captures motion information across frames, is exploited in recent video inpainting methods through propagating pixels along its trajectories. However, the hand-crafted flow-based processes in these methods are applied separately to form the whole inpainting pipeline. Thus, these methods are less efficient and rely heavily on the intermediate results from earlier stages. In this paper, we propose an End-to-End framework for Flow-Guided Video Inpainting (E^2FGVI) through elaborately designed three trainable modules, namely, flow completion, feature propagation, and content hallucination modules. The three modules correspond with the three stages of previous flow-based methods but can be jointly optimized, leading to a more efficient and effective inpainting process. Experimental results demonstrate that the proposed method outperforms state-of-the-art methods both qualitatively and quantitatively and shows promising efficiency. The code is available at <https://github.com/MCG-NKU/E2FGVI>.

1. Introduction

Video inpainting aims to fill up the “corrupted” regions with plausible and coherent content throughout video clips. It is widely applied to real-world applications such as object removal [16], video restoration [28], and video completion [7, 39]. Despite the significant progress made in image inpainting [42, 60, 61], video inpainting remains full of challenges due to the complex video scenarios and deteriorated video frames. Directly performing image inpainting on each frame independently tends to generate temporally inconsistent videos and results in severe artifacts. Both spatial structure and temporal coherence are required to be considered in high-quality video inpainting. Recent progress in deep learning motivates researchers to exploit more effective solutions [7, 8, 17, 23, 28, 33, 38, 50, 57, 63].

Among them, typical flow-based methods [17, 57] consider video inpainting as a pixel propagation problem to naturally preserve the temporal coherence. As shown in

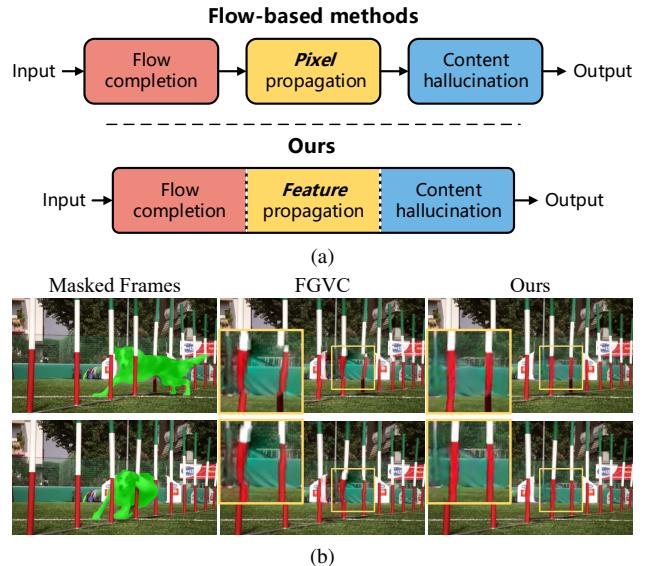


Figure 1. (a) The general pipelines of flow-based methods [17, 57] and ours. While previous flow-based methods conduct the three stages separately, our corresponding modules work in an end-to-end manner. (b) A qualitative comparison of our approach with a state-of-the-art flow-based method FGVC [17]. Due to the error accumulation and ignoring temporal information during content hallucination, FGVC fails to generate faithful and temporally consistent results compared with our method.

Fig. 1 (a), these methods can be decomposed into three inter-related stages. (1) *Flow completion*: The estimated optical flow needs to be completed first because the absence of flow fields in corrupted regions will influence the latter processes. (2) *Pixel propagation*: They fill the holes in corrupted videos by bidirectionally propagating pixels in the visible areas with the guidance of the completed optical flow. (3) *Content hallucination*: After propagation, the remaining missing regions can be hallucinated by a pre-trained image inpainting network [60, 61].

Unfortunately, even though impressive results can be obtained, the whole flow-based inpainting process must be carried out separately as many hand-crafted operations (e.g., Poisson blending, solving sparse linear equations, and indexing per-pixel flow trajectories) are involved in the first two stages. The isolated processes raise two main problems. One is that the errors that occur at earlier stages would be

*Equal contribution

†C.L. Guo is the corresponding author.

accumulated and amplified at subsequent stages, which further influences the final performance significantly. Specifically, the inaccurate flow estimation would mislead the propagation of pixels and further confuse the stage of content hallucination, producing unfaithful inpainting results. Second, these complex hand-designed operations only can be processed without GPU acceleration. The whole procedure of inferring video sequences, therefore, is very time-consuming. Taking DFVI [57] as an example, completing one video with the size of 432×240 from DAVIS [44], which contains about 70 frames, needs about 4 minutes¹, which is unacceptable in most real-world applications. Besides, except for the above-mentioned drawbacks, only using a pretrained image inpainting network at the content hallucination stage ignores the content relationships across temporal neighbors, leading to inconsistent generated content in videos (see Fig. 1 (b)).

To address these flaws, in this paper, we carefully design three trainable modules, including (1) flow completion, (2) feature propagation, and (3) content hallucination modules which simulate corresponding stages in flow-based methods and further constitute an End-to-End framework for Flow-Guided Video Inpainting (E²FGVI). Such close collaboration between the three modules alleviates the excessive dependence of intermediate results in the previously independently developed system [17, 23, 26, 57, 67] and works in a more efficient manner.

To be specific, for the *flow completion* module, we directly employ it on the masked videos for one-step completion instead of multiple complex steps. For the *feature propagation* module, in contrast to the pixel-level propagation, our flow-guided propagation process is conducted in the feature space with the assistance of deformable convolution. With more learnable sampling offsets and feature-level operations, the propagation module releases the pressure of inaccurate flow estimation. For the *content hallucination* module, we propose a temporal focal transformer to effectively model long-range dependencies on both spatial and temporal dimensions. Both local and non-local temporal neighbors are considered in this module, leading to more temporally coherent inpainting results.

Experimental results demonstrate that our framework enjoys the following two strengthens:

- State-of-the-art accuracy: Taking comparisons with previous state-of-the-art (SOTA) methods, the proposed E²FGVI achieves significant improvements on two common distortion-oriented metrics (*i.e.*, PSNR and SSIM [53]), one popular perception-oriented index (*i.e.*, VFID [51]), and one temporal consistency measurement (*i.e.*, E_{warp} [25]).
- High efficiency: Our method processes 432×240

¹We test it on Intel(R) Core(TM) i7-6700K CPU with a single NVIDIA Titan Xp GPU.

videos at 0.12 seconds per frame on a Titan Xp GPU, which is nearly $15\times$ faster than previous flow-based methods. In contrast to methods that also can be end-to-end deployed, our method shows comparable inference time. Besides, our method has the lowest computational complexity (FLOPs) among all compared SOTA methods.

We hope the proposed end-to-end framework with the aforementioned advantages could serve as a strong baseline for the video inpainting community.

2. Related Work

Video inpainting. Building upon the development of deep learning, great progress has been made in video inpainting. These methods can be roughly divided into three classes: 3D convolution-based [8, 21, 50], flow-based [17, 57], and attention-based methods [28, 29, 33, 63]. Some methods [7, 23, 28, 50] employing 3D convolution and attention usually yield temporally inconsistent results due to the limited temporal receptive fields. To generate more temporal coherence results, many works [23, 67] regard optical flows as strong priors for video inpainting and incorporate them into the network. However, directly computing optical flows between images within invalid regions is extremely difficult as these regions themselves become occlusion factors, restricting the performance. Recent flow-based methods [17, 57] perform flow completion first and use the completed optical flows to propagate indexed pixels along their trajectories. Instead of conducting hand-crafted pixel-level propagation, we design an end-to-end trainable framework that performs the propagation process at the feature space. Besides, our method benefits from recent advances in using transformers to improve the inpainting results [32, 33, 63].

Flow-based video processing. The motion information across frames well assists many video-related tasks, such as video understanding [3, 31], video segmentation [11, 48], video object detection [66], depth estimation [18, 36], video super-resolution [4, 58], frame interpolation [22, 27], *etc.* Specifically, many video restoration and enhancement algorithms [4, 24, 40, 46, 58] rely on optical flow to perform alignment for compensating the information between frames. Recent works [4, 27, 52, 54, 55] leverage deformable convolution [64] to simulate the behavior of optical flow but with more learnable offsets for more effective alignment. Our works also share the same merit as these works.

Vision transformer. Recently, Transformer [49] has gained much attention in the vision community. Vision Transformer [15] and its follow-ups [19, 34, 47, 59, 62] achieve an impressive performance on image and video representation learning [9, 13, 35, 43], image generation [41], object detection [2, 65], and many other applications [10, 12, 20, 30]. Because of the quadratic complexity of self-attention, many works deployed effective window-based at-

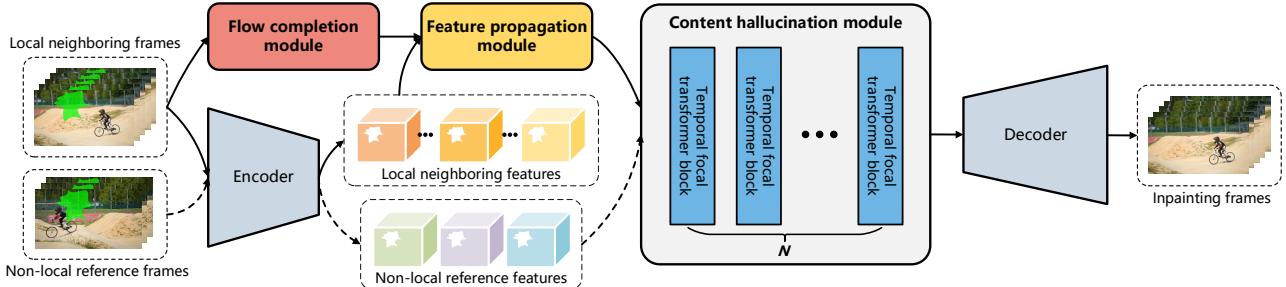


Figure 2. Overview of the proposed End-to-End framework for Flow-Guided Video Inpainting (E²FGVI). It consists of 1) a frame-level content encoder, 2) a flow completion module, 3) a feature propagation module, 4) a content hallucination module which is composed of multiple temporal focal transformer blocks, and 5) a frame-level decoder.

tentions [14, 34, 59] to reduce its computational complexity while improving the model’s capability with the limited receptive fields. Swin Transformer [34] strengthens local connections by computing self-attention through shifting local windows. Focal Transformer [59] introduces focal self-attention, which enhances the global-local interactions.

3. Method

Given a corrupted video sequence $\{X^t \in \mathbb{R}^{H \times W \times 3} | t = 1 \dots T\}$ with sequence length T and corresponding frame-wise binary masks $\{M^t \in \mathbb{R}^{H \times W \times 1} | t = 1 \dots T\}$, we aim at synthesizing faithful content which is consistent in both space and time dimensions within the corrupted (masked) areas. In the following, we discuss the main components of our method. First, we use a context encoder, which encodes all corrupted frames into lower-resolution features for computational efficiency at subsequent processing. Second, we extract and complete the optical flow between local neighbors through a flow completion module (Sec. 3.1). Third, the completed optical flow assists the features extracted from local neighbors to accomplish feature alignment and bidirectional propagation (Sec. 3.1). Fourth, multi-layer temporal focal transformers perform content hallucination by combining propagated local neighboring features with non-local reference features. (Sec. 3.2). Finally, a decoder up-scales the filled features and reconstructs them to a final video sequence $\{\hat{Y}^t \in \mathbb{R}^{H \times W \times 3} | t = 1 \dots T\}$.

Fig. 2 shows the whole pipeline of the proposed E²FGVI. It is worth noticing that all modules are differentiable and constitute an end-to-end trainable architecture.

3.1. Flow completion and feature propagation

In this section, we will detail the proposed flow-related operations. Note that we only apply flow-based modules on the features extracted from local neighboring frames because the flow estimation is substantially degraded or even fails because of the presence of large motion, which frequently occurs in non-local frames. Besides, the flow-

related operations are given at lower-resolution space for computational efficiency.

End-to-end flow completion. Before flow prediction, we first downsample the original corrupted frames X^t at 1/4 resolution, which matches the spatial resolution of encoded low-resolution features. The downsampled frames are denoted as $X_{\downarrow}^t \in \mathbb{R}^{\frac{H}{4} \times \frac{W}{4} \times 3}$. The flow prediction between adjacent frames i and j is computed by a flow estimation network \mathcal{F} :

$$\hat{F}_{i \rightarrow j} = \mathcal{F}(X_{\downarrow}^i, X_{\downarrow}^j). \quad (1)$$

We initialize the network using pretrained weights from a lightweight flow estimation network to resort to its rich knowledge about optical flows.

Following most flow-based video inpainting methods [17, 57], we estimate both forward flow $\hat{F}_{t \rightarrow t+1}$ and backward flow $\hat{F}_{t \rightarrow t-1}$ through Eq. (1) for flow-guided bidirectional propagation. Since the missing areas in corrupted videos become occlusion factors for flow estimation, which severely affects the quality of estimated flow, we need to restore the forward and backward flow before using them for feature propagation. For simplicity, we use L1 loss² to restore the bidirectional flows:

$$\mathcal{L}_{flow} = \sum_{t=1}^{T-1} \|\hat{F}_{t \rightarrow t+1} - F_{t \rightarrow t+1}\|_1 + \sum_{t=2}^T \|\hat{F}_{t \rightarrow t-1} - F_{t \rightarrow t-1}\|_1, \quad (2)$$

where $F_{t \rightarrow t+1}$ and $F_{t \rightarrow t-1}$ are the ground truth forward and backward flow, respectively, which are calculated from original uncorrupted videos.

Our flow completion module differs from DFVI [57] and FGVC [17] from two main aspects. (1) DFVI and FGVC deploy the flow completion network and propagation algorithm separately. In contrast, our flow completion module can be trained with other network components in an end-to-end manner, which facilitates the module to generate task-oriented flows [58]. (2) The flow completion in DFVI and FGVC is less efficient (> 0.4 s/flow) because they need to initialize the flow first and then refine the initialized flow

²Other loss functions can also be used in Eq. (2), but we do not observe significant improvements on the final inpainting performance.

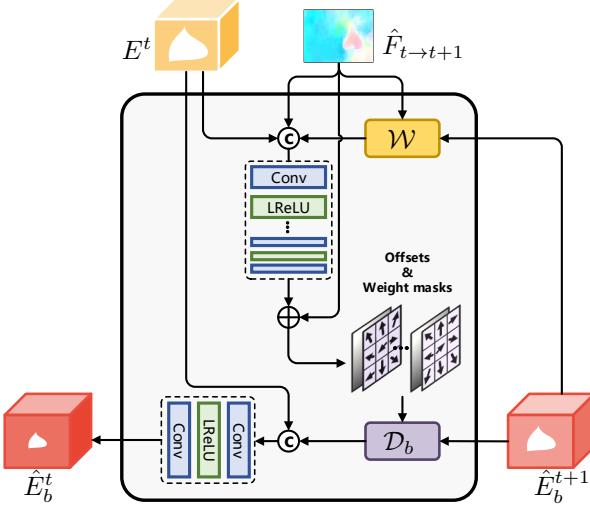


Figure 3. An example of using the completed forward flow $\hat{F}_{t \rightarrow t+1}$ to guide the feature backward propagation, where \oplus and \odot denote an addition operation and a concatenation operation, respectively. Note that the backward flow will act in the opposite direction.

with multiple stages, while we estimate and complete the flow in only one feed-forward pass with much faster speed ($< 0.01\text{s/flow}$).

Flow-guided feature propagation. Suppose $\{E^t \in \mathbb{R}^{\frac{H}{4} \times \frac{W}{4} \times C} \mid t = 1 \dots T_l\}$ are the local temporal neighboring features extracted from the context encoder, where T_l denotes the length of local neighboring frames. Taking the forward flow $\hat{F}_{t \rightarrow t+1}$ as an example, it assists us in capturing the motion of the corrupted regions from the t -th frame to the $(t+1)$ -th frame. Once the pixels in the corrupted regions at the t -th content feature is known in the valid area at the $(t+1)$ -th feature, we can intuitively exploit this valid information through warping the $(t+1)$ -th backward propagation feature \hat{E}_b^{t+1} to current time step with the help of the forward flow $\hat{F}_{t \rightarrow t+1}$. The warped feature can be further merged with current content feature E^t and updated through a backward propagation function $\mathcal{P}_b(\cdot)$:

$$\hat{E}_b^t = \mathcal{P}_b(E^t, \mathcal{W}(\hat{E}_b^{t+1}, \hat{F}_{t \rightarrow t+1})), \quad (3)$$

where $\mathcal{W}(\cdot)$ denotes the spatial warping operation based on optical flow, \hat{E}_b^t is the backward propagation feature at the t -th time step, and the propagation function $\mathcal{P}_b(\cdot)$ represents two convolutional layers with a LeakyReLU [37] activation.

The warping and merging operations in Eq. (3) are approximate to the whole propagation process in DFVI and FGVC, but we conduct them in the feature space rather than the image space. The propagation feature \hat{E}_b^t is updated step by step as faithful content is gradually involved in the corrupted area for each content feature, which also facilitates the connection across all local neighboring features with flow guidance. Unlike the hand-crafted pixel-level propagation in flow-based methods, which is very time-consuming and depends heavily on the quality of estimated flow, the

feature-level propagation adaptively merges the flow-traced information with larger receptive fields using convolutional layers and can be speeded up by GPUs.

Although the feature-level propagation can be much faster and more effective than FGVC and DFVI, it still needs to face the problem caused by the inaccurate flow estimation results in Eq. (1), which will bring irrelevant information in the propagation process and further hamper the final performance. To mitigate this problem, inspired by [4–6, 52], we employ modulated deformable convolution [64] to further index and weight the candidate feature points. As shown in Fig. 3, we first calculate the weight mask $W_{t \rightarrow t+1}$ and the offsets $\Delta F_{t \rightarrow t+1}$ relative to the estimated optical flow with:

$$[W_{t \rightarrow t+1}, \Delta F_{t \rightarrow t+1}] = \mathcal{C}_b(E^t, \mathcal{W}(\hat{E}_b^{t+1}, \hat{F}_{t \rightarrow t+1}), \hat{F}_{t \rightarrow t+1}), \quad (4)$$

where $\mathcal{C}_b(\cdot)$ denotes multiple cascading convolutional layers. Both the size of computed weight mask $M_{t \rightarrow t+1}$ and offset $\Delta F_{t \rightarrow t+1}$ are $\frac{H}{4} \times \frac{W}{4} \times K^2 \times G$, where K and G are the kernel size and the group number of deformable convolution, respectively. We can further generate $K^2 \times G$ candidate feature points for each spatial location by adding the offset $\Delta F_{t \rightarrow t+1}$ to the completed optical flow $\hat{F}_{t \rightarrow t+1}$. The relationship between the offset $\Delta F_{t \rightarrow t+1}$ and the completed optical flow $\hat{F}_{t \rightarrow t+1}$ are mutually beneficial. On the one hand, more flexible sampling locations could well compensate for the inaccurate flow completion. On the other hand, the completed flow provides promising initial sampling locations, which make it easily find more meaningful content within their surroundings. Then, we use a deformable convolutional layer to warp the backward feature \hat{E}_b^{t+1} instead of optical flow-based warping in Eq. (3) and further obtain the backward propagation feature \hat{E}_b^t through:

$$\hat{E}_b^t = \mathcal{P}_b(E^t, \mathcal{D}_b(\hat{E}_b^{t+1}, W_{t \rightarrow t+1}, \hat{F}_{t \rightarrow t+1} + \Delta F_{t \rightarrow t+1})), \quad (5)$$

where \mathcal{D}_b denotes the operation of the deformable convolutional layer. The weight mask $W_{t \rightarrow t+1}$, whose values are normalized via a sigmoid function, can be applied to each sampling pixel for measuring its validity.

The aforementioned operations are employed bidirectionally following [17, 57], while the forward propagation feature \hat{E}_f^t can be obtained in the same way but in the opposite direction. Finally, we use a learnable 1×1 sized convolution layer to fuse the forward and backward propagation features adaptively instead of using a pre-defined rule to combine the bidirectional flow traced pixels in [57].

$$\hat{E}^t = \mathcal{I}(\hat{E}_f^t, \hat{E}_b^t), \quad (6)$$

where \mathcal{I} denotes a 1×1 sized convolutional layer.

3.2. Temporal focal transformer

Only using the information provided by local temporal neighbors is not enough for video inpainting. As discussed

in [17], the corrupted content at local neighbors may appear in the non-local ones. Thus, the information in the non-local temporal neighbors can be regarded as a promising reference for these missing regions in local neighbors. Here we stack multiple temporal focal transformer blocks to effectively combine the information from local and non-local temporal neighbors for performing content hallucination.

Suppose T_{nl} is the number of selected non-local frames. $\mathbf{E}_{nl} \in \mathbb{R}^{T_{nl} \times \frac{H}{4} \times \frac{W}{4} \times C}$ is the encoded features of all non-local neighbors. $\hat{\mathbf{E}}_l \in \mathbb{R}^{T_l \times \frac{H}{4} \times \frac{W}{4} \times C}$ is the local temporal feature through concatenating the results in Eq. (6) at the temporal dimension. We use a soft split operation [33] to perform overlapped patch embedding on the concatenated local and non-local temporal features:

$$Z^0 = \text{SS}([\hat{\mathbf{E}}_l, \mathbf{E}_{nl}]) \in \mathbb{R}^{(T_l+T_{nl}) \times M \times N \times C_e}, \quad (7)$$

where SS denotes the operation of soft split. Z^0 is the embedded token that contains both local and non-local temporal information. $M \times N$ is the embedded spatial dimension, and C_e is the feature dimension.

Instead of vanilla vision transformer [15], which is frequently employed in recent works [33, 63], we use focal transformer [59] to search from both local and non-local neighbors to fill missing contents. The reasons are listed as follows: (1) Compared with performing fine-grained global attention, the computational and memory cost can be effectively reduced through window-based attention [34, 59]. (2) For each token in the missing regions, it is reasonable to perform the fine-grained self-attention only in local regions while the coarse-grained attentions globally because of the local self-similarity of an image.

Since the original focal transformer is unable to process sequence data, we propose a temporal focal transformer that essentially extends the size of focal windows from 2D to 3D. Specifically, we first split the input token Z^{n-1} , where $n \in [1, N]$ and N is the stacking number of focal transformer blocks, into a grid of sub-windows with size $s_t \times s_h \times s_w$. The split token $\hat{Z}^{n-1} \in \mathbb{R}^{(\frac{(T_l+T_{nl})}{s_t} \times \frac{M}{s_h} \times \frac{N}{s_w} \times C_e) \times (s_t \times s_h \times s_w)}$ can be directly used for computing fine-grained local attentions. To perform global attention at the coarse granularity, a linear embedding layer f_p is used to pool the sub-windows spatially via $\hat{Z}_g^{n-1} = f_p(\hat{Z}^{n-1}) \in \mathbb{R}^{(\frac{(T_l+T_{nl})}{s_t} \times \frac{M}{s_h} \times \frac{N}{s_w} \times C_e) \times s_t}$. We then calculate the query, key, and value through two linear projection layers f_q, f_{kv} :

$$Q^n = f_q(\hat{Z}^{n-1}), \quad \{K_l^n, K_g^n, V_l^n, V_g^n\} = f_{kv}(\{\hat{Z}^{n-1}, \hat{Z}_g^{n-1}\}). \quad (8)$$

To calculate attentions with local-global interactions, for the queries inside the i -th sub-window $Q_i^n \in \mathbb{R}^{s_t \times s_h \times s_w \times C_e}$, we gather the keys not only from the i -th local window $K_{l,i}^n \in \mathbb{R}^{s_t \times s_h \times s_w \times C_e}$ but also from the i -th unfolded coarse-grained window $K_{g,i}^n \in \mathbb{R}^{s_t \times s_h \times s_w \times C_e}$.

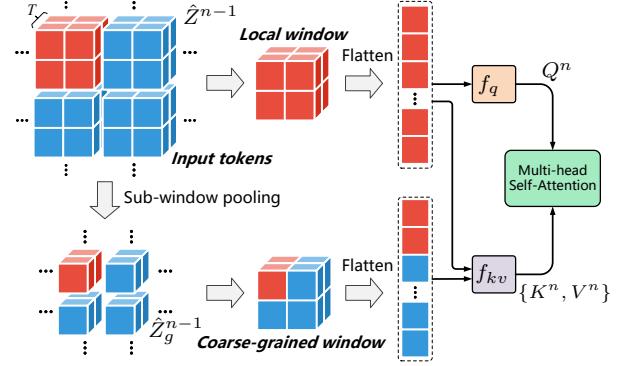


Figure 4. Illustration of temporal focal self-attention. Here we use the window size of $2 \times 2 \times 2$ as an example. We can see that the keys and values $\{K^n, V^n\}$ contain both fine-grained local information and coarse-grained global information.

This operation can be processed in parallel. We concatenate corresponding keys and values respectively by $K^n = \{K_l^n, K_g^n\}$ and $V^n = \{V_l^n, V_g^n\}$, and then calculate the focal self-attention for Q_i^n :

$$\text{Attention}(Q^n, K^n, V^n) = \text{Softmax}\left(\frac{Q^n(K^n)^T}{\sqrt{C_e}}\right)V^n. \quad (9)$$

Note that the attention function also can work in a multi-head manner. An example is shown in Fig. 4.

Finally, the whole process in the n -th focal transformer block is formulated as

$$Z'^n = \text{MFSA}(\text{LN}_1(Z^{n-1})) + Z^{n-1}, \quad (10)$$

$$Z^n = \text{F3N}(\text{LN}_2(Z'^n)) + Z'^n, \quad (11)$$

where MFSA and LN denote the multi-head focal self-attention and layer normalization [1], respectively. We use F3N [33] to link the connections across embedded patches.

3.3. Training objectives

We employ three loss functions to optimize our model. The first is the reconstruction loss which measures pixel-level differences between synthetic videos $\hat{\mathbf{Y}}$ and the original ones \mathbf{Y} through L1 distance:

$$\mathcal{L}_{rec} = \|\hat{\mathbf{Y}} - \mathbf{Y}\|_1. \quad (12)$$

The second is the adversarial loss which has been proven to be useful for the generation of high-quality and realistic content. We employ a T-PatchGAN [7] based discriminator to make the model focus on both global and local features across all temporal neighbors. The training objective of this discriminator D is:

$$\mathcal{L}_D = E_{x \sim P_{\mathbf{Y}}(x)}[\text{ReLU}(1 - D(x))] + E_{z \sim P_{\hat{\mathbf{Y}}}(z)}[\text{ReLU}(1 + D(z))], \quad (13)$$

For video inpainting generator, the adversarial loss is formulated as:

$$\mathcal{L}_{adv} = -E_{z \sim P_{\hat{\mathbf{Y}}}(z)}[D(z)], \quad (14)$$

Table 1. Quantitative comparisons with SOTA video inpainting models on YouTube-VOS [56] and DAVIS [44] datasets. \uparrow indicates higher is better. \downarrow indicates lower is better. E_{warp}^* denotes $E_{warp} \times 10^{-2}$. Each method is evaluated following the procedures in FuseFormer [33]. VINet, DFVI, and FGVC are not end-to-end training methods. Their FLOPs, thus, are not projectable.

Models	Accuracy								Efficiency	
	YouTube-VOS				DAVIS				FLOPs	Runtime (s/frame)
	PSNR \uparrow	SSIM \uparrow	VFID \downarrow	$E_{warp}^* \downarrow$	PSNR \uparrow	SSIM \uparrow	VFID \downarrow	$E_{warp}^* \downarrow$		
VINet [23]	29.20	0.9434	0.072	0.1490	28.96	0.9411	0.199	0.1785	-	-
DFVI [57]	29.16	0.9429	0.066	0.1509	28.81	0.9404	0.187	0.1608	-	2.56
LGTSM [8]	29.74	0.9504	0.070	0.1859	28.57	0.9409	0.170	0.1640	1008G	0.23
CAP [28]	31.58	0.9607	0.071	0.1470	30.28	0.9521	0.182	0.1533	861G	0.40
FGVC [17]	29.67	0.9403	0.064	0.1022	30.80	0.9497	0.165	0.1586	-	2.44
STTN [63]	32.34	0.9655	0.053	0.0907	30.67	0.9560	0.149	0.1449	1032G	0.12
FuseFormer [33]	33.29	0.9681	0.053	0.0900	32.54	0.9700	0.138	0.1362	752G	0.20
E ² FGVI (Ours)	33.71	0.9700	0.046	0.0864	33.01	0.9721	0.116	0.1315	682G	0.16

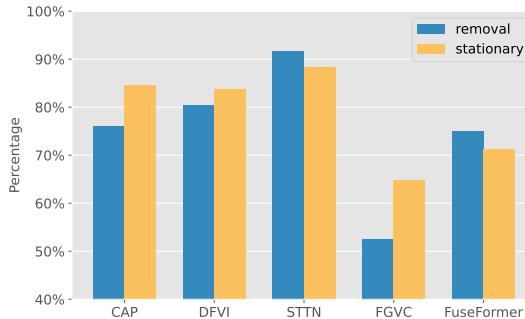


Figure 5. User study results. The vertical axis indicates the percentage of favoring our method compared to other methods.

The third loss is the flow consistency loss shown in Eq. (2). Training details can be found in supplementary materials.

4. Experiments

4.1. Settings

Dataset. To show the effectiveness of the proposed method, we evaluate it on two popular video object segmentation datasets, *i.e.*, YouTube-VOS [56] and DAVIS [44]. YouTube-VOS, with diverse scenes, consists of 3471, 474, and 508 video clips for training, validation, and test, respectively. We follow the original split mode and report the experimental metrics on the test set for YouTube-VOS. DAVIS is composed of 60 video clips for training and 90 video clips for testing. Following FuseFormer [33], 50 video clips from the test set are used for calculating metrics. We train our model on the YouTube-VOS dataset and evaluate it on both YouTube-VOS and DAVIS datasets. As for masks, during training, we generate stationary and object-like masks to simulate video completion and object removal applications following [8, 23, 28, 33, 63]. For evaluation, stationary masks are used to calculate objective metrics, and object-like masks are adopted for qualitative comparisons because of the lack of references.

Metrics. We choose PSNR, SSIM [53], VFID [51], and flow warping error E_{warp} [25] to evaluate the performance

of recent video inpainting methods. Specifically, PSNR and SSIM are frequently used metrics for distortion-oriented image and video assessment. VFID measures the perceptual similarity between two input videos and has been adopted in recent video inpainting works [33, 63]. Flow warping error E_{warp} is employed to measure the temporal stability.

4.2. Comparison

Quantitative results. We report quantitative results on YouTube-VOS [56] and DAVIS [44] under the stationary masks and compare our method with previous video inpainting methods, including VINet [23], DFVI [57], LGTSM [8], CAP [28], STTN [63], FGVC [17], and Fuseformer [33]. As shown in Tab. 1, our method substantially surpasses all previous SOTA algorithms on all four quantitative metrics. The superior results demonstrate that our method can generate videos with less distortion (PSNR and SSIM), more visually plausible content (VFID), and better spatial and temporal coherence (E_{warp}), which verifies the superiority of the proposed method.

Qualitative results. We choose three representative methods, including CAP [28], FGVC [17], and Fuseformer [33], to conduct visual comparisons. Fig. 6 shows both video completion and object removal results. While the compared methods are hard to recover reasonable details in the masked regions, the proposed method can generate faithful textural and structure information. This demonstrates the effectiveness of the proposed method.

For further comprehensive comparisons, a user study is conducted on both object removal and video completion applications. We select five methods including two flow-based methods (*i.e.*, DFVI [57] and FGVC [17]), and three attention-based methods (*i.e.*, CAP [28], STTN [63], and Fuseformer [33]). We invite 20 participants for the user study totally. Every volunteer is shown randomly sampled 40 video triplets and asked to select a visually better inpainting video. Each triplet is composed of one original video, one from our method, and one from a randomly cho-



Figure 6. Qualitative results compared with CAP [28], FGVC [17], FuseFormer [33].

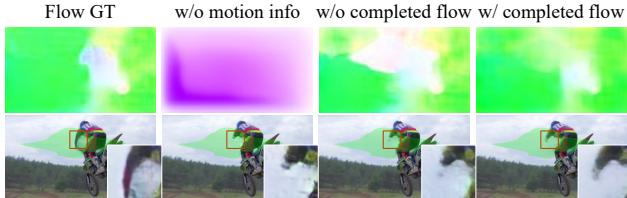


Figure 7. Ablation studies on the flow completion module. The first row shows the results generated from the flow completion modules under different situations. The second row visualizes corresponding inpainting frames.

sen method. The user study results are shown in Fig. 5. As we can see, volunteers obviously favor our results over those from almost all methods. Although such significant preference does not exist in the comparisons with FGVC, the proposed method still receives a majority of votes. This demonstrates that the proposed method could generate more visually pleasant results than compared methods.

Efficiency comparisons. We use FLOPs and inference time to measure the efficiency of each method. The FLOPs are calculated using the temporal size of 8, and the runtime is measured on a single Titan Xp GPU using DAVIS dataset. The compared results are shown in Tab. 1. The proposed method shows comparable running time with transformer-based methods and is nearly $\times 15$ faster than flow-based methods. Besides, it holds the lowest FLOPs in contrast to all other methods. This indicates that the proposed method is highly efficient for video inpainting.

Table 2. Ablation studies on the flow completion module.

Case	PSNR	SSIM
w/o motion information	32.08	0.9673
w/o completed flow	32.23	0.9682
w/ completed flow	32.35	0.9688
Flow GT	32.54	0.9698

4.3. Ablations

We perform three ablation studies on flow completion, feature propagation, and attention mechanism to verify the effectiveness of proposed modules in our framework. All ablation studies are conducted on the DAVIS dataset.

Study of flow completion module. First, we investigate that the importance of motion information for video inpainting. By only removing the flow consistency loss \mathcal{L}_{flow} , our flow completion module no longer provides information about object motions (see Fig. 7), resulting in a large performance decrease, as shown in Tab. 2. Second, we study the necessity of completing the optical flow through fixing the pretrained weights in the flow completion module. With the preliminary knowledge about optical flow, the flow completion module regards the masked regions as occlusion factors and provides initial flow estimation for visible regions (see Fig. 7). In contrast to the model without motion information, the performance has an obvious improvement. However, such model ignores the motion information in the masked regions. After we complete the flow by training the

Table 3. Investigation on the feature propagation module. ‘Flow’ indicates the flow-based warping function \mathcal{W} in Eq. (4). ‘DCN’ denotes modulated deformable convolution [64].

	(a)	(b)	(c)	(d)
Flow	✗	✓	✗	✓
DCN	✗	✗	✓	✓
PSNR	31.73/0.9653	32.15/.9677	32.17/0.9676	32.35/.9688

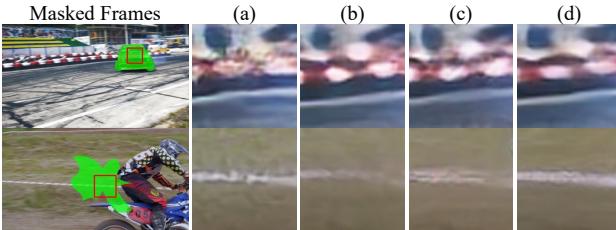


Figure 8. Qualitative results of the ablation studies on the feature propagation module. The last four columns correspond to four cases in Tab. 3.

flow completion module towards minimizing the flow consistency loss, we obtain larger PSNR and SSIM values than before. As shown in Fig. 7, the model with completed flows recovers more faithful content about the human arm. Additionally, in Tab. 2 and Fig. 7, we also show the potential upper bound of our method which estimates the optical flow between uncorrupted frames.

Study of feature propagation module. After we remove the feature propagation module from the model (case (a) in Tab. 3), the values of quantitative metrics are decreased dramatically. From Fig. 8 (a), we can see that the results generated by this model exist severe artifacts and discontinuous content. After adding flow-based warping and propagation (see Eq. (3)) to this model (case (b) in Tab. 3), since we could bring valid pixels from adjacent frames to unseen regions with the assistance of optical flow, the generated content becomes more faithful as shown in Fig. 8 (b), and the PSNR value is increased by a large margin (0.42dB). However, it is hard for flow-based warping and propagation to recover the content that cannot be traced by optical flow (the white line in Fig. 8 (b)). Besides, for the feature propagation module, which only involves deformable convolution-based warping (case (c) in Tab. 3), the structure details can be more clearly recovered with the help of more learnable offsets, but more artifacts are involved due to the lack of faithful information warped from adjacent frames in contrast to flow-based warping. By combining deformable convolution with flow guidance (case (d) in Tab. 3), the PSNR and SSIM values can be further improved. In Fig. 8 (d), this model achieves the visually best results among all variants while preserving promising structure details. This demonstrates the effectiveness of the feature propagation module.

Study of attention mechanism. We remove the flow completion and feature propagation modules to purely compare different attention mechanisms, including vanilla global attention (FuseFormer [33]), local window attention, and fo-

Table 4. Ablation study on various attention mechanisms. FuseFormer [33] is the current SOTA method that uses vanilla global attention.

Case	PSNR	SSIM	FLOPs
FuseFormer	31.74	0.9662	752G
Local attention	31.57	0.9648	497G
Focal attention	31.73	0.9653	560G



Figure 9. Two failure cases (car drifting). Current video inpainting methods fail to deal with large motion or a large number of missing object details and may produce severe artifacts.

cal attention. As shown in Tab. 4, vanilla global attention achieves the best quantitative performance while suffering from the heavy computation. Local attention introduces local windows as Video Swin Transformer [35] does. Although the FLOPs are decreased by 34%, the attention calculation is limited in the local window, leading to poor performance. Focal attention shows a good trade-off between performance and computation. Its PSNR and SSIM values are comparable to FuseFormer, and the computational cost is only increased by 12% in contrast to the local one.

4.4. Limitation

Fig. 9 shows two failure cases. When encountering large motion or a large amount of missing object details across frames, our method produces implausible content and many artifacts in masked regions as well as FGVC [17] and FuseFormer [33] do. This demonstrates that these situations are still challenging for video inpainting.

5. Conclusion

We have proposed an end-to-end trainable flow-based model for video inpainting named E²FGVI. The elaborately designed three modules (*i.e.*, flow completion, feature propagation, and content hallucination modules) are collaborated together and address many bottlenecks of previous methods. Experimental results have shown that our method achieves state-of-the-art quantitative and qualitative performance on two benchmark datasets and is efficient in terms of inference time and computational complexity. We hope it can serve as a strong baseline for future works.

Acknowledgment: This work is funded by the National Key Research and Development Program of China (NO. 2018AAA0100400), NSFC (NO. 61922046), S&T innovation project from Chinese Ministry of Education, and China Postdoctoral Science Foundation (NO.2021M701780). We also gratefully acknowledge the support of MindSpore, CANN, and Ascend AI Processor used for this research.

References

- [1] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016. 5
- [2] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *ECCV*, 2020. 2
- [3] Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *CVPR*, 2017. 2
- [4] Kelvin CK Chan, Xintao Wang, Ke Yu, Chao Dong, and Chen Change Loy. Basicvsr: The search for essential components in video super-resolution and beyond. In *CVPR*, 2021. 2, 4
- [5] Kelvin CK Chan, Xintao Wang, Ke Yu, Chao Dong, and Chen Change Loy. Understanding deformable alignment in video super-resolution. In *AAAI*, 2021. 4
- [6] Kelvin C.K. Chan, Shangchen Zhou, Xiangyu Xu, and Chen Change Loy. Basicvsr++: Improving video super-resolution with enhanced propagation and alignment. In *CVPR*, 2022. 4
- [7] Ya-Liang Chang, Zhe Yu Liu, Kuan-Ying Lee, and Winston Hsu. Free-form video inpainting with 3d gated convolution and temporal patchgan. In *ICCV*, 2019. 1, 2, 5, 11
- [8] Ya-Liang Chang, Zhe Yu Liu, Kuan-Ying Lee, and Winston Hsu. Learnable gated temporal shift module for deep video inpainting. In *BMVC*, 2019. 1, 2, 6
- [9] Mark Chen, Alec Radford, Rewon Child, Jeffrey Wu, Heewoo Jun, David Luan, and Ilya Sutskever. Generative pre-training from pixels. In *ICML*, 2020. 2
- [10] Bowen Cheng, Alexander G. Schwing, and Alexander Kirillov. Per-pixel classification is not all you need for semantic segmentation. In *NeurIPS*, 2021. 2
- [11] Jingchun Cheng, Yi-Hsuan Tsai, Shengjin Wang, and Ming-Hsuan Yang. Segflow: Joint learning for video object segmentation and optical flow. In *ICCV*, 2017. 2
- [12] Krzysztof Choromanski, Valerii Likhoshesterov, David Dohan, Xingyou Song, Andreea Gane, Tamas Sarlos, Peter Hawkins, Jared Davis, Afroz Mohiuddin, Lukasz Kaiser, et al. Rethinking attention with performers. In *ICLR*, 2021. 2
- [13] Karan Desai and Justin Johnson. Virtex: Learning visual representations from textual annotations. In *CVPR*, 2021. 2
- [14] Xiaoyi Dong, Jianmin Bao, Dongdong Chen, Weiming Zhang, Nenghai Yu, Lu Yuan, Dong Chen, and Baineng Guo. Cswin transformer: A general vision transformer backbone with cross-shaped windows. *arXiv preprint arXiv:2107.00652*, 2021. 3
- [15] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021. 2, 5
- [16] Mounira Ebdelli, Olivier Le Meur, and Christine Guillemot. Video inpainting with short-term windows: Application to object removal and error concealment. *TIP*, 2015. 1
- [17] Chen Gao, Ayush Saraf, Jia-Bin Huang, and Johannes Kopf. Flow-edge guided video completion. In *ECCV*, 2020. 1, 2, 3, 4, 5, 6, 7, 8, 11, 12
- [18] Clément Godard, Oisin Mac Aodha, Michael Firman, and Gabriel J Brostow. Digging into self-supervised monocular depth estimation. In *ICCV*, 2019. 2
- [19] Meng-Hao Guo, Cheng-Ze Lu, Zheng-Ning Liu, Ming-Ming Cheng, and Shi-Min Hu. Visual attention network. *arXiv preprint arXiv:2202.09741*, 2022. 2
- [20] Meng-Hao Guo, Tian-Xing Xu, Jiang-Jiang Liu, Zheng-Ning Liu, Peng-Tao Jiang, Tai-Jiang Mu, Song-Hai Zhang, Ralph R Martin, Ming-Ming Cheng, and Shi-Min Hu. Attention mechanisms in computer vision: A survey. *Computational Visual Media*, 2022. 2
- [21] Yuan-Ting Hu, Heng Wang, Nicolas Ballas, Kristen Grauman, and Alexander G Schwing. Proposal-based video completion. In *ECCV*, 2020. 2
- [22] Huaizu Jiang, Deqing Sun, Varun Jampani, Ming-Hsuan Yang, Erik Learned-Miller, and Jan Kautz. Super slomo: High quality estimation of multiple intermediate frames for video interpolation. In *CVPR*, 2018. 2
- [23] Dahyun Kim, Sanghyun Woo, Joon-Young Lee, and In So Kweon. Deep video inpainting. In *CVPR*, 2019. 1, 2, 6
- [24] Tae Hyun Kim, Mehdi SM Sajjadi, Michael Hirsch, and Bernhard Scholkopf. Spatio-temporal transformer network for video restoration. In *ECCV*, 2018. 2
- [25] Wei-Sheng Lai, Jia-Bin Huang, Oliver Wang, Eli Shechtman, Ersin Yumer, and Ming-Hsuan Yang. Learning blind video temporal consistency. In *ECCV*, 2018. 2, 6, 11
- [26] Dong Lao, Peihao Zhu, Peter Wonka, and Ganesh Sundaramoorthi. Flow-guided video inpainting with scene templates. In *ICCV*, 2021. 2
- [27] Hyeongmin Lee, Taeoh Kim, Tae-young Chung, Daehyun Pak, Yuseok Ban, and Sangyoun Lee. Adacof: Adaptive collaboration of flows for video frame interpolation. In *CVPR*, 2020. 2
- [28] Sungho Lee, Seoung Wug Oh, DaeYeun Won, and Seon Joo Kim. Copy-and-paste networks for deep video inpainting. In *ICCV*, 2019. 1, 2, 6, 7, 12
- [29] Ang Li, Shanshan Zhao, Xingjun Ma, Mingming Gong, Jianzhong Qi, Rui Zhang, Dacheng Tao, and Ramamohanarao Kotagiri. Short-term and long-term context aggregation network for video inpainting. In *ECCV*, 2020. 2
- [30] Jingyun Liang, Jiezhang Cao, Guolei Sun, Kai Zhang, Luc Van Gool, and Radu Timofte. Swinir: Image restoration using swin transformer. In *ICCV Workshops*, pages 1833–1844, 2021. 2
- [31] Ji Lin, Chuang Gan, and Song Han. Tsm: Temporal shift module for efficient video understanding. In *ICCV*, 2019. 2
- [32] Rui Liu, Hanming Deng, Yangyi Huang, Xiaoyu Shi, Lewei Lu, Wenxiu Sun, Xiaogang Wang, Jifeng Dai, and Hongsheng Li. Decoupled spatial-temporal transformer for video inpainting. *arXiv preprint arXiv:2104.06637*, 2021. 2
- [33] Rui Liu, Hanming Deng, Yangyi Huang, Xiaoyu Shi, Lewei Lu, Wenxiu Sun, Xiaogang Wang, Jifeng Dai, and Hongsheng Li. Fuseformer: Fusing fine-grained information in transformers for video inpainting. In *ICCV*, 2021. 1, 2, 5, 6, 7, 8, 11, 12

- [34] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *ICCV*, 2021. 2, 3, 5
- [35] Ze Liu, Jia Ning, Yue Cao, Yixuan Wei, Zheng Zhang, Stephen Lin, and Han Hu. Video swin transformer. *arXiv preprint arXiv:2106.13230*, 2021. 2, 8
- [36] Xuan Luo, Jia-Bin Huang, Richard Szeliski, Kevin Matzen, and Johannes Kopf. Consistent video depth estimation. *TOG*, 2020. 2
- [37] Andrew L Maas, Awni Y Hannun, Andrew Y Ng, et al. Rectifier nonlinearities improve neural network acoustic models. In *ICML*, 2013. 4
- [38] Alasdair Newson, Andrés Almansa, Matthieu Fradet, Yann Gousseau, and Patrick Pérez. Video inpainting of complex scenes. *Siam journal on imaging sciences*, 2014. 1
- [39] Seoung Wug Oh, Sungho Lee, Joon-Young Lee, and Seon Joo Kim. Onion-peel networks for deep video completion. In *ICCV*, 2019. 1
- [40] Jinshan Pan, Haoran Bai, and Jinhui Tang. Cascaded deep video deblurring using temporal sharpness prior. In *CVPR*, 2020. 2
- [41] Niki Parmar, Ashish Vaswani, Jakob Uszkoreit, Lukasz Kaiser, Noam Shazeer, Alexander Ku, and Dustin Tran. Image transformer. In *ICML*, 2018. 2
- [42] Deepak Pathak, Philipp Krahenbuhl, Jeff Donahue, Trevor Darrell, and Alexei A Efros. Context encoders: Feature learning by inpainting. In *CVPR*, 2016. 1
- [43] Mandela Patrick, Dylan Campbell, Yuki M. Asano, Ishan Misra, Florian Metze, Christoph Feichtenhofer, Andrea Vedaldi, and Joao F. Henriques. Keeping your eye on the ball: Trajectory attention in video transformers. In *NeurIPS*, 2021. 2
- [44] Federico Perazzi, Jordi Pont-Tuset, Brian McWilliams, Luc Van Gool, Markus Gross, and Alexander Sorkine-Hornung. A benchmark dataset and evaluation methodology for video object segmentation. In *CVPR*, 2016. 2, 6, 12, 15, 16
- [45] Anurag Ranjan and Michael J Black. Optical flow estimation using a spatial pyramid network. In *CVPR*, 2017. 11
- [46] Yapeng Tian, Yulun Zhang, Yun Fu, and Chenliang Xu. Tdan: Temporally-deformable alignment network for video super-resolution. In *CVPR*, 2020. 2
- [47] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Herve Jegou. Training data-efficient image transformers & distillation through attention. In *ICML*, 2021. 2
- [48] Yi-Hsuan Tsai, Ming-Hsuan Yang, and Michael J Black. Video segmentation via object flow. In *CVPR*, 2016. 2
- [49] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017. 2
- [50] Chuan Wang, Haibin Huang, Xiaoguang Han, and Jue Wang. Video inpainting by jointly learning temporal structure and spatial details. In *AAAI*, 2019. 1, 2
- [51] Ting-Chun Wang, Ming-Yu Liu, Jun-Yan Zhu, Guilin Liu, Andrew Tao, Jan Kautz, and Bryan Catanzaro. Video-to-video synthesis. In *NeurIPS*, 2018. 2, 6
- [52] Xintao Wang, Kelvin C.K. Chan, Ke Yu, Chao Dong, and Chen Change Loy. Edvr: Video restoration with enhanced deformable convolutional networks. In *CVPR Workshops*, 2019. 2, 4
- [53] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *TIP*, 2004. 2, 6
- [54] Xiaoyu Xiang, Yapeng Tian, Yulun Zhang, Yun Fu, Jan P Allebach, and Chenliang Xu. Zooming slow-mo: Fast and accurate one-stage space-time video super-resolution. In *CVPR*, 2020. 2
- [55] Gang Xu, Jun Xu, Zhen Li, Liang Wang, Xing Sun, and Ming-Ming Cheng. Temporal modulation network for controllable space-time video super-resolution. In *CVPR*, 2021. 2
- [56] Ning Xu, Linjie Yang, Yuchen Fan, Jianchao Yang, Dingcheng Yue, Yuchen Liang, Brian Price, Scott Cohen, and Thomas Huang. Youtube-vos: Sequence-to-sequence video object segmentation. In *ECCV*, 2018. 6, 12, 13, 14
- [57] Rui Xu, Xiaoxiao Li, Bolei Zhou, and Chen Change Loy. Deep flow-guided video inpainting. In *CVPR*, 2019. 1, 2, 3, 4, 6
- [58] Tianfan Xue, Baian Chen, Jiajun Wu, Donglai Wei, and William T Freeman. Video enhancement with task-oriented flow. *IJCV*, 2019. 2, 3
- [59] Jianwei Yang, Chunyuan Li, Pengchuan Zhang, Xiyang Dai, Bin Xiao, Lu Yuan, and Jianfeng Gao. Focal attention for long-range interactions in vision transformers. In *NeurIPS*, 2021. 2, 3, 5
- [60] Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas S Huang. Generative image inpainting with contextual attention. In *CVPR*, 2018. 1
- [61] Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas S Huang. Free-form image inpainting with gated convolution. In *ICCV*, 2019. 1, 11
- [62] Li Yuan, Yunpeng Chen, Tao Wang, Weihao Yu, Yujun Shi, Francis EH Tay, Jiashi Feng, and Shuicheng Yan. Tokens-to-token vit: Training vision transformers from scratch on imagenet. In *ICCV*, 2021. 2
- [63] Yanhong Zeng, Jianlong Fu, and Hongyang Chao. Learning joint spatial-temporal transformations for video inpainting. In *ECCV*, 2020. 1, 2, 5, 6, 11
- [64] Xizhou Zhu, Han Hu, Stephen Lin, and Jifeng Dai. Deformable convnets v2: More deformable, better results. In *CVPR*, 2019. 2, 4, 8
- [65] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. In *ICLR*, 2021. 2
- [66] Xizhou Zhu, Yujie Wang, Jifeng Dai, Lu Yuan, and Yichen Wei. Flow-guided feature aggregation for video object detection. In *ICCV*, 2017. 2
- [67] Xueyan Zou, Linjie Yang, Ding Liu, and Yong Jae Lee. Progressive temporal feature alignment network for video inpainting. In *CVPR*, 2021. 2

Appendix

A. Architecture and Training Details

Architecture. In our model, the encoder and the decoder use the same architecture as FuseFormer [33]. The channel dim C of the encoder and the decoder is set as 128. A lightweight model SPyNet [45] is employed as our flow completion module for computational efficiency. To utilize the learned flow prior in original SPyNet, we use pre-trained weights to initialize this module. The architecture details of the T-PatchGAN are identical to previous works [7, 33, 63]. The kernel size K and the group number G of deformable convolution are set as 3 and 16, respectively. The number of focal transformer blocks N is set as 8 and the embedded dim of tokens C_e is set as 512. The embedded spatial dimension $M \times N$ is 20×36 . The size of partitioned subwindow $s_t \times s_h \times s_w$ is set to $(T_l + T_{nl}) \times 5 \times 9$. At the end of the content hallucination module, we use a soft composite operator [33] to composite the embedded tokens to features, which share the same spatial size as the original ones.

Training details. For training objectives, the weights of \mathcal{L}_{rec} , \mathcal{L}_{adv} , and \mathcal{L}_{flow} are 1, 10^{-2} , and 1, respectively. Taking the memory limitations of GPUs into account, we resize all frames from videos into 432×240 for training, evaluation, and test. During training, the numbers of local (T_l) and non-local frames (T_{nl}) are 5 and 3, respectively. Local frames are continuous clips, while non-local frames are randomly sampled from videos for training. Following STTN [63] and FuseFormer [33], during evaluation and test, we use a sliding window with the size of 10 to get local neighboring frames and uniformly sample the non-local neighboring frames with a sampling rate of 10. We adopt Adam optimizer with $\beta_1 = 0$ and $\beta_2 = 0.99$. The final model is trained for 500K iterations, and the initial learning rate is set as 0.0001 for all modules and reduced by the factor of 10 at 400K iteration. In our ablation studies, we train the model for 250K iterations. We use 8 NVIDIA Tesla V100 GPUs for training and the batch size is set as 8. Our code is available ³ for reproducibility.

B. More Experiments

B.1. Completing flows in a offline manner.

To verify the effectiveness of online flow completion, we prepare completed flows using the FGVC [25] flow completion module in an offline manner. We then retrain a model with the FGVC completed flows. The PSNR value of this model is slightly higher than our end-to-end setting (32.38

³<https://github.com/MCG-NKU/E2FGVI>

vs. 32.35 (dB)). However, the inference speed is much slower than ours (1.21 vs. 0.16 (s/frame)).

B.2. Taking a deeper look to flow-guided feature propagation module

To further investigate the effectiveness of the feature propagation module, we visualize averaged local neighboring features with the temporal size of 5 before conducting content hallucination in Fig. 10. The four cases in Fig. 10 correspond to the four variants in the Tab. 3 of our main paper. For the model without feature propagation (Fig. 10(a)), obviously, we can see that corrupted regions from all frames still exist in these features, further restricting the performance of content hallucination. For the model only using flow-based warping (Fig. 10(b)) or deformable convolution-based warping (Fig. 10(c)), corrupted regions are filled with the contents warped from adjacent frames. And the deformable convolution-based warping can generate smoother content than flow-based one due to more sampling feature points. However, especially for the last two temporal features (last two columns in Fig. 10), the regions filled by the model without flow guidance have more distinct boundaries in contrast to flow-based warping, which implies that less faithful content are propagated without motion information. Through adopting deformable convolution with flow guidance, the final propagation module (Fig. 10(d)) fills the holes with the most reasonable and natural content among all cases. This is a promising demonstration of the mutually beneficial relationship between deformable offsets and completed flow fields.

B.3. Study of the hallucination ability

To purely evaluate the hallucination ability of our method, we first pre-fill the pixels which can be traced by flow fields [17]. The remaining unfill pixels are thus most likely not visible in other video frames. We then feed the pre-filled videos to an image inpainting model [61] and our model, respectively. Our hallucinated result has a much larger PSNR value than the image inpainting model on DAVIS dataset (31.74 vs. 30.80 (dB)).

Table 5. Parameters comparisons. FuseFormer* denotes a larger version of original FuseFormer.

	FuseFormer [33]	FuseFormer*	E ² FGVI
Params. (M)	36.6	41.6	41.8
PSNR/SSIM	31.74/0.9662	31.91/0.9669	32.35/0.9688

B.4. Parameter comparison

We report the parameters in Tab. 5. Although our method consumes $\sim 14\%$ more parameters than the SOTA method (*i.e.*, FuseFormer [33]), it achieves a great trade-off between performance and computational complexity among other methods (see Tab. 5). For further comparison, we add

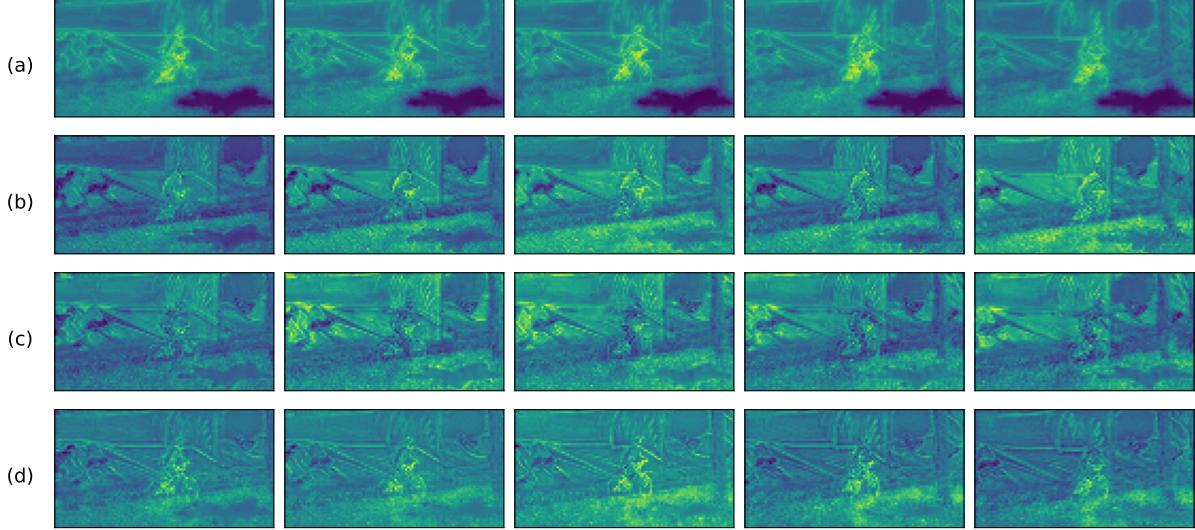


Figure 10. Visualization of the frame-wise average features before feeding into the content hallucination stage under different experimental settings: (a) without flow-guided feature propagation, (b) flow-guided feature propagation without deformable convolution (Eq. 3 of the main paper), (c) feature propagation without flow guidance, and (d) final flow-guided feature propagation module with the assistance of both flow fields and deformable convolution. **(Zoom-in for best view)**

residual blocks in FuseFormer to achieve similar parameters with ours. Our method still performs better than the larger version of FuseFormer.

B.5. More Qualitative Results

In this section, we provide additional visual results on two benchmark datasets, including YouTube-VOS [56] and DAVIS [44], to further show the superiority of the proposed E²FGVI. The reconstruction results of CAP [28], FGVC [17], and FuseFormer [33] are presented for comparisons. As shown in Fig. 11-14, our E²FGVI can generate more faithful textural and structural information and more coherent contents in masked regions than other methods. **Our demo is shown in our project page.**

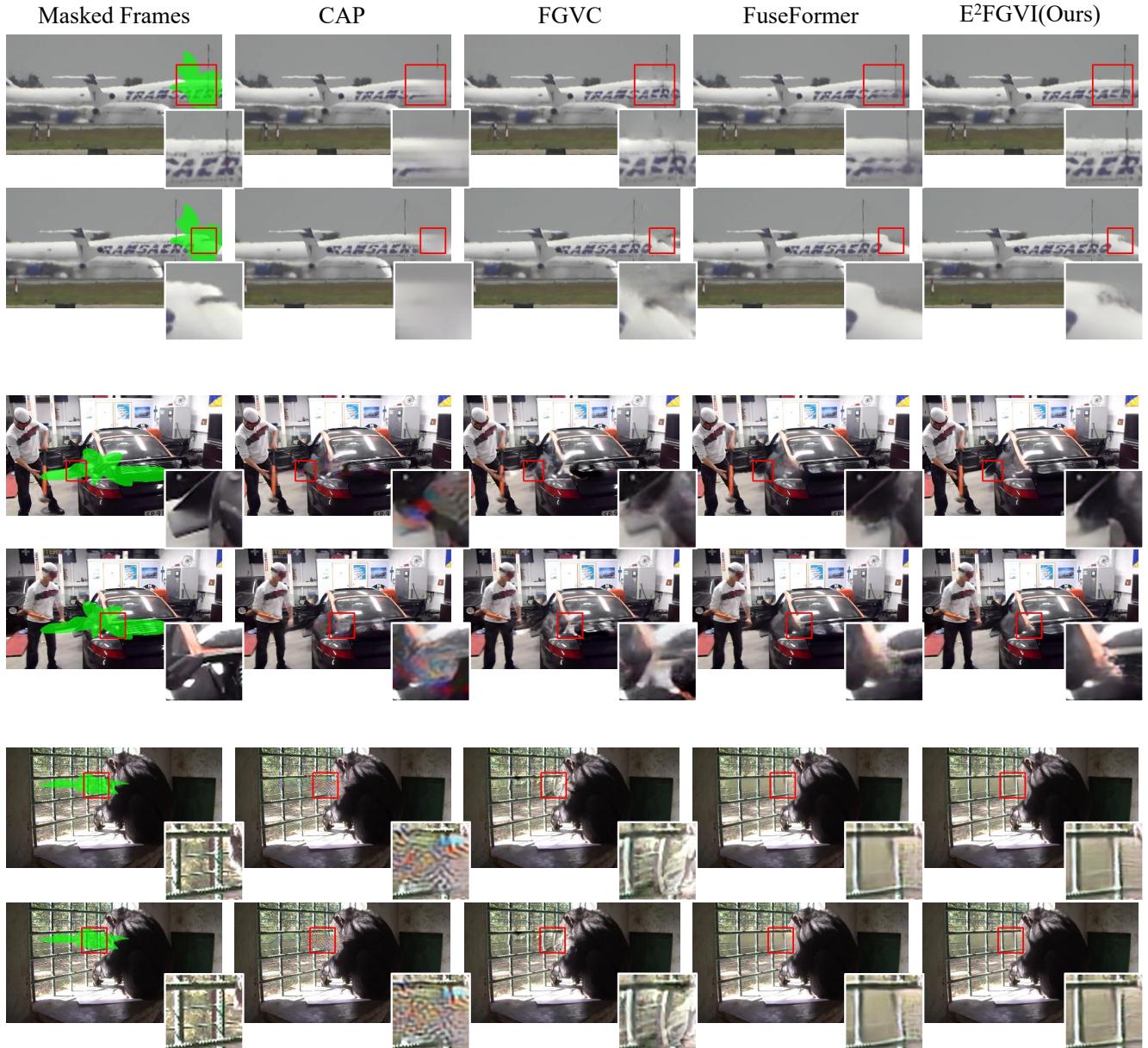


Figure 11. Qualitative video completion results on YouTube-VOS [56].



Figure 12. Qualitative video completion results on YouTube-VOS [56].

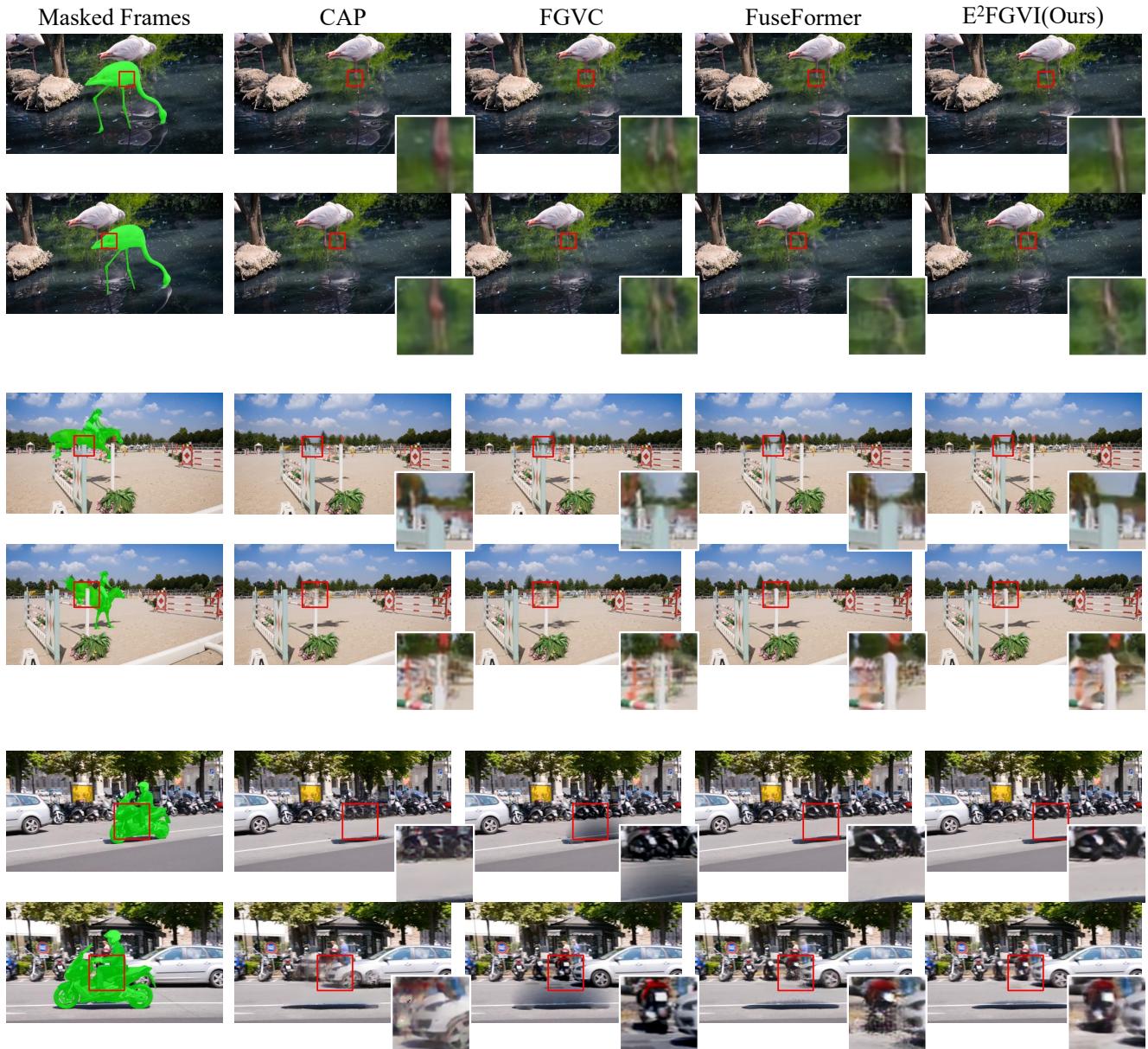


Figure 13. Qualitative object removal results on DAVIS [44].



Figure 14. Qualitative video completion results on DAVIS [44].