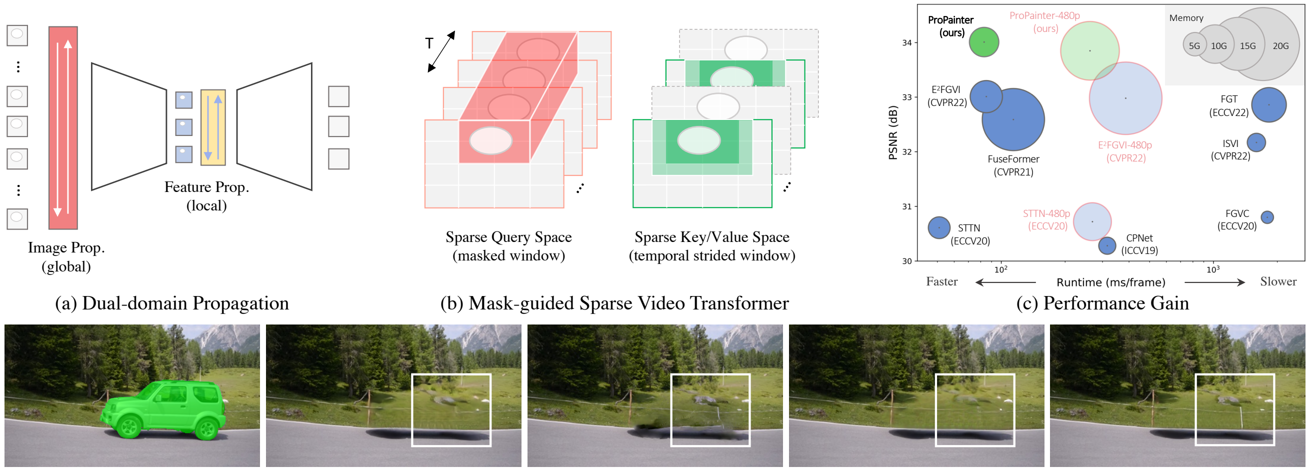
This ICCV paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the accepted version;
the final published version of the proceedings is available on IEEE Xplore.

ProPainter: Improving Propagation and Transformer for Video Inpainting

Shangchen Zhou Chongyi Li Kelvin C.K. Chan Chen Change Loy
S-Lab, Nanyang Technological University
{s200094, chongyi.li, chan0899, ccloy}@ntu.edu.sg
https://shangchenzhou.com/projects/ProPainter



(a) Dual-domain Propagation (b) Mask-guided Sparse Video Transformer (c) Performance Gain

(d) Input with object mask (e) FuseFormer (f) FGT (g) E^{2}FGVI (h) ProPainter (ours)

Figure 1: (a) Dual-domain propagation enables more effective propagation due to its global and reliable nature. (b) Mask-guided sparse video Transformer achieves high efficiency by discarding unnecessary and redundant windows. (c) ProPainter outperforms prior methods while maintaining efficiency. (d-h) In visual comparisons with FuseFormer [22], FGT [42], and E^{2}FGVI [19], our ProPainter exhibits superiority in filling complete and rich textures.

Abstract

Flow-based propagation and spatiotemporal Transformer are two mainstream mechanisms in video inpainting (VI). Despite the effectiveness of these components, they still suffer from some limitations that affect their performance. Previous propagation-based approaches are performed separately either in the image or feature domain. Global image propagation isolated from learning may cause spatial misalignment due to inaccurate optical flow. Moreover, memory or computational constraints limit the temporal range of feature propagation and video Transformer, preventing exploration of correspondence information from distant frames. To address these issues, we propose an improved framework, called ***ProPainter****, which involves enhanced* ***ProPa****gation and an efficient* ***T****rans****former****. Specifically, we introduce dual-domain propagation that combines the advantages of image and feature warping, exploiting global correspondences reliably. We also propose a mask-guided sparse video Transformer, which achieves high efficiency by discarding unnecessary and redundant tokens. With these components, ProPainter outperforms prior arts by a large margin of 1.46 dB in PSNR while maintaining appealing efficiency.*

1. Introduction

Video inpainting (VI) aims to fill gaps or missing regions in a video with visually consistent content while ensuring spatial and temporal coherence. This technique has broad applications, including video completion [10], object removal [9, 37], video restoration [31], watermark, and logo removal [19]. VI is challenging because it requires establishing accurate correspondence across distant frames for information aggregation. To address this challenge, various mechanisms have been explored, such as 3D CNN [6, 11], video internal learning [41, 27], flow-guided propagation [37, 10, 43, 42, 19], and video Transformer [22, 42, 19]. Among these mechanisms, flow-guided propagation and video Transformer have become mainstream choices for VI due to their promising performance.

Propagation-based methods in VI can be divided into two categories: image propagation and feature propagation. The former employs bidirectional global propagation in the image domain with a pre-completed flow field. While this approach can fill the majority of holes in a corrupted video, it requires an additional image or video inpainting network after propagation to hallucinate the remaining missing regions. This isolated two-step process can result in unpleasant artifacts and texture misalignment due to inaccurate flow, as shown in Figure 1(f). To address this issue, a recent approach called E^{2}FGVI [19] implements propagation in the feature domain, incorporating flow completion and content hallucination modules in an end-to-end framework. With the learnable warping module, the feature propagation module relieves the pressure of having inaccurate flow. However, E^{2}FGVI employs a downsampled flow field to match the spatial size of the feature domain, limiting the precision of spatial warping and the efficacy of propagation, potentially resulting in blurry results. Moreover, feature propagation can only be performed within a short range of video sequences due to memory and computational constraints, hindering propagation from distant frames and leading to missing texture, as shown in Figure 1(g).

Both image- and feature-based propagation have their pros and cons. In this study, we carefully revisit the VI problem and investigate the possibility of combining the strengths of both techniques. We demonstrate that with systematic redesigns and adaptation of best practices in the literature, we can achieve **dual-domain propagation**, as illustrated in Figure 1(a). To achieve reliable and efficient information propagation across a video, we identify several essential components: *i) Efficient GPU-based propagation with reliability check* – Unlike previous methods that rely on complex and time-consuming CPU-centric operations, such as indexing flow trajectories, we perform global image propagation on GPU with flow consistency check. This implementation can be inserted at the beginning of the inpainting network and jointly trained with the other modules. Thus, subsequent modules are able to correct any propagation errors and benefit from the long-range correspondence information provided by the global propagation, resulting in a significant performance improvement. *ii) Improved feature propagation* – Our implementation of feature propagation leverages flow-based deformable alignment [3], which improves robustness to occlusion and inaccurate flow completion compared to E^{2}FGVI [19]. *iii) Efficient flow completion* – We design a highly efficient recurrent network to complete flows for dual-domain propagation, which is over 40 times ($\sim$192 fps[1]) faster than SOTA method [43] while maintaining comparable performance. We demonstrate that these designs are essential to achieve efficient propagation of global and local information without texture misalignment or blurring in the filling results. An example is shown in Figure 1(h).

In addition to dual-domain propagation, we introduce an efficient **mask-guided sparse video Transformer** tailored for the VI task. The classic spatiotemporal Transformer is computationally intensive due to the quadratic number of interactions between video tokens, making it intractable for high-resolution and long temporal-length videos. For instance, contemporary Transformer-based methods, FuseFormer [22] and FGT [42], are unable to handle 480p videos with a 32G GPU[1] due to excessive memory demands. However, we observe that the inpainting mask usually covers only a small local region, such as the object area[2]. Moreover, adjacent frames contain highly redundant textures. These observations suggest that spatiotemporal attention is unnecessary for most unmasked areas, and it is adequate to consider only alternating interval frames in attention computation. Motivated by these observations, we redesign the Transformer by discarding unnecessary and redundant windows in the query and key/value space, respectively, significantly reducing computational complexity and memory without compromising inpainting performance.

The main contribution of this work is to provide a systematic study into the core problem of VI and offer a practical solution that is both effective and efficient. Propagating information in two distinct image and feature domains and combining them in a unified framework with fast GPU implementation is new for VI task. The mask-guided sparse video Transformer also offers practical insights into designing efficient spatiotemporal attention for VI task. Compared to the state-of-the-art methods, our model achieves superior performance with a large margin of 1.46 dB in PSNR, while also significantly reducing memory consumption.

2. Related Work

Numerous deep networks with different modules and propagation strategies have achieved significant success in video inpainting. These approaches can be broadly categorized into four categories:

3D convolution. Earlier video inpainting networks typically employed 3D CNNs [6, 33, 11] or temporal shift [7] to aggregate spatiotemporal information. These methods often suffer from limited receptive fields in both temporal and spatial dimensions and misalignment between adjacent frames. As a result, they are less effective in exploring distant content.

Internal learning. To fully exploit content of a video, some studies [41, 27, 30] adopt internal learning to encode and memorize the appearance and motion of the video through deep networks. However, these methods require individual training for each test video, limiting their practical use.

Flow-guided propagation. Optical flow [13, 18, 46] and homography [17, 1] are commonly used in video inpainting

[1] Tested on a single NVIDIA Tesla V100 GPU (32G).

[2] Object regions account for only 13.6% of the DAVIS [28] dataset.

networks to align neighboring reference frames to enhance temporal coherence and aggregation. However, incomplete optical flow may not provide valid propagation for completing missing regions. To address this issue, recent flow-based methods [37, 10, 12, 43, 42] focus on first completing the flow and then use it as a guidance for pixel-domain propagation. This approach simplifies RGB pixel inpainting by completing a less complex flow field. However, this offline propagation is independent of the subsequent learnable refinement module, making it difficult to correct content distortion caused by inaccurate propagation. Inspired by flow-guided recurrent networks [2, 3], Li et al. [19] proposed an end-to-end framework that jointly learns flow completion and feature propagation in the downsampled feature domain. However, downsampled flow reduces its ability to provide spatially precise warping. To overcome this limitation, we propose more faithful propagation by performing both pixel and feature propagation with flow consistency checks.

Video Transformer. Attention [17, 26, 11, 18] and Transformer [40, 21, 22, 1, 19, 42] blocks adopt spatiotemporal attention to explore recurrent textures in a video. This enables them to retrieve and aggregate tokens with similar texture or context for filling in missing regions. Liu *et al.* [22] present a fine-grained fusion Transformer based on the soft split and composition operations, which further boosts video inpainting performance. However, these methods are computationally and memory intensive. To address this issue, some Transformers [21, 1, 42] decouple the spatiotemporal attention by performing spatial and temporal attention alternately, while others [19, 42] adopt window-based Transformers [23, 38] to reduce the spatial range for efficient video attention. However, these approaches still involve redundant or unnecessary tokens. Inspired by token pruning for adaptive attention [29, 39, 25, 20, 15] in high-level tasks, our study proposes a more efficient and faster video Transformer with sparse spatiotemporal attention and a largely reduced token space while maintaining inpainting performance.

Recent studies [18, 19, 42] have demonstrated the effectiveness of combining flow-guided propagation with Transformer in VI. However, the high memory requirement of the Transformer limits the propagation range during both training and inference, severely hindering the ability to propagate temporally distant content. In this paper, we also adopt this combination strategy but propose a reliable propagation scheme, along with an efficient Transformer model that fully exploits the benefits of long-range propagation and attention. This results in superior inpainting performance while maintaining computational efficiency.

3. Methodology

Given a masked video sequence $X = \{X_t \in \mathbb{R}^{H\times W\times 3}\}_{t=1}^{T}$, which has a sequence length of T, along with corresponding mask sequence $M = \{M_t \in \mathbb{R}^{H\times W\times 1}\}_{t=1}^{T}$, the objective of video inpainting is to generate visually consistent and coherent content within the corrupted or missing regions. ProPainter, as shown in Figure 2, is composed of three key components: Recurrent Flow Completion (RFC), Dual-Domain Propagation (DDP), and Mask-guided Sparse Video Transformer (MSVT). Before feeding the sequence into ProPainter, we extract the forward and backward optical flows, denoted as $F^f = \{F_t^f = F_{t\rightarrow t+1} \in \mathbb{R}^{H\times W\times 2}\}_{t=1}^{T-1}$ and $F^b = \{F_t^b = F_{t+1\rightarrow t} \in \mathbb{R}^{H\times W\times 2}\}_{t=1}^{T-1}$ from a given video X. We first use RFC to complete the corrupted flow fields. Guided by the completed flows, we then perform global image propagation and local feature propagation sequentially. Finally, we employ multiple MSVT blocks to refine propagation features and a decoder to reconstruct the final video sequence $\hat{Y} = \{\hat{Y}_t \in \mathbb{R}^{H\times W\times 3}\}_{t=1}^{T}$. We introduce the specific design of each component below.

3.1. Recurrent Flow Completion

Pre-trained flow completion modules are commonly used in video inpainting networks [37, 10, 43, 42]. The rationale behind this approach is that it is simpler to complete missing flow than to directly fill in complex RGB content [37]. Furthermore, using completed flow to propagate pixels reduces the pressure of video inpainting and better maintains temporal coherence. E^{2}FGVI [19] proposes to insert the flow completion module into an end-to-end framework, which simplifies the inpainting pipeline. However, flow completion modules that are learned together with inpainting-oriented losses can result in a suboptimal learning process and less accurate completed flow. Moreover, the downsampled flow may limit the precision of spatial warping and the efficacy of propagation, which can result in blurred and incomplete filling content, as shown in Figure 1(g). Therefore, an independent flow completion model is not only important but also necessary for video inpainting.

To maintain temporal coherence while completing flows, previous methods [37, 42] adopt sliding-window-based networks to aggregate optical flow information from adjacent frames, which are highly correlated. However, these methods can be computationally expensive as repeated inferences are required in the overlapping frames. To improve efficiency and enhance flow coherence further, we adopt a recurrent network [2, 3] for flow completion, which provides precise optical flow fields for subsequent propagation modules.

We complete forward and backward flows using the same process, thus we denote F^f and F^b as F for simplicity. We first encode the flows F_t into a downsampled feature f_t with a downsampling ratio of 8. Next, we employ deformable alignment [3] that is based on deformable convolution (DCN) [8, 45], to bidirectionally propagate the flow information from nearby frames for flow completion. For simplicity, we only describe the backward propagation

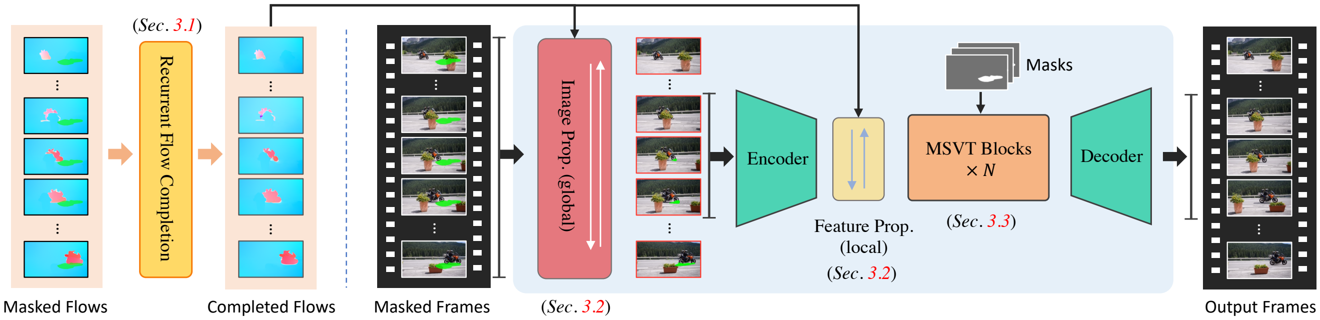


Figure 2: ProPainter comprises three key components: recurrent flow completion, dual-domain propagation, and mask-guided sparse Transformer. First, we employ a highly efficient recurrent flow completion network to complete the corrupted flow fields. We then perform propagation in both image and feature domains, which are jointly trained. This approach enables us to explore correspondences from both global and local temporal frames, resulting in more reliable and effective propagation. The subsequent mask-guided sparse Transformer blocks refine the propagated features using spatiotemporal attention, aided by a sparse strategy that considers only a subset of the tokens. This enhances efficiency and reduces memory consumption, while maintaining performance.

process here. Taking the concatenated feature $c(f_t, \hat{f}_{t+1})$, where $\hat{f}_{t+1}$ is the propagation feature of the `t+1`-th frame, as input a lightweight network with a stack of convolutions is employed to compute DCN offsets $o_{t\rightarrow t+1}$ and modulation masks $m_{t\rightarrow t+1}$. DCN alignment propagation can be expressed as:

$$\hat{f}_t = \mathcal{R}\big(\mathcal{D}(\hat{f}_{t+1}; o_{t\rightarrow t+1}, m_{t\rightarrow t+1}), f_t\big), \tag{1}$$

where $\mathcal{D}(\cdot)$ denotes deformable convolution, and $\mathcal{R}(\cdot)$ denotes the convolution layers that fuse the aligned and current features. In this way, information of $(t+1)$-th flow can be adaptively transferred to the current t-th flow. Finally, a decoder is used to reconstruct the completed flows $\hat{F}_t$. For clarity, an illustration of deformable alignment is provided in the supplementary material.

3.2. Dual-domain Propagation

After completing the flow, we perform global and local propagation in the image and feature domains, respectively. We employ distinct alignment operations and strategies for each domain. Both domains involve bidirectional propagation in the forward and backward directions. Here, we elaborate on the backward propagation since the forward propagation follows the same process.

Image propagation. To maintain efficiency and simplicity, we adopt flow-based warping for image propagation, along with a simple reliability check strategy. This process does not involve any learnable operation. In the case of a video sequence X with binary masks M (a pixel with value 1 represents masked region) and completed flows $\hat{F}$, we first check the validity of completed flow based on forward-backward consistency error [37, 10]:

$$\mathcal{E}_{t\rightarrow t+1}(p) = \left\| \hat{F}_{t\rightarrow t+1}(p) + \hat{F}_{t+1\rightarrow t}(p + \hat{F}_{t\rightarrow t+1}(p)) \right\|_2^2, \tag{2}$$

where p denotes a pixel position of the current frame. Only pixels with a small consistency error will be propagated, i.e., $C_1 : \mathcal{E}_{t\rightarrow t+1}(p) < \epsilon$, where ϵ is a threshold and set to 5. Furthermore, we only consider the masked areas of the current frame X_t that needs to be filled, i.e., $C_2 : M_t(p) = 1$, and we only propagate the unmasked areas from neighboring frame X_{t+1}, i.e., $C_3 : M_{t+1}(p + \hat{F}_{t\rightarrow t+1}(p)) = 0$. By enforcing the three constraints, a reliable propagation area A_r is identified as:

$$A_r(p) = \begin{cases} 1 & \text{if } p \in C_1 \cap C_2 \cap C_3, \\ 0 & \text{otherwise.} \end{cases} \tag{3}$$

The process of image propagation is expressed as:

$$\hat{X}_t = \mathcal{W}\big(X_{t+1}, \hat{F}_{t\rightarrow t+1}\big) * A_r + X_t * (1 - A_r), \tag{4}$$

where $\mathcal{W}(\cdot)$ denotes warping operation. To ensure continuous propagation, we promptly update the mask M_t of the current frame and convert the propagated area to the unmasked status by updating masks via $\hat{M}_t = M_t - A_r$. After global image propagation, we obtain a partially filled video sequence $\hat{X}$, which greatly eases the learning process for subsequent modules.

Feature propagation. We use an image encoder with the same structure as previous works [22, 19] to extract features from a local sequence $\hat{X}_{t=1}^{T_l}$, denoted as $\{e_t \in \mathbb{R}^{\frac{H}{4}\times\frac{W}{4}\times C}\}_{t=1}^{T_l}$. Similar to E^{2}FGVI [19], we also adopt flow-guided deformable alignment module [3] for feature propagation, which has demonstrated remarkable benefits

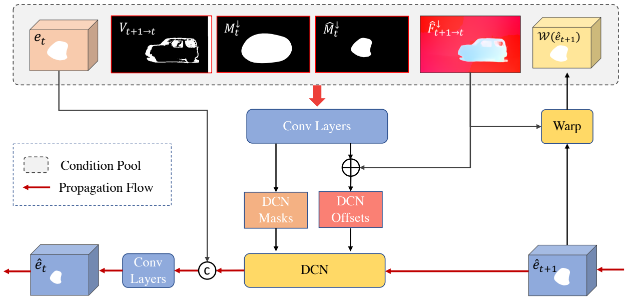


Figure 3: Flow-guided deformable alignment is effective by taking reliable completed flows and mask-aware conditions. We concatenate the validated flow map, original mask, and updated mask into conditions to produce DCN offsets (residue to optical flow). A DCN is then applied to align the propagation feature from the previous frame. Finally, a CNN block is employed to fuse the current and aligned features, achieving the propagation feature of the current frame.

in various low-level video tasks [5, 4, 44]. Unlike the deformable alignment used in Sec. 3.1 that directly learns DCN offsets, flow-guided deformable alignment employs the completed flow as a base offset and refines it by learning offset residue. However, our design differs from E^{2}FGVI in that we offer richer conditions for learning DCN offsets. As illustrated in Figure 3, apart from the current feature e_t, warped propagation feature $\mathcal{W}(\hat{e}_{t+1}, \hat{F}^{\downarrow}_{t\to t+1})$, and completed flows $\hat{F}^{\downarrow}_{t\to t+1}$, we additionally introduce the flow valid map $V_{t+1\to t}$ calculated by consistency check (Eq. 2), as well as the original mask $M_t^{\downarrow}$, and updated mask $\hat{M}_t^{\downarrow}$ after image propagation. With these conditions, a stack of convolutions is employed to predict the DCN offset residue $\widetilde{o}_{t\to t+1}$ and modulation masks $m_{t\to t+1}$. The flow-guided DCN alignment propagation is expressed as:

$$\hat{e}_t = \mathcal{R}\big(\mathcal{D}(\hat{e}_{t+1}; \hat{F}^{\downarrow}_{t\to t+1} + \widetilde{o}_{t\to t+1}, m_{t\to t+1}), f_t\big), \quad (5)$$

where $\downarrow$ denotes downsampling. The improved reliability of flow and the additional awareness of mask as a condition make our flow-guided deformable alignment module more stable to learn than previous designs [3, 19]. The current step is able to focus more on truly challenging regions where flow is invalid and former image propagation is unreliable.

3.3. Mask-Guided Sparse Video Transformer

While video Transformers have achieved excellent performance in video inpainting, they can be computationally and memory intensive, posing a challenge to their practical application. E^{2}FGVI and FGT have addressed this issue by using window-based Transformer blocks, but they still have some efficiency limitations. To overcome this, we propose a novel sparse video Transformer that builds on the window-based approach. Given a video sequence feature $E_l \in \mathbb{R}^{T_l \times \frac{H}{4} \times \frac{W}{4} \times C}$, we use the soft split operation [22] to generate patch embeddings $Z \in \mathbb{R}^{T_l \times M \times N \times C_z}$. We partition Z into $m \times n$ non-overlapping windows, resulting in partitioned features $Z_w \in \mathbb{R}^{T_l \times m \times n \times h \times w \times C_z}$, where $m \times n$ and $h \times w$ are the number and size of the windows, respectively. We obtain the query Q, key K, and value V from Z_w through linear layers. We design sparse strategies for both query and key/value spaces separately. Note that we also apply the window expand strategy [22] and integrate global tokens [42] into key and value, enabling us to use a small window size of 5×9 in our experiments. We omit them from the following discussion since they do not affect our sparse strategy designs.

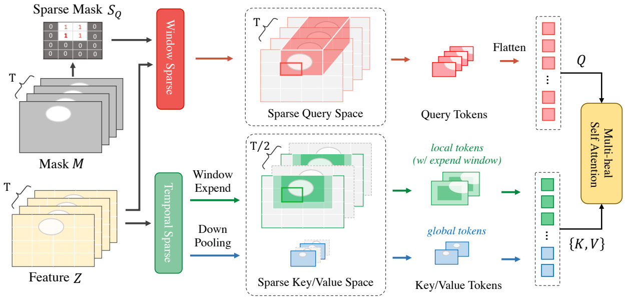


Figure 4: Mask-guided sparse video Transformer. To reduce computational complexity and memory usage, our mask-guided sparse Transformer filters out unnecessary and redundant windows in the query and key/value space, respectively, before applying self-attention. To enlarge spatial interrelation range, we also adopt the window expand strategy [38] and pooling global tokens [42, 19].

Sparse Query Space. We observe that mask regions often occupy only a small area of the video, such as in the case of object removal in the DAVIS [28] dataset, where the proportion of object regions is only 13.6%. This indicates that spatiotemporal attention may not be necessary for all query windows. To exploit this observation, we selectively apply attention to query windows that intersect with the mask regions. Specifically, we first use nearest neighbor interpolation to downsample the mask sequence $M \in \mathbb{R}^{T_l \times H \times W}$ to $M^{\downarrow} \in \mathbb{R}^{T_l \times m \times n}$, where $m \times n$ is the number of non-overlapping windows after partitioning. We then sum it up in the temporal dimension and obtain sparse mask $S_Q \in \mathbb{R}^{m \times n}$ for query cubes following the equation:

$$S_Q = Clip\left(\sum\nolimits_{t=1}^{T_l} M_t^{\downarrow},\, 1\right), \quad (6)$$

where $Clip$ represents a clipping function that set S_Q to 1 if $\sum_{t=1}^{T_l} M_t^{\downarrow} > 0$. In other words, if the query cube at a window (i, j) has never contained any mask region in the past frames, then $S_Q(i, j) = 0$, indicating that spatiotemporal attention within this window can be skipped.

Table 1: Quantitative comparisons on YouTube-VOS [36] and DAVIS [28] datasets. The best and second performances are marked in red and blue, respectively. E^*_{warp} denotes E_{warp} $(\times 10^{-3})$. All methods are evaluated following their default settings. Since DFVI, FGVC, ISVI, and FGT involve several CPU processes, their FLOPs cannot be accurately projected.

	Accuracy								Efficiency	
	YouTube-VOS				DAVIS				FLOPs	Runtime
Models	PSNR ↑	SSIM ↑	VFID ↓	E^*_{warp} ↓	PSNR ↑	SSIM ↑	VFID ↓	E^*_{warp} ↓	(10 frames)	(s/frame)
DFVI [37]	29.16	0.9429	0.066	1.651	28.81	0.9404	0.187	1.596	-	0.837
CPNet [17]	31.58	0.9607	0.071	1.622	30.28	0.9521	0.182	1.521	1407G	0.316
FGVC [10]	29.67	0.9403	0.064	1.163	30.80	0.9497	0.165	1.571	-	1.795
STTN [40]	32.34	0.9655	0.053	1.061	30.61	0.9560	0.149	1.438	1315G	0.051
TSAM [46]	30.22	0.9468	0.070	1.014	30.67	0.9548	0.146	1.235	1001G	0.068
FuseFormer [22]	33.32	0.9681	0.053	1.053	32.59	0.9701	0.137	1.349	1025G	0.114
ISVI [43]	30.34	0.9458	0.077	1.008	32.17	0.9588	0.189	1.291	-	1.594
FGT [42]	32.17	0.9599	0.054	1.025	32.86	0.9650	0.129	1.323	-	1.828
E^{2}FGVI [19]	33.71	0.9700	0.046	1.013	33.01	0.9721	0.116	1.289	986G	0.085
ProPainter (Ours)	34.43	0.9735	0.042	0.974	34.47	0.9776	0.098	1.187	808G	0.083

Sparse Key/Value Space. Due to the highly redundant and repetitive textures in adjacent frames, it is unnecessary to include all frames as key/value tokens in each Transformer block. Instead, we will only include strided temporal frames alternately, with a temporal stride of 2 in our design. That is, in each odd-numbered Transformer block, only odd-number frames are activated to participate in self-attention with their key and value, while even-number blocks include only even-number frames. By doing so, the key and value space is reduced by half, effectively reducing the computation and memory cost of the Transformer module. After filtering out unnecessary and redundant windows based on our sparse strategy, we perform self-attention on the remaining windows to extract refined features. These features are then gathered using a soft composition operation [22] for subsequent modules. Experimental results suggest that our design significantly reduces the computational cost of video Transformers while maintaining performance for video inpainting.

3.4. Training Objectives

Flow Completion. We utilize L1 loss as the reconstruction loss and a second-order smoothness constraint on the flow field [24] to promote the collinearity of neighboring flows and thus enhance the smoothness of the completed flow field.
Video Inpainting. We adopt L1 loss as the reconstruction loss for all pixels. To enhance the realistic and temporal consistency of video inpainting results, we also employ an adversarial loss that is measured using a T-PatchGAN [6] discriminator. The details and formulation of these losses are provided in the supplementary material.

4. Experiments

Datasets. We use the training set of YouTube-VOS [36] with 3471 video sequences to train our networks. Two widely-used test sets are adopted for evaluation: YouTube-VOS [36] and DAVIS [28], which consist of 508 and 90 video sequences, respectively. For the DAVIS test set, following FuseFormer [22] and E^{2}FGVI [19], we use 50 video clips for evaluations. During training, we follow [13, 17, 22, 19] and generate stationary and object masks in a random fashion to simulate the masks in video completion and object removal tasks. As for evaluation, we adopt the stationary masks provided in [19] to calculate quantitative scores, and the object masks are extracted from their segmentation labels for qualitative comparisons. Video frames are sized to 432×240 for training and evaluation.
Training Details and Metrics. We use RAFT [32] to extract optical flow in our approach. For training the RFC network, we set the flow sequence length to 10 and perform deformable propagation on feature maps that are downsampled by a factor of 8 for faster processing. We adopt 8 Transformer blocks for the inpainting modules and use a local video sequence of length 10. The Transformer window size is 5×9, and the extended size is half of the window size. We train both the RFC and inpainting modules using the Adam [14] optimizer with a batch size of 8, setting the initial learning rate to 10^{-4} and running 700k iterations[3] for each. We implement our method using the PyTorch framework and train it on 8 NVIDIA Tesla V100 (32G) GPUs.

We employ the widely used PSNR and SSIM metrics [35] to evaluate the reconstruction performance and VFID [34] scores to measure the perceptual similarity between input videos and outputs, as used in recent video inpainting studies [22, 19]. Additionally, we report the flow warping error E_{warp} [16] to assess the temporal consistency and smoothness of the resulting video sequences.

[3] We set 450k training iterations for ablation study.

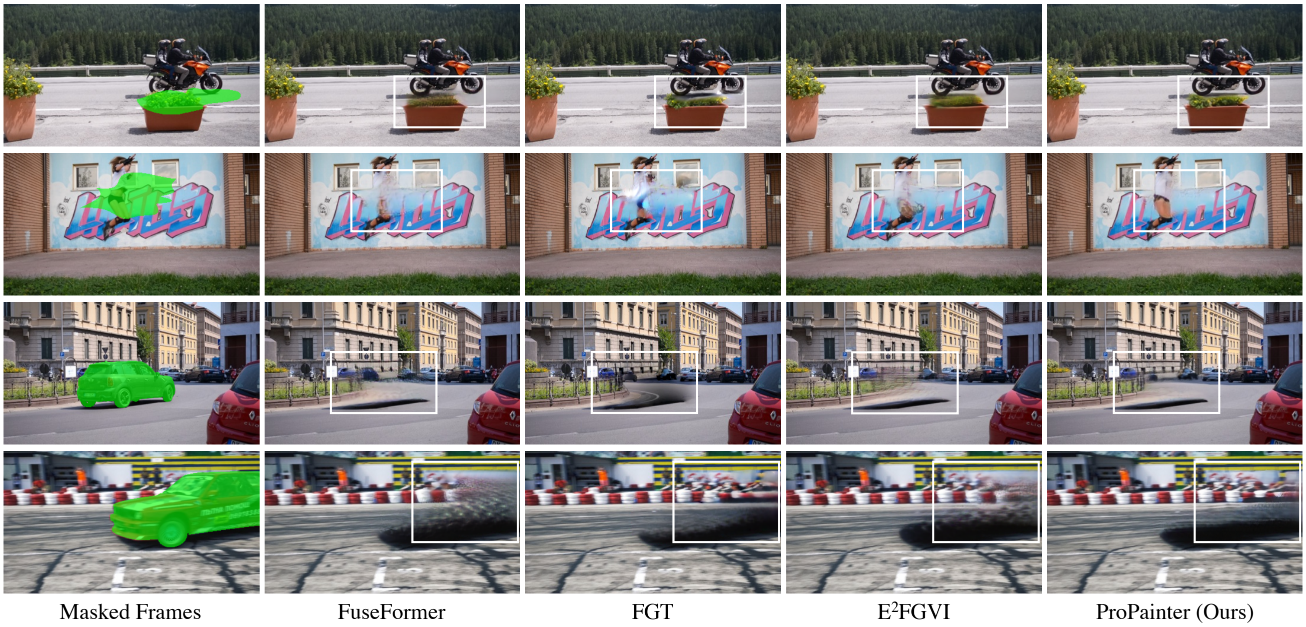


Figure 5: Qualitative comparisons on both video completion and object removal. Our ProPainter exhibits superiority in producing complete and faithful textures, resulting in enhanced spatiotemporal coherence for video inpainting.

4.1. Comparisons

Quantitative Evaluation. We compare ProPainter with nine state-of-the-art methods including DFVI [37], CPNet [17], FGVC [10], STTN [40], TSAM [46], Fuseformer [22], ISVI [43], FGT [42], and E^{2}FGVI [19] on both YouTube-VOS [36] and DAVIS [28]. Thanks to the efficient design, ProPainter uses a temporal length of 20 for inference. Table 1 shows that ProPainter outperforms other methods in all quantitative metrics, especially on the DAVIS dataset, where our method surpasses the state-of-the-art method by 1.14 dB in PSNR. The results suggest that our method has superior inpainting capability, enabling it to produce higher-quality, faithful, and seamless videos.

Qualitative Evaluation. For the visual comparison, we compare our method with FuseFormer [22], FGT [42], and E^{2}FGVI [19], which are representative methods of Transformer-, image propagation-, and feature propagation-based approaches, respectively. Figure 5 presents four comparison results for video completion and object removal. Our method uses dual-domain propagation to ensure reliable and long-range propagation. It completes missing regions with coherence and clear contents, while other compared methods tend to fail or produce unpleasant inpainting results such as texture distortions and black hazy region in FGT [42] results, as well as artifacts in FuseFormer [22] and E^{2}FGVI [19].

Efficiency Comparison. Table 1 presents the efficiency comparisons between all methods in terms of FLOPs and running time. The FLOPs of all methods are computed based on a temporal length of 10. We consider all learnable modules (including the recurrent flow completion) in our ProPainter to calculate the FLOPs. The running time records the time of all processes in each method, including inpainting, as well as flow calculation and flow completion if involved. To keep efficiency, we use only five iterations of the RAFT network to calculate optical flow.

Table 2: Comparisons of flow completion networks. Our network offers a dual benefit with high accuracy and efficiency.

EPE ↓	DFVI [37]	FGVC [10]	FGT [42]	ISVI [43]	Ours
YouTube-VOS	0.046	0.032	0.021	**0.019**	0.020
DAVIS	0.107	0.082	0.052	**0.051**	**0.051**
Runtime (s/frame)	0.130	1.125	0.312	0.231	**0.005**

Flow Completion Comparisons. We compare our recurrent flow network with previous approaches [37, 10, 43] on both YouTube-VOS and DAVIS datasets. Table 2 presents the end-point-error (EPE) of flow completion and running time of each method. Our recurrent network offers a dual benefit with high accuracy and efficiency. Compared to previous methods, our network is approximately 40 times faster while maintaining a comparable flow completion accuracy to the state-of-the-art methods.

4.2. Ablation Study

Effectiveness of Image Propagation. Table 3 shows that Exp. (a) experiences a significant performance drop when image propagation is removed. Moreover, the model's propagation ability is reduced without image propagation, as presented in Figure 7, causing it to fail to complete missing

Table 3: Ablation study of dual-main propagation and sparse Transformer.

Exp.	(a) w/o Img Prop.	(b) w/ Img Prop. in FGVC	(c) w/o Feat Prop.	(d) w/ Feat Prop. in E^{2}FGVI	(f) Full Tokens	ProPainter
PSNR	33.05	32.91	33.17	33.94	34.18	34.15
SSIM	0.9724	0.9687	0.9732	0.9756	0.9765	0.9764

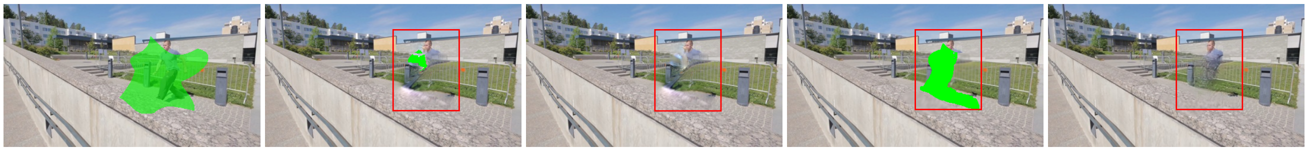

Masked Frame | Img Prop. of FGVC | w/ Img Prop. of FGVC (Exp. b) | Img Prop. (Ours) | ProPainter (Ours)

Figure 6: Visual comparison on image propagation methods of FGVC [10] and ours.

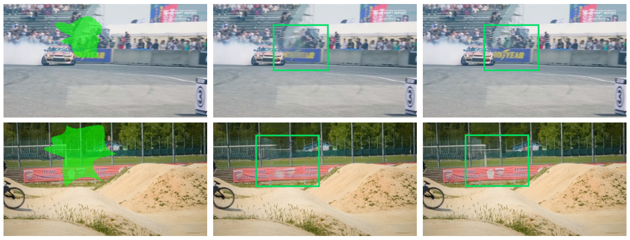

Masked Frames | w/o Img Prop. (Exp. a) | w/ Img Prop.

Figure 7: Comparison of w/ and w/o image propagation.

content with details. To verify the effectiveness of our reliability check strategy in image propagation, we replaced our design with the FGVC image propagation module in Exp. (b) (without retraining), resulting in a noticeable decrease in PSNR. This is because the FGVC image propagation method is prone to being affected by incorrect optical flow, leading to severe texture distortion that subsequent modules cannot correct. Our model can effectively aware and stop unreliable propagation areas using a simple reliability check via Eq.2, and generate more faithful inpainting results.

Effectiveness of Feature Propagation. Similarly, we observe a slight decrease in performance by either removing feature propagation, *i.e.*, Exp. (c), or replacing it with the Feature propagation of E^{2}FGVI, *i.e.*, Exp. (d), indicating the effectiveness of the feature propagation modules and our reliability mask-aware conditions. This suggests that our design, which learns reliable DCN offsets in the feature domain, can further complement and enhance the propagation ability in the image domain.

Effectiveness of Sparse Transformer. In theory, our strategy of using masks to guide sparsity only eliminates redundant and unnecessary tokens (windows), while preserving essential information. This means that there should be no adverse effect on performance. To confirm this, we conducted Exp. (d), comparing our approach to standard self-attention without sparse filtering. Our results indicate that our sparse Transformer block performs almost as well as the standard one, indicating that it can achieve high efficiency without sacrificing performance.

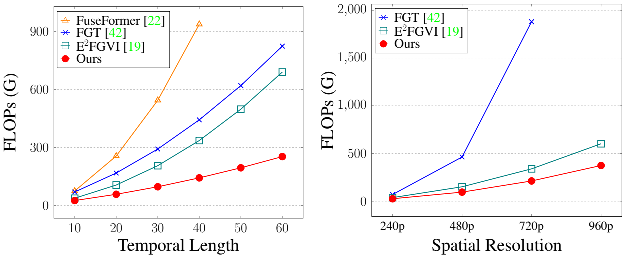


Figure 8: FLOPs cures of different Transformer blocks.

Efficiency of Sparse Transformer. In Figure 8, we compare the FLOPs of different Transformer blocks with respect to temporal length and spatial resolution, including those used in FuseFormer [22], FGT [42], and E^{2}FGVI [19]. We use a mask with a missing region ratio of 1/6 (higher than the average object ratio of 13.6% in DAVIS) to calculate the FLOPs of our mask-guided sparse Transformer. The curves indicate that the efficiency advantage of our sparse Transformer becomes more prominent as the temporal length and video resolution increase, indicating great potential for developing longer-range spatiotemporal attention and applying it to larger resolution videos.

5. Conclusion

This study introduces a novel and improved video inpainting framework called ProPainter. It incorporates an enhanced dual-domain propagation and an efficient mask-guided sparse video Transformer. Thanks to the two modules, our ProPainter exhibits reliable and precise propagation capabilities over long distances, significantly improving the performance of video inpainting while maintaining high efficiency in terms of running time and computational complexity. We believe that the designs in ProPainter will provide valuable insights to the video inpainting community.

Acknowledgement. This study is supported under the RIE2020 Industry Alignment Fund Industry Collaboration Projects (IAF-ICP) Funding Initiative, as well as cash and in-kind contribution from the industry partner(s).

References

[1] Jiayin Cai, Changlin Li, Xin Tao, Chun Yuan, and Yu-Wing Tai. DeViT: Deformed vision transformers in video inpainting. In *ACM MM*, 2022. 2, 3

[2] 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. 3

[3] Kelvin CK Chan, Shangchen Zhou, Xiangyu Xu, and Chen Change Loy. BasicVSR++: Improving video super-resolution with enhanced propagation and alignment. In *CVPR*, 2022. 2, 3, 4, 5

[4] Kelvin CK Chan, Shangchen Zhou, Xiangyu Xu, and Chen Change Loy. Investigating tradeoffs in real-world video super-resolution. In *CVPR*, 2022. 5

[5] Kelvin CK Chan, Shangchen Zhou, Xiangyu Xu, and Chen Change Loy. On the generalization of BasicVSR++ to video deblurring and denoising. *arXiv preprint arXiv:2204.05308*, 2022. 5

[6] 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, 6

[7] Ya-Liang Chang, Zhe Yu Liu, Kuan-Ying Lee, and Winston Hsu. Learnable gated temporal shift module for deep video inpainting. In *BMVC*, 2019. 2

[8] Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable convolutional networks. In *ICCV*, 2017. 3

[9] Mounira Ebdelli, Olivier Le Meur, and Christine Guillemot. Video inpainting with short-term windows: application to object removal and error concealment. *IEEE TIP*. 1

[10] Chen Gao, Ayush Saraf, Jia-Bin Huang, and Johannes Kopf. Flow-edge guided video completion. In *ECCV*, 2020. 1, 3, 4, 6, 7, 8

[11] Yuan-Ting Hu, Heng Wang, Nicolas Ballas, Kristen Grauman, and Alexander G Schwing. Proposal-based video completion. In *ECCV*, 2020. 1, 2, 3

[12] Lei Ke, Yu-Wing Tai, and Chi-Keung Tang. Occlusion-aware video object inpainting. In *ICCV*, 2021. 3

[13] Dahun Kim, Sanghyun Woo, Joon-Young Lee, and In So Kweon. Deep video inpainting. In *CVPR*, 2019. 2, 6

[14] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2014. 6

[15] Zhenglun Kong, Peiyan Dong, Xiaolong Ma, Xin Meng, Wei Niu, Mengshu Sun, Bin Ren, Minghai Qin, Hao Tang, and Yanzhi Wang. SpViT: Enabling faster vision transformers via soft token pruning. In *ECCV*, 2022. 3

[16] Wei-Sheng Lai, Jia-Bin Huang, Oliver Wang, Eli Shechtman, Ersin Yumer, and Ming-Hsuan Yang. Learning blind video temporal consistency. In *ECCV*, 2018. 6

[17] Sungho Lee, Seoung Wug Oh, DaeYeun Won, and Seon Joo Kim. Copy-and-paste networks for deep video inpainting. In *ICCV*, 2019. 2, 3, 6, 7

[18] 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, 3

[19] Zhen Li, Cheng-Ze Lu, Jianhua Qin, Chun-Le Guo, and Ming-Ming Cheng. Towards an end-to-end framework for flow-guided video inpainting. In *CVPR*, 2022. 1, 2, 3, 4, 5, 6, 7, 8

[20] Youwei Liang, Chongjian Ge, Zhan Tong, Yibing Song, Jue Wang, and Pengtao Xie. Not all patches are what you need: Expediting vision transformers via token reorganizations. In *ICLR*, 2022. 3

[21] 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. 3

[22] 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, 3, 4, 5, 6, 7, 8

[23] 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. 3

[24] Simon Meister, Junhwa Hur, and Stefan Roth. Unflow: Unsupervised learning of optical flow with a bidirectional census loss. In *AAAI*, 2018. 6

[25] Lingchen Meng, Hengduo Li, Bor-Chun Chen, Shiyi Lan, Zuxuan Wu, Yu-Gang Jiang, and Ser-Nam Lim. AdaViT: Adaptive vision transformers for efficient image recognition. In *CVPR*, 2022. 3

[26] Seoung Wug Oh, Sungho Lee, Joon-Young Lee, and Seon Joo Kim. Onion-peel networks for deep video completion. In *ICCV*, 2019. 3

[27] Hao Ouyang, Tengfei Wang, and Qifeng Chen. Internal video inpainting by implicit long-range propagation. In *ICCV*, 2021. 1, 2

[28] 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, 5, 6, 7

[29] Yongming Rao, Wenliang Zhao, Benlin Liu, Jiwen Lu, Jie Zhou, and Cho-Jui Hsieh. DynamicViT: Efficient vision transformers with dynamic token sparsification. In *NeurIPS*, 2021. 3

[30] Jingjing Ren, Qingqing Zheng, Yuanyuan Zhao, Xuemiao Xu, and Chen Li. DLFormer: Discrete latent transformer for video inpainting. In *CVPR*, 2022. 2

[31] Nick C Tang, Chiou-Ting Hsu, Chih-Wen Su, Timothy K Shih, and Hong-Yuan Mark Liao. Video inpainting on digitized vintage films via maintaining spatiotemporal continuity. *IEEE TMM*. 1

[32] Zachary Teed and Jia Deng. Raft: Recurrent all-pairs field transforms for optical flow. In *ECCV*, 2020. 6

[33] Chuan Wang, Haibin Huang, Xiaoguang Han, and Jue Wang. Video inpainting by jointly learning temporal structure and spatial details. In *AAAI*, 2019. 2

[34] 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. 6

[35] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *TIP*, 2004. 6

[36] 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, 7

[37] Rui Xu, Xiaoxiao Li, Bolei Zhou, and Chen Change Loy. Deep flow-guided video inpainting. In *CVPR*, 2019. 1, 3, 4, 6, 7

[38] 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. 3, 5

[39] Hongxu Yin, Arash Vahdat, Jose M Alvarez, Arun Mallya, Jan Kautz, and Pavlo Molchanov. A-ViT: Adaptive tokens for efficient vision transformer. In *CVPR*, 2022. 3

[40] Yanhong Zeng, Jianlong Fu, and Hongyang Chao. Learning joint spatial-temporal transformations for video inpainting. In *ECCV*, 2020. 3, 6, 7

[41] Haotian Zhang, Long Mai, Ning Xu, Zhaowen Wang, John Collomosse, and Hailin Jin. An internal learning approach to video inpainting. In *ICCV*, 2019. 1, 2

[42] Kaidong Zhang, Jingjing Fu, and Dong Liu. Flow-guided transformer for video inpainting. In *ECCV*, 2022. 1, 2, 3, 5, 6, 7, 8

[43] Kaidong Zhang, Jingjing Fu, and Dong Liu. Inertia-guided flow completion and style fusion for video inpainting. In *CVPR*, 2022. 1, 2, 3, 6, 7

[44] Shangchen Zhou, Jiawei Zhang, Jinshan Pan, Haozhe Xie, Wangmeng Zuo, and Jimmy Ren. Spatio-temporal filter adaptive network for video deblurring. In *ICCV*, 2019. 5

[45] Xizhou Zhu, Han Hu, Stephen Lin, and Jifeng Dai. Deformable convnets v2: More deformable, better results. In *CVPR*, 2019. 3

[46] Xueyan Zou, Linjie Yang, Ding Liu, and Yong Jae Lee. Progressive temporal feature alignment network for video inpainting. In *CVPR*, 2021. 2, 6, 7