Supplementary Material: Flow-Guided Transformer for Video Inpainting

Kaidong Zhang[1], Jingjing Fu[2(✉)], and Dong Liu[1]

[1]University of Science and Technology of China [2]Microsoft Research Asia
richu@mail.ustc.edu.cn, jifu@microsoft.com, dongeliu@ustc.edu.cn

1 Network structure

In our method, we adopt three networks, including RAFT [7] for optical flow extraction, **L**ocal **A**ggregation **F**low **C**ompletion network (LAFC) for flow completion, and **F**low-**G**uided **T**ransformer (FGT) for content synthesis. We will provide detailed description of two designed network structures in the following subsections.

1.1 Local aggregation flow completion network

In this part, we provide detailed network structure of the **L**ocal **A**ggregation **F**low **C**ompletion network (LAFC). We illustrate the network structure in Fig. 1. We also provide the detailed kernel size, stride and channel size in Tab 1.

1.2 Flow-Guided Transformer

For fair comparisons, we keep the encoder for frame patch embedding and the decoder the same as FFM [5]. As for the encoder for flow patch embedding, we adopt a relatively smaller CNN encoder, because the flows have been completed by LAFC. The patch size is 7×7, stride is 3×3 and padding is 3×3. The channel number of the frame token is 512, which is also kept the same as FFM [5], and the channel number of the flow token is 256. The detailed network structure can be viewed in Tab. 2.

2 Detailed loss function

2.1 Loss function in LAFC

LAFC uses L1 loss to penalize the completed target flow in the corrupted and the valid regions.

$$\begin{aligned} L_c &= \left\| M_t \odot (F_t - \hat{F}_t) \right\|_1 / \left\| M_t \right\|_1 \\ L_v &= \left\| (1 - M_t) \odot (F_t - \hat{F}_t) \right\|_1 / \left\| (1 - M_t) \right\|_1 \end{aligned} \tag{1}$$

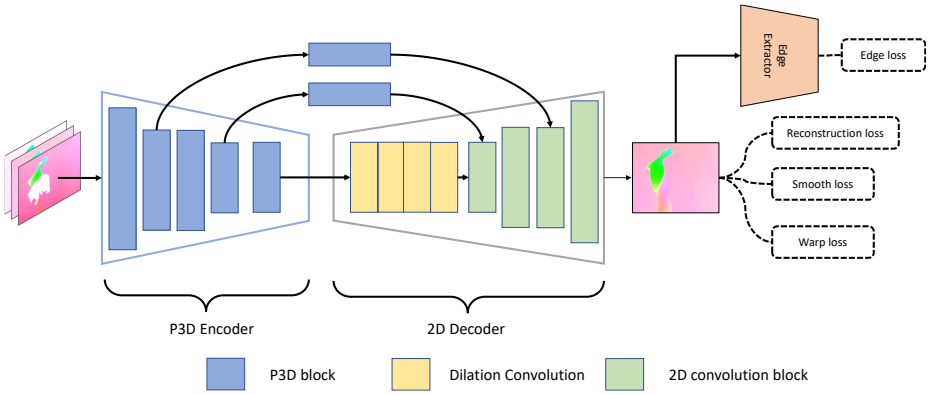


Fig. 1: The detailed network structure of the local aggregation flow completion network

where $\odot$ represents Hadamard product. L_c and L_v represent the reconstruction loss in the corrupted and the valid regions, respectively.

Since the optical flows are piece-wise smooth, we impose first-order and second-order smoothness loss to $\hat{F}_t$.

$$L_s = \left\| \nabla \hat{F}_t \right\|_1 + \left\| \triangle \hat{F}_t \right\|_1 \tag{2}$$

LAFC adopts the combination of L_c, L_v, L_s, L_w and L_e as the loss function, which is formulated as.

$$L_F = \lambda_1 L_c + \lambda_2 L_v + \lambda_3 L_s + \lambda_4 L_w + \lambda_5 L_e \tag{3}$$

We simply set λ_1, λ_2 and λ_5 to 1, λ_3 to 0.5 and λ_4 to 0.01 for the balance of magnitude between different loss terms. Our warp loss is borrowed from VINet [3], which expels the occlusion regions during warp loss calculation.

2.2 Loss function in FGT.

FGT utilizes the combination of reconstruction loss in corrupted and valid regions L_{yc} and L_{yv} together with adversarial loss L_{adv} to guide the training process. The reconstruction loss is shown below.

$$\begin{aligned} L_{yc} &= \left\| M_t \odot (Y_t - \hat{Y}_t) \right\|_1 / \left\| M_t \right\|_1 \\ L_{yv} &= \left\| (1 - M_t) \odot (Y_t - \hat{Y}_t) \right\|_1 / \left\| (1 - M_t) \right\|_1 \end{aligned} \tag{4}$$

We adopt hinge loss as the adversarial loss.

$$L_{adv} = -\mathbb{E}_{z \sim P_{\hat{Y}_t(z)}}[D(z)] \tag{5}$$

Therefore, the generator loss is the combination of the loss terms described above.

Module	Block	Filter size	In channels	Out channels	Stride/Up	Dilation
P3D encoder	P3D Conv	(3,5,5)	3	48	1	1
	P3D Conv	(3,3,3)	48	96	2↓	1
	P3D Conv	(3,3,3)	96	144	1	1
	P3D Conv	(3,3,3)	144	192	2↓	1
	P3D Conv	(3,3,3)	192	192	1	1
Skip connection	P3D Conv	(3,3,3)	96	96	1	1
	P3D Conv	(3,3,3)	192	192	1	1
Decoder	2DConv	(3,3)	192	192	1	8
	2DConv	(3,3)	192	192	1	4
	2DConv	(3,3)	192	192	1	2
	2DConv	(3,3)	192	192	1	1
	2DConv	(3,3)	384	144	2↑	1
	2DConv	(3,3)	144	96	1	1
	2DConv	(3,3)	192	48	2↑	1
	2DConv	(3,3)	48	24	1	1
	2DConv	(3,3)	24	2	1	1

Table 1: The detailed structure of LAFC. The filter size of P3D Conv is arranged as (t, h, w). We keep the temporal resolution unchanged except the last block in P3D Encoder and the blocks in skip connection. The input channel of the first P3D Conv is 3 because we concatenate the corrupted optical flows with their corresponding masks.

$$L_y = \lambda_{y1} L_{yc} + \lambda_{y2} L_{yv} + \lambda_{y3} L_{adv} \tag{6}$$

Following previous works [10, 5], we simply set λ_{y1} and λ_{y2} to 1, and λ_{y3} to 0.01. The discriminator loss is formulated as below.

$$\begin{aligned} L_D = &\mathbb{E}_{x \sim P_{Y_t}(x)}[\mathrm{ReLU}(1 + D(x))] \\ &+ \mathbb{E}_{z \sim P_{\hat{Y}_t}(z)}[\mathrm{ReLU}(1 - D(z))] \end{aligned} \tag{7}$$

Where D is the discriminator, Y_t represents the ground truth frame and $\hat{Y}_t$ denotes as the result frame.

3 The details of gradient propagation procedure

After flow completion, we propagate the content along all the frames under the guidance from completed flows. Following FGVC [2], we also propagate the gradient but not pixels, because the gradient propagation can avoid the warp error in low frequency component, which is beneficial for the overall visual quality. Given the completed forward optical flows $\hat{F}_f$ and backward optical flows $\hat{F}_b$, we warp the gradients extracted from the corrupted frames along the motion trajectory formed by the completed flows. If some regions can be filled by both

Module	Block	Filter size	In channels	Out channels	Stride/Up	Group/Head
Frame encoder	2D Conv	(3,3)	4	64	2↓	1
	2D Conv	(3,3)	64	64	1	1
	2D Conv	(3,3)	64	128	2↓	1
	2D Conv	(3,3)	128	256	1	1
	2D Conv	(3,3)	256	384	1	1
	2D Conv	(3,3)	640	512	1	2
	2D Conv	(3,3)	768	384	1	4
	2D Conv	(3,3)	640	256	1	8
	2D Conv	(3,3)	512	128	1	1
Flow encoder	2D Conv	(3,3)	3	64	1	1
	2D Conv	(3,3)	64	128	2↓	1
	2D Conv	(3,3)	128	128	1	1
	2D Conv	(3,3)	128	128	2↓	1
PEG	2D Conv	(3,3)	512	512	1	512
Trasformer blocks	T-S × 4	-	512	512	1	4
Decoder	2DConv	(3,3)	128	128	2↑	1
	2DConv	(3,3)	128	64	1	1
	2DConv	(3,3)	64	64	2↑	1
	2DConv	(3,3)	64	3	1	1

Table 2: The detailed structure of FGT. We borrow the frame encoder and the decoder from FFM [5]. "T-S" represents the interleaved temporal and spatial transformer blocks, which has been analyzed in the main paper. "Group" means the channel groups in convolution block and "Head" means the number of heads of MHSA in transformers.

forward and backward propagation, we fuse the content propagated from forward and backward direction based on the forward-backward consistency check, which can be formulated as.

$$\hat{C}_{t\to t-1}(p) = \left\| \hat{F}_b(t-1,p) + \hat{F}_f(t-1, p+\hat{F}_b(t-1,p)) \right\|_2^2 \tag{8}$$

where $\hat{C}_{t\to t-1}(p)$ represents the consistency between frame t and frame $t-1$ in pixel p, $\hat{F}_b(t-1,p)$ represents the backward warp of pixel p from frame t to frame $t-1$, and $\hat{F}_f(t-1,p)$ denotes as forward warp of pixel p from frame $t-1$ to frame t. Then, we calculate the fusion weights based on the forward-backward consistency check, as shown below.

$$\begin{aligned} \omega_{t\to t-1} &= \frac{exp(-\hat{C}_{t\to t-1}/d)}{exp(-\hat{C}_{t\to t-1}/d) + exp(-\hat{C}_{t\to t+1}/d) + \epsilon} \\ \omega_{t\to t+1} &= \frac{exp(-\hat{C}_{t\to t+1}/d)}{exp(-\hat{C}_{t\to t-1}/d) + exp(-\hat{C}_{t\to t+1}/d) + \epsilon} \end{aligned} \tag{9}$$

where $\omega_{t\to t-1}$ represents the fusion weight of the forward warped gradient from frame $t-1$ to frame t, and $\omega_{t\to t+1}$ represents the fusion weight of the backward warped gradient from frame $t+1$ to frame t. d is the temperature weight and ϵ is an extremely small number to prevent the zero division error. Following FGVC [2], we simply set d to 0.1 and ϵ to $1e-7$. We fuse the forward and backward warped gradient based on the weights $\omega_{t\to t-1}$ and $\omega_{t\to t+1}$.

$$\begin{aligned}\nabla_x \hat{X}_t &= \omega_{t\to t-1}\nabla_x X_{t-1} + \omega_{t\to t+1}\nabla_x X_{t+1}\\ \nabla_y \hat{X}_t &= \omega_{t\to t-1}\nabla_y X_{t-1} + \omega_{t\to t+1}\nabla_y X_{t+1}\end{aligned} \tag{10}$$

where $\nabla_x X_t$ represents the gradient map extracted from X_t along x direction, and $\nabla_y X_t$ represents the gradient map extracted from X_t along y direction.

After we obtain the warped gradient maps $\nabla_x \hat{X}$ and $\nabla_y \hat{X}$, we adopt Poisson blending [6] to synthesize the filled regions. As for the rest unfilled corrupted regions, we adopt our proposed **F**low-**G**uided **T**ransformer (FGT) to fill these regions.

4 More experiment results

4.1 The size of temporal window

We keep the spatial transformer as the same in the main paper, and adjust the zone size of temporal MHSA. As described in the main paper, we adopt 2×2 zones in our method, which means we generate 4 cubes across the spatial and temporal dimension, and perform temporal MHSA inside each cube. We provide the additional results on 1×1 and 4×4 zones in Tab. 3. Compared with the 1×1 zone, our method can achieve slightly better performance, which means the all-pair attention strategy adopted in STTN [10] or FFM [5] is not the best practice due to the consideration of too many irrelevant tokens. When we perform the temporal MHSA based on 4×4 zones, the performance drops significantly, which confirms our argument on window size. That is, the large window in temporal MHSA can compensate the position offset of the relevant content, which leads to more accurate attention retrieval.

4.2 The size of spatial window and global downsample size in dual perspective spatial MHSA.

We provide the quantitative results in Tab. 4. We adjust the range of global downsampling size (GD) from 2 to 16 while keep the local window size (LW) as 8, which is the default parameter we adopt for local window size. Compared with GD = 2 setting, our method (GD = 4) can achieve significant performance boost. Such comparison reveals downsampling by small ratio cannot purify the global token map effectively, and the resulted token maps undermine the attention retrieval performance. When GD further enlarges from 4 to 16, the performance

Table 3: Ablation study results about the zone size in temporal MHSA. We adopt the 2×2 zones, which means we divide the feaures into 4 cubes along height and width dimension. 1×1 zone means we do not adopt window partition in temporal transformer, but perform attention across all the tokens spatiotemporally. The bold font indicates our choice.

Zones	square			object		
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
1×1	31.49	0.958	0.039	33.11	0.945	0.050
2×2	**31.62**	**0.959**	**0.038**	**33.25**	**0.946**	**0.048**
4×4	31.39	0.957	0.040	33.02	0.945	0.050

Table 4: Ablation study about the local window size (LW) and the global downsampling size (GD). The last row (LW = 1 and GD = 1 indicates that we do not adopt the window partition strategy to the spatial transformer and each token in the token map can get access to all the other tokens in the same token map during attention retrieval. The bold font indicates our choice.

LW	GD	square			object		
		PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
8	2	31.47	0.958	0.039	33.02	0.946	0.050
	4	**31.62**	**0.959**	**0.038**	**33.25**	**0.946**	**0.048**
	8	31.53	0.959	0.038	33.09	0.946	0.049
	16	31.35	0.957	0.039	32.92	0.945	0.051
2	4	31.47	0.958	0.040	33.11	0.946	0.049
4		31.57	0.959	0.039	33.16	0.946	0.049
8		**31.62**	**0.959**	**0.038**	**33.25**	**0.946**	**0.048**
16		31.29	0.957	0.041	32.94	0.945	0.053
1	1	31.51	0.958	0.038	33.08	0.945	0.050

drops gradually, which indicates too large downsampling size may cause lost of useful information, which also causes a sub-optimal result.

As for the ablation about LW, we fix GD to 4, and adjust the size of LW from 2 to 16. When LW enlarges from 2 to 8, the performance gradually increases, which means the modest growth of LW is beneficial due to the rich context in the local window. However, when the size of LW further enlarges, the performance drops significantly. We argue that too large LW may include more irrelevant tokens, which may lead to interference between local and global tokens. Therefore, the "small window + global tokens" choice in dual perspective spatial MHSA is reasonable. What's more, compared with the spatial MHSA without window partition (the last row in Tab. 4), our method is competitive and even slightly better, which means all-pair attention in transformer is a sub-optimal choice due to the interference of irrelevant tokens.

Table 5: Ablation study about the flow tokens in the flow guidance integration module. w/o: Without the flow guidance integration module. S: Insert flow guidance integration module to the spatial transformer blocks. ST: Insert flow guidance integration module to both spatial and temporal transformer blocks.

Flows	square			object		
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
w/o Flow	31.62	0.959	0.038	33.25	0.946	0.048
S	31.87	0.961	0.036	33.52	0.947	0.045
ST	31.59	0.959	0.038	33.42	0.947	0.047

4.3 Add flow tokens to temporal transformer blocks

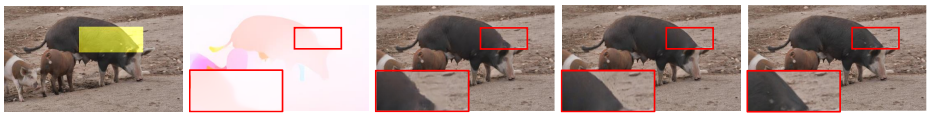
As discussed in the main paper, we only integrate flow tokens in the spatial transformer blocks. We argue that the locally correlated optical flows cannot provide reliable motion discrepancy to instruct the temporal attention process, especially when the frames are distant in temporal dimension. We provide the results of the flow guidance integration to both spatial and temporal temporal blocks in Tab. 5. We observe when we integrate flow information in the temporal transformer blocks, the performance does not get further improved. The integration of flow information to the temporal MHSA may even impact the attention retrieval process, which undermines the video inpainting quality.

4.4 Oracle study: FGT with GT flows.

We adopt the GT flows in the flow guidance integration module to perform an oracle study about the flow guidance in attention retrieval process, and we provide the results in Tab. 6. Although we can get performance boost with the guidance of our completed flows in FGT, there is obvious performance gap between our method and the method that adopts GT flows. That is to say, the performance of FGT is affected by the flow completion quality. FGT could achieve better performance, if we can generate completed flows with better quality. Therefore, designing an effective flow completion network is critical to enhance the performance of FGT. We leave the research of better flow completion method in the future work.

4.5 Qualitative comparisons about the flow-reweight module

In this part, we will provide qualitative comparisons about FGT w/ and w/o the flow-reweight module in Fig. 2, as an compensation of the quantitative results with the flow-reweight module in the main paper. Our flow-reweight module can relieve the negative impact of the completed flows with large distortion, with respect to the interaction between the frame and the flow features. As for the regions with small flow distortion, the flow-reweight module can fully utilize the motion discrepancy between objects and background for better attention retrieval.



(a) Input (b) Completed flow (c) w/o. flow-reweight (d) w/ flow-reweight (e) GT

Fig. 2: The qualitative comparisons about the flow-reweight module in the flow guidance integration. With flow-reweight module, our method can correct the regions with large flow distortion.

Table 6: Oracle study about the GT flows in the flow guidance integration module.

Flows	square			object		
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
Ours	31.87	0.961	0.036	33.52	0.947	0.045
GT	32.27	0.962	0.034	33.86	0.950	0.043

4.6 Qualitative comparisons between FGT and transformer-based video inpainting methods

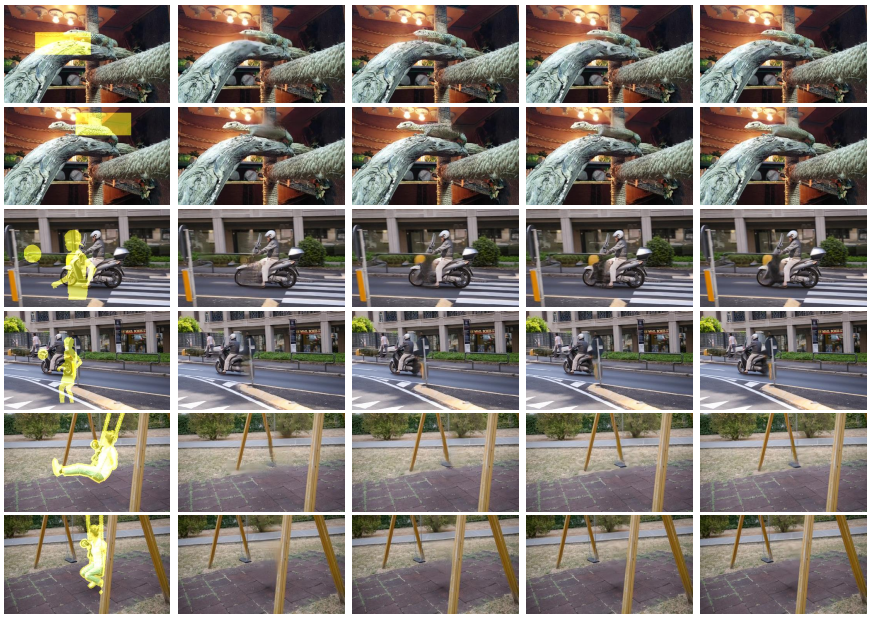
We provide the qualitative comparison between FGT and current transformer-based video inpainting methods [10, 5, 4] in Fig. 3. We adopt the FGT to synthesize all the pixels in the corrupted regions for fair comparisons. Compared with other transformer-based video inpainting methods, FGT can synthesize more complete structure and reasonable details.

5 User study

In order to validate the subjective quality of our method in object removal, we perform a user study. Our baseline includes one flow-guided method FGVC [2] and two transformer methods DSTT [4] and FFM [5]. We randomly sample 20 videos from DAVIS [1] and recruit 24 volunteers for user study. We illustrate the results in Fig. 4. Compared with previous baselines, our method can achieve impressive subjective quality.

6 Runtime analysis

We analyze the running time of our method. Our workstation equips a P40 GPU and an Intel(R) Xeon(R) CPU E5-2690 v4 @ 2.60GHz CPU. We omit the I/O overhead, because it varies across different workstations. We list the runtime of different components in our method in Tab. 7. The speed of our method is competitive with the FGVC [2]. What's more, if the corrupted regions are large, FGVC [2] needs more iterations to synthesize a video, which means the time-consuming Poisson blending has to be used multiple times. In such case, our method only performs Poisson blending once, and the rest regions are synthesized with FGT. As a result, our method is significantly faster than FGVC.



(a) Input (b) STTN [10] (c) DSTT [4] (d) FFM [5] (e) FGT

Fig. 3: The qualitative comparison between FGT and the recent baselines

If we remove the flow-guided content propagation stage, our method can obtain a even faster speed but at the sacrifice of the inpainting performance (shown in the quantitative analysis part in the main paper), which indicates the existence of performance-speed tradeoff in our method.

7 Limitations

Our method has limitations reflected in the following two aspects. First, compared with other transformer-based methods, the speed of FGT is relatively slow due to the existence of the flow extraction and completion stages. However, the motion discrepancy in the completed optical flows can boosts the performance of FGT, which indicates the performance-speed trade-off in our method. Second, when the motion magnitude is quite large in the videos, the flow completion quality of our method degrades accordingly, which affects the performance of flow-guided content propagation and FGT.

8 More visual results

To illustrate the frame-wise video inpainting quality, we provide more visual results about the flow completion quality in Fig. 5, the frame synthesis quality in Fig 6, the object removal performance in Fig.7 and the visual results about frame synthesis on Youtube-VOS [8] dataset in Fig. 8.

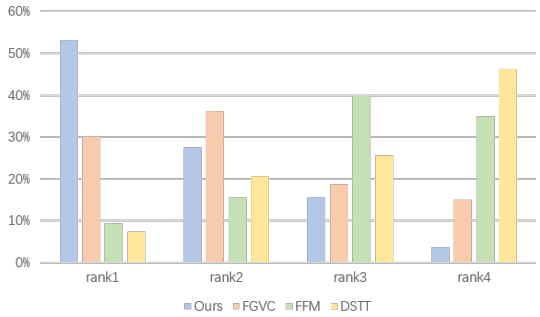


Fig. 4: The user study between our method and recent SOTA baselines, including FGVC [2], FFM [5] and DSTT [4].

Table 7: The runtime analysis on "bike-packing" sequence from DAVIS, which contains 68 frames. We adopt the square mask to corrupt this video sequence.

Time	Ours
Flow computation	16.13s
Flow completion	21.20s
Flow guided gradient propagation	10.43s
Poisson blending	87.91s
FGT content synthesis	8.42s
Sum on the full sequence	144.09s
Average run time of each frame	2.11s

(a) masked flow (b) DFGVI (c) FGVC (d) Ours (e) GT

Fig. 5: The visual comparisons of flow completion between our method and recent baselines, including DFGVI [9] and FGVC [2].

Fig. 6: The visual comparisons of frame-wise video inpainting quality on DAVIS dataset. We compare our method with recent baselines, including STTN [10], TSAM [11], DSTT [4], FFM [5] and FGVC [2]. Best viewed with zoom-in.

Fig. 7: The comparisons of the object removal performance between our method and the recent baselines, including STTN [10], TSAM [11], DSTT [4], FFM [5] and FGVC [2]. Best viewed with zoom-in.

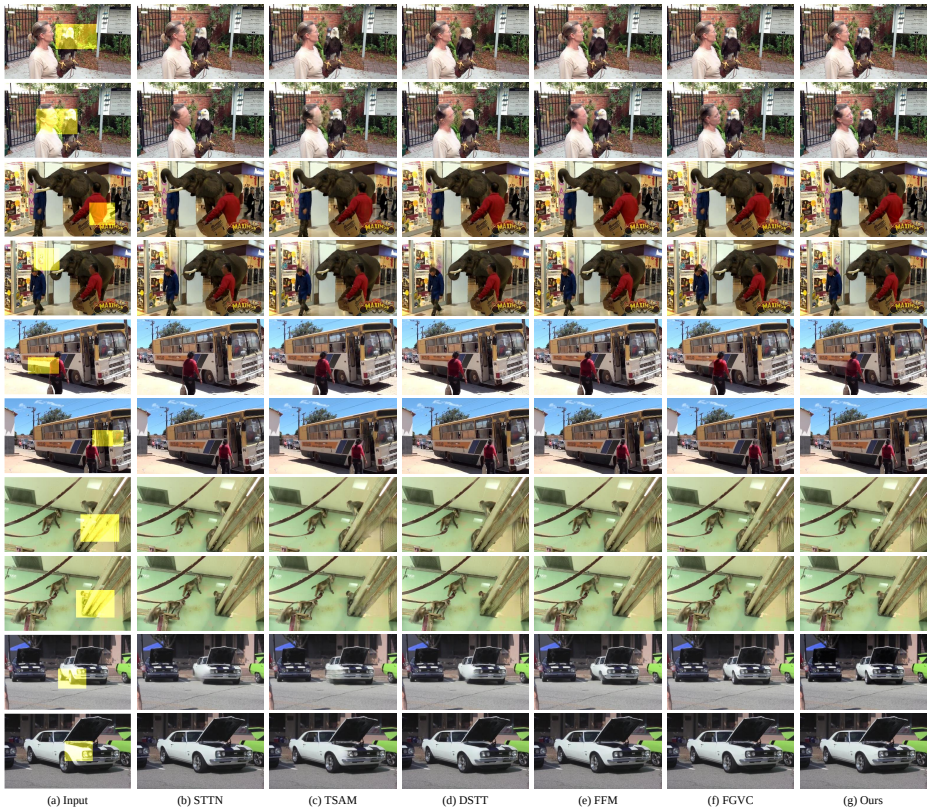


Fig. 8: The qualitative comparisons between our method and the recent baselines, including STTN [10], TSAM [11], DSTT [4], FFM [5] and FGVC [2] on Youtube-VOS dataset. Best viewed with zoom-in.

References

1. Caelles, S., Montes, A., Maninis, K.K., Chen, Y., Gool, L.V., Perazzi, F., Pont-Tuset, J.: The 2018 DAVIS challenge on video object segmentation. arXiv preprint arXiv:1803.00557 (2018)
2. Gao, C., Saraf, A., Huang, J.B., Kopf, J.: Flow-edge guided video completion. In: ECCV. pp. 713–729 (2020)
3. Kim, D., Woo, S., Lee, J.Y., Kweon, I.S.: Deep video inpainting. In: CVPR. pp. 5792–5801 (2019)
4. Liu, R., Deng, H., Huang, Y., Shi, X., Lu, L., Sun, W., Wang, X., Dai, J., Li, H.: Decoupled spatial-temporal transformer for video inpainting (2021)
5. Liu, R., Deng, H., Huang, Y., Shi, X., Lu, L., Sun, W., Wang, X., Dai, J., Li, H.: Fuseformer: Fusing fine-grained information in transformers for video inpainting. In: ICCV (2021)
6. Pérez, P., Gangnet, M., Blake, A.: Poisson image editing. TOG **22**(3), 313–318 (Jul 2003). https://doi.org/10.1145/882262.882269, https://doi.org/10.1145/882262.882269
7. Teed, Z., Deng, J.: Raft: Recurrent all-pairs field transforms for optical flow. In: ECCV. pp. 402–419. Springer (2020)
8. Xu, N., Yang, L., Fan, Y., Yue, D., Liang, Y., Yang, J., Huang, T.: Youtube-vos: A large-scale video object segmentation benchmark. arXiv preprint arXiv:1809.03327 (2018)
9. Xu, R., Li, X., Zhou, B., Loy, C.C.: Deep flow-guided video inpainting. In: CVPR. pp. 3723–3732 (2019)
10. Zeng, Y., Fu, J., Chao, H.: Learning joint spatial-temporal transformations for video inpainting. In: ECCV. pp. 528–543 (2020)
11. Zou, X., Yang, L., Liu, D., Lee, Y.J.: Progressive temporal feature alignment network for video inpainting. In: CVPR (2021)