HomoGen: Enhanced Video Inpainting via Homography Propagation and Diffusion

Supplementary Materials

Ding Ding[1*] Yueming Pan[2] Ruoyu Feng[3] Qi Dai[4] Kai Qiu[4] Jianmin Bao[4]
Chong Luo[4†] Zhenzhong Chen[1]
[1]Wuhan University [2]Xi'an Jiaotong University [3]University of Science and Technology of China
[4]Microsoft Research Asia

A. Architecture

HomoGen features a two-stage pipeline that consists of HPP and PCVDM. For more clarity, we detail the architectures of key components in HomoGen: 3D VAE, LiteControlNet, 3D U-Net, and perceiver attention layers.

3D VAE. We only integrate temporal layers into the original 2D VAE decoder from Stable Diffusion [9] to improve the temporal smoothness of generated videos. This modification ensures that the distribution of the latent space remains unchanged, where the latent variable is encoded from the fixed 2D VAE encoder, thereby reducing the burden of subsequent fine-tuning the 3D U-Net. We add a 1D temporal residual block following each 2D spatial residual block in the 3D VAE decoder. Additionally, we introduce a 1D temporal self-attention block at the beginning of the decoder. The hyperparameter settings for the 3D VAE refer to Tab. 1.

LiteControlNet. The proposed LiteControlNet extracts multi-scale features from the latents of the priors derived from HPP, built upon ControlNet [14] with a lightweight design. It stacks four $3 \times 3 \times 3$ convolutional layers, where the first one uses a stride of 1 and the last three use a stride of 2 for downsampling. Compared to ControlNet, LiteControlNet employs a single convolutional layer per latent scale, structurally aligned with the corresponding latent scale in the 3D U-Net encoder. Additionally, it removes the sampling step as input, for which the prior features are computed only once during sampling rather than at each step as in ControlNet. The hyperparameter settings for LiteControlNet refer to Tab. 1.

3D U-Net. To extend the original 2D U-Net to a 3D U-Net, we add a 1D temporal convolutional layer following each 2D spatial convolutional layer in the original 2D residual blocks, thereby constructing pseudo-3D residual blocks. Additionally, we introduce a 1D temporal self-attention block and a window-attention block after each 2D spatial self-attention block in the 3D U-Net. A cross-attention block is added after each window-attention block to inject the GoP-wise CLIP latents into the 3D U-Net encoder. As proposed in Sec. 3.3.2 of our main paper, the prior feature at each scale is projected by a projector, *i.e.*, a $1 \times 1 \times 1$ convolutional layer, then injected into and before each pseudo-3D residual block in the 3D U-Net encoder. The hyperparameter settings for the 3D U-Net refer to Tab. 1.

Perceiver Attention Layers. To extract spatiotemporal semantics from OpenCLIP image embeddings, we employ perceiver attention layers that iteratively distill inputs into a tight latent bottleneck, enabling scalability for large inputs. Four perceiver attention layers are stacked, each configured with an embedding dimension of 1280, a query length of 256, and 64 attention head channels.

B. Various Prior Injection Strategies

In Sec. 4.3.2 of our main paper, we compare the proposed content-adaptive control mechanism against alternative strategies, including pasting, concatenation, cross-attention, and noise prior, for injecting homography-based priors into PCVDM. Presented below are the detailed implementations of the injection strategies under comparison.

Pasting. Pasting represents pasting the priors into the corrupted regions directly before the corrupted video is fed into PCVDM. Subsequently, the mask $\mathbf{m}^p$ indicating the regions of the propagated pixels, is downsampled by a factor of 8 and concatenated with the downsampled mask $\mathbf{m}$, the noised video latent $\mathbf{z}^t$, and the corrupted video latent $\mathbf{z}$ in the latent space. These concatenated features are then fed into the 3D U-Net.

Concatenation. Concatenation represents concatenating the prior latent $\mathbf{z}^p = \mathcal{E}(\mathbf{x}^p)$ with downsampled $\mathbf{m}^p$, downsampled $\mathbf{m}$, $\mathbf{z}^t$, and $\mathbf{z}$ in the latent space, which are then fed into the 3D U-Net.

*This work was done when Ding Ding was an intern at MSRA.
†Corresponding author.

Table 1. Hyperparameters for the added layers in HomoGen, built upon the Stable Diffusion image inpainting model. Each element in "[∗, ∗, ...]" denotes the number of layers or blocks in the corresponding scale of the latent.

Models	Hyperparameter	Encoding Blocks	Middle Blocks	Decoding Blocks
3D VAE	Model channels	/	/	128
	Temporal convolution channel multipliers	/	/	[4, 4, 2, 1]
	Temporal convolution kernel size	/	/	3
	Temporal residual blocks	/	/	[3, 3, 3, 3]
	Temporal attention resolutions	/	/	[4]
	Temporal attention head channels	/	/	512
	Temporal attention blocks	/	/	[1, 0, 0, 0]
3D U-Net	Model channels	320	320	320
	Temporal convolution channel multipliers	[1, 2, 4, 4]	[4]	[4, 4, 2, 1]
	Temporal convolution kernel size	3	3	3
	Temporal residual blocks	[2, 2, 2, 2]	[2]	[3, 3, 3, 3]
	Attention resolutions	[1, 2, 4]	[4]	[4, 2, 1]
	Attention head channels	64	64	64
	Window size in window-attention (h, w, t)	(4, 7, 24)	(4, 7, 24)	(4, 7, 24)
	Temporal attention blocks	[2, 2, 2, 0]	[1]	[0, 3, 3, 3]
	Window-attention blocks	[2, 2, 2, 0]	[1]	[0, 3, 3, 3]
	GoP-wise CLIP latent channels	1280	1280	/
	Sampling step embedding channels	1280	1280	1280
	Prior projector channel multipliers	[1, 2, 4, 4]	[4]	/
	Prior projector convolution kernel size	(1, 1, 1)	(1, 1, 1)	/
LiteControlNet	Model channels	320	/	/
	3D convolution channel multipliers	[1, 2, 4, 4]	/	/
	3D convolution channel kernel size	(3, 3, 3)	/	/
	3D convolution layers	[1, 1, 1, 1]	/	/

Cross-attention. Cross-attention represents injecting $\mathbf{z}^p$ into the 3D U-Net via a cross-attention block positioned after each window-attention block in the 3D U-Net encoder.

Noise Prior. Noise prior represents that $\mathbf{z}^p$ is first scaled by a coefficient λ and then added into the noise applied to a GoP during both training and inference [11]. To determine an optimal λ, we test values from 0 to 0.1 in increments of 0.02, ultimately setting λ to 0.02 for performance reporting.

C. Training and Inference Details

C.1. Training Configurations

We train the 3D VAE and the 3D U-Net separately. For training the 3D VAE, we leverage pre-trained weights of the 2D VAE[1] from Stable Diffusion as spatial weights. During training, we fix the spatial pre-trained weights and optimize the added temporal weights. For training objectives, we utilize L2 loss as the reconstruction loss, *i.e.*, L_{rec}, and LPIPS loss as the perceptual loss, *i.e.*, L_{lpips}. Additionally, to improve the realism and temporal consistency of video reconstruction, we incorporate an adversarial loss, *i.e.*, L_{adv}, which is measured using a T-PatchGAN discriminator [1]. The weights for L_{rec}, L_{lpips}, and L_{adv} are set to 1, 0.1, and 0.01, respectively. The 3D VAE is trained on 4 NVIDIA Tesla A100 (80G) GPUs, using the AdamW optimizer [5] with a batch size of 8, setting the initial learning rate to 5×10^{-6} and running 300k iterations.

For training the 3D U-Net, we initialize spatial weights using pre-trained weights of the 2D U-Net[2] from Stable Diffusion v-2.0 image inpainting model. During training, we fix the spatial pre-trained weights and optimize the added weights. For the training objective, we utilize L2 loss as the latent reconstruction loss. The 3D U-Net is trained on 16 NVIDIA Tesla A100 (40G) GPUs, using the AdamW optimizer [5] with a batch size of 32, setting the initial learning rate to 2×10^{-5} and running 400k iterations.

[1] https://huggingface.co/stabilityai/sd-vae-ft-mse

[2] https://huggingface.co/stabilityai/stable-diffusion-2-inpainting

Table 2. Examples of the text prompts generated for the DAVIS test set by GPT-4o.

Video ID	Object Removal	Video Completion
bear	*A rocky enclosure with large stones and green foliage against a stone wall backdrop, with mulch covering the ground.*	*A brown bear walks through a rocky enclosure with a stone wall background, surrounded by foliage and wood chips.*
blackswan	*A lush green bank lines a calm river, reflecting the surrounding foliage and creating a serene natural scene.*	*A black swan gracefully glides through the water, its reflection mirroring the lush green foliage on the shore.*
boat	*A serene coastal scene with clear blue water and a hilly landscape in the background.*	*A white boat is speeding away from the rocky shore with white houses in the background, creating a foamy wake in the calm blue sea.*
car-roundabout	*A busy street corner in a city with parked cars, buildings, and a directional road sign surrounded by greenery.*	*A shiny black Mini Cooper speeds along a city street corner with historic buildings and a monument in the background.*
car-shadow	*A quiet urban intersection with a traffic light, under a clear sky.*	*An urban street corner with a silver car turning while a pedestrian crosses the crosswalk.*
dance-twirl	*A group of people, including children, are seated on hay bales and colorful stools, attentively watching an outdoor event.*	*A dancer in a blue costume gracefully performs before an audience seated on rows of benches, against a backdrop of hay bales and planters.*
dog	*A dry, patchy yard with scattered leaves, a wire fence in the background, and a lush green bush on the right side.*	*A dog speeds through an obstacle course, weaving between red and white poles on a green field.*
elephant	*A rocky outdoor enclosure with trees and a building in the background, surrounded by greenery and boulders.*	*An elephant walks through a sandy enclosure, kicking up dust with its massive feet, surrounded by rocks and trees.*
goat	*A rocky landscape with sparse vegetation and rugged terrain, viewed from an elevated vantage point.*	*A goat navigates a rocky mountainside, showing its agility and balance in the rugged terrain.*

We preprocess the original videos into GoPs as training samples. Specifically, we sample 24 frames at 24 FPS to form a GoP from each video in the training set. 60% of the GoPs are established following the method proposed in Sec. 3.4 of our main paper, while 40% of the GoPs are selected from segments of original videos, enabling learning without temporal references for inferring the first GoP of a video. Each frame is center-cropped and resized to 256×448. We divide each mask into an 8×8 grid and set all elements within the grids containing the corrupted regions to 1, which prevents information loss when the mask is downsampled by a factor of 8 before being fed into the 3D U-Net.

C.2. Inference Details

During inference, we employ the EulerEDM sampler [8] with 10 steps for conditional diffusion and classifier-free guidance of magnitude 7.5. All videos in the test sets are segmented into GoPs for sequential inference with the GoP size of 24 frames, as proposed in Sec. 3.4 of our main paper. $N^{overlap}$ is set to 3. Following widely adopted preprocessing strategies [4, 13, 16], we dilate the binary masks $\mathbf{m}$ where 1 indicates corrupted regions and 0 indicates originally visible regions, to avoid edge information leakage. We also divide each mask into an 8×8 grid and set all elements within the grids containing the corrupted regions to 1. For fair comparisons with previous work, test videos are sized with a spatial resolution of 240×432 and padded to 256×448 by replicating edge values.

To address the absence of textual prompts in the test sets, we employ GPT-4o [6] to generate the scene descriptions of ideally inpainted videos. GPT-4o ("o" for "omni") is a multi-modal model with exceptional capabilities in visual understanding, language comprehension, and conversational interaction. GPT-4o accepts any combination of text, audio, image, and video as input prompts, and generates any combination of text, audio, and image outputs. The text prompt generation process is akin to consulting a sagacious expert with remarkable insight and knowledge. Specifically, we brief GPT-4o about our objective to create scene descriptions tailored for video inpainting and clearly define our requirements. The prompt we used is as follows:

I am currently working on describing scenes from corrupted videos and require your help.

Table 3. Quantitative comparisons of HomeGen with LiteControlNet and ControlNet.

Model	PSNR	SSIM	VFID	Runtime
w/ ControlNet	34.8237	0.97369	0.03225	0.3353
w/ LiteControlNet	34.8109	0.97352	0.03230	0.2718

Table 4. Quantitative comparisons of PCVDM with and without mask $\mathbf{m}_i^p$ indicating the regions being propagated in HPP.

Model	PSNR	SSIM	VFID
w/ mask $\mathbf{m}^p$	34.43	0.9719	0.035
w/o mask $\mathbf{m}^p$	34.81	0.9735	0.032

> *For each video, I shall provide its central frame. Based on the central frame, compose a description of the scene and disregard the gray areas representing the corrupted regions covered by masks. Focus solely on the scene, objects, and movements. No more than 40 words.*

Subsequently, we provide GPT-4o with the center frame of each video, enabling it to generate the required descriptions. Examples of the text prompts generated for the DAVIS test set are presented in Tab. 2.

D. More Ablations

We conduct further ablation studies on the technologies employed in HomoGen, utilizing the YouTube-VOS dataset for evaluation.

D.1. Study of LiteControlNet

As described in Sec. 3.3.2, we propose LiteControlNet, which features a lightweight structure compared to ControlNet. Here, we evaluate HomoGen with LiteControlNet or ControlNet, and the results in Tab. 3 show that LiteControlNet achieves an 18.94% runtime reduction with minor accuracy decreases. Runtimes (s/frame) are measured on an NVIDIA Tesla A100 (40G) GPU.

D.2. Study of Excluding Masks of HPP Results.

As described in Sec. 3.2, mask $\mathbf{m}^p$ indicating the regions being propagated are derived from HPP. However, we suggest excluding $\mathbf{m}^p$ from PCVDM. As propagated regions may carry distortions and are not always valuable, PCVDM needs to learn to distinguish informative propagated content from noise instead of directly using a mask to indicate valuable regions. We compare HomoGen with and without $\mathbf{m}^p$, and the evaluation results in Tab. 4 show that excluding $\mathbf{m}^p$ leads to non-trivial performance improvements.

E. Practical Object Removal

To enhance HomoGen's practicality in real-world applications, we introduce a practical object removal pipeline that enables users to select and remove specific objects from videos. Similar to previous designs [15, 16], our object removal pipeline is divided into two stages: user-friendly annotation and sequential inpainting, as illustrated in Fig. 1. In the user-friendly annotation stage, users interactively create masks to mark the objects they wish to remove. On the initial frame of each video, object regions are identified using Segment-Anything Model (SAM) [3] based on user prompts, *e.g.*, points, bounding boxes, rough masks, *etc*. The segmentation is then extended across the entire video using XMem [2] to ensure comprehensive object coverage. For sequential inpainting, we employ the proposed solution in Sec. 3.4 of our main paper for inpainting long videos. Specifically, we first segment the entire video into GoPs with an overlap of $N^{overlap}$ that provides temporal references and maintains temporal coherence. During processing each GoP, the text prompts may either be provided by the user or generated by GPT-4o using the center frame of the GoP as detailed in Sec. C.2. Using the annotated masks, text prompts, and the corresponding GoP, HomoGen removes the specified objects, leaving behind clean and coherent backgrounds.

F. More Qualitative Results

We present additional visual comparisons of HomoGen with ProPainter [16]. Fig. 2 and Fig. 3 present the comparisons of video completion performance on the YouTube-VOS test set [12]. Fig. 4 and Fig. 5 present the comparisons of video completion and object removal performance on the DAVIS test set [7]. Fig. 6 and Fig. 7 present the comparisons of object removal performance on the RORD test set [10]. As shown in Fig. 2 - 7, HomoGen can generate more realistic and coherent content in the corrupted regions.

References

[1] 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. 2

[2] Ho Kei Cheng and Alexander G Schwing. Xmem: Long-term video object segmentation with an atkinson-shiffrin memory model. In *ECCV*, 2022. 4

[3] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. Segment anything. In *ICCV*, 2023. 4

[4] 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. 3

[5] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *ICLR*, 2019. 2

[6] OpenAI. Gpt-4o contributions. https://openai.com/gpt-4o-contributions, 2024. 3

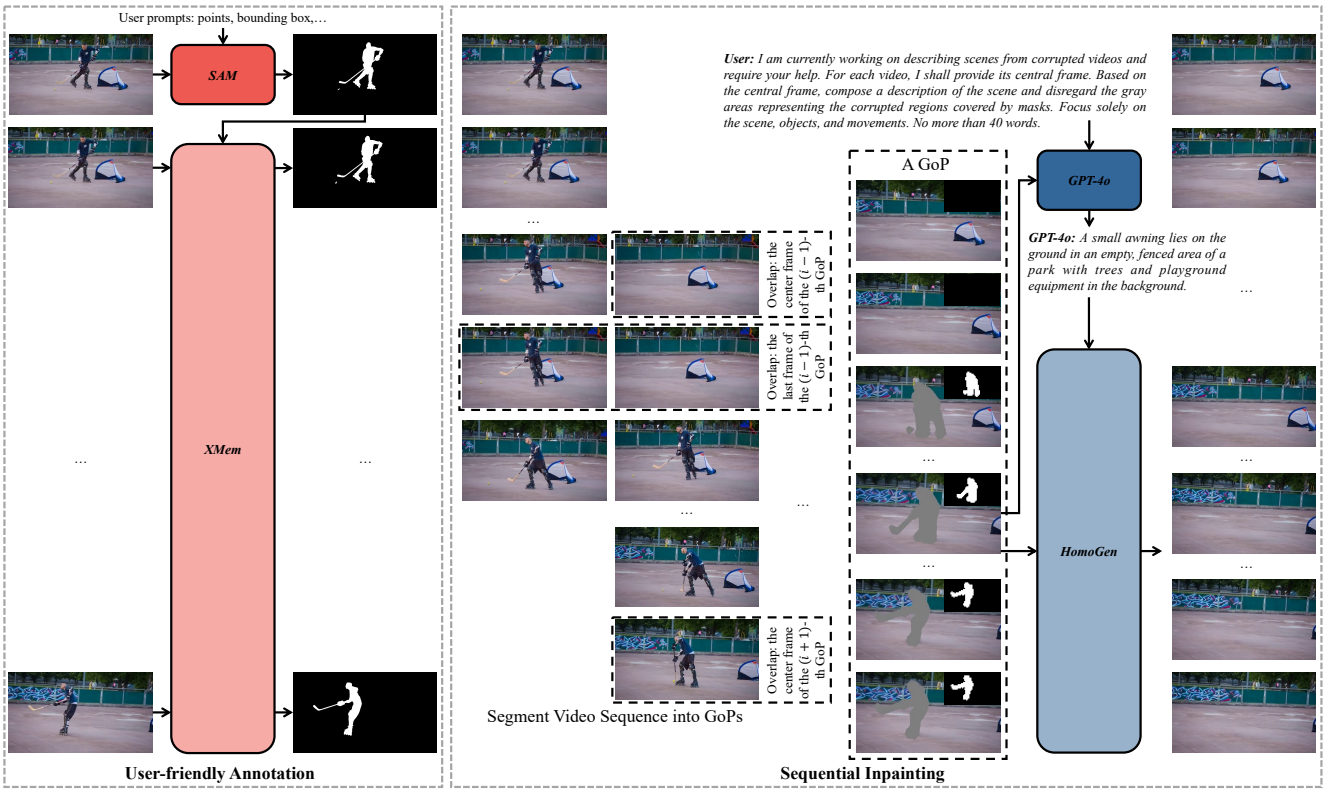

Figure 1. Illustration of the pipeline for practical object removal.

[7] 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. 4

[8] Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. SDXL: improving latent diffusion models for high-resolution image synthesis. In *ICLR*, 2024. 3

[9] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *CVPR*, 2022. 1

[10] Min-Cheol Sagong, Yoon-Jae Yeo, Seung-Won Jung, and Sung-Jea Ko. Rord: A real-world object removal dataset. In *BMVC*, 2022. 4

[11] Yanhui Wang, Jianmin Bao, Wenming Weng, Ruoyu Feng, Dacheng Yin, Tao Yang, Jingxu Zhang, Qi Dai, Zhiyuan Zhao, Chunyu Wang, et al. Microcinema: A divide-and-conquer approach for text-to-video generation. In *CVPR*, 2024. 2

[12] 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. 4

[13] Kaidong Zhang, Jingjing Fu, and Dong Liu. Flow-guided transformer for video inpainting. In *ECCV*, 2022. 3

[14] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. In *ICCV*, 2023. 1

[15] Zhixing Zhang, Bichen Wu, Xiaoyan Wang, Yaqiao Luo, Luxin Zhang, Yinan Zhao, Peter Vajda, Dimitris Metaxas, and Licheng Yu. Avid: Any-length video inpainting with diffusion model. In *CVPR*, 2024. 4

[16] Shangchen Zhou, Chongyi Li, Kelvin CK Chan, and Chen Change Loy. Propainter: Improving propagation and transformer for video inpainting. In *ICCV*, 2023. 3, 4

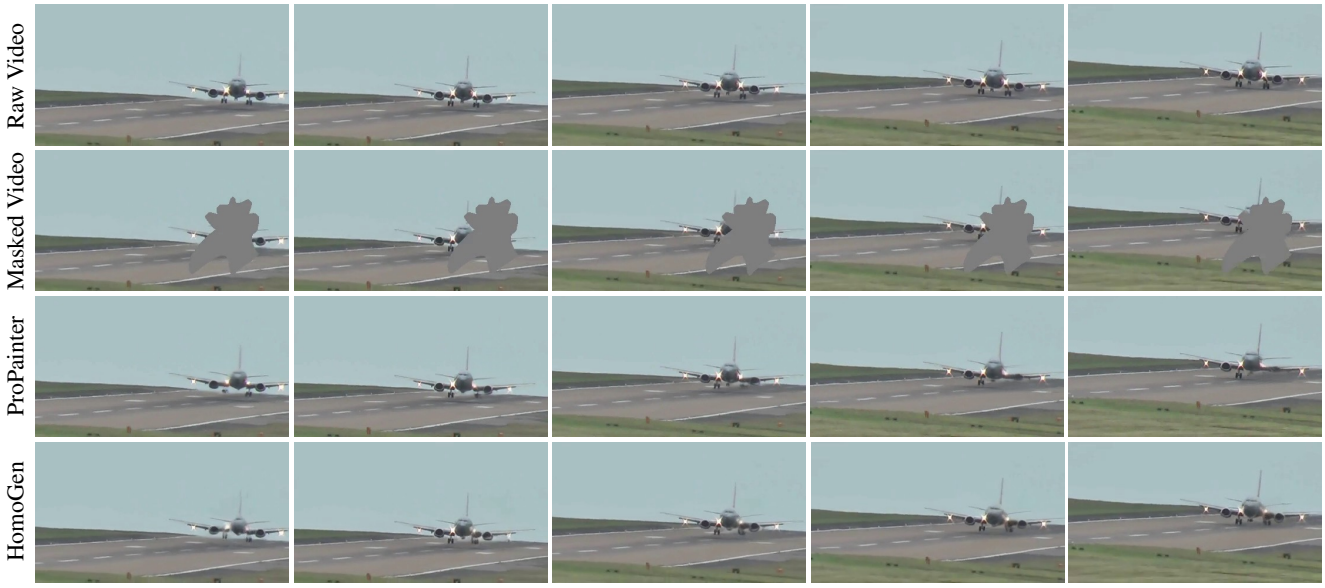

Figure 2. Qualitative video completion results on the YouTube-VOS test set. The ID of the showcased video is “4e1ef26a1e”.

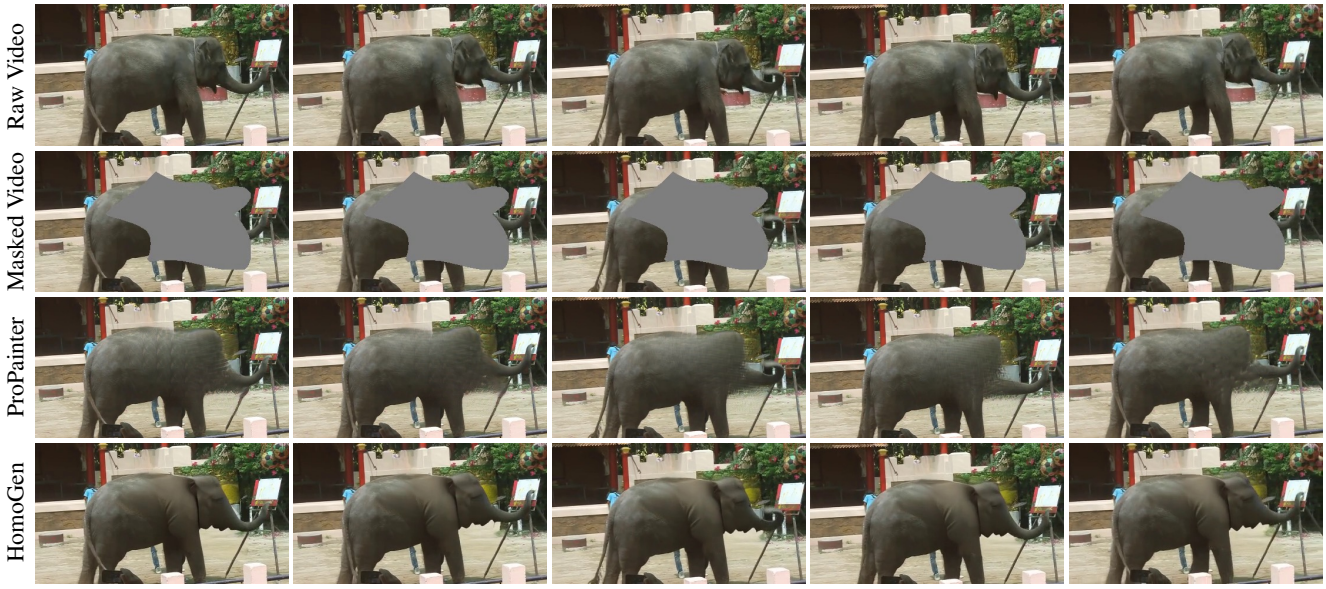

Figure 3. Qualitative video completion results on the YouTube-VOS test set. The ID of the showcased video is “cfd1e8166f”.

Figure 4. Qualitative object removal results on the DAVIS test set. The ID of the showcased video is “elephant”.

Figure 5. Qualitative video completion results on the DAVIS test set. The ID of the showcased video is “stroller”.

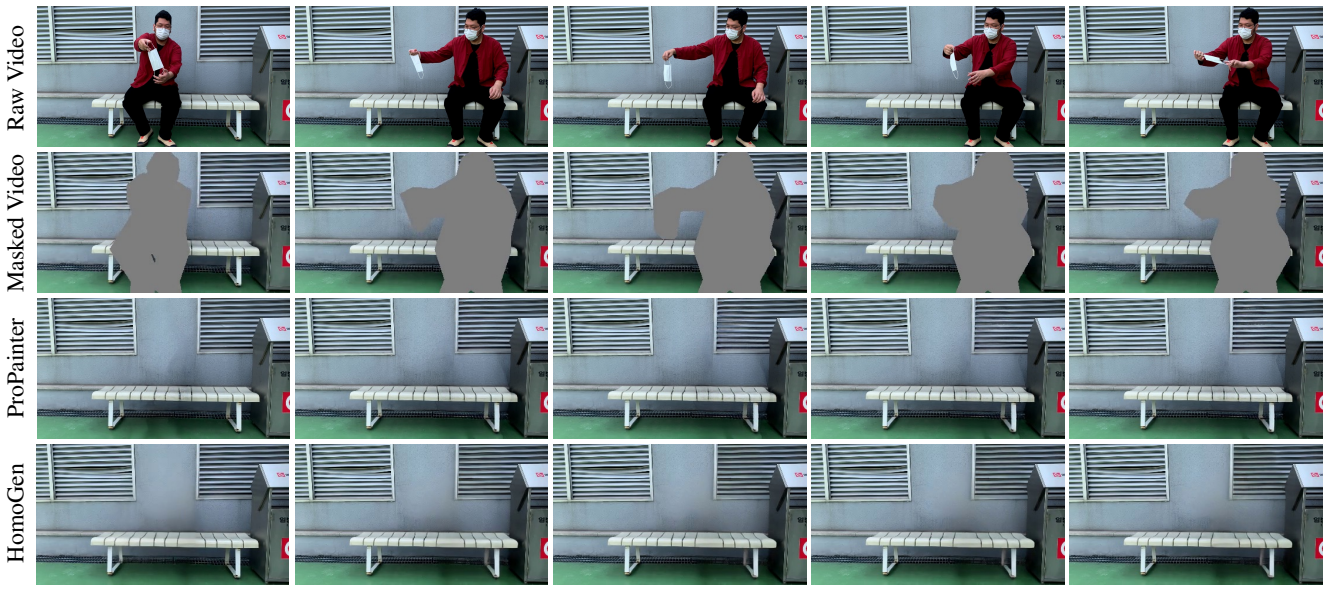

Figure 6. Qualitative object removal results on the RORD test set. The ID of the showcased video is "I-210910_O12054_W04".

Figure 7. Qualitative object removal results on the RORD test set. The ID of the showcased video is "I-211003_I09032_T04".