Generative Omnimatte: Learning to Decompose Video into Layers Supplementary Material

Yao-Chih Lee[1,2,*] Erika Lu[1] Sarah Rumbley[1] Michal Geyer[1,3]
Jia-Bin Huang[2] Tali Dekel[1,3] Forrester Cole[1]

[1]Google DeepMind [2]University of Maryland College Park [3]Weizmann Institute of Science

https://gen-omnimatte.github.io

Appendix

We provide additional discussions (Sec. A), as well as further details on the training process of our object-effect-removal model (Sec. B), the inference pipeline for object and effect removal (Sec. C), the optimization details for omnimatte reconstruction (Sec. D), runtime analysis (Sec. E), the quantitative comparisons (Sec. F), the quantitative ablation study (Sec. G). The video comparisons, training video examples, and self-attention visualization are provided in the project page.

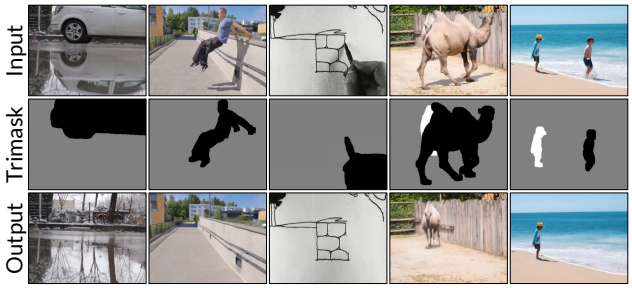

Figure 1. **Object-effect-removal results of CogVideoX-based Casper.**

A. Additional Discussions

Reproducibility. We will release our dataset for reproducibility. In addition, we finetuned a publicly available CogVideoX [56] using the same data to create a DiT version of Casper*, without adjusting the original CogVideoX hyperparameters. The results are shown in Fig. 1. The sampling process takes 66 sec for an 85-frame, 384×672 video and a trimask with 50 DDIM steps w/o CFG on an A100 GPU.

Handling occluded foreground. Our Casper model can also handle certain occlusion scenarios where the foreground object is partially obscured by background content (Fig. 2). We can treat the occluding background content as additional foreground objects and remove them to reveal the fully visible target object. Subsequently, the omnimatte optimization process utilizes the completed solo video and the clean background without occlusions as inputs to generate a complete omnimatte layer.

Impact of text prompts. We used simple, short prompts such as "a clean beach" for fine-tuning the Lumiere-based Casper. While Casper is primarily driven by the input video and trimask and less sensitive to text prompts ("empty scene" works well for both rows below), we did observe that a highly unrelated prompt (*e.g.*, "rock concert") could affect the performance, as shown in Fig. 3.

Challenges in handling similar objects. While our Kubric dataset includes multiple similar objects to help Casper handle such cases, it may still struggle with complex scenarios. We observed a similar issue in CogVideoX-based Casper, potentially due to the domain discrepancies between Kubric data and real-world videos. This could be mitigated in future work through more realistic data generation and re-introducing successful removal results into the training set.

Undesired detail changes. These artifacts are caused by Lumiere's SSR model, which can hallucinate high-frequency details. While we use a post-processing step to transfer original details, it is applied conservatively to avoid altering effect-removal areas. A future direction may be exploring an SSR model that can refer to the original video when upsampling the removal result.

*For CogVideoX-based Casper finetuning, we adopt the codes and model from the third-party github: https://github.com/aigc-apps/CogVideoX-Fun.git

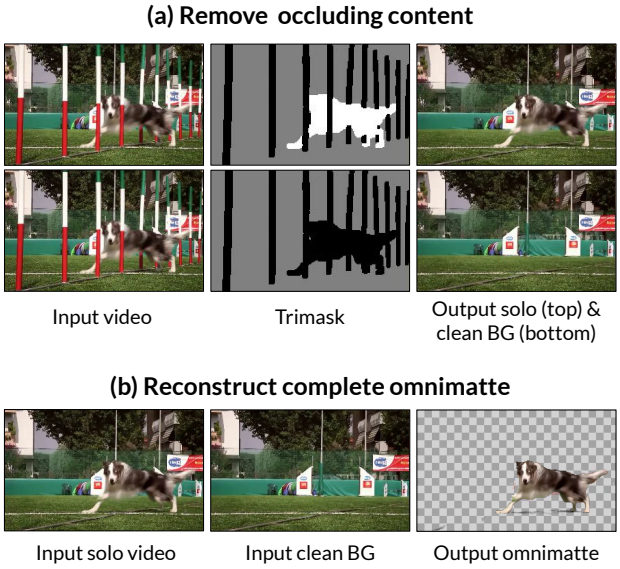

Figure 2. **Handle occluding background content.** Our method can also handle scenarios where the foreground object is occluded by background content (*e.g.*, poles). By treating the occluding poles as additional foreground objects and removing them, we obtain a complete solo video of the dog and a clean-plate background video. These two videos can then be used to reconstruct the complete omnimatte of the dog.

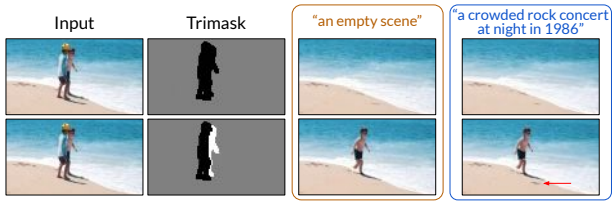

Figure 3. **Impact of the text prompt.**

B. Training Details of Our Casper Model

Omnimatte data. For the *Omnimatte* training data, we observed that the background video reconstructions produced by existing omnimatte methods [5, 7, 10] can lack sharpness and alignment with the original input video. This can lead to degradation of the original details in the preservation regions of our removal model, which is trained on pairs of original inputs and blurry background videos. To address this issue, we use the video reconstruction results of omnimatte methods as training inputs instead of the original videos. Since the video reconstruction is composed of omnimatte layers, it maintains the same quality as the target background video, preventing the model from learning quality degradation.

Kubric data generation. To synthesize our *Kubric* data, we randomly generate 1 to 6 objects in a scene. We also introduce challenging scenarios such as stationary objects, varying lighting conditions, and reflective surfaces. The Kubric generation script will be publicly released.

Object-Paste data. Due to the challenge of real data collection, Object-Paste is synthesized to enhance inpainting and background preservation, but not effect removal. Though not always reflected in metrics, we observed qualitative improvements in the main paper Fig. 9a and Supp. HTML. A tradeoff exists between background preservation and effect removal, and a high Object-Paste weight may hinder effect-removal performance. To balance this, we set its weight to 2% in the training set.

Trimask label for synthetic data. For both synthetic *Kubric* and *Object-Paste* data, we randomly switch the labels from white to gray, to encourage the model to learn preservation and inpainting capabilities in gray-labeled background areas.

Data augmentation. Video examples are randomly clipped into 80-frame segments and cropped to a 128×128px resolution for training. For real videos shorter than 80 frames, we apply temporal reflective padding to achieve the desired length. We augment the four different categories of data with different ratios during the training. Real-world examples from *Omnimatte* and *Tripod* constitute 50% of the training data, while synthesized *Kubric* and *Object-Paste* data account for approximately 48% and 2%, respectively.

We fully finetune our Casper model from the pretrained Lumiere inpainting model [1, 8] for 20,000 iterations and a batch size of 32.

C. Inference Details of Object and Effect Removal

Our Casper model takes 256 DDPM [2] sampling steps without using classifier free guidance. It is important to note that we do not cherry pick random seeds for object and effect removal outputs. We consistently apply a fixed random seed (= 0) to all input videos.

D. Omnimatte Optimization Details

The Casper model has a resolution of 128px (*e.g.*, 224×128) inherited from the Lumiere base stage. While Lumiere's SSR stage [1] upsamples the removal videos (*e.g.*, $\mathcal{I}_{\text{bg}}$) to higher resolutions, (*e.g.*, $\mathcal{I}_{\text{bg}}^{\text{SSR}}$), it may hallucinate high-frequency detail in unpredictable ways. Thus, directly reconstructing the omnimattes from the upsampled videos may result in noisy foreground layers, capturing unwanted background artifacts.

To mitigate this issue, we employ a bootstrapping strategy. Initially, we optimize the omnimatte with the base model outputs $(\mathcal{I}_i, \mathcal{I}_{\text{bg}})$ to obtain the alpha maps at the base 128px resolution. We then use a bilinear upsampling of the 128px alpha map to supervise the optimization of a higher-resolution omnimatte (*e.g.*, 640×384) from the upsampled pair $(\mathcal{I}_i^{\text{SSR}}, \mathcal{I}_{\text{bg}}^{\text{SSR}})$ for finer details. The optimizations of base and upsampled resolutions use the same algorithm (Sec. 3.5 of main paper) but with slightly different hyper-parameter settings.

Base optimization We optimize the base resolution omnimattes using the following loss function: $\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{recon}} + \lambda_{\text{sparsity}}\mathcal{L}_{\text{sparsity}} + \lambda_{\text{mask}}\mathcal{L}_{\text{mask}}$ to optimize the base resolution omnimattes. The balancing weight $\lambda_{\text{sparsity}}$ is set to 1, along with the constant weights β_0 and β_1 for the sparsity loss (Eq. 4 of the main paper) are empirically set to 1 and 10, respectively. The weight of mask supervision λ_{mask} is initialized to 20 and gradually reduced over the optimization. The optimization takes 20,000 iterations with a batch size of 20 frames at a 128px base resolution of 128px.

Upsampling optimization To bootstrap higher resolution of omnimattes (*i.e.*, foreground RGB $\mathcal{I}_{i,\text{fg}}^{\text{hr}}$ and alpha α_i^{hr}), we employ the solo video and background video of the SSR outputs, $(\mathcal{I}_i^{\text{SSR}}, \mathcal{I}_{\text{bg}}^{\text{SSR}})$ as the input pair for our optimization framework. The foreground RGB variables are initialized using the upsampled base-resolution optimized RGB $\mathcal{I}_{i,\text{fg}}$, and an additional alpha supervision loss, $\mathcal{L}_{\text{alpha}} = \|\alpha_i^{\text{hr}} - \alpha_i^{\text{up}}\|_2$, is introduced, supervised by the upsampled base-resolution alpha maps, α_i^{up}. To prevent the model from learning aliased boundaries, we disable supervision loss on the edge regions of the alpha maps. Moreover, we switch the photometric reconstruction loss, $\mathcal{L}_{\text{recon}}$ (Eq. 3 of the main paper), from L2 loss to L1 loss to mitigate the impact of outlier hallucinated high-frequency details produced by the SSR model. The loss weights $\lambda_{\text{sparsity}}$ and λ_{mask} are both initialized to 10, while the mask supervision loss $\mathcal{L}_{\text{mask}}$ is deactivated after the first 2,000 iterations. The weight of the base-resolution alpha supervision loss λ_{alpha} is set to 20. The optimization process runs for 20,000 iterations to obtain the final omnimattes.

Input video reconstruction and Detail Transfer To reconstruct the original input video from individual optimized omnimatte layers, depth information is required to determine the correct layer order in multi-object scenarios. We utilize DepthCrafter [3] to estimate video depth and define the frame-level depth order for foreground layers. Subsequently, all layers, including the clean background, are composited from back to front using the over operation [9].

During the compositing process, we compute the composited opacity for each layer. For layer pixels where the composited opacity reaches 1 (*i.e.*, fully opaque), a detail transfer step [6, 7, 10] is applied. This step copies the original details from the input video to the high-resolution omnimatte and background layers, mitigating misaligned high-frequency details that may have been hallucinated by the SSR model.

E. Runtime

For Stage 1, Casper takes 12 min to process an 80-frame video and 15-20 min for longer videos, the same as the original Lumiere-Inpainting. We run Casper on a 96GB TPU with a batch size of 4 (*e.g.*, three solo videos and a clean background). Additional objects can be run in parallel with multiple TPUs. For Lumiere SSR upsampling, it takes around 15 minutes. For Stage 2, our unoptimized code takes 7 min to produce each object layer on a 48GB TPU. Each object layer is computed independently and thus can be parallelized. Once obtaining all layers, the post-processing detail transfer takes 1 minute. For an 80-frame video of 3 objects on a single TPU, the entire process takes $12 + 15 + 7 \times 3 + 1 = 49$ min (or $12 + 15 + 7 + 1 = 35$ min with multi TPUs). In contrast, OmnimatteRF takes 3 hr to optimize a NeRF and all layers together, potentially limiting the number of objects due to GPU memory constraints.

F. Onimatte-RF Synthetic Evaluation Benchmark

Table 1 presents the per-scene evaluation scores. Omnimatte3D [10] fails to reconstruct the background layer in two scenes, resulting in all-zero outputs. OmnimatteRF [5] employs an additional background retraining step to enhance effect removal and inpainting accuracy in certain cases by leveraging a global background scene model. Our method performs the overall best in both PSNR and LPIPS [12] metrics.

G. Ablation study

Training data. Table 2 presents a quantitative ablation study on the 10 synthetic evaluation scenes from OmnimatteRF [5]. By incrementally adding four distinct data categories to the training set, the removal model achieves improved effect removal performance. For further video comparisons on real-world videos, please refer to our supplementary HTML.

Input condition. Table 3 illustrates the quantitative performance of removal models trained with various input condition settings. The original inpainting condition masks out the RGB values within the removal regions and concatenates them with a binary mask. Following ObjectDrop [11], we unmask the RGB values in the removal regions to enable the model to associate effects with content. Finally, we introduce our proposed trimask to replace the binary mask condition, alleviating ambiguity in effect removal within

Table 1. **Quantitative comparison.** Following the benchmark established in OmnimatteRF [5], we evaluate the effect-removal quality on background videos of 10 synthetic scenes. Our method achieves the best overall scores in both PSNR and LPIPS metrics. We adopt the numbers of [4, 5, 7] reported in OmnimatteRF [5]. Best results are highlighted in red and second-best in yellow. Results marked as "-" indicate failures (*e.g.*, all zeros in Omnimatte3D [10]).

Scene	Movie-Donkey		Movie-Dog		Movie-Chicken		Movie-Rooster		Movie-Dodge		Kubric-Car		Kubric-Cars		Kubric-Bag		Kubric-Chair		Kubric-Pillow		Average	
Metric	PSNR↑	LPIPS↓	PSNR↑	LPIPS↓	PSNR↑	LPIPS↓	PSNR↑	LPIPS↓	PSNR↑	LPIPS↓	PSNR↑	LPIPS↓	PSNR↑	LPIPS↓	PSNR↑	LPIPS↓	PSNR↑	LPIPS↓	PSNR↑	LPIPS↓	PSNR↑	LPIPS↓
ObjectDrop [11]	27.23	0.091	28.84	0.129	27.87	0.153	26.68	0.145	29.64	0.102	33.92	0.087	34.31	0.101	32.13	0.051	34.95	0.081	35.80	0.096	31.14	0.104
Propainter [13]	27.09	0.133	27.67	0.109	26.82	0.119	24.31	0.143	31.31	0.065	32.59	0.075	34.54	0.081	32.52	0.046	34.99	0.047	38.72	0.030	31.06	0.085
Lumiere inpainting [1]	25.31	0.157	26.97	0.159	26.60	0.162	24.39	0.163	29.82	0.101	30.05	0.201	31.04	0.202	29.32	0.165	32.11	0.143	34.77	0.075	29.04	0.153
Ominmatte [7]	19.11	0.315	21.74	0.279	20.95	0.312	23.14	0.220	23.88	0.067	31.14	0.162	31.20	0.157	23.64	0.271	26.91	0.175	21.17	0.270	24.29	0.223
LNA [4]	18.79	0.104	26.08	0.154	19.22	0.190	26.46	0.131	24.94	0.068	-	-	-	-	27.08	0.138	21.21	0.105	31.66	0.080	-	-
Omnimatte3D [10]	24.72	0.234	23.15	0.372	24.17	0.266	23.98	0.372	-	-	34.61	0.142	36.48	0.126	33.94	0.135	-	-	37.01	0.119	-	-
OmnimatteRF [5]	38.24	0.005	31.44	0.030	32.86	0.021	27.65	0.024	39.11	0.006	39.09	0.033	39.78	0.032	39.58	0.029	42.46	0.023	43.62	0.022	37.38	0.023
Ours	32.02	0.017	33.33	0.033	32.59	0.037	29.31	0.047	36.20	0.014	42.78	0.011	44.41	0.016	42.96	0.007	43.94	0.011	46.25	0.006	38.38	0.020

Table 2. **Ablation study on our training data.** To assess the individual contribution of each data category, we conduct an ablation study by incrementally adding each category to the training set. We encourage the readers to view our HTML file for visual comparisons on in-the-wild videos.

Training data category				Metric	
Omnimatte	Tripod	Kubric	Object-Paste	PSNR↑	LPIPS↓
✓	✗	✗	✗	37.06	0.027
✓	✓	✗	✗	36.97	0.026
✓	✓	✓	✗	38.36	0.020
✓	✓	✓	✓	38.38	0.020

Table 3. **Ablation study on the input conditions for the Casper model.** The original inpainting condition utilizes a binary mask, while the video condition involves zeroing out the removal region. Following ObjectDrop [11], the content within the removal regions is preserved in the condition to enable the model to associate effects outside the mask with the content inside. Finally, we replace the binary mask condition with our proposed trimask to mitigate ambiguity in effect removal within preservation regions. The full impact of these input conditions may not be evident from the evaluation of 10 synthetic background videos. We therefore encourage readers to examine comparisons on real-world videos in our supplementary HTML.

	RGB video condition	Mask condition	PSNR↑	LPIPS↓
Original inpainting	Masking removal area	Binary	38.58	0.021
ObjectDrop approach [11]	No masking	Binary	38.24	0.020
Our condition	No masking	Our trimask	38.38	0.020

preservation regions. While the effectiveness of these input conditions may not be readily apparent from the evaluation of 10 synthetic background videos, we encourage readers to examine comparisons on real-world videos in our supplementary HTML.

References

[1] Omer Bar-Tal, Hila Chefer, Omer Tov, Charles Herrmann, Roni Paiss, Shiran Zada, Ariel Ephrat, Junhwa Hur, Guanghui Liu, Amit Raj, et al. Lumiere: A space-time diffusion model for video generation. In *SIGGRAPH Asia 2024*, 2024. 2, 4

[2] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *NeurIPS*, 2020. 2

[3] Wenbo Hu, Xiangjun Gao, Xiaoyu Li, Sijie Zhao, Xiaodong Cun, Yong Zhang, Long Quan, and Ying Shan. DepthCrafter: Generating consistent long depth sequences for open-world videos. *arXiv preprint arXiv:2409.02095*, 2024. 3

[4] Yoni Kasten, Dolev Ofri, Oliver Wang, and Tali Dekel. Layered neural atlases for consistent video editing. In *SIGGRAPH Asia*, 2021. 4

[5] Geng Lin, Chen Gao, Jia-Bin Huang, Changil Kim, Yipeng Wang, Matthias Zwicker, and Ayush Saraf. OmnimatteRF: Robust omnimatte with 3d background modeling. In *ICCV*, 2023. 2, 3, 4

[6] Erika Lu, Forrester Cole, Tali Dekel, Weidi Xie, Andrew Zisserman, David Salesin, William T Freeman, and Michael Rubinstein. Layered neural rendering for retiming people in video. In *SIGGRAPH Asia*, 2020. 3

[7] Erika Lu, Forrester Cole, Tali Dekel, Andrew Zisserman, William T Freeman, and Michael Rubinstein. Omnimatte: Associating objects and their effects in video. In *CVPR*, 2021. 2, 3, 4

[8] Jingwei Ma, Erika Lu, Roni Paiss, Shiran Zada, Aleksander Holynski, Tali Dekel, Brian Curless, Michael Rubinstein, and Forrester Cole. VidPanos: Generative panoramic videos from casual panning videos. In *SIGGRAPH Asia 2024*, 2024. 2

[9] Thomas Porter and Tom Duff. Compositing digital images. In *Proceedings of the 11th annual conference on Computer graphics and interactive techniques*, 1984. 3

[10] Mohammed Suhail, Erika Lu, Zhengqi Li, Noah Snavely, Leonid Sigal, and Forrester Cole. Omnimatte3D: Associating objects and their effects in unconstrained monocular video. In *CVPR*, 2023. 2, 3, 4

[11] Daniel Winter, Matan Cohen, Shlomi Fruchter, Yael Pritch, Alex Rav-Acha, and Yedid Hoshen. ObjectDrop: Bootstrapping counterfactuals for photorealistic object removal and insertion. In *ECCV*, 2024. 3, 4

[12] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018. 3

[13] Shangchen Zhou, Chongyi Li, Kelvin C.K Chan, and Chen Change Loy. ProPainter: Improving propagation and transformer for video inpainting. In *ICCV*, 2023. 4