VideoHandles: Editing 3D Object Compositions in Videos Using Video Generative Priors

Juil Koo[1*] Paul Guerrero[2] Chun-Hao P. Huang[2] Duygu Ceylan[2] Minhyuk Sung[1]
[1]KAIST [2]Adobe Research

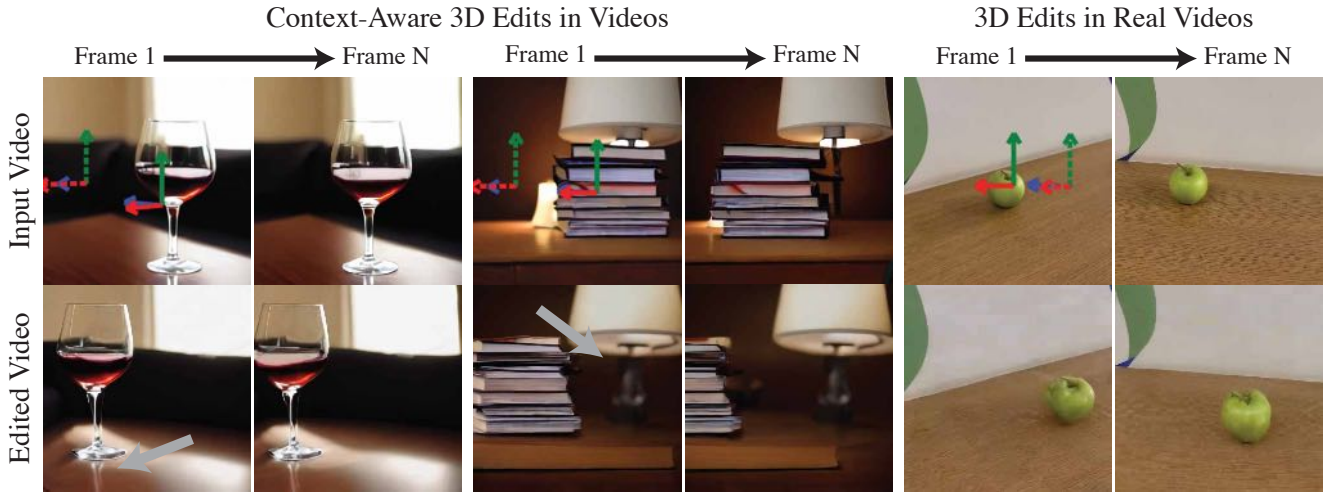

Figure 1. ***VideoHandles*** **edits 3D object composition in videos of static scenes.** Solid axes represent the original 3D position and dotted axes the user-provided target position. The edit plausibly updates effects like the reflection of the wine glass and handles disocclusions like the lamp behind the book pile that is exposed by the edit. In addition to generated videos, we can also edit real (non-generated) videos by inverting the video into its corresponding latent, as shown on the right.

Abstract

Generative methods for image and video editing leverage generative models as priors to perform edits despite incomplete information, such as changing the composition of 3D objects depicted in a single image. Recent methods have shown promising composition editing results for images. However, in the video domain, editing methods have focused on editing objects' appearance and motion, or camera motion. As a result, methods for editing object composition in videos remain largely unexplored. We propose VideoHandles as a method for editing 3D object compositions in videos of static scenes with camera motion. Our approach enables the editing of an object's 3D position across all frames of a video in a temporally consistent manner. This is achieved by lifting intermediate features of a generative model to a 3D reconstruction that is shared between all frames, editing the reconstruction, and projecting the features on the edited reconstruction back to each frame. To the best of our knowledge, this is the first generative approach to edit object compositions in videos. Our approach is simple and training-free, while outperforming state-of-the-art image editing baselines.

1. Introduction

We propose *VideoHandles* as a generative approach to edit the object composition in a video of a static scene. Our approach enables the editing of an object's 3D position in a video, resulting in a plausible, temporally consistent edit that preserves the identity of the original object. To the best of our knowledge, ours is the first generative approach that allows editing the object composition in a video. Given a pretrained flow-based video generative model, we present a novel method to edit the intermediate features from the generative model's network in a temporally consistent manner. Specifically, we lift the intermediate features of each frame to a common 3D reconstruction, effectively treating them

[*]Work done during an internship at Adobe Research.

as latent textures. We then edit the 3D location of an object using 3D translations or rotations, and project the features back to their corresponding frames. We use such projected features as guidance during the generative process to create a plausible edited video. Our approach is simple and does not require any training or finetuning that risks biasing the distribution of the generative model.

We evaluate our method on several generated and captured videos. As there are no existing methods that are specialized to editing the 3D object composition in videos, we compare to several image editing baselines that can be applied in a per-frame manner. We evaluate the results in terms of plausibility, temporal consistency, identity preservation, and adherence to the target edit. In addition to a large number of qualitative comparisons, we also conduct a user study. The results show a clear preference for our method in terms of plausibility and temporal consistency, while our method is at least on par with, or slightly better than image editing baselines in identity preservation and edit adherence. Finally, a quantitative evaluation further supports these findings.

2. VideoHandles: A 3D-Aware Video Editing Method

Consider a static input video $\boldsymbol{X}_{\text{src}} \in \mathbb{R}^{n\times h\times w\times 3}$, where objects remain stationary and only the camera moves. Our goal is to apply a 3D transformation to an object selected by the user in the first frame while preserving the identity of the input video, realism, and temporal consistency. See Figure 2 for an architecture overview.

To ensure that transformations in each frame of a video align with those in other frames, we define a 3D space in which a point cloud $\boldsymbol{P}_{\text{src}} = \{\mathbf{p}^{(j)}\}_{j=1}^{J}$ represents the 3D scene in the video with a shared coordinate system across all frames. A transformation is performed in this shared 3D space, denoted by $\mathcal{T} : \mathbb{R}^3 \to \mathbb{R}^3$, with each input frame $\mathbf{x}_{\text{src}}^{(i)}$ modeled as a 2D rendering of $\boldsymbol{P}_{\text{src}}$ from the i-th view. Specifically, we reconstruct $\boldsymbol{P}_{\text{src}}$ and estimate a camera pose for each frame from $\boldsymbol{X}_{\text{src}}$ using DUST3R [5]. By leveraging the reconstructed 3D scene from $\boldsymbol{X}_{\text{src}}$, we define a 3D-aware warping function in the 2D space of each frame.

However, directly warping pixel colors often produces unrealistic results due to inaccuracies in the reconstructed 3D scene and fails to account forcontextual effects like shadows or relightning. Inspired by DiffusionHandles [3], we instead warp the *features* of a pre-trained video generative model, using them as guidance during the generative process.

2.1. 3D-Aware Warping Function

We first describe how to obtain a 3D-aware warping function in the 2D space of each frame. Given a set of 2D coordinates $\Omega_{H,W} = \{(v,u) \mid v \in [0,H),\, u \in [0,W)\}$, the connection between the 3D space and the i-th 2D frame is established through the *projection function* $f^{(i)} : \mathbb{R}^3 \to \Omega_{H,W}$, which is defined by the i-th camera pose. Let $\mathcal{B}^{(1)} : \Omega_{H,W} \to \{0,1\}$ denote the 2D binary mask of an object selected by users in the first frame. Based on the 2D object mask in the first frame $\mathcal{B}^{(1)}$, we first partition $\boldsymbol{P}_{\text{src}}$, the point cloud reconstructed from the input video $\boldsymbol{X}_{\text{src}}$, as follows:

$$\boldsymbol{P}_f = \{\mathbf{p} \in \boldsymbol{P}_{\text{src}} \mid \mathcal{B}^{(1)}(f^{(1)}(\mathbf{p})) = 1\}, \tag{1}$$

$$\boldsymbol{P}_b = \boldsymbol{P}_{\text{src}} \setminus \boldsymbol{P}_f, \tag{2}$$

where $\boldsymbol{P}_f$ consists of points whose projections lie within the 2D masked region defined by $\mathcal{B}^{(1)}$, and $\boldsymbol{P}_b$ denotes the remaining points representing the background. By applying a 3D transformation $\mathcal{T}$ to $\boldsymbol{P}_f$ alone, we construct a rough target 3D scene represented as a point cloud:

$$\boldsymbol{P}_{\text{tgt}} = \mathcal{T}\boldsymbol{P}_f \cup \boldsymbol{P}_b. \tag{3}$$

The *lifting function* $g_{\text{src}}^{(i)} : \Omega_{H,W} \to \mathbb{R}^3$ takes a 2D coordinate $\mathbf{u} = (v,u)$ as input and returns the 3D point in $\boldsymbol{P}_{\text{src}}$ closest to the i-th camera from among the points projected close to $\mathbf{u}$:

$$g_{\text{src}}^{(i)}(\mathbf{u}) = \underset{\mathbf{p}\in\boldsymbol{P}_{\text{src},\mathbf{u}}^{(i)}}{\arg\min}\, z^{(i)}(\mathbf{p}), \tag{4}$$

where $\boldsymbol{P}_{\text{src},\mathbf{u}}^{(i)} = \{\mathbf{p} \in \boldsymbol{P}_{\text{src}} \mid \|f^{(i)}(\mathbf{p}) - \mathbf{u}\|_1 < \epsilon\}$ represents the set of 3D points that are projected close to $\mathbf{u}$ and $z^{(i)}(\mathbf{p})$ denotes the distance of point $\mathbf{p}$ from the i-th camera. Similarly, $g_{\text{tgt}}^{(i)}(\mathbf{u})$ returns the 3D point in $\boldsymbol{P}_{\text{tgt}}$ closest to the i-th camera from among the points projected close to $\mathbf{u}$. Using the functions $g_{\text{src}}^{(i)}$ and $g_{\text{tgt}}^{(i)}$, we define an occlusion-aware foreground point cloud $\boldsymbol{P}_f^{(i)} \subseteq \boldsymbol{P}_f$ for each frame as follows:

$$\boldsymbol{P}_f^{(i)} = \{g_{\text{src}}^{(i)}(\mathbf{u})\} \cap \{\mathcal{T}^{-1} g_{\text{tgt}}^{(i)}(\mathbf{u})\} \cap \boldsymbol{P}_f, \tag{5}$$

where $\mathbf{u} \in \Omega_{H\times W}$. It consists of foreground points that are not occluded by the background either before or after the transformation. Using this 3D information, we compute a 2D warping function $\mathcal{W}^{(i)} : \Omega_{H,W} \to \Omega_{H,W}$ as follows:

$$\mathcal{W}^{(i)}(\mathbf{u}) = \begin{cases} f^{(i)}\big(\mathcal{T}^{-1} g_{\text{tgt}}^{(i)}(\mathbf{u})\big), & \text{if } g_{\text{tgt}}^{(i)}(\mathbf{u}) \in \boldsymbol{P}_f^{(i)} \\ \mathbf{u}, & \text{otherwise.} \end{cases} \tag{6}$$

This warping function gives us the corresponding coordinate in the source image for any coordinate in the target image. All coordinates that do not project to the edited foreground point cloud remain unchanged. We denote warping a 2D signal $\mathcal{X} : \Omega_{H,W} \to \mathbb{R}^C$ as $(\mathcal{W}^{(i)} * \mathcal{X})(\mathbf{u}) :=$

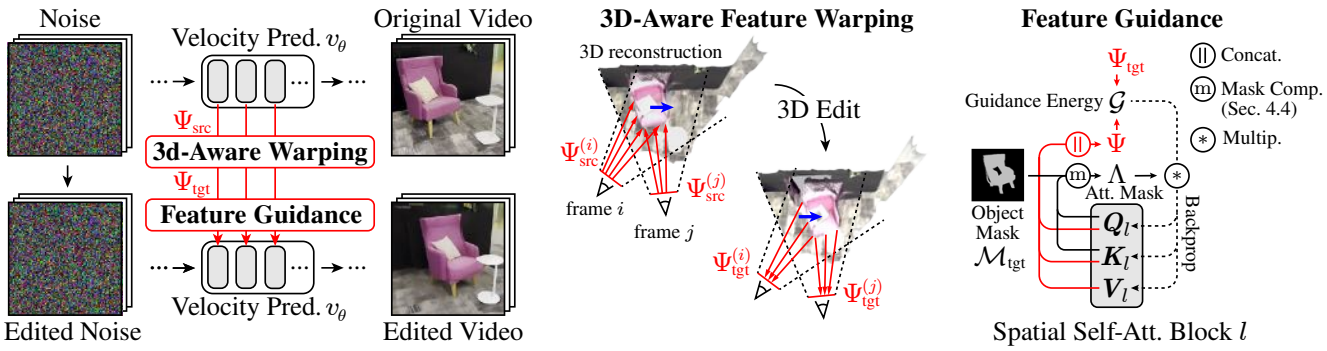

Figure 2. ***VideoHandles* Architecture.** We use the intermediate features Ψ_{src} of a video generative model to represent the identity of objects in a source video. Given a 3D transformation of an object, we can use a 3D reconstruction of the scene to warp the intermediate features consistently across frames. Guiding the video generator with these warped features Ψ_{tgt} gives us a an edited video where the object is transformed, while also maintaining the plausibility of effects like shadows and reflections.

$\mathcal{X}(\mathcal{W}^{(i)}(\mathbf{u}))$. Similarly, we denote its application to a tensor $\boldsymbol{X} \in \mathbb{R}^{\cdots \times H \times W \times \cdots}$ as $\mathcal{W}^{(i)} * \boldsymbol{X}$. Here H and W are the two spatial tensor dimensions that the warping is applied to and the ellipses denote arbitrary additional dimensions. The tensor is sampled at non-integer coordinates using linear interpolation.

2.2. Warping Video Features

The DiT architecture [4] of OpenSora, which we used for our main experiments, alternates layers that perform spatial self-attention, temporal self-attention, cross-attention to the prompt, and feed-forward computations.

Let $\boldsymbol{Q}_l(\boldsymbol{Z}_t), \boldsymbol{K}_l(\boldsymbol{Z}_t), \boldsymbol{V}_l(\boldsymbol{Z}_t) \in \mathbb{R}^{M \times H \times W \times d}$ be the query, key, and value features of the l-th self-attention layer extracted from $v^{\omega}_{\theta}(\boldsymbol{Z}_t, t, y)$, where M denotes the number of frames and d is the feature dimension. We use their concatenation from all layers as our extracted feature Ψ:

$$\Psi(\boldsymbol{Z}_t) = [\boldsymbol{Q}_l(\boldsymbol{Z}_t) \parallel \boldsymbol{K}_l(\boldsymbol{Z}_t) \parallel \boldsymbol{V}_l(\boldsymbol{Z}_t)]_{l=1}^{L}. \tag{7}$$

Let $\Psi^{(i)}(\boldsymbol{Z}_t) \in \mathbb{R}^{H \times W \times D}$ denote the feature for frame i, where D is the total dimensionality of the feature. Applying the previosuly defined warping function, given the latent of the input video $\boldsymbol{Z}^{\text{src}}_t$, its warped feature is defined as $\Psi^{(i)}_{\text{tgt}} := \mathcal{W}^{(i)} * \Psi^{(i)}(\boldsymbol{Z}^{\text{src}}_t)$.

2.3. Warping-Based Guided Generative Process

To guide the generation process of $\boldsymbol{Z}_t$ with $\Psi^{(i)}_{\text{tgt}}(\boldsymbol{Z}_t)$, we use an energy-guided generative process [1]. Given an energy function $\mathcal{G}(\boldsymbol{Z}_t)$, the gradient of $\mathcal{G}$ is injected at each step of the generative process, steering it towards minimizing the energy function:

$$\boldsymbol{Z}_{t+\Delta t} = \boldsymbol{Z}_t + \Delta t \cdot v^{\omega}_{\theta}(\boldsymbol{Z}_t, t, y) + \rho \nabla_{\boldsymbol{Z}_t} \mathcal{G}(\boldsymbol{Z}_t). \tag{8}$$

Below, we describe our specific design of $\mathcal{G}$ to edit object compositions in videos.

Object Transformation Energy. Let $\boldsymbol{M}^{(i)}_{\text{src}}, \boldsymbol{M}^{(i)}_{\text{tgt}} \in \mathbb{R}^{H \times W}$ denote the occlusion-aware 2D masks of the selected object before and after the transformation:

$$\boldsymbol{M}^{(i)}_{\text{src}}(\mathbf{u}) := \begin{cases} 1, & \text{if } \mathbf{u} \in \{f^{(i)}(\mathbf{p}) \mid \mathbf{p} \in \boldsymbol{P}^{(i)}_f\}, \\ 0, & \text{otherwise.} \end{cases}, \tag{9}$$

$$\boldsymbol{M}^{(i)}_{\text{tgt}} := \mathcal{W}^{(i)} * \boldsymbol{M}^{(i)}_{\text{src}}, \tag{10}$$

where $\mathbf{u} \in \Omega_{H \times W}$. To transform the selected object in the video, we define the *object transformation energy* $\mathcal{G}_o(\boldsymbol{Z}_t)$ as follows:

$$\sum_{i=1}^{M} \left\| \boldsymbol{M}^{(i)}_{\text{tgt}} \odot \left(\Psi^{(i)}_{\text{tgt}} - \Psi^{(i)}(\boldsymbol{Z}_t) \right) \right\|_2^2, \tag{11}$$

where $\odot$ is the element-wise product. This function measures the discrepancy between the current features $\Psi^{(i)}$ and the target features $\Psi^{(i)}_{\text{tgt}}$ within the region of the edited object $\boldsymbol{M}^{(i)}_{\text{tgt}}$.

Background Preservation Energy. To further preserve background details, we define an additional energy function called the *background preservation energy* $\mathcal{G}_b(\boldsymbol{Z}_t)$ as follows:

$$\left\| \psi_{\text{MHW}}\left(\boldsymbol{M}_b \odot \Psi_{\text{tgt}}\right) - \psi_{\text{MHW}}\left(\boldsymbol{M}_b \odot \Psi(\boldsymbol{Z}_t)\right) \right\|_2^2, \tag{12}$$

where ψ_{MHW} denotes the average over time and spatial dimensions, and $\boldsymbol{M}^{(i)}_b = \max((1 - \boldsymbol{M}^{(i)}_{\text{src}} - \boldsymbol{M}^{(i)}_{\text{tgt}}, 0)$ is the background mask. This function measures the discrepancy between the sums of the features in the background region. Unlike $\mathcal{G}_o$, $\mathcal{G}_b$ compares only the averages of the features, allowing the guidance of $\mathcal{G}_b$ to facilitate appropriate context changes according to the new object position, such as new shadows or reflections.

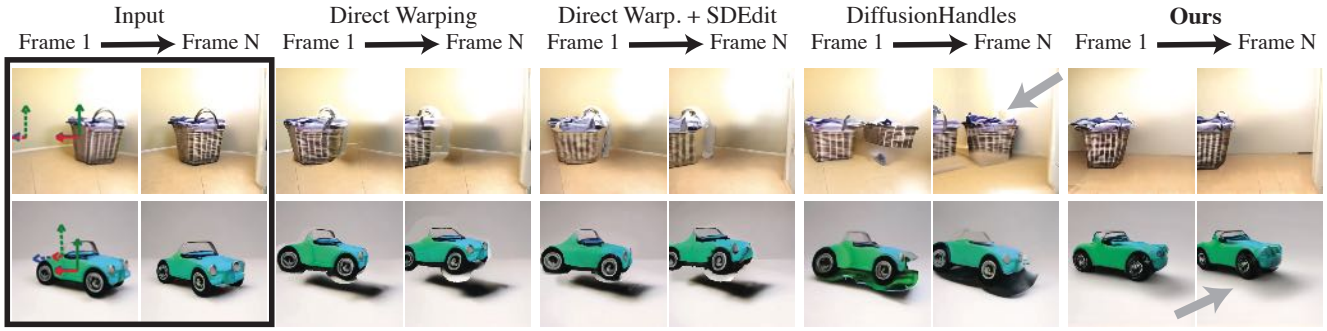

Figure 3. **A qualitative comparison with other baselines.** The examples show that ours best demonstrates plausibility by avoiding object duplication, adjusting shadows properly, and maintaining consistent outputs across frames, desipte warping errors, as illustrated in the direct frame warping outputs (column 2).

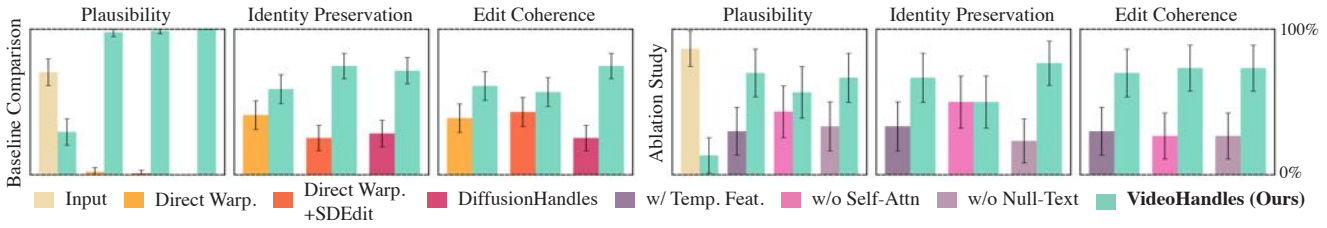

Figure 4. **User study results on the plausibility, identity preservation, and edit coherence of the edited videos.** Each bar pair shows user preferences, with the green bar for our method and the other for the baseline, along with 95% confidence intervals. We also include a comparison with the input video to represent the upper bound of plausibility.

3. Experiments

Experiment setups. For evaluation, we generate 27 input videos to be edited, each with a resolution of 320×320 and 51 frames. In the absence of prior work on modifying 3D object composition in videos, we compare our method to DiffusionHandles [3], the state-of-the-art method for composition editing in 2D images. We also compare against direct frame warping, which directly warps RGB values using the warping function. Additionally, we introduce an improved variant of the direct frame warping, whic refines the warped video using SDEdit [2].

Results. Qualitatively, our method successfully edits object composition in videos while making appropriate contextual adjustments, such as the new reflection beneath the wine glass in Figure 1 and the new shadows beneath the transformed car, apple, and vase in rows 2, 3, and 5 of Figure 3, respectively. In comparison, DiffusionHandles [3] (the fourth column in Figure 3) alters the identity of objects or the background across different frames, as seen in the second row, and frequently duplicates objects, as shown in the first row. These failures are more evident in the videos shown in our project page. Direct frame warping (the second column) and its refined one by SDEdit [2] (the third column) also typically produce visual seams (second row) and implausible objects (fourth row) due to inaccuracies in warping.

Due to the lack of standard metrics for video editing evaluation, we conducted a user study assessing plausibility, identity preservation, and edit coherence. Figure 4 shows human preferences when participants were presented with two videos–one generated by our method and the other by a competing method–along with the input video, and were asked to choose the better one based on each criterion. The results show that our method is preferred over all baselines across all criteria by significant margins. Notably, our method achieved a preference of 100% for plausibility compared to DiffusionHandles [3], and 75% and 57% for identity preservation and edit coherence compared to the SDEdit [2] output of the direct frame warping.

4. Conclusion and Limitations

We have presented *VideoHandles*, the first method that leverages the prior of video generative models for editing object composition in videos. Experimental results, including a user study, demonstrate that *VideoHandles* outperforms per-frame editing methods in terms of plausibility, identity preservation, and edit coherence. Despite the promising results, the performance of our method is still constrained by the capabilities of video generative models. In future work, we aim to explore the editing of videos with dynamic scenes.

References

[1] Dave Epstein, Allan Jabri, Ben Poole, Alexei Efros, and Aleksander Holynski. Diffusion self-guidance for controllable image generation. *NeurIPS*, 36, 2023. 3

[2] Chenlin Meng, Yutong He, Yang Song, Jiaming Song, Jiajun Wu, Jun-Yan Zhu, and Stefano Ermon. SDEdit: Guided image synthesis and editing with stochastic differential equations. In *ICLR*, 2021. 4

[3] Karran Pandey, Paul Guerrero, Matheus Gadelha, Yannick Hold-Geoffroy, Karan Singh, and Niloy J Mitra. Diffusion handles enabling 3d edits for diffusion models by lifting activations to 3d. In *CVPR*, 2024. 2, 4

[4] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *ICCV*, 2023. 3

[5] Shuzhe Wang, Vincent Leroy, Yohann Cabon, Boris Chidlovskii, and Jerome Revaud. Dust3r: Geometric 3d vision made easy. In *CVPR*, 2024. 2