Neural Assets: 3D-Aware Multi-Object Scene Synthesis with Image Diffusion Models

Ziyi Wu[3,4,†], **Yulia Rubanova**[1], **Rishabh Kabra**[1,5], **Drew A. Hudson**[1],
Igor Gilitschenski[3,4], **Yusuf Aytar**[1], **Sjoerd van Steenkiste**[2],
Kelsey R. Allen[1], **Thomas Kipf**[1]
[1]Google DeepMind [2]Google Research [3]University of Toronto [4]Vector Institute [5]UCL

Abstract

We address the problem of multi-object 3D pose control in image diffusion models. Instead of conditioning on a sequence of text tokens, we propose to use a set of per-object representations, *Neural Assets*, to control the 3D pose of individual objects in a scene. Neural Assets are obtained by pooling visual representations of objects from a reference image, such as a frame in a video, and are trained to reconstruct the respective objects in a different image, e.g., a later frame in the video. Importantly, we encode object visuals from the reference image while conditioning on object poses from the target frame. This enables learning disentangled appearance and pose features. Combining visual and 3D pose representations in a sequence-of-tokens format allows us to keep the text-to-image architecture of existing models, with Neural Assets in place of text tokens. By fine-tuning a pre-trained text-to-image diffusion model with this information, our approach enables fine-grained 3D pose and placement control of individual objects in a scene. We further demonstrate that Neural Assets can be transferred and recomposed across different scenes. Our model achieves state-of-the-art multi-object editing results on both synthetic 3D scene datasets, as well as two real-world video datasets (Objectron, Waymo Open). Additional details and video results are available at our project page.

1 Introduction

From animation movies to video games, the field of computer graphics has long relied on a traditional workflow for creating and manipulating visual content. This approach involves the creation of 3D assets, which are then placed in a scene and animated to achieve the desired visual effects. With the recent advance of deep generative models [26, 50, 77, 82], a new paradigm is emerging. Diffusion models have achieved promising results in content creation [22, 44, 70, 79] by training on web-scale text-image data [85]. Users can now expect realistic image generation, depicting almost everything describable in text. However, text alone is often insufficient for precise control over the output image.

To address this challenge, an emerging body of work has investigated alternative ways to control the image generation process. One line of work studies different forms of conditioning inputs, such as depth maps, surface normals, and semantic layouts [59, 103, 116]. Another direction is personalized image generation [30, 58, 81], which aims to synthesize a new image while preserving particular aspects of a reference image (e.g., placing an object of interest on a desired background). However, these approaches are still fundamentally limited in their 3D understanding of objects. As a result, they cannot achieve intuitive object control in the 3D space, e.g., rotation. While some recent works introduce 3D geometry to the generation process [8, 61, 67], they cannot handle multi-object real-world scenes as it is hard to obtain scalable training data (paired images and 3D annotations).

[†]Work done while interning at Google.

Contact: tkipf@google.com. Project page: neural-assets.github.io

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

https://doi.org/10.52202/079017-2429

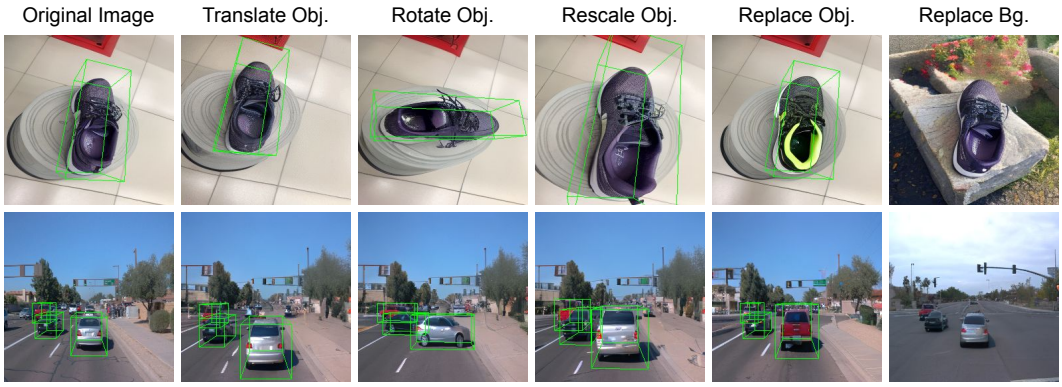


Figure 1: **3D-aware editing with our Neural Asset representations**. Given a source image and object 3D bounding boxes, we can translate, rotate, and rescale the object. In addition, we support compositional generation by transferring objects or backgrounds across images.

We address these limitations by taking inspiration from cognitive science to propose a scalable solution to 3D-aware multi-object control. When humans move through the world, their motor systems keep track of their movements through an efference copy and proprioceptive feedback [4, 96]. This allows the human perceptual system to track objects accurately across time even when the object's relative pose to the observer changes [28]. We use this observation to propose the use of *videos* of multiple objects as a scalable source of training data for 3D multi-object control. Specifically, for any two frames sampled from a video, the naturally occurring changes in the 3D pose (e.g., 3D bounding boxes) of objects can be treated as training labels for multi-object editing.

With this source of training data, we propose Neural Assets – per object latent representations with consistent 3D appearance but variable 3D pose. Neural Assets are trained by extracting their visual appearances from one frame in a video and reconstructing their appearances in a different frame in the video conditioned on the corresponding 3D bounding boxes. This supports learning consistent 3D appearance disentangled from 3D pose. We can then tokenize any number of Neural Assets and feed this sequence to a fine-tuned conditional image generator for precise, multi-object, 3D control.

Our main contributions are threefold: **(i)** A Neural Asset formulation that represent objects with disentangled appearance and pose features. By training on paired video frames, it enables fine-grained 3D control of individual objects. **(ii)** Our framework is applicable to both synthetic and real-world scenes, achieving state-of-the-art results on 3D-aware object editing. **(iii)** We extend Neural Assets to further support compositional scene generation, such as swapping the background of two scenes and transferring objects across scenes. We show the versatile control ability of our model in Fig. 1.

2 Related Work

2D spatial control in diffusion models (DMs). With the rapid growth of diffusion-based visual generation [22, 44, 70, 79, 82, 94], there have been many works aiming to inject spatial control to pre-trained DMs via 2D bounding boxes or segmentation masks. One line of research achieves this by manipulating text prompts [11, 32, 51], intermediate attention maps [12, 16, 17, 27, 41, 52, 110] or noisy latents [25, 64, 69, 90] in the diffusion process, without the need to change model weights. Closer to ours are methods that fine-tune pre-trained DMs to support additional spatial conditioning inputs [5, 29, 33, 45, 112, 114]. GLIGEN [59] introduces new attention layers to condition on bounding boxes. InstanceDiffusion [103] further supports object masks, points, and scribbles with a unified feature fusion block. To incorporate dense control signals such as depth maps and surface normals, ControlNet [116] adds zero-initialized convolution layers around the original network blocks. Recently, Boximator [99] demonstrates that such 2D control can be extended to video models with a similar technique. Several existing works [3, 107] leverage natural motion observed in video and similar to our work propose to train on paired video frames to achieve pixel-level control. In our work, we build upon pre-trained DMs and leverage 3D bounding boxes as spatial conditioning, which enables 3D-aware control such as object rotation and occlusion handling.

https://doi.org/10.52202/079017-2429

3D-aware image generation. Earlier works leverage differentiable rendering to learn 3D Generative Adversarial Networks (GANs) [34] from monocular images, with explicit 3D representations such as radiance fields [14, 15, 37, 71, 86, 111] and meshes [18, 19, 31, 73, 74]. Inspired by the great success of DMs in image generation, several works try to lift the 2D knowledge to 3D [49, 60, 62, 66, 75, 89, 98, 105]. The pioneering work 3DiM [105] and follow-up work Zero-1-to-3 [61] directly train diffusion models on multi-view renderings of 3D assets. However, this line of research only considers single objects without background, which cannot handle in-the-wild data with complex backgrounds. Closest to ours are methods that process multi-object real-world scenes [2, 72, 84, 115]. OBJect-3DIT [67] studies language-guided 3D-aware object editing by training on paired synthetic data, limiting its performance on real-world images [115]. LooseControl [8] converts 3D bounding boxes to depth maps to guide the object pose. Yet, it cannot be directly applied to edit existing images. In contrast, our Neural Asset representation captures both object appearance and 3D pose. It can be easily trained on real-world videos to achieve multi-object 3D edits.
From a methodology perspective, there have been prior works learning disentangled appearance and pose representations for 3D-aware multi-object image editing [71, 100, 111]. However, they are all based on the GAN framework [34] and do not learn generalizable object representations via an encoder. In contrast, we build upon a large-scale pre-trained image diffusion model [79] and powerful feature extractors [13], enabling editing of complex real-world scenes.

Personalized image generation. Since the seminal works DreamBooth [81] and Textual Inversion [30] which perform personalized generation via test-time fine-tuning, huge efforts have been made to achieve this in a zero-shot manner [47, 58, 88, 101, 106, 109]. Most of them are only able to synthesize one subject, and cannot control the spatial location of the generated instance. A notable exception is Subject-Diffusion [65], which leverages frozen CLIP embeddings for object appearance and 2D bounding boxes for object position. Still, it cannot explicitly control the 3D pose of objects.

Object-centric representation learning. Our Neural Asset representation is also related to recent object-centric slot representations [48, 63, 91, 92, 108] that decompose scenes into a set of object entities. Object slots provide a useful interface for editing such as object attributes [93], motions [87], 3D poses [46], and global camera poses [83]. Nevertheless, these models show significantly degraded results on real-world data. Neural Assets also consist of disentangled appearance and pose features of objects. Different from existing slot-based models, we fine-tune self-supervised visual encoders and connect them with large-scale pre-trained DMs, which scales up to complex real-world data.

3 Method: Neural Assets

Inspired by 3D assets in computer graphics software, we propose Neural Assets as learnable object-centric representations. A Neural Asset comprises an appearance and an object pose representation, which is trained to reconstruct the object via conditioning a diffusion model (Sec. 3.2). Trained on paired images, our method learns disentangled representations, enabling 3D-aware object editing and compositional generation at inference time (Sec. 3.3). Our framework is summarized in Fig. 2.

3.1 Background: 3D Assets in Computer Graphics

3D object models, or *3D assets*, are basic components of any 3D scene in computer graphics software, such as Blender [20]. A typical workflow includes selecting N 3D assets $\{\hat{a}_1, ..., \hat{a}_N\}$ from an asset library and placing them into a scene. Formally, one can define a 3D asset as a tuple $\hat{a}_i \triangleq (\mathcal{A}_i, \mathcal{P}_i)$, where $\mathcal{A}_i$ is a set of descriptors defining the asset's appearance (e.g., canonical 3D shape and surface textures) and $\mathcal{P}_i$ describes its pose (e.g., rigid transformation and scaling from its canonical pose).

3.2 Neural Assets

Inspired by 3D assets in computer graphics, our goal is to enable such capabilities (i.e., 3D control and compositional generation) in recent generative models. To achieve this, we define a *Neural Asset* as a tuple $a_i \triangleq (A_i, P_i)$, where $A_i \in \mathbb{R}^{(K \times D)}$ is a flattened sequence of K D-dimensional vectors describing the appearance of an asset, and $P_i \in \mathbb{R}^{D'}$ is a D'-dimensional embedding of the asset's pose in a scene. In other words, a Neural Asset is fully described by learnable embedding vectors, factorized into appearance and pose. This factorization enables independent control over appearance and pose of an asset, similar to how 3D object models can be controlled in traditional computer

https://doi.org/10.52202/079017-2429

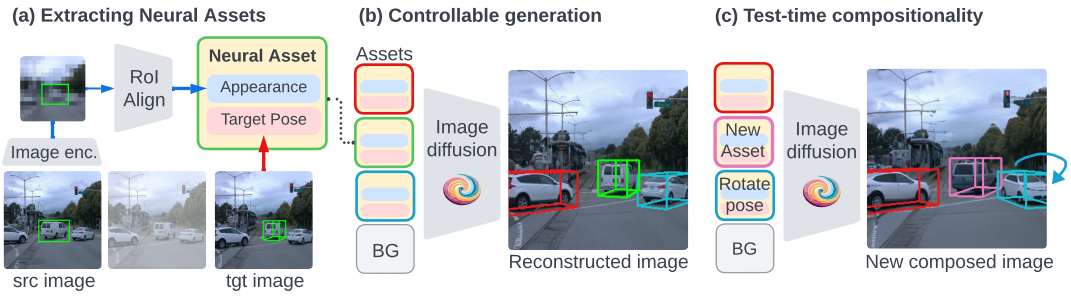


Figure 2: **Neural Assets framework**. (a) We train our model on pairs of video frames, which contain objects under different poses. We encode appearance tokens from a source image with RoIAlign, and pose tokens from the objects' 3D bounding boxes in a target image. They are combined to form our Neural Asset representations. (b) An image diffusion model is conditioned on Neural Assets and a separate background token to reconstruct the target image as the training signal. (c) During inference, we can manipulate the Neural Assets to control the objects in the generated image: rotate the object's pose (blue) or replace an object by a different one from another image (pink).

graphics software. Importantly, besides the 3D pose of assets, our approach does not require any explicit mapping of objects into 3D, such as depth maps or the NeRF representation [68].

3.2.1 Asset Encoding

In the following, we describe how both the appearance A_i and the pose P_i of a Neural Asset a_i are obtained from visual observations (such as an image or a frame in a video). Importantly, the appearance and pose representations are not necessarily encoded from the same observation, i.e., they can be encoded from two separate frames sampled from a video. We find this strategy critical to learn disentangled and controllable representations, which we will discuss in detail in Sec. 3.3.

Appearance encoding. At a high level, we wish to obtain a set of N Neural Asset appearance tokens A_i from a visual observation x_{src}, where x_{src} can be an image or a frame in a video. While one could approach this problem in a fully-unsupervised fashion, using a method such as Slot Attention [63] to decompose an image into a set of object representations, we choose to use readily-available annotations to allow fine-grained specification of objects of interest. In particular, we assume that a 2D bounding box b_i is provided for each Neural Asset a_i, specifying which object should be extracted from x_{src}. Therefore, we obtain the appearance representation A_i as follows:

$$A_i = \text{Flatten}(\text{RoIAlign}(H_i, b_i)), \quad H_i = \text{Enc}(x_{\text{src}}), \tag{1}$$

where H_i is the output feature map of a visual encoder Enc. RoIAlign [38] extracts a fixed size feature map using the provided bounding box b_i which is flattened to form the appearance token A_i. This factorization allows us to extract N object appearances from an image with just one encoder forward pass. In contrast, previous methods [65, 109] crop each object out to extract features separately, and thus requires N encoder passes. This becomes unaffordable if we jointly fine-tune the visual encoder, which is key to learning generalizable features as we will show in the ablation study.

Pose encoding. The pose token P_i of a Neural Asset a_i is the primary interface for controlling the presence and 3D pose of an object in the rendered scene. In this work, we assume that the object pose is provided in terms of a 3D bounding box, which fully specifies its location, orientation, and size in the scene. Formally, we take four corners spanning the 3D bounding box[1] and project them to the image plane to get $\{c_i^j = (h_i^j, w_i^j, d_i^j)\}_{j=1}^4$, with the projected 2D coordinate (h_i^j, w_i^j), and the 3D depth $d_{i,j}$. We obtain the pose representation P_i for a Neural Asset as follows:

$$P_i = \text{MLP}(C_i), \quad C_i = \text{Concat}[c_i^1, c_i^2, c_i^3, c_i^4], \tag{2}$$

where we first concatenate the four corners c_i^j to form $C_i \in \mathbb{R}^{12}$, and then project it to $P_i \in \mathbb{R}^{D'}$ via an MLP. We tried the Fourier coordinate encoding in prior works [59, 103] but did not find it helpful.

There are alternative ways to represent 3D bounding boxes (e.g., concatenation of center, size, and rotation commonly used in 3D object detection [57]), which we compare in Appendix B.4. In this

[1]Only three corners are needed to fully define a 3D bounding box, but we found a 4-corner representation beneficial to work with. Previous research [118] also shows that over-parametrization can benefit model learning.

https://doi.org/10.52202/079017-2429

work, we assume the availability of training data with 3D annotations – obtaining high-quality 3D object boxes for videos at scale is still an open research problem, but may soon be within reach given recent progress in monocular 3D detection [102], depth estimation [7, 113], and pose tracking [9].

Serialization of multiple Neural Assets. We encode a set of N Neural Assets into a sequence of tokens that can be appended to or used in place of text embeddings for conditioning a generative model. In particular, we first concatenate the appearance token A_i and the pose token P_i channel-wise, and then linearly project it to obtain a Neural Asset representation a_i as follows:

$$a_i = \text{Linear}(\tilde{a}_i)\,, \quad \tilde{a}_i = \text{Concat}[A_i, P_i] \in \mathbb{R}^{K \times D + D'}\,. \tag{3}$$

Channel-wise concatenation uniquely binds one pose token with one appearance representation in the presence of multiple Neural Assets. An alternative solution is to learn such association with positional encoding. Yet, it breaks the permutation-invariance of the generator against the order of input objects and leads to poor results in our preliminary experiments. Finally, we simply concatenate multiple Neural Assets along the token axis to arrive at our token sequence, which can be used as a drop-in replacement for a sequence of text tokens in a text-to-image generation model.

Background modeling. Similar to prior works [71, 111], we found it helpful to encode the scene background separately, which enables independent control thereof (e.g., swapping out the scene, or controlling global properties such as lighting). We choose the following heuristic strategy to encode the background: to avoid leakage of foreground object information, we mask all pixels within asset bounding boxes b_i. We then pass this masked image through the image encoder Enc (shared weights with the foreground asset encoder) and apply a global RoIAlign, i.e., using the entire image as region of interest, to obtain a background appearance token $A_{\text{bg}} \in \mathbb{R}^{(K \times D)}$. Similar to a Neural Asset, we also attach a pose token P_{bg} to A_{bg}. This can either be a timestep embedding of the video frame (relative to the source frame) or a relative camera pose embedding, if available. In the serialized representations, the background token is treated the same as Neural Assets, i.e., we concatenate A_{bg} and P_{bg} channel-wise and linearly project it. Finally, the foreground assets a_i and the background token are concatenated along the token dimension and used to condition the generator.

3.2.2 Generative Decoder

To generate images from Neural Assets, we make minimal assumptions about the architecture or training setup of the generative image model to ensure compatibility with future large-scale pre-trained image generators. In particular, we assume that the generative image model accepts a sequence of tokens as conditioning signal: for most base models this would be a sequence of tokens derived from text prompts, which we can easily replace with a sequence of Neural Asset tokens.

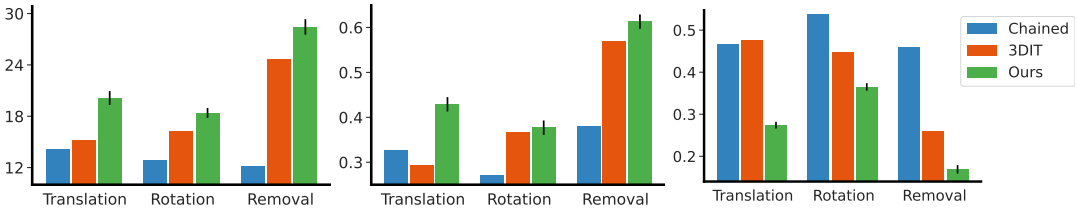
As a representative for this class of models, we adopt Stable Diffusion v2.1 [79] for the generative decoder. See Appendix C for details on this model. Starting from the pre-trained text-to-image checkpoint, we fine-tune the entire model end-to-end to accept Neural Assets tokens instead of text tokens as conditioning signal. The training and inference setup is explained in the following section.

3.3 Learning and Inference

Learning from frame pairs. As outlined in the introduction, we require a scalable data source of object-level "edits" in 3D space to effectively learn multi-object 3D control capabilities. Video data offers a natural solution to this problem: as the camera and the content of the scene moves or changes over time, objects are observed from various view points and thus in various poses and lighting conditions [78]. We exploit this signal by randomly sampling pairs of frames from video clips, where we take one frame as the "source" image x_{src} and the other frame as the "target" image x_{tgt}.

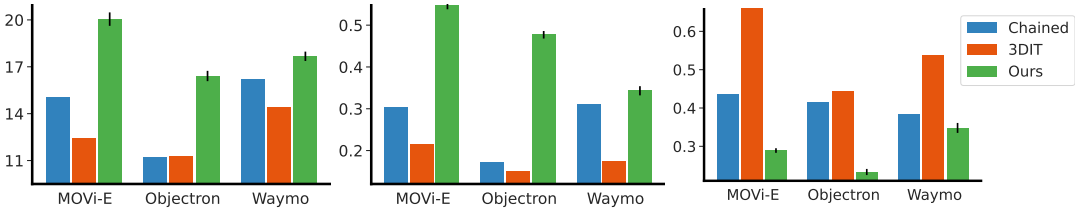
As described earlier, we obtain the appearance token A_i of Neural Assets from the *source* frame x_{src} by extracting object features using 2D box annotations. Next, we obtain the pose token P_i for each extracted asset from the *target* frame x_{tgt}, for which we need to identify the correspondences between objects in both frames. In practice, such correspondences can be obtained, for example, by applying an object tracking model on the underlying video. Finally, with the associated appearance and pose representations, we condition the image generator on them and train it to reconstruct the target frame x_{tgt}, i.e., using the denoising loss of Stable Diffusion v2.1 in our case. Such a paired frame training strategy forces the model to learn an appearance token that is invariant to object pose and leverage the pose token to synthesize the new object, avoiding the trivial solution of simple pixel-copying.

https://doi.org/10.52202/079017-2429



(a) PSNR (higher is better) (b) SSIM (higher is better) (c) LPIPS (lower is better)

Figure 3: **Single-object editing results on OBJect unseen object subset**. We evaluate on the *Translation*, *Rotation*, and *Removal* tasks. We follow 3DIT [67] to compute metrics inside the edited object's bounding box. Our results are averaged over 3 random seeds.



(a) PSNR (higher is better) (b) SSIM (higher is better) (c) LPIPS (lower is better)

Figure 4: **Multi-object editing results on MOVi-E, Objectron, and Waymo Open** (denoted as Waymo in the figures). We compute metrics inside the edited objects' bounding boxes.

Test-time controllability. The learned disentangled representations naturally enable multi-object scene-level editing as we will show in Sec. 4.3. Since we encode 3D bounding boxes to pose tokens P_i, we can move, rotate, and rescale objects by changing the box coordinates. We can also compose Neural Assets a_i across scenes to generate new scenes. In addition, our background modeling design supports swapping the environment map of the scene. Importantly, as we will see in the experiments, our image generator learns to naturally blend the objects into their new environment at new positions, with realistic lighting effects such as rendering and adapting shadows correctly.

4 Experiments

In this section, we conduct extensive experiments to answer the following questions: **(i)** Can Neural Assets enable accurate 3D object editing? **(ii)** What practical applications does our method support on real-world scenes? **(iii)** What is the impact of each design choice in our framework?

4.1 Experimental Setup

Datasets. We select four datasets with object or camera motion, which span different levels of complexity. *OBJect* [67] is introduced in 3DIT [67], which is one of our baselines. It contains 400k synthetic scenes rendered by Blender [20] with a static camera. Up to four Objaverse [21] assets are placed on a textured ground and only one object is randomly moved on the ground. For a fair comparison with 3DIT, we use 2D bounding boxes plus rotation angles as object poses, and follow them to base our model on Stable Diffusion v1.5 [79]. *MOVi-E* [36] consists of Blender simulated videos with up to 23 objects. It is more challenging than OBJect as it has linear camera motion and there can be multiple objects moving simultaneously. *Objectron* [1] is a big step up in complexity as it captures real-world objects with complex backgrounds. 15k object-centric videos covering objects from nine categories are recorded with 360° camera movement. *Waymo Open* [97] is a real-world self-driving dataset captured by car mounted cameras. We follow prior work [111] to use only the front view and filter out cars that are too small. See Appendix A.1 for more details on datasets.

Baselines. We compare to methods that can perform 3D-aware editing on existing images and have released their code. *3DIT* [67] fine-tunes Zero-1-to-3 [61] on the OBJect dataset to support translation and rotation of objects. However, it cannot render big viewpoint changes as it does not encode camera poses. Following [67], we create another baseline (dubbed *Chained*) by using SAM [55] to segment the object of interest, removing it using Stable Diffusion inpainting model [79], running Zero-1-to-3

https://doi.org/10.52202/079017-2429

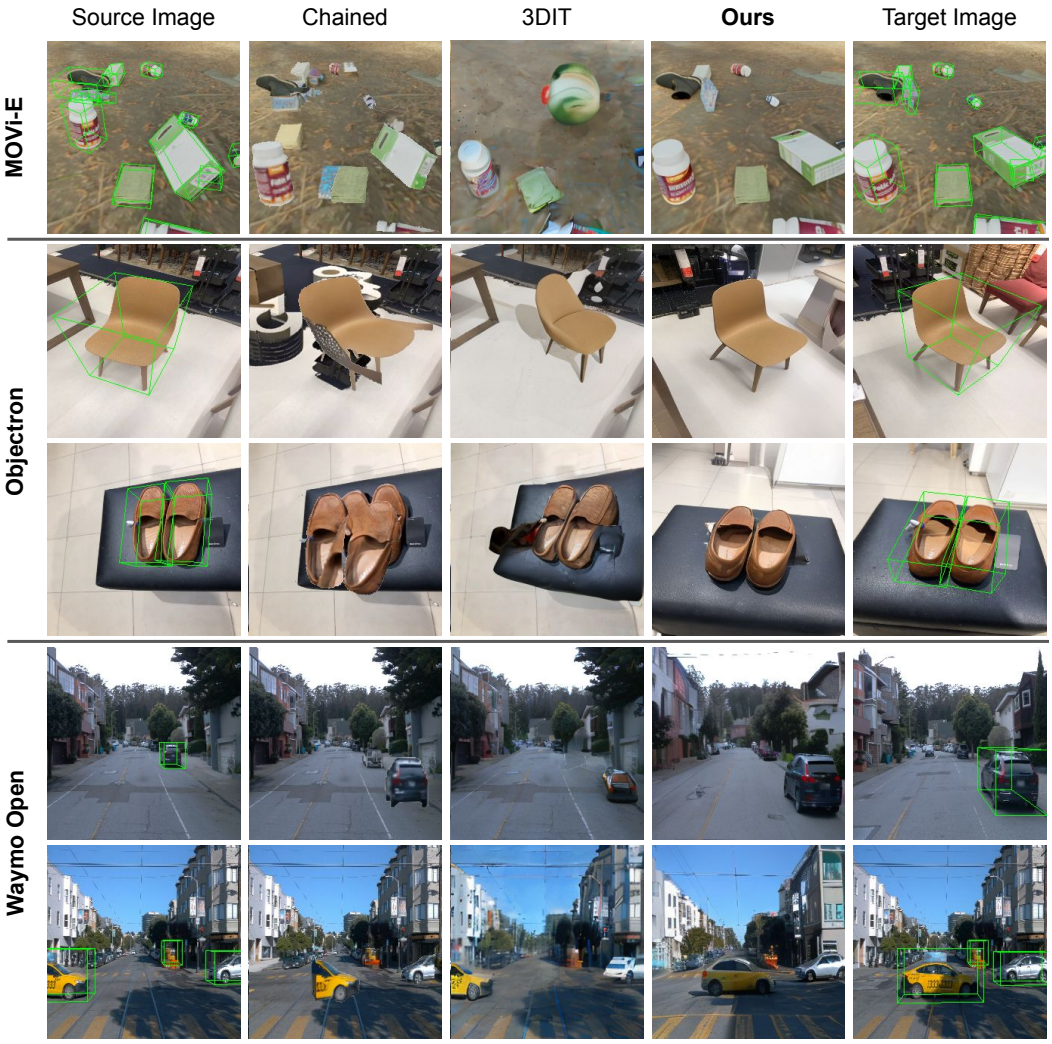


Figure 5: **Qualitative comparison on MOVi-E, Objectron, and Waymo Open**. All models generate a new image given a source image and the 3D bounding box of target objects. Our method performs the best in object identity preservation, editing accuracy, and background modeling.

to rotate and scale the object, and stitching it to the target position. Since none of these baselines can control multiple objects simultaneously, we apply them to edit all objects sequentially.

Evaluation settings. We report common metrics to measure the quality of the edited image – PSNR, SSIM [104], LPIPS [117], and FID [42]. Following prior works [49, 67], we also compute object-level metrics on cropped out image patches of edited objects. To evaluate the fidelity of edited objects, we take the DINO [13] feature similarity metric proposed in [81]. On video datasets, we randomly sample source and target images in each testing video and fix them across runs for consistent results.

Implementation Details. For all experiments, we resize images to 256×256. DINO self-supervised pre-trained ViT-B/8 [13] is adopted as the visual encoder Enc, and jointly fine-tuned with the generator. All our models are trained using the Adam optimizer [53] with a batch size of 1536 on 256 TPUv4 chips. For inference, we generate images by running the DDIM [95] sampler for 50 steps. For more training and inference details, please refer to Appendix A.4.

4.2 Main Results

Single-object editing. We first compare the ability to control the 3D pose of a single object on the OBJect dataset. Fig. 3 presents the results on the unseen object subset. We do not show FID here as it mainly measures the visual quality of generated examples, which does not reflect the editing accuracy.

https://doi.org/10.52202/079017-2429

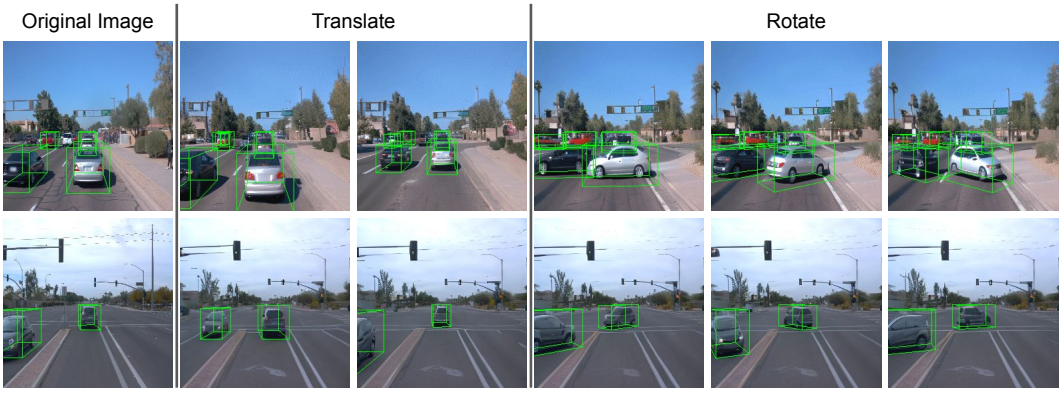


Figure 6: **Object translation and rotation** by manipulating 3D bounding boxes on Waymo Open. See our project page for videos and additional object rescaling results.

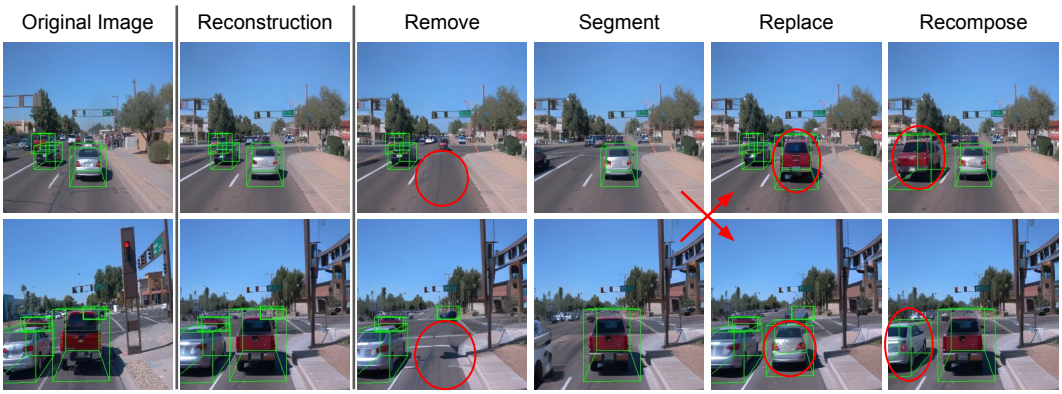


Figure 7: **Compositional generation results** on Waymo Open. By composing Neural Assets, we can remove and segment objects, as well as transfer and recompose objects between scenes.

For results on the seen object subset and FID, please refer to Appendix B.1, where we observe similar trends. Compared to baselines, our model does not condition on text (e.g., the category name of the object to edit) as in 3DIT and is not trained on curated multi-view images of 3D assets as in Zero-1-to-3. Still, we achieve state-of-the-art performance on all three tasks. This is because our Neural Assets representation learns disentangled appearance and pose features, which is able to preserve object identity while changing its placement smoothly. Also, the fine-tuned DINO encoder generalizes better to unseen objects compared to the frozen CLIP visual encoder used by baselines.

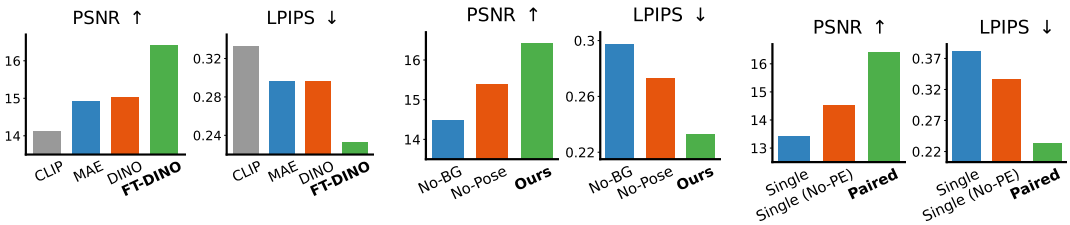
Multi-object editing. Fig. 4 shows the results on MOVi-E, Objectron, and Waymo Open, where multiple objects are manipulated in each sample. Similar to the single-object case, we compute metrics inside the object bounding boxes, and leave the image-level results to Appendix B.1. Our model outperforms baselines by a sizeable margin across datasets. Fig. 5 presents the qualitative results. When there are multiple objects of the same class in the scene (e.g., boxes in the MOVi-E example and cars on Waymo Open), 3DIT is unable to edit the correct instance. In addition, it generalizes poorly to real-world scenes. Thanks to the object cropping step, Chained baseline can identify the correct object of interest. However, the edited object is simply pasted to the target location, leading to unrealistic appearance due to missing lighting effects such as shadows. In contrast, our model is able to control all objects precisely, preserve their fidelity, and blend them into the background naturally. Since we encode the camera pose, we can also model global viewpoint change as shown in the third row. See Appendix B.1 for additional qualitative results.

4.3 Controllable Scene Generation

In this section, we show versatile control of scene objects on Waymo Open. For results on Objectron, please refer to Appendix B.3. As shown in Fig. 6, we can translate and rotate cars in driving scenes. The model understands the 3D world as objects zoom in and out when moving, and show consistent

https://doi.org/10.52202/079017-2429

Figure 8: **Transfer backgrounds between scenes** by replacing the background token on Waymo Open. The objects can adapt to new environments, e.g., the car lights are turned on at night.



(a) Visual encoder (FT: fine-tuned). (b) Background modeling. (c) Training strategy.

Figure 9: **Comparison of (a) visual encoders, (b) background modeling, and (c) training strategies on Objectron**. Bold entry denotes our full model. See text for each variant. We report PSNR and LPIPS computed within object bounding boxes, and leave other metrics to Appendix B.2.

novel views when rotating. Fig. 7 presents our ability of compositional generation, where objects are removed, segmented out, and transferred across scenes. Notice how the model handles occlusion and inpaints the scene properly. Finally, Fig. 8 demonstrates background swapping between scenes. The generator is able to harmonize objects with the new environment. For example, the car lights are turned on and rendered with specular highlight when using a background image from a night scene.

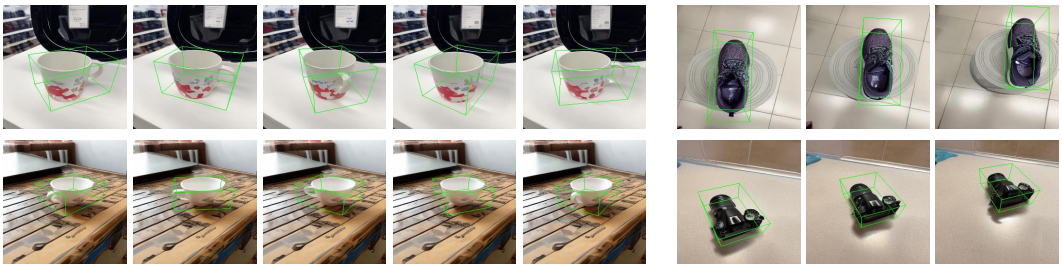
4.4 Ablation Study

We study the effect of each component in the model. All ablations are run on Objectron since it is a real-world dataset with complex background, and has higher object diversity than Waymo Open.

Visual encoder. Previous image-conditioned diffusion models [49, 61, 62] usually use the frozen image encoder of CLIP [76] to extract visual features. Instead, as shown in Fig. 9a, we found that both MAE [39] and DINO [13] pre-trained ViTs give better results. This is because CLIP's image encoder only captures high-level semantics of images, which suffices in single-object tasks, but fails in our multi-object setting. In contrast, MAE and DINO pre-training enable the model to extract more fine-grained features. Besides, DINO outperforms MAE as its features contain richer 3D information, which aligns with recent research [6]. Finally, jointly fine-tuning the image encoder learns more generalizable appearance tokens in Neural Assets, leading to the best performance.

Background modeling. We compare our full model with two variants: **(i)** not conditioning on any background tokens (dubbed *No-BG*), and **(ii)** conditioning on background appearance tokens but not using relative camera pose as pose tokens (dubbed *No-Pose*). As shown in Fig. 9b, our background modeling strategy performs the best in image-level metrics as backgrounds usually occupy a large part of real-world images. Interestingly, our method also achieves significantly better object-level metrics. This is because given background appearance and pose, the model does not need to infer them from object tokens, leading to more disentangled Neural Assets representations.

https://doi.org/10.52202/079017-2429



(a) Symmetry ambiguity when rotating objects (b) Object-camera motion entanglement

Figure 10: **Failure case analysis**. Our model mainly has two failure cases: (a) symmetry ambiguity, where the handle of the cup gets flipped when it rotates by 180 degrees; (b) camera-object motion entanglement, where the background also moves when we translate the foreground object. Both issues will likely be resolved if we train our Neural Assets model on more diverse data.

Training strategy. As described in Sec. 3.3, we train on videos and extract appearance and pose tokens from different frames. We compare such design with training on a single frame in Fig. 9c. Our paired frame training strategy clearly outperforms single frame training. Since the appearance token is extracted by a ViT with positional encoding, it already contains object position information, which acts as a shortcut for image reconstruction. Therefore, the model ignores the input object pose token, resulting in poor controllability. One way to alleviate this is removing the positional encoding in the image encoder (dubbed *NO-PE*), which still underperforms paired frame training. This is because to reconstruct objects with visual features extracted from a different frame, the model is forced to infer their underlying 3D structure instead of simply copying pixels. In addition, the generator needs to render realistic lighting effects such as shadows under the new scene configuration.

5 Conclusion

In this paper, we present Neural Assets, vector-based representations of objects and scene elements with disentangled appearance and pose features. By connecting with pre-trained image generators, we enable controllable 3D scene generation. Our method is capable of controlling multiple objects in the 3D space as well as transferring assets across scenes, both on synthetic and real-world datasets. We view our work as an important step towards general-purpose neural-based simulators.

Limitations and future works. One main failure case of our model is symmetry ambiguity. As can be seen from the rotation results in Fig. 10 (a), the handle of the cup gets flipped when it rotates by 180 degree. Another failure case that only happens on Objectron is the entanglement of global camera motion and local object movement (Fig. 10 (b)). This is because Objectron videos only contain camera motion while objects always stay static. Both issues will likely be resolved if we train our model on larger-scale datasets with more diverse object and camera motion.

An ideal Neural Asset should enable control over all potential configurations of an object such as deformation (e.g., a walking cat), rigid articulation (e.g., opening of a scissor), and structural decomposition (e.g., tomatoes being cut). In this work, we first tackle the foremost important aspect, i.e., controlling 3D rigid object pose and background composition which applies to almost all the objects. Hence our current method does not allow for controlling structural changes. However, it can be adapted when suitable datasets are developed that capture other changes in objects.

Another limitation is that our approach is currently limited to existing datasets that have 3D bounding box annotations. Yet, with recent advances in vision foundation models [9, 55, 113], we may soon have scalable 3D annotation pipelines similar to their 2D counterparts. One notable example is OmniNOCS [56], which works on both Waymo and Objectron (datasets we used in this work), and *diverse, in-the-wild Internet images* for a wide range of object classes. It can be used to create larger open domain datasets to learn Neural Assets. We see this as an interesting future direction.

Acknowledgements

We would like to thank Etienne Pot, Klaus Greff, Shlomi Fruchter, and Amir Hertz for their advise regarding infrastructure. We would further like to thank Mehdi S. M. Sajjadi, João Carreira, Sean Kirmani, Yi Yang, Daniel Zoran, David Fleet, Kevin Murphy, and Mike Mozer for helpful discussions.

References

[1] Adel Ahmadyan, Liangkai Zhang, Artsiom Ablavatski, Jianing Wei, and Matthias Grundmann. Objectron: A large scale dataset of object-centric videos in the wild with pose annotations. In *CVPR*, 2021. 6, 18

[2] Hadi Alzayer, Zhihao Xia, Xuaner Zhang, Eli Shechtman, Jia-Bin Huang, and Michael Gharbi. Magic Fixup: Streamlining photo editing by watching dynamic videos. *arXiv preprint arXiv:2403.13044*, 2024. 3

[3] Hadi Alzayer, Zhihao Xia, Xuaner Zhang, Eli Shechtman, Jia-Bin Huang, and Michael Gharbi. Magic fixup: Streamlining photo editing by watching dynamic videos. *arXiv preprint arXiv:2403.13044*, 2024. 2

[4] Michael A Arbib. *The handbook of brain theory and neural networks*. MIT Press, 2003. 2

[5] Omri Avrahami, Thomas Hayes, Oran Gafni, Sonal Gupta, Yaniv Taigman, Devi Parikh, Dani Lischinski, Ohad Fried, and Xi Yin. SpaText: Spatio-textual representation for controllable image generation. In *CVPR*, 2023. 2

[6] Mohamed El Banani, Amit Raj, Kevis-Kokitsi Maninis, Abhishek Kar, Yuanzhen Li, Michael Rubinstein, Deqing Sun, Leonidas Guibas, Justin Johnson, and Varun Jampani. Probing the 3d awareness of visual foundation models. In *CVPR*, 2024. 9

[7] Shariq Farooq Bhat, Reiner Birkl, Diana Wofk, Peter Wonka, and Matthias Müller. ZoeDepth: Zero-shot transfer by combining relative and metric depth. *arXiv preprint arXiv:2302.12288*, 2023. 5

[8] Shariq Farooq Bhat, Niloy J Mitra, and Peter Wonka. LooseControl: Lifting controlnet for generalized depth conditioning. In *CVPR*, 2024. 1, 3

[9] Jan Kautz Stan Birchfield Bowen Wen, Wei Yang. FoundationPose: Unified 6d pose estimation and tracking of novel objects. In *CVPR*, 2024. 5, 10

[10] James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: composable transformations of Python+NumPy programs, 2018. URL http://github.com/google/jax. 19

[11] Tim Brooks, Aleksander Holynski, and Alexei A Efros. InstructPix2Pix: Learning to follow image editing instructions. In *CVPR*, 2023. 2

[12] Mingdeng Cao, Xintao Wang, Zhongang Qi, Ying Shan, Xiaohu Qie, and Yinqiang Zheng. MasaCtrl: Tuning-free mutual self-attention control for consistent image synthesis and editing. In *ICCV*, 2023. 2

[13] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *ICCV*, 2021. 3, 7, 9, 19

[14] Eric R Chan, Marco Monteiro, Petr Kellnhofer, Jiajun Wu, and Gordon Wetzstein. pi-GAN: Periodic implicit generative adversarial networks for 3d-aware image synthesis. In *CVPR*, 2021. 3

[15] Eric R Chan, Connor Z Lin, Matthew A Chan, Koki Nagano, Boxiao Pan, Shalini De Mello, Orazio Gallo, Leonidas J Guibas, Jonathan Tremblay, Sameh Khamis, et al. Efficient geometry-aware 3d generative adversarial networks. In *CVPR*, 2022. 3

[16] Hila Chefer, Yuval Alaluf, Yael Vinker, Lior Wolf, and Daniel Cohen-Or. Attend-and-Excite: Attention-based semantic guidance for text-to-image diffusion models. *TOG*, 2023. 2

[17] Minghao Chen, Iro Laina, and Andrea Vedaldi. Training-free layout control with cross-attention guidance. In *WACV*, 2024. 2

https://doi.org/10.52202/079017-2429

[18] Wenzheng Chen, Huan Ling, Jun Gao, Edward Smith, Jaakko Lehtinen, Alec Jacobson, and Sanja Fidler. Learning to predict 3d objects with an interpolation-based differentiable renderer. *NeurIPS*, 2019. 3

[19] Wenzheng Chen, Joey Litalien, Jun Gao, Zian Wang, Clement Fuji Tsang, Sameh Khamis, Or Litany, and Sanja Fidler. DIB-R++: learning to predict lighting and material with a hybrid differentiable renderer. *NeurIPS*, 2021. 3

[20] Blender Online Community. *Blender - a 3D modelling and rendering package*. Blender Foundation, Stichting Blender Foundation, Amsterdam, 2018. URL http://www.blender.org. 3, 6, 18

[21] Matt Deitke, Dustin Schwenk, Jordi Salvador, Luca Weihs, Oscar Michel, Eli VanderBilt, Ludwig Schmidt, Kiana Ehsani, Aniruddha Kembhavi, and Ali Farhadi. Objaverse: A universe of annotated 3d objects. In *CVPR*, 2023. 6, 18

[22] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *NeurIPS*, 2021. 1, 2

[23] Laura Downs, Anthony Francis, Nate Koenig, Brandon Kinman, Ryan Hickman, Krista Reymann, Thomas B McHugh, and Vincent Vanhoucke. Google Scanned Objects: A high-quality dataset of 3d scanned household items. In *ICRA*, 2022. 18

[24] Gamaleldin Fathy Elsayed, Aravindh Mahendran, Sjoerd van Steenkiste, Klaus Greff, Michael Curtis Mozer, and Thomas Kipf. SAVi++: Towards end-to-end object-centric learning from real-world videos. *NeurIPS*, 2022. 18

[25] Dave Epstein, Allan Jabri, Ben Poole, Alexei Efros, and Aleksander Holynski. Diffusion self-guidance for controllable image generation. *NeurIPS*, 2023. 2

[26] Patrick Esser, Robin Rombach, and Bjorn Ommer. Taming transformers for high-resolution image synthesis. In *CVPR*, 2021. 1

[27] Weixi Feng, Xuehai He, Tsu-Jui Fu, Varun Jampani, Arjun Reddy Akula, Pradyumna Narayana, Sugato Basu, Xin Eric Wang, and William Yang Wang. Training-free structured diffusion guidance for compositional text-to-image synthesis. In *ICLR*, 2023. 2

[28] Leon Festinger and Lance Kirkpatrick Canon. Information about spatial location based on knowledge about efference. *Psychological Review*, 1965. 2

[29] Oran Gafni, Adam Polyak, Oron Ashual, Shelly Sheynin, Devi Parikh, and Yaniv Taigman. Make-A-Scene: Scene-based text-to-image generation with human priors. In *ECCV*, 2022. 2

[30] Rinon Gal, Yuval Alaluf, Yuval Atzmon, Or Patashnik, Amit Haim Bermano, Gal Chechik, and Daniel Cohen-or. An image is worth one word: Personalizing text-to-image generation using textual inversion. In *ICLR*, 2023. 1, 3

[31] Jun Gao, Tianchang Shen, Zian Wang, Wenzheng Chen, Kangxue Yin, Daiqing Li, Or Litany, Zan Gojcic, and Sanja Fidler. GET3D: A generative model of high quality 3d textured shapes learned from images. *NeurIPS*, 2022. 3

[32] Songwei Ge, Taesung Park, Jun-Yan Zhu, and Jia-Bin Huang. Expressive text-to-image generation with rich text. In *ICCV*, 2023. 2

[33] Vidit Goel, Elia Peruzzo, Yifan Jiang, Dejia Xu, Nicu Sebe, Trevor Darrell, Zhangyang Wang, and Humphrey Shi. PAIR-Diffusion: Object-level image editing with structure-and-appearance paired diffusion models. In *CVPR*, 2024. 2

[34] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative Adversarial Nets. *NeurIPS*, 2014. 3

[35] Google. Safety & fairness considerations for generative models, 2023. URL https://developers.google.com/machine-learning/resources/safety-gen-ai. 24

[36] Klaus Greff, Francois Belletti, Lucas Beyer, Carl Doersch, Yilun Du, Daniel Duckworth, David J Fleet, Dan Gnanapragasam, Florian Golemo, Charles Herrmann, et al. Kubric: A scalable dataset generator. In *CVPR*, 2022. 6, 18

[37] Jiatao Gu, Lingjie Liu, Peng Wang, and Christian Theobalt. StyleNeRF: A style-based 3d aware generator for high-resolution image synthesis. In *ICLR*, 2022. 3

[38] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask R-CNN. In *ICCV*, 2017. 4, 19

[39] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *CVPR*, 2022. 9

[40] Jonathan Heek, Anselm Levskaya, Avital Oliver, Marvin Ritter, Bertrand Rondepierre, Andreas Steiner, and Marc van Zee. Flax: A neural network library and ecosystem for JAX, 2020. URL http://github.com/google/flax. 19

[41] Amir Hertz, Ron Mokady, Jay Tenenbaum, Kfir Aberman, Yael Pritch, and Daniel Cohen-or. Prompt-to-Prompt image editing with cross-attention control. In *ICLR*, 2023. 2

[42] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. GANs trained by a two time-scale update rule converge to a local nash equilibrium. *NeurIPS*, 2017. 7, 19

[43] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. In *NeurIPS Workshop*, 2021. 19

[44] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *NeurIPS*, 2020. 1, 2, 22

[45] Li Hu, Xin Gao, Peng Zhang, Ke Sun, Bang Zhang, and Liefeng Bo. Animate Anyone: Consistent and controllable image-to-video synthesis for character animation. *arXiv preprint arXiv:2311.17117*, 2023. 2

[46] Allan Jabri, Sjoerd van Steenkiste, Emiel Hoogeboom, Mehdi SM Sajjadi, and Thomas Kipf. DORSal: Diffusion for object-centric representations of scenes et al. In *ICLR*, 2024. 3

[47] Xuhui Jia, Yang Zhao, Kelvin CK Chan, Yandong Li, Han Zhang, Boqing Gong, Tingbo Hou, Huisheng Wang, and Yu-Chuan Su. Taming encoder for zero fine-tuning image customization with text-to-image diffusion models. *arXiv preprint arXiv:2304.02642*, 2023. 3

[48] Jindong Jiang, Fei Deng, Gautam Singh, and Sungjin Ahn. Object-Centric slot diffusion. *NeurIPS*, 2023. 3

[49] Yash Kant, Ziyi Wu, Michael Vasilkovsky, Guocheng Qian, Jian Ren, Riza Alp Guler, Bernard Ghanem, Sergey Tulyakov, Igor Gilitschenski, and Aliaksandr Siarohin. SPAD: Spatially aware multiview diffusers. In *CVPR*, 2024. 3, 7, 9, 19

[50] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan. In *CVPR*, 2020. 1

[51] Bahjat Kawar, Shiran Zada, Oran Lang, Omer Tov, Huiwen Chang, Tali Dekel, Inbar Mosseri, and Michal Irani. Imagic: Text-based real image editing with diffusion models. In *CVPR*, 2023. 2

[52] Yunji Kim, Jiyoung Lee, Jin-Hwa Kim, Jung-Woo Ha, and Jun-Yan Zhu. Dense text-to-image generation with attention modulation. In *ICCV*, 2023. 2

[53] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015. 7, 19

[54] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. In *ICLR*, 2014. 22

 https://doi.org/10.52202/079017-2429

[55] 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. 6, 10, 18

[56] Akshay Krishnan, Abhijit Kundu, Kevis-Kokitsi Maninis, James Hays, and Matthew Brown. OmniNOCS: A unified nocs dataset and model for 3d lifting of 2d objects. In *ECCV*, 2024. 10

[57] Alex H Lang, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang, and Oscar Beijbom. PointPillars: Fast encoders for object detection from point clouds. In *CVPR*, 2019. 4, 22

[58] Dongxu Li, Junnan Li, and Steven Hoi. BLIP-Diffusion: Pre-trained subject representation for controllable text-to-image generation and editing. *NeurIPS*, 2023. 1, 3

[59] Yuheng Li, Haotian Liu, Qingyang Wu, Fangzhou Mu, Jianwei Yang, Jianfeng Gao, Chunyuan Li, and Yong Jae Lee. GLIGEN: Open-set grounded text-to-image generation. In *CVPR*, 2023. 1, 2, 4

[60] Chen-Hsuan Lin, Jun Gao, Luming Tang, Towaki Takikawa, Xiaohui Zeng, Xun Huang, Karsten Kreis, Sanja Fidler, Ming-Yu Liu, and Tsung-Yi Lin. Magic3D: High-resolution text-to-3d content creation. In *CVPR*, 2023. 3

[61] Ruoshi Liu, Rundi Wu, Basile Van Hoorick, Pavel Tokmakov, Sergey Zakharov, and Carl Vondrick. Zero-1-to-3: Zero-shot one image to 3d object. In *ICCV*, 2023. 1, 3, 6, 9, 18, 19

[62] Yuan Liu, Cheng Lin, Zijiao Zeng, Xiaoxiao Long, Lingjie Liu, Taku Komura, and Wenping Wang. SyncDreamer: Generating multiview-consistent images from a single-view image. In *ICLR*, 2024. 3, 9

[63] Francesco Locatello, Dirk Weissenborn, Thomas Unterthiner, Aravindh Mahendran, Georg Heigold, Jakob Uszkoreit, Alexey Dosovitskiy, and Thomas Kipf. Object-centric learning with slot attention. *NeurIPS*, 2020. 3, 4

[64] Grace Luo, Trevor Darrell, Oliver Wang, Dan B Goldman, and Aleksander Holynski. Readout Guidance: Learning control from diffusion features. In *CVPR*, 2024. 2

[65] Jian Ma, Junhao Liang, Chen Chen, and Haonan Lu. Subject-Diffusion: Open domain personalized text-to-image generation without test-time fine-tuning. In *ACM SIGGRAPH Conference Proceedings*, 2024. 3, 4

[66] Luke Melas-Kyriazi, Iro Laina, Christian Rupprecht, and Andrea Vedaldi. RealFusion: 360deg reconstruction of any object from a single image. In *CVPR*, 2023. 3

[67] Oscar Michel, Anand Bhattad, Eli VanderBilt, Ranjay Krishna, Aniruddha Kembhavi, and Tanmay Gupta. OBJECT 3DIT: Language-guided 3d-aware image editing. *NeurIPS*, 2023. 1, 3, 6, 7, 18, 19, 20

[68] B Mildenhall, PP Srinivasan, M Tancik, JT Barron, R Ramamoorthi, and R Ng. NeRF: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020. 4

[69] Chong Mou, Xintao Wang, Jiechong Song, Ying Shan, and Jian Zhang. DragonDiffusion: Enabling drag-style manipulation on diffusion models. In *ICLR*, 2024. 2

[70] Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *ICML*, 2021. 1, 2

[71] Michael Niemeyer and Andreas Geiger. GIRAFFE: Representing scenes as compositional generative neural feature fields. In *CVPR*, 2021. 3, 5

[72] Karran Pandey, Paul Guerrero, Matheus Gadelha, Yannick Hold-Geoffroy, Karan Singh, and Niloy Mitra. Diffusion Handles: Enabling 3d edits for diffusion models by lifting activations to 3d. In *CVPR*, 2024. 3, 18

[73] Dario Pavllo, Graham Spinks, Thomas Hofmann, Marie-Francine Moens, and Aurelien Lucchi. Convolutional generation of textured 3d meshes. *NeurIPS*, 2020. 3

[74] Dario Pavllo, Jonas Kohler, Thomas Hofmann, and Aurelien Lucchi. Learning generative models of textured 3d meshes from real-world images. In *ICCV*, 2021. 3

[75] Ben Poole, Ajay Jain, Jonathan T Barron, and Ben Mildenhall. DreamFusion: Text-to-3d using 2d diffusion. In *ICLR*, 2023. 3

[76] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *ICML*, 2021. 9

[77] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 2022. 1

[78] Helge Rhodin, Mathieu Salzmann, and Pascal Fua. Unsupervised geometry-aware representation for 3d human pose estimation. In *ECCV*, 2018. 5

[79] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *CVPR*, 2022. 1, 2, 3, 5, 6, 19

[80] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-Net: Convolutional networks for biomedical image segmentation. In *MICCAI*, 2015. 22

[81] Nataniel Ruiz, Yuanzhen Li, Varun Jampani, Yael Pritch, Michael Rubinstein, and Kfir Aberman. DreamBooth: Fine tuning text-to-image diffusion models for subject-driven generation. In *CVPR*, 2023. 1, 3, 7, 19

[82] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily Denton, Seyed Kamyar Seyed Ghasemipour, Burcu Karagol Ayan, S Sara Mahdavi, Rapha Gontijo Lopes, et al. Photorealistic text-to-image diffusion models with deep language understanding. In *NeurIPS*, 2022. 1, 2

[83] Mehdi SM Sajjadi, Daniel Duckworth, Aravindh Mahendran, Sjoerd van Steenkiste, Filip Pavetic, Mario Lucic, Leonidas J Guibas, Klaus Greff, and Thomas Kipf. Object scene representation transformer. *NeurIPS*, 2022. 3

[84] Kyle Sargent, Zizhang Li, Tanmay Shah, Charles Herrmann, Hong-Xing Yu, Yunzhi Zhang, Eric Ryan Chan, Dmitry Lagun, Li Fei-Fei, Deqing Sun, et al. ZeroNVS: Zero-shot 360-degree view synthesis from a single real image. In *CVPR*, 2024. 3

[85] Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, et al. LAION-5B: An open large-scale dataset for training next generation image-text models. *NeurIPS Datasets and Benchmarks Track*, 2022. 1

[86] Katja Schwarz, Yiyi Liao, Michael Niemeyer, and Andreas Geiger. GRAF: Generative radiance fields for 3d-aware image synthesis. *NeurIPS*, 2020. 3

[87] Maximilian Seitzer, Sjoerd van Steenkiste, Thomas Kipf, Klaus Greff, and Mehdi SM Sajjadi. DyST: Towards dynamic neural scene representations on real-world videos. In *ICLR*, 2024. 3

[88] Jing Shi, Wei Xiong, Zhe Lin, and Hyun Joon Jung. InstantBooth: Personalized text-to-image generation without test-time finetuning. *arXiv preprint arXiv:2304.03411*, 2023. 3

[89] Yichun Shi, Peng Wang, Jianglong Ye, Long Mai, Kejie Li, and Xiao Yang. MVDream: Multi-view diffusion for 3d generation. In *ICLR*, 2024. 3, 19

[90] Yujun Shi, Chuhui Xue, Jiachun Pan, Wenqing Zhang, Vincent YF Tan, and Song Bai. DragDiffusion: Harnessing diffusion models for interactive point-based image editing. In *CVPR*, 2024. 2

[91] Gautam Singh, Fei Deng, and Sungjin Ahn. Illiterate dall-e learns to compose. In *ICLR*, 2021. 3

[92] Gautam Singh, Yi-Fu Wu, and Sungjin Ahn. Simple unsupervised object-centric learning for complex and naturalistic videos. *NeurIPS*, 2022. 3

 https://doi.org/10.52202/079017-2429

[93] Gautam Singh, Yeongbin Kim, and Sungjin Ahn. Neural systematic binder. In *ICLR*, 2023. 3

[94] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *ICML*, 2015. 2, 22

[95] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *ICLR*, 2020. 7, 19

[96] Roger Wolcott Sperry. Neural basis of the spontaneous optokinetic response produced by visual inversion. *Journal of Comparative and Physiological Psychology*, 1950. 2

[97] Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, et al. Scalability in perception for autonomous driving: Waymo open dataset. In *CVPR*, 2020. 6, 18

[98] Haochen Wang, Xiaodan Du, Jiahao Li, Raymond A Yeh, and Greg Shakhnarovich. Score Jacobian Chaining: Lifting pretrained 2d diffusion models for 3d generation. In *CVPR*, 2023. 3

[99] Jiawei Wang, Yuchen Zhang, Jiaxin Zou, Yan Zeng, Guoqiang Wei, Liping Yuan, and Hang Li. Boximator: Generating rich and controllable motions for video synthesis. *arXiv preprint arXiv:2402.01566*, 2024. 2

[100] Qian Wang, Yiqun Wang, Michael Birsak, and Peter Wonka. BlobGAN-3D: A spatially-disentangled 3d-aware generative model for indoor scenes. *arXiv preprint arXiv:2303.14706*, 2023. 3

[101] Qixun Wang, Xu Bai, Haofan Wang, Zekui Qin, and Anthony Chen. InstantID: Zero-shot identity-preserving generation in seconds. *arXiv preprint arXiv:2401.07519*, 2024. 3

[102] Tai Wang, Xinge Zhu, Jiangmiao Pang, and Dahua Lin. FCOS3D: Fully convolutional one-stage monocular 3d object detection. In *ICCV Workshop*, 2021. 5

[103] Xudong Wang, Trevor Darrell, Sai Saketh Rambhatla, Rohit Girdhar, and Ishan Misra. InstanceDiffusion: Instance-level control for image generation. In *CVPR*, 2024. 1, 2, 4

[104] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image Quality Assessment: from error visibility to structural similarity. *TIP*, 2004. 7, 19

[105] Daniel Watson, William Chan, Ricardo Martin Brualla, Jonathan Ho, Andrea Tagliasacchi, and Mohammad Norouzi. Novel view synthesis with diffusion models. In *ICLR*, 2023. 3

[106] Yuxiang Wei, Yabo Zhang, Zhilong Ji, Jinfeng Bai, Lei Zhang, and Wangmeng Zuo. ELITE: Encoding visual concepts into textual embeddings for customized text-to-image generation. In *ICCV*, 2023. 3

[107] Olivia Wiles, A Koepke, and Andrew Zisserman. X2Face: A network for controlling face generation using images, audio, and pose codes. In *ECCV*, 2018. 2

[108] Ziyi Wu, Jingyu Hu, Wuyue Lu, Igor Gilitschenski, and Animesh Garg. SlotDiffusion: Object-centric generative modeling with diffusion models. *NeurIPS*, 2023. 3

[109] Guangxuan Xiao, Tianwei Yin, William T Freeman, Frédo Durand, and Song Han. FastComposer: Tuning-free multi-subject image generation with localized attention. *arXiv preprint arXiv:2305.10431*, 2023. 3, 4

[110] Jinheng Xie, Yuexiang Li, Yawen Huang, Haozhe Liu, Wentian Zhang, Yefeng Zheng, and Mike Zheng Shou. BoxDiff: Text-to-image synthesis with training-free box-constrained diffusion. In *ICCV*, 2023. 2

[111] Yinghao Xu, Menglei Chai, Zifan Shi, Sida Peng, Ivan Skorokhodov, Aliaksandr Siarohin, Ceyuan Yang, Yujun Shen, Hsin-Ying Lee, Bolei Zhou, et al. DiscoScene: Spatially disentangled generative radiance fields for controllable 3d-aware scene synthesis. In *CVPR*, 2023. 3, 5, 6, 18

[112] Zhongcong Xu, Jianfeng Zhang, Jun Hao Liew, Hanshu Yan, Jia-Wei Liu, Chenxu Zhang, Jiashi Feng, and Mike Zheng Shou. MagicAnimate: Temporally consistent human image animation using diffusion model. In *CVPR*, 2024. 2

[113] Lihe Yang, Bingyi Kang, Zilong Huang, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth Anything: Unleashing the power of large-scale unlabeled data. In *CVPR*, 2024. 5, 10

[114] Zhengyuan Yang, Jianfeng Wang, Zhe Gan, Linjie Li, Kevin Lin, Chenfei Wu, Nan Duan, Zicheng Liu, Ce Liu, Michael Zeng, et al. ReCo: Region-controlled text-to-image generation. In *CVPR*, 2023. 2

[115] Jiraphon Yenphraphai, Xichen Pan, Sainan Liu, Daniele Panozzo, and Saining Xie. Image Sculpting: Precise object editing with 3d geometry control. In *CVPR*, 2024. 3

[116] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. In *ICCV*, 2023. 1, 2

[117] 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. 7, 19

[118] Yi Zhou, Connelly Barnes, Jingwan Lu, Jimei Yang, and Hao Li. On the continuity of rotation representations in neural networks. In *CVPR*, 2019. 4

 https://doi.org/10.52202/079017-2429

A Detailed Experimental Setup

In this section, we provide full details on the datasets, baselines, evaluation settings, and the training and inference implementation of our model.

A.1 Datasets

OBJect [67] consists of Blender [20] rendered scenes where multiple (up to four) objects are placed on a flat textured ground. The objects come from a 59k subset of Objaverse dataset [21]. A total of 18 background maps are used to provide environmental lighting. Four types of object-level editing are provided – translation, rotation, removal, and insertion, each with 100k simulated data. Notably, only one object is edited in each data, and the translation and rotation is always on the ground (i.e., perpendicular to the gravity vector). For a fair comparison with the 3DIT baseline [67], we use the 2D rotated bounding box to represent object pose, which is composed of two corners of the 2D bounding box and the rotation angle over the gravity axis. This dataset is under the Open Data Commons Attribution License (ODC-By)[2].

MOVi-E [36] contains 10k videos simulated using Kubric [36]. Each scene contains 11 to 23 real-world objects from the Google Scanned Objects (GSO) repository [23]. At the start of each video, several objects are thrown to the ground to collide with other objects. Similar to OBJect, environmental lighting is provided by a randomly sampled environment map image. The camera follows a small linear motion. The full data generation pipeline is under the Apache 2.0 license[3].

Objectron [1] contains 15k object-centric video clips of common daily objects covering nine categories. Each video comes with object pose tracking throughout the video, and we process it to obtain 3D bounding boxes. Since this dataset does not provide 2D bounding box labels, we project the eight corners of 3D boxes to the image, and take the tight bounding box of projected points as 2D boxes. Objectron is licensed under the Computational Use of Data Agreement 1.0 (C-UDA-1.0)[4].

Waymo Open [97]. The Waymo Open Dataset consists of 1k videos of self-driving scenes recorded by car mounted cameras. Following prior works [24, 111], we take the front view camera and bounding box annotations of cars. Notably, the 3D bounding boxes only have a heading angle (rotation along the yaw-axis) annotation, and thus we treat the other two rotation angles as 0. Besides, the provided 2D boxes and 3D boxes are not aligned, preventing us from doing paired frame training. We instead project 3D boxes to get associated 2D boxes similar to on Objectron. Waymo Open is licensed under the Waymo Dataset License Agreement for Non-Commercial Use (August 2019)[5].

Data Pre-processing. For all datasets, we resize the images to 256×256 regardless of the original aspect ratio. On Objectron, we discard all videos from the bike class as it contains many blurry frames and inaccurate 3D bounding box annotations. On Waymo, we remove all cars whose 2D bounding box is smaller than 1% of the image area. We do not apply data augmentation except on Waymo Open, where we apply random horizontal flip and random resize crop following [24].

A.2 Baselines

3DIT [67] fine-tunes Zero-1-to-3 [61] to support scene-level 3D object edits. We generate the editing instruction from the target object pose, such as the translation coordinate and the rotation angle. However, this method does not support large viewpoint changes as it does not encode camera poses. We take their official code and pre-trained weights of the Multitask variant. 3DIT is under the CreativeML Open RAIL-M license[6].

Chained. This baseline is inspired by [67, 72], where we chain multiple models together to achieve 3D-aware object editing. An editing step usually contains three steps: **(i)** crop out the object of interest and inpaint its region with backgrounds, **(ii)** synthesize the object under the new pose, and **(iii)** place the object to the new location. For **(i)**, we apply SAM [55] to segment the object using 2D bounding box prompt, and inpaint the original object region with Stable Diffusion v2 inpainting

[2] https://huggingface.co/datasets/allenai/object-edit/blob/main/README.md
[3] https://github.com/google-research/kubric/blob/main/LICENSE
[4] https://github.com/google-research-datasets/Objectron#license
[5] https://waymo.com/open/terms
[6] https://github.com/allenai/object-edit/blob/main/LICENSE

model [79]. For **(ii)**, we run Zero-1-to-3 [61] to re-pose the object according to the target 3D bounding box. For **(iii)**, following [61], we first get the alpha mask of the re-posed object using an online tool[7], and insert it to the new position via alpha blending. It is worth noting that Zero-1-to-3 does not support camera rotation over the roll axis. For all models, we take their official code and pre-trained weights. SAM is under the Apache 2.0 license[8]. Stable Diffusion v2 inpainting model is under the CreativeML Open RAIL++-M License[9]. Zero-1-to-3 is under the MIT license[10]. The online alpha mask extraction tool is under the Apache 2.0 license[11].

A.3 Evaluation Settings

We report PSNR, SSIM [104], LPIPS [117], and FID [42] to measure the accuracy of the edited image. We compute metrics both on the entire image, and within the 2D bounding box of edited objects. For box-level metrics, we follow [67] to crop out each object and directly run the metric without resizing. We also evaluate the identity preservation of objects using the DINO [13] feature similarity proposed in [81], which runs a DINO self-supervised pre-trained ViT on cropped object patches to extract features and compute the cosine similarity between predicted and ground-truth image.

A.4 Our Implementation Details

Model architecture. We take Stable Diffusion (SD) v2.1 [79] as our image generator except for experiments on the OBJect dataset, where we use SD v1.5 for a fair comparison with baselines. Similar to prior works [49, 89], we also observe clearly better performance using SD v2.1 compared to v1.5. However, we note that our Neural Assets framework generalizes to any image generator that conditions on a sequence of tokens. We implement the visual encoder Enc with a DINO self-supervised pre-trained ViT-B/8 [13], which outputs a feature map of shape 28×28 given a 256×256 image. For each object, we apply RoIAlign [38] to extract a 2×2 small feature map and flatten it, i.e., the appearance token A_i has a sequence length of $K = 4$. Since the conditioning token dimension of pre-trained SD v2.1 is 1024, we use a two-layer MLP to transform the 3D bounding boxes input to $D' = 1024$, and linearly project the concatenated appearance and pose token back to 1024. For background modeling, we mask all pixels within object boxes by setting them to a fixed value of 0.5, and extract features with the same DINO encoder. Instead, the pose token is obtained by applying a different two-layer MLP on the relative camera pose between the source and the target image.

Training. We implement the entire Neural Assets framework in JAX [10] using the Flax [40] neural network library. We train all model components jointly using the Adam optimizer [53] with a batch size of 1536 on 256 TPUv5 chips (16GB memory each). We use a peak learning rate of 5×10^{-5} for the image generator and the visual encoder, and a larger learning rate of 1×10^{-3} for remaining layers (MLPs and linear projection layers). Both learning rates are linearly warmed up in the first 1,000 steps and stay constant. A gradient clipping of 1.0 is applied to stabilize training. We found that the model overfits more severely on real-world data with complex backgrounds compared to synthetic datasets. Therefore, we train the model for 200k steps on OBJect and MOVi-E which takes 24 hours, and 50k steps on Objectron and Waymo Open which takes 6 hours. In order to apply classifier-free guidance (CFG) [43], we randomly drop the appearance and pose token (i.e., setting them as zeros) with a probability of 10%. CFG improves the performance and also alleviates overfitting in training.

Inference. We run the DDIM sampler [95] for 50 steps to generate images. We found the model works well with CFG scale between 1.5 and 4, and thus choose to use 2.0 in all the experiments.

[7] https://github.com/OPHoperHPO/image-background-remove-tool
[8] https://github.com/facebookresearch/segment-anything#license
[9] https://huggingface.co/stabilityai/stable-diffusion-2-inpainting
[10] https://github.com/cvlab-columbia/zero123/blob/main/LICENSE
[11] https://github.com/OPHoperHPO/image-background-remove-tool/blob/master/LICENSE

https://doi.org/10.52202/079017-2429

Table 1: **Single-object editing results on OBJect**. We evaluate on the *Translation*, *Rotation*, and *Removal* tasks. We follow 3DIT [67] to evaluate on both seen and unseen object subsets, and compute metrics inside the edited object's bounding box. Our results are averaged over 3 random seeds.

	Seen Objects				Unseen Objects			
Model	**PSNR ↑**	**SSIM ↑**	**LPIPS ↓**	**FID ↓**	**PSNR ↑**	**SSIM ↑**	**LPIPS ↓**	**FID ↓**
Task: Translation								
Chained	13.70	0.309	0.485	0.94	14.13	0.326	0.467	0.97
3DIT	15.21	0.300	0.472	0.24	15.20	0.292	0.477	0.25
Ours	**20.58**±0.62	**0.439**±0.013	**0.273**±0.008	**0.15**±0.004	**20.13**±0.81	**0.429**±0.016	**0.274**±0.008	**0.16**±0.006
Task: Rotation								
Chained	13.18	0.269	0.540	1.00	12.85	0.270	0.538	1.69
3DIT	16.86	0.382	0.429	0.25	16.28	0.366	0.447	0.24
Ours	**18.52**±0.35	**0.391**±0.012	**0.354**±0.006	**0.14**±0.002	**18.39**±0.57	**0.377**±0.016	**0.365**±0.009	**0.15**±0.008
Task: Removal								
Chained	12.49	0.383	0.465	0.80	12.12	0.379	0.459	1.05
3DIT	24.98	0.585	0.249	0.24	24.66	0.568	0.260	0.24
Ours	**28.86**±0.88	**0.616**±0.015	**0.167**±0.010	**0.14**±0.007	**28.44**±0.91	**0.613**±0.016	**0.169**±0.010	**0.15**±0.005

Table 2: **Multi-object editing results on MOVi-E, Objectron, and Waymo Open**. We compute metrics on the entire image and inside the object bounding boxes. Ours are averaged over 3 seeds.

	Image-Level				Object-Level			
Model	**PSNR ↑**	**SSIM ↑**	**LPIPS ↓**	**FID ↓**	**PSNR ↑**	**SSIM ↑**	**LPIPS ↓**	**DINO ↑**
Dataset: MOVi-E								
Chained	14.46	0.409	0.481	4.46	15.06	0.303	0.436	0.554
3DIT	14.33	0.385	0.671	5.89	12.41	0.214	0.663	0.336
Ours	**22.03**±0.95	**0.594**±0.015	**0.277**±0.007	**2.20**±0.024	**20.05**±0.43	**0.547**±0.009	**0.289**±0.006	**0.738**±0.017
Dataset: Objectron								
Chained	11.23	0.262	0.586	2.03	11.24	0.171	0.415	0.555
3DIT	11.69	0.281	0.559	1.99	11.30	0.150	0.444	0.547
Ours	**14.83**±0.45	**0.348**±0.012	**0.446**±0.021	**0.55**±0.011	**16.41**±0.33	**0.477**±0.009	**0.233**±0.008	**0.790**±0.015
Dataset: Waymo Open								
Chained	18.28	**0.501**	0.454	2.88	16.20	0.310	0.383	0.596
3DIT	17.32	0.421	0.474	3.32	14.41	0.174	0.537	0.449
Ours	**18.71**±0.50	0.494±0.018	**0.404**±0.010	**1.64**±0.003	**17.67**±0.30	**0.343**±0.011	**0.348**±0.013	**0.653**±0.019

B Additional Experimental Results

B.1 Full Benchmark Results

We present full quantitative results on OBJect in Tab. 1, and on MOVi-E, Objectron, and Waymo Open in Tab. 2. Compared to the main paper, we report additional FID metrics and results on the unseen object subset for OBJect, while for the other three datasets, we report additional FID and DINO feature similarity metrics, plus results computed over the entire image (Image-Level). Overall, we observe similar trends as in the main paper, where our Neural Assets model significantly outperforms baselines across all datasets. In Fig. 11, we show additional qualitative comparisons.

B.2 Full Ablation Results

We present all quantitative results of our ablation studies on Objectron (Sec. 4.4) in Tab. 3, Tab. 4, and Tab. 5. We observe similar trends on all metrics at both image- and object-level.

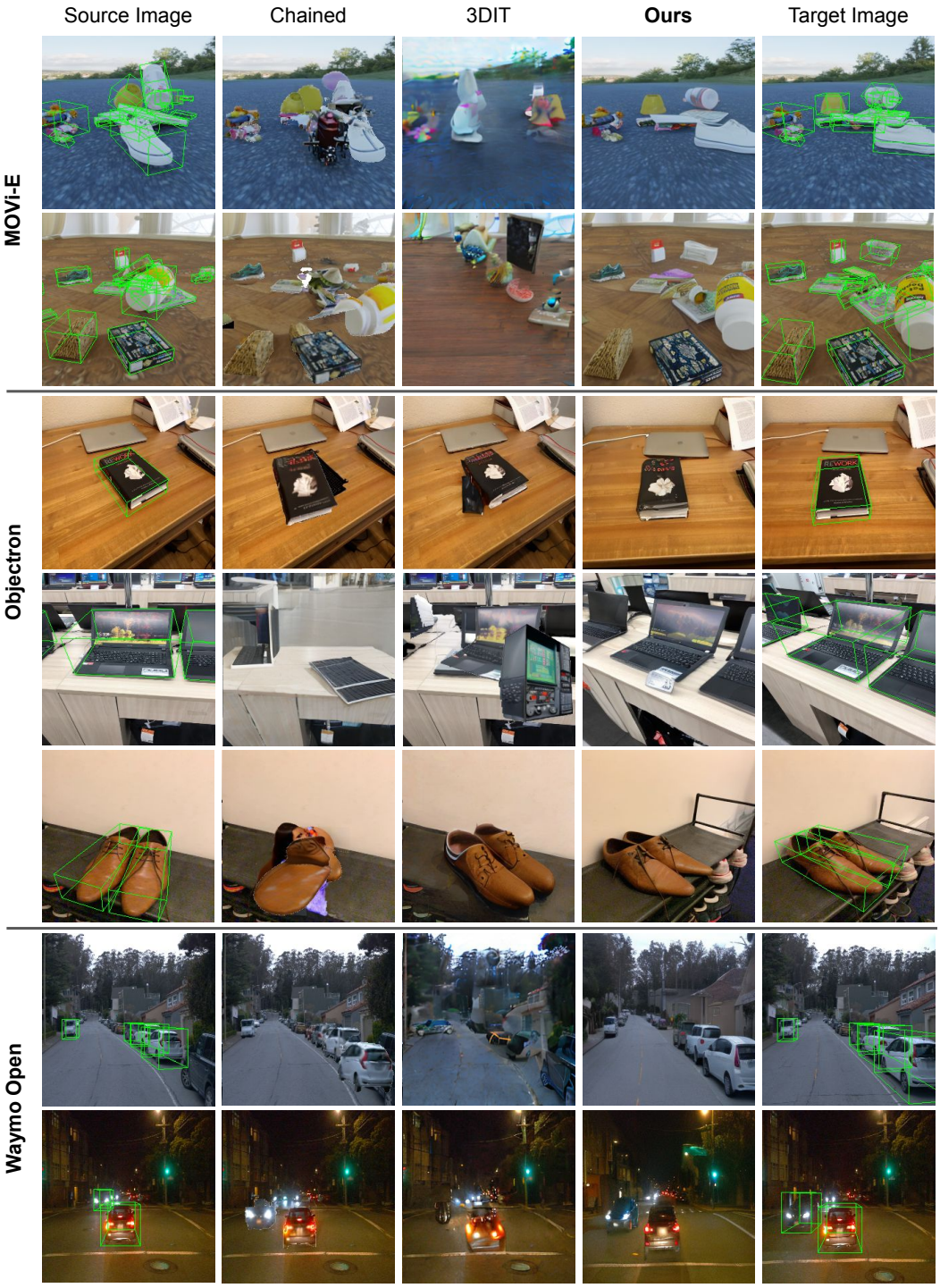


Figure 11: **More qualitative results on MOVi-E, Objectron, and Waymo Open**.

B.3 Controllable Scene Generation

In Fig. 12 and Fig. 13, we show controllable scene generation results on Objectron. Objectron videos only have global camera movement, while the objects are static. Still, our Neural Assets model learns disentangled foreground and background representations. As can be seen from the results, we can rotate the foreground objects while keeping the background fixed, or swap background between

https://doi.org/10.52202/079017-2429

Table 3: **Ablation of image encoders on Objectron**. FT-DINO stands for fine-tuning DINO ViT.

	Image-Level				Object-Level			
Encoder	**PSNR ↑**	**SSIM ↑**	**LPIPS ↓**	**FID ↓**	**PSNR ↑**	**SSIM ↑**	**LPIPS ↓**	**DINO ↑**
CLIP	12.95	0.309	0.532	0.66	14.13	0.339	0.333	0.709
MAE	13.64	0.317	0.501	0.59	14.93	0.369	0.296	0.735
DINO	13.75	0.325	0.498	0.57	15.03	0.388	0.296	0.747
FT-FINO	**14.83**	**0.348**	**0.446**	**0.55**	**16.41**	**0.477**	**0.233**	**0.790**

Table 4: **Ablation of background modeling on Objectron**. No-BG means not doing background modeling at all, while No-Pose stands for not using the relative camera pose between two frames.

	Image-Level				Object-Level			
Background	**PSNR ↑**	**SSIM ↑**	**LPIPS ↓**	**FID ↓**	**PSNR ↑**	**SSIM ↑**	**LPIPS ↓**	**DINO ↑**
No-BG	12.98	0.308	0.513	1.18	14.49	0.394	0.297	0.712
No-Pose	13.71	0.326	0.496	0.72	15.39	0.423	0.273	0.751
Ours	**14.83**	**0.348**	**0.446**	**0.55**	**16.41**	**0.477**	**0.233**	**0.790**

Table 5: **Ablation of training data on Objectron**. Single and Paired refer to training on one image or source-target pairs. No-PE means removing the positional encoding in the ViT image encoder.

	Image-Level				Object-Level			
Training Data	**PSNR ↑**	**SSIM ↑**	**LPIPS ↓**	**FID ↓**	**PSNR ↑**	**SSIM ↑**	**LPIPS ↓**	**DINO ↑**
Single	12.63	0.298	0.544	1.07	13.41	0.259	0.381	0.651
Single (No-PE)	13.74	0.323	0.503	1.01	14.51	0.315	0.337	0.683
Paired	**14.83**	**0.348**	**0.446**	**0.55**	**16.41**	**0.477**	**0.233**	**0.790**

scenes. Importantly, our model inpaints the masked background regions not occupied by the novel object, and renders realistic shadows around the object, which is far beyond simple pixel copying.

B.4 Ablation on 3D Pose Representations

In Fig. 14, we visualize the object pose representation we use. Given a 3D bounding box of an object, we project its four corners to the image space, and concatenate their 2D coordinates and depth values to obtain a 12-D pose vector. The 2D projected points resemble a local coordinate frame for the object, specifying its position, rotation, and scale. On the other hand, the depth is useful for determining the occlusion of objects.

There are alternative ways to represent the object pose, e.g., the coordinate of the 3D box center C with its size and rotation which is commonly used in 3D object detection [57]. These representations achieve similar results on MOVi-E and Objectron. However, their learned rotation controllability is significantly worse than our representation on Waymo. This is because most of the cars on Waymo are not rotated (turn left / right), leading to very few training data on object rotation. If we directly input the rotation angle to the model, it tends to ignore it. In contrast, due to prospective projection, the projected local coordinate frame of unrotated cars still look "rotated" when they are not strictly in front of the ego vehicle. This provides much more training signal to learn the rotation of objects.

C Background on Stable Diffusion

Diffusion model [44, 94] is a class of generative models that learns to generate samples by iteratively denoising from a standard Gaussian distribution. It consists of a denoiser ϵ_θ, usually implemented as a U-Net [80], which predicts the noise ϵ added to the data x. Instead of denoising raw pixels, Stable Diffusion introduces a VAE [54] tokenizer to map images to low-dimensional latent code z and applies the denoiser on it. In addition, the denoiser is conditioned on text and thus supports text-to-image generation. In this work, we simply replace the text embeddings with Neural Assets a_i and fine-tune the model to support appearance and pose control of 3D objects.

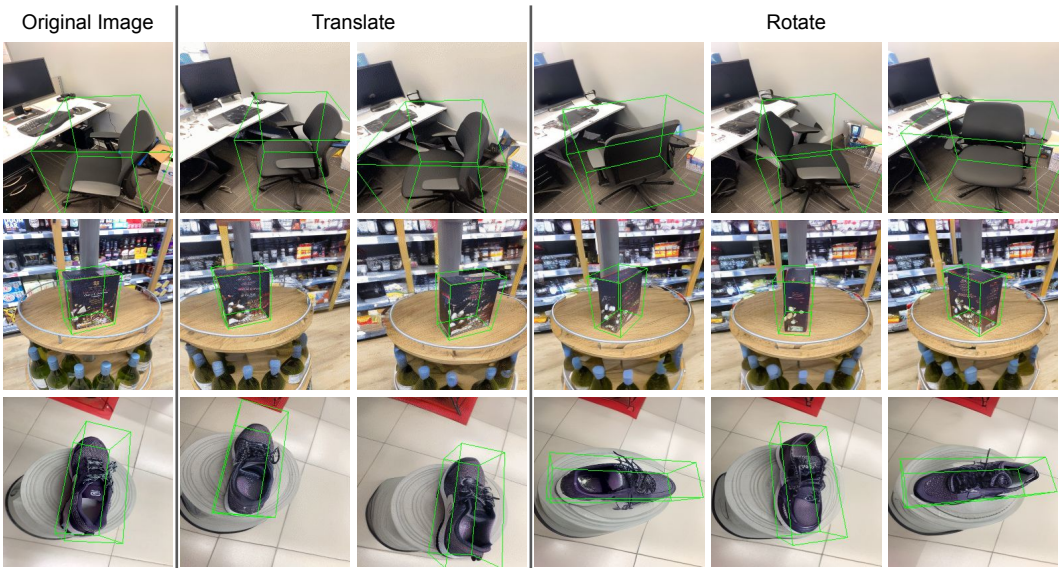


Figure 12: **Object translation and rotation results** on Objectron. Although there is only camera movement on this dataset (i.e., objects never move), the model still learns to disentangle the object pose and the camera pose. As shown in the object rotation results, the background stays fixed. See our project page for videos and additional object rescaling results.

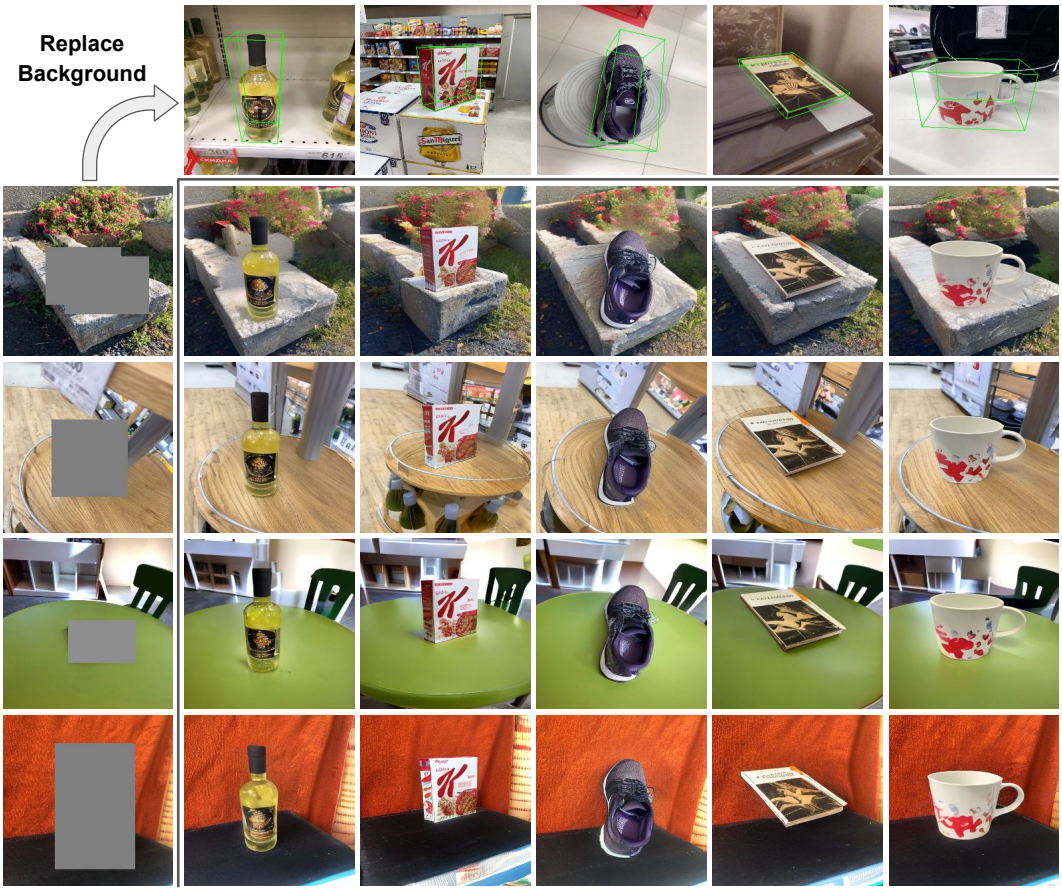
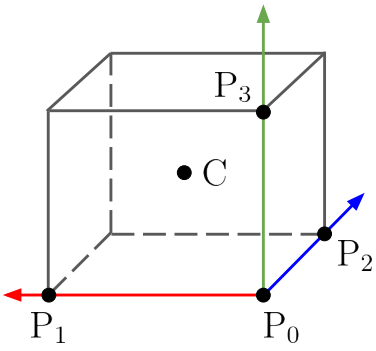
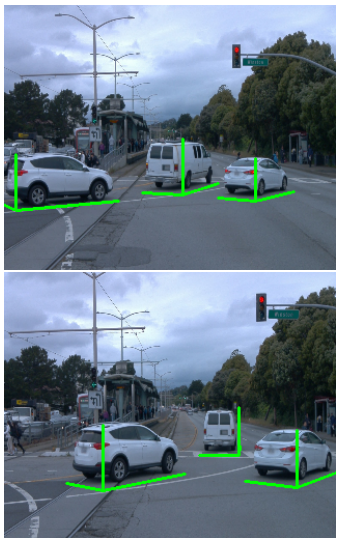


Figure 13: **Transfer backgrounds between scenes** by replacing the background token on Objectron. Note how the global camera viewpoint is adjusted to fit the foreground object. In addition, the generator is able to synthetic lighting effects such as shadows on the surfaces.

https://doi.org/10.52202/079017-2429



(a) Object pose as 3D bounding box



(b) Sample pose representation 1

(c) Sample pose representation 2

Figure 14: **Illustration of our object pose representation (a) and two examples (b)**. We project four corners $\mathrm{P}_0, \mathrm{P}_1, \mathrm{P}_2, \mathrm{P}_3$ of a 3D bounding box to the 2D image plane and concatenate them to obtain the pose token. The projected four corners form a local coordinate system of the object.

D Broader Impacts

Controllable visual generation is an important task in computer vision. Neural Assets equip generative models with an intuitive interface to control their behaviors, which enables more interpretable AI algorithms and may potentially benefit other fields such as computer graphics and robotics. We believe this work will benefit the whole research community and the society.

Potential negative societal impacts. Since we fine-tune large-scale pre-trained generative models in our pipeline, we inherit limitations of these base models, such as dataset selection bias. Such bias might be problematic when human subjects are involved, though our current approach is only capable of rigid object control and does not consider humans as an "asset" yet. Further study on how such bias affects model performance is required for mitigating negative societal impacts that could arise from this work [35].

E Funding Disclosure

This work was carried out at Google. Igor Gilitschenski contributed to the project in an advisory capacity.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: We propose Neural Assets, a new form of representations that supports multi-object controllable 3D scene generation. We have verified it with extensive experiments in Sec. 4 and Appendix B.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discuss the limitations of this work in Sec. 5.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

https://doi.org/10.52202/079017-2429

Answer: [NA]

Justification: This paper does not include theoretical results.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We provide all implementation details about dataset processing, model architecture, and training and inference in Appendix A.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: We provide all the details about our implementations in Appendix A.1 and Appendix A.4. See our project page, neural-assets.github.io, for further details.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [Yes]

Justification: We provide all implementation details about dataset processing, model architecture, evaluation metrics, and training and inference in Appendix A.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: All main results are averaged over 3 random seeds and we report error bars.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

https://doi.org/10.52202/079017-2429

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We provide all required information in Appendix A.4.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

   Answer: [Yes]

   Justification: Our research conforms with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: We discuss potential positive and negative society impacts in Appendix D.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [Yes]

    Justification: We discuss safeguards on our image generator in Appendix D.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We provide the license information in Appendix A.1 and Appendix A.2.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

https://doi.org/10.52202/079017-2429

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: No new asset is introduced in the submission.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: This paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: This paper does not involve research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.