DreamHOI: Subject-Driven Generation of 3D Human-Object Interactions with Diffusion Priors

Supplementary Material

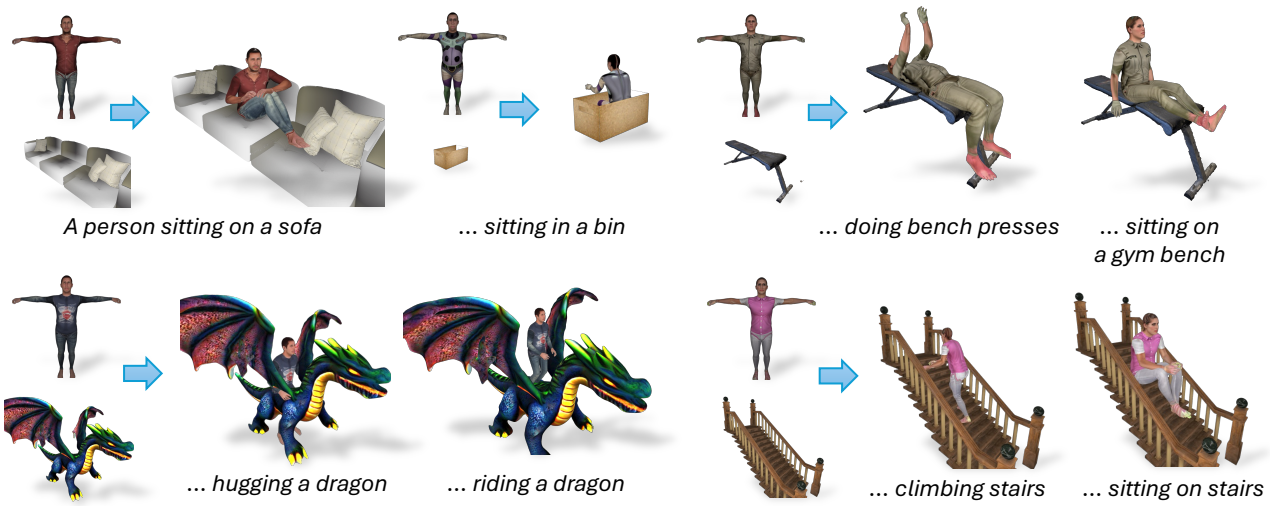

Figure 7. **Additional Results.** DreamHOI is able to generate HOIs for diverse objects and corresponding prompts.

A. Additional Results

Additional qualitative results. Additional results on a variety of prompts and objects can be found in Fig. 7. DreamHOI is capable of realistically deforming the human pose to interact with these objects faithfully to the corresponding textual prompts.

Failure cases. We show some cases where DreamHOI failed in Fig. 8. From a manual inspection, in most cases this was due to the underlying diffusion model not understanding the semantic composition, because it was too complex, vague, or exotic (respectively, in Fig. 8). In other cases this was due to the pose prediction (SMPLify-X [34]) not working properly. Therefore we believe an improvement to either would improve DreamHOI's ability to generate realistic HOIs.

Additional comparisons. In our baseline comparisons (Sec. 4.4), we considered comparing to the case where we generate a NeRF by DreamFusion using DeepFloyd IF guidance, with M_{obj} inserted. We showed that, although mostly related to the prompt, the outputs often had problems like not view-consistent or being too large. Furthermore, the output cannot preserve the identity of the human. We now compare against more recent methods MVDream [42] and ProlificDreamer [56], in Fig. 9. We find

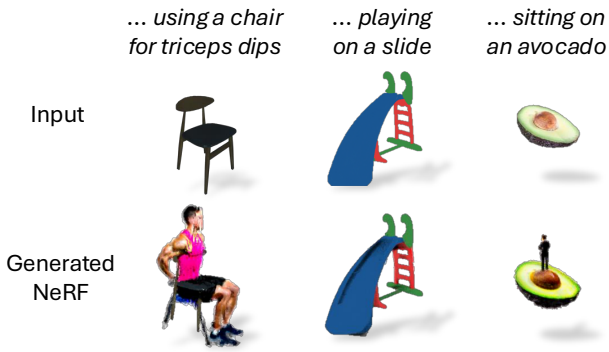

Figure 8. **Failure Cases.** We visualize failed outputs from the first-round NeRF optimization (*i.e.*, f_{θ_0}). Our pipeline is unlikely to recover if the NeRF from the initial round fails to capture the approximate spatial relationship between the human and the object.

that MVDream guidance is able to produce very detailed and view-consistent NeRFs, but fails to understand the basic compositional relations in the prompt (*e.g.*, "sit"). This suggests that models such as MVDream cannot encode rich semantic relations, and provides motivation to use DeepFloyd IF as our base model for better textual understanding. ProlificDreamer fails to generate any meaningful human,

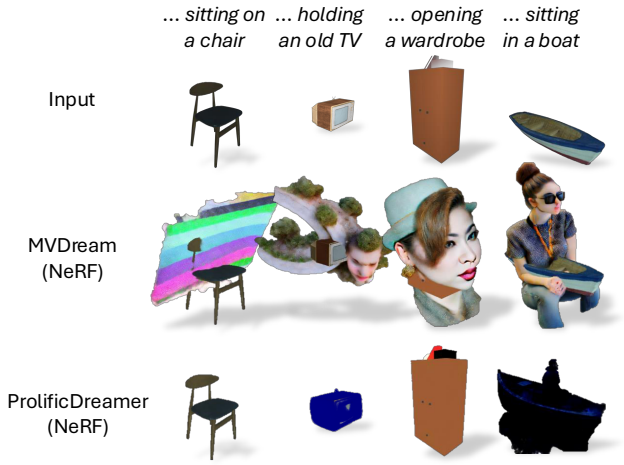

Figure 9. **Additional Comparison.** Following Fig. 4, we additionally compare DreamHOI to MVDream [42] and ProlificDreamer [56] baselines.

Figure 10. **Visualization** of SDS gradient. Left: rendered $\boldsymbol{x}_{\text{HO}}$ (M_ξ with M_{Obj}); Middle: gradient $\nabla_{\boldsymbol{x}}\mathcal{L}_{\text{SDS-HO}}$; Right: norm of the gradient.

suggesting its inability to interact with an inserted object mesh.

We would like to compare with methods specifically for human-object interaction generation, *e.g.* InterFusion [6] and HOI-Diff [35]. Unfortunately, their evaluation code and model are either unavailable or not yet open-source at the time of writing. Qualitatively, we re-iterate that none of the existing methods take a human identity as an input, so they do not achieve subject-driven HOI synthesis even if the output is realistic. InterFusion [6] offers no control over either the human identity or the object (except through the text prompt), while our method takes the human identity and M_{obj} as inputs, and they are preserved in the output. Other HOI generation methods such as HOI-Diff [7, 35] are trained on MoCap datasets such as BEHAVE [1], and can only handle a fixed number of objects. On the other hand, our method can perform zero-shot open-world generation (*e.g.*, riding a dragon, Fig. 7), thanks to priors given by 2D diffusion models. An overview of existing methods is in Sec. 2.

B. Discussions

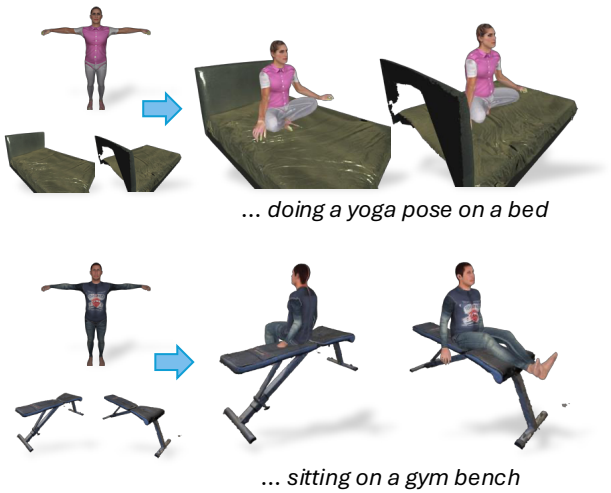

Figure 11. **Front Direction Control.** MVDream [42] guidance enables us to give an implicit "front" direction: the generated humans consistently face the $\pm x$ direction regardless of the orientation of the object.

Failure analysis of direct optimization. The most straightforward way to solve the subject-driven HOI generation task we propose is to only use explicit SMPL (or any other body model) pose parameters ξ instead of an implicit NeRF f_θ as the object for optimization. In this solution, we differentiably render the resulting SMPL mesh M_ξ with an object mesh M_{obj}, and use SDS to directly optimize ξ. This avoids the problem of translation between explicit and implicit forms and makes optimization much faster (*e.g.* SMPL only has 69 pose parameters [27]).

However, in our extensive tests, this method does not work at all. Even if we initialize from a near-optimal pose, ξ regresses to a nonsensical pose as in Fig. 4. Additional changes like making vertex colors and global position learnable do not help either. This was observed in Sec. 4.4.

To illustrate the reason, we monitor the SDS loss and gradient during the optimization of ξ in Fig. 10, for the prompt "a person sitting on a chair". In the middle and right panels, the guidance tries to add legs to the position where legs would be expected, on the seat and to its front. However, there is no way for this gradient to add legs to be propagated to ξ, because ξ can only receive gradients on pixels that the rendered M_ξ occupies. In other words, the tendency of diffusion models to add and delete limbs globally, instead of gradually moving a limb, means that SDS is not suitable for optimizing ξ directly. This necessitates the dual implicit-explicit optimization we propose.

MVDream front direction. We claimed in Sec. 4.5 that MVDream takes camera positions as input and based on bias in its training data, MVDream gives a prior of a "front"

direction (in $+x$ direction) for generating the HOI. We demonstrate this in Fig. 11. This gives us the ability to control the forward face of the entire HOI (typically the direction the person is facing, or its opposite) with respect to the object by rotating the mesh M_{Obj} in the generation.

C. Technical Details

C.1. Regularizers

The *sparsity above threshold regularizer* $\mathcal{R}_{\text{SA}}$ is defined as follows: after rendering the human-only 2D image $\boldsymbol{x}_{\text{H}} \in \mathbb{R}^{H\times W\times 4}$ where RGBA pixel colors are obtained by rendering the implicit representation of a human as NeRF f_θ as in Sec. 3.1, we compute its average density by $\bar{\boldsymbol{x}}_{\text{H}} = \frac{1}{HW}\sum_{i=1}^{H}\sum_{j=1}^{W}(\boldsymbol{x}_{\text{H}})_{i,j,4}$ where 4 means the alpha channel (computed from the sum of w_i in Eq. (1)). This approximates the "size" of a human, as rendered from a particular view. To ensure robustness, we adjust the camera distance based on its randomly sampled field of view, ensuring that the 3D unit cube consistently occupies the same area in the renderings regardless of the focal length. We describe sampling the camera position and this adjustment in Appendix C.2. We would like the size to not exceed a certain threshold $\eta = 20\%$ of the image, by minimizing the regularizer

$$\mathcal{R}_{\text{SA}} \coloneqq \text{softplus}(\bar{\boldsymbol{x}}_{\text{H}} - \eta). \tag{5}$$

The *intersection regularizer* $\mathcal{R}_{\text{I}}$ computes the average density τ (as predicted by f_θ) of all ray points $\boldsymbol{\mu}$ *inside* the object mesh M_{Obj}. Informally, it measures the volume of intersection between the human NeRF and the object. $\mathcal{R}_{\text{I}}$ discourages the model from generating body parts or other objects inside M_{Obj}, which would be invisible in $\boldsymbol{x}_{\text{HO}}$. To compute this, we first sample 128^3 points P inside the bounding box of M_{Obj} and use a mesh occupancy test to determine if they are inside M_{Obj}. Let $\boldsymbol{\mu}_{i\boldsymbol{u}}$ be the ith point along the ray cast from pixel $\boldsymbol{u}$, as in Sec. 3.1. We determine that it is inside M_{Obj} if its nearest point in P is in M_{Obj}. This avoids using an expensive occupancy test for every point in each training iteration. Let M be all points $\boldsymbol{\mu}_{i\boldsymbol{u}}$, and we compute the average density of the NeRF inside M_{Obj} as

$$\mathcal{R}_{\text{I}} \coloneqq \frac{1}{|M|}\sum_{\boldsymbol{\mu}\in M}\tau_\theta(\boldsymbol{\mu})\mathbb{1}_{\{\boldsymbol{\mu}\text{ is inside }M_{\text{obj}}\}}, \tag{6}$$

where $\tau_\theta(\boldsymbol{\mu})$ is the volume density at $\boldsymbol{\mu}$ predicted by f_θ.

C.2. Optimization

We follow MVDream [42] and optimize the initial NeRF in 2 stages. In each stage, we optimize the NeRF with AdamW optimizer (learning rate and weight decay are both set to 0.01) for 5000 steps. We render 64×64 and 256×256 images and use batch size 8 and 4 respectively in two stages. After NeRF re-initialization, MVDream guidance is no longer used, and we increase the rendering resolution to 512×512, reduce the batch size to 1, and decrease the learning rate to 0.001.

The field of view f of the camera in each optimization step is sampled uniformly at random within $[15°, 60°]$, and the camera distance to origin is set to $D/\tan(f/2)$ where the denominator is such that a unit volume in the 3D space corresponds roughly to a fixed area in the 2D space, for $\mathcal{R}_{\text{SA}}$ to work properly as in Appendix C.1, and $D \sim [0.8, 1.0]$ is a perturbation. The elevation angle is sampled uniformly from $[0°, 30°]$. Although not done in this work, we recommend lowering it to $[-30°, 30°]$ if parts of the human may be below the object for better supervision. For rendering views $\boldsymbol{x}_i$ of the NeRF for pose estimation (Sec. 4.2), we use an array of cameras with distance 3 to the origin, elevated at $40°$.

The NeRF representing the human is constrained in a ball of radius 1. We initialize it to be at the origin. The number of parameters of the NeRF MLP (f_θ) is about 12.6 million.

The background color is learned, with a lower learning rate 0.001, and is replaced during training with a random color with probability 0.5 (increased to 1 after re-initialization) in training for augmentation.

The NeRF renderer uses one ray per 2D pixel and 512 samples per ray. We do not use shading for NeRF [37] as it is costly to compute.

C.3. Guidance

For SDS, we use classifier-free guidance [13] with guidance weight set to $\omega = 50$. We include the *negative* prompt "missing limbs, missing legs, missing arms" during optimization.