Puppet-Master: Scaling Interactive Video Generation as a Motion Prior for Part-Level Dynamics

Ruining Li Chuanxia Zheng Christian Rupprecht Andrea Vedaldi
Visual Geometry Group, University of Oxford
{ruining, cxzheng, chrisr, vedaldi}@robots.ox.ac.uk

vgg-puppetmaster.github.io

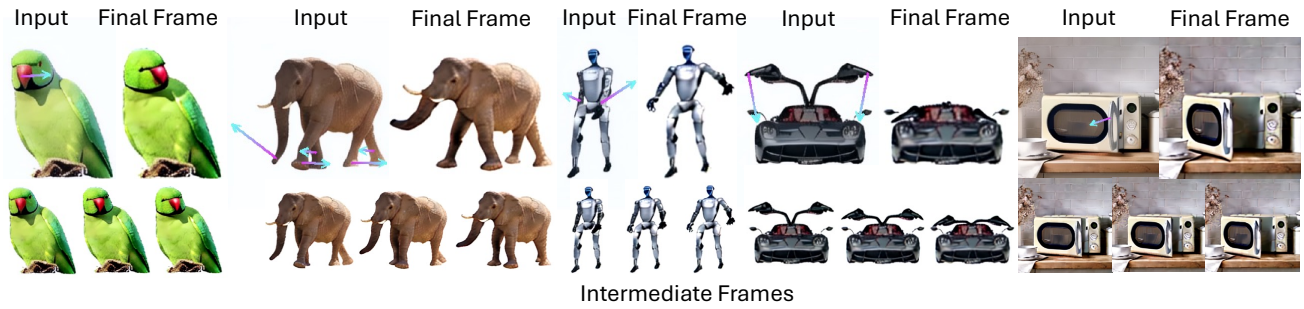

Figure 1. **Puppet-Master** generates videos depicting *internal, part-level* motion, prompted by one or more drags (arrows). Fine-tuned solely on our curated synthetic Objaverse-Animation-HQ dataset, it generalizes well to real-world scenarios and diverse object categories.

Abstract

We introduce Puppet-Master, an interactive video generator that captures the internal, part-level *motion of objects, serving as a proxy for modeling object dynamics universally. Given an image of an object and a set of "drags" specifying the trajectory of a few points on the object, the model synthesizes a video where the object's parts move accordingly. To build Puppet-Master, we extend a pre-trained image-to-video generator to encode the input drags. We also propose* all-to-first *attention, an alternative to conventional spatial attention that mitigates artifacts caused by fine-tuning a video generator on out-of-domain data. The model is fine-tuned on Objaverse-Animation-HQ, a new dataset of curated* part-level *motion clips obtained by rendering synthetic 3D animations. Unlike real videos, these synthetic clips avoid confounding part-level motion with overall object and camera motion. We extensively filter sub-optimal animations and augment the synthetic renderings with meaningful drags that emphasize the internal dynamics of objects. We demonstrate that Puppet-Master learns to generate part-level motions, unlike other motion-conditioned video generators that primarily move the object as a whole. Moreover, Puppet-Master generalizes well to out-of-domain real images, outperforming existing methods on real-world benchmarks in a* zero-shot *manner.*

1. Introduction

In this paper, we introduce Puppet-Master, an *interactive* video generator that predicts how objects move in response to external stimuli. This generator takes as input a single image of an object and a set of sparse *drags*, which specify the motion of selected points on the object. It then outputs a video of the *part-level* object motion consistent with the drags and the object's internal dynamics (Fig. 1).

We are motivated by the need for AI systems to understand how objects can move and deform in general. Researchers have developed countless models of dynamic objects, but most are specific to particular object *types*, such as faces, hands, humans, or quadrupeds [3, 38, 52, 75]. The few more general models [58] do not make strong assumptions about object types but are difficult to train due to the lack of suitable data (*e.g.*, aligned 3D meshes for [58]). None of these are good candidates for learning a "foundation" model of part-level object dynamics. Such a model should be able to express different types of natural object dynamics (Fig. 1), such as part articulation, sliding of parts, and soft deformations.

Recently, video generators trained on millions of videos have been proposed as proxies for "world models" [1, 7, 45]. Any general world model should possess an understanding of object dynamics. However, these models, trained on Internet-scale data, still struggle to capture the

nuances of *internal, part-level* dynamics. Inspired by DragAPart [31], we consider learning a *conditional* video generator that predicts the *part-level* motion of objects in pixel space in response to sparse motion trajectories. This generator takes as input a single image of an object and a set of *drags*, which specify the motion of selected points on the object. It then outputs a video of the *part-level* object motion consistent with the drags (Fig. 1).

Several authors have already explored incorporating drag-like motion prompts in image or video generation [5, 10, 16, 17, 31–33, 36, 41, 42, 44, 54, 63, 67, 70]. Many such works utilize techniques like ControlNet [72] to inject motion control into a pre-trained generator. However, when fine-tuned on real-world videos with motion conditions extracted using off-the-shelf trackers, these models often respond to drags by merely shifting or scaling entire objects, failing to capture their internal dynamics, such as a microwave door rotating shut or a fish oscillating its tail (Fig. 1 and Fig. 5). This limitation can be attributed, in part, to the various confounding components inherent in natural videos, including occlusions, background variations, and camera movements, which complicate motion learning and synthesis. Hence, the challenge is to encourage video generators to synthesize *internal, part-level* dynamics.

In this work, we aim to develop a model capable of generating part-level motion, leveraging *synthetic* data that eliminates the confounding factors present in real-world videos and emphasizes part-level dynamics. We start from a large-scale pre-trained generator, Stable Video Diffusion (SVD) [4], and show how to repurpose it for motion prediction. We make the following **contributions**.

First, we introduce new modules into the video generator for effective motion control and improved appearance generation. In particular, we incorporate *drag tokens* into cross-attention modules for enhanced conditioning. These tokens, regressed from the start and end points of each drag using an encoding function, supplement the *single* image token used in the original SVD, improving spatial awareness in cross-attention. In addition, we introduce *all-to-first* attention, which addresses the degradation in appearance quality that often arises when fine-tuning diffusion generators on out-of-distribution datasets [29, 30, 76]. In our design, *all* frames attend to the first one via a variant of self-attention. This creates a shortcut that directly propagates information from the clean conditioning frame to the others, preventing the model from getting stuck in local optima.

Our second contribution is to provide two datasets to learn part-level object motion. Both datasets comprise subsets of the 40k animated assets in Objaverse [13]. Objaverse animations vary in quality: while some display realistic object dynamics, others feature objects that (i) are static, (ii) exhibit simple translations, rotations, or scaling, or (iii) move in a physically implausible way. We introduce a systematic approach for large-scale animation curation. The resulting datasets, Objaverse-Animation (16k animations) and Objaverse-Animation-HQ (10k animations), contain progressively higher-quality 3D animations. Empirical results show that Objaverse-Animation-HQ, despite its more modest size, yields a superior model compared to Objaverse-Animation, highlighting the effectiveness of our data curation strategy.

With the new curated datasets, we train *Puppet-Master*, our new video generative model that, given a single image of an object and corresponding drags, generates an animation of the object. These animations are faithful to both the input image and the sparse motion trajectories, while exhibiting physically plausible motions at the level of individual object parts (Fig. 1). Our model works across a diverse set of object categories. Empirically, it outperforms prior works on multiple benchmarks. We also present ablations to validate our design choices. Notably, although our Puppet-Master is fine-tuned using only synthetic data, it generalizes well to real data without further tuning.

2. Related Work

Generative models. Recent advances in generative models, largely powered by diffusion models [22, 56, 57], have enabled photorealistic synthesis of images [50, 51, 53] and videos [4, 6, 19, 21], and have been extended to various other modalities [28, 60]. Generation is primarily controlled by a text or image prompt. Recent works have explored leveraging these models' prior knowledge through either score distillation sampling [25, 35, 40, 47] or fine-tuning on specialized data for downstream applications, such as multi-view images for 3D asset generation [14, 30, 37, 39, 62, 74].

Video generation for motion. Modeling object motion often relies on pre-defined shape models, *e.g.*, SMPL [38] for humans and SMAL [75] for quadrupeds, which are limited to specific categories. Videos can capture general object dynamics [7, 69], but existing video generators trained on Internet-scale videos often produce incoherent motion. Researchers have explored controlling video generation with motion trajectories. [59] extends the framework of [44] to videos, relying on the motion prior of pre-trained video generators, which may not produce high-quality results. Training-based methods *learn* drag-based control using ad-hoc training data. Early efforts [5, 12] train variational autoencoders or diffusion models to synthesize videos with objects in motion, conditioned on sparse motion trajectories derived from optical flow. [33] uses a Fourier-based motion representation for natural, oscillatory dynamics like trees and candles, generating motion with a diffusion model. DragNUWA [70] and others [16, 32, 41, 63, 67] fine-tune pre-trained video generators on large video datasets augmented with motion prompts obtained from off-the-shelf

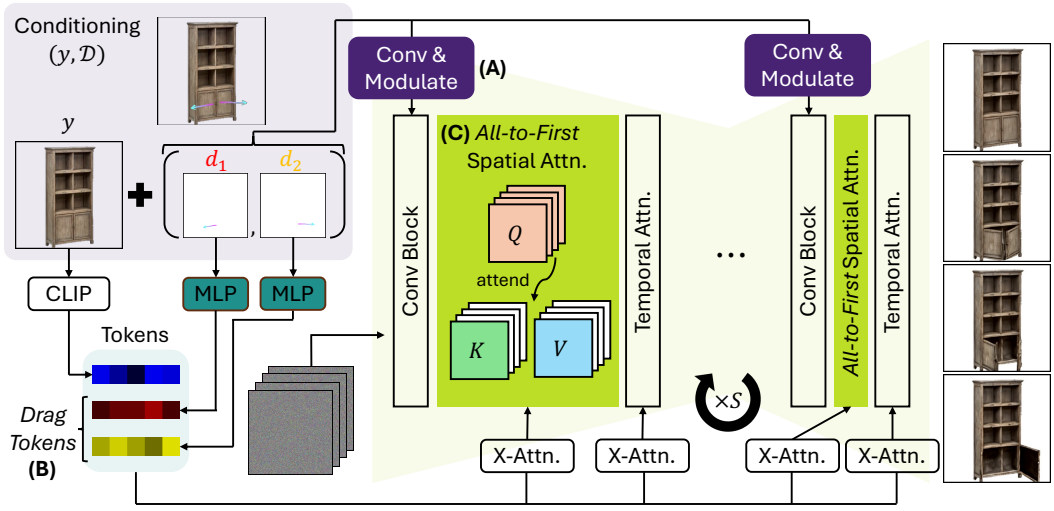

Figure 2. **Architectural Overview of Puppet-Master**. To enable precise drag conditioning, we first modify the original latent video diffusion architecture (Sec. 3.1) by (**A**) adding adaptive layer normalization modules to modulate the internal diffusion features and (**B**) adding cross attention with *drag tokens* (Sec. 3.2). Furthermore, to ensure high-quality appearance and background, we introduce (**C**) *all-to-first* attention, a drop-in replacement for the spatial self-attention modules, where every video frame attends the first one (Sec. 3.3).

trackers, enabling drag-based control in open-domain video generation. However, these methods do *not* control motion at the object part level, as their training data entangles multiple factors, making it challenging to model part-level motion. Other works leverage the motion prior of video generative models for 4D generation tasks [26, 34, 68, 71], but they lack dragging control.

3. Method

Given the initial state of an object, represented by an image y, and one or more drags $\mathcal{D} = \{d_k\}_{k=1}^{K}$, our goal is to synthesize a video $\mathcal{X} = \{x_i\}_{i=1}^{N}$ sampled from the distribution $\mathcal{X} \sim \mathbb{P}(x_1, x_2, \ldots, x_N | y, \mathcal{D})$, where N is the number of video frames. The distribution $\mathbb{P}$ should generate a physically plausible *part-level* animation of the object that responds to the drags. For generalizability, we leverage a "foundation" video generator, Stable Video Diffusion (SVD, Sec. 3.1) [4], which has a general understanding of motion, acquired by training on millions of Internet videos.

In this section, we describe how to fine-tune such a pre-trained video generator to enable part-level motion control of objects. There are two main challenges. First, the drag conditioning must be injected into the video generation pipeline to facilitate efficient learning and accurate, time-consistent motion control. This must be done without strongly interfering with the internal pre-trained video representation. Second, naïvely fine-tuning a pre-trained video diffusion model can result in artifacts such as cluttered backgrounds [30], particularly when the fine-tuning data distribution differs significantly from that of the pre-training data. To address these challenges, in Sec. 3.2, we first introduce a novel mechanism to inject the drag condition $\mathcal{D}$ into the video diffusion model. Then, in Sec. 3.3, we improve the quality of the generated videos by introducing an *all-to-first* attention mechanism, which reduces artifacts like background clutter. While we build on SVD, these techniques should be easily portable to other video generators based on diffusion.

3.1. Preliminaries: Stable Video Diffusion

SVD is an image-conditioned video generator based on diffusion, implementing a denoising process in latent space. It utilizes a variational autoencoder (VAE) (E, D), where the encoder E maps the video frames to the latent space, and the decoder D reconstructs the video from the latent codes. During training, given a pair $(\mathcal{X}, y)$ formed by a video $\mathcal{X} = x^{1:N}$ and the corresponding image prompt y, one first obtains the latent code as $z_0^{1:N} = E(x^{1:N})$, and then adds Gaussian noise $\epsilon \sim \mathcal{N}(0, \boldsymbol{I})$, obtaining the progressively more noised codes

$$z_t^{1:N} = \sqrt{\bar{\alpha}_t} z_0^{1:N} + \sqrt{1 - \bar{\alpha}_t} \epsilon^{1:N}, \quad t = 1, \ldots, T. \tag{1}$$

This uses a pre-defined noising schedule $\bar{\alpha}_0 = 1, \ldots, \bar{\alpha}_T = 0$. The denoising network ϵ_θ is trained to reverse this noising process by optimizing the objective function:

$$\min_\theta \mathbb{E}_{(x^{1:N}, y), t, \epsilon^{1:N} \sim \mathcal{N}(0, \boldsymbol{I})} \left[\| \epsilon^{1:N} - \epsilon_\theta(z_t^{1:N}, t, y) \|_2^2 \right]. \tag{2}$$

Here, ϵ_θ uses the same U-Net architecture as [6], inserting temporal convolution and temporal attention modules after the spatial modules used by [51]. The image conditioning is achieved via (1) cross-attention with the CLIP [49] embedding of the reference frame y; and (2) concatenating the

encoded reference image $E(y)$ channel-wise to $z_t^{1:N}$ as the input of the network ϵ_θ. After ϵ_θ is trained, the model generates a video $\hat{\mathcal{X}}$ prompted by y via iterative denoising from pure Gaussian noise $z_T^{1:N} \sim \mathcal{N}(0, \boldsymbol{I})$, followed by VAE decoding: $\hat{\mathcal{X}} = \hat{x}^{1:N} = D(z_0^{1:N})$.

3.2. Adding Drag Control to Video Diffusion Models

Here, we show how to add the drags $\mathcal{D}$ as an additional input to the denoiser ϵ_θ for part-level motion control. This is achieved by introducing an encoding function for the drags $\mathcal{D}$ and by extending the SVD architecture to inject the resulting code into the network. The model is then fine-tuned using videos combined with corresponding drag prompts in the form of training triplets $(\mathcal{X}, y, \mathcal{D})$. We summarize the key components of the model below and refer the reader to Appendix A for more details.

Drag encoding. Let Ω be the spatial grid $\{1, \dots, H\} \times \{1, \dots, W\}$, where $H \times W$ is the resolution of the video. A *drag* d_k is a tuple $(u_k, v_k^{1:N})$ specifying that the drag starts at location $u_k \in \Omega$ in the reference image y and lands at locations $v_k^n \in \Omega$ in subsequent frames. To encode a set of drags $\mathcal{D} = \{d_k\}_{k=1}^K$, where $K \leq K_{\max} = 5$, we use the multi-resolution encoding of [31]. Each drag d_k[1] is fed to a hand-crafted encoding function $\mathrm{enc}(\cdot, s) : \Omega^N \mapsto \mathbb{R}^{N \times s \times s \times c}$, where s is the desired encoding resolution. The encoding function captures the state of the drag in each frame. Specifically, each slice $\mathrm{enc}(d_k, s)[n]$ encodes (1) the drag's starting location u_k in the reference image, (2) its intermediate location v_k^n in the n-th frame, and (3) its final location v_k^N in the last frame. The $s \times s$ map $\mathrm{enc}(d_k, s)[n]$ is filled with values -1 except at the three locations u_k, v_k^n, and v_k^N, which are encoded using $c = 6$ channels. Finally, we obtain the encoding $\mathcal{D}_{\mathrm{enc}}^s \in \mathbb{R}^{N \times s \times s \times cK_{\max}}$ of $\mathcal{D}$ by concatenating the encodings of the K individual drags, filling extra channels with -1 if $K < K_{\max}$. The encoding function is further detailed in Appendix A.

Drag modulation. The SVD denoiser ϵ_θ comprises a sequence of U-Net blocks computing feature maps $f_s \in \mathbb{R}^{N \times s \times s \times C}$ at different resolutions s. We update each feature f_s based on the drag encoding $\mathcal{D}_{\mathrm{enc}}^s$ using an adaptive normalization module [46], *i.e.*,

$$f_s \leftarrow f_s \otimes (\mathbf{1} + \gamma_s(\mathcal{D}_{\mathrm{enc}}^s)) + \beta_s(\mathcal{D}_{\mathrm{enc}}^s), \tag{3}$$

where $\otimes$ denotes element-wise multiplication. γ_s and $\beta_s \in \mathbb{R}^{N \times s \times s \times C}$ are the *scale* and *shift* terms regressed from the drag encoding $\mathcal{D}_{\mathrm{enc}}^s$. We use convolutional layers to embed $\mathcal{D}_{\mathrm{enc}}^s$ from dimension $cK_{\max}$ to the target dimension C. We empirically find that this mechanism provides better conditioning than using only a single shift term with *no* scaling as in DragAPart [31] (see ablation in Tab. 2).

[1] With a slight abuse of notation, we assume $d_k \in \Omega^N$, as $u_k = v_k^1$ and hence $v_k^{1:N} \in \Omega^N$ fully describes d_k.

Drag tokens. In addition to drag modulation conditioning, we also condition the network ϵ_θ via SVD's built-in cross-attention modules. These modules attend to a *single* key-value pair obtained from the CLIP [49] encoding of the reference image y, and thus degenerate to a global bias term with *no* spatial awareness [55]. In contrast, we concatenate to the CLIP token additional *drag tokens* so that cross-attention is non-trivial. We use multi-layer perceptrons (MLPs) to regress an additional key-value pair from *each* drag d_k. The MLPs take the origin u_k and terminations v_k^n and v_k^N of d_k, along with the internal diffusion features sampled at these locations, which are shown to contain semantic information [2], as inputs. Overall, the cross-attention modules have $1 + K$ key-value pairs (1 is the original image CLIP embedding).

3.3. All-to-First Attention

In our preliminary experiments, we noted that the background of the generated videos does not match the input image y well, often appearing grayer. Instant3D [30] reported a similar problem when generating multiple views of a 3D object, which they addressed via careful noise initialization. [76] and [29] directly constructed training videos with a gray background, which might mitigate the issue visually.

To investigate this issue, we prompt the pre-trained SVD with an image of resolution 256×256 (the resolution used during fine-tuning). As shown in Appendix D, SVD, originally trained on 1024×576 videos, fails to generalize to very different resolutions. We hypothesize that the suboptimal results obtained through fine-tuning arise from the significant discrepancy between the distribution of SVD's training videos and that of our fine-tuning videos, both in terms of resolution and visual content. However, we noticed that the first frame of each generated video is spared from appearance degradation (Fig. 6), as the model effectively replicates the reference image. This suggests that the initial frame serves as a stable foundation for the subsequent frames. To leverage this stability, we propose an *all-to-first* attention mechanism, which introduces a *shortcut* from each noised frame to the first frame via attention.

Previous works [9, 64, 66] have shown that attention between the noised branch and the reference branch improves generation quality for image editing and novel view synthesis tasks. In our *all-to-first* attention, each noised frame attends to the first (reference) frame. We implement this attention by having each frame query the key and value of the first frame, modifying all self-attention layers in the denoising U-Net ϵ_θ. More specifically, denoting the query, key, and value tensors as Q, K, and $V \in \mathbb{R}^{N \times s \times s \times C}$, we discard the key and value tensors of non-first frames, and compute the spatial attention A_i of the i-th frame as follows:

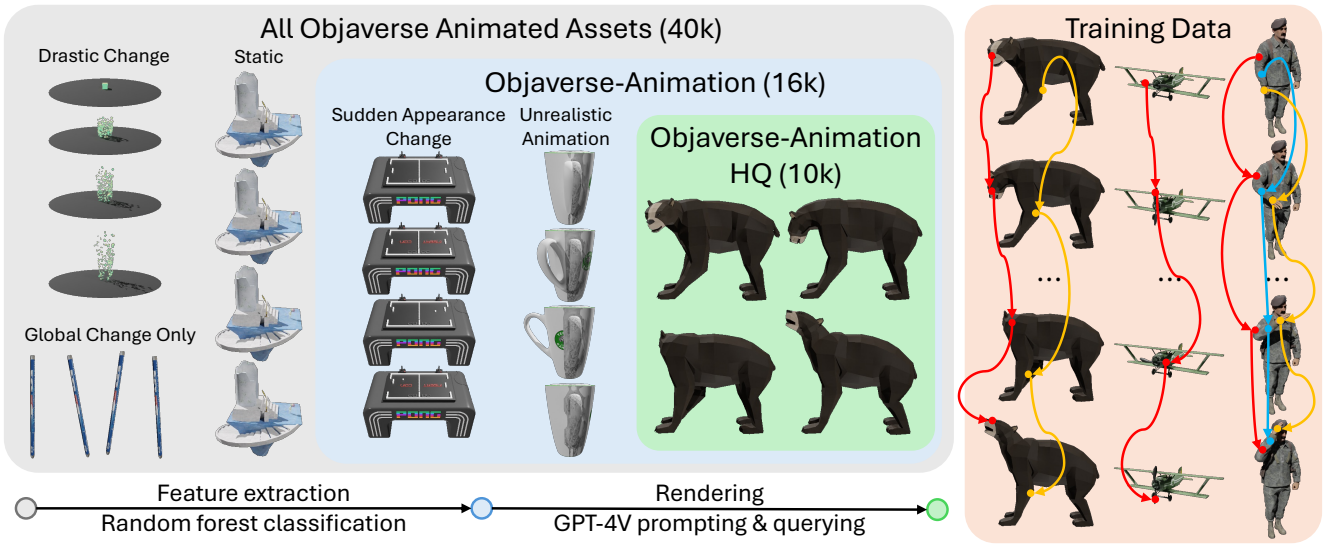

Figure 3. **Data Curation**. We propose two strategies to filter the animated assets in Objaverse, resulting in Objaverse-Animation (16k) and Objaverse-Animation-HQ (10k) of varying levels of curation, from which we construct the training data of Puppet-Master by sampling sparse motion trajectories and projecting them to 2D as drags.

$$A_i = \text{softmax}\left(\frac{\text{flat}(Q[i])\,\text{flat}(K[0])^\top}{\sqrt{D}}\right)\text{flat}(V[0]), \quad (4)$$

where $\text{flat}(\cdot) : \mathbb{R}^{s\times s\times C} \mapsto \mathbb{R}^{L\times C}$ flattens the spatial dimensions to get $L = s \times s$ tokens for attention. The benefit is two-fold: first, this shortcut to the first frame allows subsequent frames to access non-degraded appearance details of the reference image directly. Second, combined with the proposed drag encoding (Sec. 3.2), which specifies the origin u_k at the first frame for *every* frame, all-to-first attention enables the latent pixel corresponding to the drag termination (*i.e.*, v_k^n) to more easily attend to the latent pixel corresponding to the drag origin in the first frame, thereby facilitating learning.

4. Curating Data for Part-Level Object Motion

For training, we require a video dataset that captures the motion of objects at the level of parts. Motion-conditioned video generators fine-tuned using real-world video datasets often confuse part-level motion with object-level motion. It is challenging to curate a high-quality video dataset from Internet videos that features exclusively part-level dynamics. The work of [31] instead used renderings of synthetic 3D objects and their corresponding part annotations obtained from GAPartNet [18]. Unfortunately, this dataset requires manual annotation and animation of 3D object parts, which limits its scale. We instead turn to Objaverse [13], a large-scale 3D dataset of 800k models created by 3D artists, among which 40k are animated. In this section, we introduce a pipeline to extract suitable training videos from these animated assets, together with corresponding drags $\mathcal{D}$.

Identifying animations. While Objaverse [13] has 40k assets labeled as animated, not all animations are useful for our purposes (Fig. 3). Notably, in some, the objects remain static throughout the sequence, while others feature drastic changes in the objects' positions or even their appearances. Therefore, our initial step is to filter out unsuitable animations. To do so, we extract a sequence of aligned point clouds from each animated model and calculate several metrics for each sequence, including: (1) the dimensions and location of the bounding box encompassing the entire motion clip, (2) the size of the largest bounding box for the point cloud at any single timestamp, and (3) the mean and maximal displacement of all points throughout the sequence. Using these metrics, we fit a random forest classifier—trained on a subset of Objaverse animations with manually labeled decisions—to determine whether an animation should be included in the training set. This filtering excludes many assets that exhibit imperceptibly little or overly dramatic motions and results in a subset of 16k animations, which we dub Objaverse-Animation.

Further investigation reveals that this subset still contains assets with highly artificial motion, which do not mimic real-world dynamics (Fig. 3). To avoid such unrealistic dynamics leaking into our synthesized videos, we leverage the multi-modal understanding capability of GPT-4V [43] to assess motion realism. Specifically, for each animated 3D asset in Objaverse-Animation, we fix the camera at the front view and render four images at timestamps corresponding to the four quarters of the animation. We prompt GPT-4V

Method	Video	Base Model	Training Data	Drag-a-Move					Human3.6M			
				PSNR↑	SSIM↑	LPIPS↓	FVD↓	Motion Error↓	PSNR↑	SSIM↑	LPIPS↓	FVD↓
DragNUWA [70]	✓	SVD [4]	WebVid + *Internal*	20.09	0.874	0.172	281.49	17.55/15.41	17.52	0.878	0.158	466.91
DragAnything [67]	✓	SVD [4]	VIPSeg	16.71	0.799	0.296	468.46	16.09/23.21	13.29	0.767	0.305	768.63
Image Conductor [32]	✓	AnimateDiff [20]	WebVid + RealEstate10K	9.20	0.548	0.585	1138.89	20.09/27.51	8.02	0.467	0.628	1957.33
DragAPart [31]												
— *Original*	✗	SD [51]	Drag-a-Move	23.41	0.925	0.085	180.27	14.17/3.71	15.14	0.852	0.197	683.40
— *Re-Trained*	✗	SD [51]	Ours	23.78	0.927	0.082	189.10	14.34/3.73	15.25	0.860	0.188	549.64
Puppet-Master (ours)	✓	SVD [4]	Ours	24.41	0.927	0.085	246.99	12.21/3.53	17.59	0.872	0.155	454.76

Table 1. **Comparisons** with DragNUWA [70], DragAnything [67], Image Conductor [32] and DragAPart [31] on the Drag-a-Move and Human3.6M datasets. Our model has *not* been trained on Human3.6M or any other real video dataset. Colors denote best and second best.

to determine if the motion depicted is sufficiently realistic to qualify for use in training. This filtering mechanism excludes another 6k animations, yielding a subset of 10k animations, which we dub Objaverse-Animation-HQ.

Sampling drags. The goal of drag sampling is to produce a sparse set of drags $\mathcal{D} = \{d_k\}_{k=1}^{K}$, where each drag $d_k \coloneqq (u_k, v_k^{1:N})$ tracks a point u_k on the asset in pixel coordinates throughout the N frames of rendered videos. To encourage the video generator to learn a meaningful motion prior, the set should ideally be both *minimal* and *sufficient*: each group of independently moving parts should have *one* and *only one* drag corresponding to its motion trajectory, similar to Drag-a-Move [31]. For instance, there should be separate drags for different drawers of the same piece of furniture, as their motions are independent, but not for a drawer and its handle, as in this case, the motion of one *implies* that of the other. However, Objaverse [13] lacks the part-level annotation to enforce this property. To partially overcome this, we find that some Objaverse assets are constructed in a bottom-up manner, consisting of multiple sub-models that align well with semantic parts. For these assets, we sample one drag per sub-model; for the rest, we sample a random number of drags in total. For each drag, we first sample a 3D point on the visible part of the model (or sub-model) with probability proportional to the point's total displacement across N frames, and then project its ground-truth motion trajectory $p_1, \ldots, p_N \in \mathbb{R}^3$ to pixel space to obtain d_k. Once all K drags are sampled, we apply a post-processing procedure to ensure that each pair of drags is sufficiently distinct, *i.e.*, for $i \neq j$, we randomly remove one of d_i and d_j if $\|v_i^{1:N} - v_j^{1:N}\|_2^2 \leq \delta$, where δ is a threshold we empirically set to $20N$ for 256×256 renderings.

5. Experiments

The main goal of our experiments is to show that fine-tuning pre-trained video diffusion models on a high-quality *synthetic* dataset, curated to emphasize *part-level* motion, enables them to generate realistic internal dynamics of *real-world* objects, outperforming counterpart models fine-tuned on real videos. To this end, we demonstrate qualitative and quantitative improvements over prior works and excellent generalization to real cases in Sec. 5.2. The design choices that led to Puppet-Master are ablated and discussed in Sec. 5.3. In Appendix B.2, we show the effectiveness of the data curation strategy from Sec. 4. Please refer to Appendix C for implementation details.

5.1. Experiment Settings

Datasets. Puppet-Master is trained on a combined synthetic dataset of Drag-a-Move [31] and Objaverse-Animation-HQ (Sec. 4). For evaluation, we assess its effectiveness using the test set of Drag-a-Move and real data from Human3.6M [24], Amazon-Berkeley Objects [11], and CC-licensed web images in a *zero-shot* manner (*i.e.*, without tuning on real data). For quantitative evaluation, our test set contains 100 videos each from Drag-a-Move and Human3.6M, following [31].

Metrics. For quantitative results, we report the standard video quality metrics, including per-frame PSNR, SSIM, LPIPS [73], and FVD [61]. To better evaluate the model's ability to capture *part-level* dynamics, we introduce and report another motion-based metric dubbed **M**otion **E**rror, or **ME** for short, which is computed as the L2 distance between the tracks estimated from the generated and ground-truth videos (using [27]). In Tab. 1, we report two **ME** variants: the first (*before* the slash) is averaged among the origins of drags only, *i.e.*, $\{u_k\}_{k=1}^{K}$, while the second (*after* the slash) is averaged among all object foreground points. If the generated videos depict part-level dynamics, the second value should be much *smaller* than the first. This is because, in such videos, motion is restricted to the parts activated by the drags; other parts that are not required to move remain static, which matches the ground truths and reduces the overall error.

Figure 4. **Qualitative Comparison** on real images. The videos generated by Puppet-Master are more realistic and capture nuanced part-level dynamics.

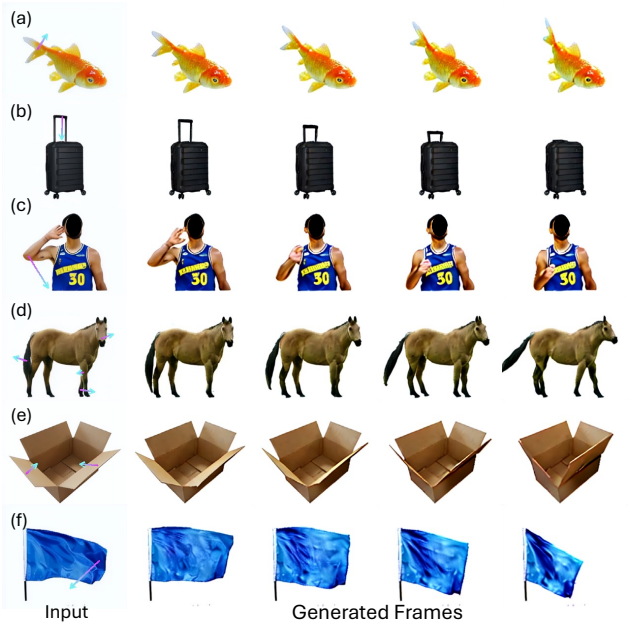

Figure 5. **Results** on *real* images. The generated videos faithfully adhere to the input drags and exhibit motions representative of the underlying categories, including humans, animals, and both articulated and softly deformable man-made objects. In addition, the model learns motion correlations among multiple parts: in (d), without explicitly prompting the rear legs, all four move in sync, while in (e), where the top flaps move independently, only the dragged ones are animated.

5.2. Main Results

Quantitative comparison. In Tab. 1, we compare Puppet-Master to three state-of-the-art motion-conditioned video generators: DragNUWA [70], DragAnything [67], and Image Conductor [32], all trained on real data. On the Drag-a-Move test set, our model consistently outperforms previous models across all metrics. Interestingly, among the four video generators in Tab. 1, only Puppet-Master achieves a markedly better score in the second **ME** metric. This highlights Puppet-Master's superior capability in capturing *part-level* motion dynamics, while DragNUWA, DragAnything, and Image Conductor predominantly induce whole-object movements, so many points incur large errors.

To assess cross-domain generalizability, we evaluate Puppet-Master on Human3.6M [24], an unseen dataset captured in the real world. On this out-of-domain test set, Puppet-Master outperforms prior models on most metrics, despite *not* being fine-tuned on any real videos.

We also report the metrics of DragAPart [31], a drag-conditioned *image* generator for part-level motion. The original DragAPart was trained only on the Drag-a-Move dataset. For fairness, we fine-tune it with the identical data setting as Puppet-Master, and evaluate the performance of both checkpoints (*Original*[2] and *Re-Trained* in Tab. 1). The videos are obtained from N independently generated frames conditioned on gradually extending drags. While its samples exhibit high visual quality in individual frames, they lack temporal smoothness, characterized by abrupt transitions and discontinuities in movement, resulting in a larger motion error[3] (Fig. 4a). Furthermore, DragAPart fails to generalize to out-of-domain cases (Tab. 1 Human3.6M), as its base model, Stable Diffusion, was not trained on videos and lacks inherent motion priors.

Qualitative comparison. We compare samples generated by Puppet-Master and prior models in Fig. 4. In addition to the baselines in Tab. 1, we also compare with Sora [8], a commercial video generator with text and keyframe control. DragAPart, which builds on an image generator, produces samples that lack motion consistency across frames (Fig. 4a). Other video generators *cannot* generate part-level dynamics, introducing unrealistic distortions (Fig. 4be) or scaling or moving the entire object (Fig. 4cd). This includes Sora, which has been trained with orders of magnitude more

[2] *Original* is not ranked as it is trained on single-category data only and hence not an open-domain generator.

[3] FVD is *not* an informative metric for motion quality. Prior works [15, 65] noted that FVD is biased towards the quality of individual frames and does *not* sufficiently account for motion.

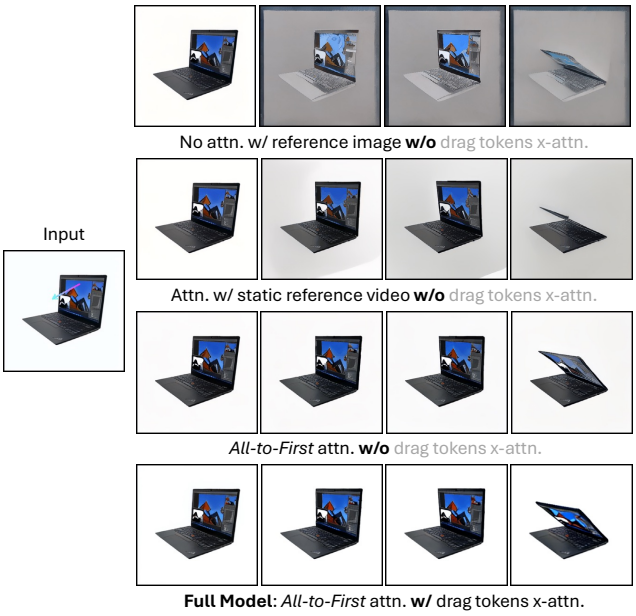

Figure 6. **Visualization** of samples generated by different model designs, where we show the last frame and the first three frames. While all designs produce nearly perfect first frames, our proposed *all-to-first* attention module significantly enhances sample quality. Without this module, the generated samples often exhibit suboptimal appearances and backgrounds. The cross-attention module with drag tokens further improves the appearance details.

data and compute, suggesting that uncurated Internet videos may not be an efficient source for learning the internal motion of objects. By contrast, fine-tuned solely on synthetic 3D renderings, Puppet-Master generates dynamics that are physically plausible, faithful to the input images and drags, and generalizes to real cases. More examples generated by Puppet-Master can be found in Fig. 5.

5.3. Ablations

We conduct ablations to analyze Puppet-Master. For each design choice, we train a separate model using the training split of the Drag-a-Move dataset with a batch size of 8 for 30k steps and evaluate on 100 videos from its test split. Results are shown in Tab. 2 and Fig. 6 and discussed below.

Drag conditioning. Table 2 compares Puppet-Master with several variants of conditioning mechanisms (Sec. 3.2). Adaptive normalization modules ($\mathbb{A}$ *vs.* $\mathbb{B}$) significantly improve both appearance quality (PSNR) and motion consistency (motion error ME). Additionally, we perform an ablation study on the impact of drag encoding with the final termination location v_k^N ($\mathbb{B}$ *vs.* $\mathbb{C}$). Providing the final motion destination of each drag as context for each frame proves beneficial. Incorporating drag tokens in the cross-attention modules enhances spatial awareness and is effective ($\mathbb{C}$ *vs.* $\mathbb{D}$ and Fig. 6). Notably, by combining these (*i.e.*, row $\mathbb{D}$), the model achieves a negligible rate of generated samples with incorrect motion directions.

Setting	PSNR↑	SSIM↑	LPIPS↓	FVD↓	ME↓	%WD↓
Drag conditioning						
$\mathbb{A}$: Shift only w/o end loc.	13.23	0.816	0.446	975.16	15.6	≥ 5
$\mathbb{B}$: Shift+scale w/o end loc.	22.98	0.917	0.093	223.20	**9.3**	4
$\mathbb{C}$: Shift+scale w/ end loc.	23.67	0.926	0.080	205.40	10.5	4
$\mathbb{D}$: $\mathbb{C}$ + x-attn. w/ drag tok.	**24.00**	**0.929**	**0.069**	**170.43**	9.8	**1**
Attn. w/ ref. image						
No attn.	11.96	0.771	0.391	823.00	12.4	≥ 3
Attn. w/ static ref. video	17.51	0.874	0.233	483.18	13.6	≥ 8
All-to-first attn.	**23.67**	**0.926**	**0.080**	**205.40**	**10.5**	4

Table 2. **Ablations**. In addition to the standard metrics and motion error (**ME**) which we introduced in Sec. 5.1, we also manually count the frequency of generated videos whose motion directions are opposite to the intention of their drag inputs (% wrong direction, or **%WD** in short). Here, $\geq$ indicates there are video samples whose motion directions are hard to distinguish. When ablating various designs of attention with the reference image, we use $\mathbb{C}$ as the base drag conditioning architecture.

Attention with the reference image. An evaluation of our proposed *all-to-first* attention is shown in Tab. 2 and Fig. 6. We find that *all-to-first attention* (Sec. 3.3) is essential for video quality. We also compare *all-to-first* attention with an alternative implementation inspired by the X-UNet design of [64], where we pass a static video consisting of the reference image copied N times to the same network architecture and implement cross-attention between the clean (static) reference video branch and the noised video branch. The latter strategy performs worse. We hypothesize that this is due to distribution drift between the two branches, which forces the optimization to modify the pre-trained SVD's internal representations too much.

6. Conclusion

We have introduced Puppet-Master, a video generator that enables control of object motion at the part level via a set of sparse drags. Compared to related works, Puppet-Master incorporates several architectural innovations, including adaptive layer normalization modules, cross-attention modules with drag tokens, and all-to-first spatial attention modules. Ablation studies demonstrate the effectiveness of these contributions. Puppet-Master is trained on Objaverse-Animation-HQ, a newly curated dataset of part-level object animations that we also contribute. Puppet-Master achieves state-of-the-art performance on several benchmarks and exhibits strong *zero-shot* generalization to real-world cases. It also demonstrates the viability of using video generators as proxies for learning a foundation model of the internal dynamics of objects.

Acknowledgments. This work is in part supported by a Toshiba Research Studentship, EPSRC SYN3D EP/Z001811/1, and ERC-CoG UNION 101001212. We thank Luke Melas-Kyriazi, Jinghao Zhou, Minghao Chen and Junyu Xie for useful discussions.

References

[1] Niket Agarwal, Arslan Ali, Maciej Bala, Yogesh Balaji, Erik Barker, Tiffany Cai, Prithvijit Chattopadhyay, Yongxin Chen, Yin Cui, Yifan Ding, et al. Cosmos world foundation model platform for physical ai. *arXiv preprint arXiv:2501.03575*, 2025. 1

[2] Dmitry Baranchuk, Andrey Voynov, Ivan Rubachev, Valentin Khrulkov, and Artem Babenko. Label-efficient semantic segmentation with diffusion models. In *ICLR*, 2021. 4

[3] Volker Blanz and Thomas Vetter. A morphable model for the synthesis of 3D faces. In *Proc. SIGGRAPH*, 1999. 1

[4] Andreas Blattmann, Tim Dockhorn, Sumith Kulal, Daniel Mendelevitch, Maciej Kilian, Dominik Lorenz, Yam Levi, Zion English, Vikram Voleti, Adam Letts, et al. Stable video diffusion: Scaling latent video diffusion models to large datasets. *arXiv preprint arXiv:2311.15127*, 2023. 2, 3, 6, 13

[5] Andreas Blattmann, Timo Milbich, Michael Dorkenwald, and Björn Ommer. iPOKE: Poking a still image for controlled stochastic video synthesis. In *ICCV*, 2021. 2

[6] Andreas Blattmann, Robin Rombach, Huan Ling, Tim Dockhorn, Seung Wook Kim, Sanja Fidler, and Karsten Kreis. Align your latents: High-resolution video synthesis with latent diffusion models. In *CVPR*, 2023. 2, 3

[7] Tim Brooks, Bill Peebles, Connor Holmes, Will DePue, Yufei Guo, Li Jing, David Schnurr, Joe Taylor, Troy Luhman, Eric Luhman, Clarence Ng, Ricky Wang, and Aditya Ramesh. Video generation models as world simulators. Technical report, OpenAI, 2024. 1, 2, 13

[8] Tim Brooks, Bill Peebles, Connor Holmes, Will DePue, Yufei Guo, Li Jing, David Schnurr, Joe Taylor, Troy Luhman, Eric Luhman, Clarence Ng, Ricky Wang, and Aditya Ramesh. Video generation models as world simulators. 2024. 7

[9] 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. 4

[10] Tsai-Shien Chen, Chieh Hubert Lin, Hung-Yu Tseng, Tsung-Yi Lin, and Ming-Hsuan Yang. Motion-conditioned diffusion model for controllable video synthesis. *arXiv preprint arXiv:2304.14404*, 2023. 2

[11] Jasmine Collins, Shubham Goel, Kenan Deng, Achleshwar Luthra, Leon Xu, Erhan Gundogdu, Xi Zhang, Tomas F Yago Vicente, Thomas Dideriksen, Himanshu Arora, Matthieu Guillaumin, and Jitendra Malik. Abo: Dataset and benchmarks for real-world 3d object understanding. In *CVPR*, 2022. 6

[12] Aram Davtyan and Paolo Favaro. Learn the force we can: Enabling sparse motion control in multi-object video generation. In *AAAI*, 2024. 2

[13] 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. 2, 5, 6

[14] Ruiqi Gao, Aleksander Holynski, Philipp Henzler, Arthur Brussee, Ricardo Martin-Brualla, Pratul Srinivasan, Jonathan T Barron, and Ben Poole. Cat3d: Create anything in 3d with multi-view diffusion models. *arXiv preprint arXiv:2405.10314*, 2024. 2

[15] Songwei Ge, Aniruddha Mahapatra, Gaurav Parmar, Jun-Yan Zhu, and Jia-Bin Huang. On the content bias in fréchet video distance. In *CVPR*, 2024. 7

[16] Daniel Geng, Charles Herrmann, Junhwa Hur, Forrester Cole, Serena Zhang, Tobias Pfaff, Tatiana Lopez-Guevara, Carl Doersch, Yusuf Aytar, Michael Rubinstein, et al. Motion prompting: Controlling video generation with motion trajectories. *arXiv preprint arXiv:2412.02700*, 2024. 2

[17] Daniel Geng and Andrew Owens. Motion guidance: Diffusion-based image editing with differentiable motion estimators. In *ICLR*, 2024. 2

[18] Haoran Geng, Helin Xu, Chengyang Zhao, Chao Xu, Li Yi, Siyuan Huang, and He Wang. Gapartnet: Cross-category domain-generalizable object perception and manipulation via generalizable and actionable parts. In *CVPR*, 2023. 5

[19] Rohit Girdhar, Mannat Singh, Andrew Brown, Quentin Duval, Samaneh Azadi, Sai Saketh Rambhatla, Akbar Shah, Xi Yin, Devi Parikh, and Ishan Misra. Emu video: Factorizing text-to-video generation by explicit image conditioning. *arXiv preprint arXiv:2311.10709*, 2023. 2

[20] Yuwei Guo, Ceyuan Yang, Anyi Rao, Zhengyang Liang, Yaohui Wang, Yu Qiao, Maneesh Agrawala, Dahua Lin, and Bo Dai. Animatediff: Animate your personalized text-to-image diffusion models without specific tuning. In *ICLR*, 2024. 6

[21] Jonathan Ho, William Chan, Chitwan Saharia, Jay Whang, Ruiqi Gao, Alexey Gritsenko, Diederik P Kingma, Ben Poole, Mohammad Norouzi, David J Fleet, et al. Imagen video: High definition video generation with diffusion models. *arXiv preprint arXiv:2210.02303*, 2022. 2

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

[23] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022. 13

[24] Catalin Ionescu, Dragos Papava, Vlad Olaru, and Cristian Sminchisescu. Human3.6m: Large scale datasets and predictive methods for 3d human sensing in natural environments. *PAMI*, 2014. 6, 7

[25] Tomas Jakab, Ruining Li, Shangzhe Wu, Christian Rupprecht, and Andrea Vedaldi. Farm3D: Learning articulated 3d animals by distilling 2d diffusion. In *3DV*, 2024. 2

[26] Yanqin Jiang, Chaohui Yu, Chenjie Cao, Fan Wang, Weiming Hu, and Jin Gao. Animate3d: Animating any 3d model with multi-view video diffusion. *arXiv preprint arXiv:2407.11398*, 2024. 3

[27] Nikita Karaev, Ignacio Rocco, Benjamin Graham, Natalia Neverova, Andrea Vedaldi, and Christian Rupprecht. Cotracker: It is better to track together. In *ECCV*, 2024. 6

[28] Jiahui Lei, Congyue Deng, Bokui Shen, Leonidas Guibas, and Kostas Daniilidis. Nap: Neural 3d articulation prior. In

NeurIPS, 2023. 2

[29] Bing Li, Cheng Zheng, Wenxuan Zhu, Jinjie Mai, Biao Zhang, Peterm Wonka, and Bernard Ghanem. Vivid-zoo: Multi-view video generation with diffusion model. *arXiv preprint arXiv:2406.08659*, 2024. 2, 4

[30] Jiahao Li, Hao Tan, Kai Zhang, Zexiang Xu, Fujun Luan, Yinghao Xu, Yicong Hong, Kalyan Sunkavalli, Greg Shakhnarovich, and Sai Bi. Instant3D: Fast text-to-3D with sparse-view generation and large reconstruction model. In *ICLR*, 2024. 2, 3, 4

[31] Ruining Li, Chuanxia Zheng, Christian Rupprecht, and Andrea Vedaldi. Dragapart: Learning a part-level motion prior for articulated objects. In *ECCV*, 2024. 2, 4, 5, 6, 7, 12, 13

[32] Yaowei Li, Xintao Wang, Zhaoyang Zhang, Zhouxia Wang, Ziyang Yuan, Liangbin Xie, Yuexian Zou, and Ying Shan. Image conductor: Precision control for interactive video synthesis. *arXiv preprint arXiv:2406.15339*, 2024. 2, 6, 7, 13

[33] Zhengqi Li, Richard Tucker, Noah Snavely, and Aleksander Holynski. Generative image dynamics. In *CVPR*, 2024. 2

[34] Hanwen Liang, Yuyang Yin, Dejia Xu, Hanxue Liang, Zhangyang Wang, Konstantinos N Plataniotis, Yao Zhao, and Yunchao Wei. Diffusion4d: Fast spatial-temporal consistent 4d generation via video diffusion models. *arXiv preprint arXiv:2405.16645*, 2024. 3

[35] 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. 2

[36] Pengyang Ling, Lin Chen, Pan Zhang, Huaian Chen, and Yi Jin. Freedrag: Point tracking is not you need for interactive point-based image editing. In *CVPR*, 2024. 2

[37] 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. 2

[38] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J. Black. SMPL: a skinned multi-person linear model. In *ACM TOG*, 2015. 1, 2

[39] Luke Melas-Kyriazi, Iro Laina, Christian Rupprecht, Natalia Neverova, Andrea Vedaldi, Oran Gafni, and Filippos Kokkinos. Im-3d: Iterative multiview diffusion and reconstruction for high-quality 3d generation. In *ICLR*, 2024. 2

[40] Luke Melas-Kyriazi, Iro Laina, Christian Rupprecht, and Andrea Vedaldi. Realfusion: 360deg reconstruction of any object from a single image. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pages 8446–8455, 2023. 2

[41] Chong Mou, Mingdeng Cao, Xintao Wang, Zhaoyang Zhang, Ying Shan, and Jian Zhang. Revideo: Remake a video with motion and content control. *arXiv preprint arXiv:2405.13865*, 2024. 2

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

[43] OpenAI. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023. 5

[44] Xingang Pan, Ayush Tewari, Thomas Leimkühler, Lingjie Liu, Abhimitra Meka, and Christian Theobalt. Drag your gan: Interactive point-based manipulation on the generative image manifold. In *ACM SIGGRAPH*, 2023. 2

[45] Jack Parker-Holder, Philip Ball, Jake Bruce, Vibhavari Dasagi, Kristian Holsheimer, Christos Kaplanis, Alexandre Moufarek, Guy Scully, Jeremy Shar, Jimmy Shi, Stephen Spencer, Jessica Yung, Michael Dennis, Sultan Kenjeyev, Shangbang Long, Vlad Mnih, Harris Chan, Maxime Gazeau, Bonnie Li, Fabio Pardo, Luyu Wang, Lei Zhang, Frederic Besse, Tim Harley, Anna Mitenkova, Jane Wang, Jeff Clune, Demis Hassabis, Raia Hadsell, Adrian Bolton, Satinder Singh, and Tim Rocktäschel. Genie 2: A large-scale foundation world model, 2024. 1

[46] Ethan Perez, Florian Strub, Harm De Vries, Vincent Dumoulin, and Aaron Courville. Film: Visual reasoning with a general conditioning layer. In *AAAI*, 2018. 4

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

[48] Lingteng Qiu, Guanying Chen, Xiaodong Gu, Qi Zuo, Mutian Xu, Yushuang Wu, Weihao Yuan, Zilong Dong, Liefeng Bo, and Xiaoguang Han. Richdreamer: A generalizable normal-depth diffusion model for detail richness in text-to-3d. In *CVPR*, 2024. 12

[49] 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. 3, 4

[50] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation. In *ICML*, 2021. 2

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

[52] Javier Romero, Dimitrios Tzionas, and Michael J. Black. Embodied hands: Modeling and capturing hands and bodies together. *Proc. SIGGRAPH*, 36(6), 2022. 1

[53] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily L Denton, Kamyar Ghasemipour, Raphael Gontijo Lopes, Burcu Karagol Ayan, Tim Salimans, et al. Photorealistic text-to-image diffusion models with deep language understanding. In *NeurIPS*, 2022. 2

[54] 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

[55] Ido Sobol, Chenfeng Xu, and Or Litany. Zero-to-hero: Enhancing zero-shot novel view synthesis via attention map filtering. *arXiv preprint arXiv:2405.18677*, 2024. 4

[56] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. In *NeurIPS*, 2019. 2

[57] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *ICLR*, 2021. 2

[58] Jiapeng Tang, Markhasin Lev, Wang Bi, Thies Justus, and Matthias Nießner. Neural shape deformation priors. In *NeurIPS*, 2022. 1

[59] Yao Teng, Enze Xie, Yue Wu, Haoyu Han, Zhenguo Li, and Xihui Liu. Drag-a-video: Non-rigid video editing with point-

based interaction. *arXiv preprint arXiv:2312.02936*, 2023. 2

[60] Guy Tevet, Sigal Raab, Brian Gordon, Yoni Shafir, Daniel Cohen-or, and Amit Haim Bermano. Human motion diffusion model. In *ICLR*, 2022. 2

[61] Thomas Unterthiner, Sjoerd Van Steenkiste, Karol Kurach, Raphaël Marinier, Marcin Michalski, and Sylvain Gelly. Fvd: A new metric for video generation. In *ICLR*, 2019. 6

[62] Vikram Voleti, Chun-Han Yao, Mark Boss, Adam Letts, David Pankratz, Dmitry Tochilkin, Christian Laforte, Robin Rombach, and Varun Jampani. Sv3d: Novel multi-view synthesis and 3d generation from a single image using latent video diffusion. *arXiv preprint arXiv:2403.12008*, 2024. 2

[63] Zhouxia Wang, Ziyang Yuan, Xintao Wang, Tianshui Chen, Menghan Xia, Ping Luo, and Ying Shan. Motionctrl: A unified and flexible motion controller for video generation. *arXiv preprint arXiv:2312.03641*, 2023. 2

[64] Daniel Watson, William Chan, Ricardo Martin Brualla, Jonathan Ho, Andrea Tagliasacchi, and Mohammad Norouzi. Novel view synthesis with diffusion models. In *ICLR*, 2023. 4, 8

[65] Daniel Watson, Saurabh Saxena, Lala Li, Andrea Tagliasacchi, and David J Fleet. Controlling space and time with diffusion models. *arXiv preprint arXiv:2407.07860*, 2024. 7

[66] Haohan Weng, Tianyu Yang, Jianan Wang, Yu Li, Tong Zhang, CL Chen, and Lei Zhang. Consistent123: Improve consistency for one image to 3d object synthesis. *arXiv preprint arXiv:2310.08092*, 2023. 4

[67] Weijia Wu, Zhuang Li, Yuchao Gu, Rui Zhao, Yefei He, David Junhao Zhang, Mike Zheng Shou, Yan Li, Tingting Gao, and Di Zhang. Draganything: Motion control for anything using entity representation. In *ECCV*, 2024. 2, 6, 7, 13

[68] Yiming Xie, Chun-Han Yao, Vikram Voleti, Huaizu Jiang, and Varun Jampani. SV4D: Dynamic 3d content generation with multi-frame and multi-view consistency. *arXiv preprint arXiv:2407.17470*, 2024. 3

[69] Sherry Yang, Jacob Walker, Jack Parker-Holder, Yilun Du, Jake Bruce, Andre Barreto, Pieter Abbeel, and Dale Schuurmans. Video as the new language for real-world decision making. In *ICML*, 2024. 2

[70] Shengming Yin, Chenfei Wu, Jian Liang, Jie Shi, Houqiang Li, Gong Ming, and Nan Duan. Dragnuwa: Fine-grained control in video generation by integrating text, image, and trajectory. *arXiv preprint arXiv:2308.08089*, 2023. 2, 6, 7, 13

[71] Haiyu Zhang, Xinyuan Chen, Yaohui Wang, Xihui Liu, Yunhong Wang, and Yu Qiao. 4diffusion: Multi-view video diffusion model for 4d generation. *arXiv preprint arXiv:2405.20674*, 2024. 3

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

[73] 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. 6

[74] Chuanxia Zheng and Andrea Vedaldi. Free3d: Consistent novel view synthesis without 3d representation. In *CVPR*, 2024. 2

[75] Silvia Zuffi, Angjoo Kanazawa, David W. Jacobs, and Michael J. Black. 3D menagerie: Modeling the 3D shape and pose of animals. In *CVPR*, 2017. 1, 2

[76] Qi Zuo, Xiaodong Gu, Lingteng Qiu, Yuan Dong, Zhengyi Zhao, Weihao Yuan, Rui Peng, Siyu Zhu, Zilong Dong, Liefeng Bo, et al. Videomv: Consistent multi-view generation based on large video generative model. *arXiv preprint arXiv:2403.12010*, 2024. 2, 4

A. Additional Details of the Drag Encoding

Here, we give a formal definition of $\mathrm{enc}(\cdot, s)$ introduced in Sec. 3.2. Recall that $\mathrm{enc}(\cdot, s)$ encodes each drag $d_k := (u_k, v_k^{1:N})$ into an embedding of shape $N \times s \times s \times 6$. For each frame n, the first, middle, and last two channels (of the $c = 6$ in total) encode the spatial location of u_k, v_k^n, and v_k^N, respectively. Formally, $\mathrm{enc}(d_k, s)[n, :, :, : 2]$ is a tensor of all negative ones except for $\mathrm{enc}(d_k, s)[n, \lfloor \frac{s \cdot h}{H} \rfloor, \lfloor \frac{s \cdot w}{W} \rfloor, : 2] = \left(\frac{s \cdot h}{H} - \lfloor \frac{s \cdot h}{H} \rfloor, \frac{s \cdot w}{W} - \lfloor \frac{s \cdot w}{W} \rfloor\right)$ where $u_k = (h, w) \in \Omega = \{1, \ldots, H\} \times \{1, \ldots, W\}$. The other 4 channels are defined similarly, with u_k replaced by v_k^n and v_k^N.

B. Additional Details of Data Curation

B.1. Implementation Details

We use the categorization provided by GObjaverse [48] and exclude 3D models classified as '`Poor-Quality`' as a pre-filtering step prior to our proposed filtering pipelines (Sec. 4).

When using GPT-4V to filter Objaverse-Animation into Objaverse-Animation-HQ, we designed the following prompt to cover a wide range of cases to be excluded:

> **System**: You are a 3D artist, and now you are being shown some animation videos depicting an animated 3D asset. You are asked to filter out some animations.
> You should filter out the animations that:
> (1) have trivial or no motion, i.e., the object is simply scaling, rotating, or moving as a whole without part-level dynamics;
> or (2) depict a scene and only a small component in the scene is moving;
> or (3) have motion that is imaginary, i.e., the motion is not the usual way of how the object moves and it's hard for humans to anticipate;
> or (4) have very large global motion so that the object exits the frame partially or fully in one of the frames;
> or (5) have changes in object color that are not due to lighting changes;
> or (6) have motion that causes different parts of the same object to disconnect, overlap in an unnatural way, or disappear;
> or (7) have motion that is very chaotic, for example objects exploding or bursting apart.
> **User**: For the following animation (as frames of a video), `frame1`, `frame2`, `frame3`, `frame4`, tell me, in a single word 'Yes' or 'No', whether the video should be filtered out or not.

The cost of GPT-4V data filtering is about $500.

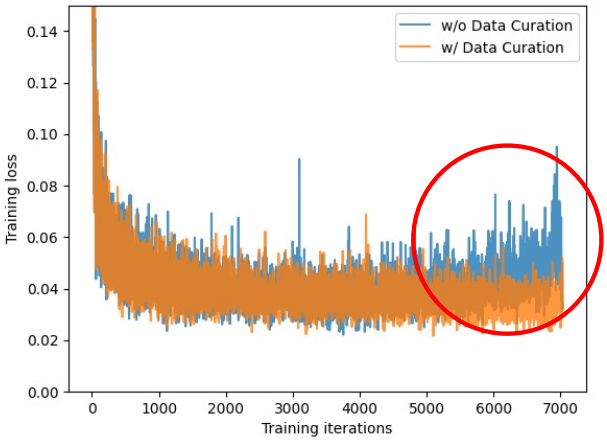

Figure 7. Data curation helps stabilize training.

Setting	PSNR↑	SSIM↑	LPIPS↓	FVD↓
w/o Data Curation	6.04	0.411	0.703	1475.35
w/ Data Curation	**19.87**	**0.884**	**0.181**	**624.47**

Table 3. Training on more abundant but lower-quality data leads to lower generation quality. Here, 'w/o Data Curation' model is trained on Objaverse-Animation while 'w/ Data Curation' model is trained on Objaverse-Animation-HQ. Both models are trained for 7k iterations. Evaluation is performed on the test split of Drag-a-Move.

B.2. Less is More: Data Curation Helps at Scale

To verify that our data curation strategy from Sec. 4 is effective, we compare two models trained on Objaverse-Animation and Objaverse-Animation-HQ, respectively, under the same hyperparameter setting. The training dynamics are visualized in Fig. 7. The optimization collapses towards 7k iterations when the model is trained on a less curated dataset, resulting in much lower-quality video samples (Tab. 3). This suggests that when fine-tuning a pre-trained video diffusion model to generate part-level motion, the quality of the data is more critical than its quantity.

C. Additional Experiment Details

C.1. Training Details

Data. Our final model is fine-tuned on the combined dataset of Drag-a-Move [31] and Objaverse-Animation-HQ (Sec. 4). During training, we balance various types of part-level dynamics to control the data distribution. We achieve this by leveraging the categorization provided by GObjaverse [48] and sampling individual data points with the following hand-crafted distribution: p(Drag-a-Move) $= 0.3$, p(Objaverse-Animation-HQ, category '`Human-Shape`') $= 0.25$, p(Objaverse-Animation-HQ,

category 'Animals') = 0.25, p(Objaverse-Animation-HQ, category 'Daily-Used') = 0.05, p(Objaverse-Animation-HQ, other categories) = 0.15.

Architecture. We zero-initialize the final convolutional layer of each adaptive normalization module before fine-tuning. With our introduced modules, the parameter count increases to 1.68B from the original 1.5B in SVD.

Training. We fine-tune the base SVD on videos of 256×256 resolution and $N = 14$ frames with a batch size of 64 for $12,500$ iterations. We adopt SVD's continuous-time noise scheduler, shifting the noise distribution towards more noise with $\log \sigma \sim \mathcal{N}(0.7, 1.6^2)$, where σ is the continuous noise level following the presentation in [4]. Training takes roughly 10 days on a single Nvidia A6000 GPU, where we accumulate gradients for 64 steps. We enable classifier-free guidance (CFG) [23] by randomly dropping the conditional drags $\mathcal{D}$ with a probability of 0.1 during training. Additionally, we track an exponential moving average of the weights at a decay rate of 0.9999.

C.2. Inference and Evaluation Details

Inference. Unless stated otherwise, samples are generated using $S = 50$ diffusion steps. We adopt linearly increasing CFG [4] with a maximum guidance weight of 5.0. Generating a single video takes roughly 20 seconds on an Nvidia A6000 GPU.

Baselines. For DragNUWA [70], DragAnything [67], and Image Conductor [32], we use their publicly available checkpoints. DragNUWA and DragAnything operate at a resolution of 576×320, and Image Conductor at 384×256. Following previous work [31], we first pad the square input image y along the horizontal axis to the correct aspect ratio and resize it to the corresponding resolution, then remove the padding from the generated frames and resize them back to 256×256. For methods that require text prompts (*i.e.*, DragNUWA and Image Conductor), we use generic prompts to describe the category of the evaluation images (*e.g.*, 'A Furniture' for Drag-a-Move and 'A person' for Human3.6M). Note that Image Conductor is trained on 16-frame videos instead of 14-frame ones. We experimented with (1) simply generating 14 frames at inference time; and (2) generating 16 frames and discarding the last two frames. The latter gives slightly better results, which we report. We find that tasking it to generate 14-frame videos produces reasonable results which we report. All metrics are computed on 14-frame videos of resolution 256×256.

We train DragAPart [31] for 100k iterations using its official implementation on the same combined dataset of Drag-a-Move and Objaverse-Animation-HQ used for training Puppet-Master. Since DragAPart is an image-to-image model, we independently generate 14 frames conditioned on gradually extending drags to obtain the video.

For Sora [7], we uploaded the conditioning image in Fig. 4 as the start frame. Since the model does *not* support motion control, we manually crafted the following prompt to convey the motion condition:

> A photorealistic video of a modern, light grey wooden sideboard with a natural wood top. The three drawers at the top remain completely static and closed throughout the entire video, without any movement or displacement. From this initial state, only the bottom cabinet doors begin to slowly and smoothly close, moving in a natural, physically plausible manner. The motion follows proper hinge mechanics, ensuring perfect alignment, symmetry, and realism, with no jerky or unnatural movement. The camera remains fixed in the same frontal view, maintaining the exact perspective of the reference image. The lighting is soft and even, enhancing the wood texture, clean lines, and elegant design without casting harsh shadows or introducing distractions. The video maintains a high-quality, cinematic appearance, with no additional objects or background elements.

D. Video Diffusion Models on Out-of-Domain Resolutions

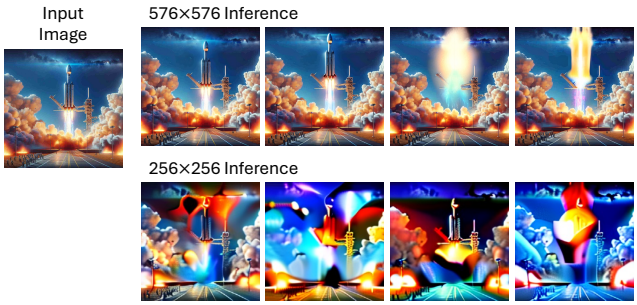

Figure 8. Stable Video Diffusion *fails* to generalize robustly to out-of-domain resolutions at inference time.

The convolution and attention modules in video diffusion models like SVD are *not* invariant to input resolution. As demonstrated in Fig. 8, our base model SVD, which was trained on videos with resolution 1024×576, *cannot* generate high-quality videos at out-of-domain resolutions such as 256×256. We hypothesize that this resolution shift makes fine-tuning susceptible to local optima, resulting in visually cluttered generations (Fig. 6). All-to-first attention (Sec. 3.3) significantly reduces this appearance degradation.

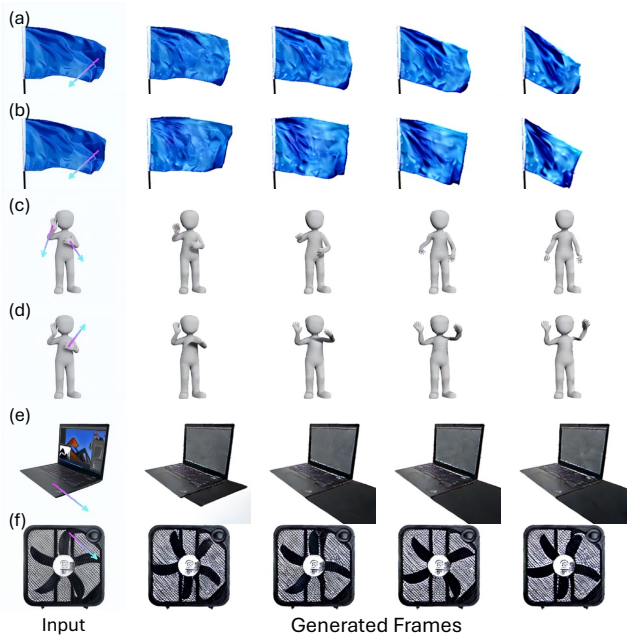

Figure 9. **More examples** generated by Puppet-Master.

Figure 10. **Results** on images with diverse backgrounds.

E. Discussions

Motion diversity. In Fig. 9(a-d), we show that Puppet-Master can generate diverse part-level animations, both across different random seeds when conditioned on the same input image and set of drags (*i.e.*, a and b), and across different sets of drags when conditioned on the same input image (*i.e.*, c and d).

Part-level *vs*. object-level motion. In this work, we focus on synthesizing *internal*, *part-level* motion. To achieve this, we curated Objaverse-Animation-HQ to specifically learn motions involving object parts being manipulated. As a result, Puppet-Master is not designed for *global* object motion and may produce artifacts when the input drag(s) do *not* correspond to meaningful part-level movement (Fig. 9e).

Failure cases. Puppet-Master may fail to maintain the shape of objects, occasionally leading to the disappearance of certain parts. This issue is particularly evident when physically plausible motion necessitates precise coordination among multiple object parts, such as the five fan blades in Fig. 9f.

Results with real-world backgrounds. Although all training frames are rendered with a white background, Puppet-Master retains some ability from the SVD backbone to handle complex backgrounds, as illustrated in Fig. 10. Better results could be obtained by incorporating, *e.g.*, random backgrounds during training.

Limitations. Another limitation of our model is its slight difficulty in preserving the exact color appearance of objects during inference on real-world images. This issue arises due to two primary factors: (1) the synthetic 3D models in Objaverse-Animation-HQ typically feature high-contrast, stylized textures, leading to a train-test discrepancy in color distributions; and (2) when testing at a lower resolution (*e.g.*, 256×256) compared to the native resolution of SVD, noise in the denoiser's output can propagate across a larger region of the image because of the fixed receptive field of convolutional layers, leading to many instances having a slightly flickering appearance.

Future work. While most motion-conditioned video generators prioritize object-level motion over fine-grained part-level motion, we have demonstrated it is feasible to learn a part-level motion prior using a modestly sized, high-quality synthetic dataset that generalizes effectively to real-world data. Future research may develop a dynamic routing mechanism that integrates both part-level and object-level dynamics.