BlobGEN-Vid: Compositional Text-to-Video Generation with Blob Video Representations

Weixi Feng[1] Chao Liu[2] Sifei Liu[2] William Yang Wang[1]
Arash Vahdat[2] Weili Nie[2]

[1]UC Santa Barbara [2]NVIDIA

blobgen-vid2.github.io

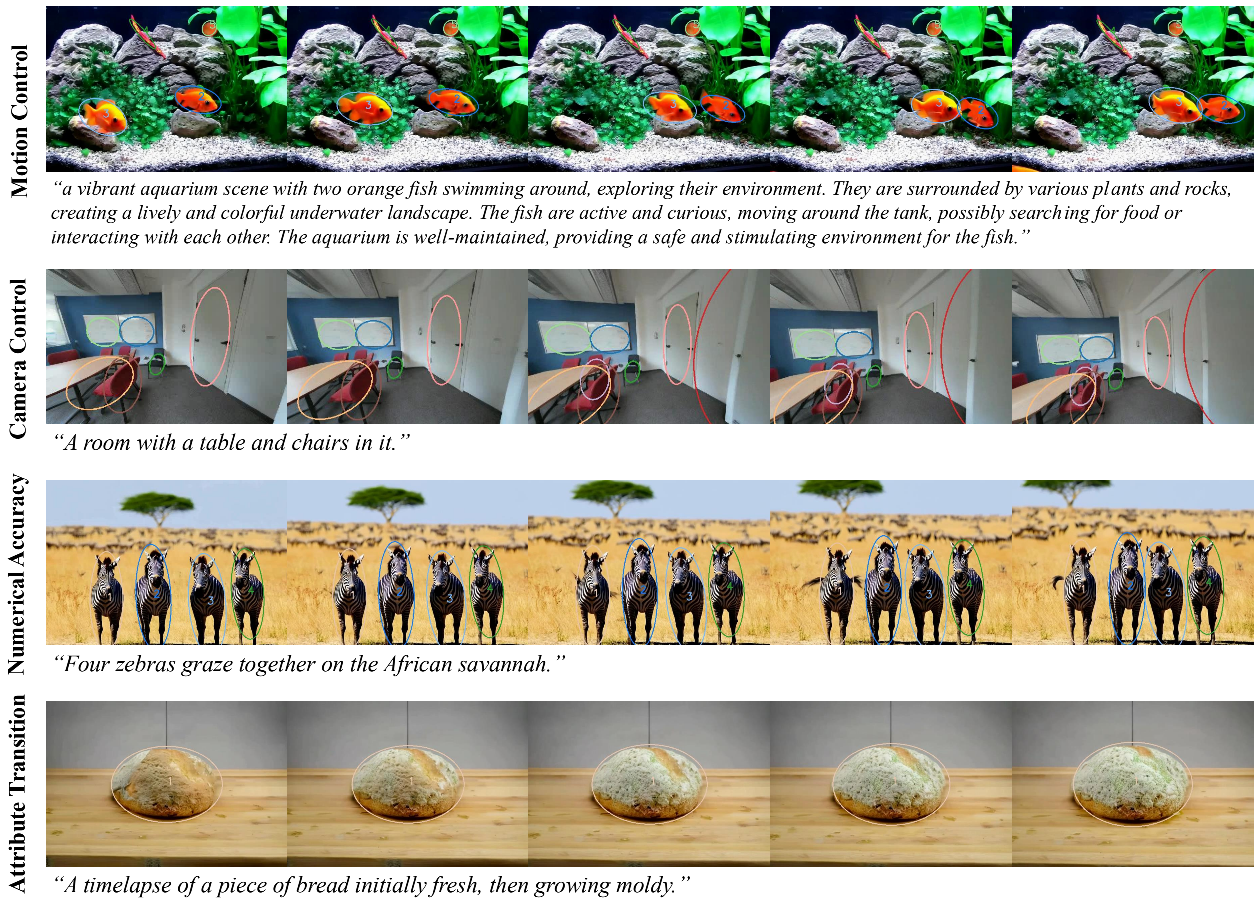

Motion Control
"a vibrant aquarium scene with two orange fish swimming around, exploring their environment. They are surrounded by various plants and rocks, creating a lively and colorful underwater landscape. The fish are active and curious, moving around the tank, possibly searching for food or interacting with each other. The aquarium is well-maintained, providing a safe and stimulating environment for the fish."
Camera Control
"A room with a table and chairs in it."
Numerical Accuracy
"Four zebras graze together on the African savannah."
Attribute Transition
"A timelapse of a piece of bread initially fresh, then growing moldy."

Figure 1. With blob video representations, BlobGEN-Vid can support fine-grained controllability in text-to-video generation in terms of motion control, camera control, numerical accuracy and attribute transition. Blobs in top two rows are extracted from a video and a 3D scene, respectively, using the pre-trained segmentation model and image captioning model, while blobs in bottom two rows are generated by GPT-4o with the given global prompt as input.

Abstract

Existing video generation models struggle to follow complex text prompts and synthesize multiple objects, raising the need for additional grounding input for improved controllability. In this work, we propose to decompose videos into visual primitives – blob video representation, a general representation for controllable video generation. Based on *blob conditions, we develop a blob-grounded video diffusion model named BlobGEN-Vid that allows users to control object motions and fine-grained object appearance. In particular, we introduce a masked 3D attention module that effectively improves regional consistency across frames. In addition, we introduce a learnable module to interpolate text embeddings so that users can control semantics in specific frames and obtain smooth object transitions. We show that our framework is model-agnostic and build BlobGEN-Vid based on both U-Net and DiT-based video diffusion models. Extensive experimental results show that BlobGEN-Vid achieves superior zero-shot video generation ability and state-of-the-art layout controllability on multiple benchmarks. When combined with an LLM for layout planning, our framework even outperforms proprietary text-to-video generators in terms of compositional accuracy.*

1. Introduction

Recent advancements in text-to-video generation have enabled us to generate more realistic videos with high visual quality and intricate motions. These advancements are driven by new model architectures [2, 13, 49], improved training techniques [1, 4, 18] and large-scale video datasets [6, 46]. Despite the progress, existing text-to-video models still struggle to follow complex prompts, where they often neglect key objects or confuse multiple objects as one concept. In addition, users cannot control semantic transitions or camera motion with merely text descriptions with these models. Therefore, it remains an open challenge to enhance the compositionality and controllability of video generators with layout guidance in the diffusion process.

To resolve these challenges, recent studies propose to condition video diffusion models on visual layouts. Since a text prompt can be ambiguous in object locations and visual appearances, video generators often fail to generate scenes with large motion or complex compositions. Additional grounding inputs can guide the generation process for enhanced controllability. These layouts are usually represented by bounding boxes moving across frames [19, 22, 23, 42]. Compared to other modalities such as depth [34] or semantic maps [52], bounding boxes are easier to create and manipulate by users while providing coarse-grained information of local objects. However, 2D bounding boxes lack perspective invariance: the 3D counterpart of a 2D bounding box on an image is not a 3D bounding box and vice versa. This makes it difficult to synthesize 3D scenes using models grounded by bounding-boxes.

In this work, we introduce a new type of visual layouts for video generation, named *blob video representations*, to serve as grounding conditions. Each blob sequence corresponds to an object instance and can be automatically extracted from videos (or 3D scenes), making it a more general and robust representation for different visual domains. Specifically, a blob video representation has two components: 1) the blob parameters, which formulate a tilted ellipse to specify the object's location, size, and orientation; and 2) the blob description, which is a free-form language description of the object's visual attributes. With this definition, our blob representation enables both motion and semantic control of visual compositions. It is also convenient for users to create and manipulate such representations as the blob parameters can be represented as structured text.

While layout conditions have been widely studied in image generation [5, 20, 31, 52], directly applying these methods in video can lead to temporal inconsistency or compromised layout control [19]. Some recent studies have adapted these conditions for video generation with new techniques [19, 42]. However, they still suffer from the above issues and are limited to class conditions for each object box. To this end, we develop a blob-grounded text-to-video diffusion framework, termed BlobGEN-Vid, that is built upon existing video diffusion models using blob representations as grounding input. In our framework, we introduce a masked 3D attention module that facilitates object-centric spatial-temporal attention. We also utilize masked cross-attentions [31] to fuse free-form object descriptions into the blob regions. As some frames do not have blob captions, we integrate a context interpolation module to enhance semantic transition throughout time.

BlobGEN-Vid is a model-agnostic framework that can be applied to both UNet [12] and DiT [32] based diffusion models. Our experiments in open-domain video generation indicate that BlobGEN-Vid outperforms existing layout-guided video generators by a large margin in multiple dimensions. We evaluate BlobGEN-Vid on a wide range of benchmarks [44] and show that it improves the layout controllability by at least 20% in mIOU and prompt alignment by 5% in CLIP similarity. When combined with a large language model (LLMs) for blob planning, our pipeline outperforms proprietary video generators in mutiple aspects. Last but not least, we demonstrate that BlobGEN-Vid also achieves improved consistency and camera control in multi-view image generation in indoor scenes.

Our contributions: *(i)* We propose a new blob representation for text-to-video generation that enables fine-grained control of each object such as its motion and appearance. *(ii)* We propose BlobGEN-Vid, a blob-grounded framework that incorporates two types of masked attention modules and a context interpolation module to pre-trained video diffusion models for regional control and temporal consistency. BlobGEN-Vid can be applied to both UNet and DiT based diffusion backbones. *(iii)* We conduct extensive experiments in open-domain video generation and multi-view indoor scene generation, demonstrating BlobGEN-Vid's superior object-level controllability and temporal consistency in generating high-quality videos.

2. Related work

Text-to-video generation. The field of text-to-video generation (T2V) has gained much attention thanks to the advancement in new model architecture [2, 16, 32, 54], large-scale video datasets [6, 46], and improved training techniques [4, 7, 18]. There are mainly two streams of video diffusion models in terms of model architecture. Primitive video diffusion models such as Video LDM [2], VideoCrafter [3, 4] and some others [41, 43, 51] are achieved by adding temporal self-attention modules into a U-Net diffusion backbone [12]. The U-Net usually operates in the latent space [40] and can be inherited from a pretrained text-to-image model such as Stable Diffusion [2]. Recently, as the scale of model and dataset increases, a few models based on Diffusion Transformers (DiT) [32] have been proposed. For example, CogVideoX [49] proposes an expert DiT with stacked 3D attention blocks working on the concatenation of context embeddings and visual tokens. As synthetic videos are becoming more realistic, it is essential to endow controllability to the generation process.

Compositional video generation. While compositional generation have been extensively studied in the image domain [20, 25, 31], the compositional tasks in video generation still demand more focus [38, 48]. Recently, a few works start to tackle compositional video generation. Vico [31] regularizes text tokens' attention maps to improve scene correctness with multiple objects. VideoTetris [38] proposes a spatial-temporal composing mechanism to deal with compositional change in long video generation. Several benchmarks are proposed to characterize compositionality [28, 29]. Beyond issues carried from image generation, the temporal dimension in videos introduces new challenging problems. For example, T2V-CompBench [35] and TC-Bench [8] features dynamic binding relations or object status change. These benchmarks show that existing T2V models lack of robust compositionality in generating videos with complex scenes and motions.

Layout-guided video generation. There are several studies that attempt to add layout condition on top of a pretrained T2V model. TrackDiffusion [19] inserts trainable gated cross attentions with an instance enhancer to improve object consistency across frames. Boximator [42] applies a self-attention to fuse object category and box coordinates into visual tokens. It also proposes self-tracking technique that fine-tunes the model to generate visible bounding boxes around objects first and then forget the behavior. LVD [22] and VideoDirectorGPT [23] adopt LLMs to plan bounding boxes for several keyframes, which are then passed to a video generator. As shown later, these methods may suffer from inconsistency issue and lack of controllabbility with complex layouts.

3. Preliminary: BlobGEN

BlobGEN [31] first introduced blob representations to guide the open-domain image generation. It has shown that blobs can provide more fine-grained controllability than other visual layouts (such as bounding boxes) in previous layout-conditioned approaches [9, 20], which motivates us to use blob representations for video generation. We will next introduce the blob representations and key method design in BlobGEN, which our method is built upon.

Blob representations. The blob representations denote the object-level visual primitives in a scene, each of which consists of two components: blob parameters and blob description. The blob parameters depict an object's shape, size and location with a vector of five variables $\tau = [c_x, c_y, a, b, \theta]$ that defines a tilted ellipse, where (c_x, c_y) is the center point of the ellipse, a, b are the radii of its semi-major and semi-minor axes, and $\theta \in (-\pi, \pi]$ is the orientation angle of the ellipse. The blob description denoted by s captures the visual appearance of an object using a region-level synthetic caption extracted by an image captioning model. Compared with other visual layouts (such as boxes and semantic maps), blob representations have both two advantages: 1) they retain the fine-grained spatial and appearance information about the objects in a complex scene; and 2) they can be easily constructed and manipulated by either human users or LLMs with in-context learning, since they are essentially in the form of text sentences [31].

To encode blob representations into blob embeddings in BlobGEN, we first obtain the blob parameter embedding $\boldsymbol{e}_\tau \in \mathbb{R}^{\frac{d}{2}}$ with Fourier feature encoding [37] and the blob description embedding $\boldsymbol{e}_s := [\boldsymbol{e}_{s_1}, \cdots, \boldsymbol{e}_{s_L}] \in \mathbb{R}^{L \times \frac{d}{2}}$ with CLIP text encoder, separately, where $\frac{d}{2}$ denotes the embedding feature size and L denotes the sentence length of blob description. We then concatenate $\boldsymbol{e}_\tau$ with each $\boldsymbol{e}_{s_l}$ along the embedding feature dimension to get $\tilde{\boldsymbol{e}}_l := [\boldsymbol{e}_\tau; \boldsymbol{e}_{s_l}] \in \mathbb{R}^d$ and feed it into an MLP network to get the final blob embeddings $\boldsymbol{e}_{\text{blob}} = \text{MLP}([\tilde{\boldsymbol{e}}_1, ..., \tilde{\boldsymbol{e}}_L]) \in \mathbb{R}^{L \times d}$.

Masked cross-attention. To incorporate the blob representations into the existing text-to-image models, BlobGEN adopts the similar network design of GLIGEN [20] and introduces new masked cross-attention layers in a gated way. Specifically, in the masked cross-attention layer, each blob embedding only attends to visual features in its local region as the visual feature maps are masked by the (rescaled) blob ellipses. Assume there are N blobs in an image, and the visual feature map is denoted by $\boldsymbol{g} \in \mathbb{R}^{hw \times d_g}$, where h and w

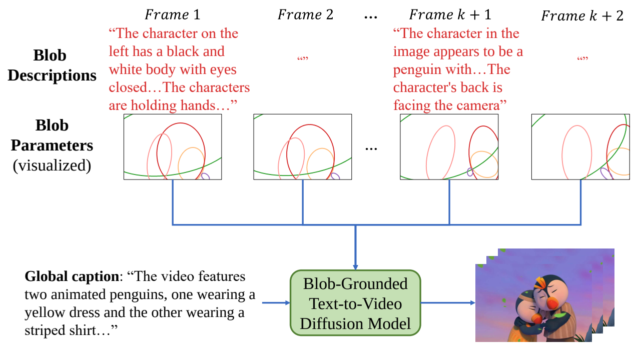

Frame 1
Frame 2
...
Frame $k+1$
Frame $k+2$
Blob Descriptions
"The character on the left has a black and white body with eyes closed…The characters are holding hands…"
""
"The character in the image appears to be a penguin with…The character's back is facing the camera"
""
Blob Parameters (visualized)
Global caption: "The video features two animated penguins, one wearing a yellow dress and the other wearing a striped shirt…"
Blob-Grounded Text-to-Video Diffusion Model

Figure 2. Blob video representations for video generation consist of blob parameters and blob descriptions. Blob parameters exist for every frame while blob descriptions are provided in every k frames. Therefore, only frames $1, k+1$, ... have blob descriptions.

represent the spatial size of the visual features map. We use n as the index for the n^{th} blob, and define the query, key and value (with different linear projections) for cross-attention as $\boldsymbol{q} := \boldsymbol{g}\boldsymbol{W}_q \in \mathbb{R}^{hw \times d_g}$, $\boldsymbol{k}^{(n)} := \boldsymbol{e}_{\text{blob}}^{(n)}\boldsymbol{W}_k^{(n)} \in \mathbb{R}^{L \times d_g}$, and $\boldsymbol{v}^{(n)} := \boldsymbol{e}_{\text{blob}}^{(n)}\boldsymbol{W}_v^{(n)} \in \mathbb{R}^{L \times d_g}$, respectively. The masked cross-attention is defined as

$$\text{MaskCA} := \text{Softmax}\left(\frac{[\boldsymbol{a}^{(1)}; \cdots; \boldsymbol{a}^{(N)}]}{\sqrt{d_g}}\right)[\boldsymbol{v}^{(1)}; \cdots; \boldsymbol{v}^{(N)}] \tag{1}$$

where the n^{th} attention weight for the j^{th} location is:

$$\boldsymbol{a}_j^{(n)} = \begin{cases} \boldsymbol{q}_j\boldsymbol{k}^{(n)T} & \text{if } \boldsymbol{m}_j^{(n)} = 1 \\ -\infty & \text{otherwise} \end{cases} \quad \text{for } j \in \{1, 2, \ldots, hw\}.$$

and the attention mask $\boldsymbol{m}^{(n)} \in \mathbb{R}^{hw}$ is determined by the n^{th} blob ellipse, where its j^{th} value is 1 if a pixel at location j is within the blob ellipse, and 0 otherwise.

With this masking design, each blob representation and its local visual feature are trained to align with each other, and thus the model becomes more disentangled. To retain the prior knowledge of pre-trained models for synthesizing high-quality images, it freezes the weights of the pre-trained diffusion model and only trains the newly added layers.

4. Method

We first describe the extension of BlobGEN to video generation, including new blob representations for the video data and new masked spatial cross-attention layers that fuse blob video representations to video diffusion networks. Furthermore, we introduce new masked 3D attention layers to improve temporal consistency in the object level. Finally, we present blob video generation based on LLMs, which can serve as a stage before BlobGEN-Vid to save human efforts from manually designing layouts.

4.1. Blob representations for videos

Given a video of frame length T, we extract objects from the first frame and track each of the extracted objects in subsequent $T-1$ frames. Accordingly, we obtain a *blob video* of the same frame length that contains N blob ellipses in each frame. Similar to BlobGEN, the n^{th} object's spatial features (including shape, size and location) in the t^{th} frame are depicted by blob parameters $\tau_t^{(n)} := [c_x, c_y, a, b, \theta]$, defined in the same way as Section 3. The blob video captures how the spatial features of each object and their spatial arrangements evolve temporally. On one hand, it can easily capture the object motion in a natural video (e.g., a cat running on the grass), by looking into the relative movement of a blob (e.g., cat) to other blobs (e.g., grass). On the other hand, it can also capture the camera motion by referring to the joint movements and/or deformations of all blobs.

Similar to BlobGEN, we also pair blobs with free-form text descriptions to provide fine-grained details of the local objects. Compared to previous works that use a single class label for each object across frames [19, 42], our blob captions complement the spatial layout with more information such as appearance attributes (color, texture, etc.) and camera focus. Besides, since many visual features of an object may change in a video, it becomes very challenging to use a single blob video caption to describe the object appearance and its dynamic variation across frames. Thus, we opt to apply multiple frame-wise object captions for each blob, which are independently extracted from an existing image captioning model. However, we do not apply blob captions to every object in every single frame because 1) it is neither efficient in the data annotation stage nor convenient for users to construct during inference, and 2) consecutive frames in most videos have little change in objects' visual features. Instead, we assign blob captions at a fixed interval across time, spacing them every k frames.

In summary, our *blob video representations* in a video are comprised of 1) blob parameters $\{\tau_t^{(n)}\}$ for every single frame ($t = 1, 2, \cdots, T$) and every single object ($n = 1, 2, \cdots, N$), and 2) blob descriptions $\{s_{t_k}^{(n)}\}$ for every k frame ($t_k = 1, k+1, \cdots, T$) and every single object ($n = 1, 2, \cdots, N$). Particularly, we denote the frames indexed by t_k as *anchor frames* since they contain both blob parameters and blob descriptions. As we will show later, we can obtain complete context features for other frames through context interpolations based on the blob captions from anchor frames. This design offers consistent contextual information to avoid modality mismatch while applying our blob video representations.

4.2. Blob-grounded text-to-video generation

To incorporate blob video grounding into the pre-trained video diffusion models, we follow the design of Blob-

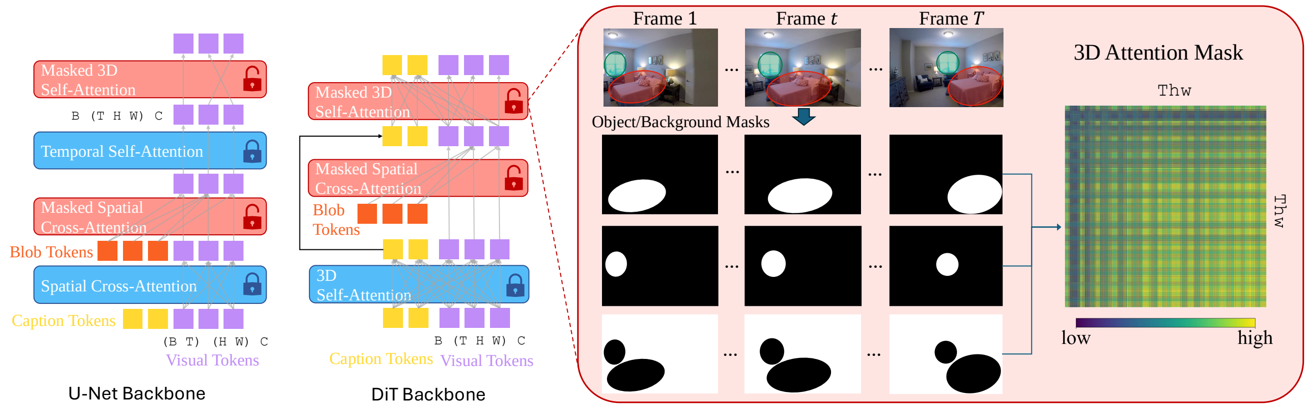


Figure 3. BlobGEN-Vid architecture with U-Net backbone or DiT backbone. Our method leverages two masked attention modules that allows: 1) visual features to attend to only corresponding blobs embeddings; 2) the same object attend to itself across frames. High-value elements in the 3D attention mask in the figure will be mapped to 0 while low-value elements are mapped to $-\infty$ as in Eq. 4. Note the multiple colors in the binary 3D attention mask are from the aliasing issue during visualization.

GEN to add new attention layers to the network. Similarly, we only trained the newly added layers while freezing the weights of pre-trained models. In the following, we introduce the key design choices of BlobGEN-Vid.

Context interpolation. To obtain blob embeddings, we follow BlobGEN to encode blob representations for each single frame independently. That is, for the n^{th} object in the t^{th} frame, we first get the blob parameter embedding $\boldsymbol{e}_{\tau}^{t,n}$ and blob description embedding $\boldsymbol{e}_{s}^{t,n}$, and concatenate them along the embedding feature dimension as input to an MLP network for its blob embedding $\boldsymbol{e}_{\text{blob}}^{t,n}$. However, not all frames are paired with blob captions, which means we do not have blob description embedding $\boldsymbol{e}_{s}^{t,n}$ for those non-anchor frames whose frame index $t \neq t_k$.

A naive approach is to encode an empty text string with CLIP text encoder and use it as the blob description embedding for all non-anchor frames. But it can easily introduce inconsistency across frames due to the large contextual mismatch. To overcome this issue, we propose a simple method called *context interpolation* that linearly interpolates the blob description embeddings of two consecutive anchor frames for each non-anchor frame in the middle. Formally, given the indices of two anchor frames t_k and t_{k+1} where $t_{k+1} = t_k + k$, the interpolated blob description embedding of the non-anchor frame indexed by $t \in (t_k, t_{k+1})$ is given by

$$\boldsymbol{e}_{s}^{t,n} = \frac{t_{k+1} - t}{k} \boldsymbol{e}_{s}^{t_{k+1},n} + \frac{t - t_k}{k} \boldsymbol{e}_{s}^{t_k,n} \tag{2}$$

Intuitively, this linear interpolation ensures a smooth semantic transitioning of object captions across all frames in the CLIP embedding space, leading to better temporal consistency and blob-guided controllability. Besides linear interpolation, some learnable nonlinear interpolations can also be considered. For example, we can train a Perceiver IO network [14] that takes the blob description embeddings of anchor frames as input and learns the blob descriptions embeddings of other frames.

Masked spatial cross-attention. The extension of masked cross-attention from BlobGEN to fuse blob video representations with video features is straightforward. Similar to spatial attention layers in many video diffusion backbones [4], both the visual features and blob embeddings are first reshaped in the form of `(B T (h w) c)` $\rightarrow$ `((B T) (h w) c)` and then they can be fused by applying the masked cross-attention in Eq. (1). That is, we fuse blob embeddings and visual features in the same frame independently for all the frames. This design makes the masked spatial cross-attention layers to solely focus on promoting the frame-wise alignment of generated content and the blob conditioning, without worrying about temporal consistency.

Masked 3D self-attention. The masked spatial cross-attention can only apply per-frame consistency between frames and blobs and cannot guarantee temporal consistency across frames. To improve temporal consistency, we propose new masked 3D self-attention layers to enforce object-level temporal consistency. Note that even though many video diffusion models [1] based on U-Net are equipped with temporal self-attention, it only allows each "pixel" of the visual feature map in a frame to attend to "pixels" at the same spatial location in other frames. However, blobs provide a rough location of each object over time, and thus we can impose stronger coherence by biasing the attention towards the same object over time.

Specifically, in masked 3D self-attention, we flatten all three dimensions in a video feature (i.e., T, h, w) into one dimension and denote the resulting feature as $\boldsymbol{g} \in \mathbb{R}^{Thw \times d}$. Then we obtain query, key and value with three linear projections for self-attention as $\boldsymbol{q} = \boldsymbol{g}\boldsymbol{W}_q$, $\boldsymbol{k} = \boldsymbol{g}\boldsymbol{W}_k$, $\boldsymbol{v} = \boldsymbol{g}\boldsymbol{W}_v$, all in the shape of $\mathbb{R}^{Thw \times d}$. Then, the masked 3D self-attention can be written as:

$$\text{MaskSA3D} := \text{Softmax}(\frac{\boldsymbol{q}\boldsymbol{k}^T}{\sqrt{d}} + \boldsymbol{M}_{\text{blob}})\boldsymbol{v}, \tag{3}$$

where $\boldsymbol{M}_{\text{blob}} \in \mathbb{R}^{Thw \times Thw}$ is a 3D mask determined by blob ellipses across frames, which we describe in the next.

Similar to BlobGEN, we denote the binary blob mask for the n^{th} object in the t^{th} frame as $\boldsymbol{m}^{t,n} \in \mathbb{R}^{hw}$, where its i^{th} entry (denoted as $\boldsymbol{m}_i^{t,n}$) equals 1 if the location i is within the blob ellipse, and 0 otherwise. Besides the N blob masks corresponding to N objects in each frame, we introduce another binary mask, called *background mask*, as $\boldsymbol{m}^{t,\text{bg}} = 1 - \bigcup_{n=1}^{N} \boldsymbol{m}^{t,n}$, resulting in $N+1$ blob masks that cover the whole $(h \times w)$ spatial space. Given any two indices $i, j \in \{1, 2, \cdots, Thw\}$, we then define each entry of $\boldsymbol{M}_{\text{blob}}$ indexed by (i, j) as

$$\boldsymbol{M}_{\text{blob}}^{i,j} = \begin{cases} 0 & \text{if } \boldsymbol{m}_i^{t,n} \wedge \boldsymbol{m}_j^{t',n} = 1, \ \ \forall t, t', n \\ 0 & \text{if } \boldsymbol{m}_i^{t,\text{bg}} \wedge \boldsymbol{m}_j^{t',\text{bg}} = 1, \ \ \forall t, t' \\ -\infty & \text{otherwise} \end{cases} \tag{4}$$

which allows the local object feature for the t frame (depicted by a blob ellipse) to only attend to local features of the same object for another frame (including the t frame itself). Note that each background feature only attends to other background features across frames. Thus, this 3D mask design implies an object-centric self-attention mechanism, leading to better object-level cross-frame consistency. Furthermore, the use of $\boldsymbol{m}^{t,\text{bg}}$ is critical in practical implementation to avoid having all-zero rows in the input to the softmax function and improve training stability.

Fig. 3 shows the overview architecture of BlobGEN-Vid. The introduced two types of attention modules can be inserted into both U-Net and DiT-based diffusion models with minimal modification. We always arrange our masked 3D self-attention after the masked spatial cross-attention as a bottleneck for context feature fusion.

4.3. LLMs for blob generation

Inspired by previous work in using LLMs for layout planning [21–23], we also generate video layouts with in-context learning and structured text. Since video layouts need to expand over time dimension and may have multiple objects per frame, it is important to find a robust structure to represent them. Instead of using self-defined template [38] or stylesheet language [9], we form the layouts as nested dictionaries where frame index, object id, blob parameters and captions are settled in different layers of the structure. LLMs interpret and generate outputs in the same json format that can be directly parsed into blob layouts per frame. In addition, we only generate blobs for a sparse set of frames while interpolate the intermediate blob parameters to make the stage more efficient. We append our detailed in-context prompts in the Appendix.

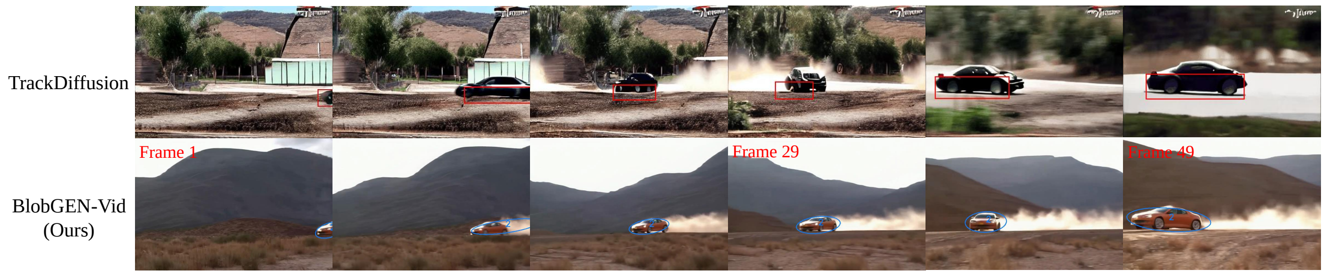
5. Experiment

5.1. Experiment setup

Data preparation. Since there are no available video datasets that provides ground truth blob annotations align with our setting in Sec. 4.1, we build an annotation pipeline to extract blob parameters and captions. In general, we apply Grounding DINO [27] or ODISE [45] to the first frame of each video and obtain segmentation masks. Then we apply SAM2 [33] to every other frame to track the objects. After obtaining all segmentation masks throughout the video, we fit an ellipse to every mask by optimizing the Intersection Over Union (IOU) between the ellipse and the mask area. Using the segmentation masks, we also crop out objects in every eight frames and apply LLaVA-NeXT [26] to get blob captions in these frames.

Our video-text pairs mainly come from OpenVid-1M [30], VidGEN-1M [36], and a small subset of HD-VILA [46]. We apply heavy filtering to each stage of the annotation pipeline to maintain high quality and good balance between human and non-human objects. We end up with $\sim$1M videos having dense blob annotations for training. For indoor scene videos, we use the processed ScanNet++ [50] from BlobGEN-3D [25] where the frame blobs are projected from 3D blobs fitted on the point cloud segmentation extracted from scenes.

Benchmarks and metrics. We have three different experiment settings/domains. First, we test **layout-to-video** generation on 717 validation and test videos of Youtube-VIS 2021 [47]. We report FVD [39] against the ground truth 717 videos for general visual quality, mean Intersection-over-Union (mIOU) for layout controllability, rCLIP$_t$ and rCLIP$_i$ for prompt-video alignment, and regional cross-frame CLIP similarity (rCFC) for object consistency. rCLIP$_t$ is the CLIP cosine similarity between each blob caption and blob-bounded region in the generated videos, and rCLIP$_i$ is the similarity between regions from generated videos and ground truth videos. Secondly, to evaluate **compositionality in T2V** setting, we report results on T2V-CompBench [35] and TC-Bench [8], which evaluate composition changes over time in different aspects. They both equip large multimodal models or detection/tracking models for different aspects. Lastly, we evaluate **multi-view indoor** videos on 1392 test videos from ScanNet++. We report FID and IS for frame quality, FVD



Prompt: "The video shows a car driving on a dirt road, kicking up dust as it speeds along. The car is orange and appears to be a sports model. The road is surrounded by mountains and hills, and the sky is overcast"

Figure 4. Layout-to-video generation results on YoutubeVIS-2021 [47]. The visualized layouts are ground truth layouts fed into the models during inference. Our method shows better prompt-video alignment than the strongest baseline TrackDiffusion [19].

Method	YoutubeVIS-2021[47]				
	FVD ↓	mIOU ↑	rCLIP$_t$ ↑	rCLIP$_i$ ↑	rCFC ↑
TrackDiffusion [19]	464	0.4916	0.2731	0.8041	0.9403
LVD [22]	558	0.2814	0.2613	0.8028	0.8608
VideoTetris [38]	590	0.1658	0.2669	0.7991	0.4231
BlobGEN-Vid (VC2)	396	**0.6119**	0.2794	0.8223	0.9491
BlobGEN-Vid (CogVideoX-5B)	**317**	0.5982	**0.2888**	**0.8364**	**0.9580**

Table 1. Evaluation results of layout-guided video diffusion models on YoutubeVIS-2021 benchmarks.

for video quality, PSNR and cross-frame CLIP similarity (CFC) and rCFC for consistency. For details of our data and evaluation setup, refer to the Appendix.

5.2. Layout-grounded video generation.

As is shown in Table 1, our method outperforms all baselines in nearly all aspects of evaluation. In particular, our method based on VC2 achieves the highest mIOU score (0.6119), over 20% improvement in spatial controllability over TrackDiffusion [19]. BlobGEN-Vid based on CogVideoX-5B achieves a slightly lower mIOU score as it is trained on only half of the dataset but still outperforms all baselines. As for prompt-video alignment, our method outperforms all baselines by achieving 0.2888 rCLIP$_t$ and 0.8364 rCLIP$_i$ scores. As our blob captions are free-form language descriptions of the objects instead of a coarse-grained category name or id, they provide rich semantics to facilitate control of fine-grained details.

We show an qualitative comparison between TrackDiffusion (TD) and our method in Fig. 4. BlobGEN-Vid tightly follow the blobs to generate the car while the objects in TD's video often reach out of the box. In addition, our method also demonstrate better prompt-video alignment by showing "orange sport car" while the baseline cannot control such semantics because it uses the category label "car".

5.3. Text-to-video generation

Table 2 shows the evaluation results of our complete text-to-video generation pipeline by combining GPT-4o and BlobGEN-Vid (CogVideoX-5B-based). We adopt in-context learning methods and input two fixed exemplars to GPT-4o to obtain blob parameters and blob captions for a sparse set of frames. Then we linearly interpolate blob parameters and feed the blob conditions into BlobGEN-Vid to generate videos. Our pipeline outperforms proprietary video generators in four challenging compositional issues, including dynamic attribute binding, spatial relation accuracy, motion binding and numerical accuracy. We show more qualitative examples in the Appendix.

5.4. Multi-view scene generation

For multi-view indoor scene generation, we directly compare to BlobGEN-3D [25] as shown in Table 3. BlobGEN-3D is fine-tuned from BlobGEN [31] with a depth-conditioned ControlNet [52] and a warped previous frame. It is an image diffusion model that generates free-view indoor images in an autoregressive frame-by-frame manner.

We can see BlobGEN-Vid outperforms BlobGEN-3D in all metrics, especially in video consistency. While BlobGEN-3D without depth condition (row 2 and 5) achieves the lowest FID score, it fails to maintain cross-frame consistency as indicated by the low PSNR (10.06) and CFC values (0.9168). When our proposed masked 3D attention is removed from BlobGEN-Vid (comparing row 7 and 8), PSNR and CFC both decrease, justifying the effectiveness of 3D masks in improving consistency.

We also show qualitative comparison in Fig. 5. The red boxes annotate inconsistent objects in the generated image sequences. BlobGEN-Vid without masked 3D self-attention tend to generate the door in different colors as the camera pose changes, while BlobGEN-Vid generate more consistent appearance of the door.

5.5. Ablation study

In Table 4, we present the ablation study on three factors: Masked 3D self-attention, context interpolation method, and training data. We observe that adding the masked 3D self-attention is crucial to facilitate video diffusion models

Method	**T2V-CompBench [35]**					**TC-Bench [8]**					
	Consist-Attr ↑	Dynamic-Attr ↑	Spatial ↑	Motion ↑	Numeracy ↑	Attr. Transition		Obj. Relation		Background Shift	
						TCR ↑	TC-Score ↑	TCR ↑	TC-Score ↑	TCR ↑	TC-Score ↑
Open-Sora v1.2 [54]	0.6600	0.1714	0.5406	0.2388	0.2556	6.15	0.6509	7.66	0.7406	2.35	0.5847
CogVideoX-5B [49]	-	-	-	-	-	8.08	0.6930	10.64	0.7237	4.71	0.6338
LVD [22] (w/ GPT-4)	0.5595	0.1499	0.5469	0.2699	0.0991	5.77	0.6215	**12.77**	0.7081	1.96	0.5042
VideoTetris [38] (w/ LLM)	0.7125	0.2066	0.5148	0.2204	0.2609	-	-	-	-	-	-
Pika 1.0	0.6513	0.1744	0.5043	0.2221	0.2613	5.77	0.6520	8.51	0.7242	1.96	0.6070
Dream Machine	0.6900	0.2002	0.5387	0.2713	0.2109	9.80	0.7319	**12.77**	0.7755	5.88	0.6284
Kling 1.0	**0.8045**	0.2256	0.6150	0.2448	0.3044	7.69	0.6888	10.64	0.7819	3.92	0.6183
Gen-3 Alpha	0.7045	0.2078	0.5533	0.3111	0.2169	9.62	**0.7507**	10.64	0.7073	**27.45**	**0.7488**
GPT-4o + BlobGEN-Vid (Ours)	0.7400	**0.2650**	**0.6725**	**0.3880**	**0.3910**	**15.39**	0.7055	**12.77**	**0.7944**	10.42	0.6852

Table 2. Comparison between BlobGEN-Vid and major proprietary text-to-video diffusion models/systems on two benchmarks emphasizing video compositionality: T2V-CompBench [35] and TC-Bench [8].

	Method	**Image-based Metrics**		
		FID ↓	IS ↑	CLIP Sim. ↑
1	BlobGEN-3D (blob only)	**21.24**	5.38	0.2301
2	BlobGEN-3D [25]	31.28	5.02	0.2231
3	BlobGEN-Vid w/o Mask 3D Attn	30.72	5.66	0.2319
4	BlobGEN-Vid (Ours)	27.94	**5.70**	**0.2320**

		Video-based Metrics			
		FVD ↓	PSNR ↑	CFC ↑	rCFC ↑
5	BlobGEN-3D (blob only)	335	10.06	0.9168	0.9197
6	BlobGEN-3D [25]	468	15.23	0.9322	0.9347
7	BlobGEN-Vid w/o Mask 3D Attn	**142**	21.71	0.9432	0.9424
8	BlobGEN-Vid (Ours)	161	**22.20**	**0.9453**	**0.9456**

Table 3. Evaluation results on ScanNet++ [50] test split with image and video metrics. BlobGEN-Vid is based on VC2.

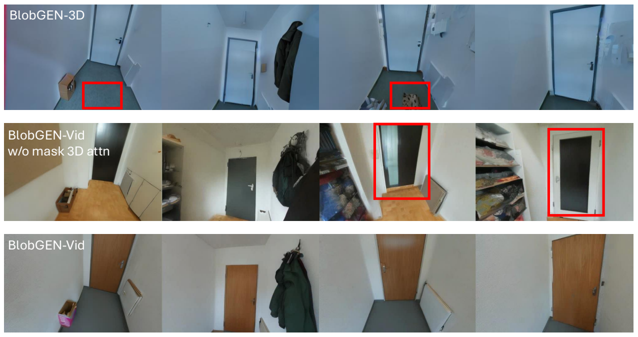


Figure 5. Qualitative results on ScanNet++, where our method, especially with masked 3D attention, shows much better consistency in the door appearance than BlobGEN-3D.

	Mask 3D Attn	Context Interp.	Training Data	FVD ↓	mIOU ↑	rCLIP$_t$ ↑	rCLIP$_i$ ↑	rCFC ↑
1		Linear	160K	617	0.2585	0.2697	0.7961	0.9232
2	✓	Linear	160K	368	0.5623	0.2804	0.8211	**0.9500**
3	✓	Linear	400K	**346**	0.5771	0.2794	0.8200	0.9480
4	✓	Slerp	400K	378	0.5702	0.2763	0.8142	0.9438
5	✓	Perceiver	400K	352	0.5926	**0.2806**	0.8204	0.9459
6	✓	Perceiver	1M	396	**0.6119**	0.2794	**0.8223**	0.9491

Table 4. Ablation study on model architecture, interpolation method and training data size. The base model is VideoCrafter2.

to generate consistent objects, as indicated by the significant discrepancy between rows 1 and 2 in all metrics. Specifically, adding masked 3D attention improves FVD by 40% and mIOU by 117% with the same amount of data and interpolation method.

As for context interpolation, we investigate three different approaches including linear, slerp and PerceiverIO. While slerp has been widely used in GANs to interpolate latent features [15], we do not find any advantage of it in our setting. We conjecture that the embedding space of CLIP text encoder may have a different topology that makes it less effective. Linear and perceiver-based interpolation illustrate slightly different behavior. The former achieves a larger rCFC value while PerceiverIO shows stronger layout controllability as indicated by the mIOU values.

Finally, we do observe the effectiveness of scaling training data from 160K to 400K and eventually to 1M videos. The effects are mainly reflected as stronger layout controllability as indicated by mIOU values. Models trained with 400K videos (row 3-5) all show better mIOU compared to row 1-2. Increasing the data to 1M (row 6) further boosts mIOU by 3.2% and rCLIP$_i$ and rCFC as well. It is important to realize that FVD may not be a robust metric [10] and the fact that higher rCFC values do not necessarily indicate better performance [8].

6. Conclusions

In this work, we propose a new layout design for text-to-video generation, called blob representations. The representation contains blob parameters for each object in every frame and paired blob captions in a sparse set of frame. Our free-form blob captions also provide more fine-grained semantics of each object. We then introduce a framework termed BlobGEN-Vid that endows video diffusion models with the ability to condition on blob inputs. BlobGEN-Vid consists of a context interpolation module, allowing more flexible semantic transition, and masked 3D attention blocks to enforce object consistency across frames. We demonstrate the effectiveness of our frame in multiple visual domains and settings. BlobGEN-Vid achieves strong performance in open-domain video generation and multi-view image generation. When combined with an LLM, it shows great potential in compositionality and outperforms proprietary video generators in multiple aspects.

References

[1] 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, 5

[2] 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22563–22575, 2023. 2, 3

[3] Haoxin Chen, Menghan Xia, Yingqing He, Yong Zhang, Xiaodong Cun, Shaoshu Yang, Jinbo Xing, Yaofang Liu, Qifeng Chen, Xintao Wang, et al. Videocrafter1: Open diffusion models for high-quality video generation. *arXiv preprint arXiv:2310.19512*, 2023. 3

[4] Haoxin Chen, Yong Zhang, Xiaodong Cun, Menghan Xia, Xintao Wang, Chao Weng, and Ying Shan. Videocrafter2: Overcoming data limitations for high-quality video diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7310–7320, 2024. 2, 3, 5, 13

[5] Minghao Chen, Iro Laina, and Andrea Vedaldi. Training-free layout control with cross-attention guidance. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 5343–5353, 2024. 2

[6] Tsai-Shien Chen, Aliaksandr Siarohin, Willi Menapace, Ekaterina Deyneka, Hsiang-wei Chao, Byung Eun Jeon, Yuwei Fang, Hsin-Ying Lee, Jian Ren, Ming-Hsuan Yang, et al. Panda-70m: Captioning 70m videos with multiple cross-modality teachers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13320–13331, 2024. 2, 3

[7] Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. In *Forty-first International Conference on Machine Learning*, 2024. 3

[8] Weixi Feng, Jiachen Li, Michael Saxon, Tsu-jui Fu, Wenhu Chen, and William Yang Wang. Tc-bench: Benchmarking temporal compositionality in text-to-video and image-to-video generation. *arXiv preprint arXiv:2406.08656*, 2024. 3, 6, 8, 15

[9] Weixi Feng, Wanrong Zhu, Tsu-jui Fu, Varun Jampani, Arjun Akula, Xuehai He, Sugato Basu, Xin Eric Wang, and William Yang Wang. Layoutgpt: Compositional visual planning and generation with large language models. *Advances in Neural Information Processing Systems*, 36, 2024. 3, 6

[10] Songwei Ge, Aniruddha Mahapatra, Gaurav Parmar, Jun-Yan Zhu, and Jia-Bin Huang. On the content bias in fréchet video distance. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7277–7288, 2024. 8

[11] Amir Hertz, Ron Mokady, Jay Tenenbaum, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. Prompt-to-prompt image editing with cross attention control. *arXiv preprint arXiv:2208.01626*, 2022. 13

[12] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020. 2, 3

[13] Wenyi Hong, Ming Ding, Wendi Zheng, Xinghan Liu, and Jie Tang. Cogvideo: Large-scale pretraining for text-to-video generation via transformers. In *The Eleventh International Conference on Learning Representations*, 2023. 2

[14] Andrew Jaegle, Sebastian Borgeaud, Jean-Baptiste Alayrac, Carl Doersch, Catalin Ionescu, David Ding, Skanda Koppula, Daniel Zoran, Andrew Brock, Evan Shelhamer, et al. Perceiver io: A general architecture for structured inputs & outputs. *arXiv preprint arXiv:2107.14795*, 2021. 5, 12

[15] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4401–4410, 2019. 8

[16] PKU-Yuan Lab and Tuzhan AI etc. Open-sora-plan, 2024. 3

[17] Guillaume Le Moing, Jean Ponce, and Cordelia Schmid. Dense optical tracking: Connecting the dots. In *CVPR*, 2024. 15

[18] Jiachen Li, Weixi Feng, Tsu-Jui Fu, Xinyi Wang, Sugato Basu, Wenhu Chen, and William Yang Wang. T2v-turbo: Breaking the quality bottleneck of video consistency model with mixed reward feedback. *arXiv preprint arXiv:2405.18750*, 2024. 2, 3

[19] Pengxiang Li, Zhili Liu, Kai Chen, Lanqing Hong, Yunzhi Zhuge, Dit-Yan Yeung, Huchuan Lu, and Xu Jia. Trackdiffusion: Multi-object tracking data generation via diffusion models. *arXiv preprint arXiv:2312.00651*, 2023. 2, 3, 4, 7, 15

[20] 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22511–22521, 2023. 2, 3

[21] Long Lian, Boyi Li, Adam Yala, and Trevor Darrell. Llm-grounded diffusion: Enhancing prompt understanding of text-to-image diffusion models with large language models. *Transactions on Machine Learning Research*, 2024. 6

[22] Long Lian, Baifeng Shi, Adam Yala, Trevor Darrell, and Boyi Li. Llm-grounded video diffusion models. In *The Twelfth International Conference on Learning Representations*, 2024. 2, 3, 7, 8, 15

[23] Han Lin, Abhay Zala, Jaemin Cho, and Mohit Bansal. Videodirectorgpt: Consistent multi-scene video generation via llm-guided planning. *arXiv preprint arXiv:2309.15091*, 2023. 2, 3, 6

[24] Pengyang Ling, Jiazi Bu, Pan Zhang, Xiaoyi Dong, Yuhang Zang, Tong Wu, Huaian Chen, Jiaqi Wang, and Yi Jin. Motionclone: Training-free motion cloning for controllable video generation. *arXiv preprint arXiv:2406.05338*, 2024. 13

[25] Chao Liu, Weili Nie, Sifei Liu, Abhishek Badki, Hang Su, Morteza Mardini, Benjamin Eckart, and Arash Vahdat.

Blobgen-3d: Compositional 3d-consistent freeview image generation with 3d blobs. In *SIGGRAPH Asia 2024 Conference Papers*, 2024. 3, 6, 7, 8

[26] Haotian Liu, Chunyuan Li, Yuheng Li, Bo Li, Yuanhan Zhang, Sheng Shen, and Yong Jae Lee. Llava-next: Improved reasoning, ocr, and world knowledge, 2024. 6, 12, 15

[27] Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Qing Jiang, Chunyuan Li, Jianwei Yang, Hang Su, et al. Grounding dino: Marrying dino with grounded pre-training for open-set object detection. *arXiv preprint arXiv:2303.05499*, 2023. 6, 12, 14, 15

[28] Yaofang Liu, Xiaodong Cun, Xuebo Liu, Xintao Wang, Yong Zhang, Haoxin Chen, Yang Liu, Tieyong Zeng, Raymond Chan, and Ying Shan. Evalcrafter: Benchmarking and evaluating large video generation models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22139–22149, 2024. 3

[29] Yuanxin Liu, Lei Li, Shuhuai Ren, Rundong Gao, Shicheng Li, Sishuo Chen, Xu Sun, and Lu Hou. Fetv: A benchmark for fine-grained evaluation of open-domain text-to-video generation. *Advances in Neural Information Processing Systems*, 36, 2024. 3

[30] Kepan Nan, Rui Xie, Penghao Zhou, Tiehan Fan, Zhenheng Yang, Zhijie Chen, Xiang Li, Jian Yang, and Ying Tai. Openvid-1m: A large-scale high-quality dataset for text-to-video generation. *arXiv preprint arXiv:2407.02371*, 2024. 6, 13

[31] Weili Nie, Sifei Liu, Morteza Mardani, Chao Liu, Benjamin Eckart, and Arash Vahdat. Compositional text-to-image generation with dense blob representations. In *Forty-first International Conference on Machine Learning*, 2024. 2, 3, 7

[32] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4195–4205, 2023. 2, 3

[33] Nikhila Ravi, Valentin Gabeur, Yuan-Ting Hu, Ronghang Hu, Chaitanya Ryali, Tengyu Ma, Haitham Khedr, Roman Rädle, Chloe Rolland, Laura Gustafson, et al. Sam 2: Segment anything in images and videos. *arXiv preprint arXiv:2408.00714*, 2024. 6, 12, 14

[34] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022. 2

[35] Kaiyue Sun, Kaiyi Huang, Xian Liu, Yue Wu, Zihan Xu, Zhenguo Li, and Xihui Liu. T2v-compbench: A comprehensive benchmark for compositional text-to-video generation. *arXiv preprint arXiv:2407.14505*, 2024. 3, 6, 8, 15

[36] Zhiyu Tan, Xiaomeng Yang, Luozheng Qin, and Hao Li. Vidgen-1m: A large-scale dataset for text-to-video generation. *arXiv preprint arXiv:2408.02629*, 2024. 6, 13

[37] Matthew Tancik, Pratul Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains. *Advances in neural information processing systems*, 33:7537–7547, 2020. 3

[38] Ye Tian, Ling Yang, Haotian Yang, Yuan Gao, Yufan Deng, Jingmin Chen, Xintao Wang, Zhaochen Yu, Xin Tao, Pengfei Wan, et al. Videotetris: Towards compositional text-to-video generation. *arXiv preprint arXiv:2406.04277*, 2024. 3, 6, 7, 8, 15

[39] Thomas Unterthiner, Sjoerd Van Steenkiste, Karol Kurach, Raphael Marinier, Marcin Michalski, and Sylvain Gelly. Towards accurate generative models of video: A new metric & challenges. *arXiv preprint arXiv:1812.01717*, 2018. 6

[40] Arash Vahdat, Karsten Kreis, and Jan Kautz. Score-based generative modeling in latent space. *Advances in neural information processing systems*, 34:11287–11302, 2021. 3

[41] Jiuniu Wang, Hangjie Yuan, Dayou Chen, Yingya Zhang, Xiang Wang, and Shiwei Zhang. Modelscope text-to-video technical report. *arXiv preprint arXiv:2308.06571*, 2023. 3

[42] Jiawei Wang, Yuchen Zhang, Jiaxin Zou, Yan Zeng, Guoqiang Wei, Liping Yuan, and Hang Li. Boximator: Generating rich and controllable motions for video synthesis. In *Forty-first International Conference on Machine Learning*, 2024. 2, 3, 4, 12

[43] Yaohui Wang, Xinyuan Chen, Xin Ma, Shangchen Zhou, Ziqi Huang, Yi Wang, Ceyuan Yang, Yinan He, Jiashuo Yu, Peiqing Yang, et al. Lavie: High-quality video generation with cascaded latent diffusion models. *arXiv preprint arXiv:2309.15103*, 2023. 3

[44] Jun Xu, Tao Mei, Ting Yao, and Yong Rui. Msr-vtt: A large video description dataset for bridging video and language. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5288–5296, 2016. 2

[45] Jiarui Xu, Sifei Liu, Arash Vahdat, Wonmin Byeon, Xiaolong Wang, and Shalini De Mello. Open-vocabulary panoptic segmentation with text-to-image diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2955–2966, 2023. 6, 12

[46] Hongwei Xue, Tiankai Hang, Yanhong Zeng, Yuchong Sun, Bei Liu, Huan Yang, Jianlong Fu, and Baining Guo. Advancing high-resolution video-language representation with large-scale video transcriptions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5036–5045, 2022. 2, 3, 6, 13

[47] Linjie Yang, Yuchen Fan, Yang Fu, and Ning Xu. The 3rd large-scale video object segmentation challenge - video instance segmentation track, 2021. 6, 7

[48] Xingyi Yang and Xinchao Wang. Compositional video generation as flow equalization. *arXiv preprint arXiv:2407.06182*, 2024. 3

[49] Zhuoyi Yang, Jiayan Teng, Wendi Zheng, Ming Ding, Shiyu Huang, Jiazheng Xu, Yuanming Yang, Wenyi Hong, Xiaohan Zhang, Guanyu Feng, et al. Cogvideox: Text-to-video diffusion models with an expert transformer. *arXiv preprint arXiv:2408.06072*, 2024. 2, 3, 8, 13

[50] Chandan Yeshwanth, Yueh-Cheng Liu, Matthias Nießner, and Angela Dai. Scannet++: A high-fidelity dataset of 3d indoor scenes. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12–22, 2023. 6, 8

[51] David Junhao Zhang, Jay Zhangjie Wu, Jia-Wei Liu, Rui Zhao, Lingmin Ran, Yuchao Gu, Difei Gao, and Mike Zheng Shou. Show-1: Marrying pixel and latent diffusion models for text-to-video generation. *International Journal of Computer Vision*, pages 1–15, 2024. 3

[52] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3836–3847, 2023. 2, 7

[53] Yuanhan Zhang, Bo Li, haotian Liu, Yong jae Lee, Liangke Gui, Di Fu, Jiashi Feng, Ziwei Liu, and Chunyuan Li. Llava-next: A strong zero-shot video understanding model, 2024. 12

[54] Zangwei Zheng, Xiangyu Peng, Tianji Yang, Chenhui Shen, Shenggui Li, Hongxin Liu, Yukun Zhou, Tianyi Li, and Yang You. Open-sora: Democratizing efficient video production for all, 2024. 3, 8

BlobGEN-Vid: Compositional Text-to-Video Generation with Blob Video Representations

Supplementary Material

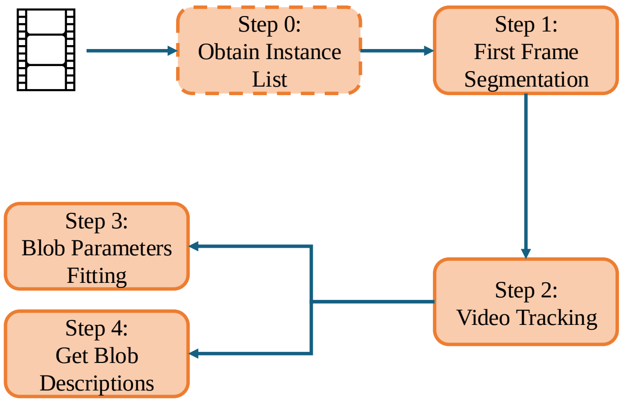


Figure 6. Our data annotation pipeline for obtaining blob video representations consists of five steps with step 0 being optional.

A. Data annotation

Data annotation pipeline. Our data annotation pipeline consists of four to five key steps as shown in Fig. 6. The step 0 is to obtain a list of objects appeared in the video using a VLM. Though previous methods [42] directly use a language parser to extract object nouns or phrases from video captions, we found the parser often extracts words that do not represent concrete entities and introduces additional noise to step 1. Thus we feed videos into LLaVA-NeXT-Video-7B [53] and prompt it to generate a list of objects that appear in each video. Using the instant list, we could apply Grounding DINO [27] in step 1 to obtain segmentation masks for the first frame of the video. We also experiment with ODISE [45], which is a panoptic segmentation model that does not require the instance list from step 0 to work. For most videos, we first apply LLaVA-NeXT+Grounding DINO to get segmentation masks. If the mask coverage is below 20% of the frame size, we apply ODISE to get more dense panoptic annotation. This helps us keep most of the videos for further annotation and hence improve data utilization rate.

After obtaining the segmentation masks in step 1, we apply SAM2 [33] to track each object mask throughout the video. To make the process efficient, we uniformly sample 1/4 of all the frames to do tracking. With the tracking masks for the frames, we can fit a set of blob parameters (c_x, c_y, a, b, θ) for each mask. For frames without tracking masks, we linearly interpolate the blob parameters from the closest neighboring frames. As for step 4, we crop a tight rectangle region around each segmentation mask, and feed it to LLaVA-v1.6-mistral-7b [26] to get blob descriptions. For efficiency, we only annotate blob descriptions for the first of every eight frames.

Qualitative visualization. We visualize two example of our data annotation results in Fig. 10 (using Grounding DINO) and Fig. 11 (using ODISE). We observe that the instance list obtained from LLaVA-NeXT-Video-7B [53] usually contain instance names in different hierarchical levels. For example, in Fig. 11, the hat, scarf, and yellow jacket are listed as separate objects. Sometimes, the model would also list "hands" as separate objects from the whole human figure. However, it has the drawback of neglecting background objects even though we explicitly emphasize "both foreground and background" objects in the prompt.

In contrast, ODISE [45] has more fine-grained segmentation of background since it applies a long list of category names merged from different datasets. As is shown in Fig. 11, ODISE segments the background into four different parts, including the sky, trees, grass and fence. However, ODISE's category set does not include some general objects or hierarchical parts of objects like "cartoon character" or "hands/arms" compared to using instance list. In addition, the segmentation labels from ODISE can be less accurate. For example, it annotates the "cartoon monkey" as "costume" and the "brown bag" as "suitcase". The issue is mitigated as we use an VLM to obtain free-form blob descriptions instead of adopting ODISE labels.

B. BlobGEN-Vid framework

Context interpolation module. As shown in Fig. 7, we first encode the blob descriptions in all frames. For non-anchor frames, we use empty strings for the encoding process and later replace them with the learned features. If an object undergoes apparent semantic change (e.g. object changing color), the blob description in the anchor frames would have different meaning, reflected as the color differences in the penultimate feature sequence from CLIP text encoder. Apart from the linear interpolation introduced in Sec. 4.2, we also experiment with a learnable module using PerceiverIO [14]. To ensure object-wise interpolation, we reshape the context embeddings as `(B T N L d)` $\rightarrow$ `((B N) (T L) d)` where T denotes the number of latent frames and L is the sequence length of the context features. In Fig. 7, we have omitted B, N and use $L = 3$ and $T = 9$ for demonstration purpose. In our implementation,

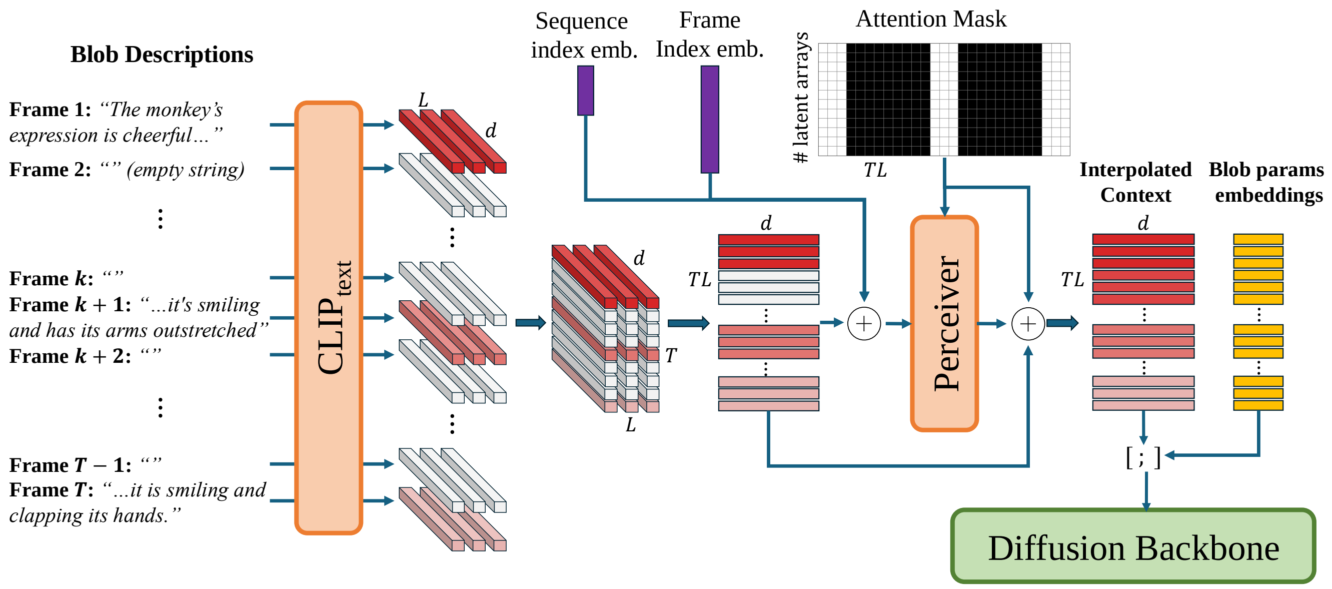


Figure 7. An illustration of the context interpolation stage using a Perceiver-based model. Note that we omit the batch size (B) and number of blobs per frame dimension (N) for simplicity. After the CLIP text encoder encodes each blob description, we merge time T and context sequence length L into one dimension and learn the context for non-anchor frames through the Perceiver module. The attention mask prevents the latent arrays to attend on blob descriptions that are empty strings, implying that Perceiver only relies on anchor frames' text embeddings to infer intermediate text embeddings.

the CLIP text encoder outputs $L = 77$ context features and there are $T = 13$ (CogVideoX) or $T = 16$ (VC2) frames. While PerceiverIO was originally proposed to handle inputs of different modalities, we adopt it for the sake of simplicity and flexibility, as it allows arbitrary number of anchor frames. It facilitates handling arbitrary number and locations of the anchor frames on users' choices in the future.

DiT attention maps. While the attention maps from UNet-based image/video diffusion models are shown to reflect the spatial structure of the pixel-space outputs [11, 24], such property in DiT-based video diffusion model [49] with full 3D attention has never been proved, to the best of our knowledge. Here we show that such property still exists in full 3D attentions, which justifies our choice to add masked spatial cross-attention for per-frame context injection.

In Fig. 8, we show the attention maps between *a visual token* and the visual tokens of Frame 1. In Fig. 9, we show the attention maps between *a text token* and the visual tokens of Frame 1. Some of the highlighted regions look highly similar as the pixel space structure, proving that even in full 3D attention, the flattened visual tokens still preserves spatial structure. The phenomenon is similar as those observed in UNet-based diffusion models where the spatial and temporal attentions are separated. Please refer to CogVideoX [49] for details of the input and output of the full 3D attentions.

C. Implementation details

Training data. For open-domain video generation, our training dataset is obtained by annotating 160K OpenVid [30] videos, 460K VIDGEN [36] videos and 320K videos from HDVILA [46]. We try to maintain a good balance between human video and non-human videos where the latter outweighs the former as human figures are more challenging to synthesize. While all 940K videos are used for VideoCrafter2-based training, only half of the videos (∼500K) satisfy the length requirement of CogVideoX. Therefore, the training dataset size for BlobGEN-Vid based on CogVideoX is effectively ∼500K videos.

For the multi-view scene experiment, we use the ScanNet++ dataset, consisting of 1130 training video clips where each clip has 128 frames. As VideoCrafter2 generates videos of 16 frames, we sample 16 consecutive frames from the 128 frames with a stride of 8. Therefore, we obtain 15 sub-clips with overlaps from each 128-frame video. We prepare 517 16-frame clips for validation purpose and 1392 16-frame clips as the testing set for final evaluation.

Model Architecture and Training. We fine-tuned both VideoCrafter2 (VC2) [4] (U-Net) and CogVideoX-5B [49] (DiT) with our annotated datasets. For VC2 fine-tuning, we add one masked spatial cross-attention in every spatial transformer block and one masked 3D attention after temporal transformer blocks where the latent feature has a spa-

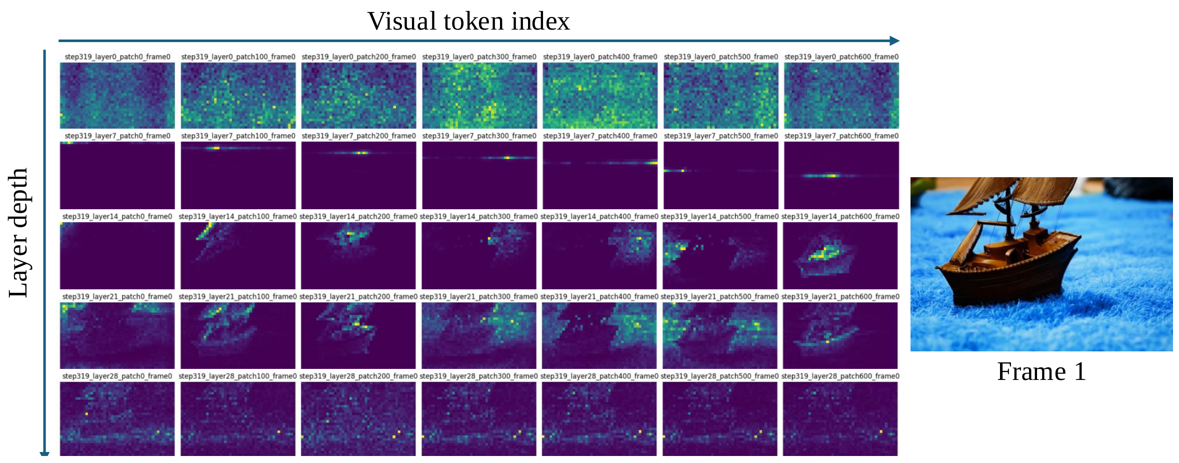


Figure 8. Attention maps between *a visual token* and other visual tokens of the first frame. Some of the maps show similar spatial structure as Frame 1 in the pixel space. The visualization proves that full 3D attention still preserves the spatial structure as in UNet-based diffusion models.

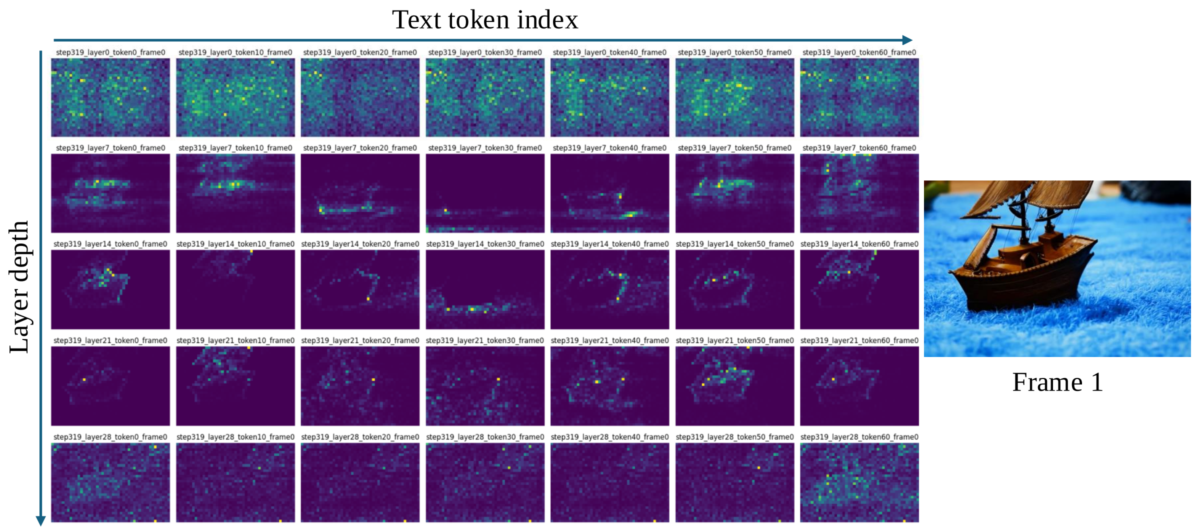


Figure 9. Attention maps between *a text token* and other visual tokens of the first frame. The highlighted regions represent the regions where the text token is highly correlated with. The visualization proves that full 3D attention still preserves the spatial structure as in UNet-based diffusion models.

tial resolution less than or equal to 32×32. For CogVideoX, we add one masked spatial cross-attention after every three DiT blocks and one masked 3D self-attention after every six DiT blocks. For open-domain and indoor videos, we fine-tuned VC2 with a learning rate 1e-4 for 20k steps with batch size 256 and 1000 warmup steps. We fine-tuned CogVideoX on open-domain videos with a learning rate 5e-5 for 6k steps. All training processes are done on 64 or 128 NVIDIA A100 GPUs.

Evaluation metrics. For **layout-to-video generation evaluation**, we apply the following metrics: FVD, mean Intersection-over-Union (mIOU), rCLIP$_t$, rCLIP$_i$ and cCFC. To compute mIOU, we first apply Grounding DINO [27]+SAM2 [33] using the ground truth object labels to obtain object bounding boxes (bboxes) per frame. Then we compute the IOU between the detected bbox in a frame with the ground truth bbox in that frame for the same object. mIOU is the average IOU value over all objects in all involved frames of all videos. If the number of objects from detection and tracking does not match with the number of objects in the ground truth annotation, we keep the most confident detection results up to the number of ground truth bboxes. Then we match each detection bbox to a unique

	Mask 3D Attn	Context Interp.	Training Data	FVD ↓	mIOU ↑	rCLIP$_t$ ↑	rCLIP$_i$ ↑	rCFC ↑
1	✓	None	400K	379	0.5614	0.2767	0.8161	0.9466
2	✓	Linear	400K	**346**	0.5771	0.2794	0.8200	**0.9480**
3	✓	Slerp	400K	378	0.5702	0.2763	0.8142	0.9438
4	✓	Perceiver	400K	352	**0.5926**	**0.2806**	**0.8204**	0.9459

Table 5. Ablation study on model architecture, context interpolation method and training data size on YTVIS-700. In the highlighted row, no interpolation method is applied. For frames without blob descriptions, we input empty strings to CLIP text encoder to get context features. We can see a consistent performance drop without the context interpolation, as indicated by all five metrics.

ground truth bbox that produces the highest possible IOU value. For rCLIP$_t$, we crop out regions using the ground truth bboxes. If the region is paired with a blob description, we use CLIP to compute the cosine similarity between the visual region and the blob description. The average similarity score over all videos, all involved frames and all objects give out the rCLIP$_t$ value. rCLIP$_i$ is computed in a similar way but using the bbox region from the ground truth video frame instead of the blob descriptions. It usually has a higher value because the compared features lie in the same output space of CLIP image encoder. As for rCFC, we utilize the detection+tracking results from mIOU and crop out the bbox regions of each object in every frame. Then we compute the cosine similarity between two regions of the same object from two consecutive frames. For one object in a generated video with T frames, this ends up with $T-1$ rCFC values. The reported rCFC is the average value over all detected objects and all videos.

Note that different methods condition on layouts in different number of frames. Therefore, for a fair comparison, we compute mIOU, rCLIP$_t$, and rCLIP$_i$ only on the frames with the layout condition. For TrackDiffusion [19], all 16 generated frames are involved as all frames are grounded on input layouts. For LVD [22], Frame 1, 4, 7, 10, 13, 16 of all 16 frames are involved. For VideoTetris [38], Frame 9, 17, 25 of all 32 frames are involved. For BlobGEN-Vid based on VC2, we compute the metrics on all 16 frames. For BlobGEN-Vid based on CogVideoX, we evaluate on Frame $4k+1$ where $k = 0, 1, ..., 12$ out of 49 frames, because there are 13 latent frames due to the $4\times$ temporal expansion rate from the VAE decoder.

For **text-to-video generation evaluation** on T2V-CompBench [35] and TC-Bench [8], we adopt the official evaluation metrics. In summary, T2V-CompBench applies different computation methods for different dimension. Consistent Attribute Binding (Consist.-Attr.) and Dynamic Attribute Binding (Dynamic-Attr.) applies LLaVA-v1.6-34B [26] to evaluate the attribute correctness. Spatial and Numeracy accuracy are computed using GroundingSAM [27] to locate and count the objects. Motion Binding is computed using GroundingSAM and Dense Optical Tracking [17]. TC-Bench adopts GPT-4 Turbo to answer a list of assertion questions related to compositions of the video. TCR is the percentage (%) of videos with all assertions passed and TC-Score is the ratio of assertions passed. For details of these metrics, we refer our readers to the original papers [8, 35].

For **multi-view image generation** in indoor scenes, we compute FID, IS, and CLIP Similarity for image-based metrics, FVD, PSNR, CFC, and rCFC for video-based metrics. CLIP Similarity refer to the average CLIP cosine similarity between each frame and the global caption of the scene. For PSNR, we warp the last frame to an image under current camera view. Then we compute the PSNR between the warped frame and the generated current frame for the regions with content, which reflects a global consistency between two frames. CFC is the average CLIP cosine similarity between any two consecutive frames in the generated videos. rCFC adopts the ground truth annotation and computes the CLIP cosine similarity between two regions of the same object from two consecutive frames. CFC and rCFC reflects video consistency in different granularity levels.

In-context learning examples. We show the full prompt and our in-context exemplars in Table 6. The layouts follow a JSON format which allows GPT-4o to produce outputs that can be robustly parsed by a JSON parser. We use two fixed exemplars for all prompts in our text-to-video generation experiments. While this simplest in-context design can lead to many flaws in the generated layouts, our pipeline of GPT-4o+BlobGEN-Vid still demonstrates strong performances in many compositional aspects, suggesting great potential in further improving the performance by more sophisticated layout generation approaches.

D. Additional Results

Ablation study. In Table 5, we show the ablation study on context interpolation methods. We emphasize the importance of context interpolation by comparing row 1 with other rows. For "None" interpolation method, we simply use empty strings for frames without blob descriptions and obtain context features from CLIP text encoder. Note that this has led to apparent performance drop in all metrics compared to using simple linear interpolation in row 2. Therefore, the existence of interpolation for context features is essential to generate consistent videos and enhance prompt-video alignment.

D.1. Additional qualitative results

We show additional qualitative results from various settings and benchmarks in Fig. 12-20.

Generate a video layout using ellipses for the given user prompt. Each ellipse should be represented with five parameters and a paired object caption. The parameters are [cx, cy, a, b, theta] where cx and cy are the center coordinates, a and b are the major and minor axes length, and theta is the rotation angle. Assume there are 13 frames in the video, and you should generate layouts for Frame0,2,4,...,12. The video resolution is 720 width and 480 height. Try to cover all objects mentioned in the prompt. You should follow the format of the following examples:

Example 1:
Prompt: The video shows a small owl perched on a branch, looking around. It appears to be in a natural habitat, surrounded by greenery. The owl is alert and focused, possibly observing its surroundings or looking for prey. The camera angle is from below, giving a clear view of the owl's feathers and features.

```json
“Frame0”: “Object2”: “blob”: [443, 252, 102, 72, -2.353],
“caption”: “The bird in the close-up image is a small, brown creature with a white belly. It appears to be in mid-flight, with its wings spread wide and its tail fanned out. The bird is perched on a tree branch, which is covered in green leaves. The bird“s eyes are open, and it seems to be looking directly at the camera. ”,
“Frame2”: “Object2”: “blob”: [438, 253, 106, 68, -2.357],
“caption”: “ The bird in the close-up image is a small, brown and white bird with a prominent beak, perched on a tree branch. The bird turns its head to the side.”,
...
“Frame12”: “Object2”: “blob”: [445, 249, 119, 57, -2.023],
“caption”: “ The bird in the close-up image is a small owl perched on a tree branch. The bird is looking upwards, turning its face away from the camera. ”
```

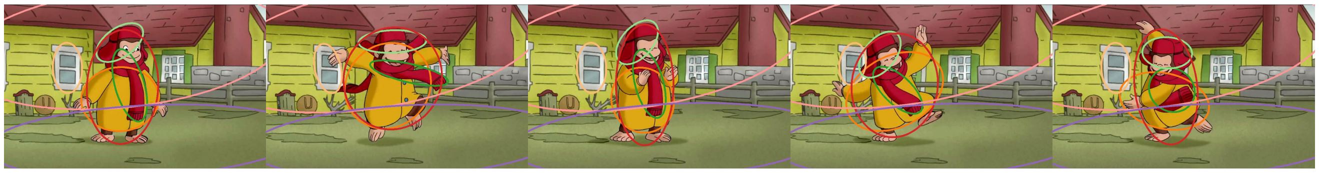
Example 2:
Prompt: The video shows a woman leading a horse while a young girl rides on its back. The girl is wearing a helmet and a riding jacket, and the woman is holding the reins. They are in a stable or a similar outdoor area with several parked cars in the background.

```json
“Frame0”: “Object2”: “blob”: [365, 277, 93, 64, 1.749],
“caption”: “The horse in the close-up image is a small, brown pony. It is wearing a saddle and a bridle, indicating it is prepared for riding. The pony appears to be walking on a street, with a red car visible in the background. ”,
“Object3”: “blob”: [165, 247, 102, 75, -3.095],
“caption”: “The car in the close-up image is a black Volkswagen Beetle. It has a distinctive rounded shape and a yellow license plate. The car appears to be in motion on a road. ”,
“Object4”: “blob”: [563, 276, 132, 44, 1.599],
“caption”: “The image is blurry, making it difficult to discern specific details about the person. The person appears to be walking, possibly in a parking lot or similar outdoor setting. The individual is holding onto a leash, suggesting they might be walking a dog. ”,
...
“Frame12”: “Object2”: “blob”: [387, 229, 136, 95, 2.685],
“caption”: “ The horse in the close-up image is a large, brown horse with a white blaze on its face. It appears to be a healthy and well-groomed animal. ”,
“Object3”: “blob”: [30, 188, 87, 70, -2.388],
“caption”: “ The car in the close-up image is a black sedan with a yellow license plate. The vehicle appears to be parked or stationary, as indicated by the lack of motion blur. ”,
“Object4”: “blob”: [670, 242, 151, 64, 1.598],
“caption”: “ The image is a close-up of a person who appears to be a woman. She is holding a leash, which suggests she might be with a pet. The woman is wearing a white top and blue jeans. ”
```

Prompt: {inference prompt}

Table 6. Our prompt for GPT-4o to generate blob layouts in text-to-video generation. We use two fixed exemplars for all prompts as shown in this table. The “{inference prompt}” represents the actual text prompt that users use to generate a video.

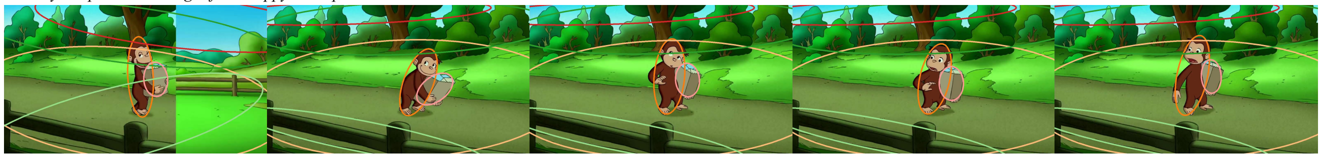
Global caption (from VIDGEN-1M): *"The video shows a cartoon character, wearing a yellow coat and red scarf, dancing in front of a house. The character is barefoot and appears to be having a good time. The house in the background has a green roof and white windows. The character's movements are fluid and rhythmic, and they seem to be enjoying themselves. The video is bright and colorful, with the character's yellow coat standing out against the green background."*



Blob Descriptions:
Frame 0:
"The monkey in the close-up image is wearing a vibrant red hat and a matching red scarf. It's dressed in a yellow coat, which stands out against the monkey's fur. <u>The monkey's expression is cheerful, with a wide smile on its face.</u>"
"The jacket in the close-up image is a vibrant yellow color, featuring a red scarf wrapped around the neck. The jacket appears to be made of a soft fabric, and it has a collar that is turned up. The design of the jacket suggests a casual and comfortable style."
"The scarf in the close-up image is red and appears to be made of a soft material, possibly wool or a wool blend. It is wrapped snugly around the character's neck, providing a pop of color against the character's yellow outfit. The scarf's vibrant red color stands out prominently in the image."
"The a hat in the close-up image is a vibrant red color, featuring a wide brim that extends outward. It appears to be made of a soft fabric, and there is a visible button on the front, suggesting a button-up design. The hat is worn by a character, adding a pop of color to the scene."
"The image shows a close-up of a window with a simple, cartoon-like style. The window has a white frame and is divided into two sections by a vertical line. The glass appears to be clear, allowing light to pass through. The background is a solid color, providing a contrast that highlights the window."
"The house in the close-up image is a charming yellow structure with a red roof. It features a white door and a green shutter, adding a pop of color to the scene. The house appears to be well-maintained and inviting, suggesting a warm and welcoming atmosphere."
"The ground in the close-up image is a light green color, with a few darker green spots scattered around. The surface appears to be a smooth, flat floor. There are no visible textures or patterns on the ground."
Frame 16 (other descriptions omitted):
"The monkey in the close-up image is a cartoon character, wearing a vibrant red scarf and a matching red hat. It's dressed in a yellow coat, which adds a pop of color to its outfit. The monkey appears to be in a joyful mood, <u>as it's smiling and has its arms outstretched.</u>"
Frame 56 (other descriptions omitted):
"The monkey in the close-up image is wearing a red hat and a yellow coat. It appears to be in a cheerful mood, <u>as it is smiling and clapping its hands</u>. The monkey's fur is a mix of brown and black colors."
Frame 88 (other descriptions omitted):
"The monkey in the close-up image is wearing a vibrant red hat and a matching red scarf. It has a cheerful expression on its face, with its mouth open as if it's laughing or singing. <u>The monkey's arms are outstretched, and it appears to be in motion, possibly dancing or celebrating</u>."
Frame 112 (other descriptions omitted):
"The monkey in the close-up image is a cartoon character, wearing a red hat and a yellow jacket. It appears to be in a cheerful mood, <u>with a smile on its face. The monkey is standing on one leg, with its arms outstretched, as if it's dancing or performing some action.</u>"

Figure 10. An example of our data annotation results using instance list and Grounding DINO for segmentation. The text color of the blob descriptions match with the blob colors in the frames. The underlined text highlights the changing part of the descriptions as the monkey's gesture and expression changes over time.

Global caption (from VIDGEN-1M): *"This video is a cartoon animation of a monkey walking on a path in a park. The monkey is carrying a bag of popcorn and appears to be enjoying it. The background consists of green trees and grass, and the path is brown. The monkey is brown with a lighter brown face and belly. The bag of popcorn is white with blue stripes. The monkey's expression changes from happy to surprised as it walks."*



Blob Descriptions:
Frame 0:
"The image is a close-up of a cartoon monkey's face. The monkey is smiling and holding a bag."
"The image is a split-screen animation, showing two different scenes. In the close-up image, there is a lush green grass that appears to be well-maintained and vibrant. The grass is dense and covers the ground completely."
"The image is a split-screen cartoon with a close-up of a monkey on the left side and a wider view of a lush green forest on the right. The monkey is holding a bag of food, possibly bananas, and is smiling. The forest scene includes trees, bushes, and a clear blue sky."
"The object in the close-up image is a wooden fence. It appears to be a simple, traditional design, with vertical slats and a horizontal top rail."
"The sky in the close-up image is a bright blue color, suggesting a clear and sunny day."
"The object in the close-up image is a brown suitcase. It appears to be made of leather and has a handle on top."

Figure 11. An example of our data annotation results using ODISE as the panoptic segmentation model. ODISE tends to segment the background into different parts, including the sky, trees, grass, and fence in this example.

The video shows a tiger lying on the ground, looking directly at the camera. It appears to be in a zoo enclosure, and there are trees and a building in the background. The tiger is seen licking its paw, and the camera zooms in on its face.

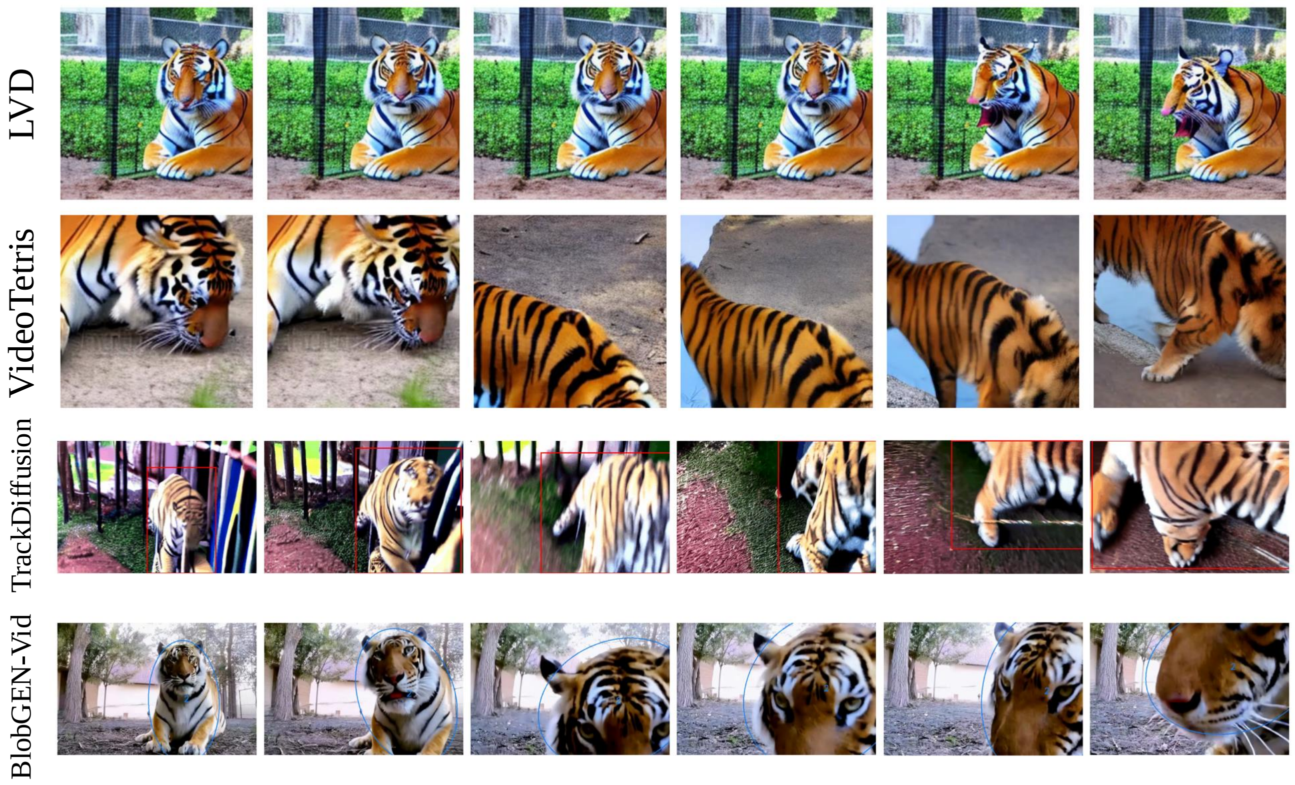


Figure 12. Qualitative examples from YoutubeVIS-700

The video shows a group of colorful birds, including parrots and parakeets, perched on a wooden stand and eating from a tray of seeds. One bird is yellow, another is green, and the third is blue. They are in a cage, and the camera zooms in on the yellow bird as it eats.

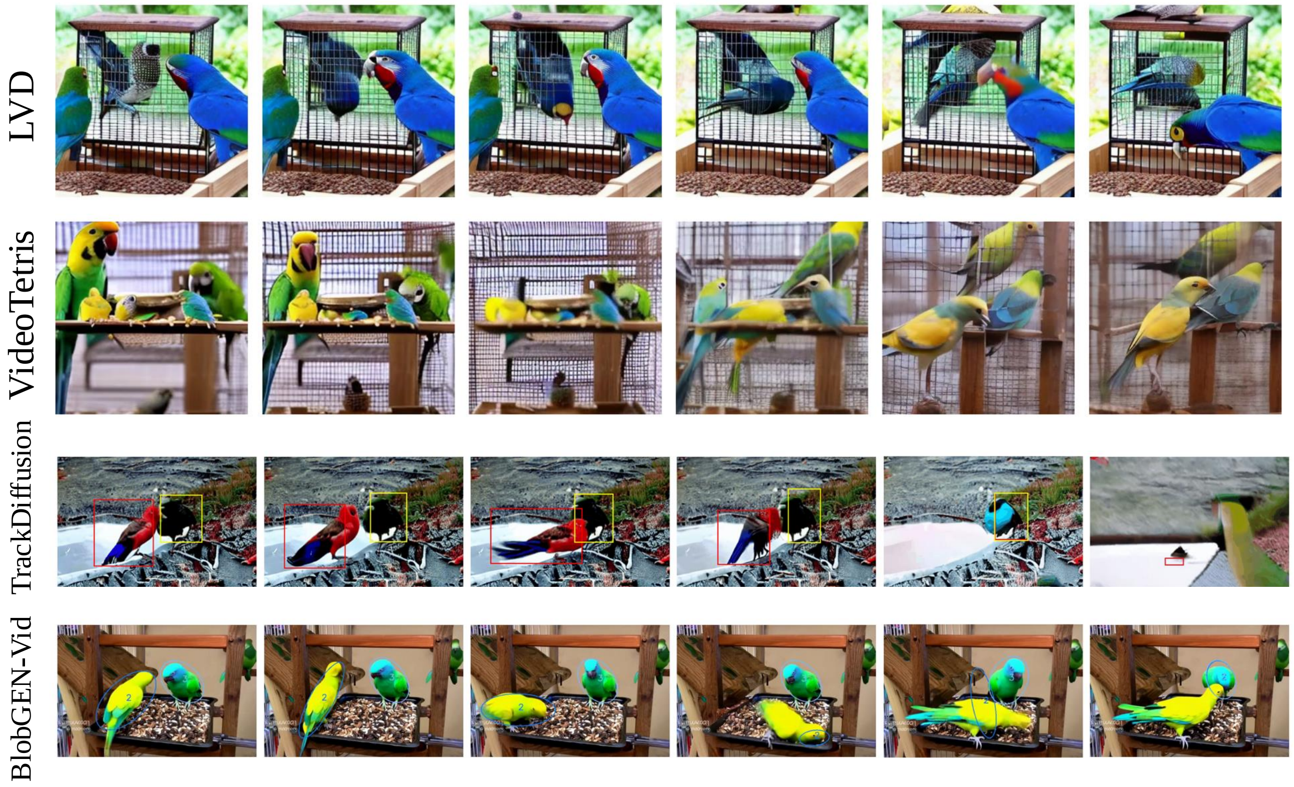


Figure 13. Qualitative examples from YoutubeVIS-700

Spatial Relationships: A cat sitting on the left of a fireplace.

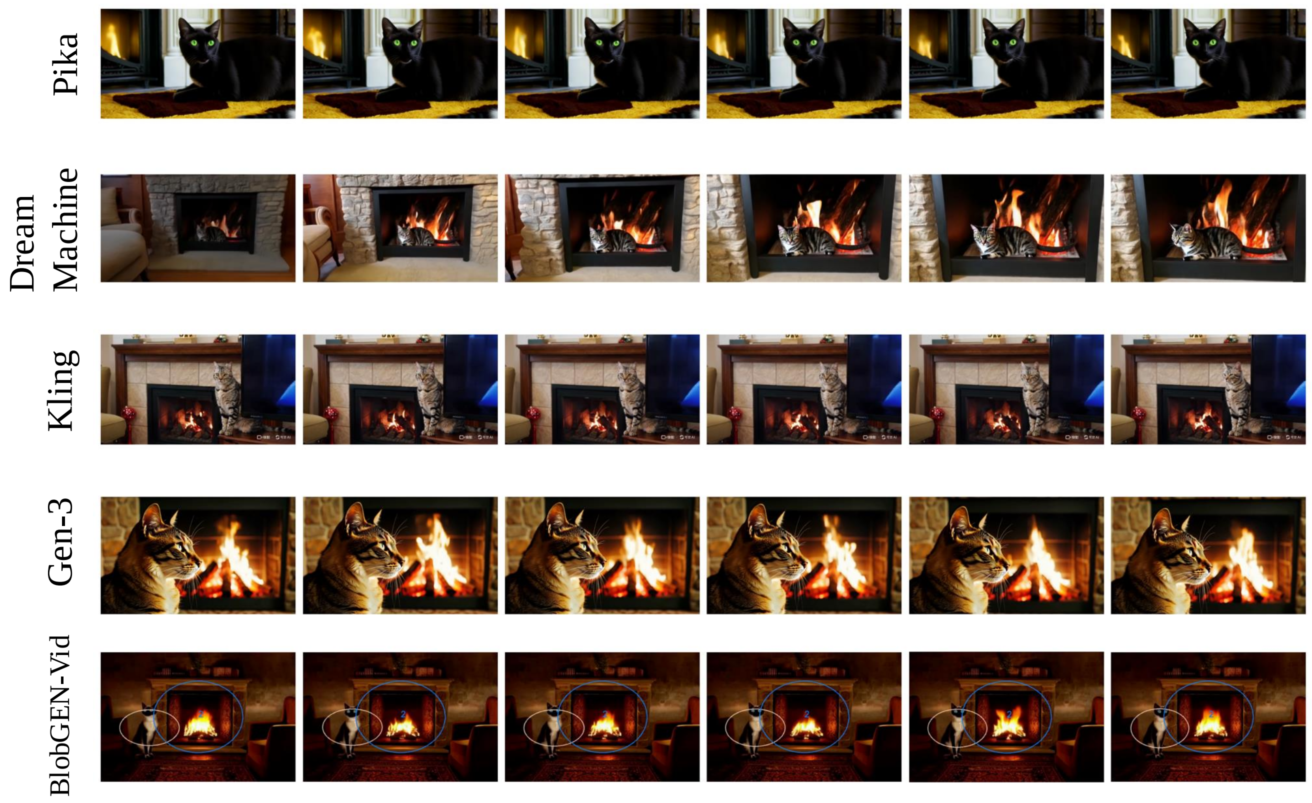


Figure 14. Qualitative examples from T2V-CompBench

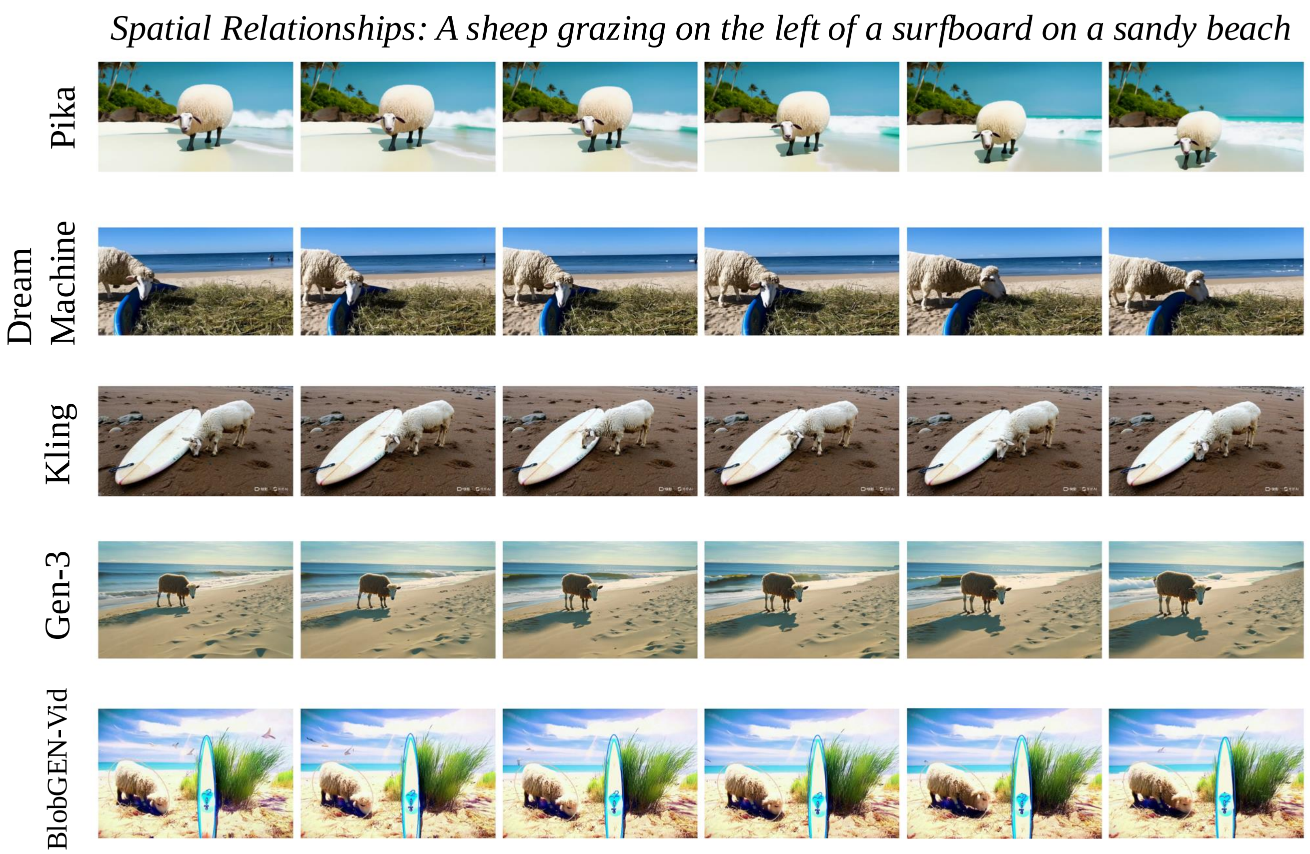

Spatial Relationships: A sheep grazing on the left of a surfboard on a sandy beach
Pika
Dream Machine
Kling
Gen-3
BlobGEN-Vid

Figure 15. Qualitative examples from T2V-CompBench

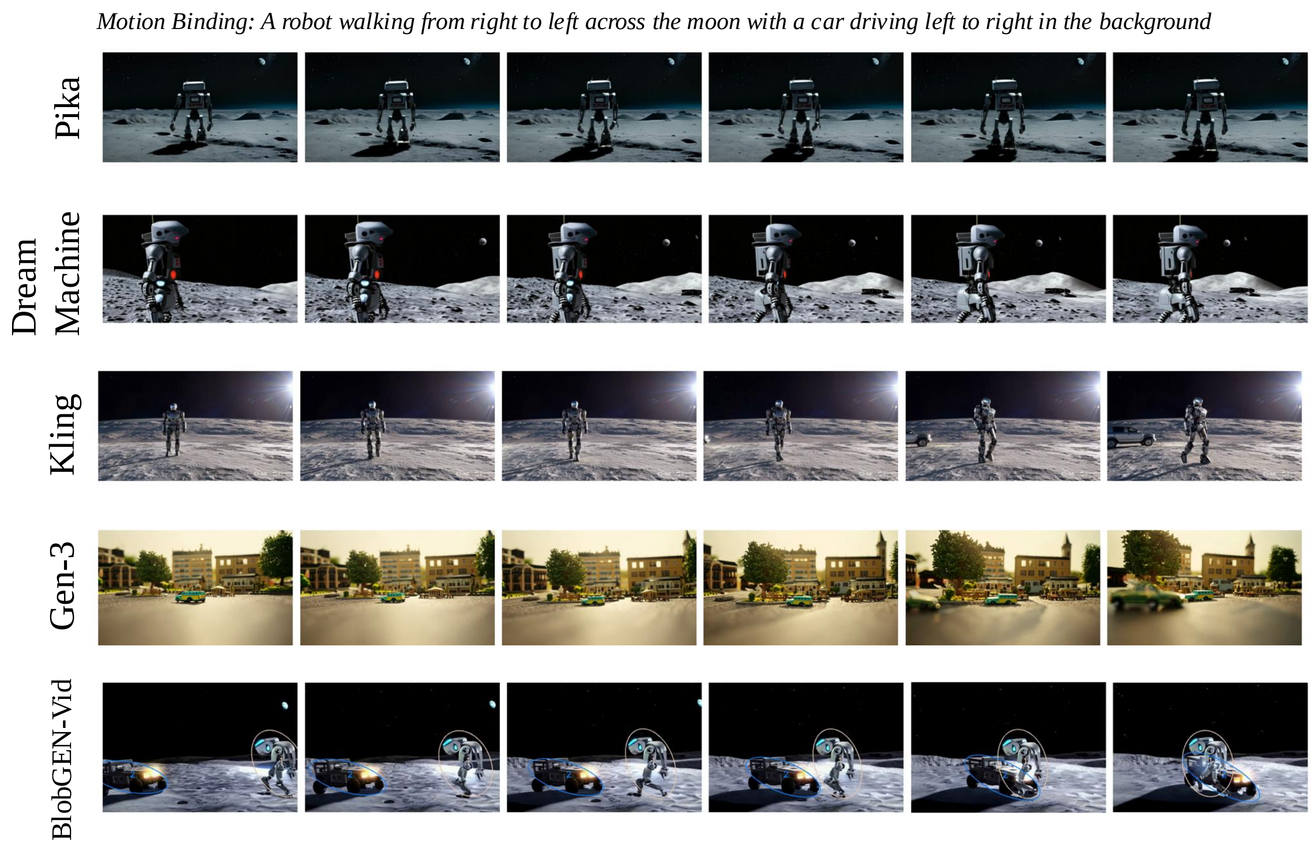

Motion Binding: A robot walking from right to left across the moon with a car driving left to right in the background
Pika
Dream Machine
Kling
Gen-3
BlobGEN-Vid

Figure 16. Qualitative examples from T2V-CompBench

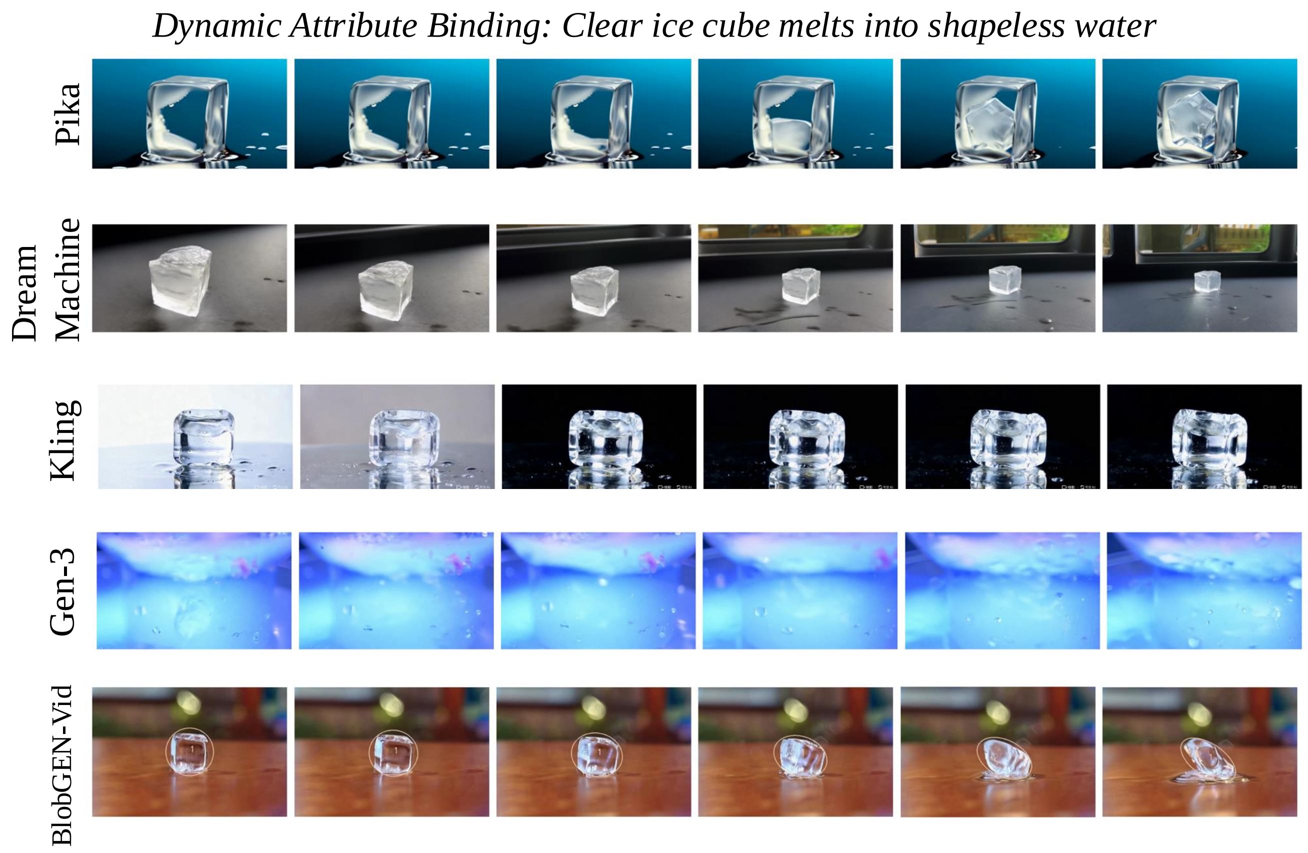

Dynamic Attribute Binding: Clear ice cube melts into shapeless water
Pika
Dream Machine
Kling
Gen-3
BlobGEN-Vid

Figure 17. Qualitative examples from T2V-CompBench

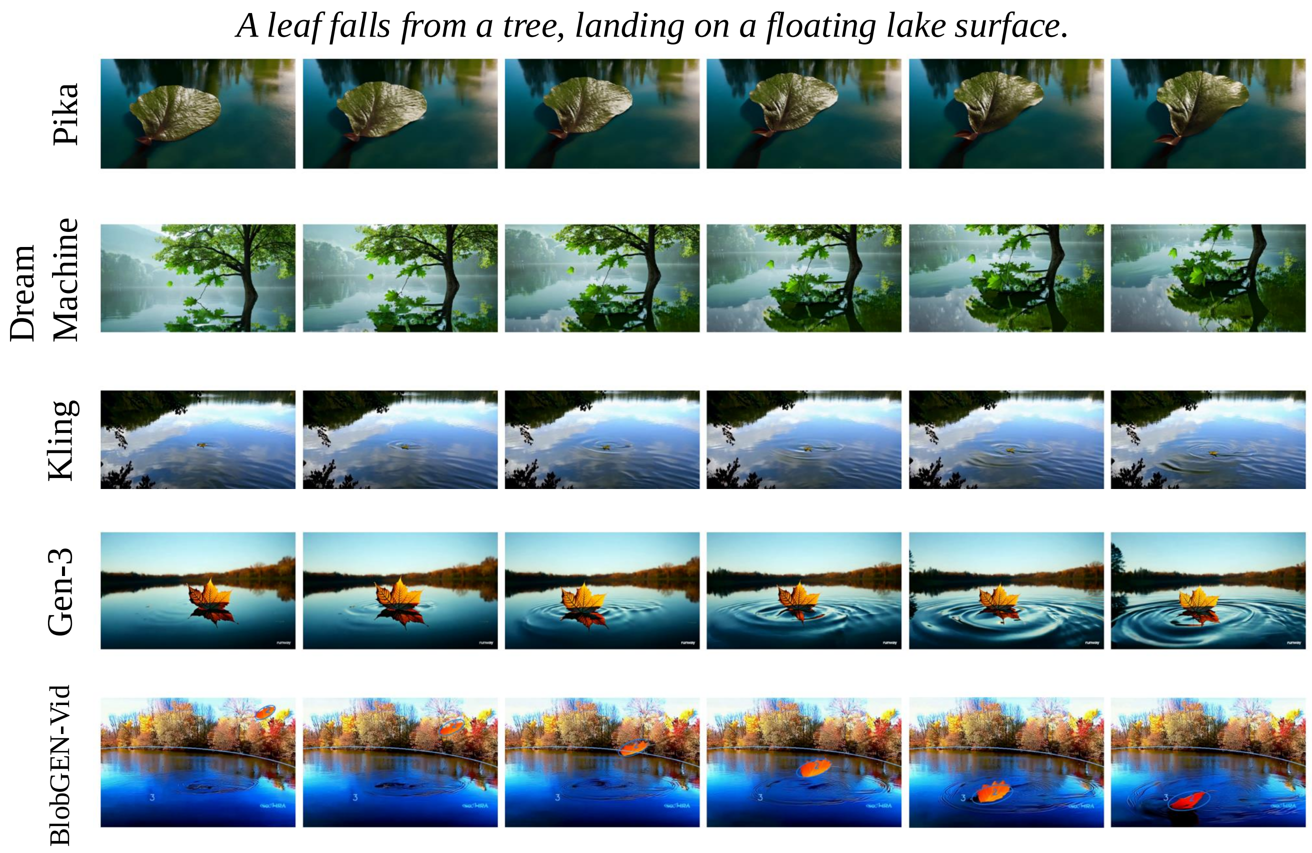

A leaf falls from a tree, landing on a floating lake surface.
Pika
Dream Machine
Kling
Gen-3
BlobGEN-Vid

Figure 18. Qualitative examples from TC-Bench

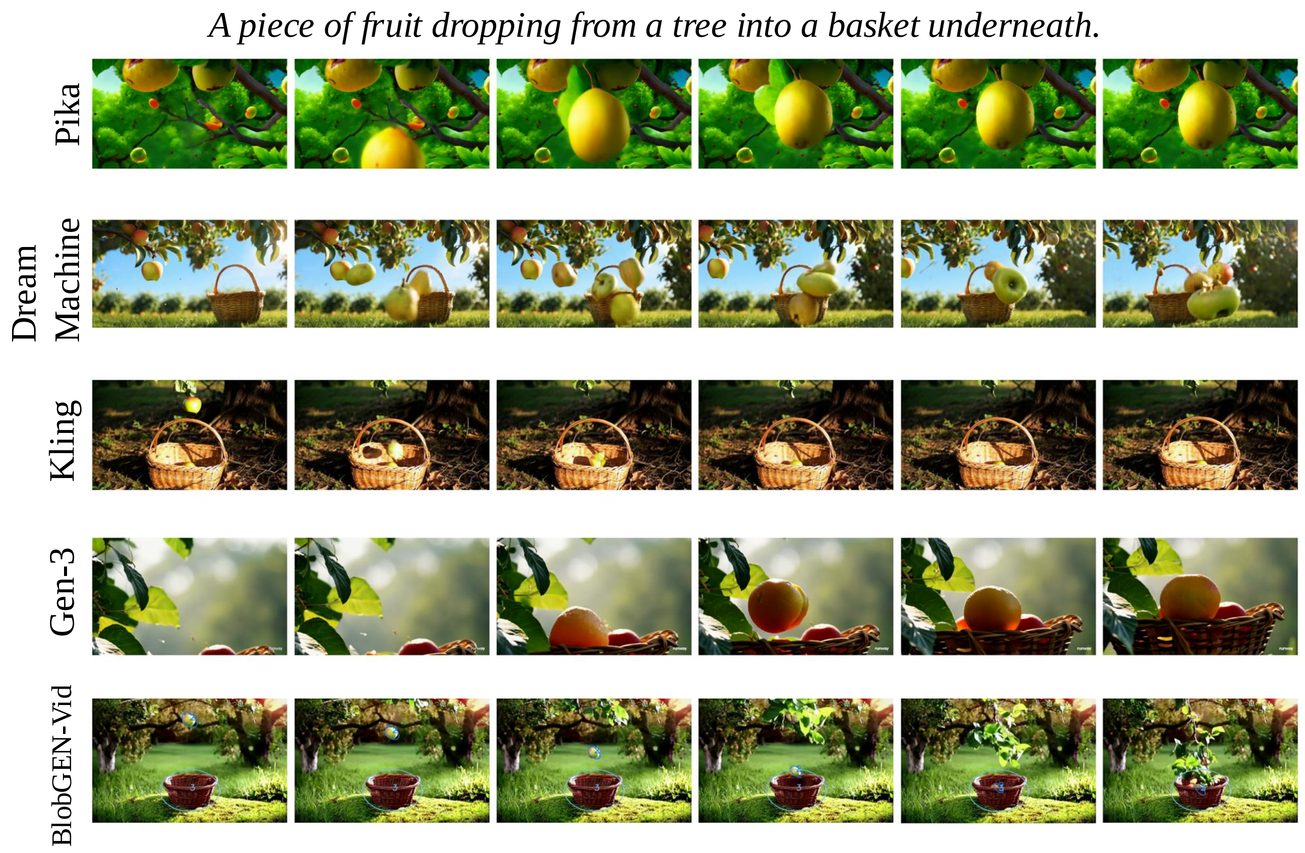

A piece of fruit dropping from a tree into a basket underneath.
Pika
Dream Machine
Kling
Gen-3
BlobGEN-Vid

Figure 19. Qualitative examples from TC-Bench

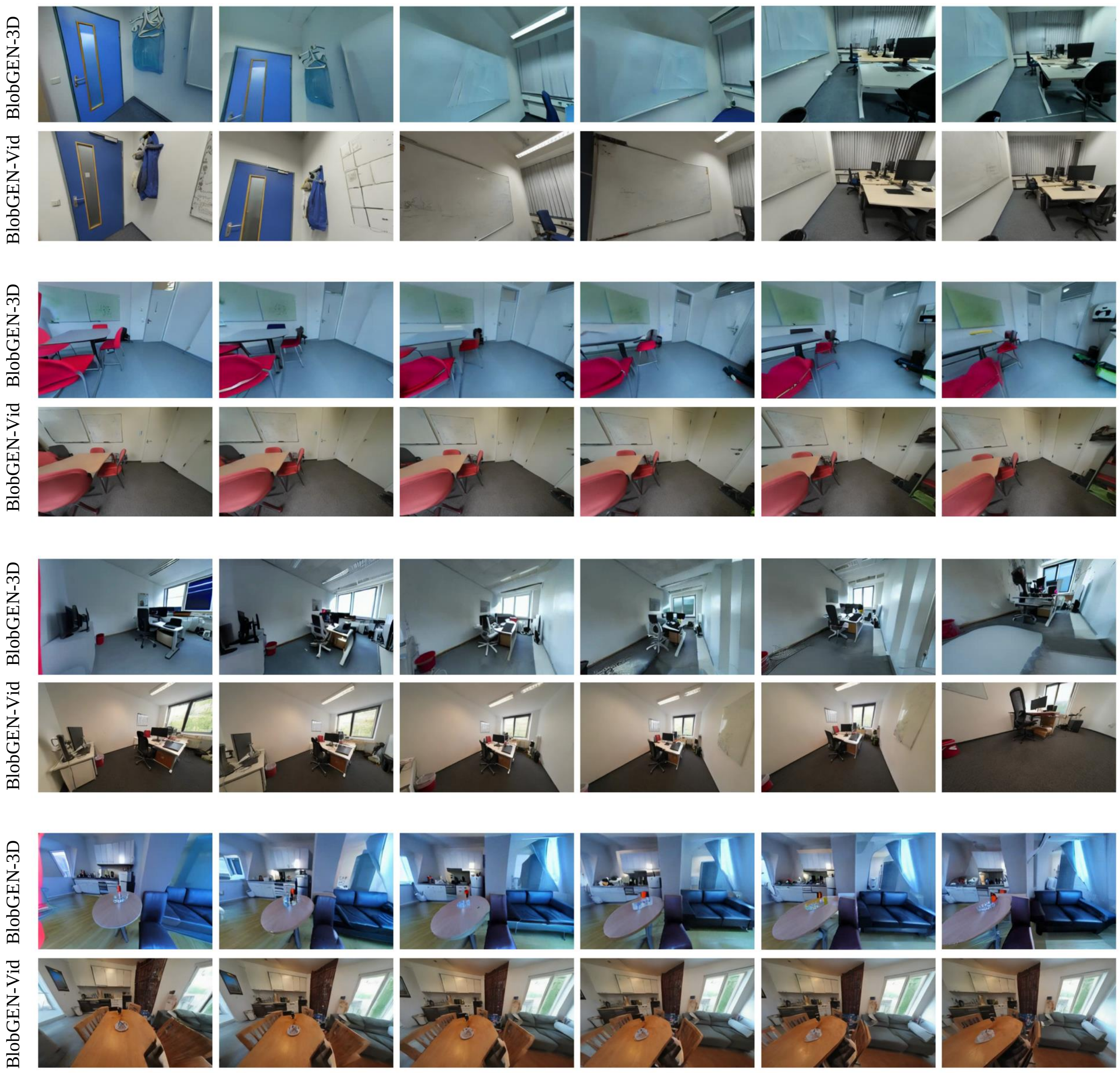


Figure 20. Qualitative examples from ScanNet++