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

Supplementary Material

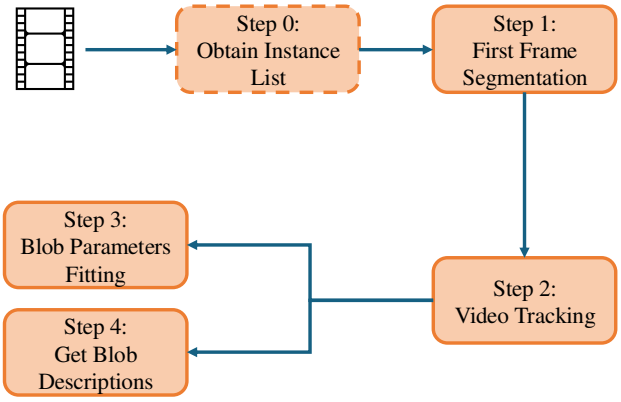


Figure 1. Our data annotation pipeline for obtaining blob video representations consists of five steps with step 0 being optional.

1. Data annotation

Data annotation pipeline. Our data annotation pipeline consists of four to five key steps as shown in Fig. 1. The step 0 is to obtain a list of objects appeared in the video using a VLM. Though previous methods [15] 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 [19] and prompt it to generate a list of objects that appear in each video. Using the instant list, we could apply Grounding DINO [9] in step 1 to obtain segmentation masks for the first frame of the video. We also experiment with ODISE [16], 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 [11] 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 [8] 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. 5 (using Grounding DINO) and Fig. 6 (using ODISE). We observe that the instance list obtained from LLaVA-NeXT-Video-7B [19] usually contain instance names in different hierarchical levels. For example, in Fig. 6, 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 [16] 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. 6, 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.

2. BlobGEN-Vid framework

Context interpolation module. As shown in Fig. 2, 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 [3]. 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. 2, we have omitted B, N and use $L = 3$ and $T = 9$ for demonstration purpose. In our implementation,

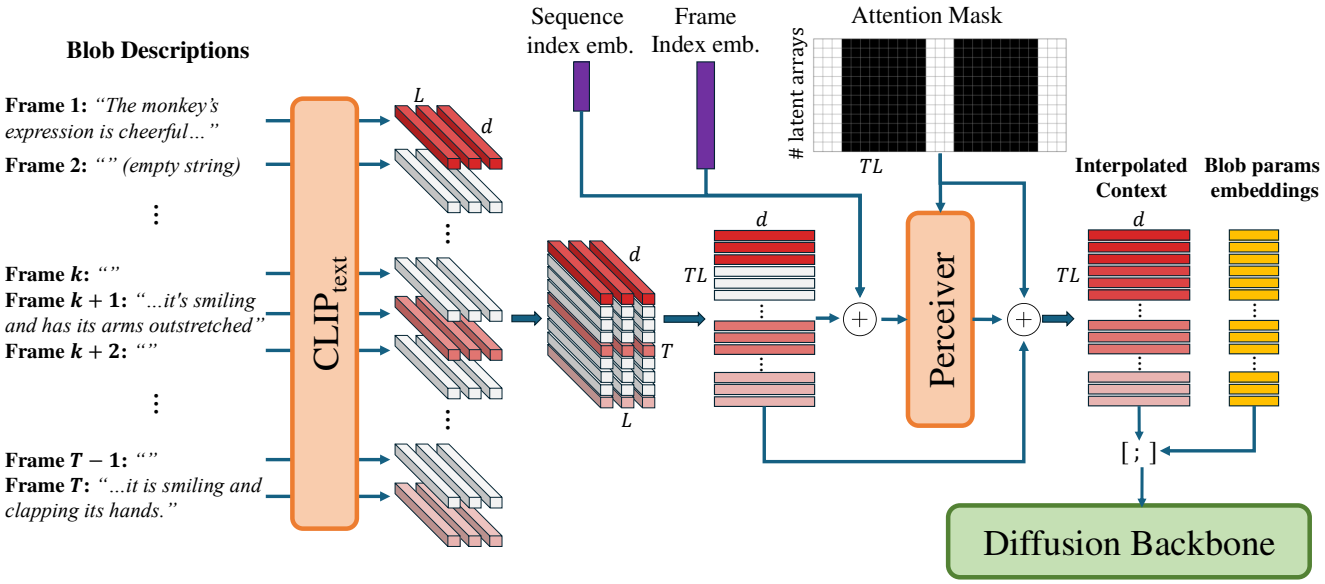


Figure 2. An illustration of the context interpolation stage using a Perceiver-based model. Note that we omit the batch size (B) and number of blobs per frame dimension (N) for simplicity. After the CLIP text encoder encodes each blob description, we merge time T and context sequence length L into one 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 [2, 7], such property in DiT-based video diffusion model [18] 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. 3, we show the attention maps between *a visual token* and the visual tokens of Frame 1. In Fig. 4, 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 [18] for details of the input and output of the full 3D attentions.

3. Implementation details

Training data. For open-domain video generation, our training dataset is obtained by annotating 160K OpenVid [10] videos, 460K VIDGEN [13] videos and 320K videos from HDVILA [17]. 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.

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 [9]+SAM2 [11] using the ground truth object labels to obtain object bounding boxes (bboxes) per frame. Then

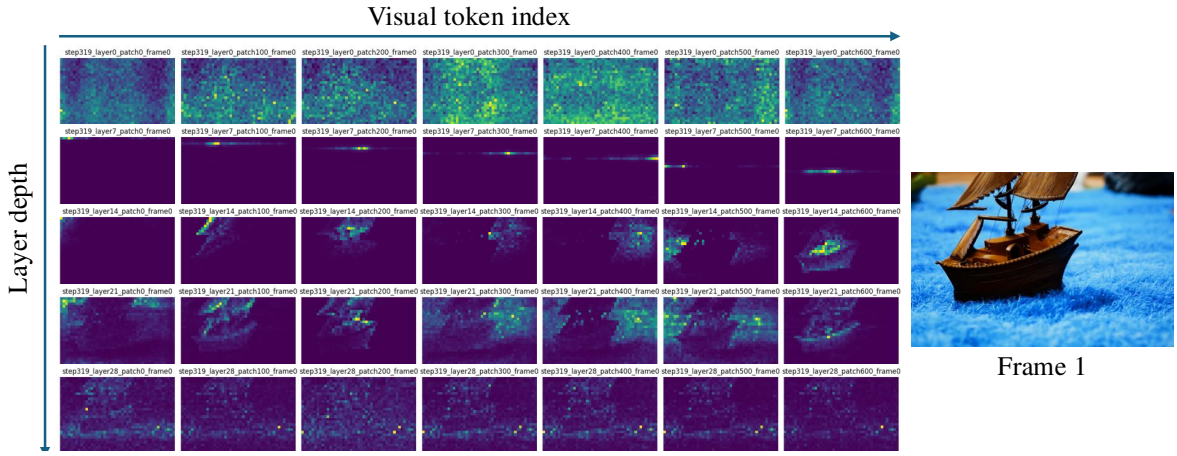


Figure 3. Attention maps between *a visual token* and other visual tokens of the first frame. Some of the maps show similar spatial structure as Frame 1 in the pixel space. The visualization proves that full 3D attention still preserves the spatial structure as in UNet-based diffusion models.

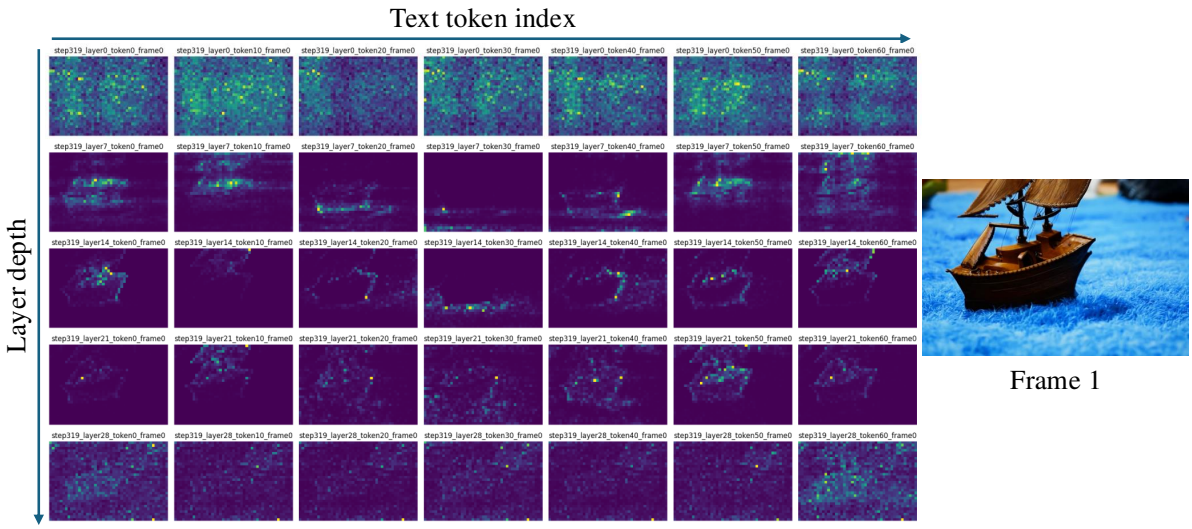


Figure 4. Attention maps between *a text token* and other visual tokens of the first frame. The highlighted regions represent the regions where the text token is highly correlated with. The visualization proves that full 3D attention still preserves the spatial structure as in UNet-based diffusion models.

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 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 [5], all 16 generated frames are involved as all frames are grounded on input layouts. For LVD [6], Frame 1, 4, 7, 10, 13, 16 of all 16 frames are involved. For VideoTetris [14], 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 [12] and TC-Bench [1], 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 [8] to evaluate the attribute correctness. Spatial and Numeracy accuracy are computed using GroundingSAM [9] to locate and count the objects. Motion Binding is computed using GroundingSAM and Dense Optical Tracking [4]. 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 [1, 12].

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 2. 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.

	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 1. 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.

4. Additional Results

Ablation study. In Table 1, 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.

4.1. Additional qualitative results

We show additional qualitative results from various settings and benchmarks in Fig. 7-15.

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 2. 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.

References

[1] 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. 4

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. The monkey's expression is cheerful, with a wide smile on its face."

"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, as it's smiling and has its arms outstretched."

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, as it is smiling and clapping its hands. The monkey's fur is a mix of brown and black colors."

Frame 88 (other descriptions omitted):

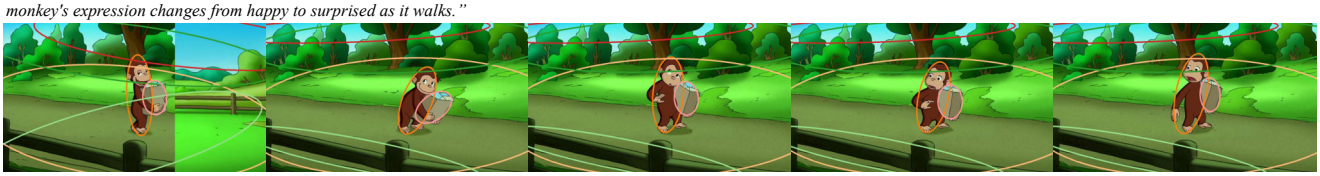
"The monkey in the close-up image is wearing a vibrant red hat and a matching red scarf. It has a cheerful expression on its face, with its mouth open as if it's laughing or singing. The monkey's arms are outstretched, and it appears to be in motion, possibly dancing or celebrating."

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, 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."

Figure 5. 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 6. 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.

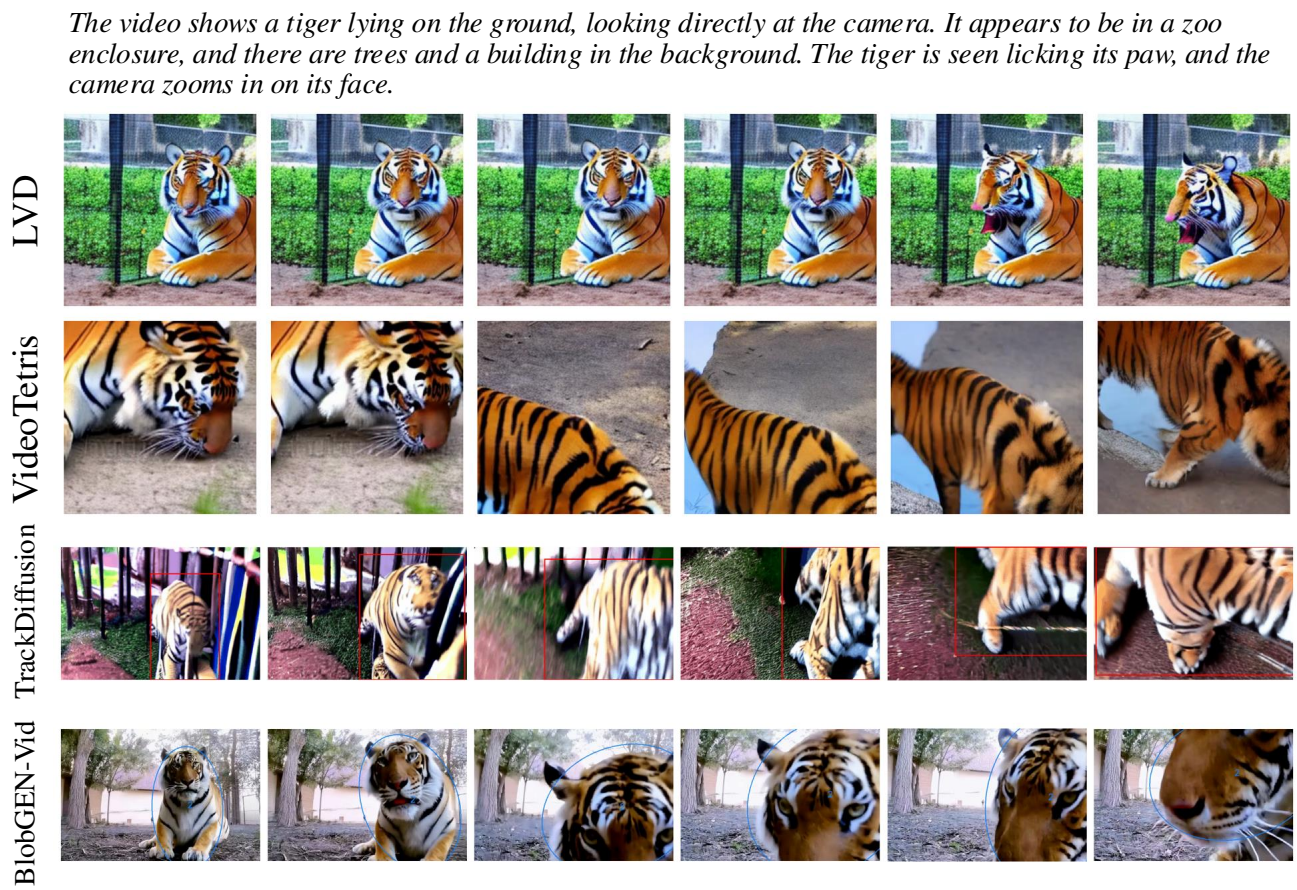


Figure 7. Qualitative examples from YoutubeVIS-700

[2] 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. 2

[3] 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. 1

[4] Guillaume Le Moing, Jean Ponce, and Cordelia Schmid. Dense optical tracking: Connecting the dots. In *CVPR*, 2024. 4

[5] 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. 4

[6] 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. 4

[7] 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. 2

[8] 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. 1, 4

[9] 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. 1, 2, 4

[10] 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. 2

[11] 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. 1, 2

[12] Kaiyue Sun, Kaiyi Huang, Xian Liu, Yue Wu, Zihan Xu, Zhenguo Li, and Xihui Liu. T2v-compbench: A comprehen-

Figure 8. Qualitative examples from YoutubeVIS-700

sive benchmark for compositional text-to-video generation. *arXiv preprint arXiv:2407.14505*, 2024. 4

[13] 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. 2

[14] 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. 4

[15] 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. 1

[16] 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. 1

[17] 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

[18] 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

[19] 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. 1

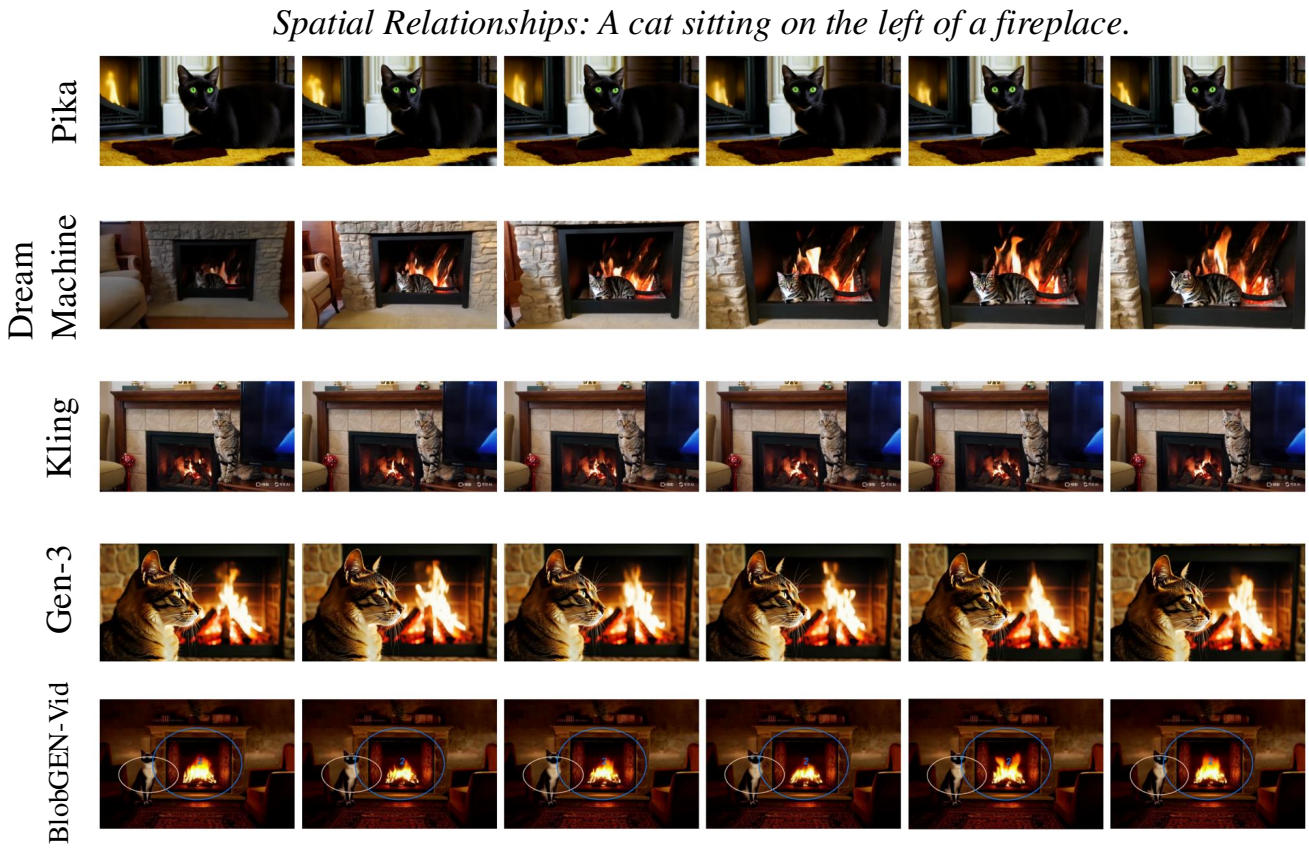


Figure 9. Qualitative examples from T2V-CompBench

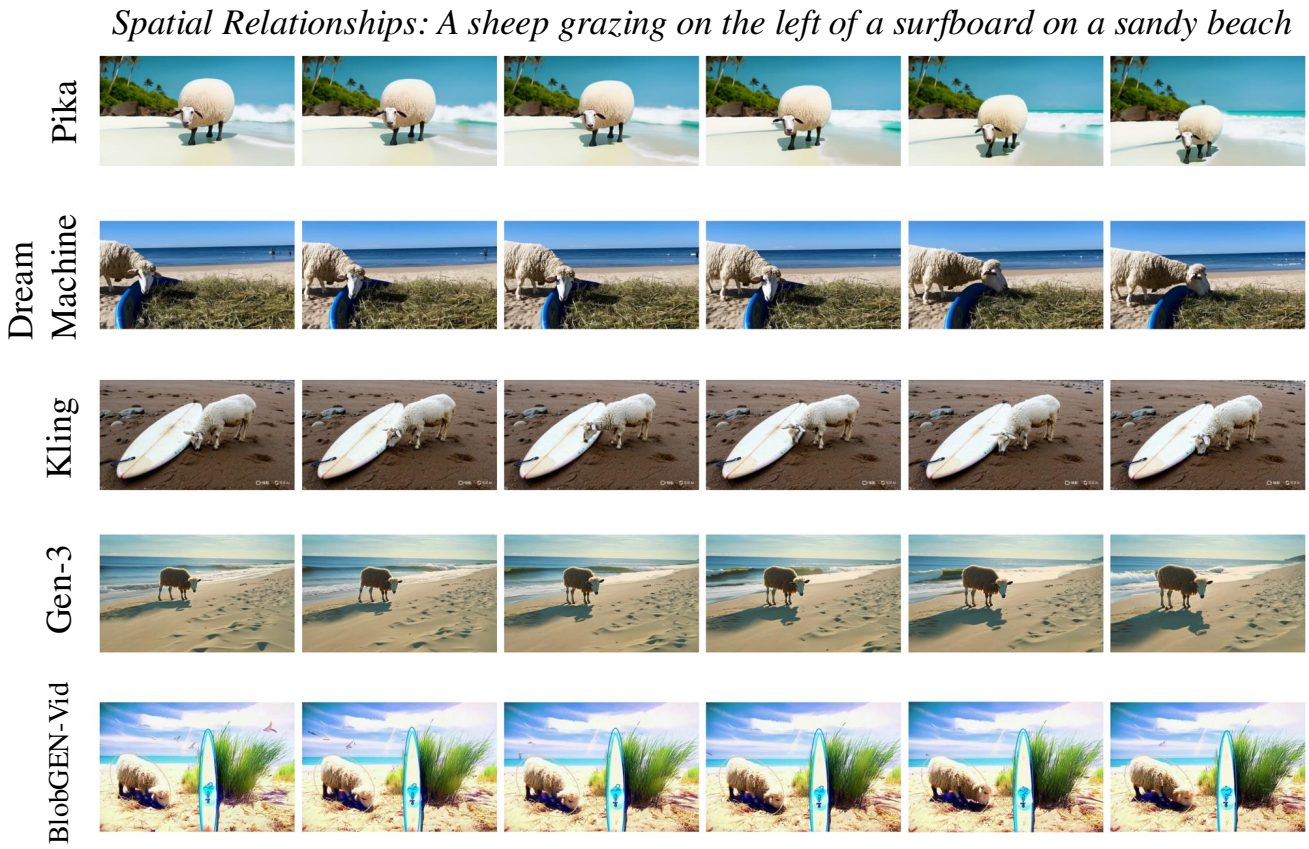


Figure 10. Qualitative examples from T2V-CompBench

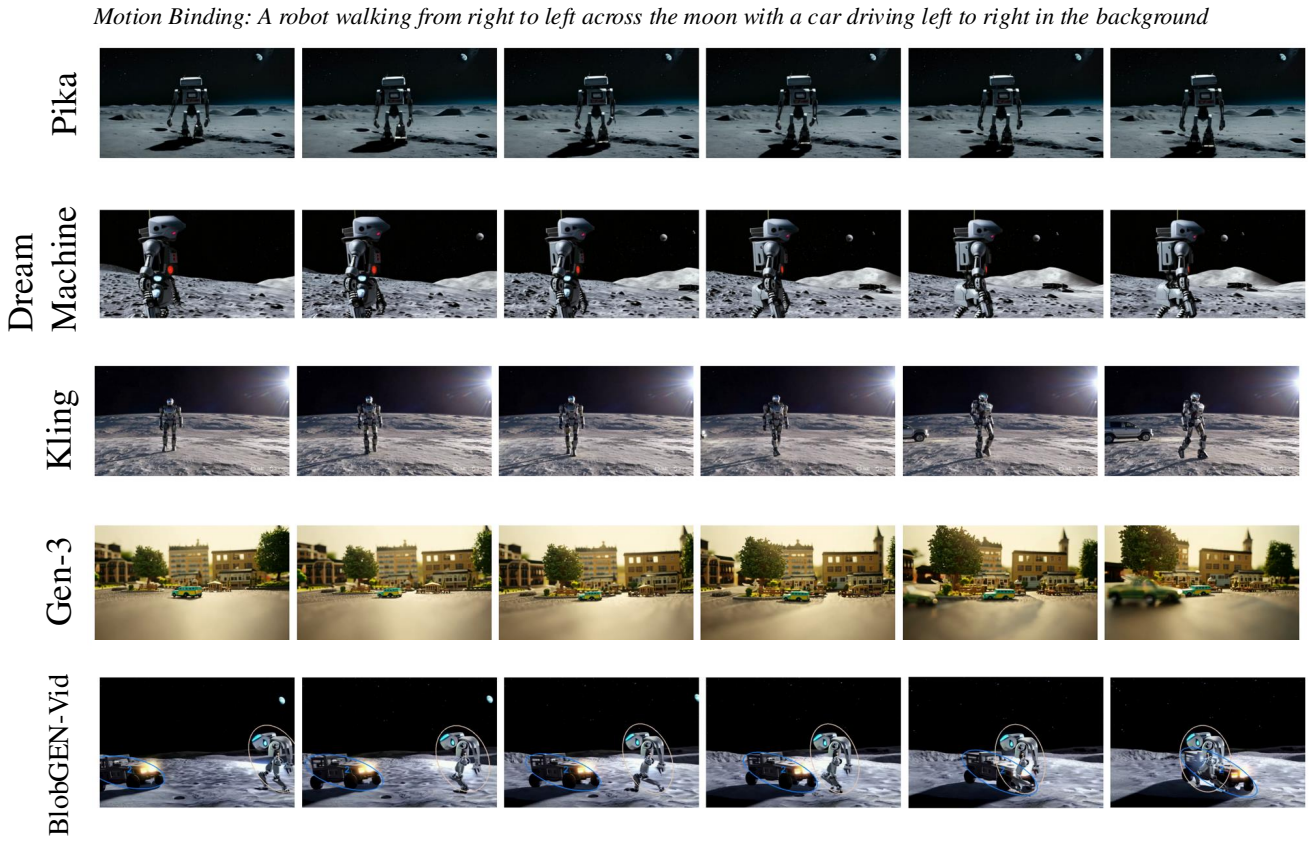


Figure 11. Qualitative examples from T2V-CompBench

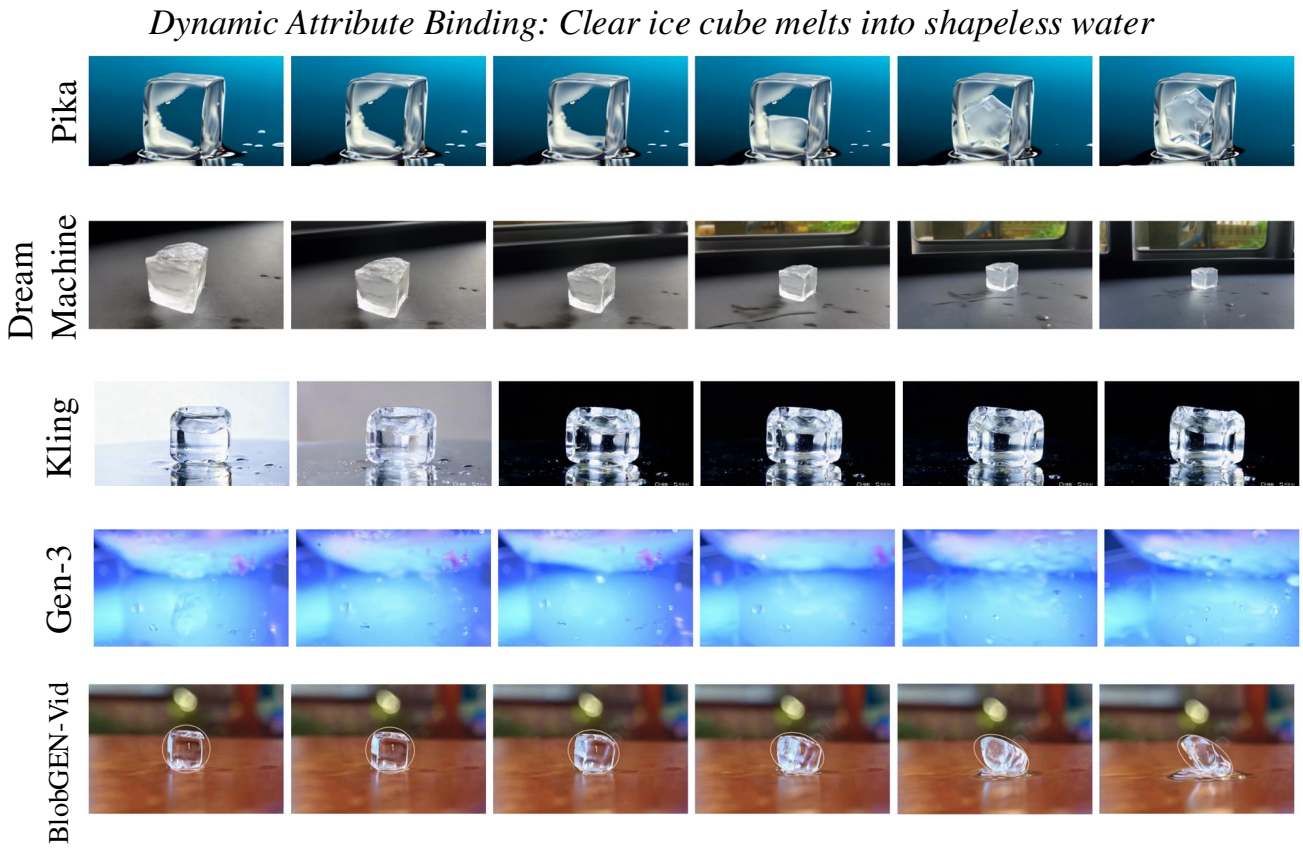


Figure 12. Qualitative examples from T2V-CompBench

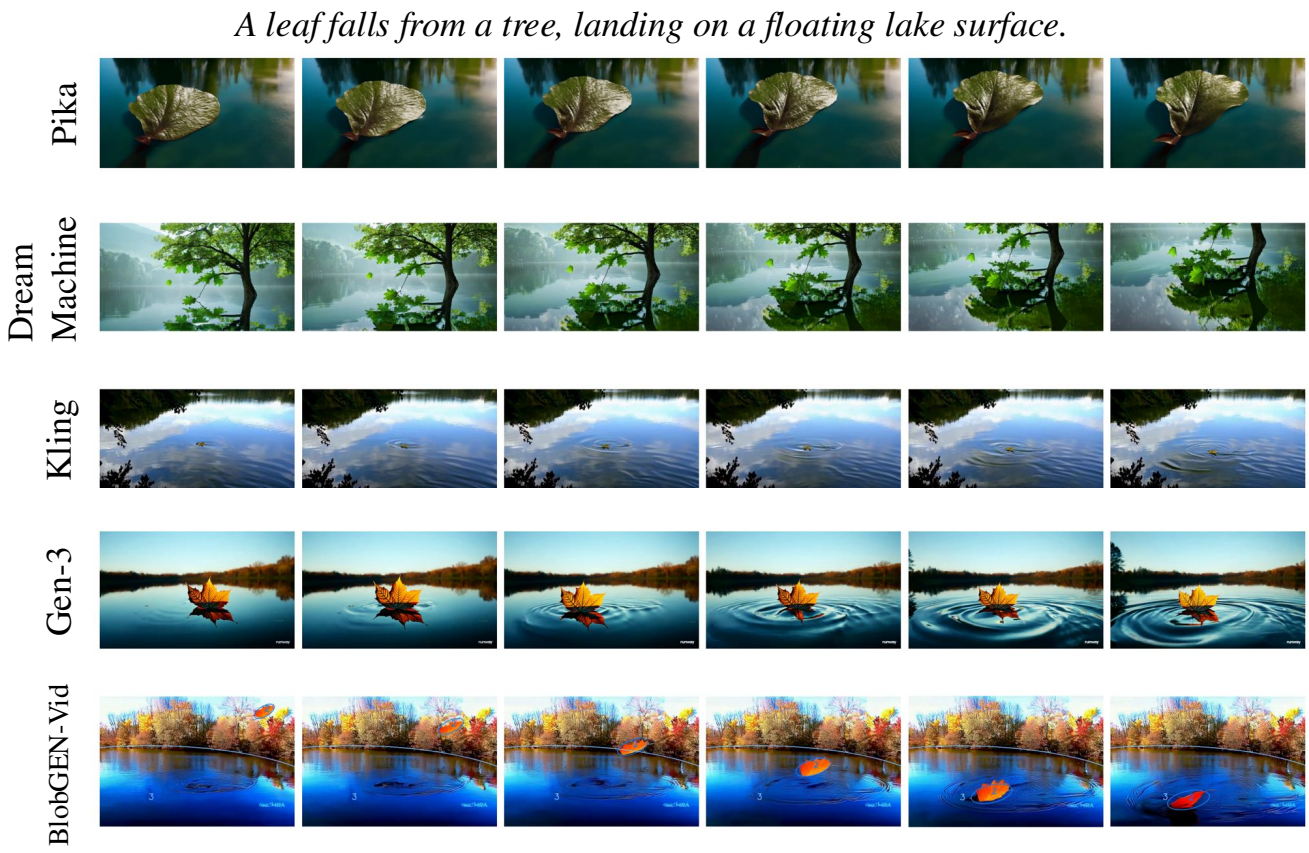


Figure 13. Qualitative examples from TC-Bench

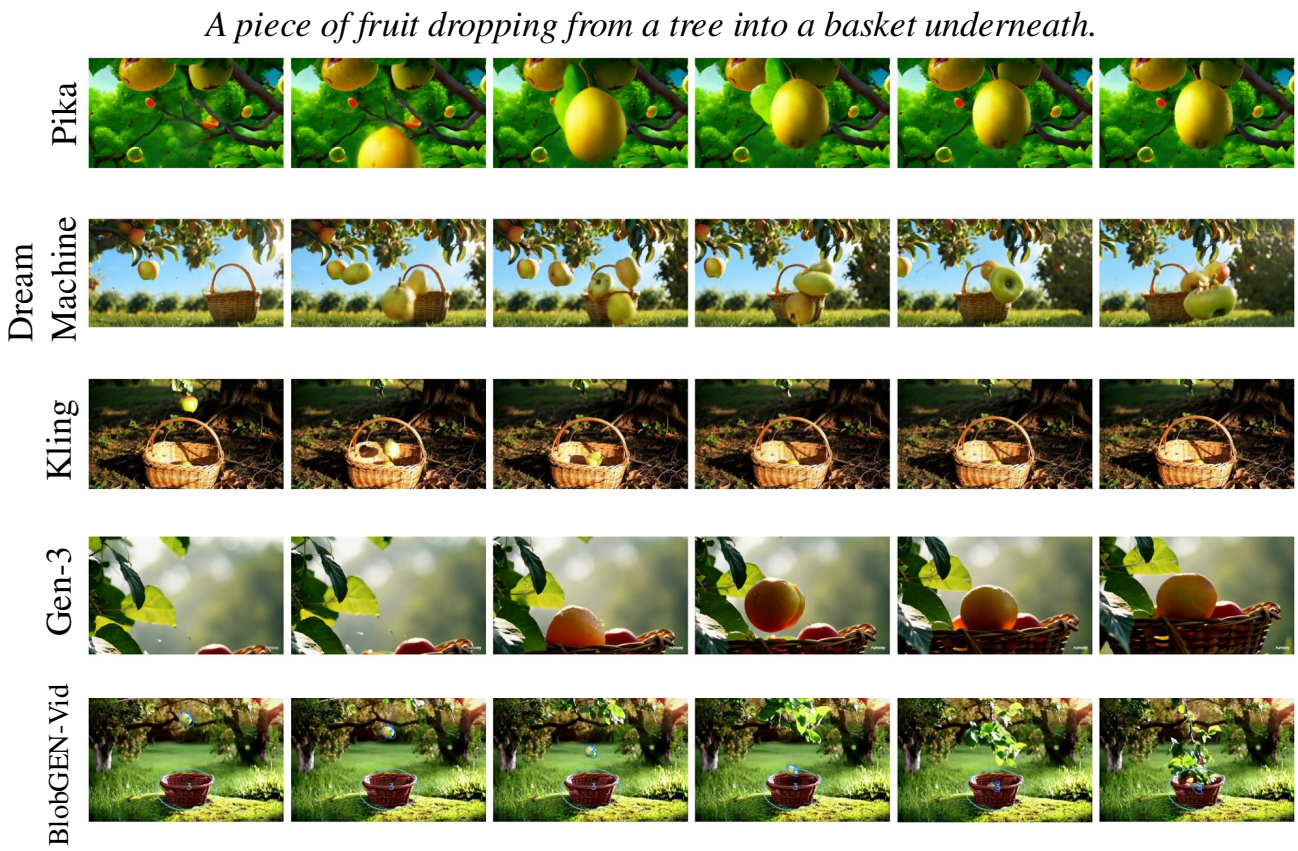


Figure 14. Qualitative examples from TC-Bench

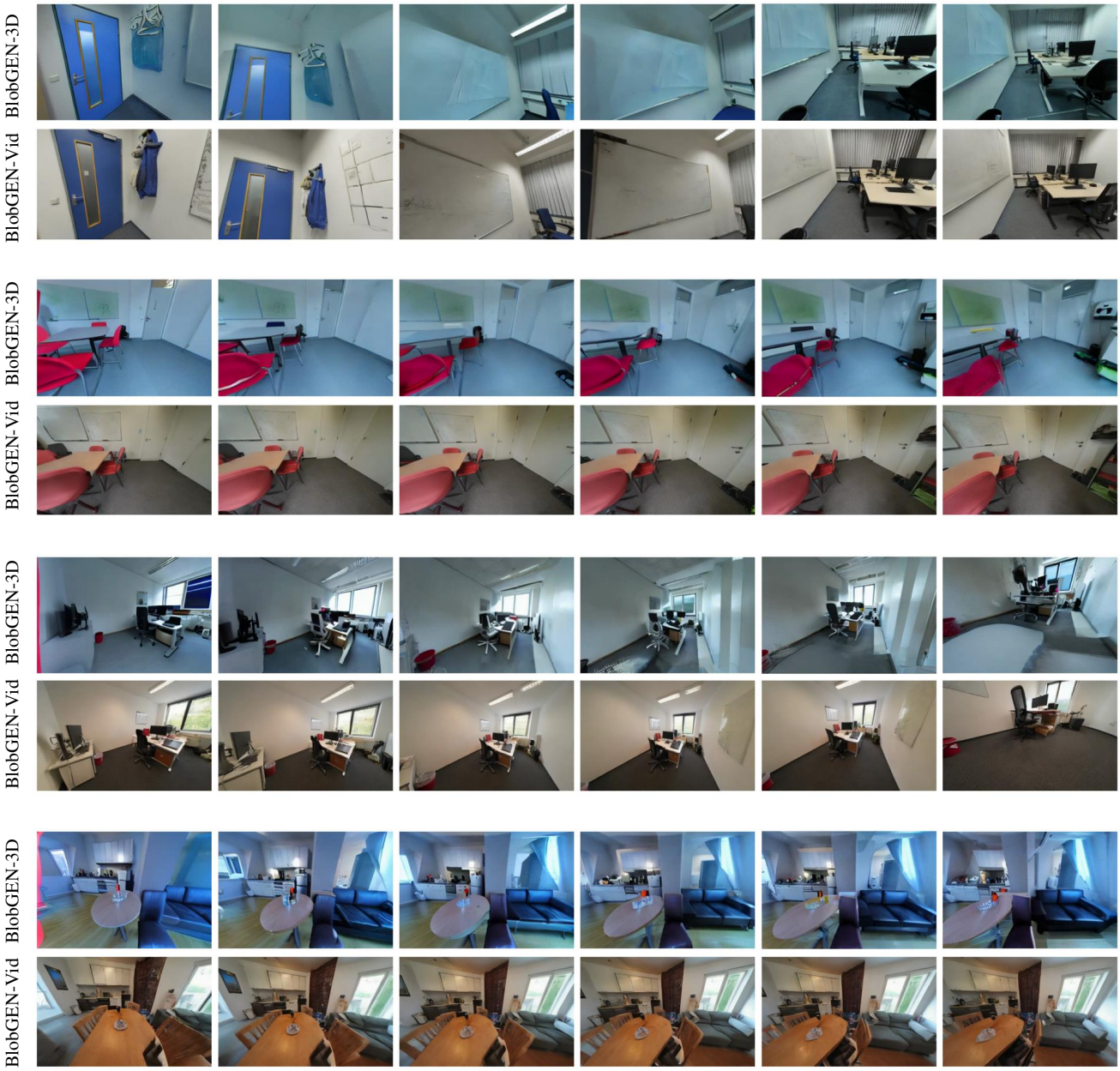


Figure 15. Qualitative examples from ScanNet++