T2V-CompBench: A Comprehensive Benchmark for Compositional Text-to-video Generation

Appendices

Appendix Contents

A. Prompt Suite

A.1. Vocabulary Construction

We utilize WordNet [46] to group words from real-user prompts provided by VidProM [68] into multi-level meta classes. From these groups, we select high-frequency object nouns, active verbs, and adjectives.

Nouns. Figure 5 illustrates the structure of these multi-level classes. The nouns organized into multi-level meta classes come along with their frequencies of occurrence in real-user prompts. To select high-frequency words, we first analyze the frequency distribution of the entire real-user prompt dataset. Figure 6(a) displays the frequencies of the 10000 most frequent nouns, arranged in descending order. It is apparent that users do not use words with uniform frequency; rather, a significant number of words center around the first 2000 most frequent words, while the remaining words receive much less attention. In this dataset, all of the top 2000 words have frequencies greater than 900, so we primarily focus on words that exceed this threshold. Following our word selection criteria outlined in Section 3, we start with words that meet our criteria, such as “dog” and “car”. From there, we select additional words within the same class as “dog” or “car” to expand our noun list, for example, “cat”, “lion” and “truck”, “boat”. As a result, we identify a total of 260 object nouns, within which 66% have frequencies greater than 900. Although the remaining words fall below this frequency, they are still commonly used in natural language and contribute to the diversity of our noun selection.

Verbs. We apply the same method to group the verbs. Figure 6(b) illustrates the frequency distribution of verbs. The top 1200 words have frequencies over 370. Then, we select 200 active and vibrant verbs, with 60% having frequencies greater than 370.

Attributes. For words that describe attributes, we select from both adjectives and nouns. When analyzing adjectives, we identify relevant words by looking for specific keywords in their definitions. If a definition includes terms like “color” or “colored”, we classify the word as a color attribute. Similarly, if the definition contains “made of” or “texture”, we categorize it under texture attributes. Word definitions including “shape” or “shaped” are classified as shape attributes. However, some words that should belong to these categories may be excluded if their definitions do not contain the relevant keywords. To address this, we also consider attributes derived from nouns. Words that represent color or shape can function as either nouns or adjectives in a sentence, so we refer to the meta classes of “color” and “shape” within the noun classification. Additionally, there is a meta class for “material”, which encompasses words that describe the texture of objects. We draw from these classes to supplement the attributes identified from adjectives. For human-related attributes, we focus on nouns in meta classes such as “body_covering”, which includes words like “hair” and “beard”, as well as “clothing”, which features words like “dress” and “hat”. In total, we select 14 color attributes, 26 shape attributes, 26 texture attributes and 14 human-related attributes.

The list of the 260 nouns, 200 verbs and 80 attributes are provided in T2V-CompBench word list.

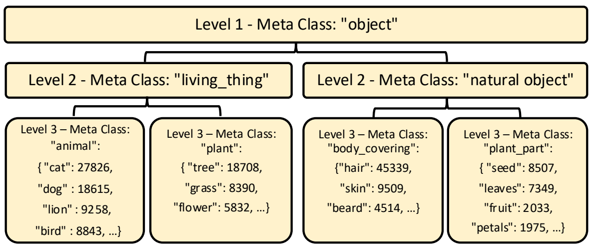
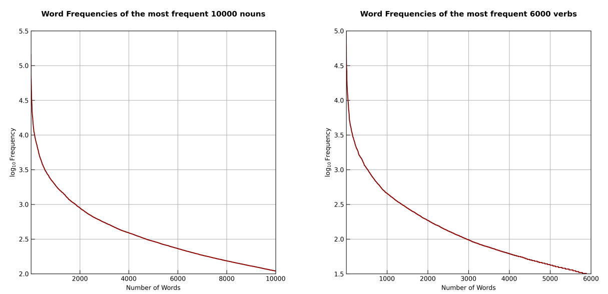


Figure 5. Illustration of multi-level meta class structure.



(a) noun frequency distribution (b) verb frequency distribution

Figure 6. **Analysis of word frequencies.** We show the frequency of occurrence for nouns and verbs in real-user prompts.

A.2. Prompt Generation with LLM

When generating prompts with LLM, we provide the high-frequency nouns, verbs, attributes or other word lists to LLM. The templates used to generate prompts are displayed in section H Table 5, 6, 7, 8, 9, 10 and 11.

A.3. Prompt Suite Statistics

T2V-CompBench stands out for its focus on multiple objects and temporal dynamics. Figure 7 shows the statistics on benchmark prompts: (1) In contrast to previous benchmarks, which predominantly focus on single-object, our prompts involve more than two nouns on average, with each prompt containing approximately 3.6 nouns. (2) T2V-CompBench considers temporal dynamics, with all prompts containing verbs, averaging at 1.4 per prompt. (3) To prevent T2V models from being distracted by irrelevant contents, we avoid using excessively long prompts, the average length is 10.4 words, ranging from 3 to 23. (4) Since we utilize an LLM to generate free-form prompts, many auxiliary words, such as those representing time and spot, are produced in addition to the provided words we instruct the LLM to select from. As a result, there is a rich variety of nouns, verbs, and adjectives present in the generated dataset, ensuring it encompasses a diverse range of topics.

After obtaining the 1400 prompts of T2V-CompBench, we use WordNet [46] to identify the metaclasses of nouns

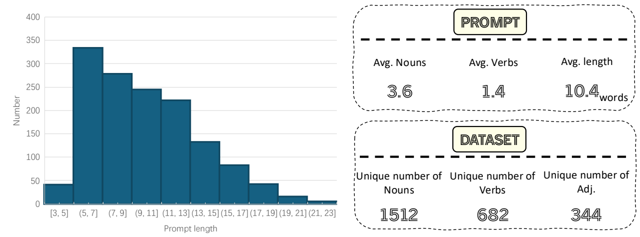
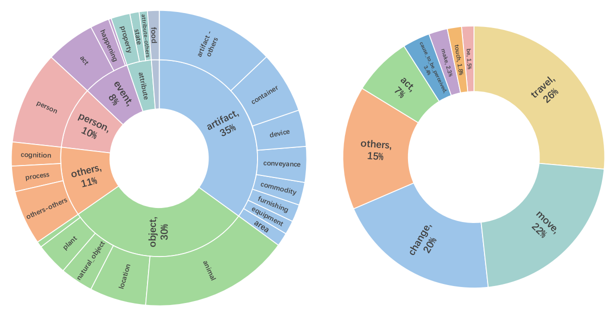


Figure 7. **Statistics on benchmark prompts.** ***Left***: Prompt length distribution. ***Right***: Prompt Suite statistics.

and verbs, and their distributions are visualized in Figure 8. In our prompt dataset, the high-frequency nouns are primarily concentrated in the categories of artifact, object, and person, most of which belong to the “thing” categories. Meanwhile, high-frequency verbs are mainly concentrated in actions like “travel”, “move”, and “change”. The occurrence of less dynamic verbs, such as those associated with “express” and “think”, is nearly negligible in our dataset.



(a) Meta types of nouns (b) Meta types of verbs

Figure 8. **Word distributions of T2V-CompBench prompts** We show the types of nouns and verbs of T2V-CompBench prompts.

A.4. Stability of 1400 Prompts and Videos

We conduct an analysis on the stability of our proposed metrics in evaluating the seven categories. Figures 9, 10, 11, 12, 13, 14, and 15 illustrate how the average score changes as the number of videos increases within these categories. Here, we generate one video for each prompt. The analysis shows that the average score tends to stabilize as the number of videos increases. Some metrics, such as G-Dino for spatial relationships and numeracy, stabilize earlier, at approximately 125 videos, while others, such as Grid-LLaVA for consistent attribute binding, action binding and object interactions, stabilize around 150 videos. The stability of these metrics can also be influenced by different T2V models, as evidenced by the varying levels of fluctuations at the ends of the lines in the figures. However, given that current T2V models typically take minutes to inference a video, it is more practical to limit the number of videos per category to 200, resulting in a total of 1400 prompts and videos.

This ensures that the evaluation does not consume excessive computational time and resources.

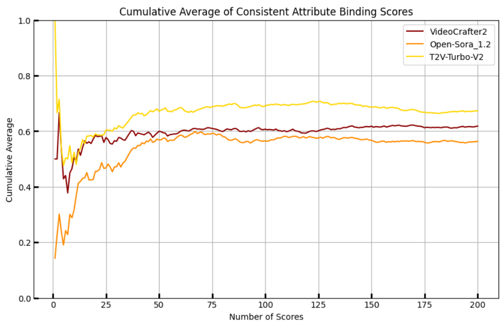


Figure 9. The stability of *Grid-LLaVA* with increasing number of videos in Consistent Attribute Binding.

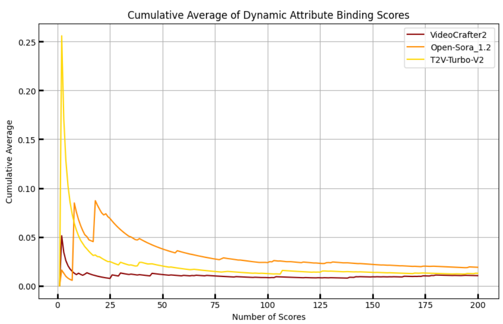


Figure 10. The stability of *D-LLaVA* with increasing number of videos in Dynamic Attribute Binding.

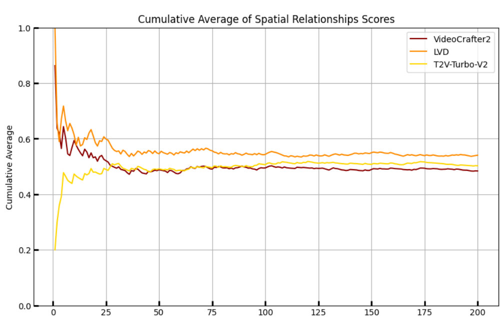


Figure 11. The stability of *G-Dino* with increasing number of videos in Spatial Relationships.

B. Text-to-Video Models

B.1. Details of Evaluated T2V Models

Recent advancements in T2V generation have introduced a variety of innovative models designed to improve video quality, temporal consistency, compositionality, and text alignment. These models leverage novel architectures and training datasets to tackle the diverse challenges of T2V synthesis. While some models [13, 66] adapt text-to-image approaches by incorporating spatio-temporal or motion

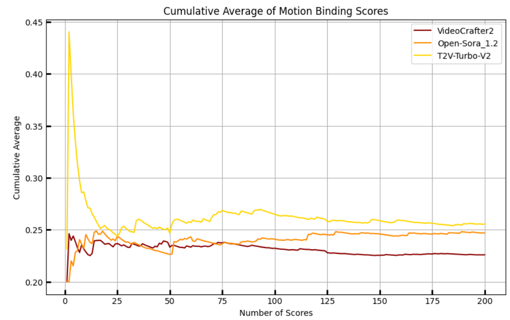


Figure 12. The stability of *DOT* with increasing number of videos in Motion Binding.

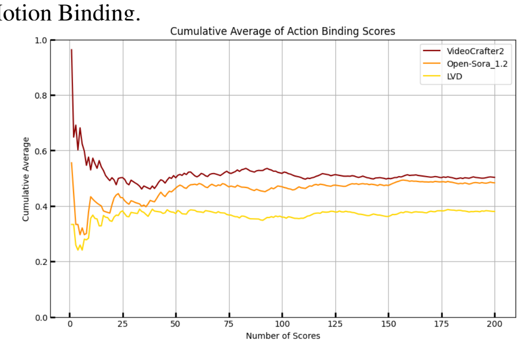


Figure 13. The stability of *Grid-LLaVA* with increasing number of videos in Action Binding.

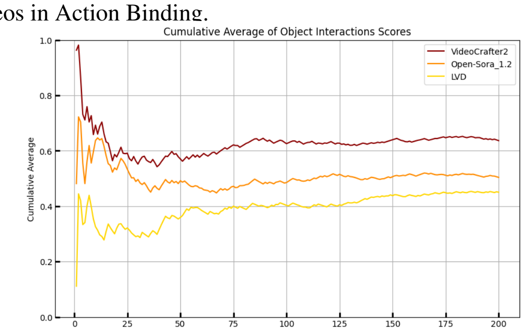


Figure 14. The stability of *Grid-LLaVA* with increasing number of videos in Object Interactions.

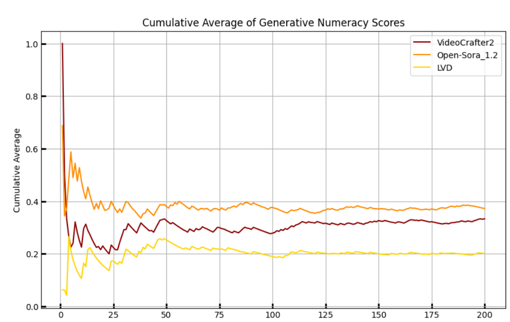


Figure 15. The stability of *G-Dino* with increasing number of videos in Generative Numeracy.

modules, others, such as Show-1 and VideoCrafter2, focus on generating high-resolution, visually appealing videos with precise text alignment. Transformer-based models, including Latte, Open-Sora, and CogVideoX-5B, explore latent space modeling and large-scale architectures for enhanced motion quality and object interactions. Additionally, specialized models like MagicTime target niche tasks, such as metamorphic time-lapse generation, while compositional models like VideoTetris, Vico, and T2V-Turbo-V2 aim to represent complex interactions between multiple objects and actions. This diverse set of models highlights the rapid progress and innovation in the T2V field, catering to various applications and use cases.

- **ModelScope** [66] is a diffusion unet-based T2V generation model that evolves from a T2I model by incorporating spatio-temporal blocks, ensuring consistent frame generation and smooth movement transitions.
- **AnimateDiff** [13] features a plug-and-play motion module that can be applied to diffusion unet-based T2I models to produce temporally smooth animations.
- **Show-1** [83] is a high-quality T2V diffusion model with unet backbone that initially employs pixel-based diffusion to create a low-resolution video, followed by a latent-based approach to upsample the video to high resolution.
- **VideoCrafter2** [6] is a diffusion unet-based video generation model that utilizing low-quality videos and synthesized high-quality images to create a video model with high visual quality and precise text-video alignment.
- **Latte** [44] is a novel Diffusion Transformer (DiT)-based video generation model, which extracts spatio-temporal tokens from input videos and utilizes Transformer blocks to model video distribution in latent space.
- **Open-Sora** [19] and **Open-Sora-Plan** [34] are both DiT-based open-source video generation projects building on Latte [44], aiming at reproducing Sora. **Open-Sora 1.1** features a 700M parameter model trained on 10M data and introduces a refined ST-DiT-2 architecture, supporting multi-resolution training. **Open-Sora 1.2** scales up to 1.1B parameter model trained on over 30M data, with features such as a video compression network and rectified flow training for enhanced video quality and fluency. **Open-Sora-Plan 1.0.0** introduces the CausalVideoVAE for efficient spatial-temporal compression and joint image-video training to enhance visual quality. **Open-Sora-Plan 1.3.0** includes the Wavelet Frequency VAE for more efficient video compression, a Prompt Refiner for better captions of training data, a high-quality data cleaning strategy, and the Skiparse Attention mechanism for faster training of 3D attention models.
- **CogVideoX-5B** [80] is a DiT-based large scale T2V model. It addresses challenges in generating coherent long-duration videos with significant movement by using a 3D VAE to compress videos in spatial and temporal dimensions, and an expert transformer with adaptive LayerNorm to enhance the text-to-video alignment.
- **Mochi** [62] is also a large scale T2V model built on Asymmetric DiT architecture, with emphasis on quality of motion and prompt following.
- **LVD** [33] is specifically designed to leverage LLM-guided layout planning for videos with multiple objects. Here, we use the version adapted from ModelScope [66]. **ZeroScope** [61] is optimized from ModelScope [66] to create high-quality 16:9 compositions and smooth videos.
- **VideoTetris** [63] is proposed to handle compositional T2V generation that involves multiple objects and dynamic changes in object numbers. By employing a Spatio-Temporal Compositional Region Diffusion method, it can seamlessly integrate objects during denoising. It also incorporates a consistency regularization method with Reference Frame Attention to ensure object coherence across scenes, resulting in videos with smooth transitions. **Vico** [79] is a framework that ensures complex compositional interactions between multiple concepts and actions can be represented properly. It extracts attention maps from all layers to create a spatial-temporal attention graph. Then it computes the max-flow from the source text token to the video target token and optimize the noisy latent code to balance these flows. **T2V-Turbo-V2** [30] introduces a consistency distillation process of a T2V model from a pretrained T2V model by integrating various supervision signals. These three model are all adapted or distilled from VideoCrafter2 [6].
- **MagicTime** [81] is trained from AnimateDiff [13]. It is designed to generate metamorphic time-lapse videos, and is trained with time-lapse videos, so we specifically test it on dynamic attribute binding.

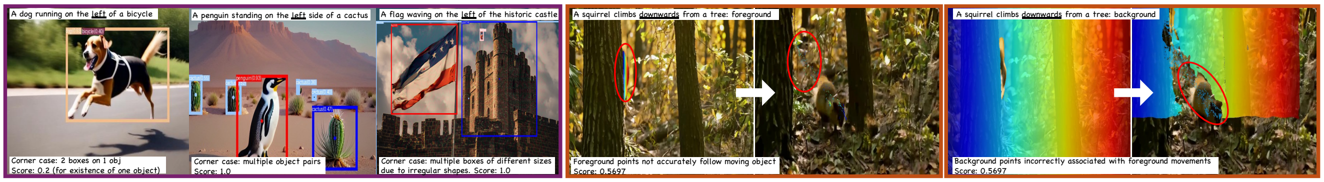
B.2. Implementation Details

We follow the official and default implementations of the T2V models in evaluation. Details of the videos generated by the T2V models, including resolution, total frames, FPS, and duration are presented in Table 3.

C. Evaluation Metrics

C.1. Comparisons across Evaluation Metrics

What are the limitations of existing metrics and how they motivate our proposed metrics. Existing metrics are not specifically designed to evaluate T2V compositionality. Conventional approaches, such as CLIP, ViCLIP, B-CLIP, and B-BLEU, compare the embedding of the original video prompt with that of selected frame(s), video, or generated caption(s), but they do not account for compositionality. We propose using expert tools such as detection, grounding, and tracking to obtain precise object locations, which allows us to evaluate spatial relationships, motion binding, and nu-



(a) Corner cases of Detection tool

(b) Corner cases of Tracking tool

Figure 16. **Corner cases during evaluation.** (a) corner cases encountered with GroundingDino, including object misidentification and duplicate detections; (b) imperfections observed with DOT, including inaccurate tracking of foreground and background points. These examples highlight the importance of robust evaluation strategies.

Model	Resolution	Frames	FPS	Duration (s)
ModelScope [66]	256×256	16	8	2.0
ZeroScope [61]	576×320	36	10	3.6
LVD [33]	256×256	16	8	2.0
AnimateDiff [13]	384×256	16	8	2.0
MagicTime [81]	512×512	16	8	2.0
Show-1 [83]	576×320	29	8	3.6
VideoCrafter2 [6]	512×320	16	8	2.0
VideoTetris [63]	512×320	16	8	2.0
Vico [79]	512×320	16	8	2.0
T2V-Turbo-V2 [30]	512×320	16	8	2.0
Latte [44]	512×512	16	8	2.0
Open-Sora 1.1 [19]	512×512	16	8	2.0
Open-Sora 1.2 [19]	640×360	51	24	2.1
Open-Sora-Plan v1.0.0 [34]	512×512	65	24	2.7
Open-Sora-Plan v1.3.0 [34]	640×352	93	18	5.2
CogVideoX-5B [80]	512×320	40	8	5.0
Mochi [62]	512×320	151	30	5.0
Pika-1.0 [52]	1280×720	72	24	3.0
Gen-2 [57]	1408×768	96	24	4.0
Gen-3 [58]	1280×768	128	24	5.3
Dreamina 1.2 [4]	1280×720	24	8	3.0
PixVerse-V3 [53]	1408×768	161	30	5.4
Kling-1.0 [27]	1280×720	153	30	5.1

Table 3. **Details of generated videos by T2V models**. The table shows resolution, total frames, FPS, and video duration in second for videos generated by T2V models.

meracy. Recent advancements in image LLMs have significantly enhanced their ability to recognize objects and attributes. Our experiments indicate that image LLMs can effectively handle attribute-related categories when provided with specially designed input formats, such as image grids or frame-by-frame sequences, along with techniques like chain-of-thought reasoning. The human correlation presented in Table 1 demonstrates that our proposed metrics are the most reliable for evaluating video compositionality.

Robustness of Our Proposed Metrics. Any evaluation tool cannot be perfect. As illustrated in Figure 16(a), there are corner cases during evaluation when using GroundingDino. For instance, a single object may be misidentified as two different objects. Additionally, multiple object pairs can be detected within one frame due to the generation of more than one object per class or because an irregularly shaped object is detected multiple times. We have implemented specific strategies to address these corner cases. For example, we filter out duplicate bounding boxes that have a high IoU. When multiple object pairs are detected, we select the most probable pair based on both their IoU and confidence scores. Similarly, Figure 16(b) shows corner cases when using DOT. After obtaining the foreground and background masks, the tracking of foreground points may not accurately follow the moving object, and the tracking of background points may be incorrectly associated with the foreground movements. To increase the robustness of DOT evaluation, we average the motion of all points within the masks to calculate the overall background and foreground motions. By implementing strategies to address corner cases, we improve the robustness of our evaluation metrics, allowing for a reliable assessment of compositionality in T2V models. The effectiveness of our metrics is also supported by the human correlation scores in Table 1.

C.2. Using MLLMs as Evaluation Metrics

How to query MLLMs. One challenge of using MLLMs as evaluation metrics is their tendency to generate hallucinations. This can manifest as mistakes in identifying visual content or selection of unmatched grades or scores.

To mitigate these hallucinations, we employ the chain-of-thought mechanism [71]. We firstly ask the MLLM to describe the image without revealing the specific question we intend to ask. This allows the MLLM to independently describe the visual content, without being affected by the subsequent questions. When querying the MLLM for evaluation, it is important to prepare sufficient grading options to differentiate performances of the models. However, too many options make it difficult for the MLLM to identify the subtle differences between them. To this end, we disentangle the evaluation aspects into parallel or sequential queries, asking about each aspect separately. This ensures a sufficient number of grades while preventing the MLLM from being overwhelmed with too many options at once. For consistent attribute binding, we ask whether each of the two objects possesses the correct attribute. For dynamic attribute binding, we ask whether the image depicts the initial or final state. For action binding, we first inquire about the presence of the two objects. Based on that response, we then ask if the actions of the present object(s) align with the

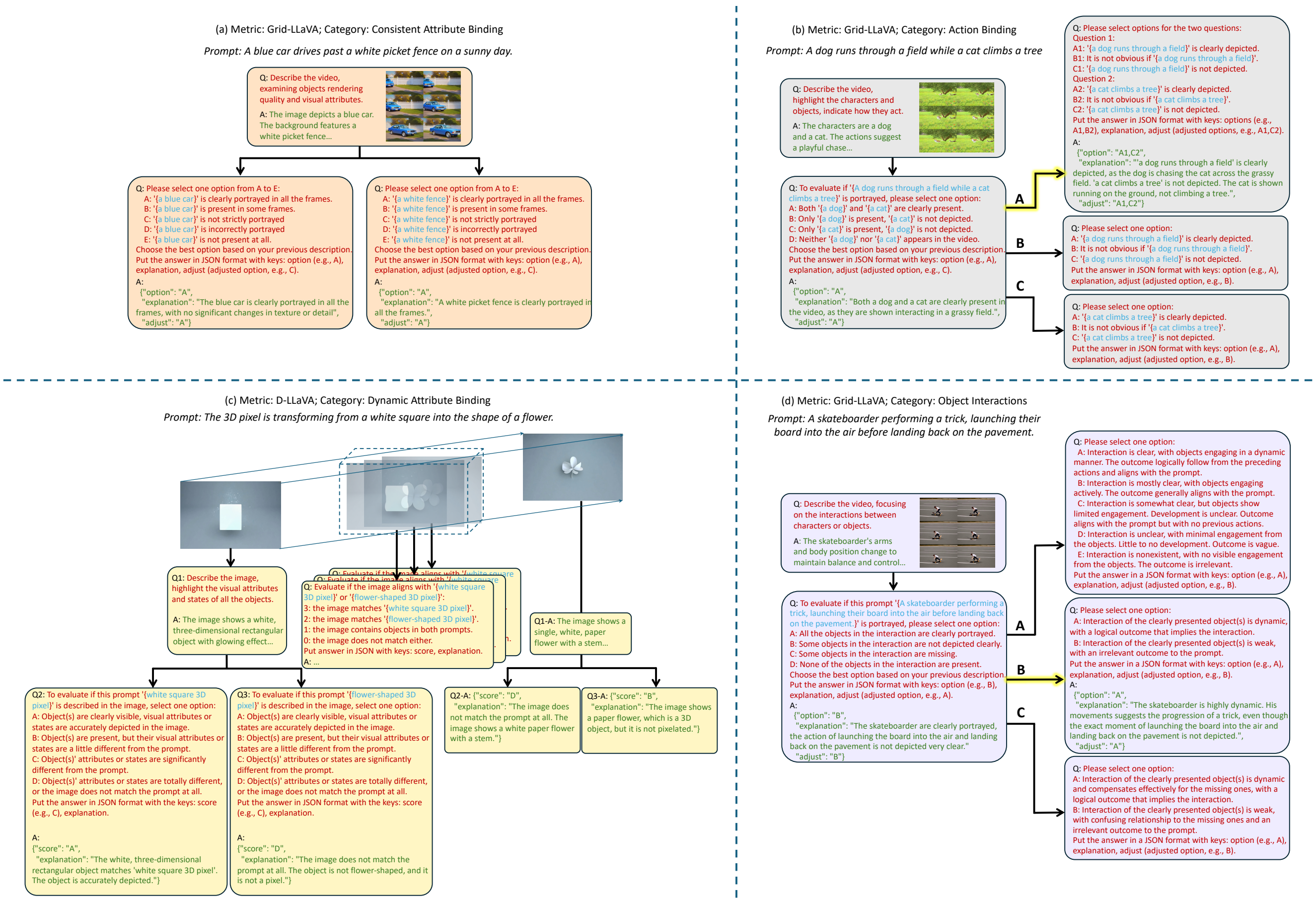


Figure 17. **Question and answer examples for MLLM-based evaluation metrics.** We replace the parts marked in cyan with metadata for different prompts. Answers are generated by LLaVA [36]. Note that the examples are simplified slightly for clarity and brevity.

prompt. For object interactions, we start by confirming the presence of objects, then evaluate the quality of their interaction. Figure 17 shows the question and answer examples for our proposed MLLM-based evaluation metrics.

How to obtain reliable and reproducible results. It is essential for evaluation metrics to produce reliable and reproducible results. To ensure that the results generated by an MLLM are reproducible, we fix its parameters and seed each time it evaluates a video. However, we empirically find that varying the seed during evaluations can change the ranking of the models, especially when their scores are close. To obtain a more reliable score for each video and ranking for the models, an effective approach is to query the MLLM multiple times and average the resulting scores. This method is similar to asking multiple annotators to rate each video, as they may have different opinions. To decide how many times to query the MLLM, we run Grid-LLaVA 8 times with the same settings, varying only the seed. This is done on videos evaluated by humans for the categories of consistent attribute binding, action binding, and object interactions. We randomly sample 2 or 3 experiments from the 8 conducted and calculate the average score for each video. We repeat this process 8 times and obtain 8 correlations with human scores for a sample size of 2 and 8 correlations for a sample size of 3. The results are summarized in Figure 18. As shown in the box plot, increasing the sample size leads to higher mean and median and generally more stable correlations, which in turn indicates that the scores are more stable. Therefore, considering both reliability and practicality, we decide to sample 3 scores for each video, and use their average as the final result.

D. Evaluation Results Analysis

As mentioned in Section 3, we divide each category in T2V-CompBench into several subgroups. In addition, for certain categories, the final score is determined through multiple stages. Here, we extract the scores for some important subgroups, and also extrapolate some meaningful data from the interim stages. The results for these sub-dimensions are documented in Table 4. In this section, we provide comprehensive explanations for these sub-dimensions and analysis for their results, along with visualizations for each category.

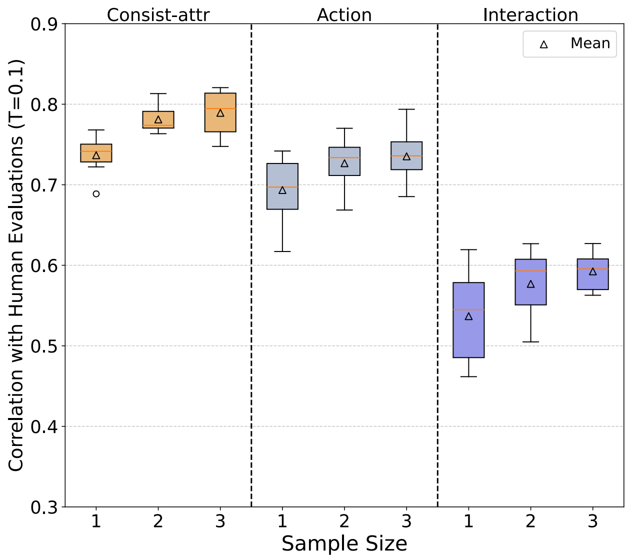


Figure 18. **Box Plot Analysis of Correlation Between Human Evaluations and Grid-LLaVA.** We show the box plot of correlations (Spearman's ρ) between human evaluations and Grid-LLaVA in consistent attribute binding, action binding and object interactions at temperature 0.1 with sample size of 1, 2, and 3.

D.1. Consistent Attribute Binding

The first three columns of Table 4 present the average score for the subgroups of color, shape, and texture within the category of consistent attribute binding. Among the three subgroups, color is the easiest to manage, followed by texture, while shape proves to be the most challenging. In Figure 19, we present a concatenation of frames from videos generated by ModelScope [66] and T2V-Turbo-V2 [30]. Not only does T2V-Turbo-V2 [30] accurately represent the color-object binding in the prompt, but it also demonstrates noticeable object movement.

Prompt: *A blue car drives past a white picket fence on a sunny day.*

(a) T2V Model: ModelScope [66]. Score: 0.5000

(b) T2V Model: T2V-Turbo-V2 [30]. Score: 0.9524

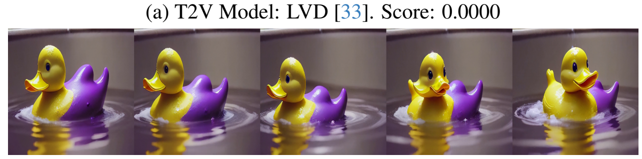
Figure 19. Visualization of Consistent Attribute Binding examples.

D.2. Dynamic Attribute Binding

For the category of dynamic attribute binding, T2V models rarely produce qualified videos. For example, PixVerse-V3 [53] achieves the highest score of 0.0687 in this category. Out of the 200 generated videos, only 31 show relevant elements or transitions, and just 9 of those exhibit meaningful transitions. Models always struggle to generate changing attributes. LVD [33] and T2V-Turbo-V2 [30] exemplify this difficulty in Figure 20. They tend to produce static content that only reflects either the initial or final state of the objects described in the prompt, or both states simultaneously. In this case, PixVerse-V3 [53] demonstrates better prompt understanding and temporal dynamics.

Prompt: *A yellow rubber duck in bath water turns a deep purple.*

(a) T2V Model: LVD [33]. Score: 0.0000



(b) T2V Model: T2V-Turbo-V2 [30]. Score: 0.0000

(c) T2V Model: PixVerse-V3 [53]. Score: 0.8000

Figure 20. Visualization of Dynamic Attribute Binding examples.

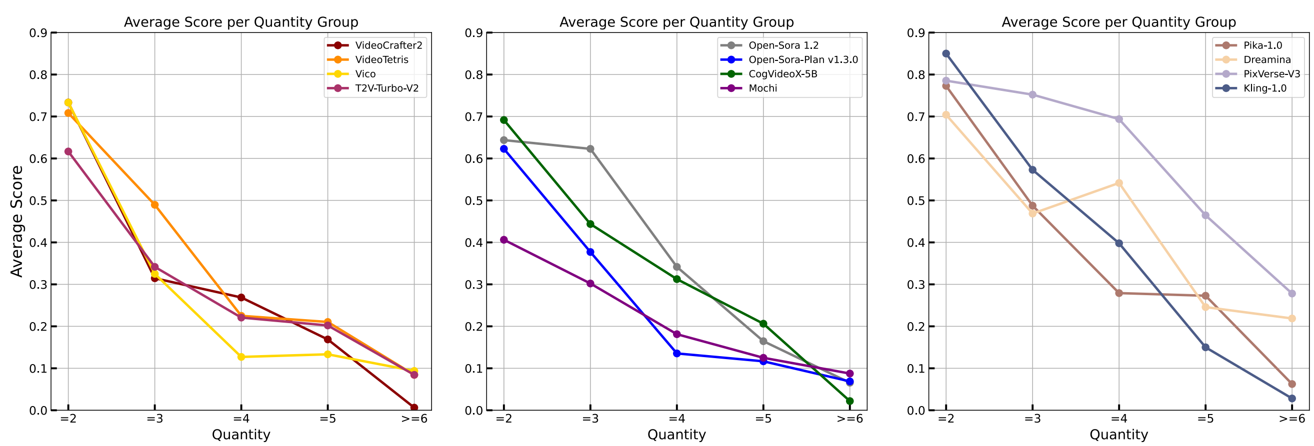
D.3. Spatial Relationships

To investigate whether T2V models can understand 2D spatial relationships, we extract three key indicators from the evaluated spatial scores. As shown in columns #4-6 of Table 4, the first indicator, "Coexist", represents the percentage of videos that successfully generate both objects involved in a specific 2D spatial relationship. A higher "Coexist" percentage indicates the model is better at generating multiple objects simultaneously. The second indicator, "Acc." (accuracy), reflects the percentage of videos that accurately depict the spatial relationships among all videos that successfully generate both objects. A higher "Acc." suggests a greater likelihood that the model comprehends 2D spatial relationships. The third indicator, "Acc.Score", is the average score of those videos with correct spatial relationships. A higher "Acc.Score" indicates that the two objects are more distinctly separated, leading to a clearer spatial layout. For the model to generate accurate spatial relationships, both "Acc." and "Acc.Score" must be high.

According to the results in Table 4, Vico [79] and Mochi [62] demonstrate the best performances in generat-

Model	Consist-attr			Spatial-2D			Motion		Action		Interaction	
Sub-dimension	Color	Shape	Texture	Coexist	Acc.	Acc.Score	Motion Level	Acc.	Common	Uncommon	Physical	Social
diffusion unet-based												
ModelScope [66]	0.5826	0.3440	0.4560	62%	44%	0.8011	16.98	32%	0.4025	0.2093	0.3815	0.5411
ZeroScope [61]	0.4829	0.1905	0.3012	50%	65%	0.7836	21.00	29%	0.4042	0.2139	0.2970	0.5422
LVD [33]	0.6155	0.4139	0.4458	63%	**90%**	**0.9054**	13.73	**47%**	0.4120	0.2528	0.3752	0.5252
AnimateDiff [13]	0.4619	0.3113	0.4196	50%	52%	0.8180	12.79	21%	0.3053	0.2009	0.3604	0.4337
Show-1 [83]	0.5964	0.4774	0.5601	62%	56%	0.8387	11.74	26%	0.4331	0.2083	0.5659	0.6830
VideoCrafter2 [6]	0.6717	0.4792	0.5780	73%	59%	0.7914	8.96	24%	0.5472	0.3259	0.5607	0.7122
VideoTetris [63]	0.6957	0.4667	0.5589	74%	53%	0.7831	9.94	21%	0.5410	0.3056	0.6200	0.6956
Vico [79]	0.6705	0.4476	0.4619	80%	50%	0.8083	9.12	20%	0.5609	0.3120	0.5263	0.6652
T2V-Turbo-V2 [30]	0.7781	0.5351	0.5452	77%	54%	0.7967	19.14	37%	0.6694	0.3657	0.5907	0.6970
DiT-based												
Latte [44]	0.4657	0.3613	0.5280	61%	51%	0.7856	12.76	14%	0.4623	0.2241	0.3904	0.4389
Open-Sora 1.1 [19]	0.5669	0.4220	0.5304	77%	69%	0.8323	10.87	25%	0.5613	0.2731	0.5070	0.6059
Open-Sora 1.2 [19]	0.6038	0.4321	0.5607	76%	50%	0.8905	15.98	30%	0.5069	0.3889	0.4396	0.5681
Open-Sora-Plan v1.0.0 [34]	0.4357	0.3887	0.4107	66%	58%	0.7320	7.54	17%	0.4583	0.1713	0.4067	0.4233
Open-Sora-Plan v1.3.0 [34]	0.7107	0.4244	0.5429	76%	59%	0.8726	15.89	22%	0.4938	0.2870	0.4393	0.4574
CogVideoX-5B [80]	0.7202	0.4333	0.5167	71%	61%	0.8706	23.33	34%	0.5824	0.3370	0.5607	0.6530
Mochi [62]	0.7043	0.4393	0.4774	78%	75%	0.8045	18.55	34%	0.5336	0.2454	0.4930	0.5833
commercial												
Pika-1.0 [52]	0.6095	0.4315	0.5119	72%	55%	0.7841	8.13	17%	0.4650	0.2648	0.4348	0.6048
Gen-2 [57]	0.6721	0.3994	0.5131	83%	53%	0.8162	7.73	20%	0.4894	0.2491	0.5681	0.6607
Gen-3 [58]	0.6652	0.5077	0.5280	68%	67%	0.8728	22.59	42%	0.5639	0.3611	0.5663	0.6148
Dreamina 1.2 [4]	0.7798	0.5601	0.5738	**90%**	62%	0.8777	12.71	23%	0.6225	0.4722	0.6411	0.7237
PixVerse-V3 [53]	0.7898	**0.5768**	**0.6381**	89%	71%	0.8231	**28.09**	38%	**0.9016**	**0.7546**	**0.8219**	**0.8400**
Kling-1.0 [27]	**0.7971**	0.4673	0.6149	**90%**	57%	0.8634	17.27	35%	0.5863	0.5481	0.6530	0.7726

Table 4. **T2V-CompBench sub-dimension evaluation results using proposed metrics**. Scores are normalized between 0 and 1. A higher score indicates better performance. **Bold** signifies the highest score within each category. Blue highlights the top score among diffusion unet-based models. Yellow highlights the top score among DiT-based models. Red highlights the top score among commercial models.



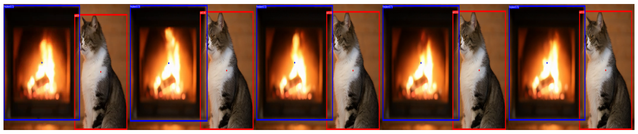
(a) diffusion unet-based models (b) DiT-based models (c) commercial models

Figure 21. **Analysis of text-to-video generation accuracy by object quantity.** We plot the average score versus the quantity groups for videos generated by prompts with single object class in Generative Numeracy. This plot draws four T2V models from each of the following groups: (a) diffusion unet-based models, (b) DiT-based models, and (c) commercial models.

ing multiple objects among open-source models. Although LVD [33] does not achieve a high "Coexist" percentage, it ranks highest in "Acc." and "Acc.Score" among all models, which verifies its strong capability in layout planning.

Figure 22 showcases a pair of videos generated by Mochi [62]. In this example, the model accurately depicts the spatial relationship between the cat and the fireplace. Notably, the model also achieves a high score in "Acc.". This suggests that the model may possess some understanding of 2D spatial relationships.

(a) Prompt: *A cat sitting on the* ***left*** *of a fireplace.* Score: 0.8394



(b) Prompt: *A cat sitting on the* ***right*** *of a fireplace.* Score: 0.9706

Figure 22. Visualization of Spatial Relationships examples generated by Mochi [62].

D.4. Motion Binding

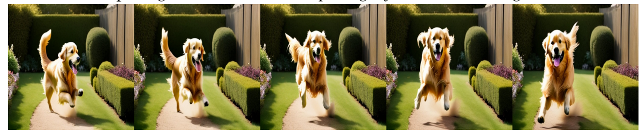
As described in Section 4, motion vectors are obtained when evaluating motion binding. Therefore, we use these vectors to derive two meaningful indicators, "Motion Level" and "Acc." for videos generated by prompts with a single object in motion binding. They are recorded in columns #7-8 in Table 4. We first normalize the image to a size of 100x100. "Motion Level" represents the displacement the object travels. Although this indicator does not account for the direction of movement, it reflects the overall motion level and the spatio-temporal dynamics of the video. "Acc." in this category represents the percentage of videos in which the object moves at least 5 units in the correct direction, among all videos that successfully generate the object. LVD [33] is particularly effective in planning the dynamic scene layouts, allowing it to determine the correct motion directions for objects. This is validated by its "Acc." of 47%, the highest percentage among all models. Another notable value is 42% "Acc." achieved by Gen-3 [58], which suggests it may have some understanding of motion direction. Additionally, we can see from "Motion Level" that PixVerse-V3 [53] and CogVideoX-5B [80] can generate significant object motion in their videos.

Figure 23 compares two examples from VideoCrafter2 [6] and Gen-3 [58]. The first features a static background with limited foreground motion, while the second displays noticeable background movement.

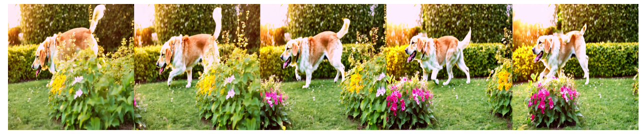
D.5. Action Binding

Columns #9 and #10 in Table 4 show the average scores for the subgroups of common and uncommon prompts in the category of action binding. Some of the uncommon prompts instruct animals to perform anthropomorphic actions, which are shown in Figure 24. This subgroup is clearly more challenging than the common prompts because executing these uncommon actions is more difficult.

Prompt: *A golden retriever scampering leftwards across a garden.*



(a) T2V Model: VideoCrafter2 [6]. Score: 0.2688



(b) T2V Model: Gen-3 [58]. Score: 0.4381

Figure 23. Visualization of Motion Binding examples.

Prompt: *A dog plays guitar while a cat takes a selfie.*

(a) T2V model: VideoCrafter2 [6]. Score: 0.7037

(b) T2V model: PixVerse-V3 [53]. Score: 0.9630

Figure 24. Visualization of Action Binding examples.

D.6. Object Interactions

The last two columns in Table 4 record the average scores for the subgroups of physical and social interactions within the category of object interactions. In these two subgroups, depicting physical interactions proves to be more challenging, as it requires an understanding of physical laws. Figure 25 illustrates two examples of physical interactions generated by Open-Sora-Plan v1.3.0 [34] and VideoTetris [63]. The first example fails to accurately represent the interaction process described in the prompt, while the second effectively captures both the progression and outcome of the interaction, following the prompt more accurately.

Prompt: *A skateboarder performing a trick, launching their board into the air before landing back on the pavement.*

(a) T2V model: Open-Sora-Plan v1.3.0 [34]. Score: 0.5185

(b) T2V model: CogVideoX-5B [80]. Score: 1.000

Figure 25. Visualization of Object Interactions examples.

D.7. Generative Numeracy

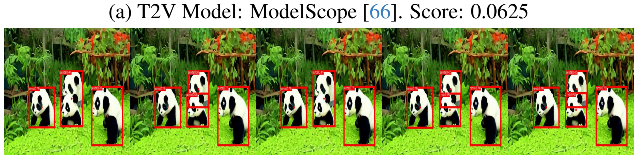
Figure 21 illustrates how the score in generative numeracy changes in relation to the quantity of object specified in the prompt. It is evident that, as the quantity increases, the average score tends to decrease.

Among all models, commercial models generally outperform open-source models. In particular, PixVerse-V3 [53] achieves the highest scores across almost all quantity groups. The diffusion unet-based models in Figure 21(a) demonstrate comparable results in terms of numeracy. Among DiT-based models in Figure 21(b), Open-Sora 1.2 [19] stands out with the best overall performance.

Figure 26 presents videos generated by ModelScope [66] and Open-Sora 1.2 [19]. Although the video generated by Open-Sora 1.2 [19] is not of realistic style, it successfully represents the correct quantity of object.

Prompt: *Four pandas munch bamboo in the garden.*

(a) T2V Model: ModelScope [66]. Score: 0.0625



(b) T2V Model: Open-Sora 1.2 [19]. Score: 0.8750

Figure 26. Visualization of Generative Numeracy examples.

E. Human Evaluation

To assess the correlation of the scores given by T2V-CompBench metrics with human preferences, we prepare a human evaluation interface for each of the seven categories. The human evaluations are conducted on Amazon Mechanical Turk (AMT). Figure 27 shows the interface for consistent attribute binding.

Annotation Instruction. Each interface includes several key components. First, we clarify the evaluation dimension on which this category is focusing. At the top of each interface, we thoroughly explain the key information in the prompt and the specific dimension of interest. For example, in consistent attribute binding, annotators are instructed to focus solely on the objects and attributes. Next, we provide clear rating criteria along with examples in this category. For each prompt-video example, we provide a detailed rationale for the assigned score based on the specific objects and attributes depicted in the video. Annotators may refer to these examples at any time when they are uncertain about the appropriate rating. Finally, we present the video-text pair that needs evaluation, along with the scoring options (ranging from 5 to 1). To reinforce the rating criteria for the annotators, we include a concise summary of the criteria following each score.

Strategies for Ensuring Quality. To ensure the quality of human evaluations, we employ several strategies:

- Interface Notification: We include a note in the interface informing annotators that we will review the evaluation results and reject tasks from workers who obviously do not follow the instructions.
- Random Sampling: We randomly sample 20% of the total tasks completed by each worker. Any task that evidently fails to follow the instructions will be rejected. If an annotator is rejected more than five times, we will block that individual from further participation, and the tasks completed by that annotator will be reassigned for evaluation.
- Multiple Raters: To balance different opinions, we require three human annotators to rate each task. The average score from these ratings is then assigned to the corresponding video-text pair.
- Selection of Annotators: We only select experienced and responsible AMT workers by establishing a high historical task acceptance threshold of 90%.
- Compensation: The estimated hourly wage for each participant is 9.60 USD. We have communicated with several annotators, and they agree that this wage is sufficiently competitive to encourage reliable task ratings. In total, we spent 195 USD on participant compensation.

These measures collectively enhance the reliability and accuracy of our human evaluations.

F. Societal Impacts

The community should be aware of the potential negative social impact that can arise from the misuse of video generation models, particularly in generating misleading or harmful content, which could exacerbate issues related to misinformation and deepfakes. Additionally, the biases inherent in the training data may lead to the perpetuation of stereotypes or exclusion of underrepresented groups, thereby influencing public perception and societal norms. Therefore, it is imperative that the evaluation of T2V models not only assesses their technical performances but also considers broader social implications, ensuring that these technologies contribute positively to society while mitigating potential risks. To this end, we plan to incorporate an evaluation dimension focused on unbiased composition in the future.

G. Limitations and Future Work

While we have made the first step in evaluating compositional T2V generation, there are still many challenges:

- A limitation of our work is the lack of a unified evaluation metric for all categories, and we believe that this limita-

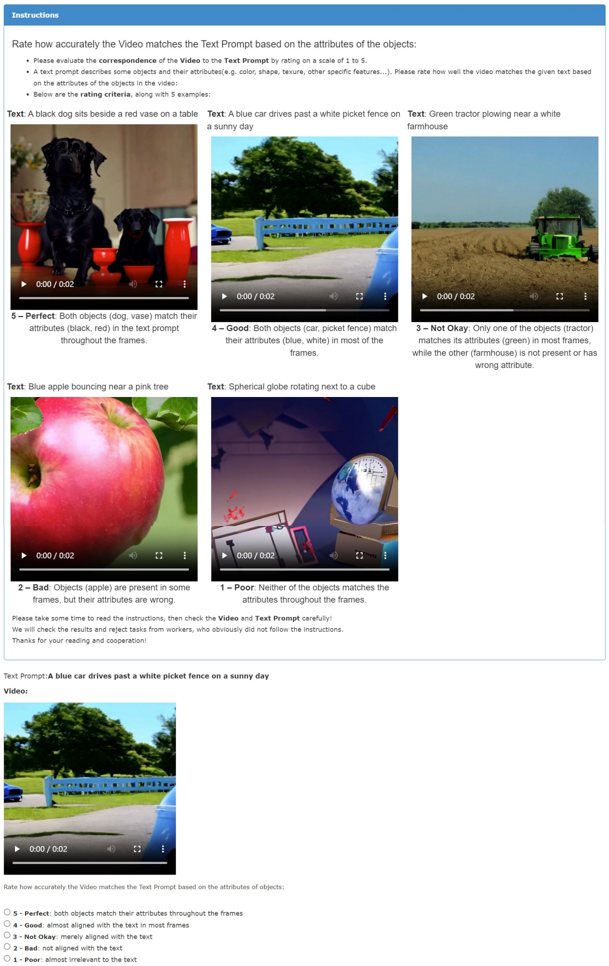


Figure 27. AMT Interface for human evaluation of video-text alignment on consistent attribute binding.

tion points out new challenges for better and larger multimodal LLMs or video understanding models. If such models become available, we can utilize them as our evaluation metrics.

- Our benchmark aims to evaluate videos within 2 to 5 seconds. For categories other than motion binding, we sample a fixed number of frames for evaluation, which may not be sufficient for videos longer than 5 seconds. In motion binding, longer videos may lead to greater object displacement and better performance. We leave the evaluation of long videos for future work.

H. Templates for Generating Prompts and Metadata Using GPT-4

This section provides the instructions used to prompt GPT-4 [48] to generate the text prompts for T2V-CompBench and the corresponding metadata for evaluation. The text prompts in T2V-CompBench and metadata generated for evaluation are available in the code repository.

Table 5, 6, 7, 8, 9, 10, 11 are the input templates used to generate the prompts for the seven categories, each with specific requirements.

Table 12, 13, 14, 15, 16, 17 are the templates used to generate the metadata for evaluation.

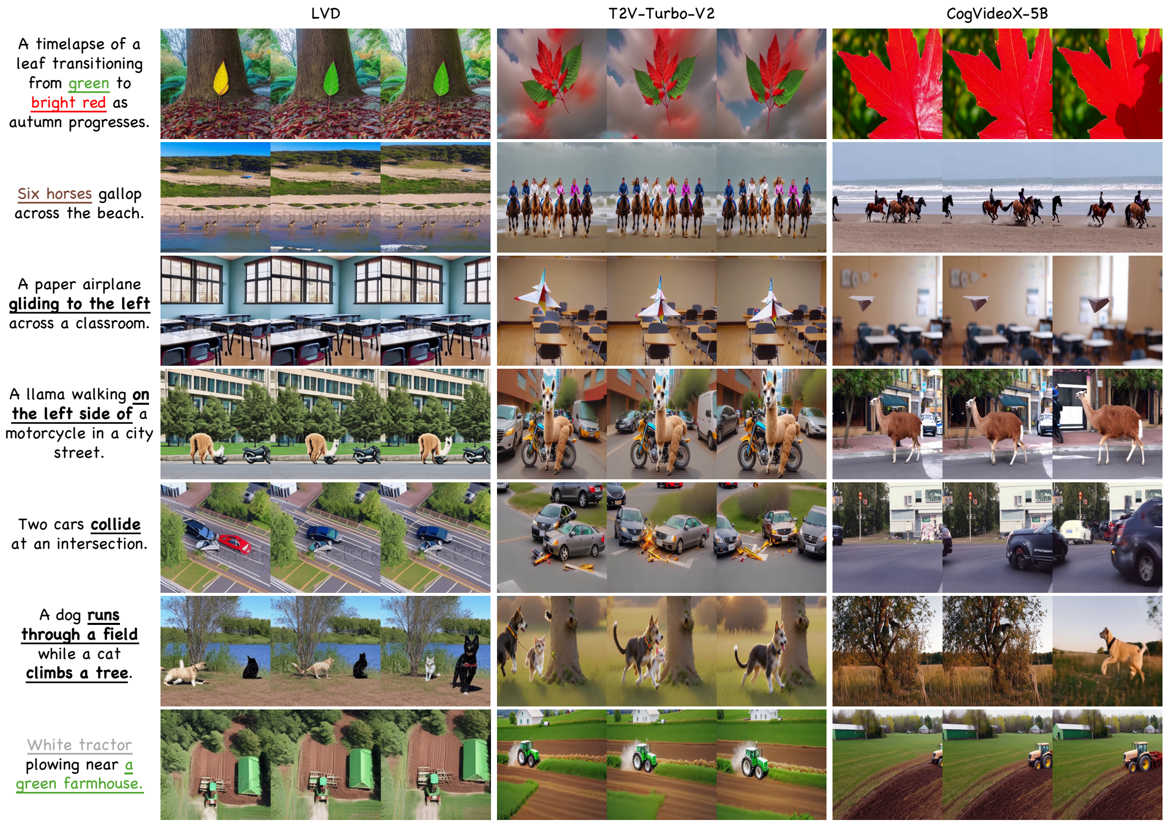


Figure 28. **Qualitative comparison with different open-source T2V models for the seven compositional categories in T2V-CompBench.** We show text-to-video examples generated by LVD [33], T2V-Turbo-V2 [30], and CogVideoX-5B [80].

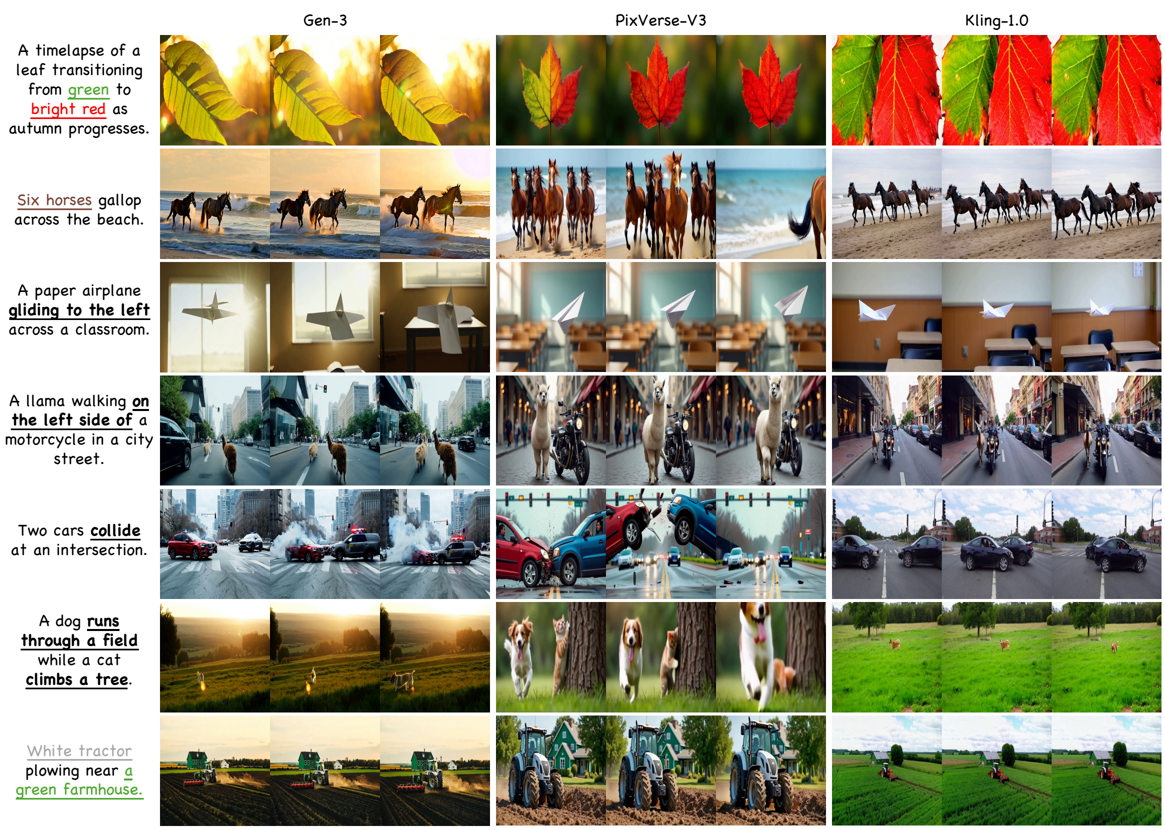


Figure 29. **Qualitative comparison with different commercial T2V models for the seven compositional categories in T2V-CompBench.** We show text-to-video examples generated by Gen-3 [58], PixVerse-V3 [53], and Kling-1.0 [27].

<USER>: This is a list of object nouns: [object nouns]. Please reply 'Yes' if received.
<ASSISTANT>: Yes
<USER>: This is a list of color attributes: [color attributes]. Please reply 'Yes' if received.
<ASSISTANT>: Yes
<USER>: This is a list of active verbs: [verbs]. Please reply 'Yes' if received.
<ASSISTANT>: Yes
<USER>: Please generate 50 prompts featuring two objects with color attributes.
Requirements:
1. Specify a unique color for each object in the prompt to enrich the visual narrative. Use the color attributes in the color attributes list provided above.
2. Select object nouns that have multiple possible colors from the list of object nouns provided above.
3. The objects can engage in separate or shared activities. Describe them using vibrant, active verbs, avoiding static verbs such as 'see', 'rest', or 'park'. Select verbs from the list of active verbs and feel free to adjust their tense.
Here are some examples:
A blue car drives past a white picket fence on a sunny day.
Yellow rubber duck floating next to a blue bath towel.

Table 5. **Template for generating consistent attribute binding prompts using GPT-4.** The attribute type in the template can be replaced by shape or texture; here we take color as an example. The object nouns, color attributes, and verbs marked in cyan should be replaced with words derived from real user prompts.

<USER>: Here are some prompts that describe color & light change, please generation 30 prompts by referring these examples:
The leaf is turning from green to yellow.
The sky is shifting from bright to dark.

Table 6. **Template for generating dynamic attribute binding prompts using GPT-4.** The attribute type in the template can be replaced by shape & size change, texture change or combined change; here we take color & light change as an example. The examples are sourced from TempCompass [42].

<USER>: This is a list of object nouns: [object nouns]. Please reply 'Yes' if received.
<ASSISTANT>: Yes
<USER>: This is a list of active verbs: [verbs]. Please reply 'Yes' if received.
<ASSISTANT>: Yes
<USER>: Please generate 50 pairs of prompts featuring two objects where, in each pair, one object is described as being on the left of the other in the first prompt, and the second prompt should state that the same object is on the right of the other.
Requirements:
1. Select object nouns from the list provided above.
2. The prompt must contain active verbs to ensure the scene is dynamic. You can select verbs from the list of active verbs and feel free to adjust their tense.
Here are some examples:
'A dog running on the left of a bicycle' and 'A dog running on the right of a bicycle'.
'A llama walking on the left side of a motorcycle in a city street' and 'A llama walking on the right side of a motorcycle in a city street'.

Table 7. **Templates for generating spatial relationships prompts using GPT-4.** The spatial relationship in the template can be replaced by above&below or in front of&behind; here we take left&right as an example. The object nouns and verbs marked in cyan should be replaced with words derived from real user prompts.

<USER>: This is a list of object nouns: [object nouns]. Please reply 'Yes' if received.
<ASSISTANT>: Yes
<USER>: Please generate 50 pairs of prompts where, in each pair, an object is being describe as moving leftwards in the first prompt and the second prompt should state the same object is moving rightwards.
Requirements:
1. Select object nouns from the list provided above.
2. Describe the objects using vibrant, active verbs.
Here are some examples:
'A golden retriever scampering leftwards across a garden' and 'A golden retriever scampering rightwards across a garden'.
'A football rolling from the right to the left on the grass' and 'A football rolling from the left to the right on the grass'.

Table 8. **Templates for generating motion binding prompts using GPT-4.** The moving direction in the template can be replaced by upwards&downwards; here we take leftward&rightwards as an example. The object nouns marked in cyan should be replaced with words derived from real user prompts.

<USER>: This is a list of object nouns: [object nouns]. Please reply 'Yes' if received.
<ASSISTANT>: Yes
<USER>: This is a list of active verbs: [verbs]. Please reply 'Yes' if received.
<ASSISTANT>: Yes
<USER>: Please generate 50 prompts that describe an object engaging in an activity, while another object is engaging in a different activity.
Requirements:
1. Select the two object nouns from the list of object nouns provided above.
2. Specify a unique verb for each object in the prompt.
3. Use vibrant, active verbs, avoiding static verbs such as 'see', 'rest', or 'park'. You can select verbs from the list of active verbs and feel free to adjust their tense.
Here are some examples:
A dog runs through a field while a cat climbs a tree.
A man takes photos and a boy dances on the street.

Table 9. **Templates for generating action binding prompts using GPT-4.** The object nouns, marked in cyan should be replaced with words derived from real user prompts.

<USER>: This is a list of object nouns: [object nouns]. Please reply 'Yes' if received.
<ASSISTANT>: Yes
<USER>: Please generate 50 prompts describing physical interaction between two objects that can change their state of motion.
Requirements:
1. For the two objects involved in the interaction, select at least one object noun from the list of object nouns provided above.
2. Use vibrant, active verbs, avoiding static verbs such as 'see', 'rest', or 'park'.
Here are some examples:
Two cars collide at an intersection.
A dog dragging a blanket off a bed, leaving it tangled on the floor.

Table 10. **Templates for generating object interactions prompts using GPT-4.** The physical interaction in the template can be replaced by social interaction; here we take physical interaction as an example. The object nouns marked in cyan should be replaced with words derived from real user prompts.

<USER>: This is a list of object nouns: [object nouns]. Please reply 'Yes' if received.
<ASSISTANT>: Yes
<USER>: This is a list of active verbs: [verbs]. Please reply 'Yes' if received.
<ASSISTANT>: Yes
<USER>: Please generate one prompt for each object in the list [object nouns] with a quantity of two. Use vibrant and active verbs in the prompt. You can select verbs from the list provided above and feel free to adjust their tense.
Here are some examples:
Two cats walking on the floor.
Two dogs running through a field.

Table 11. **Templates for generating numeracy prompts using GPT-4.** The quantity of 'two' in the template can be replaced by 'three', 'four', 'five', 'six', 'seven' and 'eight'; here we take 'two' as an example. The object nouns marked in cyan should be replaced with words derived from real user prompts, here we restrict the length of the list to 30.

<USER>: I will provide you with several prompts, each on a separate line. Each prompt will describe two objects, each associated with an attribute. For each prompt, you are to create a JSON object. This JSON object should include two keys: "prompt" and "phrases".
The "prompt" key should simply replicate the original prompt text. The "phrases" key should contain the two described objects separated by a semicolon.
For instance, for the prompt "A blue car drives past a white picket fence on a sunny day", the value under "phrases" should be "a blue car; a white picket fence".
Below are the prompts for which you need to generate the JSON objects:
A blue car drives past a white picket fence on a sunny day.
Yellow rubber duck floating next to a blue bath towel.
......

<ASSISTANT>: Here are the JSON representations for the prompts:
{"prompt": "A blue car drives past a white picket fence on a sunny day", "phrases": "a blue car; a white picket fence"},
{"prompt": "Yellow rubber duck floating next to a blue bath towel", "phrases": "yellow rubber duck; a blue bath towel"},
......

Table 12. **Template for generating the metadata of consistent attribute binding prompts using GPT-4.**

<USER>: I will provide you with several prompts, each on a separate line. Each prompt describes a color, shape, texture or combined change of one or multiple objects. For each prompt, you are to create a JSON object. This JSON object should include the following keys:
"prompt": simply replicate the original prompt text.
"state 0": the object(s) with initial states.
"state 1": the object(s) with final states.
For instance, for the prompt "The sky is shifting from bright to dark", "state 0" should be "bright sky", "state 1" should be "dark sky". For the prompt "a leaf is transitioning from green to yellow and a fresh apple is decaying", "state 0": "a green leaf and a fresh apple"; "state 1": "a yellow leaf a decayed apple".
Below are the prompts for which you need to generate the JSON objects:
The sky is shifting from bright to dark.
The leaf is turning from green to yellow.
......

<ASSISTANT>: Here are the JSON representations for the prompts:
{"prompt": "The sky is shifting from bright to dark", "state 0": "bright sky", "state 1": "dark sky"},
{"prompt": "The leaf is turning from green to yellow", "state 0": "green leaf", "state 1": "yellow leaf"},
......

Table 13. **Template for generating the metadata of dynamic attribute binding prompts using GPT-4.**

<USER>: I will provide you with several prompts, each on a separate line. For each prompt, you are to create a JSON object. This JSON object should include the following keys:
"prompt": simply replicate the original prompt text.
"spatial": contain the word from the prompt that represents the spatial relationship. Choose one from the following six options: left, right, above, below, in front of, and behind. For example, if the prompt uses the word "under", set the value of "spatial" to "below" because these words have the same meaning.
"object_1" and "object_2": These keys represent the two objects mentioned in the prompt. The value of "object_1" should be positioned to the left of, right of, above, below, in front of, or behind "object_2" based on the spatial word used in the prompt.
Here's an example to illustrate the format:
Prompt: "A toddler walking on the left of a dog in a park"
JSON object: {"prompt": "A toddler walking on the left of a dog in a park", "spatial": "left", "object_1": "toddler", "object_2": "dog"}
Below are the prompts for which you need to generate the JSON objects:
A bird flying on the left side of a hot air balloon.
A llama walking on the left side of a motorcycle in a city street.
......

<ASSISTANT>: Here are the JSON objects for each of your prompts, accurately capturing the spatial relationship, object_1, and object_2:
{"prompt": "A bird flying on the left side of a hot air balloon.", "spatial": "left", "object_1": "bird", "object_2": "hot air balloon"},
{"prompt": "A llama walking on the left side of a motorcycle in a city street.", "spatial": "left", "object_1": "llama", "object_2": "motorcycle"},
......

Table 14. **Template for generating the metadata of spatial relationships prompts using GPT-4.**

<USER>: I will provide you with several prompts, each on a separate line. Each prompt describes the movement of one or two objects in different directions. For each prompt, you are to create a JSON object. This JSON object should include the following keys:
"prompt": simply replicate the original prompt text.
"object_1": the first object mentioned in the prompt that moves in a direction.
"d_1": motion direction for "object_1" and can only be one of the following: left, right, up, or down.
"object_2": If there is a second object mentioned in the prompt that moves in a direction, this key will contain its name. Otherwise, the value will be an empty string.
"d_2": If there is "object_2", this key will represent its motion direction. Otherwise, the value will be an empty string.
Below are the prompts for which you need to generate the JSON objects:
A golden retriever scampering leftwards across a garden
A ball rolls left while a puppy dashes right
......

<ASSISTANT>: Here are the JSON representations for each prompt:
{"prompt": "A golden retriever scampering leftwards across a garden", "object_1": "golden retriever", "d_1": "left", "object_2": "", "d_2": ""}
{"prompt": "A ball rolls left while a puppy dashes right", "object_1": "ball", "d_1": "left", "object_2": "puppy", "d_2": "right"},
......

Table 15. **Template for generating the metadata of motion binding prompts using GPT-4.**

<USER>: I will provide you with several prompts, each on a separate line. For each prompt, your task is to extract the prompt into a noun phrase and a noun phrase with an action.
For example, given the prompt "A dog runs through a field while a cat climbs a tree", the output should be: {"prompt": "A dog runs through a field while a cat climbs a tree", "phrase_0": ["a dog?", "a dog runs through a field?"], "phrase_1": ["a cat?", "a cat climbs a tree?"]}.
Below are the prompts for which you need to generate the JSON objects:
A dog runs through a field while a cat climbs a tree.
A man takes photos and a boy dances on the street.
......

<ASSISTANT>: Here are the extracted phrases for each prompt:
{"prompt": "A dog runs through a field while a cat climbs a tree", "phrase_0": ["a dog?", "a dog runs through a field?"], "phrase_1": ["a cat?", "a cat climbs a tree?"]},
{"prompt": "A man takes photos and a boy dances on the street", "phrase_0": ["a man?", "a man takes photos?"], "phrase_1": ["a boy?", "a boy dances on the street?"]},
......

Table 16. **Template for generating the metadata of action binding prompts using GPT-4.**

<USER>: I will provide you with several prompts, each on a separate line. Each prompt describes a certain number of objects engaging in an activity, where each object is associated with a specific quantity. For example, the prompt may include phrases like “two cat” or “three dogs”.
For each prompt, you are to create a JSON object. This JSON object should include the following keys:
“prompt”: simply replicate the original prompt text.
“objects”: This key will contain the objects mentioned in the prompt. If there are multiple objects, they will be separated by commas.
“numbers”: This key will represent the corresponding numbers associated with each object. The numbers will be concatenated with commas.
When describing the action the objects are performing and the environment, there may be mentions of other objects, articles like “a road”, “the woods” or other unnecessary details. Please ignore these and focus only on extracting the main objects and their respective quantities.
Below are the prompts for which you need to generate the JSON objects:
Three cars speed along a road.
Two pandas munch on bamboo, and three elephants spray water.
......

<ASSISTANT>: Here are the JSON representations for each prompt:
{“prompt”: “Three cars speed along a road”, “objects”: “car”, “numbers”: “3”},
{“prompt”: “Two pandas munch on bamboo, and three elephants spray water”, “objects”: “panda,elephants”, “numbers”: “2,3”},
......

Table 17. **Template for generating the metadata of generative numeracy prompts using GPT-4.**